Buletinul Ştiinţific al Universităţii din Baia Mare
Seria B,Matematică-Informatică, vol.XIII(1997),189-198

A FAULT MODELING ALGORITHM

Virginia RUSU, Cristian RUSU

Abstract

Frequently the surfaces to be modeled have discontinuities. If the surfaces represent geological layer boundaries, the discontinuities mean faults. In all others cases the term "fault" is just a mathematical concept. A fault-modeling algorithm is described here. It was implemented in a general-purpose surface modeling software package (ZAZA). A fault-modeling sample is showed.

Keywords: surface modeling, gridding, fault-modeling algorithm, software implementation, faulted data sample.

Gridding with discontinuities

The **surface modeling** is a widespread problem in many fields: geology, topography, meteorology, health statistics, marketing, etc. Having a set of irregularly spaced samples of a surface (x, y, z) it must been estimate the surface z values at regularly spaced locations, by **gridding**. The surface samples are generally named **control points**. Gridded data facilitate any other subsequent processing.

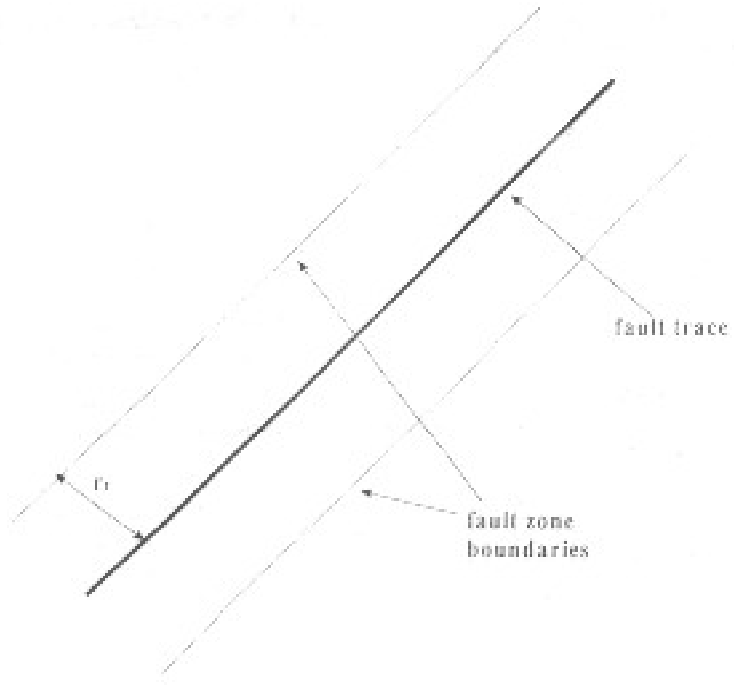
Frequently the surfaces have discontinuities. If the data represent geological layer boundaries, the discontinuities mean **faults**. In all others cases the term "fault" is just a mathematical concept.

The gridding process consists in surface z-value estimation at each node position of a regular grid, based on surrounding control points z-values. There are a lot of gridding algorithms but each of them takes into account the weighted control points z-values. Thus the significance of control points decrease with increasing distance from the grid node being interpolated.

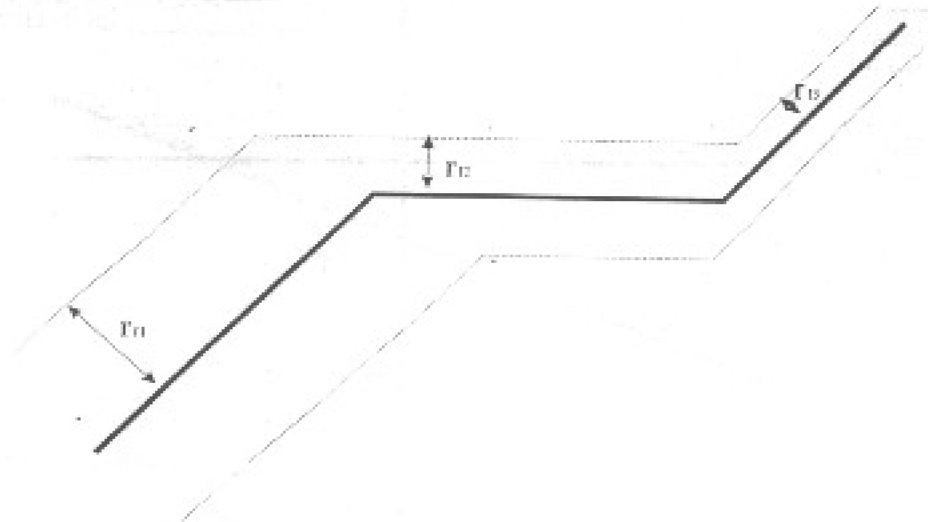
The majority of geological fault types can be modeled using a single set of surface z-values. However, there are faults that require two sets of surface z-values. We will discuss only the first situation, since it generally solves the fault-modeling problem.

A fault modeling algorithm

A fault trace consists of one or more line segments. The faults affect surfaces depending on their extent and the deformation produced. The fault extent is given by its trace. The surface deformation can be minor or major. A **major fault** produces global deformation of the surface. The surface shape on one side of the fault will be largely independent of the shape of the other side. Gridding on one side of the fault should not use data on the other side. It should be ignored regardless of its proximity to the grid node being interpolated. A **minor fault** produces small local changes in the surface shape. Gridding on one side of the fault should use data on both side, but data on the opposite side of the fault must have a smaller contribution.



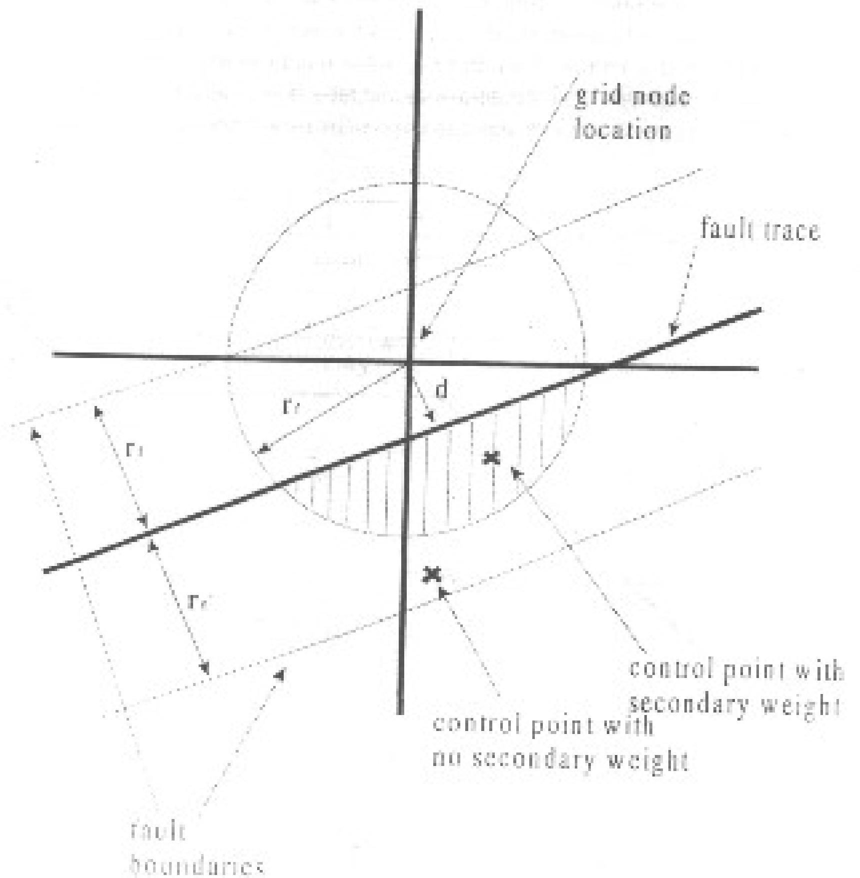
The affects of a fault are limited to a zone that parallels the fault trace. Inside the fault zone measurable changes in surface's elevation, slope, etc. are present. The fault zone thickness (r_f) directly influences the surface deformation. The user must choose a small r_f value in order to model a minor fault, and a large r_f value in order to model a major fault. An infinite r_f value would assure a total independence accros the fault, regardless of the data area extents. If the fault trace consists of more then one line segments each of them can have their own associated r_f values.



If the perpendicular distance d from the interpolated node location to a fault trace is less then the fault zone thickness r_f there might be control points where dissimilarities in surface character are assumed to be due to the fault mechanics and not to the distance. Such points are located in the hatched zone.

Secondary weights are computed for control points within hatched zone. These control points are used in the interpolation process with smaller contribution, due to the secondary weight.

The secondary weight is computed as a value of a smooth function wich increases from zero to unity as the distance from the interpolated grid node to the control point location increases from zero to r_f.



When a control point falls with hatched zone corresponding to several different fault traces, then the smallest secondary weight is computed and used to form the combined weight. The gridding processing time increases with the number of line segments used to define the fault traces and the relationship between fault placement and data distribution and density.

The algorithm implementation

Step by step we designed and developed a whole software package (ZAZA) that solves all surface modeling aspects. The last version is fully integrated, object-based, using Borland Pascal 7.0 and TurboVision objects library. The user-friendly interface and the contextual help have been built using messages in Romanian. These make it easier to be used. An English version is also available. The present version is an MS-DOS one, but a Windows one will be developed. The above mentioned algorithm was integrated in ZAZA.

ZAZA provides gridding, contouring, cross-sections, profiles, 3D-views, surface computations, areas, volumes and reserves' estimation. Users can control the working parameters in all phases of data processing. All input/output files format are Surfer compatible (Golden Software Inc.).

Among classical surface modeling techniques (gridding and triangulation) we used the first one, by at least two reasons: 1) gridded data facilitate any other subsequent processing, and 2) triangulation is more adequate for surfaces which extreme points (edges, valleys) are well known (e.g. topography). The program particularly implements local gridding techniques (weighted average, local polynomial surface fitting) but also trend surface analysis with polynomials. Other gridding methods can be further implemented.

Gridding can be made both with and without discontinuities. User can specify restricted areas or faults. Faulted data will be gridded depending on the extent of the faults and surface deformation.

Contouring is based on a continuous tracking algorithm that uses both the basic grid cells and intermediate refined grid values. The contouring process is fully interactive. Any combination of contour levels, surfaces, control points, faults and restricted areas are allowed.

Since ZAZA provides a large set of calculation on gridded data, users can easily manipulate maps made by gridding. Using gridded data can also make area and volume approximation. The volume approximation is at the base of reserves' estimation. ZAZA has been used both in commercial and university fields, with significant results. It is a general-purpose surface modeling software package, therefore it has been used not only in geologic data processing but also in health statistics.

We defined new object types, in order to store control points (*TPunctControl*), control points used in the interpolation process for a grid node (*TPunctInterpolare*), fault line segments (*TFalie*), etc.

```
PPunctControl = ^TPunctControl;
 TPunctControl = object(TObject)
  constructor Init(Noumume:string; Noux,Nouy,Nouz:real);
  destructor Done; virtual;
  private
   name:^str15;
   x,y,z:^real;
  end;

PPunctInterpolare = ^TPunctInterpolare;
TPunctInterpolare = object(TObject)
  constructor Init(Noux,Nouy,Nouz,NouDistanta,NouPondereSuplim:real;
   NouSector:byte);
  destructor Done; virtual;
  private
   x,y,z,distanta,PondereSuplim: ^real;
   sector: ^byte;
  end;

PFalie = ^TFalie;
 TFalie = object(TObject)
  constructor Init(Noux1,Nouy1,Noux2,Nouy2:real);
  destructor Done; virtual;
  private
   x1Falie,y1Falie,x2Falie,y2Falie: ^real;
  end;
```

The procedure *TestFalie* tests the surrounding fault traces of the interpolated grid node to establish on which side of them is situated a control point. The variable *InflFalie* stores the r_f parameter. *PondereSuplimentara* stores the secondary computed weight. *Distanta* stores the distance from the interpolated grid node to the control point location.

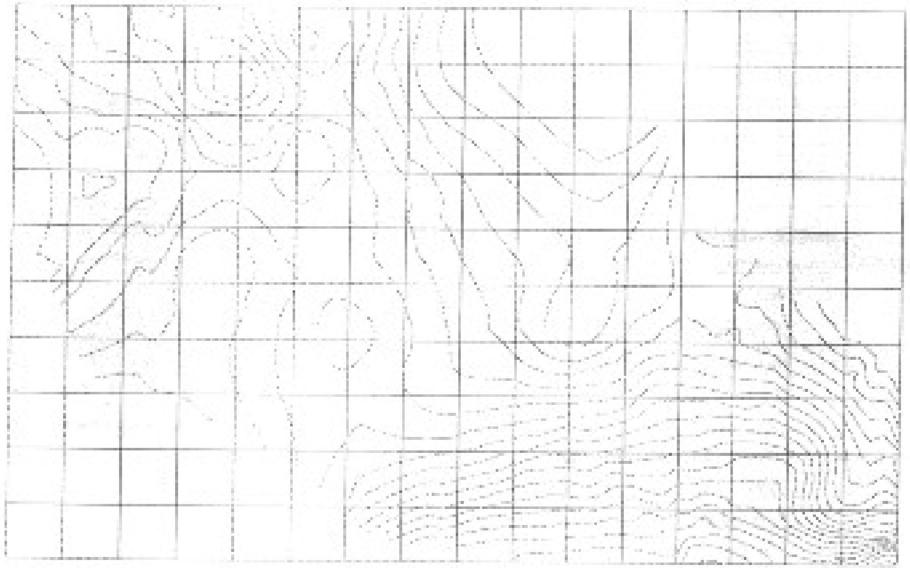
```
procedure TestFalie(PP:PFalie); far;
var a1,b1,c1,a2,b2,c2,xi,yi,xxFalie,yyFalie:real;
begin
 with PP^ do
 begin
```

```
    if xx=0 then xx:=2.9e-10
    xxFalie:=x2Falie^-x1Falie^;
    yyFalie:=y2Falie^-y1Falie^;
    if xxFalie=0 then xxFalie:=2.9e-10;
    a1:=yy/xx;
    b1:=-1;
    c1:=ynod-xnod*yy/xx;
    a2:=yyFalie/xxFalie;
    b2:=-1;
    c2:=y1Falie^-x1Falie^*yyFalie/xxFalie;
    if a1*b2<>a2*b1 then
    begin
     xi:=(b1*c2-b2*c1)/(a1*b2-a2*b1);
     yi:=(c1*a2-c2*a1)/(a1*b2-a2*b1);
     if (yi>=Minim(y1Falie^,y2Falie^)) and (yi<=Maxim(y1Falie^,y2Falie^)) and
      (xi>=Minim(x1Falie^,x2Falie^)) and (xi<=Maxim(x1Falie^,x2Falie^)) and
      (yi>=Minim(ypc,ynod)) and (yi<=Maxim(ypc,ynod)) and
      (xi>=Minim(xpc,xnod)) and (xi<=Maxim(xpc,xnod)) and
      (distanta=sqrt(sqr(xi-xnod)+sqr(yi-ynod))) then
      begin
       if abs((a2*xnod+b2*ynod+c2)/sqrt(sqr(a2)+sqr(b2))) < (xpas+ypas)/4 then
       znod:=10e30;
      if abs((a2*xnod+b2*ynod+c2)/sqrt(sqr(a2)+sqr(b2))) < InflFalie then
       if distanta>InflFalie then
        if InflFalie<>10e30
         then PondereSuplimentara:=0
        else
         if distanta/InflFalie>PondereSuplimentara
          then PondereSuplimentara:=distanta/InflFalie;
     end;
   end;
  end;
end;
```

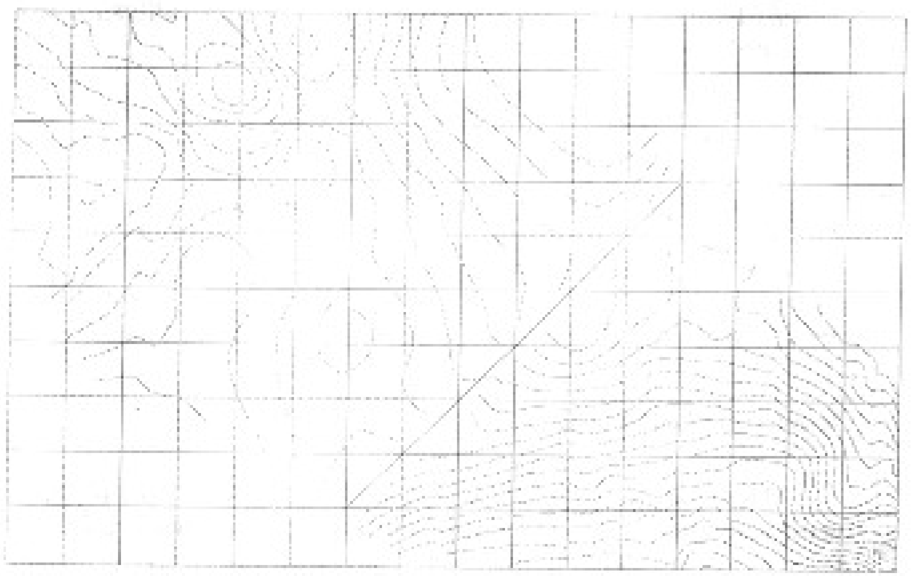
A sample of faulted data

Further on an example of surface modeling is given. The data represent the upper limit of the middle Miocen volcanoclastic deposits from an area of Northern group of Eastern Carpathian Mountains. The control points represent drillings

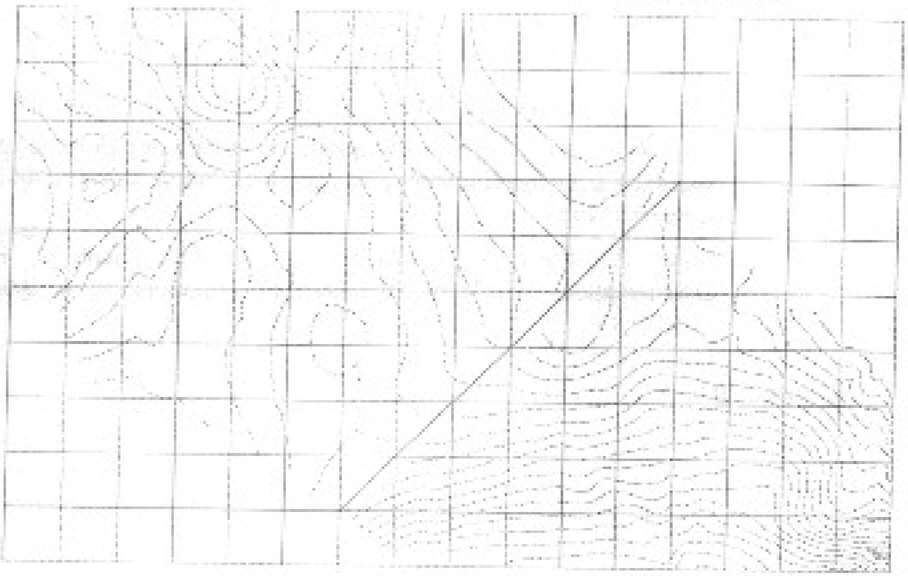
The first image represents the above mentioned surface modeled without faults



The second image represents the surface modeled with a minor fault.



The third image represents the surface modeled with a major fault.



References

1. *** Radian Corporation: CPS-1 User's Manual, Radian Corporation, 1979
2. Cosma M. a.o.: Sistem Informatic Geomar, Institutul de cercetare ştiinţifică şi inginerie tehnologică pentru tehnică de calcul şi informatică, Bucureşti, 1988
3. Rusu C., Rusu V.: Surfaces Modeling for Geologic Reserves Determination, The Third International Symposium (IGS), Baia Mare, October 1993

Received 10.05.1997

North University of Baia Mare
Department of Mathematics and Computer Science
Victoriei, 76
RO-4800
Romania
e-mail: crusu@univer.ubm.ro