

- はじめに
 - Igorインストールについて (from Igorインストール虎の巻, modified by Miyazaki)
 - 関数の種類
 - 関数作成時のルール
 - License
- MEMによる自発ラマン相当のスペクトル取得
 - データの読み込みからMEMのやり方
 - 特定の状況下での技法
 - MEMした後に, 特定の波数で画像を作りたい
 - nonresonant backgroundを測定データから作りたい
- 領域平均スペクトルの作成
 - 撮影した画像内の特定の領域の平均スペクトルを得る
 - 表示している画像内の画素値が一定以上の空間点の平均スペクトルを求める
- フィットと画像の作成
 - 領域平均スペクトルのガウスフィット
 - z stack時のfitと画像表示について
 - Image Chunkの作り方
 - ピーク位置フィット
- 特定波数と特定波数の比のイメージ解析
 - フィッティング後の画像でのratiometric imageの作成
- Igor便利技
 - グラフスタイルマクロ
 - 複数のプロットに別の色を付ける
 - 複数プロットの平均値とsd (or se, 95% CI)をプロットする
 - 箱ひげ図, バイオリンプロットで一つ一つの系列を塗分ける
 - Igor redimensionの挙動について
 - 解析虎の巻よりフィット時の初期値まとめ
 - OH region
 - aromatic CH
 - 1755と1650
 - CH region
 - 1450
 - 1400-1200のフィット
- Image plot
 - Image plotで縦軸, 横軸を設定する方法
- グラフギャラリー
- Igor proで線形回帰
- GUI based analysis (author: Kyosuke Tanaka)

はじめに

Igorインストールについて (from Igorインストール虎の巻, modified by Miyazaki)

1. マイドキュメント→wavemetrics→Igor Pro8 User Filesという名でフォルダ作成(この中にigor8インストール)
2. google ドライブ→研究室共用ソフト→igor8をダウンロード
3. 全て展開→setupIgor8開いて進んでいきinstallまで進む(上のフォルダに保存)
4. MEMを実行するには以下も必要
5. google drive上のMEM最新→64bit対応コード.zip→ダウンロード
6. 以下に従って、マイドキュメントのIgor Pro8 User Filesの該当するところに入れていく

Igor Extension(64-bit)→そのままコピペ

User Procedure→そのままコピペ

IgorProcedure →CARS_GenerateTileImage, CARS_GenerateTileImagePMT, mem_approximate, MM procedures, Other_fitのみ移動

関数の種類

1. MEM related functions

MEM (maximum entropy method) 関連の関数です.

2. Average functions

MEM後のデータで、領域平均スペクトルを得るための関数です.

3. Fitting functions

MEM後のスペクトルをガウスフィットして、画像化したりするための関数です.

4. Ratiometric_analysis

強度比を用いて解析を行う際に使用する関数です.

5. IgorTips

Igorの便利技等を書いていきます.

6. ImagePlot

Image plot用の関数を置いてあります.

7. OtherFunctions

その他関数、古い関数などです.

8. GUI

Imchi3_dataの解析用GUIです。マニュアルはpdf参照

9. BaselineSubtraction Baseline引き算用の関数です

関数作成時のルール

1. 関数名

関数名は機能を表した名前を付け、一般的な名前は避ける。

(良くない例)

Wave2Dto4D_MS (MSは私の名前だが、書いてあっても何もわかりやさくない)

(良い例)

WaveTransformer2Dto4D

作成者や作成日時は関数内のコメントに記載し、可能な限り関数名には入れない (Miyazakiなどと入れても関数名はわかりやすくならないため)

2. コメント 関数を定義したら下に、///で初めるコメントを記載する。

記載内容としては、何をする関数であるのか、誰が記載した(更新した)のか、いつ更新したのか、パラメータは何か、注意点など

コメントは基本は英語、日本語も可能とする。(記載のハードルを下げる、英語で書きにくいくらいなら日本語でもいいからとにかく書いておく)

3. 記載方法、命名規則 命名規則はCamelCase(先頭を大文字、WaveNameSelectみたいな感じ)

コメントアウトしたコードは極力消す。

定数はなるべく関数の最初に定義を行う。

マジックナンバー(42-wavenum のようにみただけではわからない具体的な数値)には必ずコメントをつける

License

The source code is licensed MIT. The website content is licensed MIT license.

MEMによる自発ラマン相当のスペクトル取得

データの読み込みからMEMのやり方

0. 用意するもの

Igor8

測定データ (.spe)

バックグラウンドデータ (.spe)

ノンレゾナントバックグラウンド (.spe)

DataLoad_Preprocessing.ipf

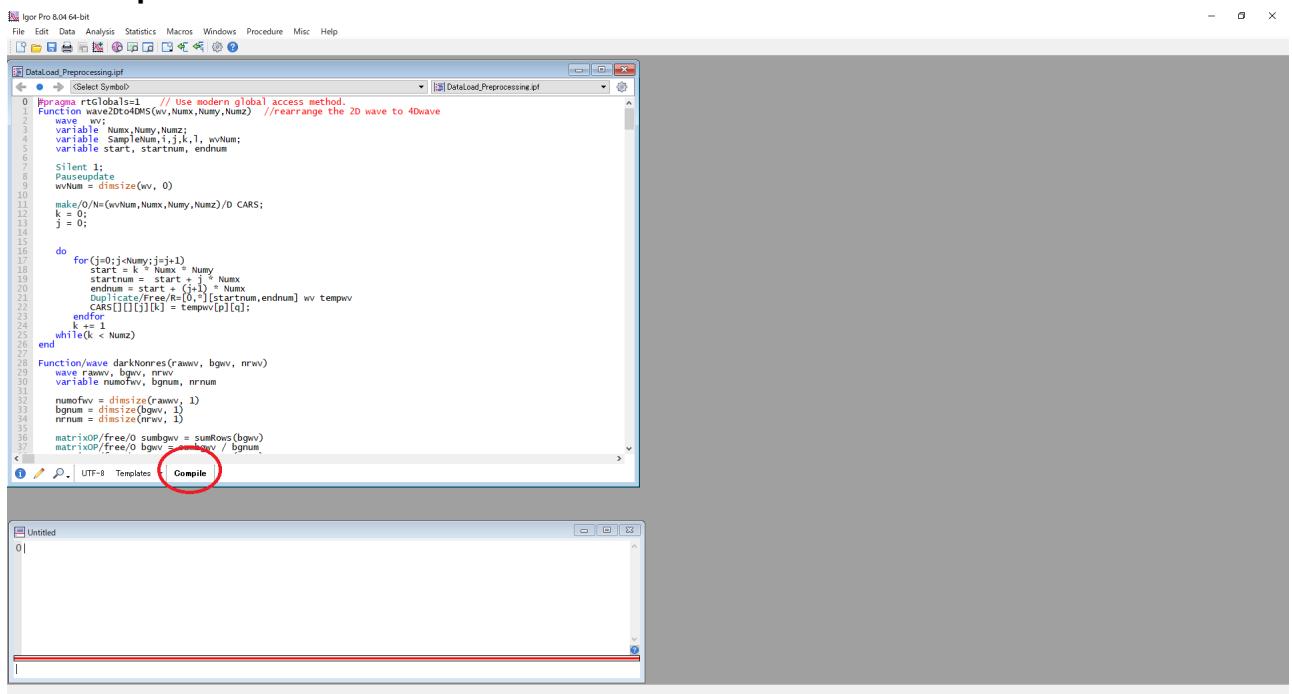
x軸 (.itx)

1. Igor8の起動

2. DataLoad_Preprocessing.ipfをドラッグアンドドロップしてcompile (下のほうに小さくcompileがあるのでクリック)

もしくはあらかじめDocuments/WaveMetrics/Igor Pro 8 User Files/Igor Proceduresの中に入れておくと、毎回compileしないで済みます。

ほかのscriptと干渉するとエラーが出るので注意



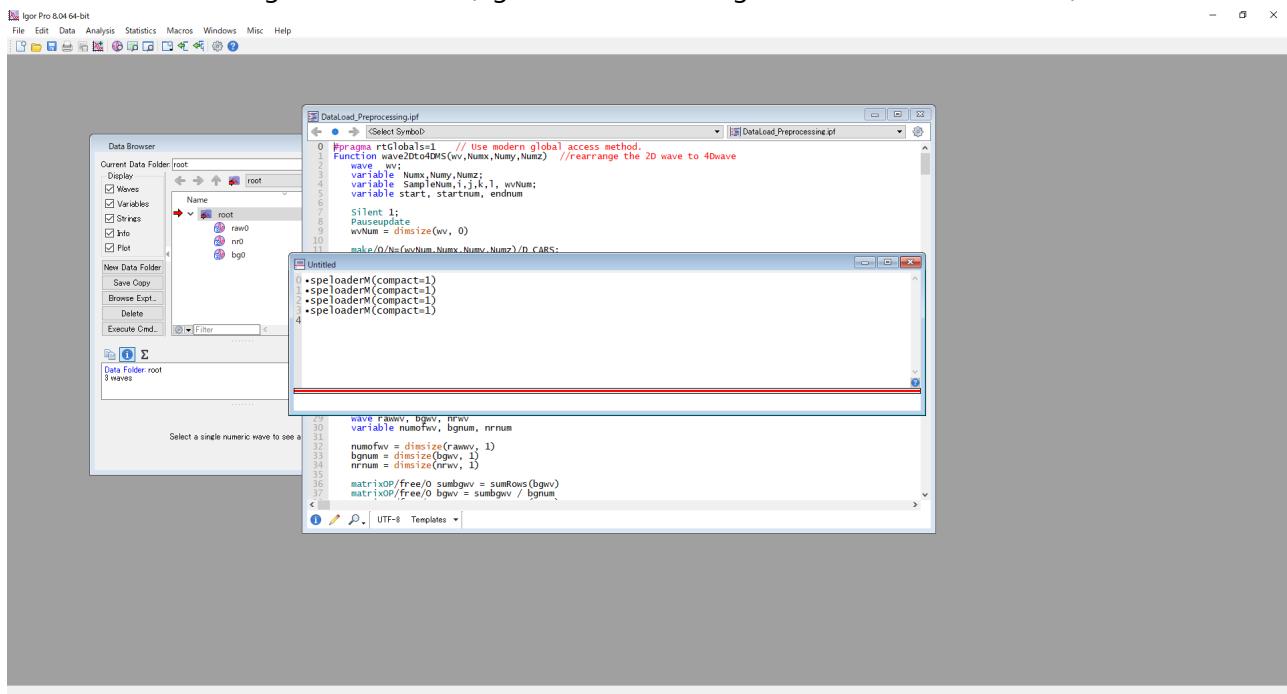
3. コマンドラインを呼び出して(ctrl+j), 以下のコード実行

```
SpeloaderM(compact=1)
// Igorでは大文字小文字の区別をしていないので、全部小文字でも動きます
```

上記でファイル選択画面が開くので、解析したい測定データを選択する

4. 読み込みの際に名前を付けられるので、適当に名前を付ける (rawなどとすると, "raw0"という名前になる。データブラウザ(ctrl+b)で確認可能)

5. 3を2回繰り返して、bg, nrを読みこむ (bg, nrと名付けるとbg0, nr0みたいな名前になる)



6. x軸を読み込むため,.itxデータをドラッグアンドドロップ

7. x軸(は勝手に"xwavelength"という名前で読み込まれるので、これを以下のコードで名前を変える。

```
rename xwavelength ramanshift
```

これ以降のコードで、x軸が“ramanshift”という名前でないと動かないコードもあるので、この手順は必ず行うこと

8. データの前処理のため、以下のコード実行

```
darknonres(raw0, bg0, nr0)
    //ここは、読み込みの際にraw, bg, nrという名前にしなかったなら、
    //自分でつけた名前で実行すること
    //コードの中身をみてもらえればわかるかと思いますが、
    //rawとnrからbgを引いて、bgを引いたrawをnrで割っています
```

9. 2次元データから4次元データへの変換

測定データ (.spe) は2次元(波数,xyz)です。これを、4次元(波数, x, y, z)に直します。

例) 波数1340点, x41点, y41点, z5点測定した場合, speloaderで読み込んだデータの形は(1340, 8405)となっています。これを(1340, 41, 41, 5)に変更します。以下のコードです

(20210930追記)

九大加納研でz方向にzigzag撮影のデータがあるということで、対応しました。

Wave2Dto4DMSの引数が一つ増えています。DataTypeという引数に1を入れると、XZ撮影した際のz方向をzigzagから通常の並び順に戻します。

xy撮影した際のz ziqzaqは、そこまで需要ないかと思い、対応していません（どうせxy1枚ずつ処理することが多

いと思ったため)

Datatypeの記述を省略すれば今まで通り使えるはずです.

z方向にzigzagしていない場合 or xyz撮影の場合

```
wave2Dto4DMS(raw0,41,41,5)
```

```
//最新の関数を使っていただいている場合には問題ありませんが,  
//昔私が作った関数では、 z方向にwaveがコピーされてしまう間違いました。  
//念のためデータブラウザでの確認をおすすめします  
//最後の引数を省略すると今まで通り動くと思います
```

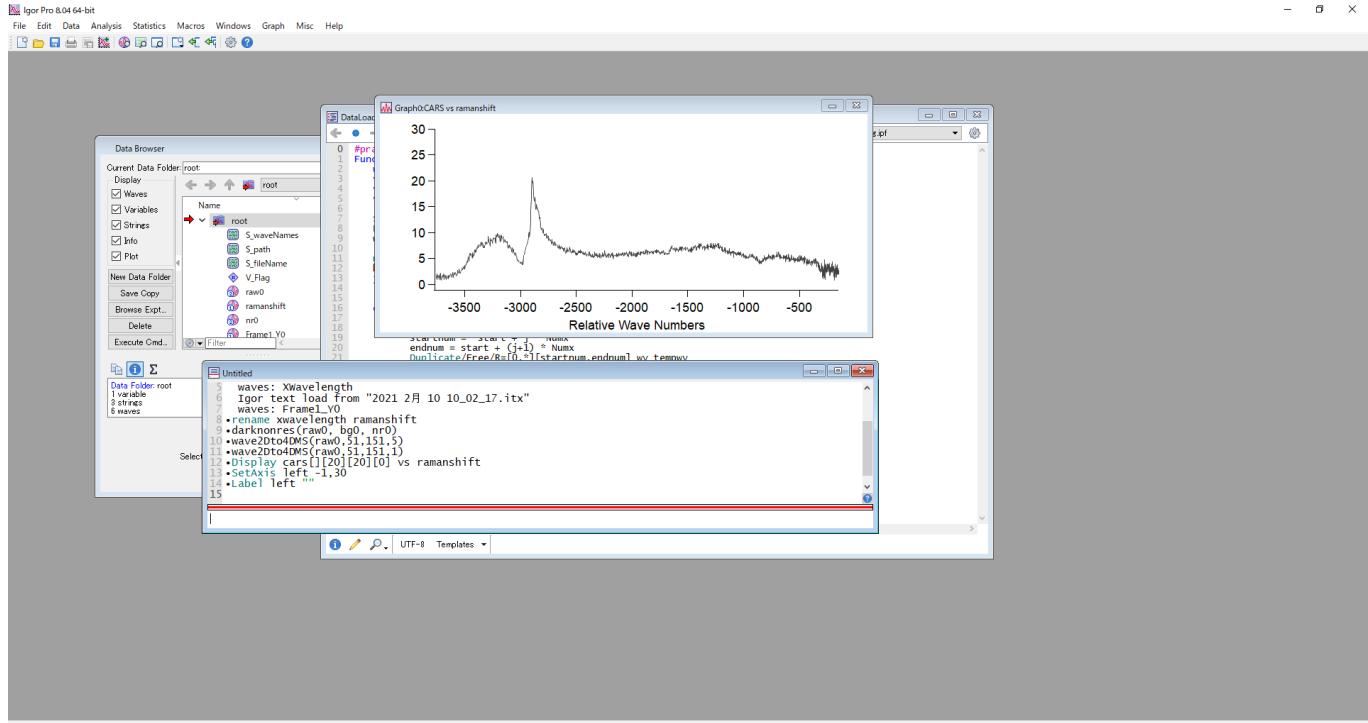
z方向にzigzagしていて、 xzy撮影の場合

```
wave2Dto4DMS(raw0,41,41,1,1)
```

10. データの確認 上記コードを実行すると、データがCARSという名前に変換されています。試しにデータの形を見てみます

Display cars[][20][20][0] vs ramanshift

```
//CARS[][20][20][0] はcARSという4次元waveのうち,  
//1次元目(波数)はすべてのデータ点,  
//空間点を示す2次元目以降はx=20, y=20, z=0の点を参照しています。
```



11. MEMをかける波数範囲の指定

以下のコードで波数の範囲を指定します

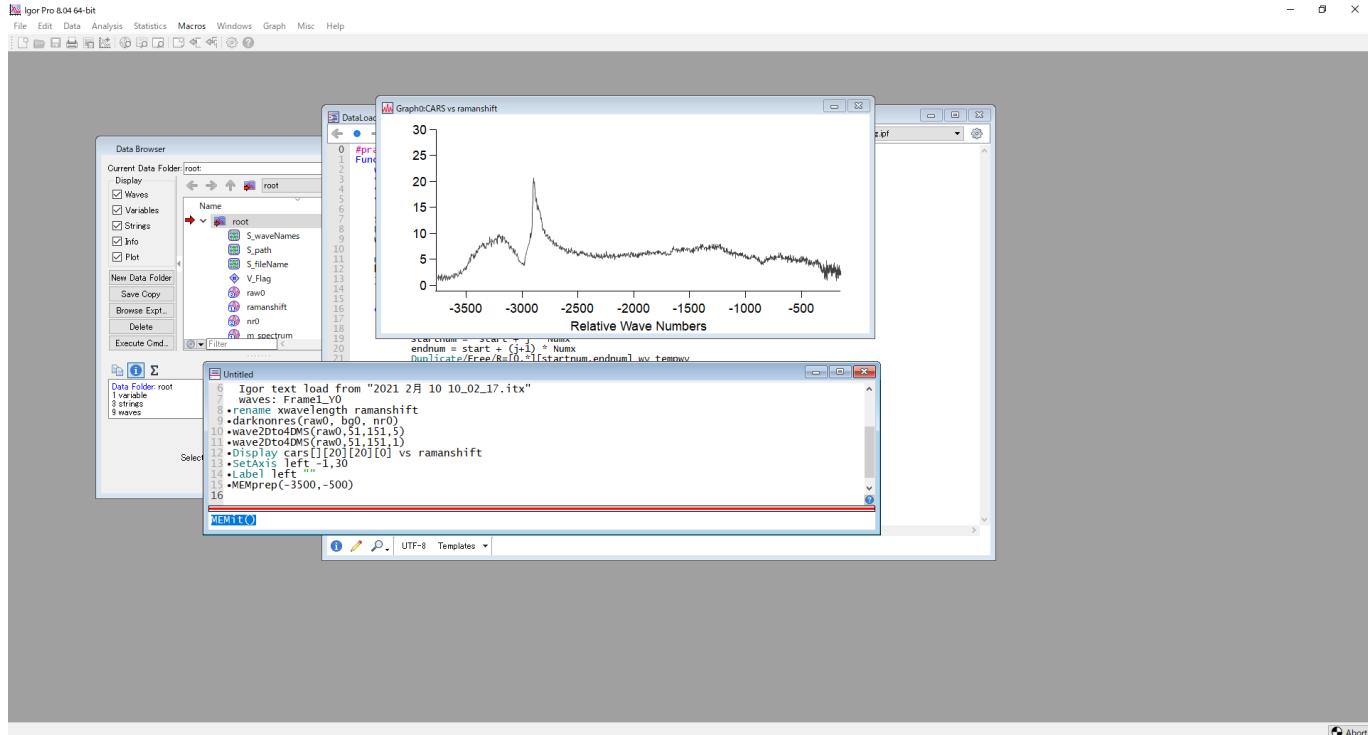
```
MEMprep(-3500, -500)
```

```
//ここは好みで、 解析したい範囲を入れてください。
```

//ただし、MEMでは原理的に、波数の端はデータの質が担保できないので、
//500まで解析したいときは450か400まで入れたほうが良いです。

12. MEMの実行

```
MEMit()
    //もしくは、 mem_time.ipfを読み込んでからmem_time()で実行すると実行時間をお出してくれます。
    //数分以上かかります
```



13. 横軸の取得

```
makeramanshift4(m_ramanshift1)
    //MEMが終わるとm_ramanshift1という名前の軸ができているので、ここは上記をそのままコピペで大丈夫です。
```

14. データセーブ

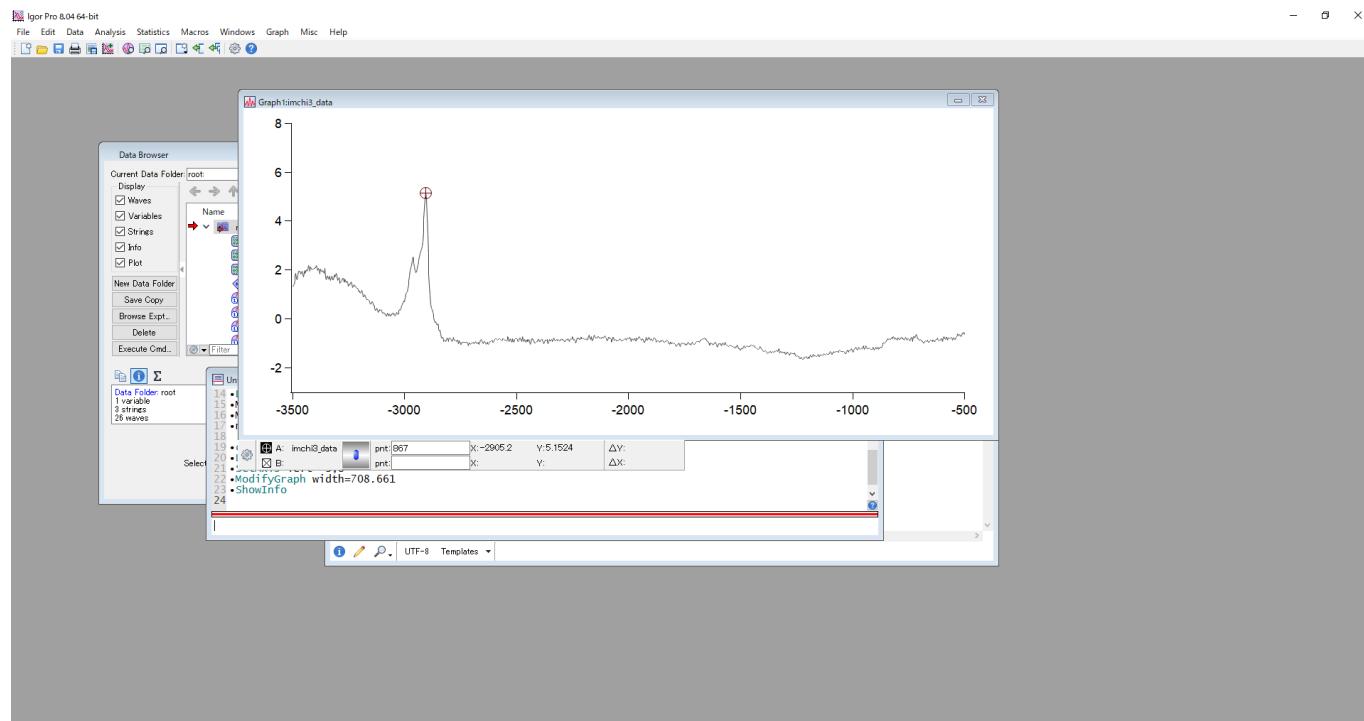
```
save/C re_ramanshift2  
save/C imchi3_data  
    //すべて終わっていれば、軸の名前がre_ramanshift2,  
    //データの名前がimchi3_dataになっているはずです。  
    //これらを希望の位置に保存しましょう
```

特定の状況下での技法

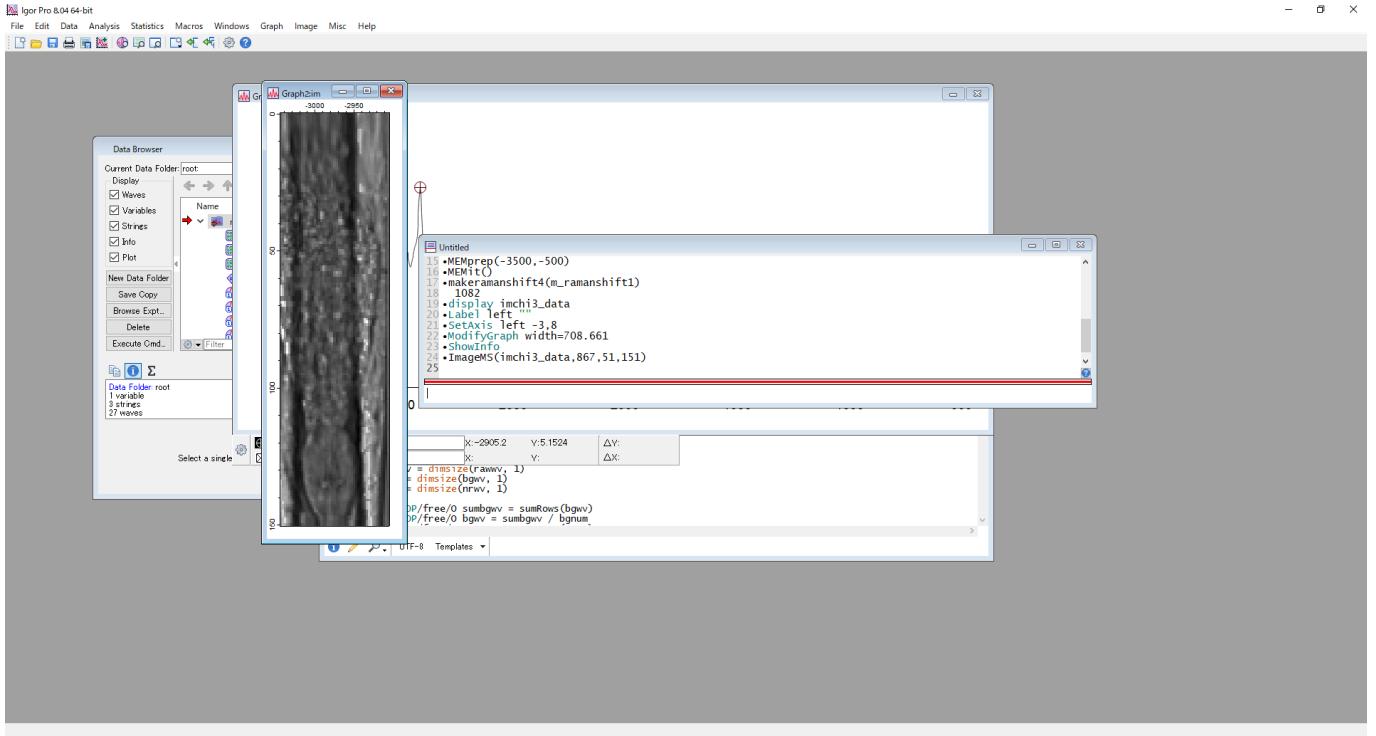
MEMした後に、特定の波数で画像を作りたい

MEMした後、特定の波数(2850など)でとりあえず画像を見たい場合には、まずは2850の位置を調べましょう。

```
display imchi3_data
//これで、横軸ramanshiftではなく、点数のグラフが出来ます。
//見たいピークにカーソル(ctrl+i)を合わせたら、ピークの波数位置がでます
//これを記憶しておきます
```



```
ImageMS(imchi3_data,230,41,41)
//2つ目の引数、230は可視化したいピークの波数位置です。
```



nonresonant backgroundを測定データから作りたい

nonresonant backgroundを、カバーガラスではなく、細胞の横の培地からとりたい時などに僕はあまり、必要としたことがありませんが、培地からnonres取るには以下です
最初に適当な波数でイメージ作成して、どこからnonresを作るかを決めましょう
適当にImageMSでも、loadからイメージ出しても大丈夫です。カーソル (ctrl+i) で位置決めして、x, y座標を書き留めておきましょう。

```

pickup_nr(xsize, xnum1, xnum2, ynum1, ynum2, oriwv)
//xsizeは画像のx幅です。41点41点の画像なら41
//xnum1~yNum2はnonresをとりたい領域のxy座標です。順番に注意
//oriwvは測定データです。.speをspeloaderで読み込めばそのデータでOK

```

領域平均スペクトルの作成

撮影した画像内の特定の領域の平均スペクトルを得る

撮影した画像から特定の領域(細胞やその中の特定の構造)の平均スペクトルを得る場合です。

0. 用意するもの

- imchi3_data
- re_ramanshift2
- rawdata (MEM前のデータ)
- region_analysis.ipf

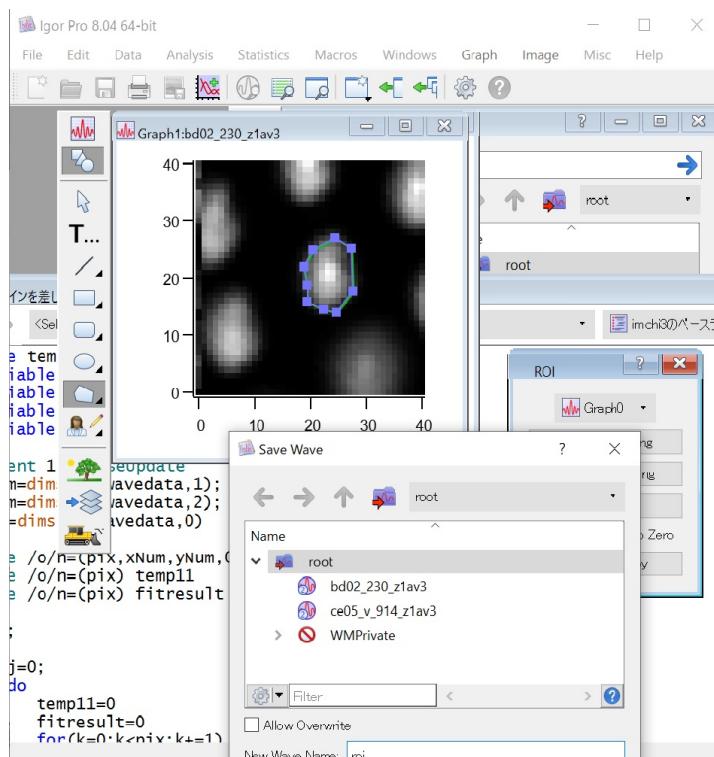
1. Data/Load Data/Load ZigZag SPE File Noshifted and Make Each Z swap AV Quickly LF PIXIS NO zigzag を使って画像を出す。
(これらはextensionを入れていないとIgorで表示されません。extensionの導入方法は別に示します。)

2. Image/Image ROI からROIマネージャーを起動

3. ROIというウィンドウがポップアップがあるので, start ROI Editingをクリック

4. 画像の左側にツールバーが出現するので, 好きなツールを選択してROIを囲む

5. roiなどの適当な名前で保存



6. 以下のコードで領域平均スペクトルを求める

```
region_analysis(imchi3_data, roiwave, znum)
//ここでroiwaveは先ほど作ったもの
//znumはimchi3_dataがz方向に複数枚の時には,
//好きなz平面の解析ができます。1枚目はznum = 0
```

```
//average_wvという名前で保存されるので、好きな名前に変更  
rename average_wv region_av
```

表示している画像内の画素値が一定以上の空間点の平均スペクトルを求める

例)

SHが一定値以上の強度の空間点の平均スペクトルを求めたいとき
すでにフィットしたが、ある波数で強度の強いピクセルを足しこみたいとき

0. 用意するもの

imchi3_data
re_ramanshift2
rawdata (MEM前のデータ)
averaging_function

1. 画像の表示

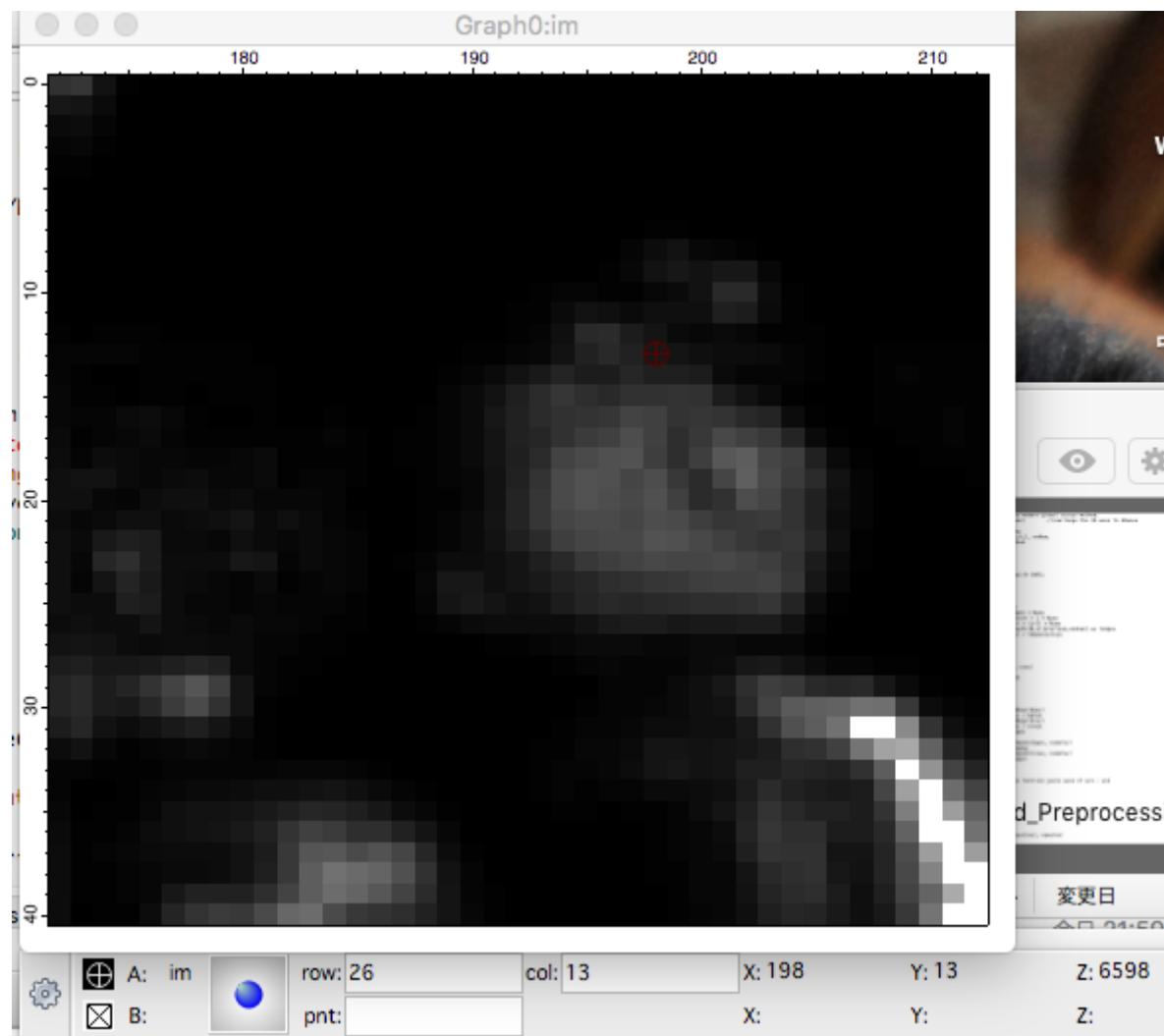
適当な方法で画像を出します。

ImageMSでも良いですし、Data/Load Data/Load ZigZag SPE File Noshifted and Make Each Z swap AV Quickly LF PIXIS NO zigzagでもOKです。

Fit後の画像で実行したいときはFitImageが出ていればOKです。

2. 閾値を決めるために、画像内の適当な位置の輝度値の確認

足し込みする閾値を決めるために、画像内の適当な点の輝度値を取得します



上の図のように、画像を出してカーソル (ctrl + i)で見ても良いですし、Windows/New tableから、画像を数値データとして表示して閾値を見るのも良いです。

3. 閾値を決めて、それ以上の領域の足し込み

閾値が決まったら以下のコードで足し込みをおこないます。

```
AveragingWithImageAndDiscrI(ImageWv, Threshold, OriginalWv)
//imageWvにはimageの名前をいれてください。
//thresholdには2. で決めた閾値を入れます。
//originalwvにはimchi3_dataなどを入れます
rename temp00 discrIwv
//temp00という名前でwaveが作成されるので、適当に名前をつけてください.
```

フィットと画像の作成

最終更新 2022/1/23

領域平均スペクトルのガウスフィット

1. 用意するもの

領域平均スペクトル (.ibw)

imchi3_data

re_ramanshift2

FittingFunctions.ipf

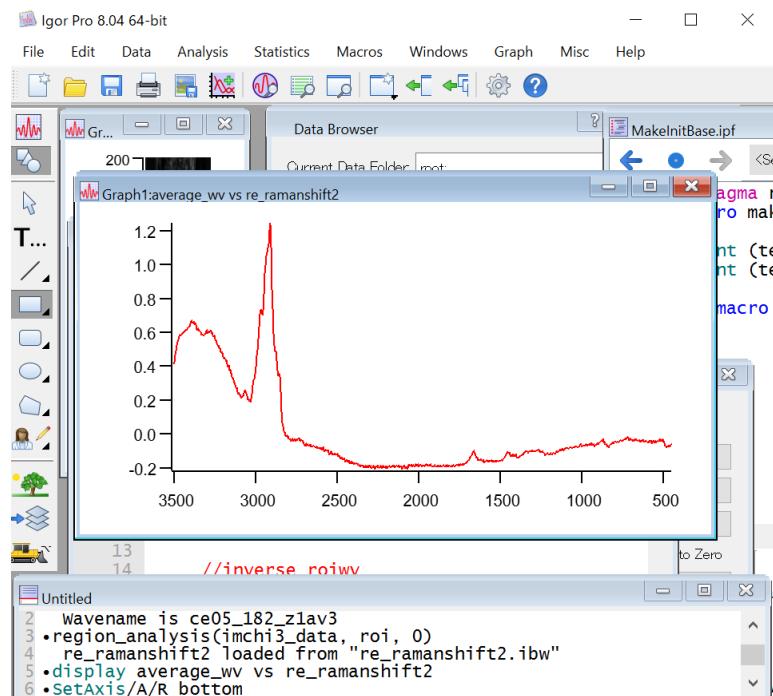
2. フィット領域の定義のため、グラフを表示

まずは、領域平均スペクトルを表示して、どの波数をフィットするかを考えます。

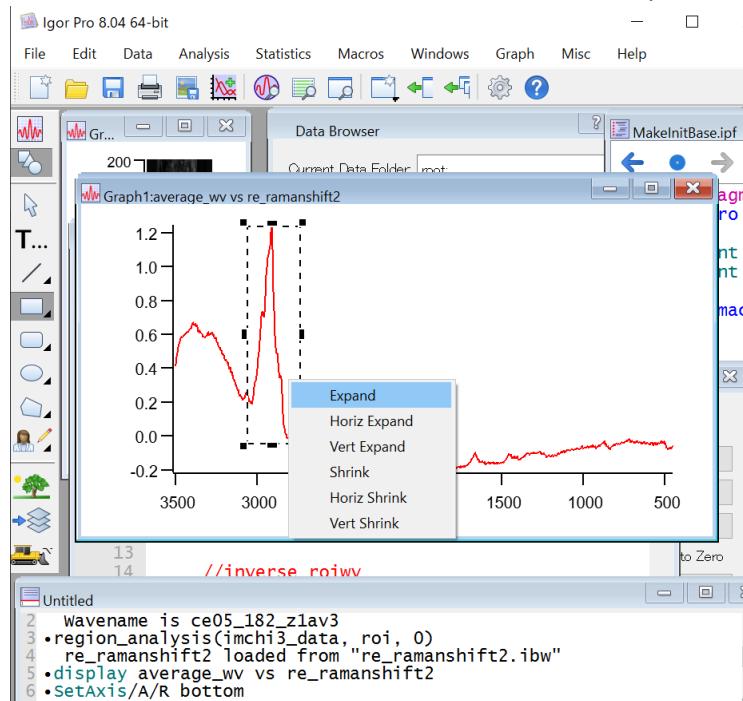
平均スペクトルを読み込んだら、以下のコードで適当に図を出します。

```
display waveav vs re_ramanshift2
```

以下のような図が出るはずなので、フィットしたい領域を拡大しましょう。



拡大は適当なところを四角で囲って、右クリック expandです。



2. wcoefの用意

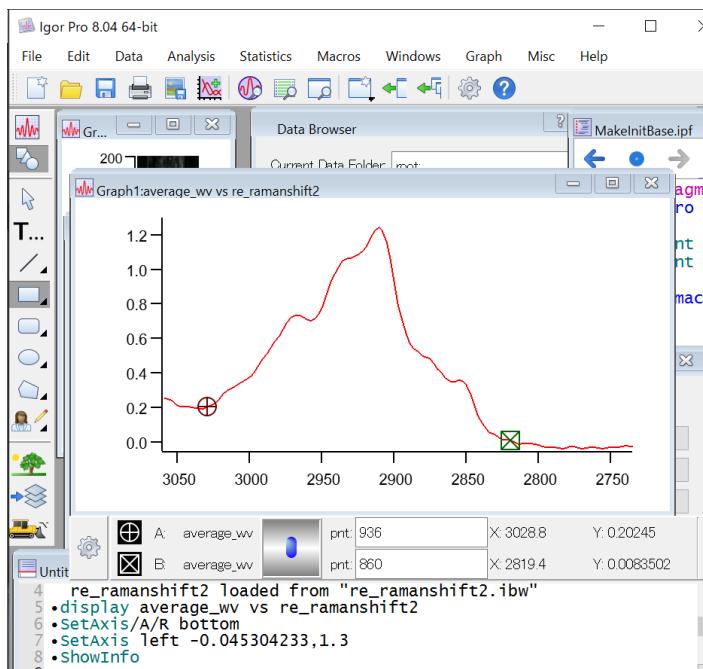
加納研で行っているガウスフィットは、あらかじめ与えたbaselineと大まかなガウス関数の形(振幅、位置、幅)でフィットをします。そのため、まずはフィットのための初期値を与える必要があります。初期値はグラフの形を見ながら定義します。解析虎の巻などで、便利な定数が書いてあったりします。例えば、CH regionを6つのガウスでフィットする場合の定数の数は、 $2(\text{baseline}) + 3(\text{gauss}) * 6 = 20$ 個となります。

```
make/O/N = 20 wcoef
//適当にwcoefという名前にしています。
wcoef = {0, 0 ,0.5, 2850, 10, 0.9, 2870, 10, 0.5, 2930, 10, 0.3, 2950, 10, 0.5,
2910, 10, 0.2, 3000, 10}
//適当に、ピークを見ながら値を代入します
//最初の2つは0で良くなりました（次のinitial fit中にベースラインも変更してくれます）
```

3. Initial fit

次に、fitする範囲を決めて、initial fitをします。

この時に、ガウスの位置や幅、ベースラインを決めています カーソル (ctrl + i) でfitしたい領域の高波数側と低波数側に置きます。

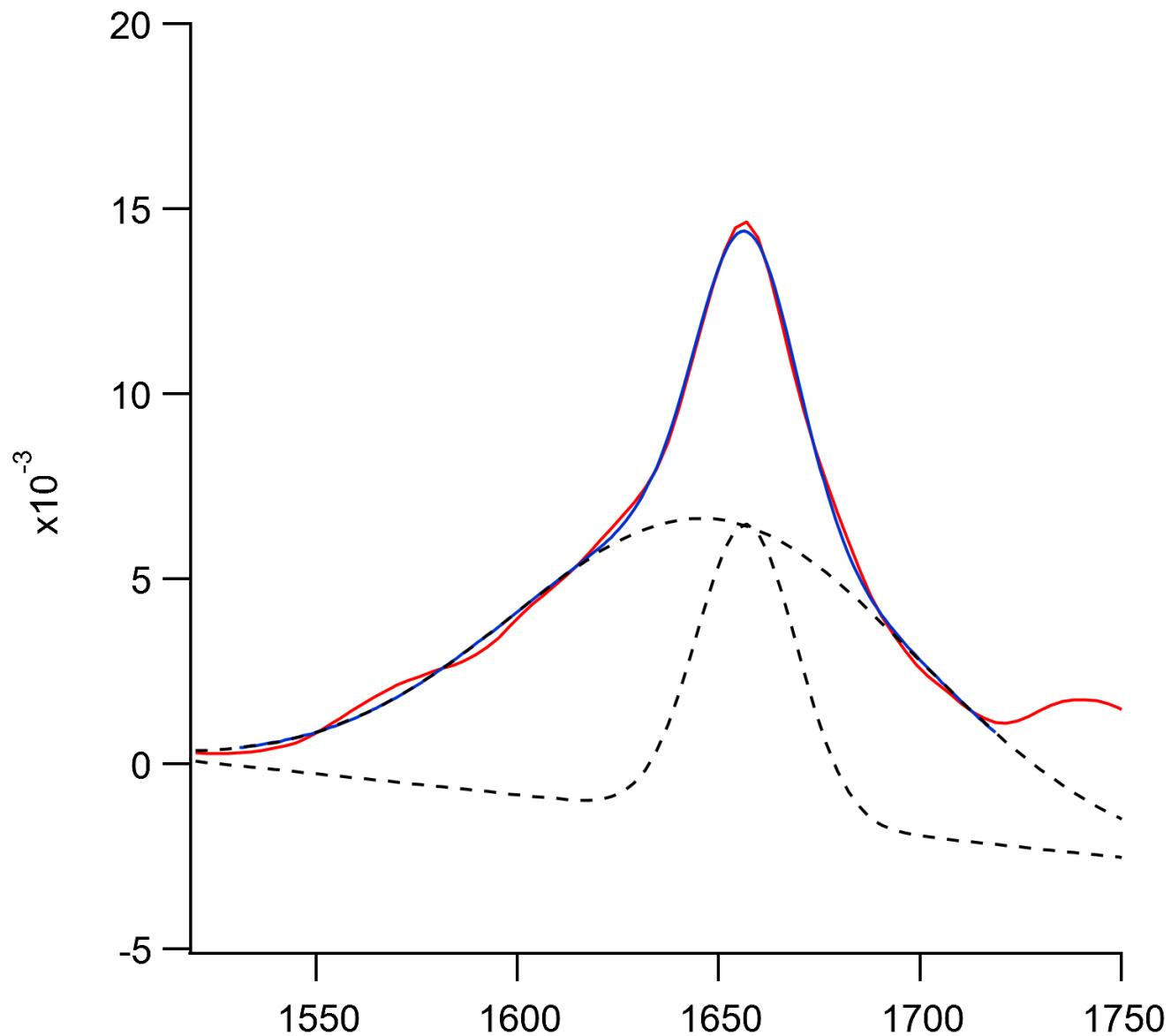


この状態で、コマンド (ctrl + j)から initialfitを実行します。

```
InitialFit(wv, axis, wcoef)
//wvは平均スペクトル（グラフ化したもの）の名前です。
//axisは横軸です
//temp00という名前以外でも大丈夫です
```

fitがうまくいくと、このようになります。

1/23の更新で個別のガウスを点線で表示するようになりました。



4. Fit Imageの作成 Fitがある程度うまくいったら, Fit imageを作成します.

```
MakeFitImages(wv,wcoef, zNum)
//最初の引数はfit したい4次元waveの名前 (例 imchi3_data)
//2つめはパラメータを入れたwaveの名前 (ここでは wcoef)
//3つめはz方向の枚数-1 (z stackしていないなら0)
```

z stack時のfitと画像表示について

z stackで何枚かとった場合, そのままmemをかけることができます.

そのままfitも3次元でかけることができます.

やり方は上記と同じですが, 最後のMakeFitImagesで5つ目の引数をz stackの枚数-1とします (5枚撮った時は4としてください.)

出力される画像はFitImage1Z0のような名前になります.

最初の数字はいくつ目のガウスでフィットしたかを示し, Zの後の数字はz stack何枚目かを示します.

Image Chunkの作り方

上記のMakeFitImagesはfitの都度、画像を生成します。連続していくつかfitしているとよくわからなくなることがあったので、fit結果をchunkにして返す関数を作りました。

```
MakeFitImageChunk(wv,wcoef, zNum)
//最初の引数はfit したい4次元waveの名前 (例 imchi3_data)
//2つめはパラメータを入れたwaveの名前 (ここでは wcoef)
//3つめはz方向の枚数-1 (z stackしていないなら0)
```

上記を実行すると画像は作成されずFitImageChunkというwaveのみが作成されます。このFitImageChunkは4次元waveで(x,y,z,GaussNum)です。GaussNumは何番目のガウスフィットかという意味です。
例えば1660, 1680で4枚のz stack, X=101, y=301のデータをフィットした場合には(101, 301, 4, 2)となります。
1チャンク目を取り出せば1660fitのZ stack imageとなっています。

ピーク位置フィット

ガウスの振幅ではなく、中心位置で画像作成

以前のmakefitimage.msからの派生なので、最新版と統合はしていない。(baseline fit等が必要になっている) 近日修正

```
MakeFitImagePeak(frompix, endpix, gausNum, wcoef, zNum)
```

特定波数と特定波数の比のイメージ解析

フィッティング後の画像でのratiometric imageの作成

0. 用意するもの

フィット後の画像 (2つ)

Ratiometric_analysis.ipf

1. ガウスフィットの実行

詳しくはフィットのreadmeで書いてあります。うまくいっていれば"Fitimage"みたいな名前のwaveがすでに手に入っているはずです。(ctrl+Bでwave ブラウザを見てみましょう)

2. ratiometric imageの作成

Ratiometric_analysis.ipfをコンパイルして以下です

```
ratiometric_image(image1, image2)
//image1とimage2には比をとりたいフィット画像を入れましょう。
//image1が分子, image2が分母になります。
```

3. ratioimageという名前のwaveが作成されて、画像が出ていると思います。

Igor便利技

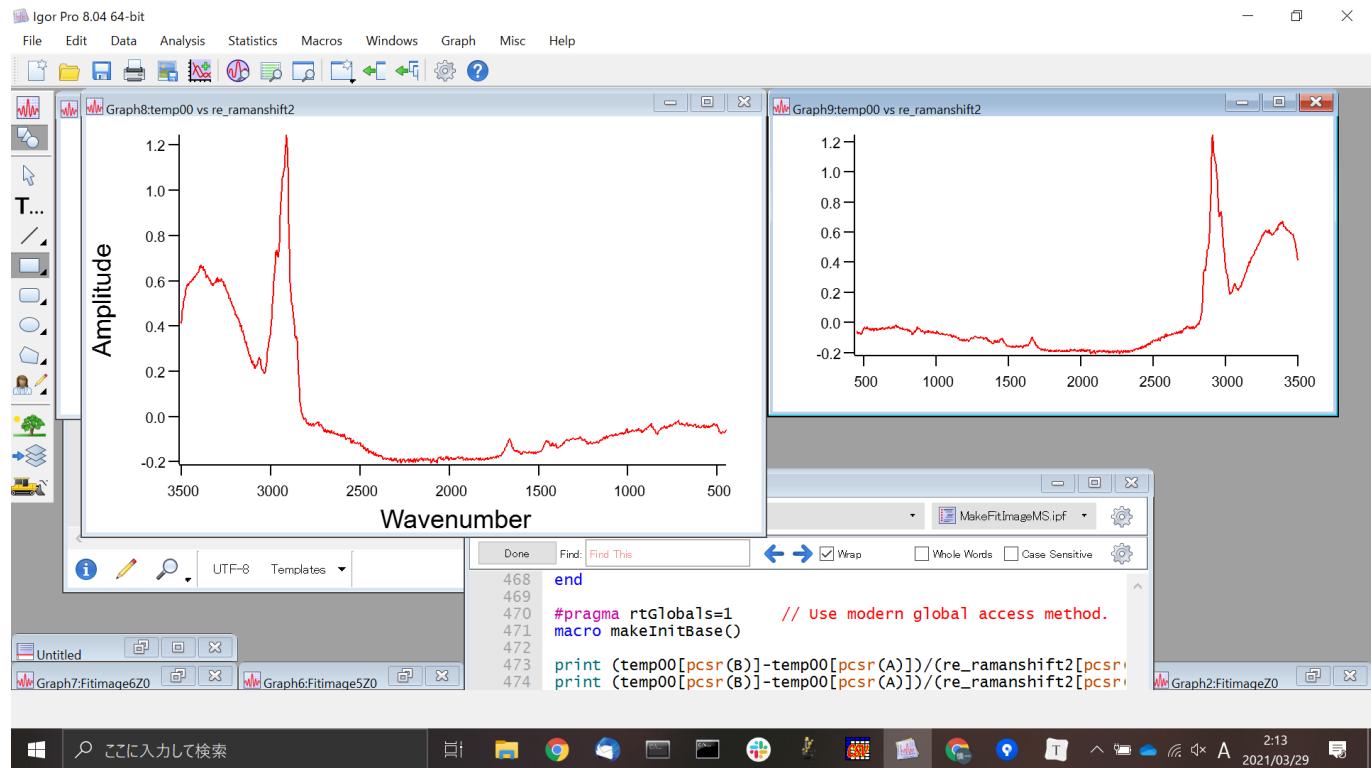
ここでは, Igorで便利な技を紹介します.

グラフスタイルマクロ

Igorにはグラフスタイルマacroという、複数の似たグラフを作る際に便利な機能があります.

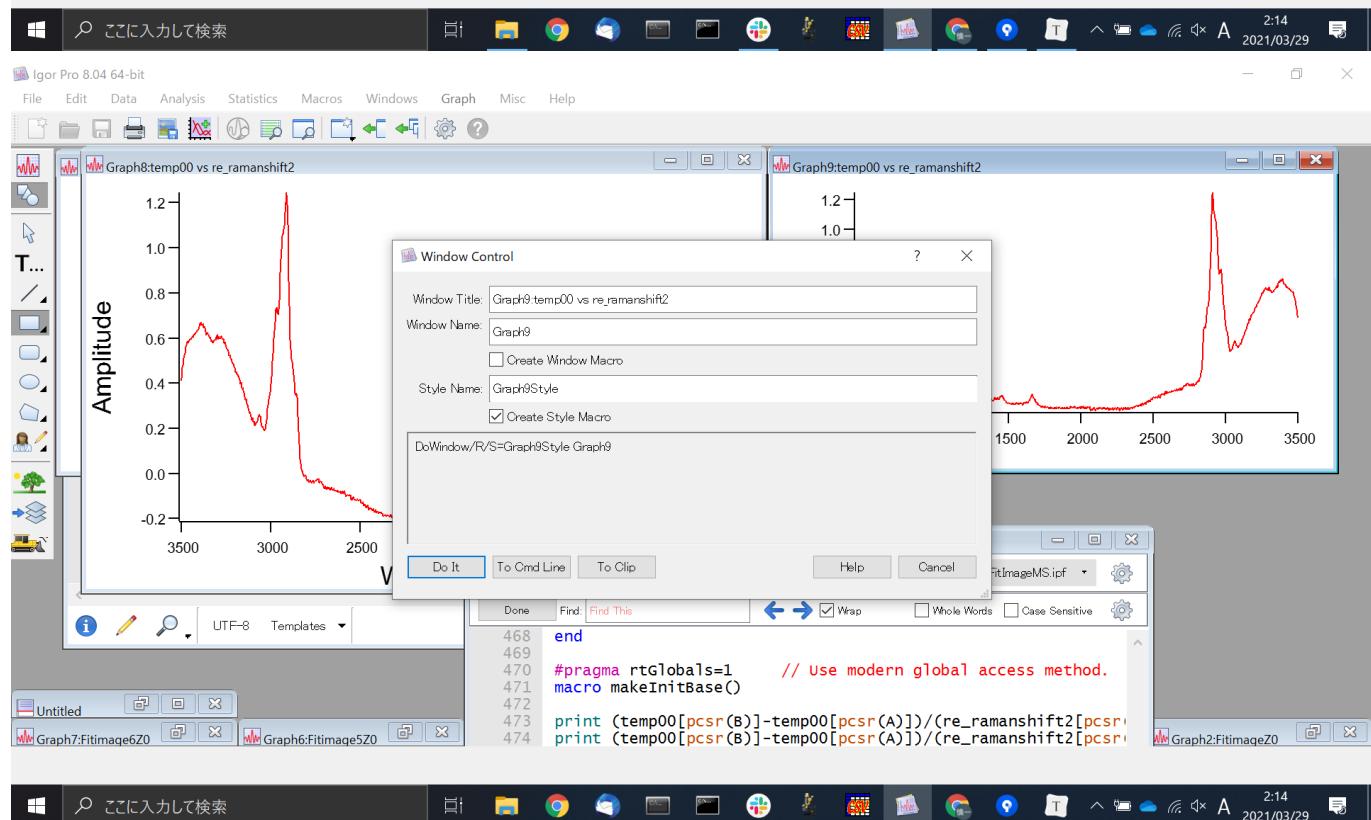
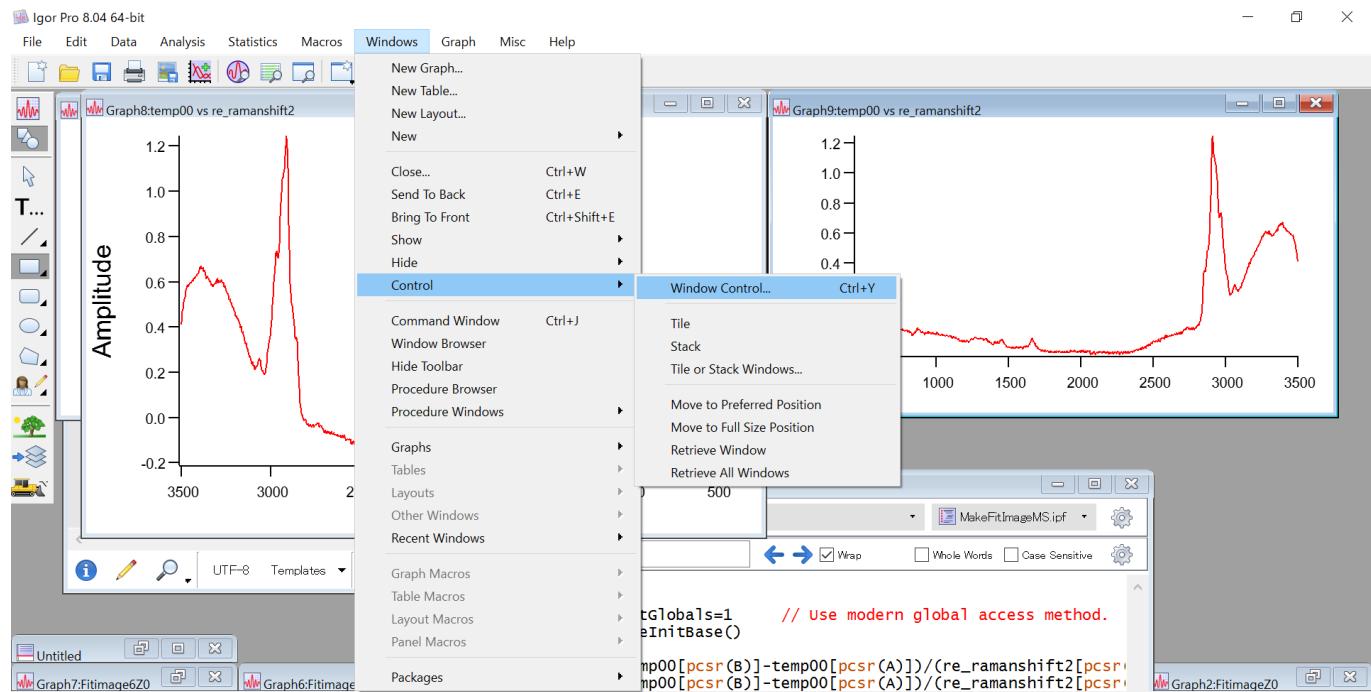
これは、一つ作ったグラフから、そのスタイル(軸やラベル、色などの情報)のマacroを作成して、ほかのグラフに適応できるという機能です.

例えば、以下の左のグラフのようなスタイルを、右のグラフに適応したいとします。



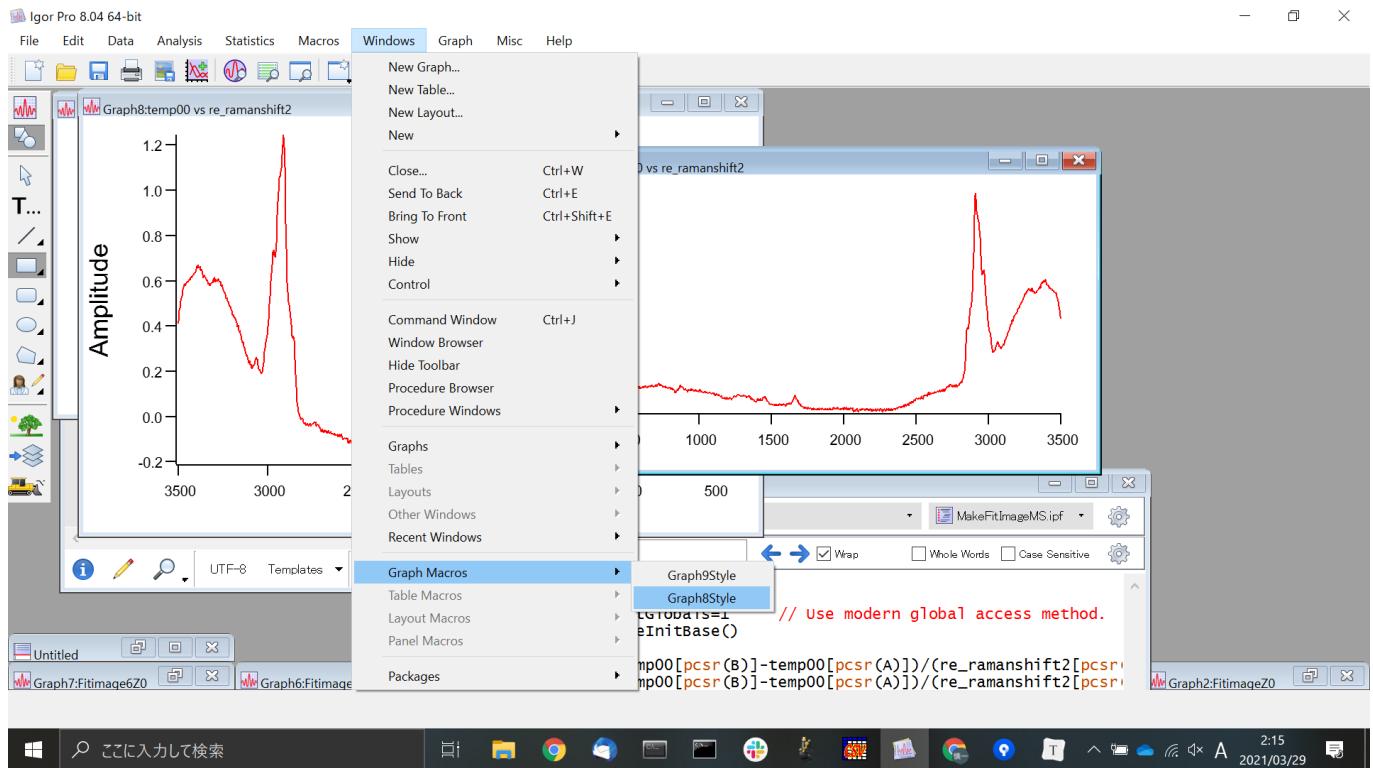
windowからcontrolを選択すると、以下のような画面が出てきます。

style macroにチェックして、OKを押します。



次にwindow/graph macroを押すと先ほどの作成したmacroが使用できます。
(ここで、styleを適応したいグラフをアクティブにしている必要があります。 一回クリックしてから上記操作

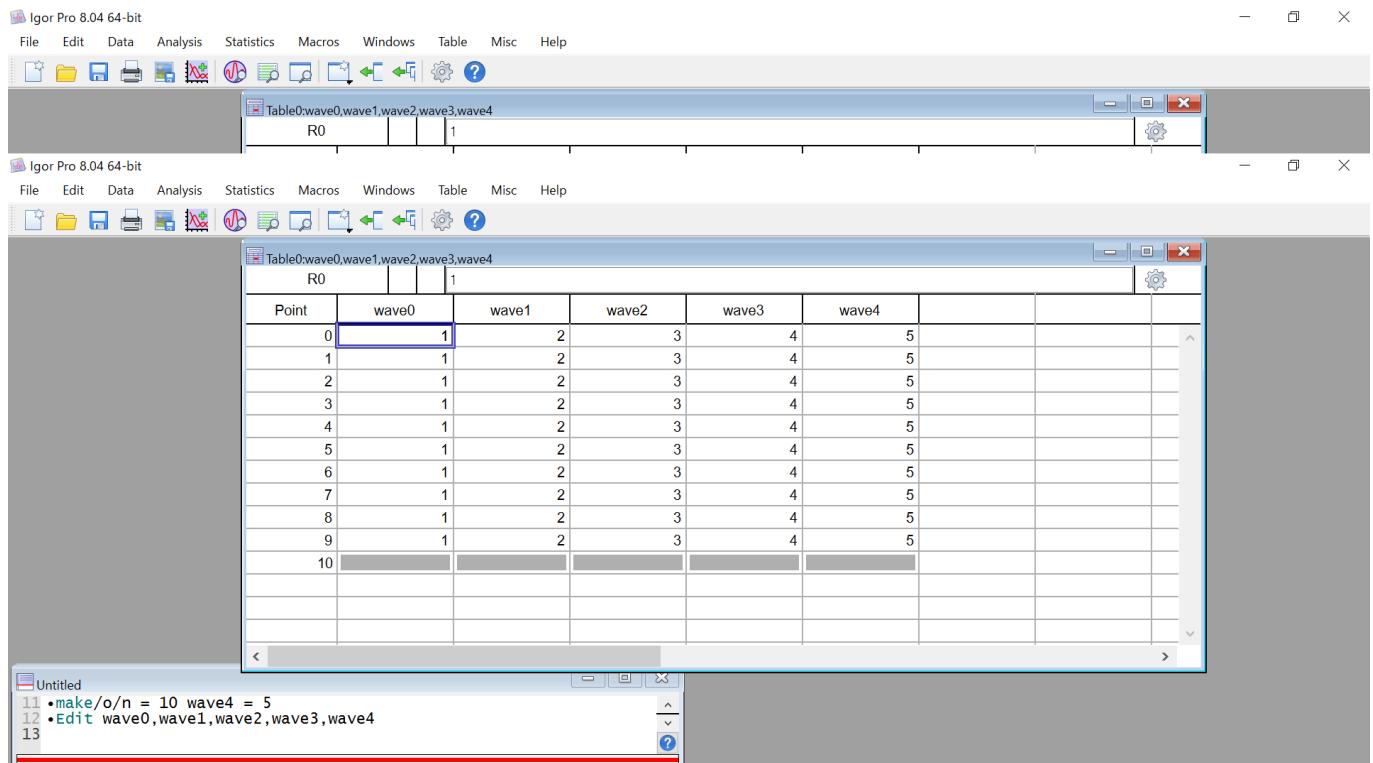
をすると安心です。)



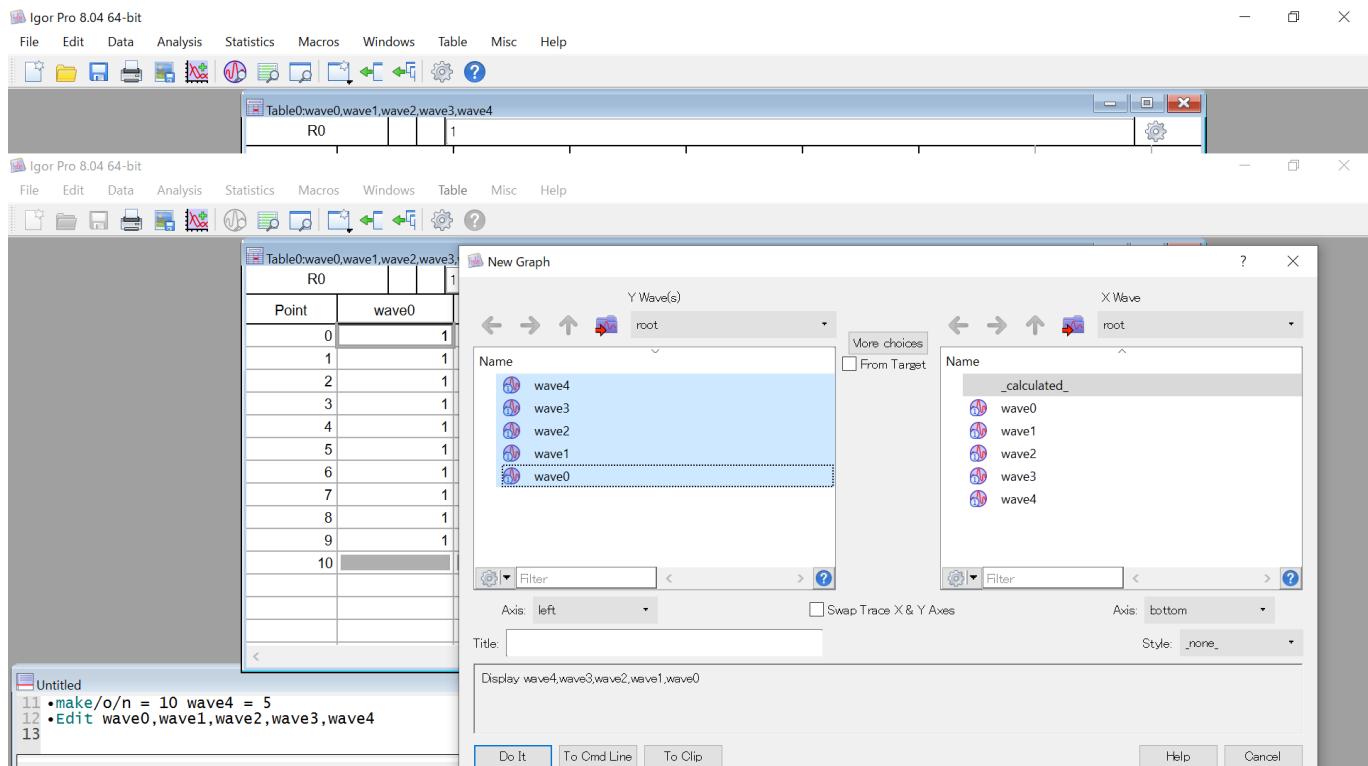
複数のプロットに別の色を付ける

ここでは、一つのグラフに複数のプロットをした際の便利な機能を紹介します。

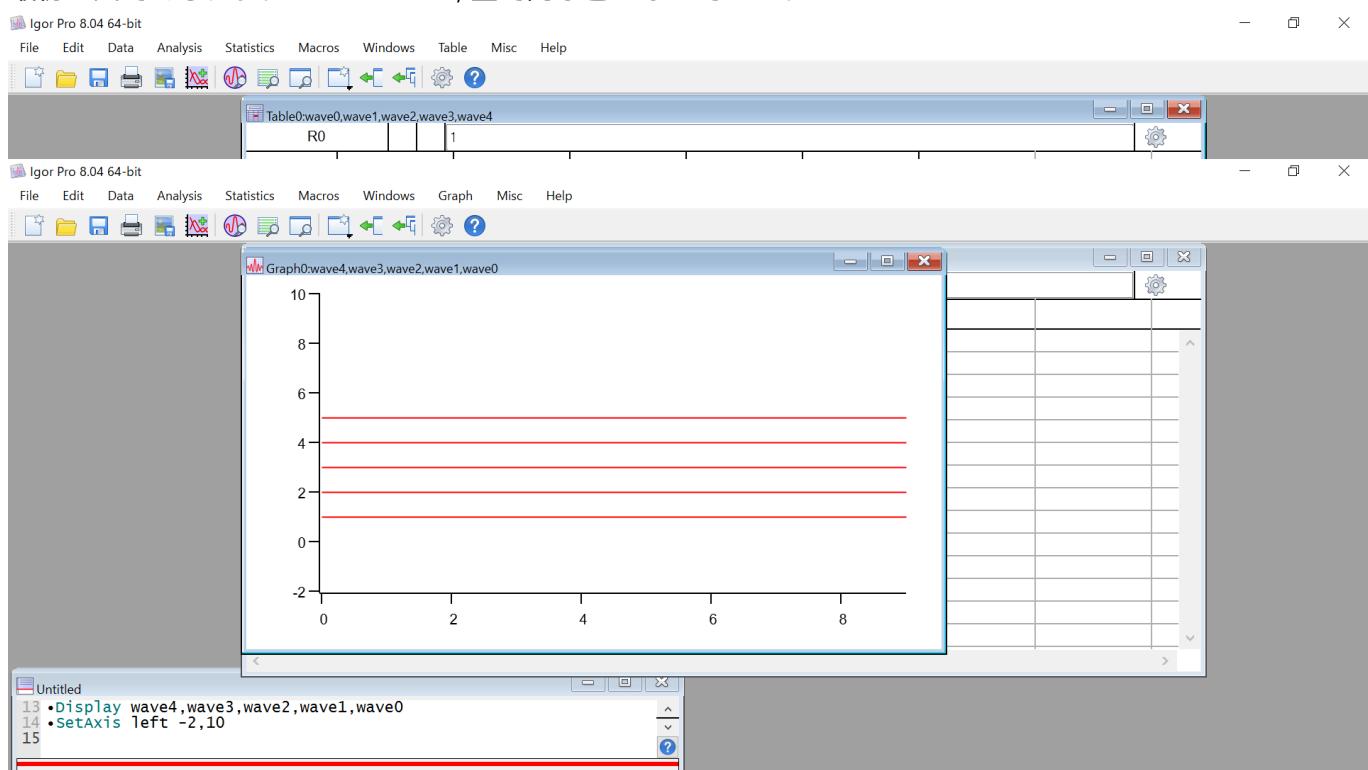
まずは適当にwaveを作ります。



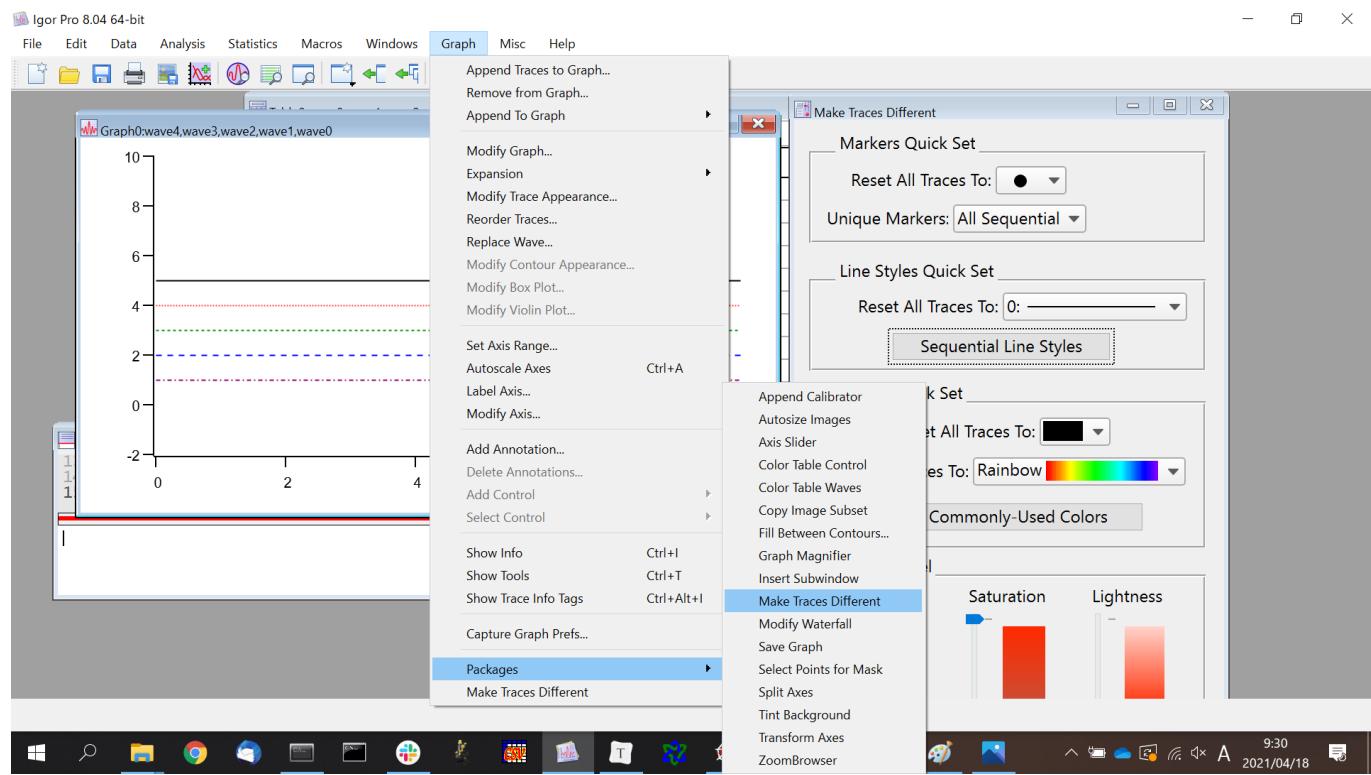
例として5つのグラフに色を付けてみます。このようにWindow/new graphで出てくるウィンドウから、y waveを複数選択してdo itで複数プロットを一つにまとめることができます。



最初に出てくるグラフはこのように、全て同じ色になっています。

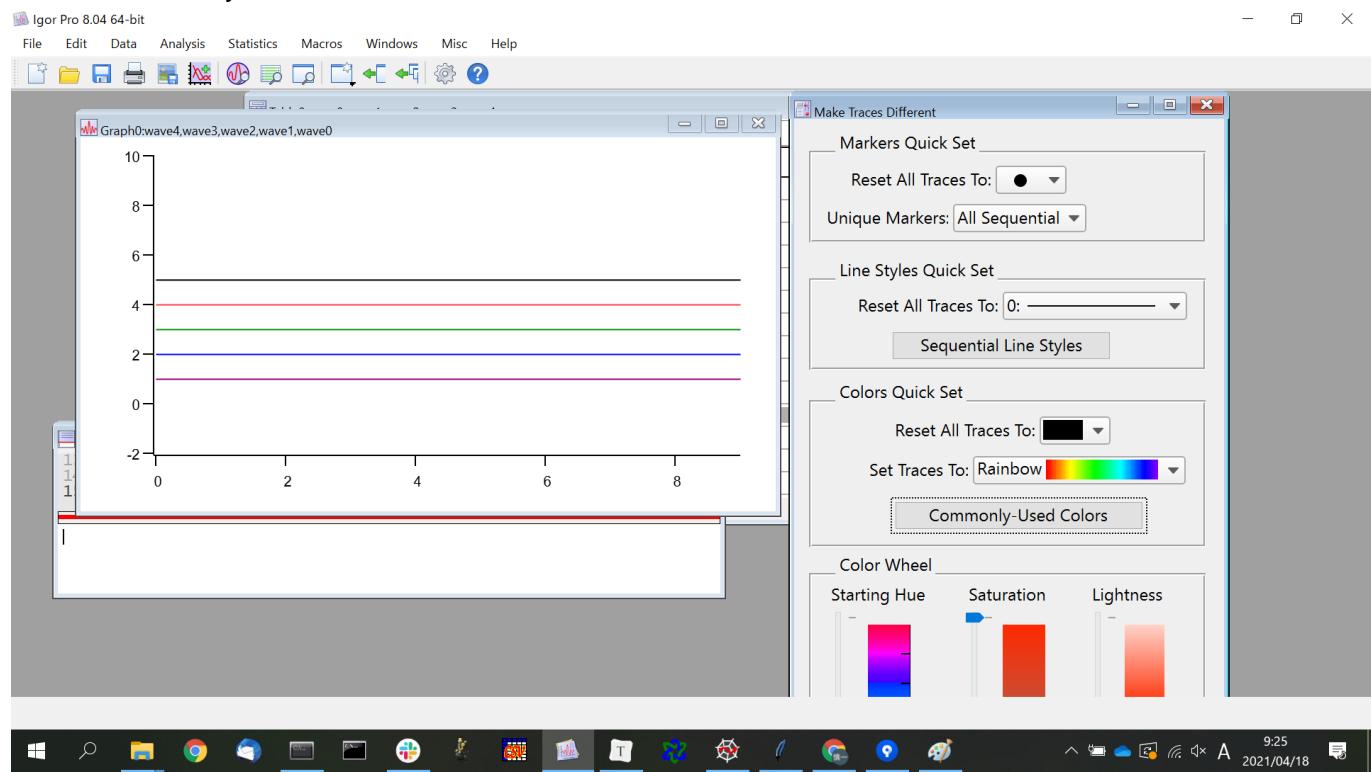


この状態で、Graph/Packages/make traces differentを選択します。

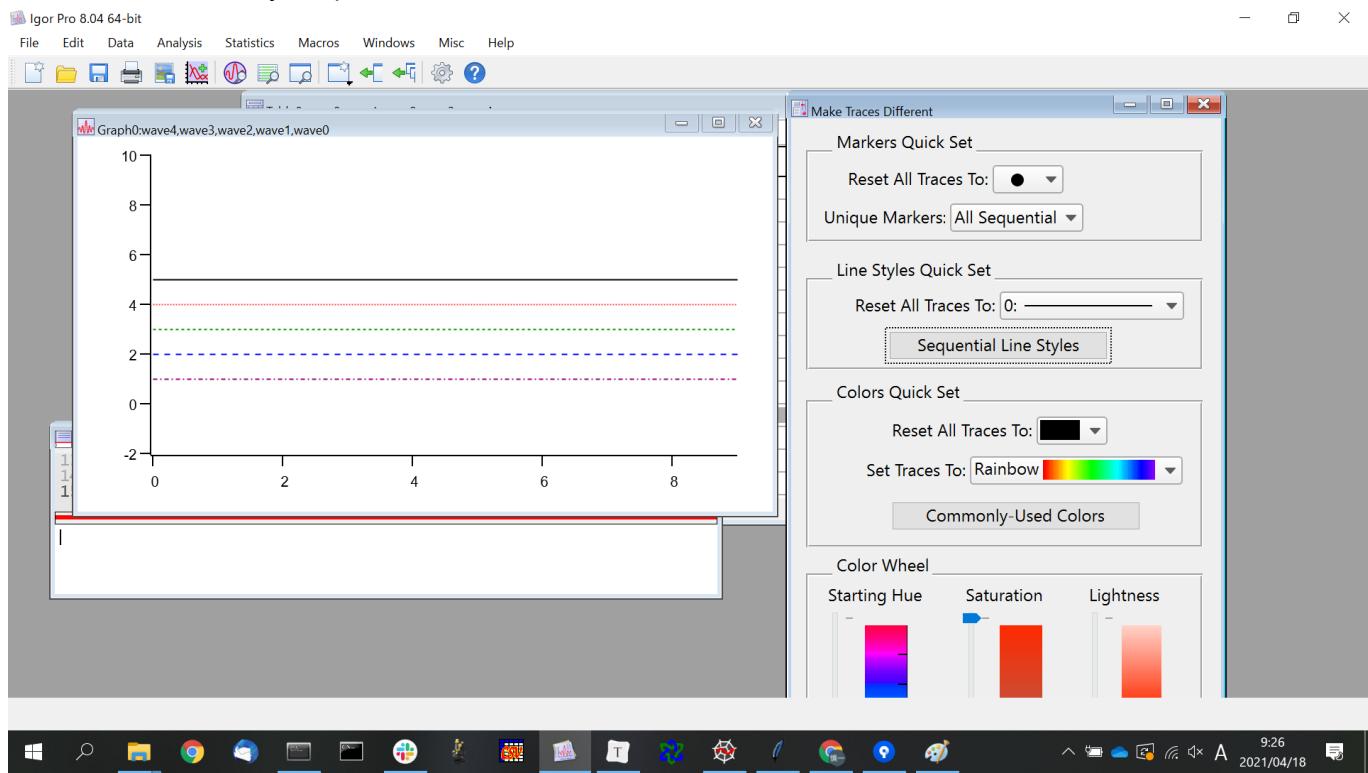


ポップアップするウィンドウの中の、Colors Quick Setで好きなカラーセットをSet Traces To のコンボボックスから選びます。(ここではRainbowにしています)

その後、commonly used colorsをクリックするとこのようにプロットに色がつきます。



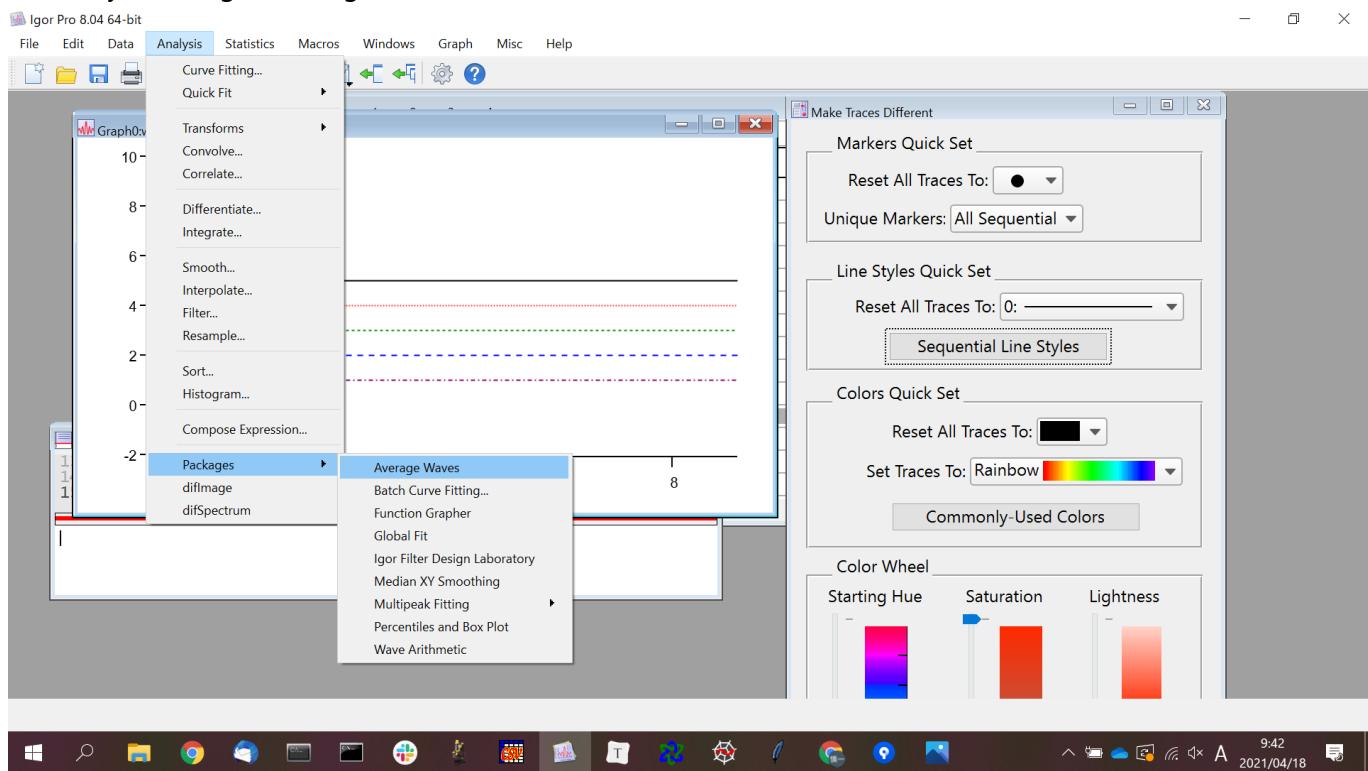
ちなみに、上のLine styles quick setをクリックするとこのように実線や点線などを変えることができます。



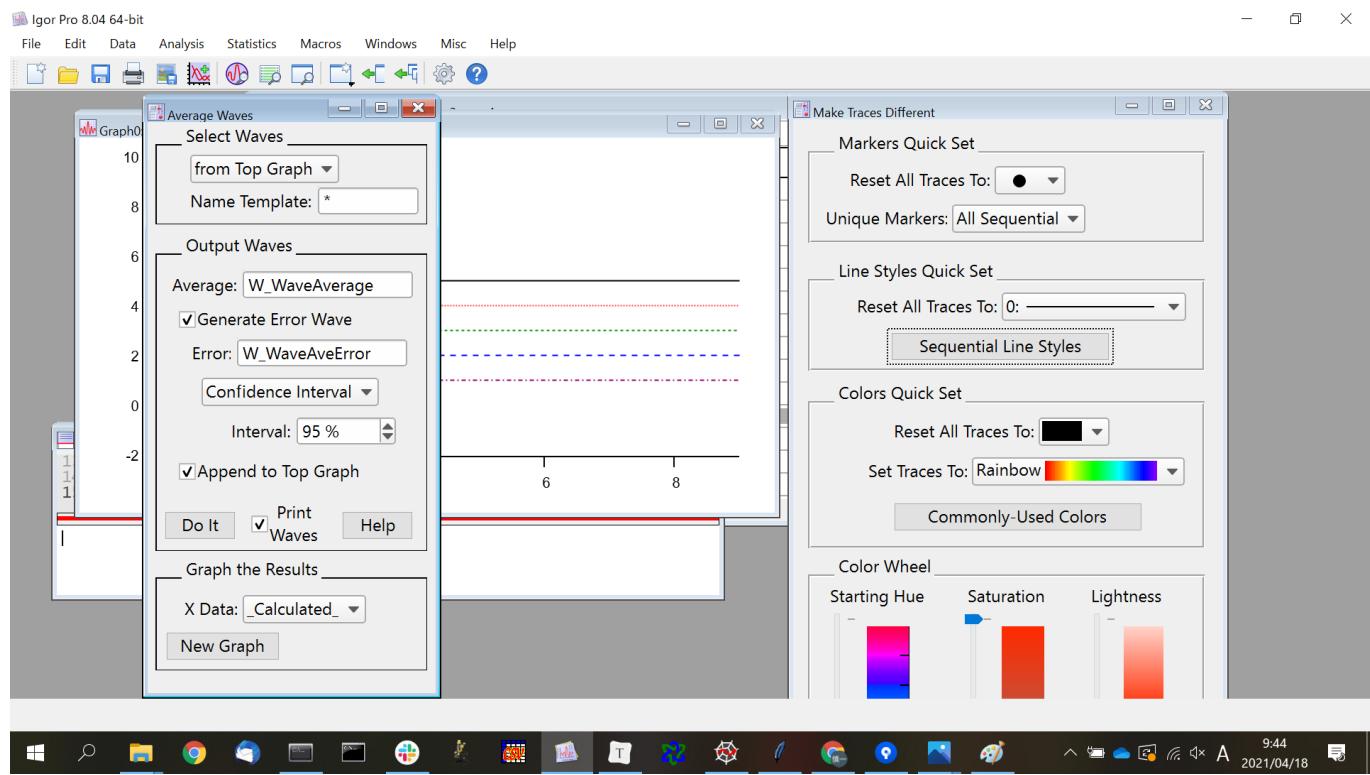
複数プロットの平均値とsd (or se, 95% CI)をプロットする

こちらはあまり使わないかもしれません、念のために

上記で紹介したような、複数プロットが表示されている状態で、(正確にはプロットしておく必要はないですが) Analysis/Packages/Average Wavesを選択します。

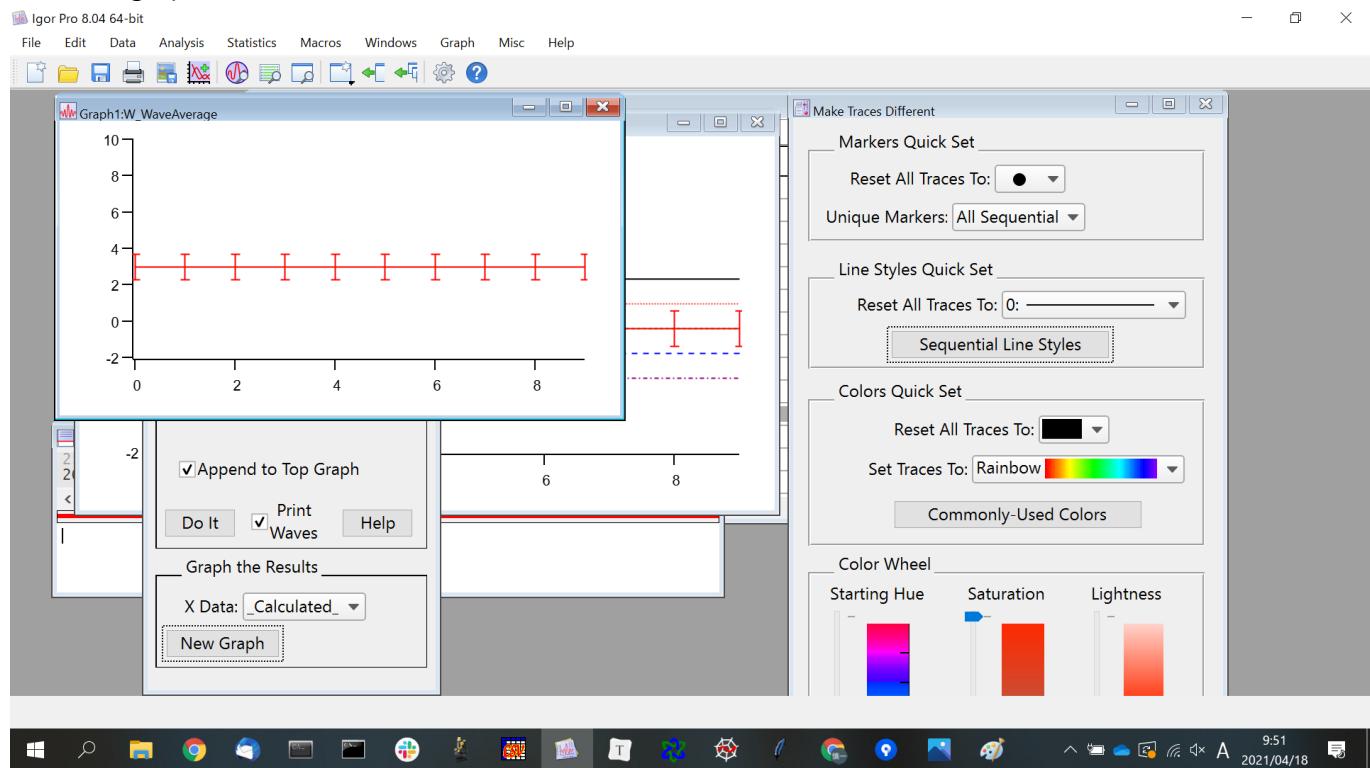


出現するウィンドウでOutput Wavesの中で、デフォルトではConfidence Intervalとなっている部分を変更すると、エラーバーを変更できます。



この状態で, Do itをクリックすると平均とエラーバーのwaveが生成されます.

次にNew graphをクリックすると先に得たwaveでグラフが生成されます.



箱ひげ図, バイオリンプロットで一つ一つの系列を塗分ける

実はこれはマニュアルにも記載ありません.

(Igorがカテゴリプロットを苦手としているのは、作り手があまりカテゴリプロットを使ったことがなかったかららしいです。) -> Igor pro9で実装とのこと、需要があれば記載予定

Igor redimensionの挙動について

Igor の redimension は便利そうだが、挙動がよくわからなかつたのでメモ

- make/o/n=6 test = p
- redimension/n = (1,2,3) test

これは、0,1,2・・・と続く test という名前の 1 次元 wave を 3 次元に変換している。

Table0:test		
R0		0
Point	test	
0	0	
1		1
2		2
3		3
4		4
5		5
6		

から

8

変換後は1行2列3レイヤの行列で、数値はきちんと保持されている。望ましいredimensionの挙動しかし次元の総数があっても以下はうまくいかない

- redimension/n = (1,2,3) test
 - redimension/n = (1,3,2) test

見ての通り、3列目が0で埋められている。

このように、redimensionを成功させるには、次元総数があつてはいるだけではなく、移動先の次元が元の次元と同じデータ点数が多くないとデータが0で埋められてしまう。

解析虎の巻よりフィット時の初期値まとめ

ここでは、fit時に有用な数字を紹介します。

元は解析虎の巻というファイルです (どなたが作ったかはわかりません)

fit時に, wcoefという名前のwaveを作つて, 以下の数字を代入するとfitがうまくいきやすいです(必ずうまくいくとは限りません)

OH region

3つのガウス

```
wcoef = {0.5,-0.0001, 0.5 ,3670,100,1.7, 3400,150, 1,3200,100}  
//baselineの値は適当です
```

aromatic CH

1つのガウス

```
wcoef={ -0.277265,0.00012024,0.1,3065,5}  
//baselineの値は適当です
```

1755と1650

2つのガウス

```
wcoef={0.0075,0.00001,0.01,1742,13.1,0.1,1656,14}  
//baselineの値は適当です
```

CH region

2614~2872

```
wcoef=  
{-2.71478,0.00102714,0.04,2850,8.45146,0.12,2868.28,10,0.09,2770,10,0.372906,2713,  
10,0.152805,2661.95,10.3172,0.35006,2690.4,13.694}  
//baselineの値は適当です
```

1450

2つのガウス

```
wcoef={-0.31,0.0001,0.1,1455,13,0.1,1435,12}  
//baselineの値は適当です
```

1400-1200のフィット

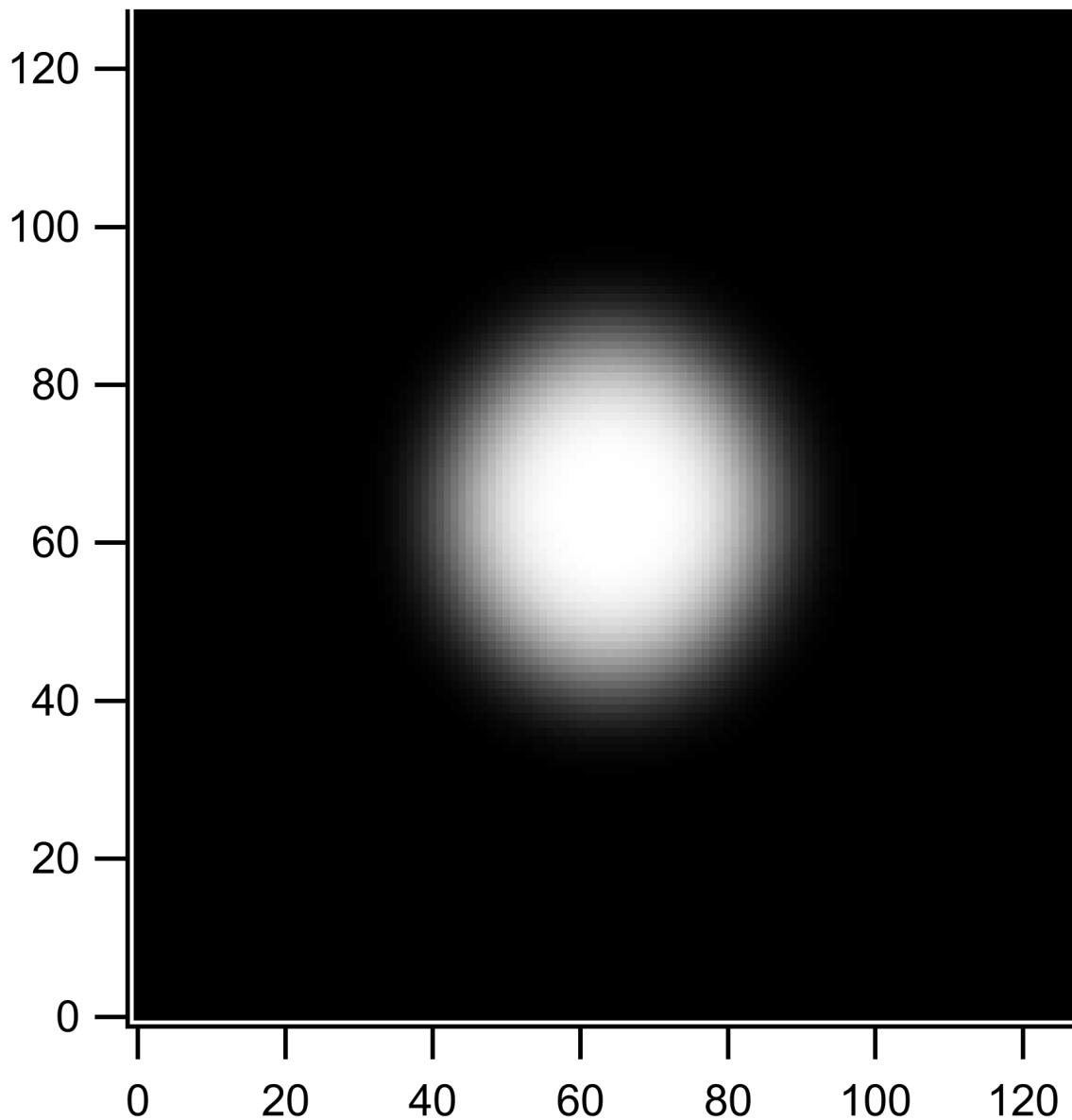
3つのガウス

```
wcoef={0.0036,0.000005,0.008,1303,6.8,0.02,1281,74,0.01,1264,10}  
//baselineの値は適当です
```

Image plot

Image plotで縦軸, 横軸を設定する方法

Image plotを普通に作成すると, 下の図のように横軸, 縦軸はともにwave pointとなります.



この縦, 横軸を単位ありの軸にする方法の解説です (ここでは距離で試してみます)

ちなみに私は線虫の神経活動を測定して, ヒートマップにする際にこの手法を使いました.

以下の手法はIgor 8 manual (help/manual) II -320あたりに記載があります.

0. 用意するもの

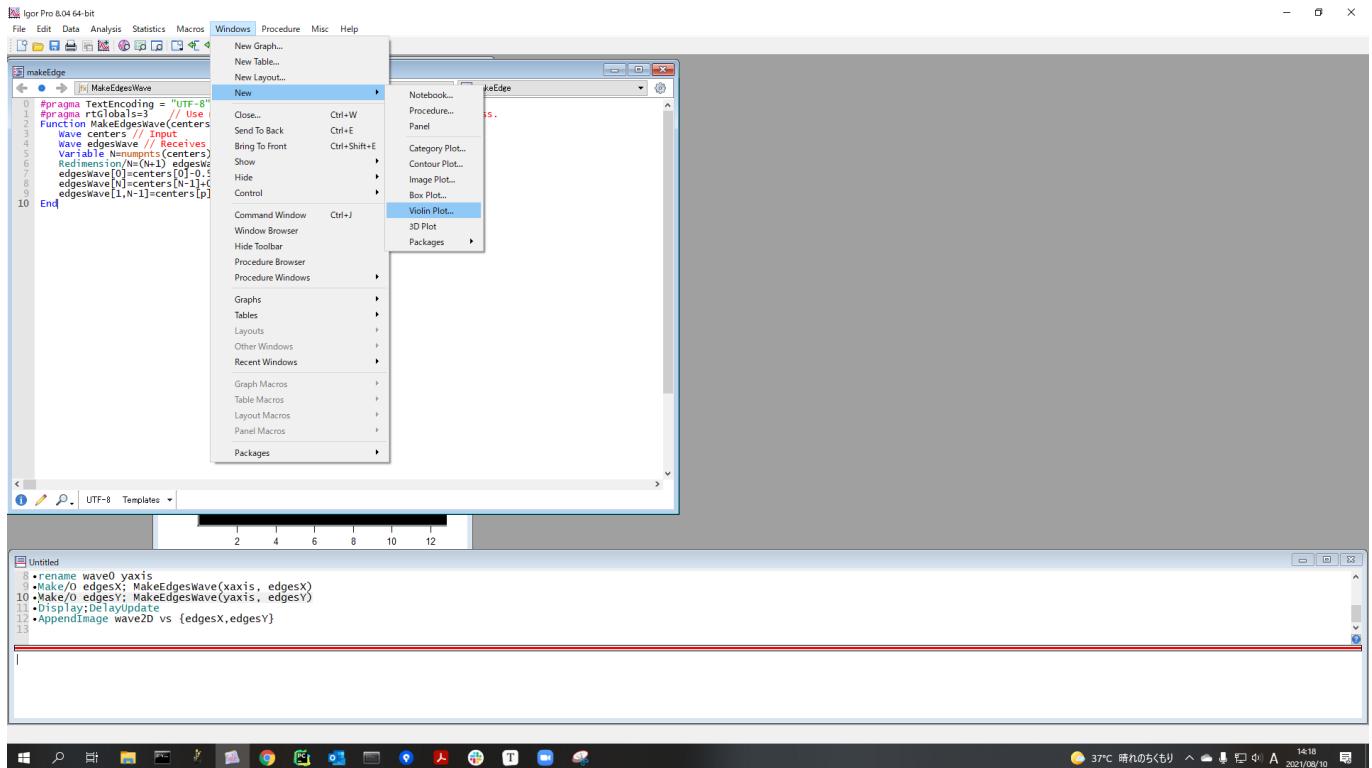
1. 出したいイメージ (2次元waveとして読み込んでください. 誤って1次元wave の集まりとして読み込んだ場合にはData/Concatenate wavesでくっつけてください)
2. 縦の軸 (1次元wave), 横の軸 (1次元wave)
3. MakeEdgeWave.ipf

1. 以下の命令を実行します

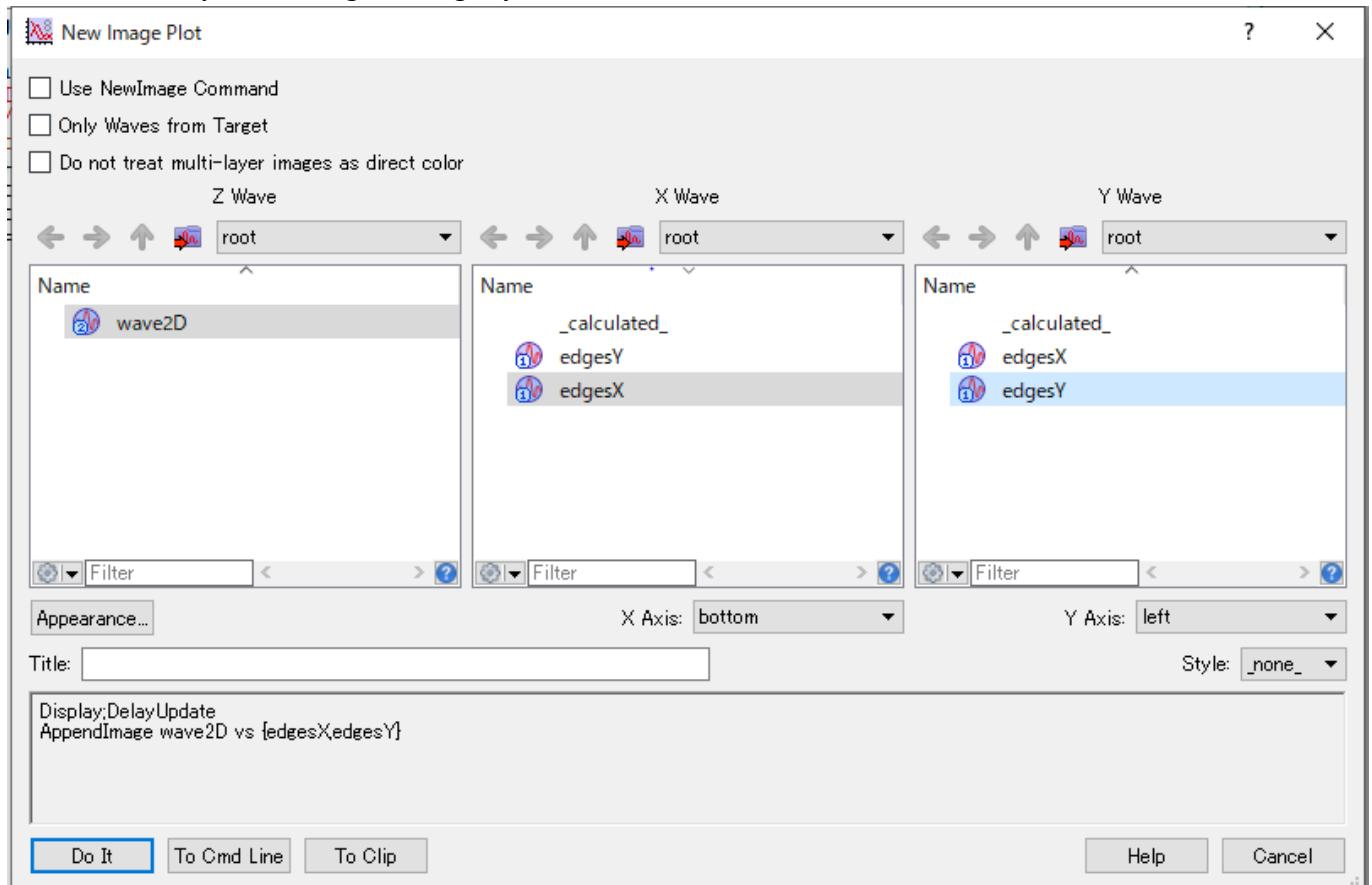
```
Make/O edgesx; MakeEdgesWave(xaxis, edgesx)
```

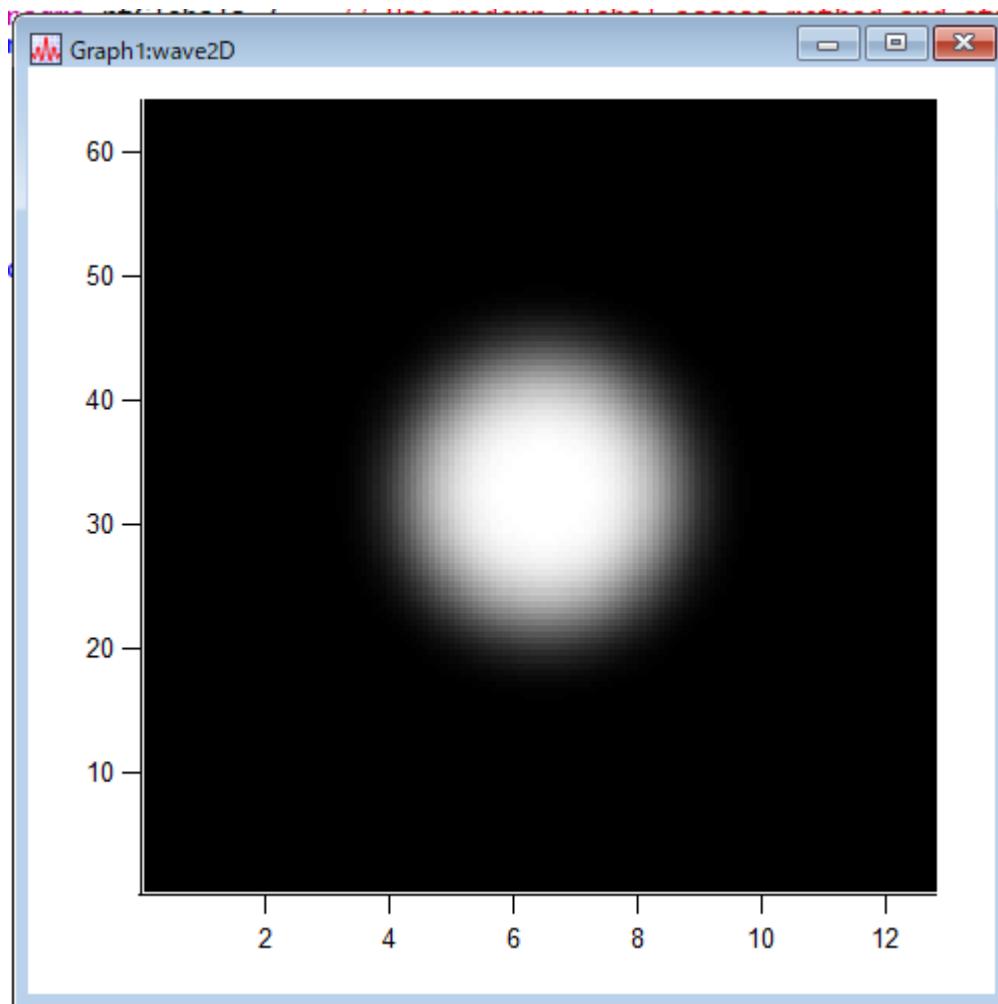
この関数がやっているのは、ただの1次元waveからImage plotの軸として使用可能なwaveへの変換です。

2. この状態でWindows/New/Image plotを実行すると以下の画面が出現します。



ここで、x wave, y waveをedges x, edges yにすると、以下の画像が出現します。





横軸、縦軸が不均一になっています。このように任意の軸を用いることが可能です

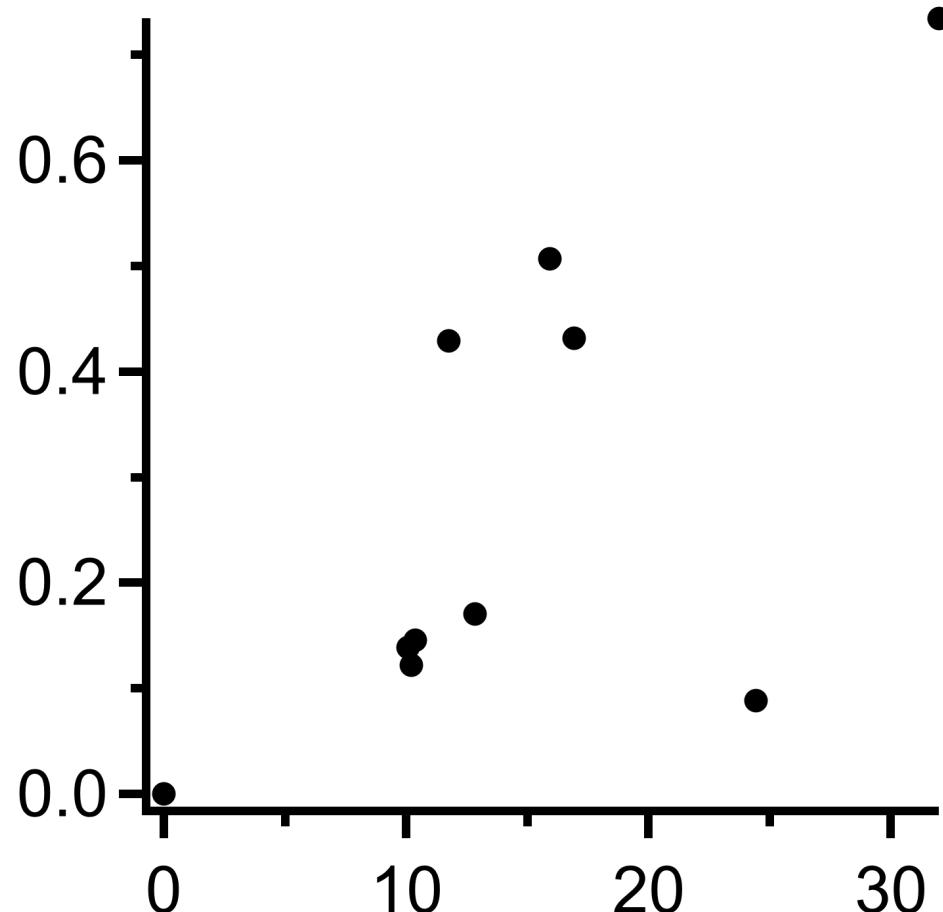
グラフギャラリー

今まで作ったグラフをおいておきます.

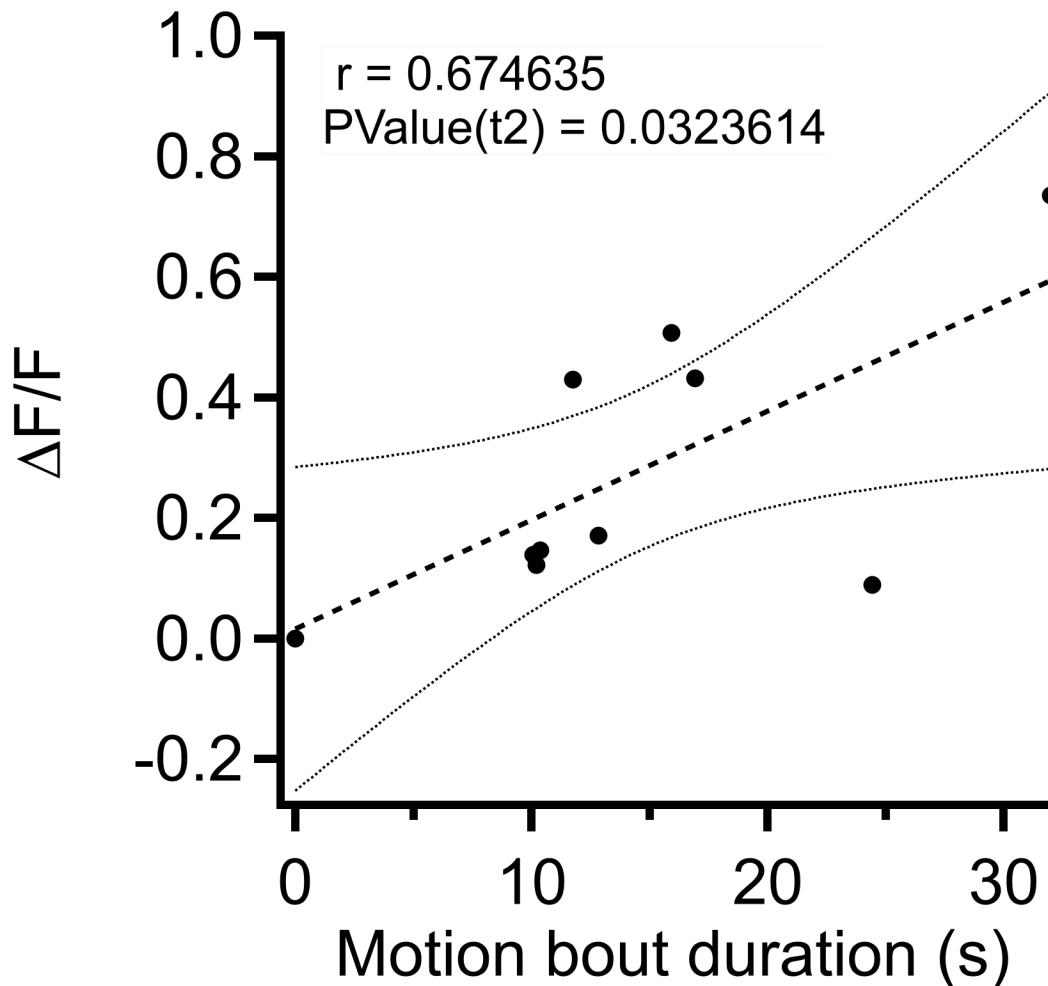
ここにあるものはIgorで作れるはずなので、必要なものがあれば適宜ご相談を.

1. Scatter plot

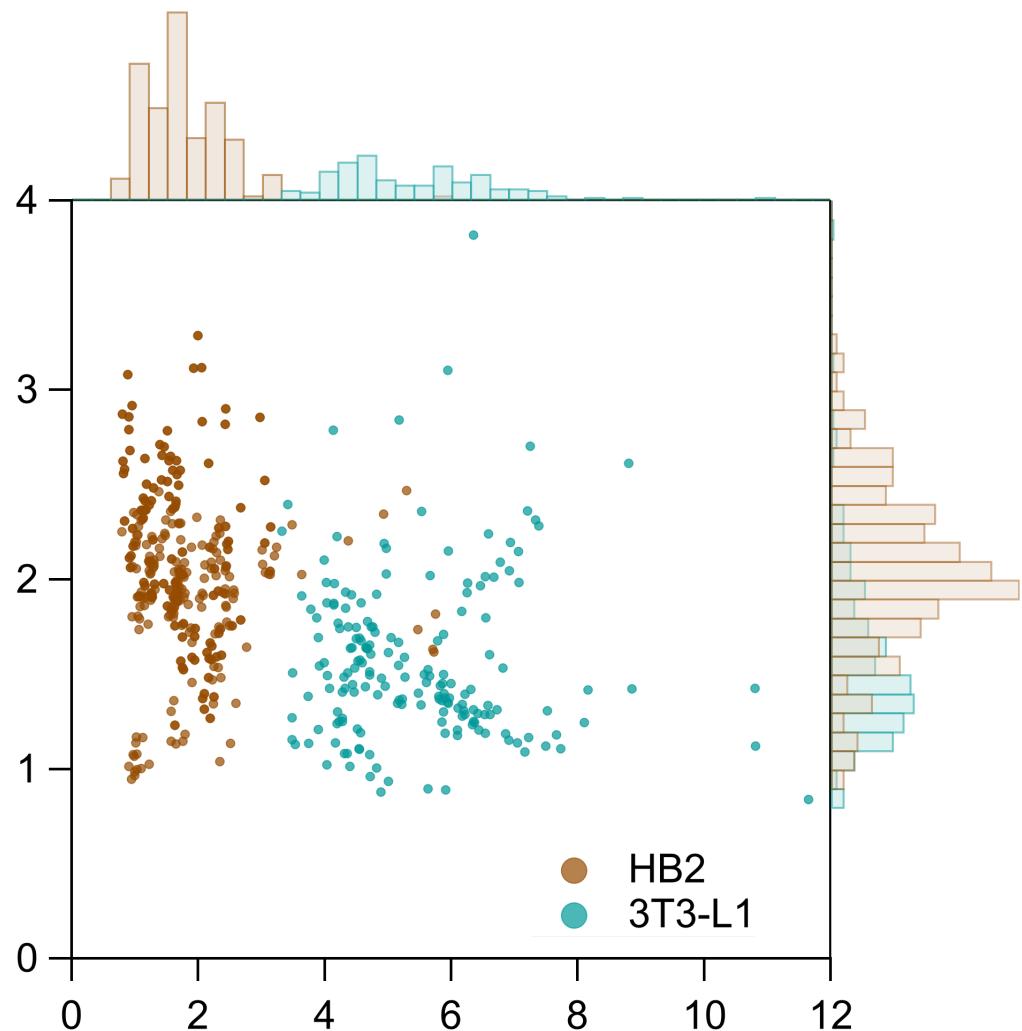
1. Scatter plot



2. Scatter plot with linear regression (and confidential interval)



3. Scatter plot with histogram



2. Beaswarm plot

Igor proで線形回帰

以下は私の備忘録です. 論文書いていて, 線形回帰をしたかった時に必要だったので

最終目標は以下のようなグラフを作成すること

0. 用意するもの

x座標にあたるデータ (wave), y座標にあたるデータ (wave)

1. Correlation analysis

まずは Statistics/Correlation analysisで相関解析を行います.

ここでは $r=0$ との検定 (無相関ではないことの検定) と r (相関係数) の計算などが行えます.

別にIgorで行わなくてもいいですが, あのグラフ化と一緒にできると楽なので

Live resultでp value見れます.

2.Scatter plot

これは単純にWindow/New graphで

X軸の値とy軸の値を選択して適当にplot

最初は上のような変なグラフですが, 見た目を調整すれば大丈夫

3. 回帰直線のプロット

回帰直線のプロットはcurve fitで行います. (ほかのやり方もあるかもですが, 今のところ知りません)

```
CurveFit line wave0 /X=wave3 /D /F={0.95, 1}
```

line: 直線でfit

wave0: fitする値 (scatter plot)のy軸の値

/X=wave3: scatter plotのx軸 /D: 倍精度 /F={0.95, 1}: 信頼区間の計算について, 1つ目の値は何%信頼区間を計算するか (普通は95%)

2つめの値は計算のモード, 1だと回帰直線の95%信頼区間のみ, 予測区間を出したければ別のコマンド

上記で実行すれば回帰直線と, 95%信頼区間がプロットされるはず

GUI based analysis (author: Kyosuke Tanaka)