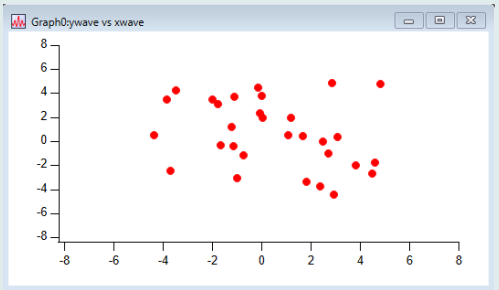
Tips 6 – 三角分割についての考察

これは、三角分割と補間に関するブログポストを手順を中心に再編集したものです。
（ https://www.wavemetrics.com/news/planar-triangulations ）

手順

範囲 [-5,5] における一様分布からサンプリングされた 30 個のポイント {xi, yi} の集合を考えます。

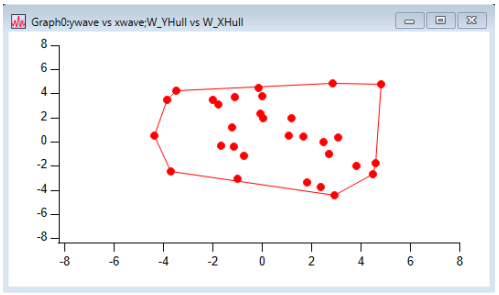
新しいエクスペリメントを作成したところからの手順で確認します。



1. コマンドウィンドウで次を入力して上記のような集合を作成し、散布図で表示します。

```
Make/O/N=30 xwave=enoise(5), ywave=enoise(5)
Display xwave vs ywave
ModifyGraph mode=3,marker=19
SetAxis bottom -8,8
SetAxis left -8,8
```

2. このポイント群の凸包（convex hull）とは、ポイント群内の任意の２ポイントを結ぶ線分がすべてその領域内に完全に収まるような、最小の境界領域のことです。
凸包を視覚化する良い方法は、ポイント群の周りにぴっちりとした輪ゴムを巻きつけるようなイメージを持つことです。



次のコマンドを使って、このデータセットの凸包を計算し、グラフに追加することができます。

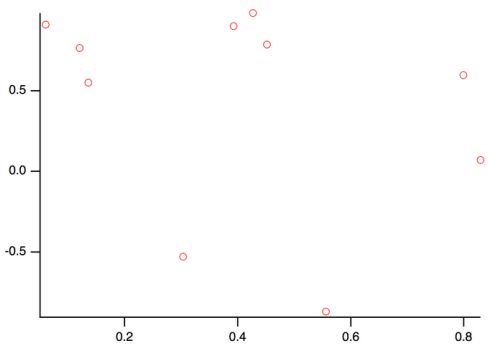
```
Convexhull/C xwave,ywave
appendTograph w_yHull vs w_xhull
```

（ブログ内の DrawPoly コマンドは塗りつぶしの目的で使われているためここでは省略します）
散布集合の凸包が作成されます。
集合内の任意の２ポイントを結ぶ線分は、すべてこの領域内に完全に含まれます。

<ここからは少し概念の説明>

{xi, yi} の各組がある z 値 zi = f(xi, yi) と関連付けられていると仮定すると、任意の {x, y} に対する z の値を知りたい場合があるかもしれません。
追加の情報がない限り、凸包内のポイントについてのみ意味のある記述を行うことができます。
次の手順の論理を理解するには、まず簡単な 1D の類推を考えてみると役立つと思います。

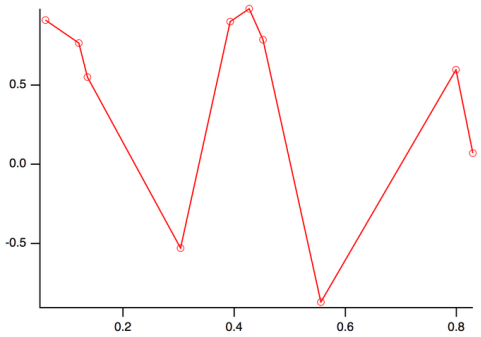
右のグラフは、ランダムな位置 {xi} でサンプリングされたポイント群 y=g(x) を示しています。



XY 平面上のランダムなポイント群

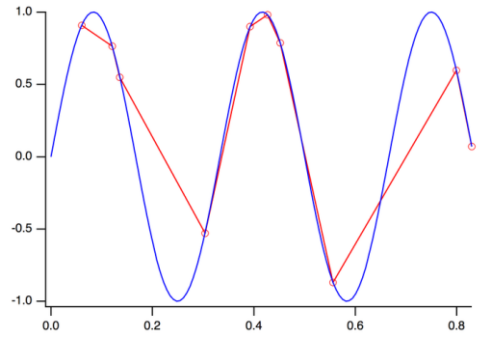
g(x) に関するこれ以上の情報がないため、まずは最も近いポイント同士を直線で結ぶことができる（線形近似）と仮定することから始めます。
実際には、まず x 値の昇順にポイントを並べ替え、その列を直線分断でつなぎます。

この場合、データが下のグラフの青い曲線からサンプリングされたものであることがわかっているとします。



最も近い点を直線で結ぶことによって得られる線形近似

先に検討した 2D 散布図の例に「線形近似」の手法を拡張するには、平面上で最も近いポイントを特定する方法を見つける必要があります。
ここでは、３つの最も近い（共線でない）ポイントの各グループが１つの三角形を定義します。
最も近いポイントを並べ替えて結ぶという 2D/3D の対応関係は、三角分割として知られています。
領域を三角形に分割する方法は複数あるかもしれませんが、ここでは、最小（三角形）角度を最大化し、４つの最も近いポイントが同じ円上にない限り一意であるドロネー三角形分割に焦点を当てます。



線形近似と元のデータ

3. この例では、ランダムな集合 xwave と ywave がガウス関数からのサンプリングを表すと仮定します。

コマンドウィンドウで次を実行します。

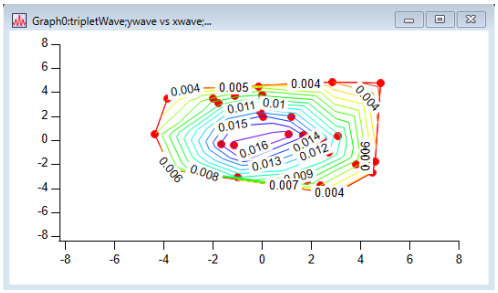
```
Make/O/N=30 zwave                          // 第３のウェーブを作成
zwave=gauss(xwave[p],0,3,ywave[p],0,3)  // ガウスサンプルで埋める
```

以降のコマンドの中には、トリプレットのウェーブ入力が必要なものがあるため、次のコマンドを使ってデータからトリプレットのウェーブを作成します。

```
Concatenate {xwave,ywave,zwave},tripletWave
```

4. 以下のコマンドを使って、データの三角分割を計算、表示できるようにします。

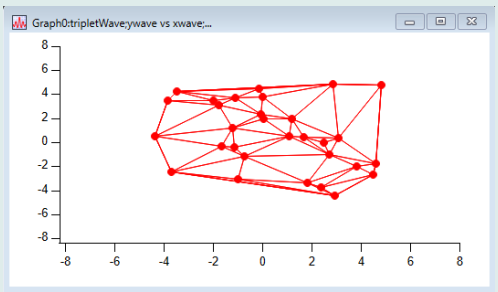
```
AppendXYZContour tripletWave
```



5. 続いて次を実行します。

```
ModifyContour tripletWave xymarkers=1,boundary=1,
                triangulation=1;DelayUpdate
ModifyContour tripletWave autoLevels={*,*,0}
ModifyContour tripletWave labels=0
```

三角分割された結果は右のように表示されます。

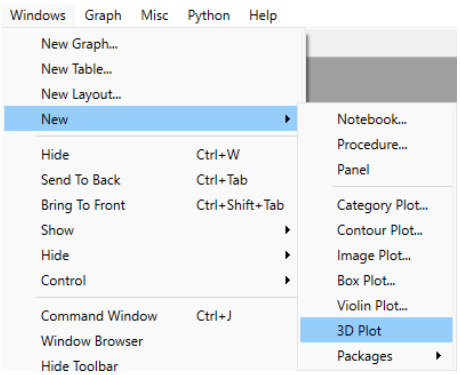


6. この三角分割を使えば、凸包内の任意のポイントが属する三角形を特定し、その頂点にある 3 つの z 値を線形補間することで、そのポイントの z 値のおおよその値を求めることができます。

コマンドウィンドウで次を実行して補間データを作成します。

```
imageinterpolate/cmsh voronoi tripletwave
```

メニュー Windows→New→3D Plot を選択します。



7. Gizmo Info パネルの Object List の下の「+」から、Surface を選択します。項目を次のように設定します。

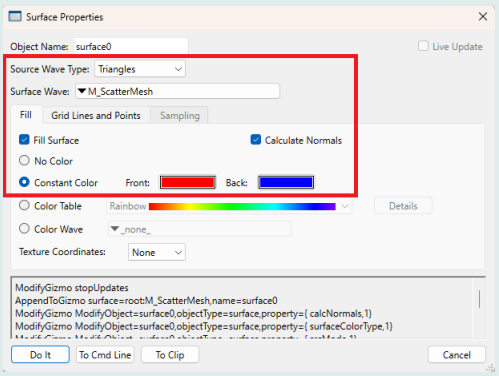
Source Wave Type : Triangles

Surface Wave : M_ScatterMesh

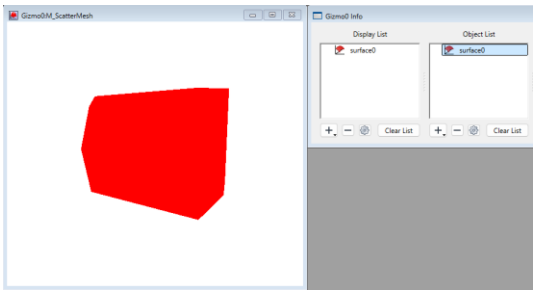
Fill Surface : チェック

Calculate Normals : チェック

Constant Color を選択

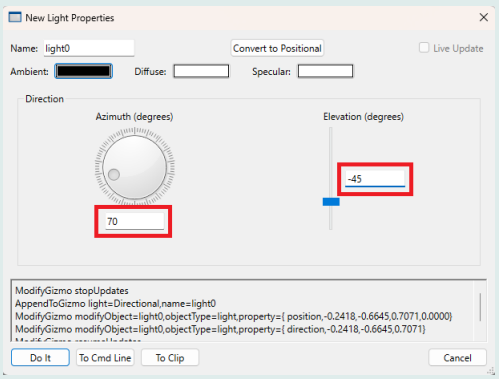


8. Do It をクリックすると、Object List に surface0 オブジェクトが追加されます。
surface0 をドラッグして Display List に追加すると、右のように Gizmo ウィンドウに表示されます。
（3D であるため、回転して確認できます）

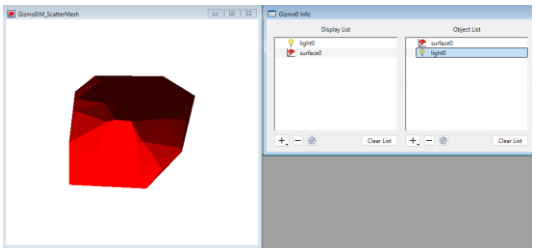


9. Object List の「+」メニューから、Light を選択します。

Azimuth を「70」、Elevation を「-45」に設定して、Do It をクリックします。



10. Object List から Display List の最上位に light0 をドラッグします。
指定された方向からの光が当たります。



線形近似の 2 次元における対応物

11. ボロノイ（Voronoi）補間を使うことで、より滑らかな曲面表現を得ることができます。

コマンドウィンドウで次を実行します。

```
ImageInterpolate/CMSH/RESL={300,300} voronoi
        tripletWave
AppendToGizmo surface=root:M_InterpolatedImage,
        name=surface1
```

surface1 のプロパティを次のように設定します。

Source Wave Type :　　Matrix

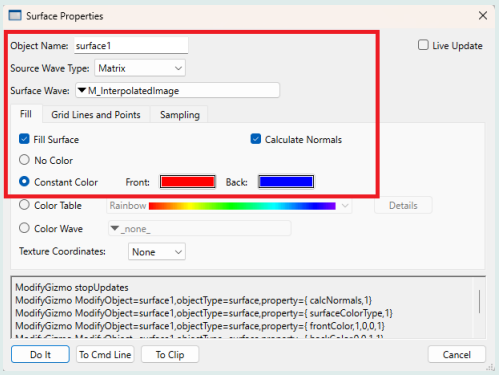
Surface Wave :　　M_InterpolateImage

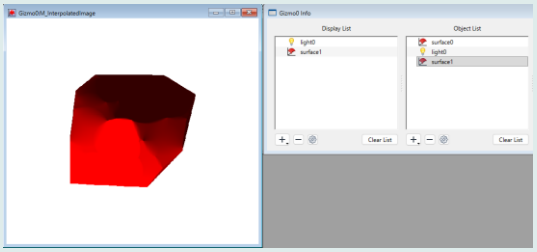
Fill Surface :　　チェック

Calculate Normals :　　チェック

Constant Color を選択

RESL={300,300} は画像の解像度を示します。





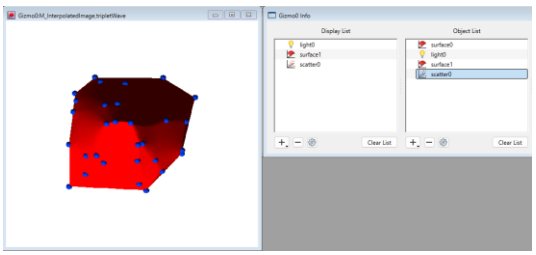
ボロノイ補間されたサーフェス

12. サーフェスに元のポイントの散布図を追加します。
Object List の「+」で Scatter を選択し、Scatter Wave には tripletWave を選択します。
Color タブで青色を選択し、Shape and Size タブで、形状は Sphere、サイズは Fixed 0.5 を指定します。

Display List に scatter0 を追加します。

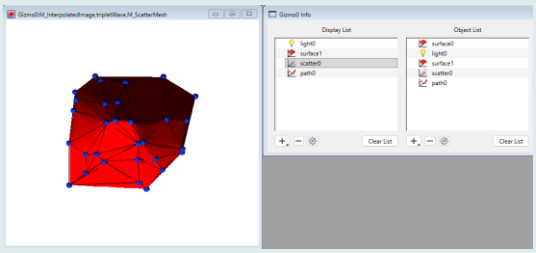
このサーフェスは滑らかになっていますが、右に示すように、元のデータポイントの Z 値はすべて実際には変更されていません。

青いマーカーは元のデータの位置を示しています。補間された曲面は、元のデータポイントをすべて明確に通過しています。



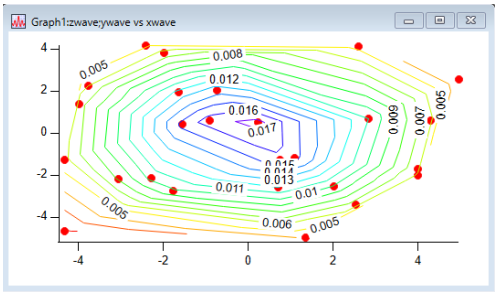
13. サーフェイスに三角形を追加すると右のようになります。

```
AppendToGizmo path=root:M_ScatterMesh,name=path0
```



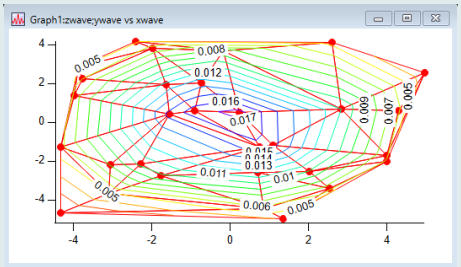
14. 同じデータを画像プロットとして表示することもできます。
コマンドウィンドウで次を実行します。

```
display ywave vs xwave
ModifyGraph mode(ywave)=3,marker(ywave)=19
AppendXYZContour zwave vs {xwave,ywave}
```



15. 基礎となる三角分割を併せて示します。

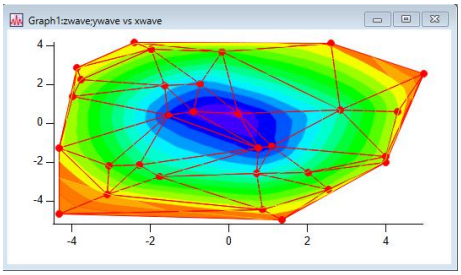
```
ModifyContour zwave triangulation=1
```



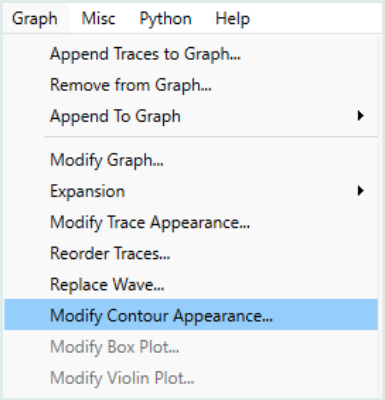
16. コンターを色で塗りつぶします。

```
ModifyContour zwave fill=1
ModifyContour zwave interpolate=2
ModifyContour zwave labels=0
```

合わせてコンターのラベルも消しておきます。



17. 定値の等高線を比較することで、ボロノイ補間の効果を確認することもできます。
補間を変更するには、Graph メニューの Modify Contour Appearance を選択します。



18. Interpolation の項で、補間の方法を変更できるので、設定を変えて変化を確認してみてください。