Bureau of Mineral Resources, Geology & Geophysics

RECORD

RECORD 1989/16

MAGNETIC MODELLING OF TWO AND THREE-DIMENSIONAL BODIES: TEST PROGRAMS AND SUBROUTINES IN FORTRAN 77

C.TARLOWSKI

(Division of Geophysics)

Department of Primary Industries & Energy

BUREAU OF MINERAL RESOURCES, GEOLOGY & GEOPHYSICS

RECORD 1989/16

MAGNETIC MODELLING OF TWO AND THREE-DIMENSIONAL BODIES: TEST PROGRAMS AND SUBROUTINES IN FORTRAN 77

C.TARLOWSKI

(Division of Geophysics)

CONTENTS

INTRODUCTION

This Record contains test programs and subroutines in FORTRAN 77 that implement the algorithms for magnetic modelling of two and three-dimensional bodies. The algorithms and derivation of the relevant mathematical formulas are given in BMR Report 291 'Magnetic modelling of two and three-dimensional bodies', by C. Tarlowski.

```
      program main
**********************************************************************
*                                                                    *
*                              NVDIKE.F77                            *
*                                                                    *
*      Program for testing subroutine for calculating magnetic      *
*            effect from a thin vertical dike.                       *
*                 Revised: 5 March 89                                *
*                                                                    *
**********************************************************************
      parameter (n=21)
      dimension ycomp(n),zcomp(n),tot(n),y(n)
      real k,incl,nvdike
      pi=4.0*atan(1.0)
c
c     open output file
c
      open(1,file='nvout',status='fresh')
c
c     generate the points on the profile
c
      lp=(n-1)/2
      step= 50.0
      do 10 i=1,lp+1
              lpp=lp+1-i
              y(i)=-step*lpp
10    continue
c
      do 20 i=lp+2,n
              lpp=i-lp-1
              y(i)=step*lpp
20    continue
c
c     input parameters
c
      incl=-60.0          ! inclination of the inducing field [degrees
      f=58000.0           ! inducing field [nT]
      sangle=90.0         ! strike angle [degrees]
      k=4.0*pi*5.0e-3     ! susceptibility of the dike [SI units]
      y0=0.0              ! horizontal position of the dike [m]
      d=1000.0            ! depth to the top of the dike [m]
      delt=200.0          ! thickness of the dike [m]
      z=0.0               ! height of the observational point [m]
c
      do 30 i=1,n
              yp=y(i)
c
c     y-component of the field
c
              ycomp(i)=nvdike(1,yp,y0,d,delt,z,k,f,incl,sangle)
c
c     z-component of the field
c
```

```
                 zcomp(i)=nvdike(2,yp,y0,d,delt,z,k,f,incl,sangle)
c
c        total field
c
                 tot(i)=  nvdike(3,yp,y0,d,delt,z,k,f,incl,sangle)
30       continue
c
         write(1,100)
100      format(1x,//,20x,'Test of the thin vertical dike',//)
         write(1,101)
101      format(15x,'* Input data *',/)
         write(1,102) incl,f,sangle,k,y0,z,d,delt
102      format(12x,'Inclination = ',f5.1,/,12x,'Total inducing field =
    +            ,f8.1,/,12x,'Strike angle = ',f5.1,/,12x,
    +            'Susceptibility = ',f9.4,/,12x,
    +            'Horizontal postion of the dike = ',f9.1,/,12x,
    +            'Height of the observational point = ',f9.1,/,12x,
    +            'Depth to the top of the dike = ',f9.1,/,12x,
    +            'Thickness of the dike = ',f9.1,///)
         write(1,103)
103      format(15x,'* Output data *',//)
         write(1,104)
104      format(12x,'i',7x,'y(i)',4x,'ycomp(i)',4x,'zcomp(i)',6x,
    +        'tot(i)',/)
         do 40 i=1,n
         write(1,105) i,y(i),ycomp(i),zcomp(i),tot(i)
40       continue
105      format(8x,i5,3x,f8.3,2x,3g13.6)
         stop
         end
         real function nvdike(l,y,y0,d,delt,z,k,f,incl,sangle)
**********************************************************************
*                                                                    *
*     Real function nvdike calculates y-component,z-component and    *
*     total magnetic field from a thin vertical dike                 *
*                                                                    *
*     Input parameters:                                              *
*                       l-switch                                     *
*                       if l=1 then nvdike=y-component               *
*                       if l=2 then nvdike=z-component               *
*                       if l=3 then nvdike=total field,              *
*                                                                    *
*                       y-horizontal distance in meters along profile,*
*                       y0-horizontal position of the dike in meters, *
*                       d-depth to the top of the dike in meters,    *
*                       delt-thickness of the dike in meters,        *
*                       z-height of the observational point in meters,*
*                       k-susceptibility in SI units,                *
*                       f-intensity of the inducing field in nT,     *
*                       incl-inclination of the inducing field in    *
*                            degrees,                                *
*                       sangle-strike angle in degrees (taken clocwise*
*                              from the magnetic north to the positive*
*                              y).                                   *
*                                                                    *
**********************************************************************
```

```
      real k,incl,inclr,jy,jz
      pi=4.0*atan(1.0)
c
c     convert degrees to radians
c
      inclr=(incl/180.0)*pi
      sanglr=(sangle/180.0)*pi
c
c     calculate y & z-component of induced magnetization,
c     in order to take into account remanence,change jy & jz
c     accordingly
c
      jy=k*f*cos(inclr)*cos(sanglr)
      jz=k*f*sin(inclr)
c
      dif1=y-y0
      dif2=d-z
      r=dif1*dif1+dif2*dif2
      s=delt/(2.0*pi*r)
c
c     y-component of the magnetic field
c
      yc=-s*(jy*dif2+jz*dif1)
c
c     z-component of the magnetic field
c
      zc=s*(jz*dif2-jy*dif1)
c
      if (l.eq.1) then
              nvdike=yc
      endif
c
      if (l.eq.2) then
              nvdike=zc
      endif
c
c     total magnetic field
c
      if (l.eq.3) then
              nvdike=yc*cos(sanglr)*cos(inclr)+zc*sin(inclr)
      endif
      return
      end
```

Test of the thin vertical dike

* Input data *

Inclination = -60.0
Total inducing field = 58000.0
Strike angle = 90.0
Susceptibility = .0628
Horizontal postion of the dike = .0
Height of the observational point = .0
Depth to the top of the dike = 1000.0
Thickness of the dike = 200.0

* Output data *

i	y(i)	ycomp(i)	zcomp(i)	tot(i)
1	-500.000	-40.1836	-80.3671	69.5999
2	-450.000	-37.5938	-83.5417	72.3492
3	-400.000	-34.6411	-86.6025	75.0000
4	-350.000	-31.3235	-89.4957	77.5055
5	-300.000	-27.6492	-92.1641	79.8165
6	-250.000	-23.6374	-94.5495	81.8823
7	-200.000	-19.3190	-96.5951	83.6538
8	-150.000	-14.7373	-98.2483	85.0855
9	-100.000	-9.94644	-99.4643	86.1386
10	-50.000	-5.01044	-100.208	86.7830
11	.000	-.182072E-04	-100.459	86.9999
12	50.000	5.01040	-100.208	86.7830
13	100.000	9.94641	-99.4643	86.1386
14	150.000	14.7372	-98.2483	85.0855
15	200.000	19.3190	-96.5951	83.6538
16	250.000	23.6374	-94.5495	81.8823
17	300.000	27.6492	-92.1641	79.8165
18	350.000	31.3235	-89.4957	77.5055
19	400.000	34.6410	-86.6025	75.0000
20	450.000	37.5937	-83.5417	72.3492
21	500.000	40.1835	-80.3671	69.5999

```
      program main
***********************************************************************
*                                                                     *
*                             NDDIKE.F77                              *
*                                                                     *
*     Program for testing subroutine for calculating magnetic field  *
*                 from a thin dipping dike.                           *
*                            Revised:5 March 89                       *
*                                                                     *
***********************************************************************
      parameter(n=21)
      dimension y(n),ycomp(n),zcomp(n),tot(n)
      real k,incl,nddike
      pi=4.0*atan(1.0)
c
c     open output file
c
      open(1,file='ndout',status='fresh')
c
c     generate point on the profile
c
      step=50.0
      l=(n-1)/2
      do 10 i=1,l+1
              ll=l+1-i
              y(i)=-step*ll
10    continue
c
      do 20 i=l+2,n
              ll=i-l-1
              y(i)=step*ll
20    continue
c
c     Input parameters
c
      incl=-60.0        ! inclination of the inducing field in degrees
      f=58000.0         ! inducing field in nanoTesla
      k=4.0*pi*1.0e-3 ! susceptibility in SI units
      sangle= 30.0      ! strike angle in degrees
      d=1000.0          ! depth to the top of the dike in m.
      z=0.0             ! height of the observational point in m.
      y0=0.0            ! horizontal position of the dike in m.
      delt=141.4        ! thickness of the dike in m.
      dip=135.0         ! dipping angle in degrees
c
c     calculate y-component,z-component and total field
c
      do 30 i=1,n
              yp=y(i)
              ycomp(i)=nddike(1,yp,y0,d,z,delt,dip,k,f,incl,sangle)
              zcomp(i)=nddike(2,yp,y0,d,z,delt,dip,k,f,incl,sangle)
              tot(i)=  nddike(3,yp,y0,d,z,delt,dip,k,f,incl,sangle)
30    continue
```

```
c
c       write out results
c
        write(1,100)
100     format(1x,//,20x,'Test of the thin dipping dike',//)
        write(1,101)
101     format(15x,'* Input data *',/)
        write(1,102) incl,f,sangle,k,y0,z,d,delt,dip
102     format(12x,'Inclination = ',f5.1,/,12x,'Total inducing field
+                  = ',f9.1,/,12x,'Strike angle = ',f5.1,/,12x,
    +             'Susceptibility = ',f9.4,/,12x,
    +             'Horizontal position of the dike = ',f9.1,/,12x,
    +             'Height of the observational point = ',f9.1,/,12x,
    +             'Depth to the top of the dike = ',f9.1,/,12x,
    +             'Thickness of the dike = ',f9.1,/,12x,
    +             'Dipping angle = ',f5.1,///)
        write(1,103)
103     format(15x,'* Output data *',/)
        write(1,104)
104     format(12x,'i',5x,'y(i)',7x,'ycomp(i)',4x,'zcomp(i)',5x,
    +    'tot(i)',/)
        do 50 i=1,n
                write(1,105) i,y(i),ycomp(i),zcomp(i),tot(i)
50      continue
105     format(8x,i5,3x,f8.3,2x,3g13.6)
        stop
        end
c
        real function nddike(l,y,y0,d,z,delt,dip,k,f,incl,sangle)
**********************************************************************
*                                                                    *
*       Function nddike calculates y-component,z-component and       *
*         total magnetic field of the thin dipping dike.             *
*                                                                    *
*       Parameters:                                                  *
*                  l-switch                                          *
*                 if l=1 then nddike=y-component                     *
*                 if l=2 then nddike=z-component                     *
*                 if l=3 then nddike=total field,                    *
*                                                                    *
*                  y-horizontal distance along profile in meters,    *
*                  y0-horizontal position of the dike in meters,     *
*                  d-depth to the top of the dike in meters,         *
*                  z-height of the observational point in meters,    *
*                  delt-thickness of the dike in meters,             *
*                 dip-dipping angle in degrees, clockwise from       *
*                     positive y axis,                               *
*                 k-susceptibility in SI units,                      *
*                  f-intensity of the inducing field in nT,          *
*                 incl-inclination of the inducing field in degrees,*
*                  sangle-strike angle in degrees, clockwise from    *
*                        magnetic north to positive y axis.          *
*                                                                    *
**********************************************************************
        real k,jy,jz,inclr,jper,jpar,incl
        pi=4.0*atan(1.0)
```

```
c
c       convert degrees to radians
c
        inclr=(incl/180.0)*pi
        sanglr=(sangle/180.0)*pi
        dipr=(dip/180.0)*pi
c
c       calculate  y- and z-component of magnetization,
c       in order to take into account remanence change jy and jz
c       accordingly
c
        jy=k*f*cos(inclr)*cos(sanglr)
        jz=k*f*sin(inclr)
c
        diff1=y-y0
        diff2=d-z
        r=diff1*diff1+diff2*diff2
        s=(2.0*delt)/(4.0*pi*r)
c
c       rotate jy and jz
c
        fi=dipr-0.5*pi
c
        jper=jy*cos(fi)+jz*sin(fi)
        jpar=-jy*sin(fi)+jz*cos(fi)
c
c       calculate y-component
c
        yc=-s*(jper*diff2+jpar*diff1)
c
c       calculate z-component
c
        zc=s*(-jper*diff1+jpar*diff2)
c
        if(l.eq.1) nddike=yc
c
        if(l.eq.2) nddike=zc
c
        if(l.eq.3) then
                nddike=yc*cos(sanglr)*cos(inclr)+zc*sin(inclr)
        end if
        return
        end
```

Test of the thin dipping dike

* Input data *

Inclination = -60.0
Total inducing field = 58000.0
Strike angle = 30.0
Susceptibility = .0126
Horizontal position of the dike = .0
Height of the observational point = .0
Depth to the top of the dike = 1000.0
Thickness of the dike = 141.4
Dipping angle = 135.0

* Output data *

i	y(i)	ycomp(i)	zcomp(i)	tot(i)
1	-500.000	-2.00889	-14.0621	11.3083
2	-450.000	-1.46177	-14.4088	11.8454
3	-400.000	-.865905	-14.7202	12.3731
4	-350.000	-.223716	-14.9883	12.8833
5	-300.000	.460740	-15.2048	13.3672
6	-250.000	1.18168	-15.3620	13.8156
7	-200.000	1.93160	-15.4529	14.2190
8	-150.000	2.70141	-15.4718	14.5687
9	-100.000	3.48071	-15.4146	14.8567
10	-50.000	4.25820	-15.2795	15.0763
11	.000	5.02218	-15.0666	15.2227
12	50.000	5.76110	-14.7785	15.2932
13	100.000	6.46420	-14.4202	15.2873
14	150.000	7.12192	-13.9983	15.2067
15	200.000	7.72644	-13.5213	15.0554
16	250.000	8.27183	-12.9986	14.8389
17	300.000	8.75427	-12.4403	14.5643
18	350.000	9.17192	-11.8564	14.2395
19	400.000	9.52483	-11.2566	13.8729
20	450.000	9.81467	-10.6500	13.4730
21	500.000	10.0444	-10.0444	13.0480

```
        program main
********************************************************************
*                                                                  *
*                          TDDIKE.F77                              *
*                                                                  *
*       Program for testing subroutine for calculating magnetic    *
*               effect from a thick dipping dike.                  *
*                     Revised: 25 March 88                         *
*                                                                  *
********************************************************************
        parameter (n=21)
        dimension y(n),ycomp(n),zcomp(n),tot(n)
        real incl,k
        pi=4.0*atan(1.0)
c
c       open output file
c
        open(1,file='tdout',status='fresh')
c

c       generate points on the profile
c
        step=50.0
        l=(n-1)/2
        do 10 i=1,l+1
                ll=l+1-i
                y(i)=-step*ll
10      continue
c
        do 20 i=l+2,n
                ll=i-l-1
                y(i)=step*ll
20      continue
c
c       input parameters
        f=58000.0        ! inducing field in [nT]
        d=1000.0         ! depth to the top of the dike [m]
        incl=-60.0       ! inclination of the inducing field [degrees]
        sangle= 90.0     ! strike angle [degrees]
        delta= 135.0     ! dipping angle of teh dike [degrees]
        k=4.0*pi*1.0e-3  ! susceptibility of the dike [SI units]
        z=0.0            ! height of the observational point [m]
        y1=-100.0     ! position of the left top corner on y axis [m]
        y2= 100.0     ! position of the right top corner on y axis [m]
c
c       find y, z-component and total field
c

        do 30 i=1,n
                yp=y(i)
                ycomp(i)=tddike(1,yp,y1,y2,d,delta,z,k,f,incl,sangle)
                zcomp(i)=tddike(2,yp,y1,y2,d,delta,z,k,f,incl,sangle)
                tot(i)  =tddike(3,yp,y1,y2,d,delta,z,k,f,incl,sangle)
30      continue
```

```
c
        write(1,100)
100     format(1x,//,20x,'* Test of the thick dike *',//)
        write(1,101)
101     format(15x,'* Input data *',/)
        write(1,102) f,incl,sangle,k,z,d,delta,y1,y2
102     format(12x,'Total inducing field = ',f9.1,/,12x,
   +            'Inclination of the inducing field = ',f5.1,/,12x,
   +            'Strike angle = ',f5.1,/,12x,
   +            'Susceptibility = ',f9.4,/,12x,
   +            'Height of the observational point = ',f9.1,/,12x,
   +            'Depth to the top of the dike = ',f9.1,/,12x,
   +            'Dipping angle of the dike = ',f5.1,/,12x,
   +            'Position on the y-axis of the left corner = ',f9.1,
   +             /,12x,'Position on the y-axis of the right corner ='
   +              ,1x,f9.1,///)
        write(1,103)
103     format(15x,'* Output data *',//)
        write(1,104)
104     format(12x,'i',7x,'y(i)',5x,'ycomp(i)',6x,'zcomp(i)',6x,
   +          'tot(i)',/)
        do 35 i=1,n
                write(1,200) i,y(i),ycomp(i),zcomp(i),tot(i)
35      continue
c
200     format(8x,i5,3x,f8.3,2x,3g13.6)
        stop
        end
        real function tddike(l,y,y1,y2,d,delta,z,k,f,incl,sangle)
*********************************************************************
*                                                                   *
*       Real function tddike calculates y-component,z-component and *
*       total magnetic field from the thick dike or plate.          *
*                                                                   *
*       Input parameters:                                           *
*                       l - switch                                  *
*                       if l=1 then ttdike=y-component              *
*                       if l=2 then ttdike=z-component              *
*                       if l=3 then ttdike=total field,             *
*                                                                   *
*               y-horizontal position in meters along profile,      *
*               y1-horizontal position of the left corner in meters,*
*               y1-horizontal position of the right corner in meters*
*               d-depth to the top of the dike in meters,           *
*               delta-dipping angle (from +y to the dike) in degrees*
*               z-height of the observational point in meters,      *
*               k-susceptibility in SI units,                       *
*               f-total inducing field in nanoTesla,                *
*               incl-inclination of the total inducing field in     *
*                    in degrees,                                    *
*              sangle-strike angle in degrees taken clockwise from  *
*                     the magnetic North to the positive y-axis.    *
*                                                                   *
*                                                                   *
*********************************************************************
        real incl,inclr,k,jy,jz
        pi=4.0*atan(1.0)
```

```
c
c       convert degrees to radians
c
        inclr=(incl/180.0)*pi
        sanglr=(sangle/180.0)*pi
        deltr=(delta/180.0)*pi
c
c       calculate jy and jz components of the magnetization,
c       in order to take into account remanence change jy and jz
c       accordingly
c
        jy=k*f*cos(inclr)*cos(sanglr)
        jz=k*f*sin(inclr)
c
        diff11=y-y1
        diff21=d-z
c
        diff12=y-y2
        diff22=d-z
c
        r1=sqrt(diff11*diff11+diff21*diff21)
        r2=sqrt(diff12*diff12+diff22*diff22)
c
        s=sin(deltr)/(2.0*pi)
c
        p=jz*sin(deltr)+jy*cos(deltr)
        q=jz*cos(deltr)-jy*sin(deltr)
c
        v=alog(r1/r2)
c
        w=(atan(diff11/diff21)-atan(diff12/diff22))
c
c       y-component of the magnetic field
c
        yc=s*(-p*v+q*w)
c
c       z-component of the magnetic field
c
        zc=s*(q*v+p*w)
c
        if(l.eq.1) tddike=yc
c
        if(l.eq.2) tddike=zc
c
        if(l.eq.3) then
        tddike=yc*cos(sanglr)*cos(inclr)+zc*sin(inclr)
        end if
        return
        end
```

* Test of the thick dike *

* Input data *

Total inducing field = 58000.0
Inclination of the inducing field = -60.0
Strike angle = 30.0
Susceptibility = .0126
Height of the observational point = .0
Depth to the top of the dike = 1000.0
Dipping angle of the dike = 135.0
Position on the y-axis of the left corner = -100.0
Position on the y-axis of the right corner = 100.0

* Output data *

i	y(i)	ycomp(i)	zcomp(i)	tot(i)
1	-500.000	-1.97615	-14.0460	11.3085
2	-450.000	-1.42962	-14.3875	11.8409
3	-400.000	-.835278	-14.6935	12.3633
4	-350.000	-.195722	-14.9562	12.8677
5	-300.000	.484996	-15.1676	13.3455
6	-250.000	1.20106	-15.3202	13.7878
7	-200.000	1.94503	-15.4074	14.1854
8	-150.000	2.70808	-15.4236	14.5299
9	-100.000	3.47999	-15.3651	14.8135
10	-50.000	4.24984	-15.2301	15.0299
11	.000	5.00629	-15.0189	15.1746
12	50.000	5.73818	-14.7340	15.2447
13	100.000	6.43506	-14.3801	15.2400
14	150.000	7.08768	-13.9637	15.1620
15	200.000	7.68838	-13.4929	15.0144
16	250.000	8.23124	-12.9768	14.8025
17	300.000	8.71256	-12.4251	14.5331
18	350.000	9.13029	-11.8476	14.2138
19	400.000	9.48428	-11.2537	13.8528
20	450.000	9.77611	-10.6522	13.4583
21	500.000	10.0085	-10.0511	13.0383

```
      program main
************************************************************************
*                                                                      *
*                             EDGE.F77                                 *
*                                                                      *
*     Program for testing subroutine for calculating magnetic         *
*                effect from an edge.                                  *
*             Revised: 5 March 89                                      *
*                                                                      *
************************************************************************
      parameter (n=21)
      dimension y(n),ycomp(n),zcomp(n),tot(n)
      real incl,k
      pi=4.0*atan(1.0)
c
c     open output file
c
      open(1,file='edgeout',status='fresh')
c
c     generate points on the profile
c
      step= 50.0
      l=(n-1)/2
      do 10 i=1,l+1
              ll=l+1-i
              y(i)=-step*ll
10    continue
c
      do 20 i=l+2,n
              ll=i-l-1
              y(i)=step*ll
20    continue
c
c     input parameters
c
      f=58000.0          ! inducing field in [nT]
      d=1000.0           ! depth to the top of the edge [m]
      incl=-60.0         ! inclination of the inducing field [degrees]
      sangle=30.0        ! strike angle [degrees]
      delta=45.0         ! angle determining the edge [degrees]
      k=4.0*pi*1.0e-3    ! susceptibility of the edge [SI units]
      z=0.0              ! height of the observational point [m]
      y0= 100.0          ! horizontal position of the edge [m]
c
      do 30 i=1,n
              yp=y(i)
              ycomp(i)=edge(1,yp,y0,d,delta,z,k,f,incl,sangle)
              zcomp(i)=edge(2,yp,y0,d,delta,z,k,f,incl,sangle)
              tot(i)  =edge(3,yp,y0,d,delta,z,k,f,incl,sangle)
30    continue
c
      write(1,100)
100   format(1x,//,20x,'Test of the edge',//)
```

```
        write(1,101)
101     format(15x,'* Input data *',/)
        write(1,102) incl,f,sangle,k,y0,z,d,delta
102     format(12x,'Inclination = ',f5.1,/,12x,
    +            'Total inducing field = ',f8.1,/,12x,
    +            'Strike angle = ',f5.1,/,12x,
    +            'Susceptibility = ',f9.4,/,12x,
    +            'Horizontal position of the edge = ',f9.1,/,12x,
    +            'Height of the observational point = ',f9.1,/,12x,
    +            'Depth to the edge = ',f9.1,/,12x,
    +            'Angle determinig the edge = ',f9.1,///)
        write(1,103)
103     format(15x,'* Output data *',//)
        write(1,104)
104     format(12x,'i',7x,'y(i)',6x,'ycomp(i)',5x,'zcomp(i)',6x,
    +           'tot(i)',/)
        do 35 i=1,n
                write(1,200) i,y(i),ycomp(i),zcomp(i),tot(i)
35      continue
200     format( 8x,i5,3x,f10.3,2x,3g13.6)
        stop
        end
        real function edge(l,y,y0,d,delta,z,k,f,incl,sangle)
**********************************************************************
*                                                                    *
*       Real function edge calculates y-component,z-component and   *
*       total magnetic field from the edge.                         *
*                                                                    *
*       Input parameters:                                           *
*                       l - switch                                  *
*                       if l=1 then edge=y-component                *
*                       if l=2 then edge=z-component                *
*                       if l=3 then edge=total field,               *
*                                                                    *
*               y-horizontal position in meteres along profile,     *
*               y0-horizontal position of the edge in meters,       *
*               d-depth to the edge in meters,                      *
*               delta-angle in degrees determining the edge,        *
*               z-height of the observational point in meters,      *
*               k-susceptibility in SI units,                       *
*               f-intensity of the inducing field in nT,            *
*               incl-inclination in degrees of the inducing field,  *
*               sangle-strike angle in degrees (taken clockwise from*
*                      magnetic north to the positive y axis).      *
*                                                                    *
**********************************************************************
        real incl,inclr,k,jy,jz
        pi=4.0*atan(1.0)
c
c       convert degrees to radians
c
        inclr=(incl/180.0)*pi
        sanglr=(sangle/180.0)*pi
        deltr=(delta/180.0)*pi
c
c       calculate jy and jz components of the magnetization,
```

```
c       in order to take into account remanence change jy and jz
c       accordingly
c
        jy=k*f*cos(inclr)*cos(sanglr)
        jz=k*f*sin(inclr)
c
        dif1=y-y0
        dif2=d-z
c
        r=sqrt(dif1*dif1+dif2*dif2)
c
        s=sin(deltr)/(2.0*pi)
c
        p=jz*sin(deltr)+jy*cos(deltr)
        q=jz*cos(deltr)-jy*sin(deltr)
c
        v=alog(r)
        w=atan(dif1/dif2)
c
        yc=s*(-p*v+q*w)
c
        zc=s*(q*v+p*w)
c
        if(l.eq.1) edge=yc
c
        if(l.eq.2) edge=zc
c
        if(l.eq.3) then
                edge=yc*cos(sanglr)*cos(inclr)+zc*sin(inclr)
        end if
        return
        end
```

Test of the edge

* Input data *

Inclination = -60.0
Total inducing field = 58000.0
Strike angle = 30.0
Susceptibility = .0126
Horizontal position of the edge = 100.0
Height of the observational point = .0
Depth to the edge = 1000.0
Angle determinig the edge = 45.0

* Output data *

i	y(i)	ycomp(i)	zcomp(i)	tot(i)
1	-500.000	218.065	-518.470	543.433
2	-450.000	214.691	-517.786	541.380
3	-400.000	211.221	-517.221	539.388
4	-350.000	207.661	-516.786	537.470
5	-300.000	204.019	-516.494	535.640
6	-250.000	200.304	-516.357	533.912
7	-200.000	196.528	-516.386	532.302
8	-150.000	192.705	-516.590	530.824
9	-100.000	188.851	-516.979	529.492
10	-50.000	184.984	-517.558	528.319
11	.000	181.121	-518.331	527.315
12	50.000	177.282	-519.299	526.491
13	100.000	173.486	-520.459	525.853
14	150.000	169.754	-521.808	525.404
15	200.000	166.102	-523.337	525.147
16	250.000	162.548	-525.036	525.080
17	300.000	159.106	-526.894	525.199
18	350.000	155.790	-528.896	525.496
19	400.000	152.609	-531.026	525.963
20	450.000	149.571	-533.268	526.589
21	500.000	146.681	-535.606	527.364

```
      program main
**********************************************************************
*                                                                    *
*                           POLYGON.F77                              *
*                                                                    *
*       Program for testing subroutine for calculating magnetic     *
*             effect from an arbitrary 2D polygonal body.            *
*                                                                    *
*                       Revised: 5 March 89                          *
*                                                                    *
**********************************************************************
       parameter (n=21)
       dimension y(n),ycomp(n),zcomp(n),tot(n),ycor(5),zcor(5)
       real incl,k
       pi=4.0*atan(1.0)
c
c      open output file
c
       open(1,file='polyout',status='fresh')
c
c      generate points on the profile
c
       l=(n-1)/2
       step=30.0
       do 10 i=1,l+1
            ll=l+1-i
            y(i)=-step*ll
10     continue
c
       do 20 i=l+2,n
            ll=i-l-1
            y(i)=step*ll
20     continue
c
c      input parameters
c
       k=4.0*pi*1.0e-3     ! susceptibility in SI units
       sangle=315.0        ! strike angle in degrees
       z=0.0               ! height of the observational point in meters
       incl=-60.0          ! inclination of the inducing field in degrees
       f=58000.0           ! inducing field in nanoTesla
       nc=4                ! number of corners of the body
c
c      coordinates of corners (the coordinates of the first corner
c      are repeated in ycor(5) and zcor(5)
c
       ycor(1)=40.0
       ycor(2)=20.0
       ycor(3)=-60.0
       ycor(4)=-40.0
       ycor(5)=40.0
       zcor(1)=20.0
       zcor(2)=40.0
```

```
      zcor(3)=40.0
      zcor(4)=20.0
      zcor(5)=20.0
c
c     calculate y,z-component and total magnetic field
c
      do 30 i=1,n
      yp=y(i)
      ycomp(i)=polygon(1,nc,yp,z,ycor,zcor,k,f,incl,sangle)
      zcomp(i)=polygon(2,nc,yp,z,ycor,zcor,k,f,incl,sangle)
      tot(i)  =polygon(3,nc,yp,z,ycor,zcor,k,f,incl,sangle)
30    continue
      write(1,100)
100   format(1x,///,20x,'Test of polygonal body',//)
      write(1,101)
101   format(15x,'* Input data *',/)
      write(1,102) incl,f,sangle,k,z
102   format(12x,'Inclination= ',f5.1,/,12x,'Total inducing field= ',
     +      f8.1,/,12x,'Strike angle= ',f5.1,/,12x,'Susceptibility=',
     +      f9.4,/,12x,'Height of the observational point=',f9.1)
      write(1,103) nc
103   format(12x,'Number of corners nc=',i2,/)
      write(1,104)
104   format(12x,'Coordinates of the corners',/)
      write(1,105)
105   format(21x,'y',9x,'z')
      write(1,106) (i,ycor(i),zcor(i),i=1,nc+1)
106   format(12x,i2,2x,f8.3,3x,f8.3)
      write(1,107)
107   format(1x,/,15x,'* Output data *',//)
      write(1,108)
108   format(12x,'i',7x,'y(i)',5x,'ycomp(i)',5x,'zcomp(i)',5x,'tot(i)'
     +          ,/)
      do 888 i=1,n
          write(1,109)  i, y(i),ycomp(i),zcomp(i),tot(i)
888   continue
109   format(8x,i5,3x,f8.3,3x,3g13.6)
      stop
      end
      real function polygon(l,n,y,z,ycor,zcor,k,f,incl,sangle)
***********************************************************************
*                                                                     *
*     Real function polygon calculates y-component,z-component and    *
*     total magnetic field from an arbitrary polygonal body           *
*                                                                     *
*       Input parameters:                                             *
*                       l-switch                                      *
*                       if l=1 then polygon=y-component               *
*                       if l=2 then polygon=z-component               *
*                       if l=3 then polygon=total field,              *
*                                                                     *
*                     n-integer indicating the number of corners,     *
*                     y-horizontal distance in meters along profile,*
*                     z-height of the observational point in meters,*
*                     ycor-n+1 element vector containing y coords of*
*                          the corners (n+1=first) in meters,         *
```

```
*                     zcor-n+1 element vector conatining z coords of*
*                     the corner  (n+1=first) in meters,           *
*                     k-suceptibility of the polygon in SI units,  *
*                     f-total inducing field in nanoTesla,         *
*                     incl-inclinatin of the total inducing field in*
*                     degrees,                                     *
*                     sangle-strike angle in degrees (the angle    *
*                          between magnetic North and positive     *
*                          y axis taken clockwise).                *
*                                                                  *
*********************************************************************
        real k,incl,inclr,jy,jz
        dimension ycor(n+1),zcor(n+1)
        pi=4.0*atan(1.0)
c
c       conversion to radians
c
        inclr=(incl/180.0)*pi
        sanglr=(sangle/180.0)*pi
c
        jy=k*f*cos(inclr)*cos(sanglr)
        jz=k*f*sin(inclr)
c
        s=0.5*pi
c
c
        ycor(n+1)=ycor(1)
        zcor(n+1)=zcor(1)
c
c       calculate p,q
c
        p=0.0
        q=0.0
        do 10 i=1,n
        tet1=atan((y-ycor(i))/(zcor(i)-z))
        tet2=atan((y-ycor(i+1))/(zcor(i+1)-z))
        dtet=tet1-tet2
        z21=zcor(i+1)-zcor(i)
        y12=ycor(i)-ycor(i+1)
       r1=sqrt((ycor(i)-y)*(ycor(i)-y)+(zcor(i)-z)*(zcor(i)-z))
       r2=sqrt((ycor(i+1)-y)*(ycor(i+1)-y)+(zcor(i+1)-z)*(zcor(i+1)-z)
        rr=z21*z21+y12*y12
        q=q+(dtet*z21*y12-z21*z21*alog(r2/r1))/rr
        p=p+(dtet*z21*z21+z21*y12*alog(r2/r1))/rr
10      continue
c
c       calculate y-component
c
        yc=s*(jy*p+jz*q)
c
c       calculate z-component
c
        zc=s*(jy*q-jz*p)
c
        if(l.eq.1) polygon=yc
c
```

```
      if(l.eq.2) polygon=zc
c
c     calculate total field
c
      if(l.eq.3) then
      polygon=yc*cos(sanglr)*cos(inclr)+zc*sin(inclr)
      endif
      return
      end
```

Test of polygonal body

* Input data *

Inclination= -60.0
Total inducing field= 58000.0
Strike angle= 315.0
Susceptibility= .0126
Height of the observational point= .0
Number of corners nc= 4

Coordinates of the corners

	y	z
1	40.000	20.000
2	20.000	40.000
3	-60.000	40.000
4	-40.000	20.000
5	40.000	20.000

* Output data *

i	y(i)	ycomp(i)	zcomp(i)	tot(i)
1	-300.000	.362251	2.04797	-1.64552
2	-270.000	.385721	2.56263	-2.08293
3	-240.000	.385783	3.29627	-2.71826
4	-210.000	.320345	4.39010	-3.68868
5	-180.000	.730292E-01	6.11598	-5.27077
6	-150.000	-.715840	9.03351	-8.07633
7	-120.000	-3.35054	14.3283	-13.5933
8	-90.000	-13.4254	23.5430	-25.1354
9	-60.000	-49.2251	22.5216	-36.9079
10	-30.000	-59.5641	-43.0547	16.2274
11	.000	-10.9710	-68.5153	55.4571
12	30.000	47.1270	-50.8180	60.6716
13	60.000	37.4906	5.40418	8.57474
14	90.000	15.9083	9.23870	-2.37653
15	120.000	8.17640	6.86573	-3.05511
16	150.000	4.86623	4.98269	-2.59467
17	180.000	3.19244	3.72107	-2.09385
18	210.000	2.24139	2.86760	-1.69096
19	240.000	1.65363	2.27123	-1.38229
20	270.000	1.26705	1.84076	-1.14617
21	300.000	.999987	1.52075	-.963456

```
      program main
**********************************************************************
*                                                                    *
*                         VERPRISM.F77                               *
*                                                                    *
*        Program for testing subroutine for calculating the total    *
*        magnetic field from a bottomless vertical prism.            *
*                         Revised: 4 March 89                        *
*                                                                    *
**********************************************************************
        parameter (n=9)
        real k,incl
        external zeta
        dimension x(n),y(n),tfield(n,n)
        pi=4.0*atan(1.0)
c
        open(1,file='verout',status='fresh')
c
c       input parameters
c
        f=58000.0       ! total inducing field in nanoTesla
        incl=-60.0      ! inclination of the inducing field in degrees
        sangle=30.0     ! strike angle in degrees
        k=4.0*pi*1.0e-3 ! susceptibility of the prism in SI units
        d=20.0          ! depth to the top of the prism in meters
        x1=-10.0        ! x-coordinate of the edge in meters (the left
c                         one to the middle of the prism)
        x2=10.0         ! x-coordinate of the edge in meters (the right
c                           one to the middle of the prism)
        y1=-10.0        ! y-coordinate of the edge in meters (the left
c                           one to the middle of the prism)
        y2=10.0         ! y-coordinate of the edge in meters (the right
c                           one to the middle of the prism)
        z=0.0           ! height of the observational point in meters
c
c       determine (x,y) points at which the total magnetic field will
c                             be calculated
        step=15.0
c
        xmin=-60.0
        ymin=-60.0
c
        do 10 i=1,n
                x(i)=xmin+step*(i-1)
10      continue
c
        do 20 j=1,n
                y(j)=ymin+step*(j-1)
20      continue
c
c       calculate the total field at (x,y,z) points
```

```
c
      do 50 i=1,n
            xp=x(i)
            do 50 j=1,n
            yp=y(j)
      call vprism(x1,x2,y1,y2,d,k,f,incl,sangle,xp,yp,z,totf,
     +                       zeta)
      tfield(i,j)=totf
50    continue
c
c     print the results
c
      write(1,100)
100   format(1x,//,20x,'Test of the vertical prism',// )
      write(1,200)
200   format(15x,'* Input data *',//)
      write(1,300) incl,f,sangle,k,d,x1,x2,y1,y2,z
300   format(12x,'Inclination = ',f5.1,/,12x,'Total field = ',
     +      f8.1,/,12x,'Strike angle = ',f5.1,/,12x,
     +      'Susceptibility = ',f9.5,/,12x,'Depth = ',
     +      f5.1,/,12x,'x1 =',f5.1,/,12x,'x2 = ',f5.1,/,12x,
     +      'y1 = ',f5.1,/,12x,'y2 = ',f5.1,/,12x,'z = ',f5.1,// )
c
      write(1,400)
400   format(15x,'* Output data *',//)
      write(1,600)
600   format(28x,'Total field in nT'//)
      write(1,'(10x,9i7,/)') (j,j=1,n)
      write(1,'(11x,9f7.1)') (y(j),j=1,n)
      write(1,'(/)')
      write(1,'(1x,i3,1x,f5.1,1x,9f7.1)') (i,x(i),(tfield(i,j),j=1,n
     +      i=n,1,-1)
      stop
      end
c
     subroutine vprism(x1,x2,y1,y2,h,k,f,incl,sangle,x,y,z,totf,zeta
************************************************************************
*                                                                      *
*     Subroutine vprism calculates the total magnetic field from      *
*        uniformly magnetized bottomless vertical prism.               *
*                                                                      *
*     Input parameters:                                                *
*           x1 - x coordinate of the left edge of the prism in m,*
*           x2 - x coordinate of the right edge of the prism in m*
*           y1 - y coordinate of the left edge of the prism in m,*
*           y2 - y coordinate of the right edge of the prism in m*
*           h - depth to the top of the prism in m                     *
*           k - susceptibility of the prism in SI units,               *
*           f - total inducing field in nT,                            *
*           incl - inclination of the inducing field in degrees, *
*           sangle - strike angle in degrees (an angle from            *
*                    magnetic North to positive Y axis taken           *
*                    clockwise),                                       *
*           x,y,z - coordinates of the observational point in m, *
```

```
*                                                                         *
*       Output parameters:                                                *
*                       totf - total magnetic field from the prism        *
*                              at (x,y,z) point in nanoTesla.             *
*                                                                         *
*        Required external subroutines:                                   *
*                                    real function zeta.                  *
*                                                                         *
*        Function zeta is an implementation of the equation (10).         *
*       (See: Bhattacharyya, B.,K., Magnetic anomalies due to prism-      *
*            shaped bodies with arbitrary polarization,Geophysics,        *
*             vol.29, No.4, pp. 517-531.)                                 *
*                                                                         *
***************************************************************************
      real k,incl,inclr,lt,mt,nt,la,ma,na,m
      external zeta
      pi=4.0*atan(1.0)
c
c     conversion to radians
c
      sanglr=(sangle/180.0)*pi
      inclr=(incl/180.0)*pi
c
c     find lt,mt,nt,la,ma,na
c
      lt=cos(inclr)*sin(sanglr)
      mt=cos(inclr)*cos(sanglr)
      nt=sin(inclr)
c
      la=lt
      ma=mt
      na=nt
c
      alfa12=lt*ma+mt*la
      alfa13=lt*na+nt*la
      alfa23=mt*na+nt*ma
c
      gamma1=la*lt
      gamma2=ma*mt
      gamma3=na*nt
c
      m=k*f/(4.0*pi)
      alfal=x1-x
      alfau=x2-x
      betal=y1-y
      betau=y2-y
c
c     calculate the total field
c
      totf=(zeta(alfau,betau,alfa23,alfa13,alfa12,gamma1,gamma2,
   +       gamma3,h,z,m)+
   +       zeta(alfal,betal,alfa23,alfa13,alfa12,gamma1,gamma2,
   +       gamma3,h,z,m))-
   +      (zeta(alfal,betau,alfa23,alfa13,alfa12,gamma1,gamma2,
```

```
     +        gamma3,h,z,m)+
     +        zeta(alfau,betal,alfa23,alfa13,alfa12,gamma1,gamma2,
     +        gamma3,h,z,m))
      return
      end
c
c     Real function zeta
c
      real function zeta(alfav,betav,alfa23,alfa13,alfa12,gamma1,
     +                   gamma2,gamma3,h,z,m)
      real m
c
      r0r0=alfav*alfav+betav*betav+(h-z)*(h-z)
      r0=sqrt(r0r0)
      v1=(r0-alfav)/(r0+alfav)
      v2=(r0-betav)/(r0+betav)
      v3=(r0+h-z)
c
      w=alfav*betav
      w1=alfav*alfav+r0*(h-z)+(h-z)*(h-z)
      w2=r0r0+r0*(h-z)-alfav*alfav
      w3=r0*(h-z)
c
      zeta=m*(0.5*alfa23*alog(v1)+0.5*alfa13*alog(v2)-
     +        alfa12*alog(v3)-gamma1*atan(w/w1)-
     +        gamma2*atan(w/w2)+gamma3*atan(w/w3))
      return
      end
```

Test of the vertical prism

* Input data *

Inclination = -60.0
Total field = 58000.0
Strike angle = 30.0
Susceptibility = .01257
Depth = 20.0
x1 =-10.0
x2 = 10.0
y1 = -10.0
y2 = 10.0
z = .0

* Output data *

Total field in nT

		1	2	3	4	5	6	7	8	9
		-60.0	-45.0	-30.0	-15.0	.0	15.0	30.0	45.0	60.0
9	60.0	-.7	-.4	.4	1.6	3.1	4.2	4.5	4.1	3.4
8	45.0	-1.1	-.8	.2	2.6	5.7	7.6	7.3	5.9	4.5
7	30.0	-1.6	-1.5	-.3	4.4	12.0	15.0	12.0	8.2	5.6
6	15.0	-2.2	-2.6	-1.6	6.8	25.5	28.5	17.7	10.1	6.2
5	.0	-2.5	-3.4	-3.6	4.5	29.2	32.4	18.1	9.7	5.8
4	-15.0	-2.6	-3.6	-4.6	-1.9	9.4	15.0	10.8	6.7	4.4
3	-30.0	-2.3	-3.2	-4.0	-3.6	-.5	3.0	4.0	3.5	2.8
2	-45.0	-2.0	-2.5	-3.0	-2.9	-1.8	-.2	1.0	1.4	1.5
1	-60.0	-1.6	-1.9	-2.1	-2.1	-1.6	-.9	-.1	.4	.7

```
      program main
*********************************************************************
*                                                                   *
*                            DIPPRISM.F77                           *
*                                                                   *
*       Program for testing subroutine for calculating the total    *
*       magnetic field from a finite dipping prism.                 *
*                            Revised: 2 March 89                    *
*                                                                   *
*********************************************************************
      parameter (n=9)
      real incl,k
      dimension x(n),y(n),tfield(n,n)
      pi=4.0*atan(1.0)
c
c     open output file
c
      open(1,file='dipout',status='fresh')
c
c     input parameters
c
      tf=58000.0       ! total inducing field in nanoTesla
      incl=-60.0       ! inclination of the inducing field in degrees
      sangle=30.0      ! strike angle in degrees
      k=4.0*pi*1.0e-3 ! susceptibility of the prism in SI units
      beta=0.0         ! rotation angle in degrees
      fi=135.0         ! dipping angle of the prism in degrees
      z1=20.0          ! depth to the top of the prism in meters
      z2=30.0          ! depth to the bottom of the prism in meters
      z=0.0            ! height of the observational point in meters
      x1=-10.0         ! x-coordinate in meters of the left top egde
c                          of the prism
      x2=10.0          ! x-coordinate in meters of the right top edge
c                          of the prism
      y1=-10.0         ! y-coordinate in meters of the edge (to the
c                          left from the origin)
      y2=10.0          ! y-coordinate in meters of the edge (to the
c                          right from the origin)
c
c     determine (x,y) points at which the total magnetic field will
c                    be calculated
c
      step=15.0
      xmin=-60.0
      ymin=-60.0
      do 5 i=1,n
              x(i)=xmin+(i-1)*step
              y(i)=ymin+(i-1)*step
5     continue
c
c     calculate total magnetic field at (x,y,z) points
c
      do 10 i=1,n
```

```
              xp=x(i)
              do 10 j=1,n
              yp=y(j)
              call dprism(x1,x2,y1,y2,z1,z2,fi,k,tf,sangle,incl,
     +                    xp,yp,z,beta,totf)
              tfield(i,j)=totf
10       continue
c
c        print the results
c
        write(1,100)
100     format(1x,//,20x,'Test of 3D dipping prism',//)
        write(1,101)
101     format(15x,'* Input data *',/)
        write(1,102) incl,f,sangle,k,beta,fi,z,x1,y1,z1,x2,y2,z2
102     format(12x,'Inclination = ',f5.1,/,12x,'Total inducing field= '
     +    ,f8.1,/,12x,'Strike angle = ',f5.1,/,12x,'Susceptibility= ',
     +     f9.5,/,12x,'Rotation angle = ',f5.1,/,12x,'Dipping angle= ',
     +     f5.1,/,12x,'Height of the observational points= ',f8.1,/,
     +     12x,'x1=',f5.1,2x,'y1=',f5.1,2x,'z1=',f5.1,/,12x,
     +     'x2=',f5.1,2x,'y2=',f5.1,2x,'z2=',f5.1,///)
c
        write(1,103)
103     format(15x,'* Output data * ',//)
        write(1,575)
575     format(1x,//,28x,'Total field in nT',/)
        write(1,'(11x,9i7  )') (j, j=1,n)
        write(1,'( /)')
        write(1,'(11x,9f7.1)') (y(j),j=1,n)
        write(1,'(/)')
        write(1,'(1x,i3,f6.1,1x,9f7.1)') (i,x(i),(tfield(i,j),j=1,n),
     +  i=n,1,-1)
        stop
        end
c
        subroutine dprism(x1,x2,y1,y2,z1,z2,fi,k,tf,sangle,incl,x,y,z,
     +                   beta,totf)
******************************************************************
*                                                                *
*    Subroutine dprism calculates the total magnetic field from  *
*               uniformly magnetized dipping prism.              *
*                                                                *
*    Input parameters:                                           *
*            x1 - x coordinate of the top left edge in meters,   *
*            x2 - x coordinate of the top right edge in meters,  *
*            y1 - y coordinate of the egde in meters ( to the left *
*                 from the midlle of the prism),                 *
*            y2 - y coordinate of the edge in meters ( to the right *
*                 from the middle of the prism),                 *
*            z1 - depth to the top of the prism in meters,       *
*            z2 - depth to the bottom of the prism in meters,    *
*            fi - dipping of the prism (angle in degrees from +ve X *
*                 to the dipping side of the prism taken clockwise),*
*            k  - susceptibility of the prism in SI units,       *
*            tf - total inducing field in nanoTesla,             *
*            sangle - strike angle (angle in degrees from magnetic  *
```

```
*                  North to positive Y axis taken clockwise),       *
*         incl - inclination of the inducing field in degrees,      *
*                (incl<0 for Southern Hemisphere),                  *
*         x - x coordinate of the observational point in meters,   *
*         y - y      "          "           "           "      ,   *
*         z - z      "          "           "           "      ,   *
*         beta - angle in degrees from +ve X axis in the prism      *
*                coordinate system to +ve X' axis of the            *
*                chosen system of coordinate (taken clockwise).     *
*                                                                    *
*    Output parameter:                                              *
*                  totf - total magnetic field at the (x,y,z)       *
*                         observational point in nanoTesla.         *
*                                                                    *
*    Note:                                                          *
*        This subroutine calculates the total magnetic field from   *
*        the dipping prism in prism coordinates i.e. when X & Y are *
*        parallel to the sides of the prism. In order to calculate  *
*        the anomalous field from the prism in diffrent system of   *
*        coordinates it is necessary to rotate X and Y axes by the  *
*        angle beta (clockwise) . The vertical sides of the prism   *
*        have to perpendicular to the strike. The remanent          *
*        magnetization was not taken into account.                  *
*                                                                    *
* (See:Hjelt,S.,E.,1972,Magnetostatic anomalies of dipping prism,   *
*                        Geoexploration,10,pp239-254)               *
*                                                                    *
*                                                                    *
*      ---------------------------------------------> x             *
*      |                                                            *
*      |                                                            *
*      |        (x1,y1,z1)                   (x2,y1,z1)             *
*      |          x------------------------------x                  *
*      |          |                              |                  *
*      |          |                              |                  *
*      |          |                              |                  *
*      |          x------------------------------x                  *
*      |        (x1,y2,z1)                   (x2,y2,z1)             *
*      |                                                            *
*      |  y                                                         *
*      V                                                            *
*                     Top view                                      *
*                                                                    *
*                                                                    *
*      ----------------------------------------------> x            *
*      |                                                            *
*      |                                                            *
*      |-z1.........x-------------------------x                     *
*      |           /                         /                      *
*      |          /                         /                       *
*      |         /                         / (fi)                   *
*      |        /                         /                         *
*      |-z2....x-------------------------x                          *
*      |                                                            *
*      |  z                                                         *
*      V                                                            *
```

```
*                              Side view                              *
*                                                                     *
***********************************************************************
      real j1,j2,j3,k,incl,jx,jy,jz,inclr
      dimension u(2),v(2),w(2),f(8),g(8),a(8),b(8),s(8),t(8)
      pi=4.0*atan(1.0)
c
c     conversion to radians
c
      sanglr=(sangle/180.0)*pi
      inclr=(incl/180.0)*pi
      betar=(beta/180.0)*pi
      fir=(fi/180.0)*pi
c
c     rotation of X & Y
c
      x11=x*cos(betar)-y*sin(betar)
      y11=x*sin(betar)+y*cos(betar)
c
c     calculation of jx,jy,jz
c
      jx=k*tf*cos(inclr)*sin(sanglr)/(4.0*pi)
      jy=k*tf*cos(inclr)*cos(sanglr)/(4.0*pi)
      jz=k*tf*sin(inclr)/(4.0*pi)
c
c     calculation of j1,j2,j3
c
      j1=jx*sin(fir)-jz*cos(fir)
      j2=jy
      j3=jz
c
c     calculation of u,v,w
c
      h=z2-z1
      u(1)=x11-x2
      u(2)=x11-x1
      v(1)=y11-y2
      v(2)=y11-y1
      w(1)=z-z2
      w(2)=z-z1
c
c     calculation of f's
c
      f1u=0.0
      f1v=0.0
      f1w=0.0
      f2u=0.0
      f2v=0.0
      f2w=0.0
      m=1
      do 10 i=1,2
         do 10 j=1,2
         do 10 l=1,2
         if(l.eq.1) u(i)=u(i)-h/tan(fir)
         if((abs(u(i))-0.001).lt.0.0) u(i)=0.001
         if((abs(v(j))-0.001).lt.0.0) v(j)=0.001
```

```
            if((abs(w(l))-0.001).lt.0.0) w(l)=0.001
            r=sqrt(u(i)*u(i)+v(j)*v(j)+w(l)*w(l))
            p=u(i)*cos(fir)+w(l)*sin(fir)
            q=u(i)*sin(fir)-w(l)*cos(fir)
            if((abs(q)-0.001).lt.0.0) q=0.001
            if((abs(p)-0.001).lt.0.0) p=0.001
            f(m)=atan((v(j)*p)/(q*r))
            g(m)=atan((v(j)*v(j)*cos(fir)-w(l)*q)/(v(j)*sin(fir)*r))
            b(m)=atan((v(j)*u(i))/(w(l)*r))
            s(m)=alog(u(i)+r)
            t(m)=alog(v(j)+r)
            zm=p+r
            if(zm.le.0.0) zm=1.0e-6
            a(m)=alog(zm)
            ijl=i+j+l
            if(ijl.eq.4.or.ijl.eq.6) go to 7
6            f(m)=-f(m)
            g(m)=-g(m)
            b(m)=-b(m)
            s(m)=-s(m)
            t(m)=-t(m)
            a(m)=-a(m)
7            continue
            if(l.eq.1) u(i)=u(i)+h*cos(fir)/sin(fir)
            m=m+1
10          continue
c
         do 20 ik=1,8
               flu=flu+f(ik)
               flv=flv+g(ik)
               flw=flw+b(ik)
               f2u=f2u+s(ik)
               f2v=f2v+t(ik)
               f2w=f2w+a(ik)
20       continue
c
c        calculation of t factors
c
         t11=cos(fir)*f2v-sin(fir)*flu
         t12=sin(fir)*f2w
         t13=f2v
         t21=f2w
         t22=flv
         t23=f2u
         t31=cos(fir)*flu+sin(fir)*f2v
         t32=f2u-cos(fir)*f2w
         t33=-flw
c
c        hx,hy,hz components
c
         hx=j1*t11+j2*t12+j3*t13
         hy=j1*t21+j2*t22+j3*t23
         hz=j1*t31+j2*t32+j3*t33
c
c
         totf=hx*cos(inclr)*sin(sanglr)+hy*cos(sanglr)*cos(inclr)+
```

```
     +        hz*sin(inclr)
c
           return
           end
```

Test of 3D dipping prism

* Input data *

Inclination = -60.0
Total inducing field = .0
Strike angle = 30.0
Susceptibility = .01257
Rotation angle = .0
Dipping angle = 135.0
Height of the observational points = .0
x1=-10.0 y1=-10.0 z1= 20.0
x2= 10.0 y2= 10.0 z2= 30.0

* Output data *

Total field in nT

		1	2	3	4	5	6	7	8	9
		-60.0	-45.0	-30.0	-15.0	.0	15.0	30.0	45.0	60.0
9	60.0	-.3	-.4	-.4	-.3	-.1	.1	.2	.2	.1
8	45.0	-.4	-.6	-.6	-.4	.2	.6	.6	.4	.3
7	30.0	-.6	-.9	-1.1	-.3	1.7	2.7	1.8	.9	.4
6	15.0	-.7	-1.3	-1.9	.3	8.4	9.2	4.2	1.6	.6
5	.0	-.8	-1.7	-2.9	.1	15.5	15.7	5.7	1.8	.6
4	-15.0	-.8	-1.6	-3.1	-2.7	5.5	7.9	3.3	1.1	.4
3	-30.0	-.7	-1.2	-2.3	-3.0	-1.6	.3	.4	.2	.1
2	-45.0	-.5	-.8	-1.3	-1.8	-1.7	-1.0	-.5	-.2	-.1
1	-60.0	-.3	-.5	-.8	-1.0	-1.0	-.8	-.5	-.3	-.2

```
      program main
*******************************************************************
*                                                                 *
*                         TRIANEL.F77                             *
*                                                                 *
*     Program for testing the subroutine for calculating total    *
*     magnetic effect from an arbitrarily oriented triangular     *
*                         element.                                *
*                                                                 *
*  The subroutine was tested on the example of 3D dipping prism   *
*  The surface of this prism was divided into 12 triangular       *
*  elements and contributions from them were added up. The prism  *
*  and the way of dividing it into triangular elements are given  *
*  on Fig.A1-A2.                                                  *
*                                                                 *
*                         Revised: 9 March 89                     *
*                                                                 *
*******************************************************************
      parameter (n=36,nf=3,lq=9,nfacet=12)
      external funct
      dimension coord(n,nf),a(nf,nf),tfield(lq,lq),x(lq),y(lq)
      integer facet
      real k,incl,inclr,jx,jy,jz
c
c     generate matrix containing coords of each triangular element
c
      data ((coord(i,j),j=1,nf),i=1,n)/
c             x      y      z       vertex number
    +      10.0,-10.0, 20.0,     !    1
    +     -10.0,-10.0, 20.0,     !    2
    +     -10.0, 10.0, 20.0,     !    3
c
    +     -10.0, 10.0, 20.0,     !    3
    +      10.0, 10.0, 20.0,     !    4
    +      10.0,-10.0, 20.0,     !    1
c
    +     -20.0,-10.0, 30.0,     !    6
    +     -10.0,-10.0, 20.0,     !    2
    +      10.0,-10.0, 20.0,     !    1
c
    +      10.0,-10.0, 20.0,     !    1
    +       0.0,-10.0, 30.0,     !    5
    +     -20.0,-10.0, 30.0,     !    6
c
    +      10.0, 10.0, 20.0,     !    4
    +     -10.0, 10.0, 20.0,     !    3
    +     -20.0, 10.0, 30.0,     !    7
c
    +     -20.0, 10.0, 30.0,     !    7
    +       0.0, 10.0, 30.0,     !    8
    +      10.0, 10.0, 20.0,     !    4
c
    +       0.0,-10.0, 30.0,     !    5
```

```
     +       10.0,-10.0, 20.0,    !    1
     +       10.0, 10.0, 20.0,    !    4
c
     +       10.0, 10.0, 20.0,    !    4
     +        0.0, 10.0, 30.0,    !    8
     +        0.0,-10.0, 30.0,    !    5
c
     +      -20.0, 10.0, 30.0,    !    7
     +      -10.0, 10.0, 20.0,    !    3
     +      -10.0,-10.0, 20.0,    !    2
c
     +      -10.0,-10.0, 20.0,    !    2
     +      -20.0,-10.0, 30.0,    !    6
     +      -20.0, 10.0, 30.0,    !    7
c
     +      -20.0, 10.0, 30.0,    !    7
     +      -20.0,-10.0, 30.0,    !    6
     +        0.0,-10.0, 30.0,    !    5
c
     +        0.0,-10.0, 30.0,    !    5
     +        0.0, 10.0, 30.0,    !    8
     +      -20.0, 10.0, 30.0/    !    7
c
c        open output file
c
      open(1,file='trianout',status='fresh')
c
c     input parameters
c
      pi=4.0*atan(1.0)
      f=58000.0         ! total inducing field in nanoTesla
      incl=-60.0        ! inclination of the inducing field in degrees
      sangle=30.0       ! strike angle in degrees (clockwise from Nmag
c                         to positive y axis)
      k=4.0*pi*1.0e-3 ! susceptibility contrast in SI units
      z=0.0             ! height of the observational point in meters
c
c     conversion to radians
c
      inclr=(incl/180.0)*pi
      sanglr=(sangle/180.0)*pi
c
c     determine unit vector in the direction of the inducing field
c
      ufx=cos(inclr)*sin(sanglr)
      ufy=cos(inclr)*cos(sanglr)
      ufz=sin(inclr)
c
c     determine vector of magnetization(in order to take into account
c     remanent magnetization,jx,jy,jz should be changed accordingly)
      jx=ufx*k*f
      jy=ufy*k*f
      jz=ufz*k*f
c
c     determine (x,y) points at which the total magnetic field will
c     be calculated
```

```
c
        step=15.0
        xmin=-60.0
        ymin=-60.0
        do 5 i=1,lq
            x(i)=xmin+(i-1)*step
            y(i)=ymin+(i-1)*step
5       continue
c
c       calculate the total field
c
        do 20 i=1,lq
              xp=x(i)
              do 20 j=1,lq
                    yp=y(j)
c
c       start calculating and summing up contribution from each element
c
        sumf=0.0
        do 30 facet=1,nfacet
                 ip=3*facet-2
                 is=3*facet-1
                 ik=3*facet
c
c       transfer coordinates of the element vertices into a(3,3)
c
        do 35 jk=1,nf
              a(1,jk)=coord(ip,jk)
              a(2,jk)=coord(is,jk)
              a(3,jk)=coord(ik,jk)
35      continue
c
c       call the subroutine for calculating the magnetic field from
c       a triangular element
c
        call field(funct,a,jx,jy,jz,ufx,ufy,ufz,xp,yp,z,totf)
c
        sumf=sumf+totf
30      continue
        tfield(i,j)=sumf
20      continue
c
c       print the results
c
        write(1,900)
900     format(1x,//,20x,'Test of dipping prism ',//)
        write(1,910)
910     format(15x,'* Input data *',//)
        write(1,920) incl,f,sangle,k,z
920     format(12x,'Inclination = ',f5.1,/,12x,'Total inducing field ='
    +            ,f8.1,/,12x,'Strike angle = ',f5.1,/,12x,
    +            'Susceptibility contrast in SI = ',f9.5,/,12x,
    +            'Height of the observational points in m = ', f8.1,//)
c
        write(1,930)
930     format(15x,'* Output data *',//)
```

```
      write(1,940)
940   format(1x,//,28x,'Total field in nT',//)
      write(1,950) (j,j=1,lq)
950   format(10x,9i7,/)
      write(1,960) (y(j),j=1,lq)
960   format(11x,9f7.1,/)
      write(1,970) (i, x(i),(tfield(i,j),j=1,lq) ,i=lq,1,-1)
970   format(1x,i2,1x,f5.1,1x,9f7.1)
      stop
      end
c
      subroutine field(funct,a,jx,jy,jz,mx,my,mz,x,y,z,totf)
**********************************************************************
*                                                                    *
*      Subroutine field calculates the magnetic effect from one      *
*             arbitrary oriented triangular element.                 *
*                                                                    *
*      Input parameters:                                             *
*                         a - an 3x3 matrix containing coords of the *
*                             vertices of the element i.e.           *
*                                x1,y1,z1                            *
*                                x2,y2,z2                            *
*                                x3,y3,z3,                           *
*                       the vertices are taken in anticlocwise order *
*                       when looking from outside body,              *
*                                                                    *
*                       (jx,jy,jz) - components of the magnetization *
*                                    vector in SI units,             *
*                                                                    *
*                       (mx,my,mz) - components of the unit vector   *
*                                    in the direction of the total   *
*                                    inducing field,                 *
*                                                                    *
*                       (x,y,z) - point at which the magnetic field  *
*                                 is be calculated.                  *
*                                                                    *
*     Output parameter:                                              *
*                      totf - total magnetic field in nanoTesla  at  *
*                             (x,y,z) point from the element.        *
*                                                                    *
*     Requied : external function funct                              *
*               subroutines:vecab,vecpr,transg,gauss.                *
*                                                                    *
**********************************************************************
      external funct
      real jx,jy,jz,mx,my,mz,nx,ny,nz,lx,ly,lz
      parameter (n=3)
      dimension a(n,n),vec(n),tr(n,n),b(n,n),p(n)
      pi=4.0*atan(1.0)
c
      p(1)=x
      p(2)=y
      p(3)=z
c
c
c
```

```
c     find transformation matrix tr and transform coordinates to the
c     (p,q,r) system

      call transg(p,a,b,tr,vec)
c
c     find a, b, c
c
      ab=mx*tr(1,1)+my*tr(1,2)+mz*tr(1,3)
      bb=mx*tr(2,1)+my*tr(2,2)+mz*tr(2,3)
      cb=mx*tr(3,1)+my*tr(3,2)+mz*tr(3,3)
c
c     calculate the integral I (see eq.81)
c
      call gauss(funct,b(1,1),b(2,1),b(3,1),b(1,2),b(2,2),b(1,3),
     +           ab,bb,cb,s)
c
c     calculate the total field (see eq.74)
c
      totf=-(jx*vec(1)+jy*vec(2)+jz*vec(3))*s/(4.0*pi)
c
      return
      end
c
      subroutine vecab(xp,yp,zp,xe,ye,ze,cosx,cosy,cosz)
*********************************************************************
*    Subroutine vecab calculates an unit vector in the              *
*      direction  (xp,yp,zp)---->(xe,ye,ze).                        *
*                                                                   *
*    Input parameters:                                              *
*                       (xp,yp,zp)-coordinates of the begining of the *
*                                  vector,                          *
*                       (xe,ye,ze)-coordinates of the end of vector.  *
*                                                                   *
*    Output parameters:                                             *
*                        (cosx,cosy,cosz)-components of the unit    *
*                                          vector.                  *
*********************************************************************
      dx=xe-xp
      dy=ye-yp
      dz=ze-zp
      d=sqrt(dx*dx+dy*dy+dz*dz)
      cosx=dx/d
      cosy=dy/d
      cosz=dz/d
      return
      end
c
      subroutine vecpr(a1,a2,a3,b1,b2,b3,c1,c2,c3)
********************************************************************
*  Subroutine vecpr calculates vector product of two vectors a & b.*
*                                                                  *
*  Input parameters:                                               *
*                       (a1,a2,a3)-components of the first vector,  *
*                       (b1,b2,b3)-components of the second vector. *
*                                                                  *
*  Output parameters:                                              *
```

```
*                         (c1,c2,c3)-components of the vector product. *
*******************************************************************
      x=a2*b3-a3*b2
      y=a3*b1-a1*b3
      z=a1*b2-a2*b1
      d=sqrt(x*x+y*y+z*z)
      c1=x/d
      c2=y/d
      c3=z/d
      return
      end
c
      subroutine multp(a,xi,yi,zi,xo,yo,zo)
********************************************************************
*   Subroutine  multp performs multiplication y=Ax                 *
*                                                                  *
*   Input parameters:                                              *
*                   a-an nxn matrix,                               *
*                   (xi,yi,zi)-components of the vector x          *
*                                                                  *
*   Output parameters:                                             *
*                   (xo,yo,zo)-components of the vector y.         *
********************************************************************
      parameter (n=3)
      dimension a(n,n),vin(n),vtr(n)
      vin(1)=xi
      vin(2)=yi
      vin(3)=zi
      do 10 i=1,n
           s=0.0
           do 15 j=1,n
              s=s+a(i,j)*vin(j)
15          continue
           vtr(i)=s
10    continue
      xo=vtr(1)
      yo=vtr(2)
      zo=vtr(3)
      return
      end
c
      subroutine transg(p,a,b,tr,vec)
*******************************************************************
*  Subroutine transg calculates the transformation matrix from    *
*  the (xyz) system to (pqr) system and performs transformation.  *
*                                                                 *
*  Input parameters:                                              *
*                  (p(1),p(2),p(3))-(x,y,z) coords of the point,  *
*              | a(1,1)=x1, a(1,2)=y1, a(1,3)=z1|                 *
*     a(nxn)=  | a(2,1)=x2, a(2,2)=y2, a(2,3)=z2|  coords of the ver *
*              | a(3,1)=x3, a(3,2)=y3, a(3,3)=Z3|     tices.      *
*                                                                 *
*  Output parameters:                                             *
*              | b(1,1)=p1, b(1,2)=q1, b(1,3)=r1|                 *
*     b(nxn)=  | b(2,1)=p2, b(2,2)=q2, b(2,3)=r2|  coords of the ver *
*              | b(3,1)=p3, b(3,2)=q3, b(3,3)=r3|  tices in (p,q,r)  *
```

```
*                                                                    *
*                |tr(1,1)=lx,  tr(1,2)=ly,  tr(1,3)=lz|              *
*     tr(nxn)=|tr(2,1)=mx,  tr(2,2)=my,  tr(2,3)=mz| transformation  *
*                |tr(3,1)=nx,  tr(3,2)=ny,  tr(3,3)=nz|     matrix   *
*                                                                    *
*               (vec(1),vec(2),vec(3))-temporary vector              *
**********************************************************************
      parameter (n=3)
      dimension p(n),vec(n),a(n,n),b(n,n),tr(n,n)
      real lx,ly,lz,nx,ny,nz,mx,my,mz
c
c     find vector l=(lx,ly,lz) parllel to the side 13
c
      call vecab(a(1,1),a(1,2),a(1,3),a(3,1),a(3,2),a(3,3),lx,ly,lz)
c
c     find vector t=(tx,ty,tz) parallel to the side 12
c
      call vecab(a(1,1),a(1,2),a(1,3),a(2,1),a(2,2),a(2,3),tx,ty,tz)
c
c     find vector n=(nx,ny,nz) perpendicular to l & t vectors
c
      call vecpr(lx,ly,lz,tx,ty,tz,nx,ny,nz)
c
c     find vector m=(mx,my,mz) perpendicular to l & n vectors
c
      call vecpr(nx,ny,nz,lx,ly,lz,mx,my,mz)
c
c     form transformation matrix
c
      tr(1,1)=lx
      tr(1,2)=ly
      tr(1,3)=lz
      tr(2,1)=mx
      tr(2,2)=my
      tr(2,3)=mz
      tr(3,1)=nx
      tr(3,2)=ny
      tr(3,3)=nz
c
c     transform point (x,y,z)
c
      call  multp(tr,p(1),p(2),p(3),vec(1),vec(2),vec(3))
c
c     transform points 1,2 and 3
c
      call multp(tr,a(1,1),a(1,2),a(1,3),b(1,1),b(1,2),b(1,3))
      call multp(tr,a(2,1),a(2,2),a(2,3),b(2,1),b(2,2),b(2,3))
      call multp(tr,a(3,1),a(3,2),a(3,3),b(3,1),b(3,2),b(3,3))
c
c     shift the origin
c
      do 10 j=1,n
             do 10 i=1,n
                     b(i,j)=b(i,j)-vec(j)
10    continue
      vec(1)=nx
```

```
      vec(2)=ny
      vec(3)=nz
      return
      end
c
      subroutine gauss(funct,p1,p2,p3,q1,q2,r1,aa,bb,cc,s)
*********************************************************************
*                                                                   *
*    Subroutine gauss calculates integral I (see eq.81) using       *
*    a 10-point Gauss-Legendre integration formula.                 *
*                                                                   *
*    Handbook of Mathematical Constants-Abramowitz & Stegun-p887,   *
*    25.4.30 and p.916 table 25.4.                                  *
*                                                                   *
*      Input parameters:                                            *
*                    funct-function to be integrated (external),    *
*                    p1,p2,p3,q1,q2,r1,aa,bb,cc-parameters of the   *
*                    function (see eq.81),where q1,q2 are the limits*
*                    of integration.                                *
*                                                                   *
*       Output parameters:                                          *
*                        s-returned value of the integral.          *
*                                                                   *
*********************************************************************
      external funct
      dimension zero(5),coeff(5)
      data zero /0.14887434,0.43339539,0.67940957,0.86506337,
     +          0.97390653/
     +   coeff /0.29552422,0.26926672,0.21908636,0.14945135,
     +          0.06667134/
c
      a=q1
      b=q2
      c1=(b-a)*0.5
      c2=(a+b)*0.5
      y=0.0
c
      do 100 i=1,5
             c3=c1*zero(i)
      y=y+c1*coeff(i)*(funct(p1,p2,p3,q1,q2,r1,aa,bb,cc,c2+c3)
     +               +funct(p1,p2,p3,q1,q2,r1,aa,bb,cc,c2-c3))
100   continue
      s=y
      return
      end
c
      function funct(p1,p2,p3,q1,q2,r1,a,b,c,q)
*********************************************************************
*                                                                   *
*   Function funct gives the value of the integrand (see eqs.76-78) *
*                                                                   *
*********************************************************************
      real num1,num2,den1,den2
c
      dq=q2-q1
      g1=(p2-p1)/dq
```

```
      g2=(p2-p3)/dq
      h1=(p1*q2-p2*q1)/dq
      h2=(p3*q2-p2*q1)/dq
      v=q*q+r1*r1
      pv1=g1*q+h1
      pv2=g2*q+h2
      num1=a*v-(b*q+c*r1)*pv1
      num2=a*v-(b*q+c*r1)*pv2
      den1=v*sqrt(pv1*pv1+v)
      den2=v*sqrt(pv2*pv2+v)
      funct=(num2/den2)-(num1/den1)
c
      return
      end
```

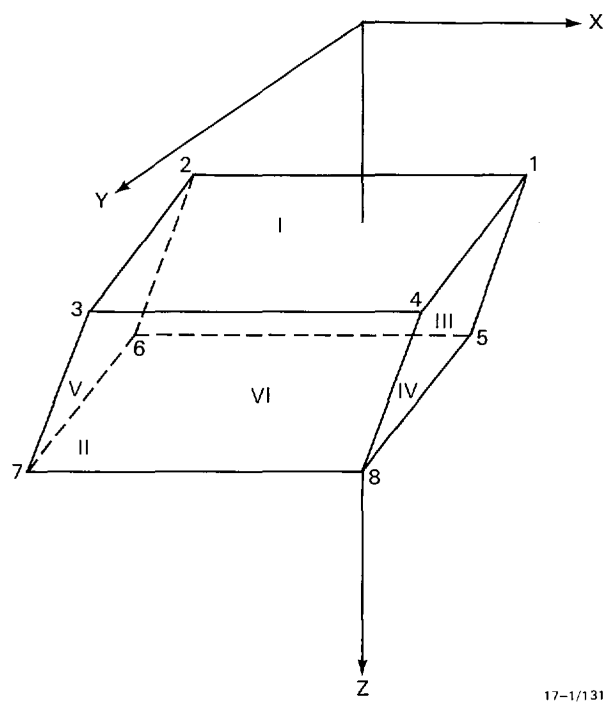

Fig. A1. Dipping prism.

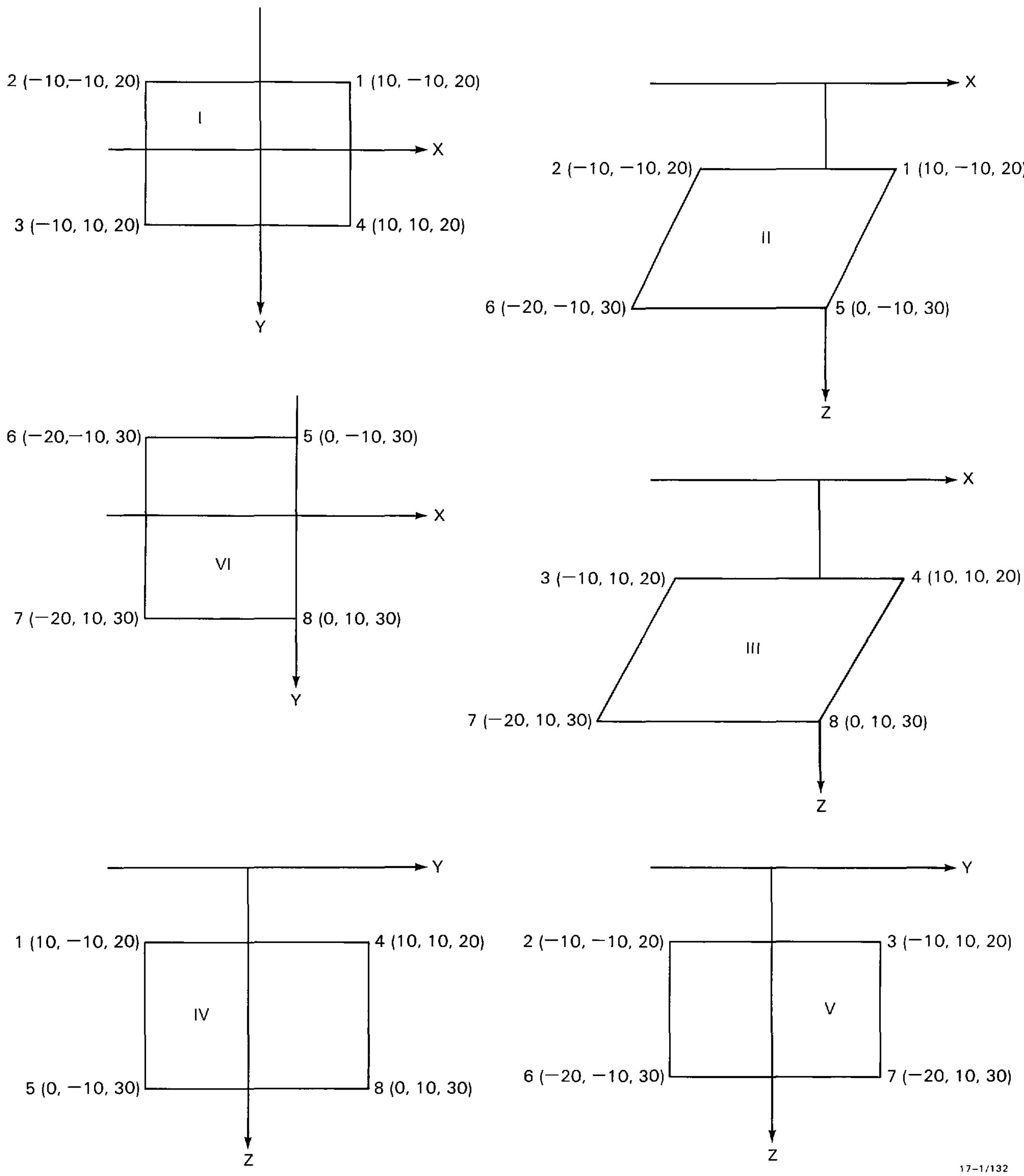

Fig. A2. Faces of the dipping prism.

```
      program main
*********************************************************************
*                                                                   *
*                        RECTANEL.F77                               *
*                                                                   *
*     Program for testing the subroutine for calculating  total     *
*     field from an arbitrarily oriented rectangular element.       *
*                                                                   *
*     The subroutine was tested on the example of 3D finite prism.*
*     The surface of the prism was divided into 6 rectangular       *
*     elements and contributions from these surfaces were added up*
*     The prism and the way of dividing it are given on Fig.A3-A4 *
*                                                                   *
*                                                                   *
*                        Revised: 2 March 89                        *
*                                                                   *
*********************************************************************
      parameter (np=3,n=18,lq=9,nfacet=6)
      external funct
      dimension coord(n,np),acoord(np,np),tfield(lq,lq),x(lq),y(lq)
      integer facet
      real jx,jy,jz,incl,inclr,k
c
c     generate matrix containing coords of each rectangular  element
c
      data ((coord(i,j),j=1,np),i=1,n)/
c               x      y      z    vertex  number
     +        -10.0,-10.0, 20.0, ! 1
     +        -10.0, 10.0, 20.0, ! 2 first face
     +         10.0,-10.0, 20.0, ! 3
c
     +        -10.0,-10.0, 40.0, ! 5
     +         10.0,-10.0, 40.0, ! 7 second face
     +        -10.0, 10.0, 40.0, ! 6
c
     +        -10.0,-10.0, 20.0, ! 1
     +         10.0,-10.0, 20.0, ! 3 third  face
     +        -10.0,-10.0, 40.0, ! 5
c
     +        -10.0, 10.0, 20.0, ! 2
     +        -10.0, 10.0, 40.0, ! 6 fourth face
     +         10.0, 10.0, 20.0, ! 4
c
     +         10.0,-10.0, 20.0, ! 3
     +         10.0, 10.0, 20.0, ! 4 fifth face
     +         10.0,-10.0, 40.0, ! 7
c
     +        -10.0,-10.0, 20.0, ! 1
     +        -10.0,-10.0, 40.0, ! 5  sixth face
     +        -10.0, 10.0, 20.0/ ! 2
c
      pi=4.0*atan(1.0)
c
```

```
c       open output file
c
        open(1,file='recout',status='fresh')
c
c       input parameters
c
        f=58000.0           ! total inducing field in nanoTesla
        incl=-60.0          ! inclination of the inducing field in degrees
        sangle=30.0         ! strike angle in degrees ( clockwise from
c                             Nmag to positive y axis)
        k=4.0*pi*1.0e-3 ! susceptibility of the prism in SI units
        z=0.0               ! height of the observational point in meters
c
c       conversion to radians
c
        inclr=(incl/180.0)*pi
        sanglr=(sangle/180.0)*pi
c
c       determine unit vector in the direction of the inducing field
c
        ex=cos(inclr)*sin(sanglr)
        ey=cos(inclr)*cos(sanglr)
        ez=sin(inclr)
c
c       determine vector of magnetization (in order to take into
c       account remanent magnetization jx,jy,jz should be changed
c       accordingly
c
        jx=ex*k*f
        jy=ey*k*f
        jz=ez*k*f
c
c       determine (x,y) points at which the total magnetic field
c       will be calculated
c
        step=15.0
        xmin=-60.0
        ymin=-60.0
        do 5 i=1,lq
                x(i)=xmin+(i-1)*step
                y(i)=ymin+(i-1)*step
5       continue
c
c       calculate the total field
c
        do 20 i=1,lq
                xp=x(i)
                do 20 j=1,lq
                        yp=y(j)
c
c       start calculating and summing up contribution from each face
c
        sumf=0.0
        do 30 facet=1,nfacet
                ip=3*facet-2
                is=3*facet-1
```

```
                ik=3*facet
c
        do 35 jl=1,np
                acoord(1,jl)=coord(ip,jl)
                acoord(2,jl)=coord(is,jl)
                acoord(3,jl)=coord(ik,jl)
35      continue
c
c       call the subroutine for calculating the magnetic field from a
c                rectangular element
c
        call rectanel(funct,acoord,np,jx,jy,jz,ex,ey,ez,xp,yp,z,totf)
c
        sumf=sumf+totf
30      continue
        tfield(i,j)=sumf
20      continue
c
c       print the results
c
        write(1,900)
900     format(1x,//,20x,'Test of rectangular prism',//)
        write(1,910)
910     format(15x,'* Input data *',//)
        write(1,920) incl,f,sangle,k,z
920     format(12x,'Inclination= ',f5.1,/,12x,'Total inducing field='
    +            ,f8.1,/,12x,'Strike angle=',f5.1,/,12x,
    +            'Susceptibility contrast in SI = ',f9.5,/,12x,
    +            'Height of the observational points in m=',f8.1,//)
c
        write(1,930)
930     format(15x,'* Output data *',//)
        write(1,940)
940     format(1x,//,26x,'Total magnetic field in nT',//)
        write(1,950) (j,j=1,lq)
950     format(8x,9i7,/)
        write(1,960) (y(j),j=1,lq)
960     format(9x,9f7.1,/)
        write(1,970) (i,x(i),(tfield(i,j),j=1,lq),i=lq,1,-1)
970     format(1x,i2,1x,f5.1,1x,9f7.1)
        stop
        end
        subroutine rectanel(funct,a,n,jx,jy,jz,ex,ey,ez,x,y,z,totf)
*********************************************************************
*                                                                   *
*    Subroutine rectanel calculates the magnetic effect from an     *
*    arbitrarily oriented rectangular element.                      *
*                                                                   *
*       Input pararmeters:                                          *
*               a - an 3x3 matrix containing coordinates of         *
*                   the vertices  of the element i.e.               *
*                          x1,y1,z1                                 *
*                          x2,y2,z2                                 *
*                          x3,y3,z3                                 *
*                   the vertices are taken in such order that       *
*                   the vector product of vectors from point 1      *
```

```
*                    to point 2 and form point 1 to point 3          *
*                    gives the vector in the direction of an         *
*                    outward normal to the element,                  *
*                                                                    *
*       n - integer number (the dimension of the matrix a),          *
*                                                                    *
*       (jx,jy,jz) - components of the magnetization vector          *
*                     in SI units,                                   *
*                                                                    *
*       (ex,ey,ez) - components of the unit vector in the            *
*                     direction of the inducing field,               *
*                                                                    *
*       (x,y,z) - coordinates in meters of the point at which        *
*                  magnetic field will be calculated.                *
*                                                                    *
*    Output parameters:                                              *
*                 totf - total magnetic field in nanoTesla at        *
*                        (x,y,z) point.                              *
*                                                                    *
*    Required : external function funct (see below)                  *
*                                                                    *
*    Subroutines:transg,gauss                                        *
*                                                                    *
**********************************************************************
      parameter (nk=3)
      dimension a(n,n),trans(nk,nk),b(nk,nk),vec(nk),p(nk)
      external funct
      real jx,jy,jz
      pi=4.0*atan(1.0)
c
      p(1)=x
      p(2)=y
      p(3)=z
c
      call transg(p,a,b,trans,vec)
c
c     find aa,bb,cc,s
c
      aa=ex*trans(1,1)+ey*trans(1,2)+ez*trans(1,3)
      bb=ex*trans(2,1)+ey*trans(2,2)+ez*trans(2,3)
      cc=ex*trans(3,1)+ey*trans(3,2)+ez*trans(3,3)
c
      s=-(jx*vec(1)+jy*vec(2)+jz*vec(3))/(4.0*pi)
c
c     find the integral
c
      call gauss(funct,b(1,2),b(2,2),aa,bb,cc,b(1,3),b(3,1),t1)
      call gauss(funct,b(1,2),b(2,2),aa,bb,cc,b(1,3),b(1,1),t2)
c
      totf=s*(t1-t2)
c
      return
      end
c
      real function funct(a,b,c,p,r,q)
**********************************************************************
```

```
*                                                                      *
*       Function funct gives values of the integrand (see eq.82)       *
*                                                                      *
************************************************************************
        real num
c
        v=q*q+r*r
        w=sqrt(p*p+q*q+r*r)
c
        num=a*v-p*(b*q+c*r)
        den=v*w
c
        funct=num/den
        return
        end
c
        subroutine vecpr(a1,a2,a3,b1,b2,b3,c1,c2,c3)
************************************************************************
*                                                                      *
*       Subroutine vecpr calculates vector product of two vectors      *
*                       a & b.                                         *
*                                                                      *
*       Input parameters:                                              *
*                       (a1,a2,a3)-components of the first vector,     *
*                       (b1,b2,b3)-components of the second vector.    *
*                                                                      *
*       Output parameters:                                             *
*                   (c1,c2,c2)-components of the vector product.       *
*                                                                      *
************************************************************************
c
        x=a2*b3-a3*b2
        y=a3*b1-a1*b3
        z=a1*b2-a2*b1
c
        d=sqrt(x*x+y*y+z*z)
c
        c1=x/d
        c2=y/d
        c3=z/d
c
        return
        end
c
        subroutine mult(a,xi,yi,zi,x0,y0,z0)
************************************************************************
*                                                                      *
*       Subroutine mult performs multiplication y=A*x                  *
*                                                                      *
*       Input parameters:                                              *
*                       a - an nxn matrix,                             *
*                       (xi,yi,zi) -components of the vector x         *
*                                                                      *
*       Output parameters:                                             *
*                        (x0,y0,z0)-components of the vector y.        *
*                                                                      *
```

```
*******************************************************************
        parameter (n=3)
        dimension a(n,n),vin(n),vtr(n)
        vin(1)=xi
        vin(2)=yi
        vin(3)=zi
c
        do 10 i=1,n
                s=0.0
                do 15 j=1,n
                        s=s+a(i,j)*vin(j)
15              continue
                vtr(i)=s
10      continue
c
        x0=vtr(1)
        y0=vtr(2)
        z0=vtr(3)
c
        return
        end
c
        subroutine vecab(xp,yp,zp,xe,ye,ze,cosx,cosy,cosz)
*********************************************************************
*                                                                   *
*       Subroutine vecab calculates an unit vector in the direction *
*                       (xp,yp,zp)------>(xe,ye,ze).                *
*                                                                   *
*       Input parameters:                                           *
*               (xp,yp,zp)-coordinates of the begining of the       *
*                          vector,                                  *
*               (xe,ye,ze)-coordinates of the end of the            *
*                          vector.                                  *
*       Output parameters:                                          *
*               (cosx,cosy,cosz)-components of the unit vector.     *
*                                                                   *
*********************************************************************
        dx=xe-xp
        dy=ye-yp
        dz=ze-zp
c
        d=sqrt(dx*dx+dy*dy+dz*dz)
c
        cosx=dx/d
        cosy=dy/d
        cosz=dz/d
c
        return
        end
c
        subroutine gauss(funct,qa,qb,aa,bb,cc,r,p,t)
*********************************************************************
*                                                                   *
*    Subroutine gauss calculates integral I (see eq. 76) using  a   *
*       10-point Gauss-Legendre integration formula.                *
*                                                                   *
```

```
*       Handbook of Mathematical Constants-Abramowitz & Stegun-p887,*
*       25.4.30 and p916 table 25.4.                                *
*                                                                   *
*    Input parameters:                                              *
*               funct-function to be integrated (external),         *
*               qa-lower limit of integration,                      *
*               qb-upper limit of integration,                      *
*               aa,bb,cc,p,r-parameters defining the function       *
*               (see eq. 82)                                        *
*    Output parmeter:                                               *
*               t-returned value of the integral.                   *
*                                                                   *
*    Required:                                                      *
*       external function funct.                                    *
*                                                                   *
*********************************************************************
        external funct
        parameter (n=5)
        dimension zero(n),coeff(n)
        data zero /0.14887434,0.43339539,0.67940957,0.865063337,
     +           0.97390653/
        data coeff/0.29552422,0.26926672,0.21908636,0.14945135,
     +           0.06667134/
c
        t=0.0
        c1=(qb-qa)/2.0
        c2=(qb+qa)/2.0
c
        do 10 i=1,n
                c3=c1*zero(i)
                t=t+c1*coeff(i)*(funct(aa,bb,cc,p,r,c2+c3)+
     +                       funct(aa,bb,cc,p,r,c2-c3))
10      continue
        return
        end
        subroutine transg(p,a,b,trans,vec)
********************************************************************
*                                                                  *
*   Subroutine transg calculates the transformation matrix for     *
*   tranforming coordinates in (xyz)-system to coordinates in      *
*   (pqr)-system,performs this transformation and finds a vector   *
*   normal to the rectangular element.                             *
*                                                                  *
*   Input parameters:                                              *
*               p-vector (p(1),p(2),p(3)) containing               *
*                 coordinates of the point,                        *
*                                                                  *
*       | a(1,1)=x1, a(1,2)=y1, a(1,3)=z1 |                        *
*       | a(2,1)=x2, a(2,2)=y2, a(2,3)=z2 | matrix of the ver-     *
*       | a(3,1)=x3, a(3,2)=y3, a(3,3)=z3 |     tices.             *
*                                                                  *
*   Output parameters:                                             *
*       | b(1,1)=p1, b(1,2)=q1, b(1,3)=r1 |                        *
*       | b(2,1)=p2, b(2,2)=q2, b(2,3)=r2 |                        *
*       | b(3,1)=p3, b(3,2)=q3, b(3,3)=r3 |                        *
*                                                                  *
```

```
*       | trans(1,1)=lx, trans(1,2)=ly, trans(1,3)=lz |          *
*       | trans(1,2)=mx, trans(2,2)=my, trans(2,3)=mz |          *
*       | trans(1,3)=nx, trans(3,1)=ny, trans(3,3)=nz |          *
*                                                                *
*       vec-(vec(1),vec(2),vec(3))-vector normal to the          *
*                                  element.                      *
*   Required:                                                    *
*       subroutines:vecab,vecpr,mult                             *
*                                                                *
******************************************************************
      parameter (n=3)
      real lx,ly,lz,mx,my,mz,nx,ny,nz
      dimension a(n,n),trans(n,n),vec(n),b(n,n),p(n)
c
c
      call vecab(a(1,1),a(1,2),a(1,3),a(3,1),a(3,2),a(3,3),lx,ly,lz)
c
      call vecab(a(1,1),a(1,2),a(1,3),a(2,1),a(2,2),a(2,3),mx,my,mz)
c
      call vecpr(lx,ly,lz,mx,my,mz,nx,ny,nz)
c
      trans(1,1)=lx
      trans(1,2)=ly
      trans(1,3)=lz
      trans(2,1)=mx
      trans(2,2)=my
      trans(2,3)=mz
      trans(3,1)=nx
      trans(3,2)=ny
      trans(3,3)=nz
c
      call mult(trans,p(1),p(2),p(3),vec(1),vec(2),vec(3))
c
      call mult(trans,a(1,1),a(1,2),a(1,3),b(1,1),b(1,2),b(1,3))
      call mult(trans,a(2,1),a(2,2),a(2,3),b(2,1),b(2,2),b(2,3))
      call mult(trans,a(3,1),a(3,2),a(3,3),b(3,1),b(3,2),b(3,3))
c
      do 10 j=1,n
      do 10 i=1,n
          b(i,j)=b(i,j)-vec(j)
10    continue
c
      vec(1)=nx
      vec(2)=ny
      vec(3)=nz
c
      return
      end
```

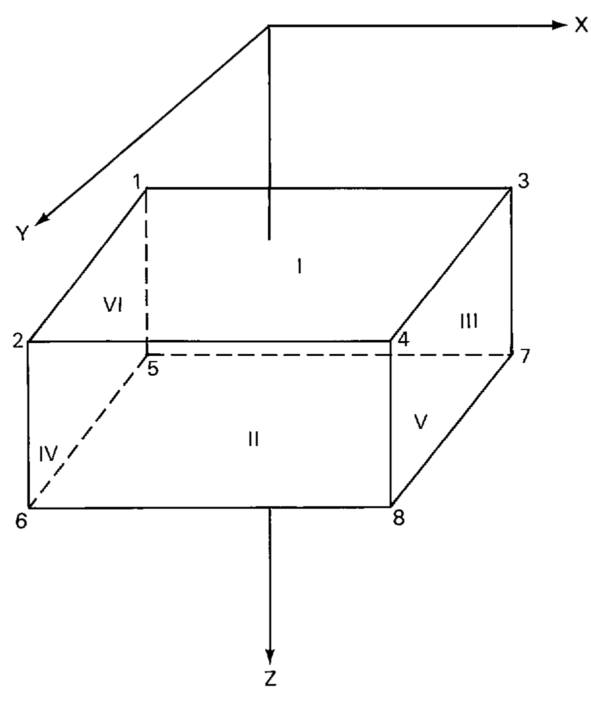

Fig. A3. Rectangular prism.

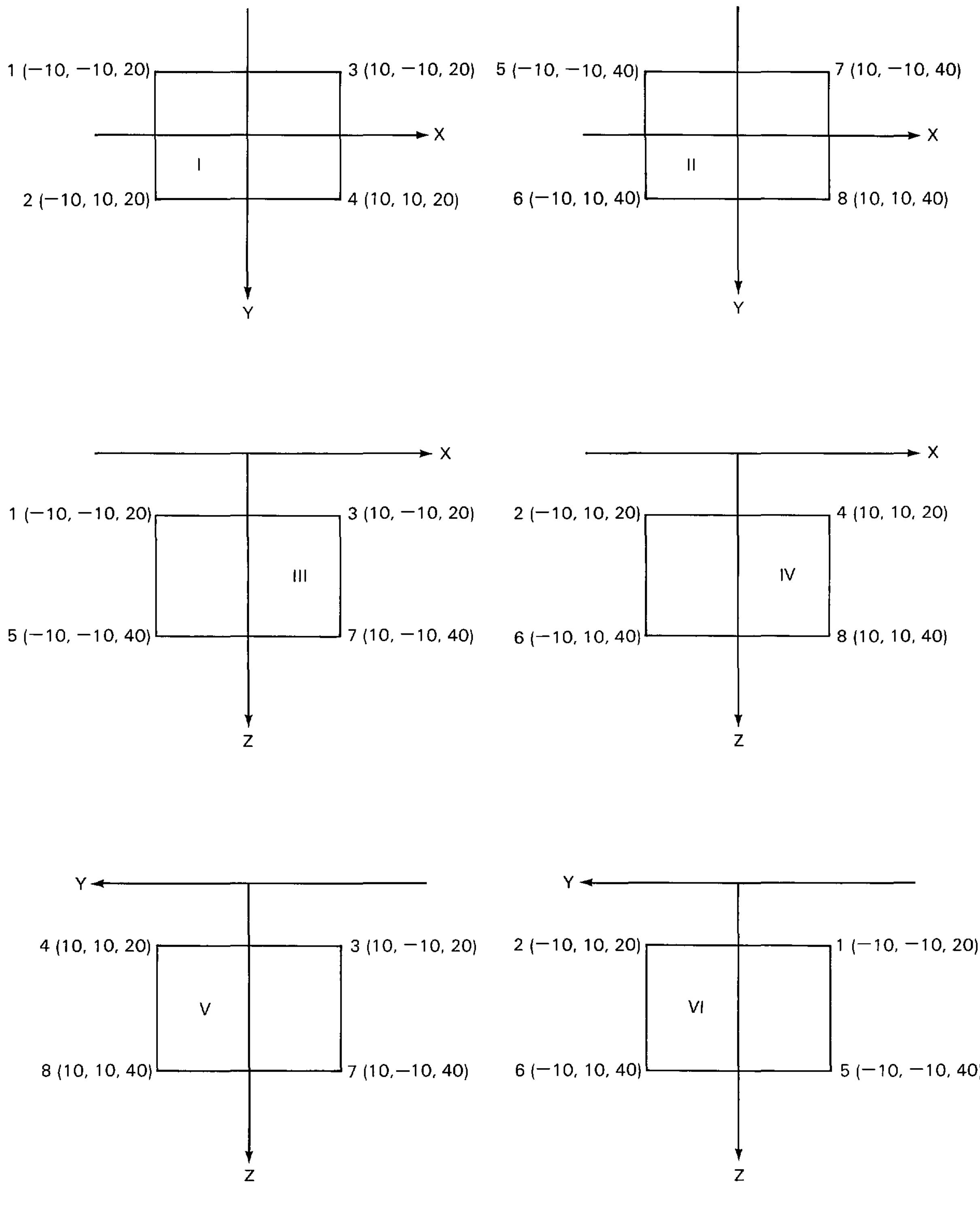

Fig. A4. Faces of the rectangular prism.