```
%
% This function solves any type of spherical triangle
% A spherical triangle has 6 basic elements: three angles (A,B,C) and three sides (a,b,c).
% Given any three of the 6 basic elements, the function finds the other three.
%
% Usage of function
% for example, Given 3 elements of a spherical triangle: a=2, b=3, C=104.4944158
%[A1, B1, A2, B2,C1,C2,a1,b1,a2,b2,c1,c2] = SphericTriangleSolver(A,B,C,a,b,c)
%[A1, B1, A2, B2,C1,C2,a1,b1,a2,b2,c1,c2] = SphericTriangleSolver(0,0,104.4944158,2,3,0)

% Attention, the unknown (A,B,c) 3 elements of the spherical triangle should be replaced with zero.

% A spherical triangle has 6 basic elements: three angles (A,B,C) and three sides (a,b,c).
% Given any three of the 6 basic elements, the function finds the other three.
%
%
% INPUT:
% A,B,C,a,b,c   element of spherical triangle. Only 3 values should be entered.
% the others must be zero.
% Inputs units must be [decimal degree]



%OUTPUT:
%
%  A1, B1,C1,a1,b1,c1  are 1. solutions of the spherical triangle.
%  A2, B2,C2,a2,b2,c2  are 2. solutions of the spherical triangle.
% outputs units are degrees minutes seconds [dd mm ss]

% Usage of function
%[A1, B1, A2, B2,C1,C2,a1,b1,a2,b2,c1,c2] = SphericTriangleSolver(0,0,104.4944158,2,3,0)
```

```
% Attention, the unknown, desired 3 elements of the spherical triangle↙
should be replaced with zero.

% While preparing this function, Rody P.S. Oldenhuis subfunctions are↙
used.


function [A1, B1, A2, B2,C1,C2,a1,b1,a2,b2,c1,c2] =↙
SphericTriangleSolver(A,B,C,a,b,c)

format long
%
ro=180/pi;
A=0;,B=0;,C=0;     a=2;,b=3;,c=4;
%A=28.97193110;,B=46.5843689140;,C=104.4944158550;     a=0;,b=0;,c=0;
%A=0;,B=46.5843689140;,C=0;     a=2;,b=0;,c=4;
%A=0;,B=0;,C=104.49441585500;     a=0;,b=3;,c=4;
%A=0;,B=46.58436891400;,C=104.4944158550;     a=0;,b=0;,c=4;

%A=112.62944444;,B=65.30972222220;,C=0;     a=0;,b=70.874166666;,c=0;
%A=0;,B=0;,C=0;     a=73.701111110;,b=70.874166666;,c=7.0355555550;
%A=112.62944444;,B=65.30972222220;,C=0;     a=0;,b=70.874166666;,c=0;

%A=120.57333330;,B=0;,C=0;     a=105.4811111110;,b=75.7294444440;,c=0;
%A=112.62944444;,B=65.30972222220;,C=0;     a=0;,b=70.874166666;,c=0;

%A=60.46011667;,B=42.55760065;,C=80.6425750;     a=0;,b=0;,c=0;

%A=25;,B=0;,C=0;     a=0;,b=50;,c=70;
%A=76.255;,B=115.795;,C=0;     a=0;,b=0;,c=81.4367777770;
%A=30;,B=0;,C=0;     a=70;,b=0;,c=80;

%a=148.573333;,b=142.1933333;c=0;     A=153.12666666;,B=0;,C=0;
%A=132.695;,B=107.1163889;,C=0;     a=146.345833333;,b=0;,c=0;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
A1=0;A2=0;B1=0;B2=0 ;C1=0;C2=0;a1=0;a2=0;b1=0;b2=0 ;c1=0;c2=0;

fprintf('Givens  [decimal degree]   \n')
fprintf(' Angles     A=%3d %2.0f %6.4f B=%3d %2.0f %6.4f C=%3d %2.0f %↙
6.4f\n', degrees2dms(A), degrees2dms(B), degrees2dms(C))
```

```
fprintf('Sides   a=%3d %2.0f %6.4f b=%3d %2.0f %6.4f c=%3d %2.0f %6.4↙
f\n', degrees2dms(a), degrees2dms(b), degrees2dms(c))
fprintf('Requested  [dd mm ss]   \n')
A=A/ro;,B=B/ro;,C=C/ro;  a=a/ro;,b=b/ro;,c=c/ro;


%1.option KKK


if (A==0 & B==0 & C==0)
[A1,B1, C1, A2,B2,C2] = sss(a,b,c);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('A1=%3d %2.0f %6.4f B1=%3d %2.0f %6.4f C1=%3d %2.0f %6.4f\n',↙
degrees2dms(A1), degrees2dms(B1), degrees2dms(C1))
%fprintf('A2=%3d %2.0f %6.4f B2=%3d %2.0f %6.4f C2=%3d %2.0f %6.4f\n',↙
degrees2dms(A2), degrees2dms(B2), degrees2dms(C2))

end
%2.option AAA

if (a==0 & b==0 & c==0)
[a1, b1, c1, a2, b2, c2] = aaa(A,B,C);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('a1=%3d %2.0f %6.4f b1=%3d %2.0f %6.4f c1=%3d %2.0f %6.4f\n',↙
degrees2dms(a1), degrees2dms(b1), degrees2dms(c1))
%fprintf('a2=%3d %2.0f %6.4f b2=%3d %2.0f %6.4f c2=%3d %2.0f %6.4f\n',↙
degrees2dms(a2), degrees2dms(b2), degrees2dms(c2))

end

%3.option KAK  [c1, A1, B1, c2, A2, B2] = sas(a, C, b)

if (c==0 & A==0 & B==0)
[c1, A1, B1, c2, A2, B2] = sas(a,C,b);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;
```

```
fprintf('c1=%3d %2.0f %6.4f A1=%3d %2.0f %6.4f B1=%3d %2.0f %6.4f\n',↙
degrees2dms(c1), degrees2dms(A1), degrees2dms(B1))
%fprintf('c2=%3d %2.0f %6.4f A2=%3d %2.0f %6.4f B2=%3d %2.0f %6.4f\n',↙
degrees2dms(c2), degrees2dms(A2), degrees2dms(B2))

end

if (a==0 & B==0 & C==0)
[a1, B1, C1, a2, B2, C2] = sas(b,A,c);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('a1=%3d %2.0f %6.4f B1=%3d %2.0f %6.4f C1=%3d %2.0f %6.4f\n',↙
degrees2dms(a1), degrees2dms(B1), degrees2dms(C1))
%fprintf('a2=%3d %2.0f %6.4f B2=%3d %2.0f %6.4f C2=%3d %2.0f %6.4f\n',↙
degrees2dms(a2), degrees2dms(B2), degrees2dms(C2))
end
%[c1, A1, B1, c2, A2, B2] = sas(a, C, b)

if (b==0 & A==0 & C==0)
[b1,C1, A1, b2, C2, A2] = sas(c,B,a);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('b1=%3d %2.0f %6.4f C1=%3d %2.0f %6.4f A1=%3d %2.0f %6.4f\n',↙
degrees2dms(b1), degrees2dms(C1), degrees2dms(A1))
%fprintf('b2=%3d %2.0f %6.4f C2=%3d %2.0f %6.4f A2=%3d %2.0f %6.4f\n',↙
degrees2dms(b2), degrees2dms(C2), degrees2dms(A2))

end

%4.option AKA   [C1, a1, b1, C2, a2, b2] = asa(A, B, c)

if (C==0 & a==0 & b==0)
[C1,a1, b1, C2 a2, b2] = asa(A,B,c);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('C1=%3d %2.0f %6.4f a1=%3d %2.0f %6.4f b1=%3d %2.0f %6.4f\n',↙
degrees2dms(C1), degrees2dms(a1), degrees2dms(b1))
%fprintf('C2=%3d %2.0f %6.4f a2=%3d %2.0f %6.4f b2=%3d %2.0f %6.4f\n',↙
degrees2dms(C2), degrees2dms(a2), degrees2dms(b2))

end

if (A==0 & b==0 & c==0)
[A1, b1, c1, A2, b2, c2] = asa(B,C,a);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('A1=%3d %2.0f %6.4f b1=%3d %2.0f %6.4f c1=%3d %2.0f %6.4f\n',↙
degrees2dms(A1), degrees2dms(b1), degrees2dms(c1))
%fprintf('A2=%3d %2.0f %6.4f b2=%3d %2.0f %6.4f c2=%3d %2.0f %6.4f\n',↙
degrees2dms(A2), degrees2dms(b2), degrees2dms(c2))

end

if (B==0 & a==0 & c==0)
[B1,  c1,a1, B2,c2, a2] = asa(C,A,b);[B1,  c1,a1, B2,c2, a2] =[B1,  c1,↙
a1, B2,c2, a2] *ro;
fprintf('a1=%3d %2.0f %6.4f c1=%3d %2.0f %6.4f B1=%3d %2.0f %6.4f\n',↙
degrees2dms(a1), degrees2dms(c1), degrees2dms(B1))
%fprintf('a2=%3d %2.0f %6.4f c2=%3d %2.0f %6.4f B2=%3d %2.0f %6.4f\n',↙
degrees2dms(a2), degrees2dms(c2), degrees2dms(B2))

end

%5.option KKA

if (c==0 & B==0 & C==0)
[c1, B1, C1, c2, B2, C2] = ssa(a,b,A);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('B1=%3d %2.0f %6.4f C1=%3d %2.0f %6.4f c1=%3d %2.0f %6.4f\n',↙
degrees2dms(c1), degrees2dms(B1), degrees2dms(C1))
fprintf('B2=%3d %2.0f %6.4f C2=%3d %2.0f %6.4f c2=%3d %2.0f %6.4f\n',↙
degrees2dms(c2), degrees2dms(B2), degrees2dms(C2))

end

if (a==0 & A==0 & C==0)
```

```
[a1,A1, C1, a2, A2, C2] = ssa(b,c,B);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;


fprintf('C1=%3d %2.0f %6.4f A1=%3d %2.0f %6.4f a1=%3d %2.0f %6.4f\n',↙
degrees2dms(a1), degrees2dms(A1), degrees2dms(C1))
fprintf('C2=%3d %2.0f %6.4f A2=%3d %2.0f %6.4f a2=%3d %2.0f %6.4f\n',↙
degrees2dms(a2), degrees2dms(A2), degrees2dms(C2))

end

if (b==0 & A==0 & B==0)
[A1, B1, b1, A2, B2, b2] = ssa(c,a,C);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('b1=%3d %2.0f %6.4f A1=%3d %2.0f %6.4f B1=%3d %2.0f %6.4f\n',↙
degrees2dms(b1), degrees2dms(A1), degrees2dms(B1))
fprintf('b2=%3d %2.0f %6.4f A2=%3d %2.0f %6.4f B2=%3d %2.0f %6.4f\n',↙
degrees2dms(b2), degrees2dms(A2), degrees2dms(B2))

end

%6.option AAK

if (C==0 & c==0 & b==0)
[b1, c1, C1, b2, c2, C2] = aas(A,B,a);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('c1=%3d %2.0f %6.4f b1=%3d %2.0f %6.4f C1=%3d %2.0f %6.4f\n',↙
degrees2dms(c1), degrees2dms(b1), degrees2dms(C1))
fprintf('c2=%3d %2.0f %6.4f b2=%3d %2.0f %6.4f C2=%3d %2.0f %6.4f\n',↙
degrees2dms(c2), degrees2dms(b2), degrees2dms(C2))

end

if (A==0 & a==0 & c==0)
[A1, a1, c1, A2, a2, c2] = aas(B,C,b);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
```

```
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('A1=%3d %2.0f %6.4f c1=%3d %2.0f %6.4f a1=%3d %2.0f %6.4f\n',↙
degrees2dms(c1), degrees2dms(A1), degrees2dms(a1))
fprintf('A2=%3d %2.0f %6.4f c2=%3d %2.0f %6.4f a2=%3d %2.0f %6.4f\n',↙
degrees2dms(c2), degrees2dms(A2), degrees2dms(a2))

end

if (B==0 & a==0 & b==0)
[a1, B1, b1, a2, B2, b2] = aas(C,A,c);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('a1=%3d %2.0f %6.4f B1=%3d %2.0f %6.4f c1=%3d %2.0f %6.4f\n',↙
degrees2dms(a1), degrees2dms(b1), degrees2dms(B1))
fprintf('a2=%3d %2.0f %6.4f B2=%3d %2.0f %6.4f c2=%3d %2.0f %6.4f\n',↙
degrees2dms(a2), degrees2dms(b2), degrees2dms(B2))

end

%%%%%============================================
%7.option  AKK

if (b==0 & B==0 & C==0)
[c1, B1, C1, c2, B2, C2] = ssa(a,c,A);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;
fprintf('C1=%3d %2.0f %6.4f B1=%3d %2.0f %6.4f b1=%3d %2.0f %6.4f\n',↙
degrees2dms(c1), degrees2dms(B1), degrees2dms(C1))
fprintf('C2=%3d %2.0f %6.4f B2=%3d %2.0f %6.4f b2=%3d %2.0f %6.4f\n',↙
degrees2dms(c2), degrees2dms(B2), degrees2dms(C2))
end

if (c==0 & A==0 & C==0)
[c1, A1, C1, c2, A2, C2] = ssa(b,a,B);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('A1=%3d %2.0f %6.4f c1=%3d %2.0f %6.4f C1=%3d %2.0f %6.4f\n',↙
degrees2dms(c1), degrees2dms(C1), degrees2dms(A1))
```

```
fprintf('A2=%3d %2.0f %6.4f c2=%3d %2.0f %6.4f C2=%3d %2.0f %6.4f\n',↙
degrees2dms(c2), degrees2dms(C2), degrees2dms(A2))
end

if (a==0 & A==0 & B==0)
[a1, A1, B1, a2, A2, B2] = ssa(c,b,C);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('B1=%3d %2.0f %6.4f a1=%3d %2.0f %6.4f A1=%3d %2.0f %6.4f\n',↙
degrees2dms(a1), degrees2dms(B1), degrees2dms(A1))
fprintf('B2=%3d %2.0f %6.4f a2=%3d %2.0f %6.4f A2=%3d %2.0f %6.4f\n',↙
degrees2dms(a2), degrees2dms(B2), degrees2dms(A2))
end


% 8.option  KAA

if (c==0 & B==0 & b==0)
[c1, b1, B1, c2, b2,B2] = aas2(A,C,a);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('c1=%3d %2.0f %6.4f B1=%3d %2.0f %6.4f b1=%3d %2.0f %6.4f\n',↙
degrees2dms(c1), degrees2dms(B1), degrees2dms(b1))
fprintf('c2=%3d %2.0f %6.4f B2=%3d %2.0f %6.4f b2=%3d %2.0f %6.4f\n',↙
degrees2dms(c2), degrees2dms(B2), degrees2dms(b2))
end

if (c==0 & a==0 & C==0)
[c1, a1, C1, c2, a2,C2] = aas2(A,B,b);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('c1=%3d %2.0f %6.4f a1=%3d %2.0f %6.4f C1=%3d %2.0f %6.4f\n',↙
degrees2dms(a1), degrees2dms(c1), degrees2dms(C1))
fprintf('c2=%3d %2.0f %6.4f a2=%3d %2.0f %6.4f C2=%3d %2.0f %6.4f\n',↙
degrees2dms(a2), degrees2dms(c2), degrees2dms(C2))
end

if (b==0 & a==0 & A==0)
```

```
[b1, a1, A1, b2, a2,A2] = aas2(C,B,c);
A1=ro*A1;A2=ro*A2;B1=ro*B1;B2=ro*B2 ;C1=ro*C1;C2=ro*C2;a1=ro*a1;↙
a2=ro*a2;b1=ro*b1;b2=ro*b2 ;c1=ro*c1;c2=ro*c2;

fprintf('a1=%3d %2.0f %6.4f A1=%3d %2.0f %6.4f b1=%3d %2.0f %6.4f\n',↙
degrees2dms(a1), degrees2dms(A1), degrees2dms(b1))
fprintf('a2=%3d %2.0f %6.4f A2=%3d %2.0f %6.4f b2=%3d %2.0f %6.4f\n',↙
degrees2dms(a2), degrees2dms(A2), degrees2dms(b2))
end

%a1, B1, c1degrees2dms
function [a1, b1, c1, a2, b2, c2] = aaa(A, B, C)
%AAA  gives both solutions to the angle-angle-angle problem, in radians.
%
%   AAA(A, B, C) will result in NaNs if the existence condition
%   |pi - |A|-|B|| <= |C| <= pi - ||A| - |B||
%   is not met.
%



    % first solution
    a1 = acos2( (cos(A) + cos(B).*cos(C)) ./ (sin(B).*sin(C)),  A );
    b1 = acos2( (cos(B) + cos(A).*cos(C)) ./ (sin(A).*sin(C)),  B );
    c1 = acos2( (cos(C) + cos(A).*cos(B)) ./ (sin(A).*sin(B)),  C );

    % second solution
    a2 = 2*pi - a1;
    b2 = 2*pi - b1;
    c2 = 2*pi - c1;

    % check constraints
    indices = ( ...
        abs(pi - abs(A)-abs(B)) > abs(C) | ...
        abs(C) > pi - abs(abs(A)-abs(B)) );
    a1(indices) = NaN;    a2(indices) = NaN;
    b1(indices) = NaN;    b2(indices) = NaN;
    c1(indices) = NaN;    c2(indices) = NaN;

end
function [b1, c1, C1, b2, c2, C2] = aas(A, B, a)
%AAS   gives both solutions to the angle-angle-side problem, in radians.
```

```
%
%   AAS(A, B, a) may result in a vector filled with NaNs if the↙
existence
%   condition |sin(B)sin(a)| <= |sin(A)| is not met. This function uses↙
the
%   Middle Side Law function MSL.m and Middle Angle Law function MAL.m↙
to
%   determine the solutions.
%

    % first solution
    b0 = asin( (sin(B).*sin(a))./sin(A) );
    b0(imag(b0) ~= 0) = NaN;

    b1 = mod(b0, 2*pi);
    c1 = msl(a, b1, A, B);
    C1 = mal(A, B, a, b1);

    % second solution
    b2 = mod(pi - b1, 2*pi);
    c2 = msl(a, b2, A, B);
    C2 = mal(A, B, a, b2);




    % check constraints
    indices = ( abs(sin(B).*sin(a)) > abs(sin(A)) );
    b1(indices) = NaN;    c1(indices) = NaN;
    C1(indices) = NaN;    b2(indices) = NaN;
    c2(indices) = NaN;    C2(indices) = NaN;

end


% Middle-angle-law
function C = mal(A, B, a, b)
%MAL    Computes the missing angle in a spherical triangle, in radians.
%
%   MAL(A, B, a, b) is the implementation of the Middle Angle Law, and
%   returns the missing angle C.
```

```
%
%   See also MSL, MALD.

    % sine & cosine of C
    % NOTE: denomenator not needed
    sinC  =  sin(A).*cos(B).*cos(b) + sin(B).*cos(A).*cos(a);
    cosC  = -cos(A).*cos(B) + sin(A).*sin(B).*cos(a).*cos(b);

    % C is the arctangent of the ratio of these two
    C = mod( atan2(sinC, cosC), 2*pi);

end

% Middle-side-law
function c = msl(a, b, A, B)
%MSL    Computes the missing side in a spherical triangle, in radians.
%
%   MSL(a, b, A, B) is the implementation of the Middle Side Law, and
%   returns the missing angular side c.

    sinc  = (sin(a).*cos(b).*cos(B) + sin(b).*cos(a).*cos(A));
    cosc  = (cos(a).*cos(b) - sin(a).*sin(b).*cos(A).*cos(B));

    % c is the arctangent of the sine over the cosine
    c = mod( atan2(sinc, cosc), 2*pi);

end


function signedcos = acos2(alpha, beta)
%ACOS2      4-quadrant arccosine function, in radians.
%
%   ACOS2(alpha, beta) computes the four-quadrant arccosine of the amgle
%   [alpha]. For arguments |alpha| > 1, the result is NaN. The resulting
%   angle is not uniquely determined by alpha, nor by the lengths or
%   order of the sides of the triangle (as in ATAN2), so an additional
%   argument [beta] is required. If [beta] < pi/2, the small angle
%   (0 <= alpha <= pi/2) is returned. If [beta] > pi/2, the large angle
%   (pi/2 < alpha < pi) is returned.
%
%   See also acos2d.
```

```
    H = 2*( mod(beta, 2*pi) < pi ) - 1;
    H(~isreal(H)) = NaN;

    % compute signed arc-cosine
    signedcos = H .* acos(alpha);

    % set complex results to NaN & take the modulus
    signedcos(imag(signedcos) ~= 0) = NaN;
    signedcos = mod(signedcos, 2*pi);

    % determine alphaues for zero-alphaued acos
    ind1 = (signedcos == 0);
    ind2     = (H < 0);                 ind3     = (H > 0);
    indices1 = ((ind1 + ind2) == 2);    indices2 = ((ind1 + ind3) == 2);
    signedcos(indices1) = pi;           signedcos(indices2) = 0;

end

function [C1, a1, b1, C2, a2, b2] = asa(A, B, c)
%ASA   gives both solutions to the angle-side-angle problem, in radians.
%
%   ASA(A, B, c) returns the missing values C, a, b. It uses the
%   four-quadrant arccosine function ACOS2 to determine these values.
%

    % first solution
    % NOTE: normal acos (in stead of acos2) is indeed correct.
    C1 = acos( -cos(A) .*cos(B) + sin(A).*sin(B).*cos(c));
    a1 = acos( (cos(A) + cos(B).*cos(C1)) ./ (sin(B).*sin(C1)));
    b1 = acos( (cos(B) + cos(A).*cos(C1)) ./ (sin(A).*sin(C1)));

    C1(imag(C1) ~= 0) = NaN;
    a1(imag(a1) ~= 0) = NaN;
    b1(imag(b1) ~= 0) = NaN;

    % second solution
    C2 = 2*pi - C1;
    a2 = mod(a1 + pi, 2*pi);
    b2 = mod(b1 + pi, 2*pi);

end
function [c1, A1, B1, c2, A2, B2] = sas(a, C, b)
%SAS   gives both solutions to the side-angle-side problem, in radians.
%
%   SAS(a, C, b) returns the remaining unknowns of the spherical↙
triangle,
%   [c1, A1, B1, c2, A2, B2].
%



    % first solution
    c1 = acos2( cos(a).*cos(b) + sin(a).*sin(b).*cos(C),  C );
    A1 = acos2( (cos(a) - cos(b).*cos(c1))./(sin(b).*sin(c1)),  a );
    B1 = acos2( (cos(b) - cos(a).*cos(c1))./(sin(a).*sin(c1)),  b );

    % second solution
    c2 = 2*pi - c1;
    A2 = mod(A1 + pi, 2*pi);
    B2 = mod(B1 + pi, 2*pi);

end

function [B1, C1, c1, B2, C2, c2] = ssa(a, b, A)
%SSA   gives both solutions to the side-side-angle problem, in radians.
%
%   SSA(a, b, A) will result in NaNs if the existence condition
%   |sin b * sin A| <= | sin a |  is not met.
%
%


    % first solution
    B0 = asin(sin(b).*sin(A)./sin(a));
    B0(imag(B0) ~= 0) = NaN;

    B1 = mod(B0, 2*pi);
    C1 = mal(A, B1, a, b);
    c1 = msl(a, b, A, B1);

    % second solution
    B2 = mod(pi - B1, 2*pi);
    C2 = mal(A, B2, a, b);
    c2 = msl(a, b, A, B2);

    % check constraints
    indices = ( abs(sin(b).*sin(A)) > abs(sin(a)) );
    B1(indices) = NaN;    C1(indices) = NaN;
    c1(indices) = NaN;    B2(indices) = NaN;
    C2(indices) = NaN;    c2(indices) = NaN;

end




function [A1, B1, C1, A2, B2, C2] = sss(a, b, c)
%SSS   gives both solutions to the side-side-side problem, in radians.
%
%   SSS(a, b, c) results in NaNs for those indices where the existence
%   condition |pi - a| - |pi - b| <= |pi - c| <= |pi - a| + |pi -b| is↙
not
%   met.
%

    % first solution
    A1 = acos2( (cos(a) - cos(b).*cos(c))./(sin(b).*sin(c)), a);
    B1 = acos2( (cos(b) - cos(a).*cos(c))./(sin(a).*sin(c)), b);
    C1 = acos2( (cos(c) - cos(a).*cos(b))./(sin(a).*sin(b)), c);

    % second solution
    A2 = 2*pi - A1;
    B2 = 2*pi - B1;
    C2 = 2*pi - C1;

    % check constraints
    indices = ( ...
        (abs(pi-a) - abs(pi-b)) > abs(pi-c) | ...
        abs(pi-c) > (abs(pi-a) + abs(pi-b)) );
    A1(indices) = NaN;   B1(indices) = NaN;   C1(indices) = NaN;
    A2(indices) = NaN;   B2(indices) = NaN;   C2(indices) = NaN;
```

```
end
function [a1, c1, C1, a2, c2, C2] = aas2(A, B, b)

%benimki    aas2

a1=asin(sin(A)/sin(B)*sin(b));
a2=pi-a1;
%1.çözüm

C1=2* atan(cos((a1-b)/2)/tan((A+B)/2)/cos((a1+b)/2));
C2=2* atan(cos((a2-b)/2)/tan((A+B)/2)/cos((a2+b)/2));

c1=2* atan(tan((a1+b)/2)*cos((A+B)/2)/cos((A-B)/2));
c2=2* atan(tan((a2+b)/2)*cos((A+B)/2)/cos((A-B)/2));


end
end
```