GEOPHYSICS, VOL. 66, NO. 2 (MARCH-APRIL 2001); P. 521–526, 1 FIG.

DOI:10.1190/1.1444942

New method for fast computation of gravity and magnetic anomalies from arbitrary polyhedra

Bijendra Singh* and D. Guptasarma*

ABSTRACT

We show that at any point the gravity field from a solid body bounded by plane surfaces and having uniform density can be computed as a field from a fictitious distribution of surface mass-density on the same body. The surface mass density at every surface element is equal to the product of the volume density of the body and the scalar product of (1) the unit outward vector normal to that surface element and (2) the position vector of the surface element with respect to the point of observation. Accordingly, the contribution to the gravity field from any plane surface of the body vanishes if the observation point lies in the plane of that surface. As a result, we can compute the gravity field everywhere, including points inside, on the surface, on an edge, or at a corner of the body where more than two surfaces meet.

This new result lets us compute the gravity field using exactly the same simple procedure as for the magnetic field of a uniformly magnetized object, computed from an equivalent surface distribution of magnetic pole density. To get the gravity field while computing the magnetic field, one simply uses the product of this surface mass density and the universal gravitational constant instead of the surface magnetic pole density. Therefore, the same computer program can be used to compute the gravity, the magnetic field, or both simultaneously. This simple and novel approach makes the numerical computations much faster than all other previously published schemes.

INTRODUCTION

A classical problem in gravity and magnetic exploration is computing theoretical anomalies caused by idealized models of assumed shapes. Many workers have published different methods for carrying out such computation, and textbooks on potential theory, e.g., Routh (1892), provide various formulas for these models. Early workers like Barton (1929) dealt with computing the gradients of the gravity field. Hubbert (1948) uses a line-integral approach to compute gravitational attraction of 2-D masses. Bhattacharyya (1964), Nagy (1966), and Plouff (1976) present closed-form analytical solutions for prism-shaped bodies, whereas Talwani and Ewing (1960) and Talwani (1965) use numerical integration techniques to compute the fields from models of arbitrary shape by dividing them into polygonal prisms or laminae. Barnett (1976) and Coggon (1976) provide different formulations for computing the gravity and magnetic fields from polyhedrons of arbitrary shape, density, and direction of magnetization. Okabe (1979) and Götze and Lahmeyer (1988) solve the problem by line integration along the edges of a polyhedral model instead of integrating over its faces. Pohánka (1988) also performs line integration to compute gravity fields from a polyhedron, elaborates on some problems met in numerical computations, and suggests procedures to avoid some of these problems. Although these formulations allow computation at all points of space (except at the corners of a polyhedron for the magnetic field), they generally require repeated transformation of the coordinate axes and extensive use of trigonometric functions.

More recently, Furness (1994) expresses the components of a magnetic field of homogeneously magnetized arbitrary polyhedra in terms of the magnetic scalar potential from (a) a uniform double layer over the polygon surface and (b) finite-length uniform pole density along the edges of the body. Holstein and Ketteridge (1996) use Stokes' theorem to reduce the surface integrals to line integrals and propose a combination of numerical and analytical procedures to compute a gravity field from homogeneous polyhedra.

Guptasarma and Singh (1999) show that a magnetic field from a uniformly magnetized polyhedral solid is the same as that from a surface distribution of magnetic pole density equal to the outward normal component of the intensity of magnetization. The field from such a polyhedron can be computed easily by converting the surface integral over each polygonal facet into a new line integral around its boundary. This approach makes the computation straightforward, avoids all complicated coordinate transformations, and allows the computation of the field at all points inside, on the surface, or outside the body

Published on Geophysics Online October 17, 2000. Manuscript received by the Editor February 23, 1999; revised manuscript received May 26, 2000.
*National Geophysical Research Institute, Uppal Road, Hyderabad 500 007, India. E-mail: ngrigravity@yahoo.com; puturani@hd1.vsnl.net.in.

except on the edge and at corners where three or more facets meet.

In this paper, we present a new technique for computing the gravity field of an arbitrary polyhedron of uniform density, based on the approach of our earlier method mentioned above. The computation can be carried out for all points of observation, including corners of the body. This extension is based on a new artifice that we have not seen in the literature on potential fields.

GRAVITY FIELD FROM A FINITE-SIZED BODY

The component of the gravity field vector **F** in any direction **a** can be written as the surface integral (Barnett, 1976; Coggon, 1976)

$$\mathbf{F}\cdot\mathbf{a} = -G\rho\iint (1/r)\,\mathbf{a}\cdot\mathbf{u_n}\,ds, \tag{1}$$

where G is the universal gravitational constant, ρ is the uniform volume density of the body, r is the distance $(x^2+y^2+z^2)^{1/2}$ from the point of observation (taken as the origin of a right-handed Cartesian system of coordinates with the z-axis positive downward, as illustrated in Figure 1) to a surface element of area ds at (x, y, z) on the surface of the body, **a** is a unit vector in the direction in which the component of the field **F** is being calculated, $\mathbf{u_n}$ is the unit outward normal vector at the surface element ds, the symbol $\cdot$ (raised dot) represents the scalar product of the vectors it connects, and the integration is carried out over the entire bounding surface of the body.

Now we seek a surface mass density distribution, which would produce the same field everywhere as the solid body. For this we consider a field from a surface element ds at position vector **r** in the direction of **r**, that is, along the unit vector $(\mathbf{r}/r)$. The gravitational attraction attributable to a positive mass density on such a surface element is toward the surface element. Replacing **a** by $(\mathbf{r}/r)$, the integrand in equation (1) becomes $-G\rho\mathbf{r}\cdot\mathbf{u_n}(1/r^2)\,ds$. According to the inverse square law, this field is the negative of the attraction, at the origin, because of the element ds if the surface mass density at the element is taken to be equal to the product $\rho\mathbf{r}\cdot\mathbf{u_n}$. The net field **F** at the point of observation from surface mass density $\rho\mathbf{r}\cdot\mathbf{u_n}$ can then be found by integrating $G\rho\mathbf{r}\cdot\mathbf{u_n}\,(1/r^2)\,ds$ over the entire boundary surface of the body. We get

$$\mathbf{F} = G\rho\iint (1/r)(\mathbf{r}/r)\cdot\mathbf{u_n}\,ds = G\iint(\rho\mathbf{r}\cdot\mathbf{u_n})/r^2\,ds$$
$$= G\iint \sigma'\,ds/r^2. \tag{2}$$

Thus, the attraction from a solid body, at the origin, is the same as that from a fictitious distribution of masses on its surface, the surface mass density (σ') everywhere taken to be equal to the product

$$\sigma' = \rho\mathbf{r}\cdot\mathbf{u_n}. \tag{3}$$

This surface mass density is fictitious; it not only changes with the position of the observation point, but it can also be negative or positive. It is merely an artifice, which can be used for computing the gravity field from the body.

To get the component of a field from an element ds along one of the cardinal directions, say, z, we must multiply the integrand by the ratio (z/r). Integrating over the entire surface of the body, we get F_z, the z-component of the field **F**. The x- and y-components of the field are obtained similarly by replacing z by x and y, respectively. Thus, we have

$$F_x = G\rho\iint \mathbf{r}\cdot\mathbf{u_n}(x/r^3)\,ds,$$
$$F_y = G\rho\iint \mathbf{r}\cdot\mathbf{u_n}(y/r^3)\,ds, \tag{4}$$

and

$$F_z = G\rho\iint \mathbf{r}\cdot\mathbf{u_n}(z/r^3)\,ds.$$

If the body is bounded by a number of plane facets, the integration can be done separately on each of these facets and the results added. Integrating over the ith facet having outward normal $\mathbf{u}_i$, we can write

$$F_{zi} = G\rho d_i\iint_i (z/r^3)\,ds, \tag{5}$$

where d_i is the product $\mathbf{r}\cdot\mathbf{u}_i$, a constant for the ith facet, its absolute value being equal to the perpendicular distance of the origin from the plane of that facet.

Summing over all the facets of the body and repeating the same procedure for the x- and y-components, we get

$$F_x = G\sum_i \rho\, d_i\iint_i (x/r^3)\,ds,$$
$$F_y = G\sum_i \rho\, d_i\iint_i (y/r^3)\,ds, \tag{6}$$

and

$$F_z = G\sum_i \rho d_i\iint_i (z/r^3)\,ds.$$

The unit outward normal vector $\mathbf{u}_i$ for the ith facet of the polyhedron can be found in a number of ways. Let the ith facet have m vertices and the position vector of its kth vertex, seen in counterclockwise order from outside the body, be $\mathbf{a}_{i,k}$. The vector $\mathbf{u}_i$ is then given by

$$\mathbf{u}_i = \frac{\mathbf{n}_i}{|\mathbf{n}_i|}, \tag{7}$$

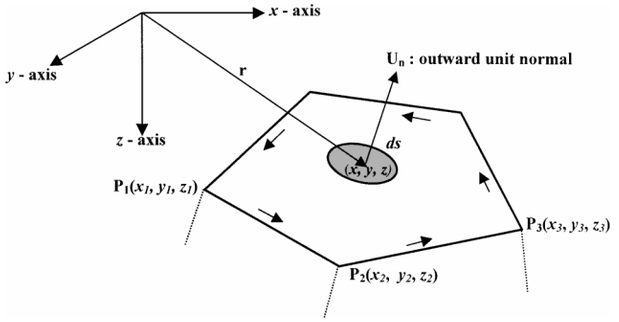

FIG. 1. View of a polygonal surface in a right-handed Cartesian system of coordinates. The scalar product of the unit outward normal vector $\mathbf{u_n}$ with direction cosines ℓ, m, n, and the radius vector **r** of any point (x, y, z) on the polygon, given by $\ell x + my + nz$, is negative if the outside surface of the polygon is being seen from the observation point at the origin but is positive otherwise. The small solid arrows show the direction of the line integration around the edges.

where

$$\mathbf{n}_i = \sum_{l=2}^{m-1} (\mathbf{a}_{i,l} - \mathbf{a}_{i,1}) \times (\mathbf{a}_{i,l+1} - \mathbf{a}_{i,1}) \tag{8}$$

and the symbol || represents the absolute value of the quantity enclosed (see Pohánka, 1988). If the coordinates of any one corner of such a facet are (x_1, y_1, z_1) and the components of $\mathbf{u}_i$ are (ℓ, m, n), the product $\mathbf{r} \cdot \mathbf{u}_i$ is given by $d_i = \mathbf{r} \cdot \mathbf{u}_i = \ell x_1 + m y_1 + n z_1$. The value d_i can be positive or negative, depending on the orientations of $\mathbf{r}$ and $\mathbf{u}_i$.

MAGNETIC FIELD FROM A FINITE-SIZED BODY

We turn briefly to computing a magnetic field from a uniformly magnetized solid. With the assumption of uniform magnetization, the equivalent surface pole density σ_i at the ith facet is numerically equal to the component of the intensity of magnetization $\mathbf{M}$ of the polyhedron parallel to the outward normal for that facet. This is given by the product $\mathbf{u}_i \cdot \mathbf{M}$. Taking the observation point at the origin of a right-handed Cartesian system of coordinates with the z-axis positive downward, we use the inverse square law and, remembering that the field from a positive surface pole density on an element ds is away from that element, we get the x-, y-, and z-components of the magnetic field as

$$\begin{aligned} H_x &= -\sum_i \sigma_i \iint_i (x/r^3)\, ds, \\ H_y &= -\sum_i \sigma_i \iint_i (y/r^3)\, ds, \\ H_z &= -\sum_i \sigma_i \iint_i (z/r^3)\, ds, \end{aligned} \tag{9}$$

where σ_i is the surface pole density at the ith facet and the summation is over all the facets of the body.

EVALUATING SURFACE INTEGRALS

As explained in Guptasarma and Singh (1999), the surface integrals in equations (9) can be evaluated for each facet by converting them into line integrals around the polygonal boundary of that facet. Also used is the fact that the component of the field from each plane facet along the outward normal to that facet is numerically equal to the solid angle Ω subtended by the facet at the observation point. With (ℓ, m, n) as the Cartesian components of the unit outward normal for the ith facet, the x-, y-, and z-components of the magnetic field from that facet are given by

$$\begin{aligned} H_x &= \sigma_i(\ell\Omega + nQ_i - mR_i), \\ H_y &= \sigma_i(m\Omega + \ell R_i - nP_i), \\ H_z &= \sigma_i(n\Omega + mP_i - \ell Q_i), \end{aligned} \tag{10}$$

where P_i, Q_i, and R_i are computed for the ith facet by summing up the contributions from each edge of its boundary in the manner given below.

The contributions P_{ij}, Q_{ij}, and R_{ij} from the jth edge of the polygonal boundary of the ith facet are given by

$$P_{ij} = IL_x, \quad Q_{ij} = IL_y, \quad \text{and} \quad R_{ij} = IL_z, \tag{11}$$

where $L_x = x_2 - x_1$, $L_y = y_2 - y_1$, and $L_z = z_2 - z_1$ are Cartesian components of the length of the edge $L = (L_x^2 + L_y^2 + L_z^2)^{1/2}$ and where (x_1, y_1, z_1) and (x_2, y_2, z_2) are the coordinates of the beginning and the end of the edge. The value of I in equations (11) is given by (Guptasarma and Singh, 1999)

$$I = (1/L) \ln\left[\left(\sqrt{\left(L^2 + b + r_1^2\right)} + L + b/2L\right)/(r_1 + b/2L)\right], \quad \text{if } (r_1 + b/2L) \neq 0, \tag{12}$$

and

$$I = (1/L) \ln\left[|(L - r_1)|/r_1\right], \quad \text{if } (r_1 + b/2L) = 0, \tag{13}$$

where $b = 2(x_1 L_x + y_1 L_y + z_1 L_z)$; ln represents the natural logarithm; and $r_1 = (x_1^2 + y_1^2 + z_1^2)^{1/2}$ is the distance from the origin to the beginning of the edge. The beginning and the end of each edge are reckoned by considering that the edges are traversed in the counterclockwise direction while viewing the polygonal facet from the outside of the body. (We have found that the components of the magnetic field at the origin may not always be obtained correctly with this formula when $(r_1 + b/2L) = 0$.)

The sums of the contributions from all the edges of the ith facet—$P_i = \sum_j P_{ij}$, $Q_i = \sum_j Q_{ij}$, and $R_i = \sum_j R_{ij}$—are substituted in equations (10) to get the contribution from the ith facet. The absolute value of Ω is calculated by any standard procedure, such as the one given by Todhunter and Leathem (1943) or the straightforward procedure given in Appendix A of Guptasarma and Singh (1999). The latter procedure works also for polygons having some internal angles larger than π. The sign of Ω is taken as positive if the origin is on the outer side of the facet and as negative otherwise. Whether the origin is on the outer or the inner side of the facet is easily found from the sign of the scalar product of the vectors $\mathbf{u_n}$ and the position vector of any of the corners of the polygonal facet. If this sign is positive, the origin is on the inner side; Ω is then replaced by $-\Omega$ for that facet.

The sum of the contributions from all facets of the body gives the magnetic field at the origin.

Simultaneous computation of gravity and magnetic anomalies

By comparing equations (6) and (9), we find that the components of gravity and magnetic fields have the same form. Therefore, the components of the gravity field of a polyhedron can be obtained by carrying out the same computation as that for the components of the magnetic field by changing the value of σ_i in equations (10) to $-G\rho d_i$ for the ith facet. Thus,

$$\begin{aligned} F_x &= -G\rho d_i(\ell\Omega + nQ_i - mR_i), \\ F_y &= -G\rho d_i(m\Omega + \ell R_i - nP_i), \\ F_z &= -G\rho d_i(n\Omega + mP_i - \ell Q_i). \end{aligned} \tag{14}$$

In the above procedures, fields are calculated at the origin. As such, a translation of the body is made for every observation point, bringing the observation point to the origin.

Since the gravity field is obtained by carrying out the same numerical computations as that needed for the magnetic field, except for replacing the constant σ_i for each facet by $-G\rho d_i$, it is possible to compute the gravity and magnetic fields simultaneously. When the observation point is near a corner of the body at a small distance r, the magnitude of the computed magnetic field increases as $\ln(1/r)$, as it should. As such, we cannot

compute the magnetic field if the observation point happens to be at a corner or on the edge of the body.

This difficulty does not appear in computing the gravity field at a corner or on an edge because all the facets meeting there can be omitted from the summation process indicated in equation (6) since $d_i = 0$. For an observation point very close to a corner but not actually on it, the distances from the facets meeting at that corner reduce as the point comes closer. As a result, the contributions from these facets quickly become very small. However, possible problems arising from the finite arithmetic accuracy of computers need to be guarded against in the usual manner, as shown in the next section.

If the vertical component of the gravity field is required, as in gravity prospecting, then only the component F_z needs to be computed and the quantity R_i is ignored. Thus, if the magnetic field is not required, the computational burden for the gravity field becomes very much less. As in computing magnetic fields, the number of evaluations of the line integral I in equations (11) can be halved because, in integrating around the boundary of each polygonal facet, every edge gets traversed twice.

AVOIDING PROBLEMS IN NUMERICAL COMPUTATIONS

As mentioned, problems may appear while computing the numerical value of expressions in equations (10), (11), and (14) unless some precautions are taken. The situations that need to be considered are those in which the observation point is (1) very far from the model compared to its linear dimensions or (2) extremely close to or at a corner of the model or on its edge.

In case 1, the quantities $L_x = x_2 - x_1$, $L_y = y_2 - y_1$, and $L_z = z_2 - z_1$ in equations (11) may vanish if the coordinates have very large values. This problem does not arise if the components of the edge lengths are computed in advance and saved as constants instead of computing them afresh for every observation point.

In case 2, the integral I cannot be computed if $(r_1 + b/2L)$ in the argument of the logarithm tends to be zero. This happens if the observation point lies in the line of the edge but falls (a) between P_1 and P_2 or (b) outside the edge but nearer to P_2. In the first case the magnetic field cannot be computed. In the second, the remedy is to interchange the coordinates of ends 1 and 2, carry out the computation, and change the sign of the result. As already mentioned, this difficulty does not arise for the computation of the gravity field.

These precautions can be easily built into a simple program in any convenient programming language. The above algorithm has been implemented in a computer program and tested successfully for different polyhedral models, including the trapezohedral model given by Coggon (1976) for both gravity and magnetic fields. In the Appendix, we provide the source code, written in MATLAB V. 4.2, to simultaneously compute gravity and magnetic fields.

CONCLUSIONS

The new formulation clearly shows that at any point a gravity field from a solid body having uniform volume density can be computed as the field from a fictitious distribution of surface mass density on the same body. This surface mass density at every surface element is equal to the product of the volume density of the body and the scalar product of (1) the unit outward vector normal to that surface element and (2) the position vector of the surface element with respect to the point of observation. We have also shown that the steps needed to compute the gravity field are the same as those required to compute the magnetic field, except that the surface magnetic pole density must be replaced by the negative of this surface mass density. The simple scheme given by Guptasarma and Singh (1999) to compute the magnetic field may thus be used to compute the gravity, the magnetic field, or both simultaneously.

The gravity field can be computed at all points, including points on the surface of the body or at any of its corners. There is no need to make elaborate coordinate transformations or complex trigonometric calculations, as with all previously published schemes. This new idea, used in conjunction with the computational scheme of our earlier paper, will substantially simplify and speed up the numerical modeling of gravity and magnetic anomalies from finite bodies with plane bounding surfaces.

ACKNOWLEDGMENTS

The authors thank the Director of NGRI for his kind permission to publish the paper. Colin T. Barnett, one of the reviewers of our earlier paper for the computation of magnetic fields, suggested that we try to find an equally simple method for gravity. Reviewer Afif H. Saad and Associate Editor Robert Pawlowski suggested changes resulting in an improved presentation. The equivalence of the surface mass distribution to a solid model in the case of gravity was first noticed by B.S. in closed-form expressions for the gravity field of infinite 2-D prisms.

REFERENCES

Barnett, C. T., 1976, Theoretical modeling of the magnetic and gravitational fields of an arbitrarily shaped three-dimensional body: Geophysics, **41**, 1353–1364.
Barton, D. C., 1929, Calculations in the interpretation of observations with the Eötvös torsion balance: AIME, **81**, 481–504.
Bhattacharyya, B. K., 1964, Magnetic anomalies from prism-shaped bodies with arbitrary polarization: Geophysics, **29**, 517–531.
Coggon, J. H., 1976, Magnetic and gravity anomalies of polyhedra: Geoexploration, **14**, 93–105.
Furness, P., 1994, A physical approach to computing magnetic fields: Geophys. Prosp., **42**, 405–416.
Götze, H.-J., and Lahmeyer, B., 1988, Application of three-dimensional interactive modeling in gravity and magnetics: Geophysics, **53**, 1096–1108.
Guptasarma, D., and Singh, B., 1999, New scheme for computing the magnetic field resulting from a uniformly magnetized arbitrary polyhedron: Geophysics, **64**, 70–74.
Holstein, H., and Ketteridge, B., 1996, Gravimetric analysis of uniform polyhedra: Geophysics, **61**, 357–364.
Hubbert, M. K., 1948, A line integral method of computing the gravimetric effects of two-dimensional masses: Geophysics, **13**, 215–225.
Nagy, D., 1966, Gravitational attraction of a right rectangular prism: Geophysics, **31**, 362–371.
Okabe, M., 1979, Analytical expressions for gravity anomalies due to homogeneous polyhedral bodies and translations into magnetic anomalies: Geophysics, **44**, 730–741.
Plouff, D., 1976, Gravity and magnetic fields of polygonal prisms and application to magnetic terrain corrections: Geophysics, **41**, 727–741.
Pohánka, V., 1988, Optimum expressions for computation of the gravity field of a homogeneous polyhedral body: Geophys. Prosp., **36**, 733–751.
Routh, E. J., 1892, A treatise on analytical statics, vol. **II**: Cambridge Univ. Press.
Talwani, M., 1965, Computation with the help of a digital computer of magnetic anomalies caused by bodies of arbitrary shape: Geophysics, **30**, 797–817.
Talwani, M., and Ewing, M., 1960, Rapid computation of gravitational attraction of three-dimensional bodies of arbitrary shape: Geophysics, **25**, 203–225.
Todhunter, I., and Leathem, J. G., 1943, Spherical trigonometry: Macmillan Publ. Co.

APPENDIX

SOURCE CODE FOR COMPUTATION

The source code, grvmag3d.m, an m-file written in MATLAB v. 4.2, is a straightforward implementation of the algorithm presented in this paper. The program requires the file angle.m and a model file. The model file provides the model parameters and also specifies whether the magnetic, the gravity, or both anomalies are to be computed. A model file, trapezod.m, for the trapezohedron model from Coggon (1976) is provided. The variables defining the model parameters are explained in the comments in the file. The angle function program computes the angle between planes OP_1P_2 and OP_2P_3 (see Figure 1) to get the solid angle subtended by a polygonal facet at the origin, using the scheme given by Guptasarma and Singh (1999).

As written, grvmag3d.m computes the cardinal components of the magnetic (Hx, Hy, and Hz), and the gravity (Gx, Gy, and Gz) fields over a rectangular array of stations along one or more north–south profiles with uniformly spaced stations on uniformly spaced profiles. The correct total magnetic field anomaly (Dt) and the usual approximation (Dta) obtained as the projection of the anomalous field along the direction of the ambient earth's field are also computed. Comments within the source code lines (written after the % sign) facilitate reading the code for modifying it or rewriting in some other programming language. Comments paragraphs must be omitted from the code or entered with a % sign at the beginning of each line.

Program file: grvmag3d.m

```
Comments: Program for simultaneous computation of gravity
& magnetic fields from a 3-D polyhedron. With all distances
in meters, model density in g/cm^3, ambient magnetic induction
and remnant magnetization in gamma, and the magnetic sus-
ceptibility in SI, it gives gravity fields in milligals and magnetic
fields in gamma.
Gc= 6.6732e-3; % Universal Gravitational constant.
trapezod % Change this filename to compute other models
for i=1:2,close(figure(i)),end % clear old figures if present
Nedges=sum(Face(1:Nf,1)); Edge=zeros(Nedges,8);
% Get edgelengths
for f=1:Nf, indx=[Face(f,2:Face(f,1)+1) Face(f,2)];
for t=1:Face(f,1); edgeno=sum(Face(1:f-1,1))+t;
ends=indx(t:t+1); p1=Corner(ends(1),:);
p2=Corner(ends(2),:);
V=p2-p1; L=norm(V);Edge(edgeno,1:3)=V;
Edge(edgeno,4) =L;
Edge(edgeno,7:8)=ends;end,end
for t=1:Nf, ss=zeros(1,3); for t1=2:Face(t,1) -1;
v1=Corner(Face(t,t1+2),:) -Corner(Face(t,2),:);
v2=Corner(Face(t,t1+1),:) -Corner(Face(t,2),:);
ss=ss+cross(v2,v1); end, Un(t,:)=ss./norm(ss); end
[X,Y]=meshgrid([s_end:stn_spcng:n_end], ...
[w_end:prof_spcng:e_end]);
[npro nstn]=size(X);
if calgrv,Gx=zeros(size(X)); Gy=Gx; Gz=Gx;end
if calmag,Hin=Hincl*pi/180; Dec=Decl*pi/180;

cx=cos(Hin)*cos(Dec); cy=cos(Hin)*sin(Dec);
cz = sin(Hin);
Uh=[cx cy cz];
H=Hintn .* Uh; % The ambient magnetic field
Ind_magn=Susc.*H/(4*pi); % Induced magnetization
Min=Mincl*pi/180; Mdec=Mdecl*pi/180;
mcx=cos(Min) *cos(Mdec);
mcy=cos(Min)*sin(Mdec); mcz=sin(Min);
Um=[mcx mcy mcz];
Rem_magn=Mstrength .* Um; % Remnant magnetization
Net_magn=Rem_magn+Ind_magn; % Net magnetization
Pd=(Un * Net_magn')'; % Pole densities
 Hx=zeros(size(X)); Hy=Hx; Hz=Hx;end

Comments: Now, for each observation point do the follow-
ing: For each face find solid angle; for each side find p,q,r, and
add p,q,r of sides to get P,Q,R for the face; if calmag=1, find
hx,hy,hz; if calgrv=1, find gx,gy,gz. Add the components from
all the faces to get Hx,Hy,Hz and Gx,Gy,Gz at the station.
for pr=1:npro,for st=1:nstn,opt=[X(pr,st) Y(pr,st) 0];
fsign=zeros(1,Nf); Omega=zeros(1,Nf);
for t=1:Ncor, cor(t,:) = Corner(t,:) -opt; end % shift origin
for f=1:Nf, nsides=Face(f,1); cors=Face(f,2:nsides+1);
Edge(:,5:6)=zeros(Nedges,2); % Clear record of integration
indx=[1:nsides 1 2];for t=1:nsides, crs(t,:)=cor(cors(t),:);end
   % Find if the face is seen from inside
fsign(f)=sign(dot(Un(f,:),crs(1,:)));
   % Find solid angle W subtended by face f at opt
dp1=dot(crs(indx(1),:),Un(f,:)); dp=abs(dp1);
if dp= =0, Omega(f)=0; end, if dp~=0, W=0; for t=1:nsides
p1=crs(indx(t),:); p2=crs(indx(t+1),:); p3=crs(indx(t+2),:);
W=W + 2angle(p1,p2,p3,Un(f,:)); end
W=W- (nsides-2).*pi; Omega(f)= -fsign(f)*W; end
indx=[1:nsides 1 2]; for t=1:nsides, crs(t,:)=cor(cors(t),:);end
   % Integrate over each side, if not done, and save result
PQR=[0 0 0];
for t=1:nsides, p1=crs(indx(t),:); p2=crs(indx(t+1),:);
Eno=sum(Face(1:f-1,1))+t; % Edge number
if Edge(Eno,6)= =1,I=Edge(Eno,5);V=Edge(Eno,1:3);
pqr=I .* V; PQR=PQR+pqr; end
if Edge(Eno,6) =1
chsgn=1; % if origin,p1 & p2 are on a st line
if dot(p1,p2)./(norm(p1)*norm(p2))==1
if norm(p1)>norm(p2) % and p1 farther than p2
chsgn=-1; psave=p1; p1=p2; p2=psave; % interchange p1,p2
end, end
V=Edge(Eno,1:3); L=Edge(Eno,4); L2=L*L;
b=2* (dot(V,p1));
r1=norm(p1); r12=r1*r1; b2=b/L/2;
if r1+b2 == 0,V= -Edge(Eno,1:3);b=2*(dot(V,p1));
b2=b/L/2; end
if r1+b2 ~= 0
I = (1/L).* log ((sqrt(L2 + b + r12) + L + b2)./(r1 + b2));end
s=find(Edge(:,7)== Edge(Eno,8) & Edge(:,8) ...
== Edge(Eno,7));
I=I*chsgn; % change sign of I if p1,p2 were interchanged
Edge(Eno,5)=I;Edge(s,5)=I;Edge(Eno,6)=1;Edge(s,6)=1;
pqr=I .* V; PQR=PQR+pqr; end, end
% From Omega,l,m,n,PQR, get components of field due to
% face f
l=Un(f,1);m=Un(f,2);n=Un(f,3);p=PQR(1,1);
```

DOI:10.1190/1.1444942

```
q=PQR(1,2);r=PQR(1,3);
if calmag== 1,
hx=Pd(f)*(l*Omega(f)+n*q−m*r); Hx(pr,st)=Hx(pr,st)+hx;
hy=Pd(f)*(m*Omega(f)+l*r−n*p); Hy(pr,st)=Hy(pr,st)+hy;
hz=Pd(f)*(n*Omega(f)+m*p−l*q); Hz(pr,st)=Hz(pr,st)+hz;
end
if calgrv== 1, if dp~=0 % if distance to face is non-zero
gx=−dens*Gc*dp1*(l*Omega(f)+n*q−m* r);
Gx(pr,st)=Gx(pr,st)+ gx;
gy=−dens*Gc*dp1*(m*Omega(f)+l*r−n* p);
Gy(pr,st)=Gy(pr,st)+ gy;
gz=−dens*Gc*dp1*(n*Omega(f)+m*p−l* q);
Gz(pr,st)=Gz(pr,st)+ gz;
end,end
end, end, end % end of faces, stns, profiles
if calmag== 1
Htot=sqrt((Hx+H(1,1)).^2 + (Hy+H(1,2)).^2 . . .
+ (Hz+H(1,3)).^2);
Dt=Htot−Hintn; % Correct change in Total field
% Approx. change in Total field
Dta=Hx.*cx+Hy.*cy+Hz.*cz;end
```

Model file: trapezod.m

Comments: The model file, trapezod.m, is an example of trapezohedron model from Coggon (1976). This example shows how model parameters are to be given. The variables are Ncor = number of corners of the model; Hintn = total intensity of ambient magnetic induction, gamma; Hincl = inclination of Hintn, degrees, downward from horizontal; Decl = declination of Hintn, clockwise from north; Susc = magnetic volume susceptibility in units SI, a dimensionless number equal to $(\mu_r - 1)$, where μ_r = magnetic permeability of the model relative to free space; Mstrength, Mincl, Mdecl = magnitude (gamma), inclination and declination (degrees), respectively, of remnant magnetic induction; Nf = number of faces; Fht = height of observation plane above origin, meters; and dens = density of model, g/cm^3.

```
calgrv=1; % Change to zero if gravity field is not required
calmag=1; % Change to zero if magnetic field is not required
Ncor=26; Hintn=50000; Hincl=50; Decl=0; Susc=0.01
Mstrength=0; Mincl=0; Mdecl=0;
```

Comments: Corner is an array of *x, y, z* coordinates of corners in meters, one in each row, in a right-handed system with *x*-axis northward, *y*-axis eastward, and *z*-axis downward. Corners may be given in any order.

```
Corner = [100 0 0; 75 −75 0; 0 −100 0; −75 −75 0; −100 0 0;...
−75 75 0; 0 100 0; 75 75 0; 75 0 −75; 60 −60 −60; 0 −75 −75;...
−60 −60 −60; −75 0 −75; −60 60 −60; 0 75 −75; 60 60 −60;...
0 0 −100; 75 0 75; 60 −60 60; 0 −75 75; −60 −60 60; −75 0 75;...
−60 60 60; 0 75 75; 60 60 60; 0 0 100];
fht=200; dens=10;
% Add fht to depths of all corners
Corner(:,3)=Corner(:,3)+fht;
```

Comments: In each row of Face, the first number is the number of corners forming a face; the following are row numbers of the Corner array with coordinates of the corners which form that face, seen in ccw order from outside the object. The faces may have any orientation and may be given in any order, but all faces must be included.

```
Face=zeros([50,9]); % Initialize a sufficiently large array
Face(1,1:5)=[4 1 2 10 9]; Face(2,1:5)=[4 2 3 11 10];
Face(3,1:5)=[4 3 4 12 11]; Face(4,1:5)=[4 4 5 13 12];
Face(5,1:5)=[4 5 6 14 13]; Face(6,1:5)=[4 6 7 15 14];
Face(7,1:5)=[4 7 8 16 15]; Face(8,1:5)=[4 8 1 9 16];
Face(9,1:5)=[4 9 10 11 17]; Face(10,1:5)=[4 11 12 13 17];
Face(11,1:5)=[4 13 14 15 17]; Face(12,1:5)=[4 15 16 9 17];
Face(13,1:5)=[4 1 18 19 2]; Face(14,1:5)=[4 2 19 20 3];
Face(15,1:5)=[4 3 20 21 4]; Face(16,1:5)=[4 4 21 22 5];
Face(17,1:5)=[4 5 22 23 6]; Face(18,1:5)=[4 6 23 24 7];
Face(19,1:5)=[4 7 24 25 8]; Face(20,1:5)=[4 8 25 18 1];
Face(21,1:5)=[4 20 19 18 26]; Face(22,1:5)=[4 22 21 20 26];
Face(23,1:5)=[4 24 23 22 26]; Face(24,1:5)=[4 18 25 24 26];
```

Comments: Rectangular grid of stations for computing fields. Profiles are along the *x*-axis (north–south direction). All values are in meters.

```
s_end= −320; % Starting value of x; south end of profiles
stn_spcng = 40; % Stepsize in north direction; stn interval
n_end= 320; % Last x; maximum north coordinate
w_end= 0; % y value for westernmost profile
prof_spcng=1; % Profile spacing
e_end= 0; % y value for easternmost profile
```

Function program: angle.m

```
function[ang, perp]=angle(p1, p2, p3, Un)
```

Comments: Angle.m finds the angle between planes O-p1-p2 and O-p2-p3, where p1,p2,p3 are coordinates of three points, taken in ccw order as seen from origin O. This is used by grvmag3d for finding the solid angle subtended by a polygon at the origin. Un is the unit outward normal vector to the polygon.

```
inout=sign(sum(Un .* p1)); % Check if face is seen from inside
x2=p2(1,1); y2=p2(1,2); z2=p2(1,3);
if inout>0 % seen from inside; interchange p1 and p3
x3=p1(1,1); y3=p1(1,2); z3=p1(1,3);
x1=p3(1,1); y1=p3(1,2); z1=p3(1,3);
elseif inout<0 % seen from outside; keep p1 and p3 as they are
x1=p1(1,1); y1=p1(1,2); z1=p1(1,3);
x3=p3(1,1); y3=p3(1,2); z3=p3(1,3);end
% Normals
n1=[(y2*z1−y1*z2) (x1*z2−x2*z1) (x2*y1−x1*y2)];
n2=− [(y3*z2−y2*z3) (x2*z3−x3*z2) (x3*y2−x2*y3)];
n1=n1./norm(n1); n2=n2./norm(n2);
perp=sum([x3 y3 z3].*n1);
% sign of perp is −ve if points p1 p2 p3 are in cw order
perp=sign(perp);r=sum((n1.* n2)); ang=acos(r); if perp<0
ang=2*pi−ang; end,if inout= =0, ang=0; perp=1; end, return
```