Date: Fri, 30 May 2008 13:47:40 +0900
From: Tsukasa NAKANO
To: Junji Torii
Cc: Shunji KASAMA
Subject: ovoid_fitting

とりいさま、

産総研の中野司です。旧来の slice シリーズのうちでとりあえず改造したものはこれで最後です（残りのものの改造は研究室の引っ越しが終わってから）。今回は２および３次元画像上の物体像の形状を楕円もしくは楕円体近似するプログラム群の改造版を紹介します。旧版と新版（２次元画像用の t_*と３次元用の si_*があります）の対応関係は以下の通りです：

旧版	新版	機能
sliceOF2	t_of	２次元像の楕円近似を行う
sliceOF2.oblate	t_of_oblate	〃（楕円軸の定義が異なる）
sliceOF	si_of	３次元像の楕円体近似を行う
sliceOF.oblate	si_of_oblate	〃（楕円体軸の定義が異なる）
なし	t_ofwd	重みつき楕円近似を行う
なし	t_ofwd_oblate	〃（楕円軸の定義が異なる）
of_md	si_ofwd	重みつき楕円体近似を行う
of_md.oblate	si_ofwd_oblate	〃（楕円体軸の定義が異なる）
なし	t_op	楕円を描く
sliceO	si_op	楕円体を描く

新旧のプログラム群の違いは以下の通りです：

[1] 新版は旧版より少しだけ高速に処理を行うことができる。
[2] 新版は Windows の DOS 窓でも実行することができる。
[3] 新版には旧版になかった２次元画像専用のものがある。
[4] 新版の起動法や出力は旧版のものと少しだけ異なっている。

下記の書庫ファイルに新版のプログラム群のソースファイルと Windows 用の実行ファイルなどを入れておきましたので、どうぞご利用下さい：

http://www-bl20.spring8.or.jp/~sp8ct/tmp/ovoid.taz
http://www-bl20.spring8.or.jp/~sp8ct/tmp/ovoid.zip

新版のプログラムの説明を以下に書きます。

(1) 物体像の楕円もしくは楕円体近似

ここで紹介する物体像の楕円（２次元像の場合）もしくは楕円体（３次元像）近似の詳細は前記の書庫ファイル中の PDF ファイル ovoid.pdf に記されています：

http://www-bl20.spring8.or.jp/~sp8ct/tmp/ovoid.pdf

ただし、その文書の説明はぼくがこの方法を思いついた時の「そもそも」の方法論に後付けした理屈を加えたものなので、わかりにくいかもしれません。ここでは「そもそも」の方について少し説明します。

注１：後述するプログラムの説明で用いている楕円もしくは楕円体のパラメータは ovoid.pdf の中で使っているものと同じです。これらの定義は ovoid.pdf をご覧下さい。

注２：ぼくは ovoid.pdf の最初の版を 10 年くらい前に書きました。そして、その内容を池田進さんが書いた以下の論文（前にも紹介したような気がします）に付録として載せてもらいました：

http://www-bl20.spring8.or.jp/~sp8ct/tmp/ikeda_MinMag.pdf

ところが、最新版の ovoid.pdf にも書いてあるように、この付録（初版の ovoid.pdf）に記した計算式にちょっとした間違いがありました。そういう訳なので、ぼくの楕円体近似の説明にこの論文を引用する場合は、その付録に書いてある計算式の間違いに言及していただけるとうれしいです。

さて、楕円もしくは楕円体近似の「そもそも」の方法論ですが、これは測定データの「最小自乗法（Least Square Method ; LSM）」による直線近似をもとにした手法です。この E-mail に添付したファイル lsm.gif をご覧下さい。その左図に例示したように、通常の LSM では位置 x = xi で測定した値 y = yi（ただし、i = 1 ～N は測定データを区別する添字）にあてはめる直線 y = g×x + c の係数値 g と c を測定値とその近似値の差 di = yi - (g×xi + c) の自乗和 R = Σi di^2 が最小になるように決めます。

今の例の場合、具体的には ∂R/∂g = 0 と ∂R/∂c = 0 から得られる連立方程式（「正規方程式」）を解くことにより係数値 g と c を算出できます。

このように、通常の LSM ではデータの位置 xi の誤差を考慮せずに、測定値 yi のバラツキだけをならすように直線を決めています。これに対してデータが２次元の点の座標値(xi, yi) の場合には、それらの分布（配列）を近似する直線 u×x + v×y + w = 0 として、各点からおろした垂線の足の長さ Di の自乗和 M = Σi Di^2 が最小になる係数値のものを使うのが

合理的だと考えました（添付したファイル lsm.gif の右図)。

今の例の場合、Di を表す式は具体的には以下の通りです：

$$Di = | u \times xi + v \times yi + w |$$

そして、$\partial M/\partial w = 0$ より以下の式が得られます：

$$\Sigma i (u \times xi + v \times yi + w) = 0 \rightarrow u \times \Sigma i\, xi + v \times \Sigma i\, yi + w \times \Sigma i\, 1 = 0$$

ここで、$\Sigma i\, 1 = N$（点の総数）です。また、点群の重心を

$$(x0, y0) = \Sigma i\, (xi, yi) / N$$

と書くと、得られた式は

$$u \times x0 + v \times y0 + w = 0$$

となります。つまり、求める直線は点群の重心を通ります。この結論は３次元の場合も成り立ちます。

ぼくの楕円（もしくは楕円体）近似ではこのような離散的な点群ではなく画像上の物体像が占める領域を取り扱っていますが、いずれにせよ、それに関して計算した M の値を最小にするように直線をあてはめることが基本です。そして、この M の値は物理的には「物体像のモーメント（正確には、単位密度の物体像のある回転軸の回りの慣性モーメント)」と解釈できますが、それは後付けの理屈です。

M を「モーメント」と呼ぶのは説明に便利だということ以外にもうひとつ別の理由があります（こちらの方がより深刻な理由です)。通常の LSM による直線近似では M に相当する R の値は点群に対する「直線のあてはめの良否」を判定するのに使われます。ところが、ぼくの方法では M の値を楕円もしくは楕円体の軸半径に換算して使います。つまり、統計学の観点から言うと、評価に使うべき値をそれ以外の用途に流用しています。さらに、ぼくの方法には「楕円や楕円体のあてはめの良否」の判定基準になる量がありません。それゆえ、この方法は統計学的には不完全で、物理的なモデルを導入することでそれをごまかしています。

このような訳で、ぼくが考案した楕円もしくは楕円体近似は本質的には画像上の物体像の「軸方向」を決める手法です。正直に言うと、楕円や楕円体の軸半径は「オマケ」で推定した値です。もしくは、このような軸半径は軸方向の「推定精度」を表す値と言った方が良いかもしれません。

なお、ついでながら、物理学で使うモーメントは「点の質量」や「物体像の密度」を加味した値です。LSM の計算ではこれらは「重み」に相当します。つまり、それぞれのデータ

の測定誤差の逆数に相当する値を wi として、R = Σi (wi×di^2) を最小にするのが「重みつき最小自乗法」です。これと同様に物体像の密度を繰り込んだモーメント M を用いて楕円や楕円体近似を行うこともできます。後述するプログラム*_ofwd*(”wd” は”with density” のつもり）では画像の画素値を物体像の密度と見なしてそれを行っています。

(2) 物体像の楕円もしくは楕円体近似のプログラムの説明

(2-1) 物体像の近似楕円体のパラメータの計算

t_of および t_of_oblate

起動法

t_of　TIFF　uX　uY

t_of_oblate　TIFF　uX　uY

機能

２次元画像 TIFF を読み込み、画素値ごとの領域を画素のつながりを考慮せずに個別の物体像と見なす（画素値０の画素群も領域として取り扱うことに注意)。uX と uY を x と y 方向それぞれの画素の辺長として、個々の像を近似する楕円のパラメータを計算する。ただし、t_of_oblate では近似楕円の軸半径 A と B の定義を t_of のものと逆にした計算を行う：

	t_of	t_of_oblate
A	短軸	長軸半径
B	長軸	短軸半径

最終的には、画素値ごとの以下の７個の値をタブコード区切りで１行にまとめて標準出力に書き出す：

[1] 画素値

[2] その画素値の領域に属する画素数

[3,4] 近似楕円の中心（== 物体像の重心）の x および y 座標値

[5] 楕円の軸の方向を指す角度 θ （単位は度）

[6,7] 楕円の A および B 軸半径

なお、これらの出力において、楕円の軸半径はもちろんのことそれらの中心の座標も起動パラメータ uX と uY で指定した単 位の値になっている。

si_of および si_of_oblate

起動法

si_of　directory　nameFile　uX　uY　uZ

si_of_oblate　directory　nameFile　uX　uY　uZ

機能

ディレクトリ directory の下の nameFile で指定された３次元画像を読み込み、画素値ごとの領域を画素のつながりを考慮せずに個別の物体像と見なす（画素値０の画素群も領域として取り扱うことに注意)。uX、uY、uZ を x、y、z 方向それぞれの画素の辺長として、個々の像を近似する楕円体のパラメータを計算する。ただし、si_of_oblate では近似楕円体の軸半径 A、B、C の定義を si_of のものと逆にした計算を行う：

	si_of	si_of_oblate
A	短軸	長軸半径
B	中軸	中軸半径
C	長軸	短軸半径

最終的には、画素値ごとの以下の 11 個の値をタブコード区切りで１行にまとめて標準出力に書き出す：

[1] 画素値
[2] その画素値の領域に属する画素数
[3,4,5] 近似楕円体の中心（== 物体像の重心）の x、y および z 座標値
[6,7] 楕円体の C 軸方向を指す経度と緯度（単位は度）
[8] 楕円体の A もしくは B 軸の方向を指す角度 θ （単位は度）
[9,10,11] 楕円体の A、B および C 軸半径

なお、これらの出力において、楕円体の軸半径はもちろんのこと、それらの中心の座標もパラメータ uX、uY、uZ で指定した単位の値になっている。

(2-2) 画素値を「重み」とする楕円体近似

t_ofwd および t_ofwd_oblate

起動法

t_ofwd　TIFF　uX　uY

t_ofwd_oblate　TIFF　uX　uY

機能

２次元画像 TIFF の画素値を「密度」（画素値０は密度０）と見なして、画像全体を１個の楕円で近似する。パラメータ uX と uY の意味や２個のプログラム t_ofwd* の違いなどは t_of*の場合と同じ。最終的には以下の８個の値をタブコード区切りでまとめた２行を標準出力に書き出す：

1行目（3個の値を含む）

[1,2] もとの画像の x および y 方向の画素数

[3] もとの画像の「総質量（== 画素値の総和)」

2行目（5個の値を含む）

[1,2] 近似楕円の中心（== 物体像の重心）の x および y 座標値

[3] 楕円の軸の方向を指す角度 θ （単位は度）

[4,5] 楕円の A および B 軸半径

si_ofwd および si_ofwd_oblate

起動法

si_ofwd　directory　nameFile　uX　uY　uZ

si_ofwd_oblate　directory　nameFile　uX　uY　uZ

機能

ディレクトリ directory の下の nameFile で指定された3次元画像の画素値を「密度」（画素値0は密度0）と見なして、画像全体を1個の楕円体で近似する。パラメータ uX、uY、uZ の意味や2個のプログラム si_ofwd*の違いなどは si_of*の場合と同じ。最終的には以下の 13 個の値をタブコード区切りでまとめた2行を標準出力に書き出す：

1行目（4個の値を含む）

[1,2,3] もとの画像の x、y および z 方向の画素数

[4] もとの画像の「総質量（== 画素値の総和)」

2行目（9個の値を含む）

[1,2,3] 近似楕円体の中心（== 物体像の重心）の x、y および z 座標値

[4,5] 楕円体の C 軸方向を指す経度と緯度（単位は度）

[6] 楕円体の A もしくは B 軸の方向を指す角度 θ （単位は度）

[7,8,9] 楕円体の A、B および C 軸半径

これら2個の画素値を重みとした楕円もしくは楕円体近似プログラムは楕円もしくは楕円体近似よりも画像上の物体像の「中心」を決める際に有用です。例えば、物体像の縦断面のスライス画像を作成する場合、これらのプログラムで決めた重心を通るものにすれば、物体像の「要所（～高画素値の部分)」を含む断面画像になります。

(3) 楕円体を描くプログラム

t_op

起動法

t_op　orgTIFF　Ux　Uy　newTIFF

t_op　Nx　Ny　PV　Ux　Uy newTIFF

機能

このプログラムは以下の６個のパラメータの値を含む行（値の区切りは空白もしくはタブコード）を標準入力から読み込み、それらが表す楕円を x と y 方向の辺長がそれぞれ Ux と Uy の画素から構成される２次元画像の上に描く：

[1] 楕円の内部を塗りつぶすのに用いる画素値
[2,3] 楕円の中心の x および y 座標値
[4] 楕円の軸の方向を指す角度 θ （単位は度）
[5] 楕円の A および B 軸半径

１番目の方法で起動すると既存の２次元画像 orgTIFF の上に楕円を上書きする。また、２番目の方法では x と y 方向の画素数が Nx と Ny で画素値すべてが PV の画像の上に楕円を描く。いずれの場合も結果の画像を newTIFF に保存する。

si_op

起動法

si_op　orgDir　nameFile　Ux　Uy　Uz　newDir
si_op　Nx　Ny　Nz　PV　Ux　Uy　Uz　newDir

機能

このプログラムは以下の 10 個のパラメータの値を含む行（値の区切りは空白もしくはタブコード）を標準入力から読み込み、それらが表す楕円体を x、y、z 方向の辺長がそれぞれ Ux、Uy、Uz の画素から構成される３次元画像の上に描く：

[1] 楕円体の内部を塗りつぶすのに用いる画素値
[2,3,4] 楕円体の中心の x、y および z 座標値
[5,6] 楕円体の C 軸方向を指す経度と緯度（単位は度）
[7] 楕円体の A もしくは B 軸の方向を指す角度 θ （単位は度）
[8,9,10] 楕円体の A、B および C 軸半径

１番目の方法で起動すると、ディレクトリ orgDir の下にある nameFile で指定した既存の３次元画像の上に楕円体を上書きする。また、２番目の方法では x、y、z 方向の画素数が Nx、Ny、Nz ですべての画素値が PV の画像の上に楕円を描く。いずれの場合も結果の画像をディレクトリ newDir の下の z 座標値（０～）を名前とするスライス画像のファイルとして保存する。

添付ファイル lsm.gif