URL.

http://www.scilab.org	–	ресурс SciLab.
http://www.astonshell.com/rus	–	ресурс хорошего текстового редактора Bred.
http://www.openoffice.org/	–	ресурс OpenOffice.org.
http://www.foxitsoftware.com/	–	ресурс Foxit Software, содержит Foxit Reader, для просмотра и печати PDF.
http://www.cs.wisc.edu/~ghost/	–	ресурс Ghostscript, системы для просмотра, конвертирования и печати EPS, EPSI, PS, PDF.
http://www.ghostgum.com.au/	–	ресурс Ghostgum Software, содержит GSView, позволяющий работать с системой Ghostscript в визуальном режиме.
http://windjview.sourceforge.net	–	ресурс WinDjView, программы для просмотра DjVu.

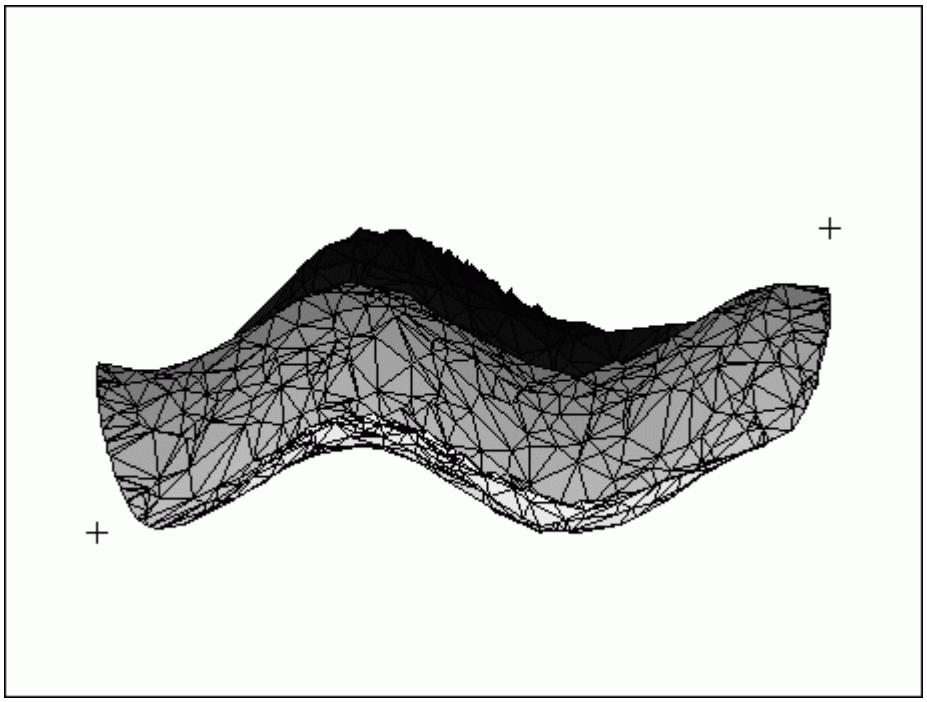
«И не будет тьмы после нас...»

А.Н.Каретин.

04.11.2008г.

А.Н.Каретин.

Применение пакета матричных вычислений SciLab 4.1.2 для решения геодезических задач по наблюдению за поверхностями.



Typeset by OpenOffice.org Write

2008г.

Введение.

Все программные продукты, разработанные для обработки результатов наблюдений за поверхностями, являются дорогостоящими и закрытыми. Помимо этого, данные программные продукты не предоставляют необходимой деталировки результата, а в большинстве случаев данные программные продукты не позволяют решать простые с геометрической точки зрения задачи. То есть, эти продукты решают определенные задачи, но не позволяют, в силу непрозрачности данных продуктов, варьировать элементы этих задач. Что в свою очередь сужает производительность данных программных продуктов, а в определенных случаях делает их просто неэффективными. Это относится в первую очередь к такому продукту как CREDO_TER. Этот программный продукт выдает настолько утрированный результат, что невозможно убедиться в его надежности или дорабатывать данный результат.

В данной методичке будет показан не только уровень технологий, распространяемых по лицензии Open Source, но и их невероятную прозрачность и полноту представляемого результата.

Автор данной методики желает представить читателям программный продукт, распространяющийся по лицензии Open Source, и именуемый SciLab 4.1.2. Данный программный продукт является системой матричных вычислений с очень хорошо проработанным векторизатором числовых последовательностей (данные векторизаторы являются основой всех матричных систем вычислений). Автор данной методики очень сожалеет, что дальнейшее развитие данного продукта пошло по пути большей совместимости с MatLab & Matlab toolbox, вместо наращивания своих собственных возможностей. Подобный выбор явно обозначен какими-то целями, и я не могу осуждать этот выбор не зная целей, но то, что данный выбор идет в ущерб концепции SciLab, очевидно. Простой пример, ежели мне предложат выполнить какие либо матричные вычисления (именно матричные вычисления, а не имитационное моделирование), и будут предложены MatLab и SciLab для выполнения, я конечно выберу SciLab. Не из-за каких-то левых симпатий, разработчику не свойственны симпатии, он берет то, от чего ожидает максимальный результат. Автор данной инструкции настоятельно рекомендует обождать с ознакомлением с более поздними версиями SciLab, так как они не внесли никаких реализаций в структуру матричных вычислений, а были направлены исключительно на развитие совместимости с MatLab toolbox. Большая погрешность с моей точки зрения. Разработка же производных векторизаторов естественным образом полостью ликвидировала бы совместимость MatLab и SciLab, но результирующая прозрачность работы SciLab просто бы сделала нонсенс из системы MatLab. При этом всего лишь на время! Мульти-векторизаторы на базе целочисленной, множественной и мультипликативной основах, естественно имеют очень сложное теоретическое продолжение, но на практическом уровне имеют колоссальные возможности. Возможности линеаризации задач при этом имеют лишь локальные ограничения. Линеаризация же задач при этом происходит автоматически. Почему они не пошли по этому пути? Для меня, разработчика, это очень непонятно. Надеюсь, я ошибаюсь!

«И не будет тьмы после нас...»

А.Н.Каретин.

04.11.2008г.

ДЛЯ ЗАМЕТОК.

```
 plot2d(dltl,zl,[-1,-2,3]);
 taxes=get("current_axes");
 taxes.axes_visible="off";
 xfpolys(dlts,zps,[1:n]);
endfunction

function []=triplot3dgray(Tri,X,Y,Z,gmm,tt,kz)
 [lhs,rhs]=argn(0);
 if rhs<7, gmm=0; tt=0.3; kz=1; end;
 [n,dltl,zl,dlts,zps]=triplot3dbase(Tri,X,Y,Z,gmm,tt,kz);
 xbasc();
 tgraph=gca();tgraph.isoview='on';
 xset("colormap",graycolormap(n));
 plot2d(dltl,zl,[-1,-2,3]);
 taxes=get("current_axes");
 taxes.axes_visible="off";
 xfpolys(dlts,zps,[1:n]);
endfunction
```

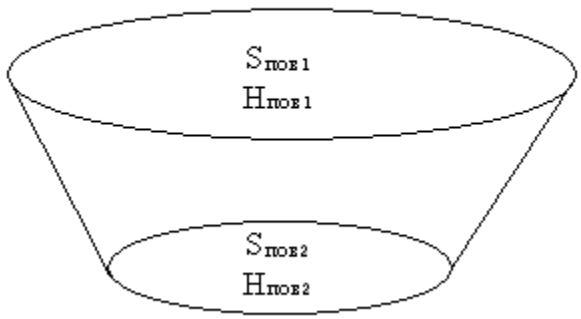
Теоретические основы задач обработки наблюдений за поверхностями.

Задачи обработки результатов наблюдений за поверхностями ставят своей целью получение для любых поверхностей, как бы сложны они не были, определенных макро-характеристик. К таким характеристикам относится уровень поверхности ($H_{пов}$), уровенная площадь поверхности ($S_{пов}$), элементы среднего уклона поверхности [$i_{пов}$, $a(i_{пов})$], искривление поверхности ($j_{пов}$), максимальный уклон поверхности (i_{max}).

Перечисленные выше характеристики поверхностей необходимы только для масштабных задач, относящихся к поверхностям в целом. Эти характеристики оказывают влияние на все задачи меньшего масштаба.

Наиболее часто данные характеристики требуются для расчета объема грунта ($V_{гр}$), заключенного между двумя поверхностями. Поверхности при этом обычно представлены разными наборами точек, наблюдаемыми в разные периоды времени.

Схема расчета в этом случае выглядит так.



$$V_{гр} = \frac{1}{3}\left(S_{пов1} + S_{пов2} + \sqrt{S_{пов1} \times S_{пов2}}\right)\left(H_{пов1} - H_{пов2}\right)$$

Как видно имея макро-характеристики поверхностей задача расчета объема грунта тут же решается одной простой формулой и не является сложной.

Сложным здесь является получение макро-характеристик для сложных поверхностей. И этому будет посвящена следующая глава.

Получение макро-характеристик поверхностей (Sпов, Hпов).

Поверхность, заданная наборами точек, может быть также представлена набором непересекающихся треугольников. Построение данного набора треугольников называется триангуляцией. Распространенным и повсеместно применяемым способом триангуляции является триангуляция Делоне. Эта триангуляция соблюдает набор определенных геометрических правил и хорошо отражает триангулируемую поверхность.

После того, как произведена триангуляция (точки объедены в треугольники), для каждого треугольника можно найти площадь и уровень его центра:

$$S_{тр} = \frac{1}{2}\left|(x_1 + x_2)(y_1 - y_2) + (x_2 + x_3)(y_2 - y_3) + (x_3 + x_1)(y_3 - y_1)\right|$$

$$H_{тр} = \frac{1}{3}(H_1 + H_2 + H_3)$$

Получив площади и уровни всех треугольников, можно получить площадь и уровень всей поверхности:

$$S_{пов} = \sum S_{тр}$$

$$H_{пов} = \frac{\sum (S_{тр} \times H_{тр})}{S_{пов}}$$

Если произвести подобные расчеты для двух поверхностей, относящихся к одному объекту, можно спокойно решить задачу поставленную в предыдущей главе.

```
 [X,Y,n,minz,maxz,xps,yps]=triplotbase(Tri,X,Y,Z);
 xbasc();
 a=gca();a.isoview='on';
 xset("colormap",graycolormap(n));
 colorbar(minz,maxz);
 plot2d(X,Y,[-1,-2,3]);
 xfpolys(xps,yps,[1:n]);
endfunction

function [pk,dlt]=pkdelta(X,Y,X0,Y0,sina,cosa)
 pk=((Y-Y0).*sina+(X-X0).*cosa);
 dlt=((Y-Y0).*cosa-(X-X0).*sina);
endfunction

function [n,dltl,zl,dlts,zps]=triplot3dbase(Tri,X,Y,Z,gmm,tt,kz)
 [m,n]=size(Tri);
 c=zeros(n,3);
 c(:,3)=1;
 [xp,yp,zp,ztri]=triparam(Tri,X,Y,Z);
 xmax=max(xp);
 xmin=min(xp);
 ymax=max(yp);
 ymin=min(yp);
 zmin=min(zp);
 zmax=max(zp);
 x0=(xmax+xmin)./2;
 y0=(ymax+ymin)./2
 dx=xmax-xmin;
 dy=ymax-ymin;
 L=sqrt(dx^2+dy^2);
 sinb=dy/L;
 cosb=dx/L;
 sina=sinb.*cos(gmm)+cosb.*sin(gmm);
 cosa=cosb.*cos(gmm)-sinb.*sin(gmm);
 stt=sin(tt);
 ctt=cos(tt);
 xp1=xp(1,:);
 xp2=xp(2,:);
 xp3=xp(3,:);
 yp1=yp(1,:);
 yp2=yp(2,:);
 yp3=yp(3,:);
 [pk,dlt]=pkdelta(X,Y,x0,y0,sina,cosa);
 [pk1,dlt1]=pkdelta(xp1,yp1,x0,y0,sina,cosa);
 [pk2,dlt2]=pkdelta(xp2,yp2,x0,y0,sina,cosa);
 [pk3,dlt3]=pkdelta(xp3,yp3,x0,y0,sina,cosa);
 pkt=(pk1+pk2+pk3)./3;
 [pkts,nt]=sort(pkt','r')';
 pks=[pk1(nt);pk2(nt);pk3(nt)];
 dlts=[dlt1(nt);dlt2(nt);dlt3(nt)];
 zps=zp(:,nt);
 dlts=dlts;
 zps=((zps-zmin).*ctt.*kz+pks.*stt);
 dltl=[min(dlts),max(dlts)];
 zl=[min(zps),max(zps)];
endfunction

function []=triplot3djet(Tri,X,Y,Z,gmm,tt,kz)
 [lhs,rhs]=argn(0);
 if rhs<7, gmm=0; tt=0.3; kz=1; end;
 [n,dltl,zl,dlts,zps]=triplot3dbase(Tri,X,Y,Z,gmm,tt,kz);
 xbasc();
 tgraph=gca();tgraph.isoview='on';
 xset("colormap",jetcolormap(n));
```

ПРИЛОЖЕНИЕ.

Листинги функций SciLab.

```
function [xp,yp,zp,ztri]=triparam(Tri,X,Y,Z)
 [m,n]=size(Tri);
 xp=matrix(X(Tri),m,n);
 yp=matrix(Y(Tri),m,n);
 zp=matrix(Z(Tri),m,n);
 ztri=sum(zp,1)/3;
endfunction

function [stri,ztri]=triarea(Tri,X,Y,Z)
 [m,n]=size(Tri);
 [xp,yp,zp,ztri]=triparam(Tri,X,Y,Z);
 x1=xp(1,:);
 x2=xp(2,:);
 x3=xp(3,:);
 y1=yp(1,:);
 y2=yp(2,:);
 y3=yp(3,:);
 stri=1/2*abs((x1+x2).*(y1-y2)+(x2+x3).*(y2-y3)+(x3+x1).*(y3-y1));
 ztri=ztri';
 stri=stri';
endfunction

function [sarea,zarea]=triareafill(stri,ztri)
 sarea=sum(stri,1);
 zarea=sum(stri.*ztri,1)/sarea;
endfunction

function [sarea,zarea]=triareafast(Tri,X,Y,Z)
 [stri,ztri]=triarea(Tri,X,Y,Z);
 [sarea,zarea]=triareafill(stri,ztri);
endfunction

function [X,Y,n,minz,maxz,xps,yps]=triplotbase(Tri,X,Y,Z)
 [m,n]=size(Tri);
 [xp,yp,zp,ztri]=triparam(Tri,X,Y,Z);
 [zts,nt]=sort(ztri','r')';
 xp1=xp(1,nt);
 xp2=xp(2,nt);
 xp3=xp(3,nt);
 yp1=yp(1,nt);
 yp2=yp(2,nt);
 yp3=yp(3,nt);
 xps=[xp1;xp2;xp3];
 yps=[yp1;yp2;yp3];
 maxz=max(ztri);
 minz=min(ztri);
endfunction

function []=triplotjet(Tri,X,Y,Z)
 [X,Y,n,minz,maxz,xps,yps]=triplotbase(Tri,X,Y,Z);
 xbasc();
 a=gca();a.isoview='on';
 xset("colormap",jetcolormap(n));
 colorbar(minz,maxz);
 plot2d(X,Y,[-1,-2,3]);
 xfpolys(xps,yps,[1:n]);
endfunction

function []=triplotgray(Tri,X,Y,Z)
```

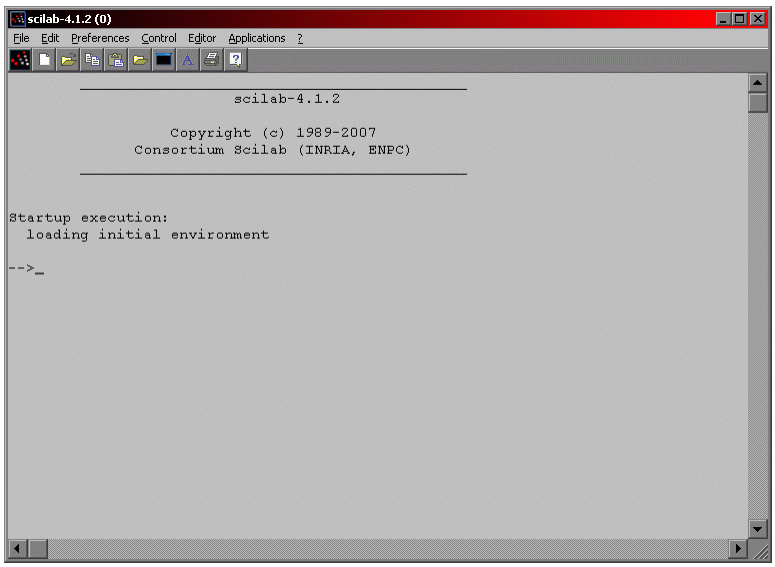
Применение пакета SciLab 4.1.2 для получения макро-параметров поверхностей.

Для получения макро-параметров поверхностей и их визуализации, автором данной методички был набран набор функций SciLab. Листинги данных функций представлены в приложении данной методички. В данной главе будет объяснено только их назначение.

Для работы со SciLab надо подготовить исходные данные. То есть сформировать текстовой файл, состоящий из четырех столбцов (N Yг Xг Hг), разделенных пробелами. Обращаю внимание на порядок записи Yг-Xг, а не Xг-Yг.Например:

```
1   1050.718   15170.314   106.256
2   1181.451   15158.645   98.934
…
999   1214.016   15192.955   104.422
1000   1206.050   15203.639   102.292
```

Перед началом работы надо набрать функции из приложения в текстовом редакторе, сохранить данный текст (чтобы не набирать его каждый раз). После этого можно запускать SciLab.



Чтобы задействовать функции, надо скопировать их из подготовленного текстового файла и вставит в окно SciLab. После этого их можно вызывать так же, как встроенные.

Теперь все готово к расчету. Считываем исходные данные:

```
--> NXYZ=read('d:\temp\test1.nxyz',1000,4);
```

Здесь `'d:\temp\test1.nxyz'` — путь к файлу исходных данных,
1000 — количество исходных точек,
4 — количество параметров для одной точки (N Yг Xг Hг).

Разбиваем этот массив на составляющие:

```
--> N=NXYZ(:,1)';
--> X=NXYZ(:,2)';
--> Y=NXYZ(:,3)';
--> Z=NXYZ(:,4)';
```

Производим триангуляцию поверхности:

`--> Tri=mesh2d(X,Y);` - для площади, ограниченной выпуклой оболочкой,

`--> Tri=mesh2d(X,Y,gr);` - для площади, ограниченной определенной границей.

Для второго случая лучше все точки границы в исходном файле разместить в начале файла, в

порядке обхода контура против часовой стрелки.
Тогда:

```
--> gr=[1:m];
```

где `m` – количество точек границы.

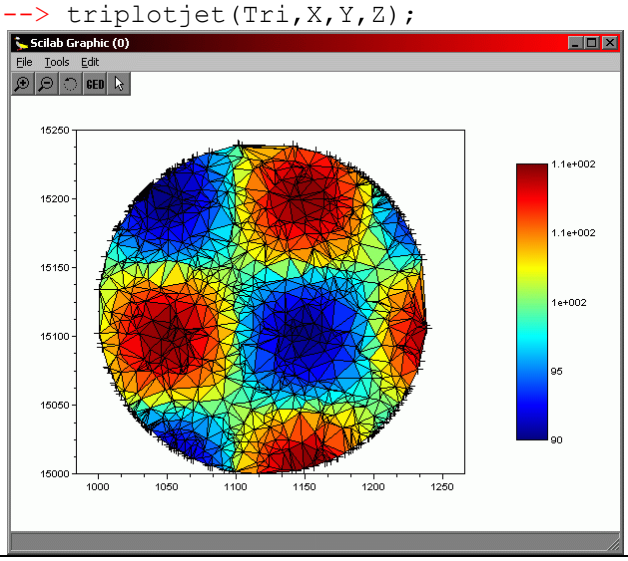
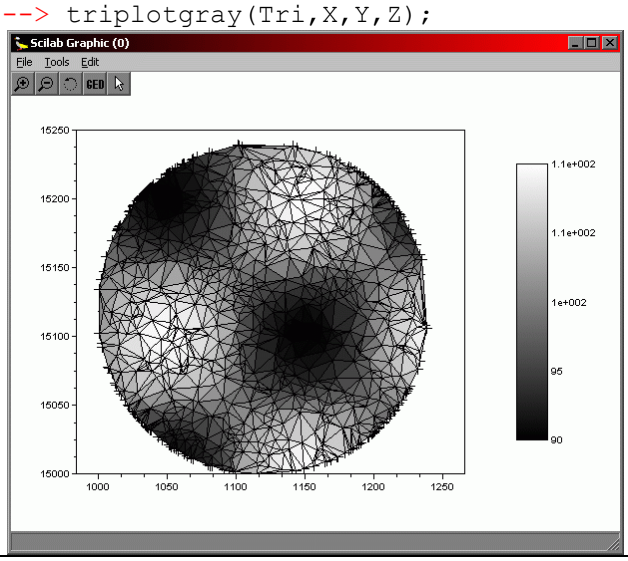
Произведя триангуляцию (если она конечно возможна), переходим к расчету уровня поверхности и ее площади:

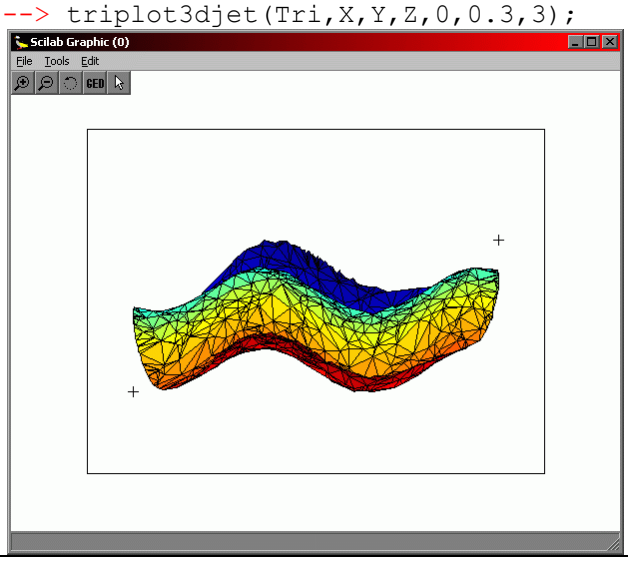
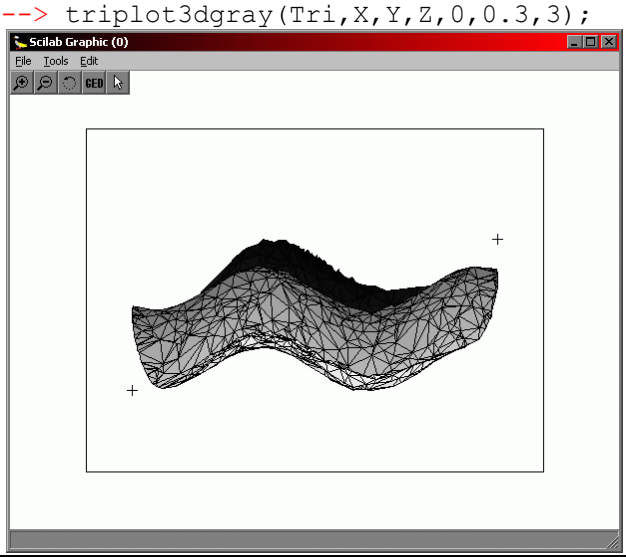
```
--> [sarea,zarea]=triareafast(Tri,X,Y,Z)
```

Эта функция выдаст сразу и площадь поверхности ($S_{пов}$=sarea) и ее уровень ($H_{пов}$=zarea). Для большей деталировки по треугольникам можно воспользоваться функциями, в нее входящими:

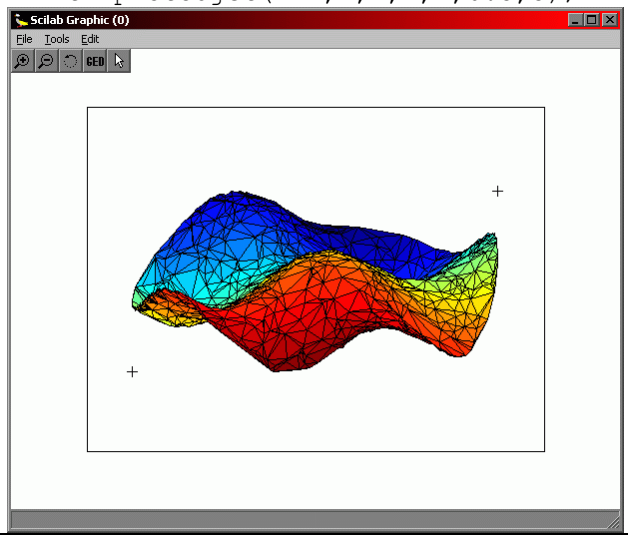
```
--> [stri,ztri]=triarea(Tri,X,Y,Z);
```

Но расчет, как бы ни надежно он был сделан, требует графического представления, как для отчета, так и для проверки. Здесь существует два вида представлений.

2D вид представления:	
для цветных принтеров	для черно-белых принтеров
`--> triplotjet(Tri,X,Y,Z);`	`--> triplotgray(Tri,X,Y,Z);`
	

3D вид представления:	
для цветных принтеров:	для черно-белых принтеров
`--> triplot3djet(Tri,X,Y,Z,0,0.3,3);`	`--> triplot3dgray(Tri,X,Y,Z,0,0.3,3);`
	

```
--> triplot3djet(Tri,X,Y,Z,2,0.3,3);
```



В этих функциях существуют три дополнительных параметра:
1) gamma – угол поворота в плоскости XY,
2) teta — угол поворота в вертикальной плоскости,
3) Kz – коэффициент вертикального масштабирования, для более наглядного представления результата.

Эти параметры необязательны. По умолчанию gamma=0, teta=0.3, Kz=1.

```
--> triplot3djet(Tri,X,Y,Z,1,0.7,3);
```