**Computing planetary positions - a tutorial with worked examples**

By **Paul Schlyter, Stockholm, Sweden**  
email: [pausch@stjarnhimlen.se](mailto:pausch@stjarnhimlen.se) or WWW: <http://stjarnhimlen.se/>  
  
[Break out of a frame](http://www.stjarnhimlen.se/comp/tutorial.html)

* [1. Fundamentals](http://www.stjarnhimlen.se/comp/tutorial.html#1)
* [2. Some useful functions](http://www.stjarnhimlen.se/comp/tutorial.html#2)
* [3. Rectangular and spherical coordinates](http://www.stjarnhimlen.se/comp/tutorial.html#3)
* [4. The time scale. A test date](http://www.stjarnhimlen.se/comp/tutorial.html#4)
* [5. The Sun's position](http://www.stjarnhimlen.se/comp/tutorial.html#5)
* [6. Sidereal time and hour angle. Altitude and azimuth](http://www.stjarnhimlen.se/comp/tutorial.html#6)
* [7. The Moon's position](http://www.stjarnhimlen.se/comp/tutorial.html#7)
* [8. The Moon's position with higher accuracy. Perturbations](http://www.stjarnhimlen.se/comp/tutorial.html#8)
* [9. The Moon's topocentric position](http://www.stjarnhimlen.se/comp/tutorial.html#9)
* [10. The orbital elements of the planets](http://www.stjarnhimlen.se/comp/tutorial.html#10)
* [11. The heliocentric positions of the planets](http://www.stjarnhimlen.se/comp/tutorial.html#11)
* [12. Higher accuracy - perturbations](http://www.stjarnhimlen.se/comp/tutorial.html#12)
* [13. Precession](http://www.stjarnhimlen.se/comp/tutorial.html#13)
* [14. Geocentric positions of the planets](http://www.stjarnhimlen.se/comp/tutorial.html#14)
* [15. The elongation and physical ephemerides of the planets](http://www.stjarnhimlen.se/comp/tutorial.html#15)
* [16. The positions of comets. Comet Encke and Levy](http://www.stjarnhimlen.se/comp/tutorial.html#16)

[How to compute planetary positions](http://www.stjarnhimlen.se/comp/ppcomp.html)  
[Computing rise and set times](http://www.stjarnhimlen.se/comp/riset.html)  
  
  
Today it's not really that difficult to compute a planet's position from its orbital elements. The only thing you'll need is a computer and a suitable program. If you want to write such a program yourself, this text contains the formulae you need. The aim here is to obtain the planetary positions at any date during the 20'th and 21'st century with an error of one or at the most two arc minutes, and to compute the position of an asteroid or a comet from its orbital elements.  
  
No programs are given, because different computers and calculators are programmed in different languages. It is easier to convert formulae to your favourite language than to translate a program from one programming language to another. Therefore formulae are presented instead. Each formula is accompanied with one numeric example - this will enable you to check your implementation of these formulae. Just remember that all numerical quantities contain rounding errors, therefore you may get slightly different results in your program compared to the numerical results presented here. When I checked these numerical values I used a HP-48SX pocket calculator, which uses 12 digits of accuracy.

**1. Fundamentals**

A celestial body usually orbits the sun in an elliptical orbit. Perturbations from other planets causes small deviations from this elliptical orbit, but an unperturbed elliptical orbit can be used as a first approximation, and sometimes as the final approximation. If the distance from the Sun to the planet always is the same, then the planet follows a circular orbit. No planet does this, but the orbits of Venus and Neptune are very close to circles. Among the planets, Mercury and Pluto have orbits that deviate the most from a circle, i.e. are the most eccentric. Many asteroids have even more eccentric orbits, but the most eccentric orbits are to be found among the comets. Halley's comet, for instance, is closer to the Sun than Venus at perihelion, but farther away from the Sun than Neptune at aphelion. Some comets have even more eccentric orbits that are best approximated as a parabola. These orbits are not closed - a comet following a parabolic orbit passes the Sun only once, never to return. In reality these orbits are extremely elongated ellipses though, and these comets will eventually return, sometimes after many millennia.  
  
The **perihelion** and **aphelion** are the points in the orbit when the planet is closest to and most distant from the Sun. A parabolic orbit only has a perihelion of course.  
  
The **perigee** and **apogee** are points in the Moon's orbit (or the orbit of an artificial Earth satellite) which are closest to and most distant from the Earth.  
  
The **celestial sphere** is an imaginary sphere around the observer, at an arbitrary distance.  
  
The **celestial equator** is the Earth's equatorial plane projected on the celestial sphere.  
  
The **ecliptic** is the plane of the Earth's orbit. This is also the plane of the Sun's yearly apparent motion. The ecliptic is inclined by approximately 23.4\_deg to the celestial equator. The ecliptic intersects the celestial equator at two points: The Vernal Point (or "the first point of Aries"), and the Autumnal Point. The Vernal Point is the point of origin for two different commonly used celestial coordinates: equatorial coordinates and ecliptic coordinates.  
  
**Right Ascension** and **Declination** are equatorial coordinates using the celestial equator as a fundamental plane. At the Vernal Point both the Right Ascension and the Declination are zero. The Right Ascension is usually measured in hours and minutes, where one revolution is 24 hours (which means 1 hour equals 15 degrees). It's counted countersunwise along the celestial equator. The Declination goes from +90 to -90 degrees, and it's positive north of, and negative south of, the celestial equator.  
  
**Longitude** and **Latitude** are ecliptic coordinates, which use the ecliptic as a fundamental plane. Both are measured in degrees, and these coorinates too are both zero at the Vernal Point. The Longitude is counted countersunwise along the ecliptic. The Latitude is positive north of the ecliptic. Of course longitude and latitude are also used as terrestial coordinates, to measure a position of a point on the surface of the Earth.  
  
**Heliocentric**, **Geocentric**, **Topocentric**. A position relative to the Sun is heliocentric. If the position is relative to the center of the Earth, then it's geocentric. A topocentric position is relative to an observer on the surface of the Earth. Within the aim of our accuracy of 1-2 arc minutes, the difference between geocentric and topocentric position is negligible for all celestial bodies except the Moon (and some occasional asteroid which happens to pass very close to the Earth).  
  
The **orbital elements** consist of 6 quantities which completely define a circular, elliptic, parabolic or hyperbolic orbit. Three of these quantities describe the shape and size of the orbit, and the position of the planet in the orbit:

a Mean distance, or semi-major axis

e Eccentricity

T Time at perihelion

A cirular orbit has zero eccentricity. An elliptical orbit has an eccentricity between zero and one. A parabolic orbit has an eccentricity of exactly one. Finally, a hyperbolic orbit has an eccentricity larger than one. A parabolic orbit has an infinite semi-major axis, a, therefore one instead gives the perihelion distance, q, for a parabolic orbit:

q Perihelion distance = a \* (1 - e)

It is customary to give q instead of a also for hyperbolic orbit, and for elliptical orbits with eccentricity close to one.  
  
The three remaining orbital elements define the orientation of the orbit in space:

i Inclination, i.e. the "tilt" of the orbit relative to the

ecliptic. The inclination varies from 0 to 180 degrees. If

the inclination is larger than 90 degrees, the planet is in

a retrogade orbit, i.e. it moves "backwards". The most

well-known celestial body with retrogade motion is Comet Halley.

N (usually written as "Capital Omega") Longitude of Ascending

Node. This is the angle, along the ecliptic, from the Vernal

Point to the Ascending Node, which is the intersection between

the orbit and the ecliptic, where the planet moves from south

of to north of the ecliptic, i.e. from negative to positive

latitudes.

w (usually written as "small Omega") The angle from the Ascending

node to the Perihelion, along the orbit.

These are the primary orbital elements. From these many secondary orbital elements can be computed:

q Perihelion distance = a \* (1 - e)

Q Aphelion distance = a \* (1 + e)

P Orbital period = 365.256898326 \* a\*\*1.5/sqrt(1+m) days,

where m = the mass of the planet in solar masses (0 for

comets and asteroids). sqrt() is the square root function.

n Daily motion = 360\_deg / P degrees/day

t Some epoch as a day count, e.g. Julian Day Number. The Time

at Perihelion, T, should then be expressed as the same day count.

t - T Time since Perihelion, usually in days

M Mean Anomaly = n \* (t - T) = (t - T) \* 360\_deg / P

Mean Anomaly is 0 at perihelion and 180 degrees at aphelion

L Mean Longitude = M + w + N

E Eccentric anomaly, defined by Kepler's equation: M = E - e \* sin(E)

An auxiliary angle to compute the position in an elliptic orbit

v True anomaly: the angle from perihelion to the planet, as seen

from the Sun

r Heliocentric distance: the planet's distance from the Sun.

This relation is valid for an elliptic orbit:

r \* cos(v) = a \* (cos(E) - e)

r \* sin(v) = a \* sqrt(1 - e\*e) \* sin(E)

x,y,z Rectangular coordinates. Used e.g. when a heliocentric

position (seen from the Sun) should be converted to a

corresponding geocentric position (seen from the Earth).

**2. Some useful functions**

When doing these orbital computations it's useful to have access to several utility functions. Some of them are in the standard library of the programming language, but others must be added by the programmer. Pocket calculators are often better equipped: they usually have sin/cos/tan and their inverses in both radians and degrees. Often one also finds functions to directly convert between rectangular and polar coordinates.  
  
On microcomputers the situation is worse. Let's start with the programming language Basic: there we can count on having sin/cos/tan and atn (=arctan) in radians, but nothing more. arcsin and arccos are missing, and the trig functions don't work in degrees. However one can add one's own function library, like e.g. this:

95 rem Constants

100 pi = 3.14159265359

110 radeg = 180/pi

115 rem arcsin and arccos

120 def fnasin(x) = atn(x/sqr(1-x\*x))

130 def fnacos(x) = pi/2 - fnasin(x)

135 rem Trig. functions in degrees

140 def fnsind(x) = sin(x/radeg)

150 def fncosd(x) = cos(x/radeg)

160 def fntand(x) = tan(x/radeg)

170 def fnasind(x) = radeg\*atn(x/sqr(1-x\*x))

180 def fnacosd(x) = 90 - fnasind(x)

190 def fnatand(x) = radeg\*atn(x)

195 rem arctan in all four quadrants

200 def fnatan2(y,x) = atn(y/x) - pi\*(x<0)

210 def fnatan2d(y,x) = radeg\*atn(y/x) - 180\*(x<0)

215 rem Normalize an angle between 0 and 360 degrees

217 rem Use Double Precision, if possible

220 def fnrev#(x#) = x# - int(x#/360#)\*360#

225 rem Cube Root (needed for parabolic orbits)

230 def fncbrt(x) = exp( log(x)/3 )

The code above follows the convensions of traditional Microsoft Basic (MBASIC/BASICA/GWBASIC). If you use some other Basic dialect, you may want to modify this code.  
  
The code above gives you sin/cos/tan and their inverses, in radians and degrees. The functions fnatan2() and fnatan2d() may need some explanation: they take two arguments, x and y, and they compute arctan(y/x) but puts the angle in the correct quadrant from -180 to +180 degrees. This is really part of a conversion from rectangular to polar coordinates, where the angle is computed. The distance is then computed by:

sqrt( x\*x + y\*y )

The function fnrev# normalizes an angle to between 0 and 360 degrees, by adding or subtracting even mutiples of 360 degrees until the final value falls between 0 and 360. The # sign means that the function and its argument use double precision. It is essential that this function uses more than single precision. More about this later.  
  
One warning: some of these functions may divide by zero. If one tries to compute fnasin(1.0), it's computed as: atn(1/sqrt(1-1.0\*1.0)) = atn(1/sqr(0)) = atn(1/0). This is not such a big problem, since in practice one rarely tries to compute the arc sine of exactly 1.0. Also, some dialects of Microsoft Basic then just print the warning message "Overflow", compute 1/0 as the highest possible floating-point number, and then continue the program. When computing the arctan of this very large number, one will get pi/2 (or 90 degrees), which is the correct result. However, if you have access to a more modern, structured, Basic, with the ability to define multi-line function, then by all means use this to write a better version of arcsin, which treats arcsin(1.0) as a special case.  
  
There's a similar problem with the fncbrt() function - it only works for positive values. With a multi-line function definition it can be rewritten to work for negative values and zero as well, if one follows these simple rules: the Cube Root of zero is of course zero. The Cube Root of a negative number is computed by making the number positive, taking the Cube Root of that positive number, and then negating the result.  
  
Two other popular programming language are C and C++. The standard library of these languages are better equipped with trigonometric functions. You'll find sin/cos/tan and their inverses, and even an atan2() function among the standard functions. All you need to do is to define some macros to get the trig functions in degrees (include *all* parentheses in these macro definitions), and to define a rev() function which reduces an angle to between 0 and 360 degrees, and a cbrt() function which computes the cube root.

#define PI 3.14159265358979323846

#define RADEG (180.0/PI)

#define DEGRAD (PI/180.0)

#define sind(x) sin((x)\*DEGRAD)

#define cosd(x) cos((x)\*DEGRAD)

#define tand(x) tan((x)\*DEGRAD)

#define asind(x) (RADEG\*asin(x))

#define acosd(x) (RADEG\*acos(x))

#define atand(x) (RADEG\*atan(x))

#define atan2d(y,x) (RADEG\*atan2((y),(x)))

double rev( double x )

{

return x - floor(x/360.0)\*360.0;

}

double cbrt( double x )

{

if ( x > 0.0 )

return exp( log(x) / 3.0 );

else if ( x < 0.0 )

return -cbrt(-x);

else /\* x == 0.0 \*/

return 0.0;

}

In C++ the macros could preferably be defined as inline functions instead - this enables better type checking and also makes overloading of these function names possible.  
  
The good ol' programming langauge FORTRAN is also well equipped with standard library trig functions: we find sin/cos/tan + inverses, and also an atan2 but only for radians. Therefore we need several function definitions to get the trig functions in degrees too. Below I give code only for SIND, ATAND, ATAN2D, plus REV and CBRT. The remaining functions COSD, TAND, ASIND and ACOSD are written in a similar way:

FUNCTION SIND(X)

PARAMETER(RADEG=57.2957795130823)

SIND = SIN( X \* (1.0/RADEG) )

END

FUNCTION ATAND(X)

PARAMETER(RADEG=57.2957795130823)

ATAND = RADEG \* ATAN(X)

END

FUNCTION ATAN2D(Y,X)

PARAMETER(RADEG=57.2957795130823)

ATAN2D = RADEG \* ATAN2(Y,X)

END

FUNCTION REV(X)

REV = X - AINT(X/360.0)\*360.0

IF (REV.LT.0.0) REV = REV + 360.0

END

FUNCTION CBRT(X)

IF (X.GE.0.0) THEN

CBRT = X \*\* (1.0/3.0)

ELSE

CBRT = -((-X)\*\*(1.0/3.0))

ENDIF

END

The programming language Pascal is not as well equipped with trig functions. We have sin, cos, tan and arctan but nothing more. Therefore we need to write our own arcsin, arccos and arctan2, plus all the trig functions in degrees, and also the functions rev and cbrt. The trig functions in degrees are trivial when the others are defined, therefore I only define arcsin, arccos, arctan2, rev and cbrt:

const pi = 3.14159265358979323846;

half\_pi = 1.57079632679489661923;

function arcsin( x : real ) : real;

begin

if x = 1.0 then

arcsin := half\_pi

else if x = -1.0 then

arcsin := -half\_pi

else

arcsin := arctan( x / sqrt( 1.0 - x\*x ) )

end;

function arccos( x : real ) : real;

begin

arccos := half\_pi - arcsin(x);

end;

function arctan2( y, x : real ) : real;

begin

if x = 0.0 then

begin

if y = 0.0 then

(\* Error! Give error message and stop program \*)

else if y > 0.0 then

arctan2 := half\_pi

else

arctan2 := -half\_pi

end

else

begin

if x > 0.0 then

arctan2 := arctan( y / x )

else if x < 0.0 then

begin

if y >= 0.0 then

arctan2 := arctan( y / x ) + pi

else

arctan2 := arctan( y / x ) - pi

end;

end;

end;

function rev( x : real ) : real;

var rv : real;

begin

rv := x - trunc(x/360.0)\*360.0;

if rv < 0.0 then

rv := rv + 360.0;

rev := rv;

end;

function cbrt( x : real ) : real;

begin

if x > 0.0 then

cbrt := exp ( ln(x) / 3.0 )

else if x < 0.0 then

cbrt := -cbrt(-x)

else

cbrt := 0.0

end;

It's well worth the effort to ensure that all these functions are available. Then you don't need to worry about these details which really don't have much to do with the problem of computing a planetary position.

**3. Rectangular and spherical coordinates**

The position of a planet can be given in one of several ways. Two different ways that we'll use are rectangular and spherical coordinates.

* [Deklination](https://de.wikipedia.org/wiki/Deklination_(Astronomie)) δ {\displaystyle \delta } (Winkelabstand von der Äquatorebene; positives [Vorzeichen](https://de.wikipedia.org/wiki/Vorzeichen_(Zahl)) für die [Nordhalbkugel](https://de.wikipedia.org/wiki/Nordhalbkugel))
* [Rektaszension](https://de.wikipedia.org/wiki/Rektaszension) α {\displaystyle \alpha } (Winkelabstand vom Frühlingspunkt; ein Stern mit größerer Rektaszension wandert später durch den Meridian als einer mit kleinerer Rektaszension)

Suppose a planet is situated at some RA, Decl and r, where RA is the Right Ascension, Decl the declination, and r the distance in some length unit. If r is unknown or irrelevant, set r = 1. Let's convert this to **rectangular coordinates**, x,y,z:

x = r \* cos(RA) \* cos(Decl)

y = r \* sin(RA) \* cos(Decl)

z = r \* sin(Decl)

(before we compute the sine/cosine of RA, we must first convert RA from hours/minutes/seconds to hours + decimals. Then the hours are converted to degrees by multiplying by 15)  
  
If we know the rectangular coordinates, we can convert to **spherical coordinates** by the formulae below:

r = sqrt( x\*x + y\*y + z\*z )

RA = atan2( y, x )

Decl = asin( z / r ) = atan2( z, sqrt( x\*x + y\*y ) )

At the north and south celestial poles, both x and y are zero. Since **atan2(0,0)** is undefined, the RA is undefined too at the celestial poles. The simplest way to handle this is to assign RA some arbitrary value, e.g. zero. Close to the celestial poles the formula asin(z/r) to compute the declination becomes sensitive to round-off errors - here the formula: **atan2(z,sqrt(x\*x+y\*y))** is preferable.  
  
Not only equatorial coordinates can be converted between spherical and rectangular. These conversions can also be applied to ecliptic and horizontal coordinates. Just exchange RA,Decl with long,lat (ecliptic coordinates) or azimuth,altitude (horizontal coordinates).  
  
A coordinate system can be rotated. If a rectangular coordinate system is rotated around, say, the X axis, one can easily compute the new x,y,z coordinates. As an example, let's consider rotating an ecliptic x,y,z system to an equatorial x,y,z system. This rotation is done around the X axis (which points to the Vernal Point, the common point of origin in ecliptic and equatorial coordinates), through an angle of oblecl (the obliquity of the ecliptic, which is approximately 23.4 degrees):

xequat = xeclip

yequat = yeclip \* cos(oblecl) - zeclip \* sin(oblecl)

zequat = yeclip \* sin(oblecl) + zeclip \* cos(oblecl)

Now the x,y,z system is equatorial. It's easily rotated back to ecliptic coordinates by simply switching sign on oblecl:

xeclip = xequat

yeclip = yequat \* cos(-oblecl) - zequat \* sin(-oblecl)

zeclip = yequat \* sin(-oblecl) + zequat \* cos(-oblecl)

When computing sin and cos of -oblecl, one can use the identities:

cos(-x) = cos(x), sin(-x) = -sin(x)

Now let's put this together to convert directly from spherical ecliptic coordinates (long, lat) to spherical equatorial coordinates (RA, Decl). Since the distance r is irrelevant in this case, let's set r=1 for simplicity.  
  
Example: At the Summer Solstice the Sun's ecliptic longitude is 90 degrees. The Sun's ecliptic latitude is always very nearly zero. Suppose the obliquity of the ecliptic is 23.4 degrees:

xeclip = cos(90\_deg) \* cos(0\_deg) = 0.0000

yeclip = sin(90\_deg) \* cos(0\_deg) = 1.0000

zeclip = sin(0\_deg) = 0.0000

Rotate through oblecl = 23.4\_deg:

xequat = 0.0000

yequat = 1.0000 \* cos(23.4\_deg) - 0.0000 \* sin(23.4\_deg)

zequat = 1.0000 \* sin(23.4\_deg) + 0.0000 \* cos(23.4\_deg)

Our equatorial rectangular coordinates become:

x = 0

y = cos(23.4\_deg) = 0.9178

z = sin(23.4\_deg) = 0.3971

The "distance", r, becomes: sqrt( 0.8423 + 0.1577 ) = 1.0000 i.e. unchanged

RA = atan2( 0.9178, 0 ) = 90\_deg

Decl = asin( 0.3971 / 1.0000 ) = 23.40\_deg

Alternatively:

Decl = atan2( 0.3971, sqrt( 0.8423 + 0.0000 ) ) = 23.40\_deg

Here we immediately see how simple it is to compute RA, thanks to the atan2() function: no need to consider in which quadrant it falls, the atan2() function handles this.

**4. The time scale. A test date.**

The time scale used here is a "day number" from 2000 Jan 0.0 TDT, which is the same as 1999 Dec 31.0 TDT, i.e. precisely at midnight TDT at the start of the last day of this century. With the modest accuracy we strive for here, one can usually disregard the difference between TDT (formerly canned ET) and UT.  
  
We call our day number d. It can be computed from a JD (Julian Day Number) or MJD (Modified Julian Day Number) like this:

d = JD - 2451543.5 = MJD - 51543.0

We can also compute d directly from the calendar date like this:

d = 367\*Y - (7\*(Y + ((M+9)/12)))/4 + (275\*M)/9 + D – 730530

Y is the year (all 4 digits!), M the month (1-12) and D the date. In this formula all divisions should be INTEGER divisions. Use "div" instead of "/" in Pascal, and "\" instead of "/" in Microsoft Basic. In C/C++ and FORTRAN it's sufficient to ensure that both operands to "/" are integers.  
  
This formula yields d as an integer, which is valid at the start (at midnight), in UT or TDT, of that calendar date. If you want d for some other time, add UT/24.0 (here the division is a floating-point division!) to the d obtained above.  
  
Example: compute d for 19 april 1990, at 0:00 UT :  
  
We can look up, or compute the JD for this moment, and we'll get: JD = 2448000.5 which yields d = -3543.0  
  
Or we can compute d directly from the calendar date:

d = 367\*1990 - (7\*(1990 + ((4+9)/12)))/4 + (275\*4)/9 + 19 - 730530

d = 730330 - (7\*(1990 + (13/12)))/4 + 1100/9 + 19 - 730530

d = 730330 - (7\*(1990 + 1))/4 + 122 + 19 - 730530

d = 730330 - (7\*1991)/4 + 122 + 19 - 730530

d = 730330 - 13937/4 + 122 + 19 - 730530

d = 730330 - 3484 + 122 + 19 - 730530 = -3543

This moment, 1990 april 19, 0:00 UT/TDT, will be our test date for most numerical examples below. d is negative since our test date, 19 april 1990, is earlier than the "point of origin" of our day number, 31 dec 1999.

**5. The Sun's position.**

Today most people know that the Earth orbits the Sun and not the other way around. But below we'll pretend as if it was the other way around. These orbital elements are thus valid for the Sun's (apparent) orbit around the Earth. All angular values are expressed in degrees:

w = 282.9404\_deg + 4.70935E-5\_deg \* d (longitude of perihelion)

a = 1.000000 (mean distance, a.u.)

e = 0.016709 - 1.151E-9 \* d (eccentricity)

M = 356.0470\_deg + 0.9856002585\_deg \* d (mean anomaly)

We also need the obliquity of the ecliptic, oblecl:

oblecl = 23.4393\_deg - 3.563E-7\_deg \* d

and the Sun's mean longitude, L:

L = w + M

By definition the Sun is (apparently) moving in the plane of the ecliptic. The inclination, i, is therefore zero, and the longitude of the ascending node, N, becomes undefined. For simplicity we'll assign the value zero to N, which means that w, the angle between acending node and perihelion, becomes equal to the longitude of the perihelion.  
  
Now let's compute the Sun's position for our test date 19 april 1990. Earlier we've computed d = -3543.0 which yields:

w = 282.7735\_deg

a = 1.000000

e = 0.016713

M = -3135.9347\_deg

We immediately notice that the mean anomaly, M, will get a large negative value. We use our function rev() to reduce this value to between 0 and 360 degrees. To do this, rev() will need to add 9\*360 = 3240 degrees to this angle:

M = 104.0653\_deg

We also compute:

L = w + M = 386.8388\_deg = 26.8388\_deg

oblecl = 23.4406\_deg

Let's go on computing an auxiliary angle, the eccentric anomaly. Since the eccentricity of the Sun's (i.e. the Earth's) orbit is so small, 0.017, a first approximation of E will be accurate enough. Below E and M are in degrees:

E = M + (180/pi) \* e \* sin(M) \* (1 + e \* cos(M))

When we plug in M and e, we get:

E = 104.9904\_deg

Now we compute the Sun's rectangular coordinates in the plane of the ecliptic, where the X axis points towards the perihelion:

x = r \* cos(v) = cos(E) - e

y = r \* sin(v) = sin(E) \* sqrt(1 - e\*e)

We plug in E and get:

x = -0.275370

y = +0.965834

Convert to distance and true anomaly:

r = sqrt(x\*x + y\*y)

v = arctan2( y, x )

Numerically we get:

r = 1.004323

v = 105.9134\_deg

Now we can compute the longitude of the Sun:

lon = v + w

lon = 105.9134\_deg + 282.7735\_deg = 388.6869\_deg = 28.6869\_deg

We're done!  
  
How close did we get to the correct values? Let's compare with the Astronomical Almanac:

Our results Astron. Almanac Difference

lon 28.6869\_deg 28.6813\_deg +0.0056\_deg = 20"

r 1.004323 1.004311 +0.000012

The error in the Sun's longitude was 20 arc seconds, which is well below our aim of an accuracy of one arc minute. The error in the distance was about 1/3 Earth radius. Not bad!  
  
Finally we'll compute the Sun's ecliptic rectangular coordinates, rotate these to equatorial coordinates, and then compute the Sun's RA and Decl:

x = r \* cos(lon)

y = r \* sin(lon)

z = 0.0

We plug in our longitude:

x = 0.881048

y = 0.482098

z = 0.0

We use oblecl = 23.4406 degrees, and rotate these coordinates:

xequat = 0.881048

yequat = 0.482098 \* cos(23.4406\_deg) - 0.0 \* sin(23.4406\_deg)

zequat = 0.482098 \* sin(23.4406\_deg) + 0.0 \* cos(23.4406\_deg)

which yields:

xequat = 0.881048

yequat = 0.442312

zequat = 0.191778

Convert to RA and Decl:

r = 1.004323 (unchanged)

RA = 26.6580\_deg = 26.6580/15 h = 1.77720 h = 1h 46m 37.9s

Decl = +11.0084\_deg = +11\_deg 0' 30"

* [Deklination](https://de.wikipedia.org/wiki/Deklination_(Astronomie)) δ {\displaystyle \delta } (Winkelabstand von der Äquatorebene; positives [Vorzeichen](https://de.wikipedia.org/wiki/Vorzeichen_(Zahl)) für die [Nordhalbkugel](https://de.wikipedia.org/wiki/Nordhalbkugel))
* [Rektaszension](https://de.wikipedia.org/wiki/Rektaszension) α {\displaystyle \alpha } (Winkelabstand vom Frühlingspunkt; ein Stern mit größerer Rektaszension wandert später durch den Meridian als einer mit kleinerer Rektaszension)

The Astronomical Almanac says:

RA = 1h 46m 36.0s Decl = +11\_deg 0' 22"

**6. Sidereal time and hour angle. Altitude and azimuth**

The Sidereal Time tells the Right Ascension of the part of the sky that's precisely south, i.e. in the meridian. Sidereal Time is a local time, which can be computed from:

SIDTIME = GMST0 + UT + LON/15

where SIDTIME, GMST0 and UT are given in hours + decimals. GMST0 is the Sidereal Time at the Greenwich meridian at 00:00 right now, and UT is the same as Greenwich time. LON is the terrestial longitude in degrees (western longitude is negative, eastern positive). To "convert" the longitude from degrees to hours we divide it by 15. If the Sidereal Time becomes negative, we add 24 hours, if it exceeds 24 hours we subtract 24 hours.  
  
Now, how do we compute GMST0? Simple - we add (or subtract) 180 degrees to (from) L, the Sun's mean longitude, which we've already computed earlier. Then we normalise the result to between 0 and 360 degrees, by applying the rev() function. Finally we divide by 15 to convert degrees to hours:

GMST0 = ( L + 180\_deg ) / 15 = L/15 + 12h

We've already computed L = 26.8388\_deg, which yields:

GMST0 = 26.8388\_deg/15 + 12h = 13.78925 hours

Now let's compute the local Sidereal Time for the time meridian of Central Europe (at 15 deg east longitude = +15 degrees long) on 19 april 1990 at 00:00 UT:

SIDTIME = GMST0 + UT + LON/15 = 13.78925h + 0 + 15\_deg/15 = 14.78925 hours

SIDTIME = 14h 47m 21.3s

To compute the altitude and azimuth we also need to know the Hour Angle, HA. The Hour Angle is zero when the clestial body is in the meridian i.e. in the south (or, from the southern heimsphere, in the north) - this is the moment when the celestial body is at its highest above the horizon.  
  
The Hour Angle increases with time (unless the object is moving faster than the Earth rotates; this is the case for most artificial satellites). It is computed from:

HA = SIDTIME - RA

Here SIDTIME and RA must be expressed in the same unit, hours or degrees. We choose hours:

HA = 14.78925h - 1.77720h = 13.01205h = 195.1808\_deg

If the Hour Angle is 180\_deg the celestial body can be seen (or not be seen, if it's below the horizon) in the north (or in the south, from the southern hemisphere). We get HA = 195\_deg for the Sun, which seems OK since it's around 01:00 local time.  
  
Now we'll convert the Sun's HA = 195.1808\_deg and Decl = +11.0084\_deg to a rectangular (x,y,z) coordinate system where the X axis points to the celestial equator in the south, the Y axis to the horizon in the west, and the Z axis to the north celestial pole: The distance, r, is here irrelevant so we set r=1 for simplicity:

x = cos(HA) \* cos(Decl) = -0.947346

y = sin(HA) \* cos(Decl) = -0.257047

z = sin(Decl) = +0.190953

Now we'll rotate this x,y,z system along an axis going east-west, i.e. the Y axis, in such a way that the Z axis will point to the zenith. At the North Pole the angle of rotation will be zero since there the north celestial pole already is in the zenith. At other latitudes the angle of rotation becomes 90\_deg - latitude. This yields:

xhor = x \* cos(90\_deg - lat) - z \* sin(90\_deg - lat)

yhor = y

zhor = x \* sin(90\_deg - lat) + z \* cos(90\_deg - lat)

Since sin(90\_deg-lat) = cos(lat) (and reverse) we'll get:

xhor = x \* sin(lat) - z \* cos(lat)

yhor = y

zhor = x \* cos(lat) + z \* sin(lat)

Finally we compute our azimuth and altitude:

azimuth = atan2( yhor, xhor ) + 180\_deg

altitude = asin( zhor ) = atan2( zhor, sqrt(xhor\*xhor+yhor\*yhor) )

Why did we add 180\_deg to the azimuth? To adapt to the most common way to specify azimuth: from North (0\_deg) through East (90\_deg), South (180\_deg), West (270\_deg) and back to North. If we didn't add 180\_deg the azimuth would be counted from South through West/etc instead. If you want to use that kind of azimuth, then don't add 180\_deg above.  
  
We select some place in central Scandinavia: the longitude is as before +15\_deg (15\_deg East), and the latitude is +60\_deg (60\_deg N):

xhor = -0.947346 \* sin(60\_deg) - (+0.190953) \* cos(60\_deg) = -0.915902

yhor = -0.257047 = -0.257047

zhor = -0.947346 \* cos(60\_deg) + (+0.190953) \* sin(60\_deg) = -0.308303

Now we've computed the horizontal coordinates in rectangular form. To get azimuth and altitude we convert to spherical coordinates (r=1):

azimuth = atan2(-0.257047,-0.915902) + 180\_deg = 375.6767\_deg = 15.6767\_deg

altitude = asin( -0.308303 ) = -17.9570\_deg

Let's round the final result to two decimals:

azimuth = 15.68\_deg, altitude = -17.96\_deg.

The Sun is thus 17.96\_deg below the horizon at this moment and place. This is very close to astronomical twilight (18\_deg below the horizon).

**7. The Moon's position**

Let's continue by computing the position of the Moon. The computatons will become more complicated, since the Moon doesn't move in the plane of the ecliptic, but in a plane inclined somewhat more than 5 degrees to the ecliptic. Also, the Sun perturbs the Moon's motion significantly, an effect we must account for.  
  
The orbital elements of the Moon are:

N = 125.1228\_deg - 0.0529538083\_deg \* d (Long asc. node)

i = 5.1454\_deg (Inclination)

w = 318.0634\_deg + 0.1643573223\_deg \* d (Arg. of perigee)

a = 60.2666 (Mean distance)

e = 0.054900 (Eccentricity)

M = 115.3654\_deg + 13.0649929509\_deg \* d (Mean anomaly)

The Moon's ascending node is moving in a retrogade ("backwards") direction one revolution in about 18.6 years. The Moon's perigee (the point of the orbit closest to the Earth) moves in a "forwards" direction one revolution in about 8.8 years. The Moon itself moves one revolution in aboout 27.5 days. The mean distance, or semi-major axis, is expressed in Earth equatorial radii).  
  
Let's compute numerical values for the Moon's orbital elements on our test date 19 april 1990 (d = -3543):

N = 312.7381\_deg

i = 5.1454\_deg

w = -264.2546\_deg

a = 60.2666 (Earth equatorial radii)

e = 0.054900

M = -46173.9046\_deg

Now the need for sufficient numerical accuracy becomes obvious. If we would compute M with normal single precision, i.e. only 7 decimal digits of accuracy, then the error in M would here be about 0.01 degrees. Had we selected a date around 1901 or 2099 then the error in M would have been about 0.1 degrees, which is definitely worse than our aim of a maximum error of one or two arc minutes. Therefore, when computing the Moon's mean anomaly, M, it's important to use at least 9 or 10 digits of accuracy.  
  
(This was a real problem around 1980, when microcomputers were a novelty. Around then, pocket calculators usually offered better precision than microcomputers. But these days are long gone: nowadays microcomputers routinely offer double precision (14-16 digits of accuracy) support in hardware; all you need to do is to select a compiler which really suports this.)  
  
All angular elements should be normalized to within 0-360 degrees, by calling the rev() function. We get:

N = 312.7381\_deg

i = 5.1454\_deg

w = 95.7454\_deg

a = 60.2666 (Earth equatorial radii)

e = 0.054900

M = 266.0954\_deg

To normalize M we had to add 129\*360 = 46440 degrees.  
  
Next, we compute E, the eccentric anomaly. We start with a first approximation (E0 and M in degrees):

E0 = M + (180\_deg/pi) \* e \* sin(M) \* (1 + e \* cos(M))

The eccentricity of the Moon's orbit is larger than of the Earth's orbit. This means that our first approximation will have a bigger error - it'll be close to the limit of what we can tolerate within our accuracy aim. If you want to be careful, you should therefore use the iteration formula below: set E0 to our first approximation, compute E1, then set E0 to E1 and compute a new E1, until the difference between E0 and E1 becomes small enough, i.e. smaller than about 0.005 degrees. Then use the last E1 as the final value. In the formula below, E0, E1 and M are in degrees:

E1 = E0 - (E0 - (180\_deg/pi) \* e \* sin(E0) - M) / (1 - e \* cos(E0))

On our test date, the first approximation of E becomes: E=262.9689\_deg The iterations then yield: E = 262.9735\_deg, 262.9735\_deg ......  
  
Now we've computed E - the next step is to compute the Moon's distance and true anomaly. First we compute rectangular (x,y) coordinates in the plane of the lunar orbit:

x = r \* cos(v) = a \* (cos(E) - e)

y = r \* sin(v) = a \* sqrt(1 - e\*e) \* sin(E)

Our test date yields:

x = -10.68095

y = -59.72377

Then we convert this to distance and true anonaly:

r = sqrt( x\*x + y\*y ) = 60.67134 Earth radii

v = atan2( y, x ) = 259.8605\_deg

Now we know the Moon's position in the plane of the lunar orbit. To compute the Moon's position in ecliptic coordinates, we apply these formulae:

xeclip = r \* ( cos(N) \* cos(v+w) - sin(N) \* sin(v+w) \* cos(i) )

yeclip = r \* ( sin(N) \* cos(v+w) + cos(N) \* sin(v+w) \* cos(i) )

zeclip = r \* sin(v+w) \* sin(i)

Our test date yields:

xeclip = +37.65311

yeclip = -47.57180

zeclip = -0.41687

Convert to ecliptic longitude, latitude, and distance:

long = 308.3616\_deg

lat = -0.3937\_deg

r = 60.6713

According to the Astronomical Almanac, the Moon's position at this moment is 306.94\_deg, and the latitude is -0.55\_deg. This differs from our figures by 1.42\_deg in longitude and 0.16\_deg in latitude!!! This difference is much larger than our aim of an error of max 1-2 arc minutes. Why is this so?

**8. The Moon's position with higher accuracy. Perturbations**

The big error in our computed lunar position is because we completely ignored the perturbations on the Moon. Below we'll compute the most important perturbation terms, and then add these as corrections to our previous figures. This will cut down the error to 1-2 arc minutes, or less.  
  
First we need several fundamental arguments:

Sun's mean longitude: Ls (already computed)

Moon's mean longitude: Lm = N + w + M (for the Moon)

Sun's mean anomaly: Ms (already computed)

Moon's mean anomaly: Mm (already computed)

Moon's mean elongation: D = Lm - Ls

Moon's argument of latitude: F = Lm – N

Let's plug in the figures for our test date:

Ms = 104.0653\_deg

Mm = 266.0954\_deg

Ls = 26.8388\_deg

Lm = 312.7381\_deg + 95.7454\_deg + 266.0954\_deg = 674.5789\_deg

= 314.5789\_deg

D = 314.5789\_deg - 26.8388\_deg = 287.7401\_deg

F = 314.5789\_deg - 312.7381\_deg = 1.8408\_deg

Now it's time to compute and add up the 12 largest perturbation terms in longitude, the 5 largest in latitude, and the 2 largest in distance. These are all the perturbation terms with an amplitude larger than 0.01\_deg in longitude resp latitude. In the lunar distance, only the perturbation terms larger than 0.1 Earth radii has been included:  
  
Perturbations in longitude (degrees):

-1.274\_deg \* sin(Mm - 2\*D) (Evection)

+0.658\_deg \* sin(2\*D) (Variation)

-0.186\_deg \* sin(Ms) (Yearly equation)

-0.059\_deg \* sin(2\*Mm - 2\*D)

-0.057\_deg \* sin(Mm - 2\*D + Ms)

+0.053\_deg \* sin(Mm + 2\*D)

+0.046\_deg \* sin(2\*D - Ms)

+0.041\_deg \* sin(Mm - Ms)

-0.035\_deg \* sin(D) (Parallactic equation)

-0.031\_deg \* sin(Mm + Ms)

-0.015\_deg \* sin(2\*F - 2\*D)

+0.011\_deg \* sin(Mm - 4\*D)

Perturbations in latitude (degrees):

-0.173\_deg \* sin(F - 2\*D)

-0.055\_deg \* sin(Mm - F - 2\*D)

-0.046\_deg \* sin(Mm + F - 2\*D)

+0.033\_deg \* sin(F + 2\*D)

+0.017\_deg \* sin(2\*Mm + F)

Perturbations in lunar distance (Earth radii):

-0.58 \* cos(Mm - 2\*D)

-0.46 \* cos(2\*D)

Some of the largest perturbation terms in longitude even have received individual names! The largest perturbation, the Evection, was discovered already by Ptolemy (he made it one of the epicycles in his theory for the Moon's motion). The two next largest perturbations, the Variation and the Yearly equation, were discovered by Tycho Brahe.  
  
If you don't need 1-2 arcmin accuracy, you don't need to compute all these perturbation terms. If you only include the two largest terms in longitude and the largest in latitude, the error in longitude rarely becomes larger than 0.25\_deg, and in latitude rarely larger than 0.15\_deg.  
  
Let's compute these perturbation terms for our test date:

longitude: -0.9847 - 0.3819 - 0.1804 + 0.0405 - 0.0244 + 0.0452 +

0.0428 + 0.0126 - 0.0333 - 0.0055 - 0.0079 - 0.0029

= -1.4132\_deg

latitude: -0.0958 - 0.0414 - 0.0365 - 0.0200 + 0.0018 = -0.1919\_deg

distance: -0.3680 + 0.3745 = +0.0066 Earth radii

Add this to the ecliptic positions we earlier computed:

long = 308.3616\_deg - 1.4132\_deg = 306.9484\_deg

lat = -0.3937\_deg - 0.1919\_deg = -0.5856\_deg

dist = 60.6713 + 0.0066 = 60.6779 Earth radii

Let's compare with the Astronomical Almanac:

longitude 306.94\_deg, latitude -0.55\_deg, distance 60.793 Earth radii

Now the agreement is much better, right?  
  
Let's continue by converting these ecliptic coordinates to Right Ascension and Declination. We do as described earlier: convert the ecliptic longitude/latitude to rectangular (x,y,z) coordinates, rotate this x,y,z, system through an angle corresponding to the obliquity of the ecliptic, then convert back to spherical coordinates. The Moon's distance doesn't matter here, and one can therefore set r=1.0. We get:

RA = 309.5011\_deg

Decl = -19.1032\_deg

According to the Astronomical Almanac:

RA = 309.4881\_deg

Decl = -19.0741\_deg

**9. The Moon's topocentric position.**

The Moon's position, as computed earlier, is geocentric, i.e. as seen by an imaginary observer at the center of the Earth. Real observers dwell on the surface of the Earth, though, and they will see a different position - the topocentric position. This position can differ by more than one degree from the geocentric position. To compute the topocentric positions, we must add a correction to the geocentric position.  
  
Let's start by computing the Moon's parallax, i.e. the apparent size of the (equatorial) radius of the Earth, as seen from the Moon:

mpar = asin( 1/r )

where r is the Moon's distance in Earth radii. It's simplest to apply the correction in horizontal coordinates (azimuth and altitude): within our accuracy aim of 1-2 arc minutes, no correction need to be applied to the azimuth. One need only apply a correction to the altitude above the horizon:

alt\_topoc = alt\_geoc - mpar \* cos(alt\_geoc)

Sometimes one needs to correct for topocentric position directly in equatorial coordinates though, e.g. if one wants to draw on a star map how the Moon passes in front of the Pleiades, as seen from some specific location. Then we need to know the Moon's geocentric Right Ascension and Declination (RA, Decl), the Local Sidereal Time (LST), and our latitude (lat).  
  
Our astronomical latitude (lat) must first be converted to a geocentric latitude (gclat) and distance from the center of the Earth (rho) in Earth equatorial radii. If we only want an approximate topocentric position, it's simplest to pretend that the Earth is a perfect sphere, and simply set:

gclat = lat, rho = 1.0

However, if we do wish to account for the flattening of the Earth, we instead compute:

gclat = lat - 0.1924\_deg \* sin(2\*lat)

rho = 0.99833 + 0.00167 \* cos(2\*lat)

Next we compute the Moon's geocentric Hour Angle (HA):

HA = LST - RA

We also need an auxiliary angle, g:

g = atan( tan(gclat) / cos(HA) )

Now we're ready to convert the geocentric Right Ascention and Declination (RA, Decl) to their topocentric values (topRA, topDecl):

topRA = RA - mpar \* rho \* cos(gclat) \* sin(HA) / cos(Decl)

topDecl = Decl - mpar \* rho \* sin(gclat) \* sin(g - Decl) / sin(g)

Let's do this correction for our test date and for the geographical position 15 deg E longitude (= +15\_deg) and 60 deg N latitude (= +60\_deg). Earlier we computed the Local Sidereal Time as LST = SIDTIME = 14.78925 hours. Multiply by 15 to get degrees: LST = 221.8388\_deg  
  
The Moon's Hour Angle becomes:

HA = LST - RA = -87.6623\_deg = 272.3377\_deg

Our latitude +60\_deg yields the following geocentric latitude and distance from the Earth's center:

gclat = 59.83\_deg rho = 0.9975

We've already computed the Moon's distance as 60.6779 Earth radii, which means the Moon's parallax is:

mpar = 0.9443\_deg

The auxiliary angle g becomes:

g = 88.642

And finally the Moon's topocentric position becomes:

topRA = 309.5011\_deg - (-0.5006\_deg) = 310.0017\_deg

topDecl = -19.1032\_deg - (+0.7758\_deg) = -19.8790\_deg

This correction to topocentric position can also be applied to the Sun and the planets. But since they're much farther away, the correction becomes much smaller. It's largest for Venus at inferior conjunction, when Venus' parallax is somewhat larger than 32 arc seconds. Within our aim of obtaining a final accuracy of 1-2 arc minutes, it might barely be justified to correct to topocentric position when Venus is close to inferior conjunction, and perhaps also when Mars is at a favourable opposition. But in all other cases this correction can safely be ignored within our accuracy aim. We only need to worry about the Moon in this case.  
  
If you want to compute topocentric coordinates for the planets anyway, you do it the same way as for the Moon, with one exception: the parallax of the planet (ppar) is computed from this formula:

ppar = (8.794/3600)\_deg / r

where r is the distance of the planet from the Earth, in astronomical units.

**10. The orbital elements of the planets**

To compute the positions of the major planets, we first must compute their orbital elements:  
  
Mercury:

N = 48.3313\_deg + 3.24587E-5\_deg \* d (Long of asc. node)

i = 7.0047\_deg + 5.00E-8\_deg \* d (Inclination)

w = 29.1241\_deg + 1.01444E-5\_deg \* d (Argument of perihelion)

a = 0.387098 (Semi-major axis)

e = 0.205635 + 5.59E-10 \* d (Eccentricity)

M = 168.6562\_deg + 4.0923344368\_deg \* d (Mean anonaly)

Venus:

N = 76.6799\_deg + 2.46590E-5\_deg \* d

i = 3.3946\_deg + 2.75E-8\_deg \* d

w = 54.8910\_deg + 1.38374E-5\_deg \* d

a = 0.723330

e = 0.006773 - 1.302E-9 \* d

M = 48.0052\_deg + 1.6021302244\_deg \* d

Mars:

N = 49.5574\_deg + 2.11081E-5\_deg \* d

i = 1.8497\_deg - 1.78E-8\_deg \* d

w = 286.5016\_deg + 2.92961E-5\_deg \* d

a = 1.523688

e = 0.093405 + 2.516E-9 \* d

M = 18.6021\_deg + 0.5240207766\_deg \* d

Jupiter:

N = 100.4542\_deg + 2.76854E-5\_deg \* d

i = 1.3030\_deg - 1.557E-7\_deg \* d

w = 273.8777\_deg + 1.64505E-5\_deg \* d

a = 5.20256

e = 0.048498 + 4.469E-9 \* d

M = 19.8950\_deg + 0.0830853001\_deg \* d

Saturn:

N = 113.6634\_deg + 2.38980E-5\_deg \* d

i = 2.4886\_deg - 1.081E-7\_deg \* d

w = 339.3939\_deg + 2.97661E-5\_deg \* d

a = 9.55475

e = 0.055546 - 9.499E-9 \* d

M = 316.9670\_deg + 0.0334442282\_deg \* d

Uranus:

N = 74.0005\_deg + 1.3978E-5\_deg \* d

i = 0.7733\_deg + 1.9E-8\_deg \* d

w = 96.6612\_deg + 3.0565E-5\_deg \* d

a = 19.18171 - 1.55E-8 \* d

e = 0.047318 + 7.45E-9 \* d

M = 142.5905\_deg + 0.011725806\_deg \* d

Neptune:

N = 131.7806\_deg + 3.0173E-5\_deg \* d

i = 1.7700\_deg - 2.55E-7\_deg \* d

w = 272.8461\_deg - 6.027E-6\_deg \* d

a = 30.05826 + 3.313E-8 \* d

e = 0.008606 + 2.15E-9 \* d

M = 260.2471\_deg + 0.005995147\_deg \* d

Let's compute all these elements for our test date, 19 april 1990 0h UT:

N i w a e M

deg deg deg a.e. deg

Mercury 48.2163 7.0045 29.0882 0.387098 0.205633 69.5153

Venus 76.5925 3.3945 54.8420 0.723330 0.006778 131.6578

Mars 49.4826 1.8498 286.3978 1.523688 0.093396 321.9965

Jupiter 100.3561 1.3036 273.8194 5.20256 0.048482 85.5238

Saturn 113.5787 2.4890 339.2884 9.55475 0.055580 198.4741

Uranus 73.9510 0.7732 96.5529 19.18176 0.047292 101.0460

Neptune 131.6737 1.7709 272.8675 30.05814 0.008598 239.0063

**11. The heliocentric positions of the planets**

The heliocentric ecliptic coordinates of the planets are computed in the same way as we computed the geocentric ecliptic coordinates of the Moon: first we compute E, the eccentric anomaly. Several planetary orbits have quite high eccentricities, which means we must use the iteration formula to obtain an accurate value of E. When we know E, we compute, as earlier, the distance r ("radius vector") and the true anomaly, v. Then we compute ecliptic rectangular (x,y,z) coordinates as we did for the Moon. Since the Moon orbits the Earth, this position of the Moon was geocentric. The planets though orbit the Sun, which means we get heliocentric positions instead. Also the semi-major axis, a, and the distance, r, which was given in Earth radii for the Moon, are given in astronomical units for the planets, where one astronomical unit is 149.6 million kilometers.  
  
Let's do this for Mercury on our test date: the first approximation of E is 81.3464\_deg, and following iterations yield 81.1572\_deg, 81.1572\_deg .... From this we find:

r = 0.374862

v = 93.0727\_deg

Mercury's heliocentric ecliptic rectangular coordinates become:

x = -0.367821

y = +0.061084

z = +0.038699

Convert to spherical coordinates:

lon = 170.5709\_deg

lat = +5.9255\_deg

r = 0.374862

The Astronomical Almanac gives these figures:

lon = 170.5701\_deg

lat = +5.9258\_deg

r = 0.374856

The agreement is almost perfect! The discrepancy is only a few arc seconds. This is because it's quite easy to get an accurate position for Mercury: it's close to the Sun where the Sun's gravitational field is strongest, and therefore perturbations from the other planets will be smallest for Mercury.  
  
If we compute the heliocentric longitude, latitude and distance for the other planets from their orbital elements, we get:

Heliocentric

longitude latitude distance

lon lat r

Mercury 170.5709\_deg +5.9255\_deg 0.374862

Venus 263.6570\_deg -0.4180\_deg 0.726607

Mars 290.6297\_deg -1.6203\_deg 1.417194

Jupiter 105.2543\_deg +0.1113\_deg 5.19508

Saturn 289.4523\_deg +0.1792\_deg 10.06118

Uranus 276.7999\_deg -0.3003\_deg 19.39628

Neptune 282.7192\_deg +0.8575\_deg 30.19284

For e.g. Saturn, the Astronomical Almanac says:

lon = 289.3864\_deg

lat = +0.1816\_deg

r = 10.01850

The difference is here much larger! For Mercury our discrepancy was only a few arc seconds, but for Saturn it's up to four arc minutes! And still we've been lucky, since sometimes the discrepancy can be up to one full degree for Saturn. This is the planet that's perturbed most severely, mostly by Jupiter.

**12. Higher accuracy - perturbations**

To reach our aim of a final accuracy of 1-2 arc minutes, we must compute Jupiter's and Saturn's perturbations on each other, and their perturbations on Uranus. The perturbations on, and by, other planets can be ignored, with our aim for 1-2 arcmin accuracy.  
  
First we need three fundamental arguments:

Jupiters mean anomaly: Mj

Saturn mean anomaly: Ms

Uranus mean anomaly: Mu

Then these terms should be added to Jupiter's heliocentric longitude:

-0.332\_deg \* sin(2\*Mj - 5\*Ms - 67.6\_deg)

-0.056\_deg \* sin(2\*Mj - 2\*Ms + 21\_deg)

+0.042\_deg \* sin(3\*Mj - 5\*Ms + 21\_deg)

-0.036\_deg \* sin(Mj - 2\*Ms)

+0.022\_deg \* cos(Mj - Ms)

+0.023\_deg \* sin(2\*Mj - 3\*Ms + 52\_deg)

-0.016\_deg \* sin(Mj - 5\*Ms - 69\_deg)

For Saturn we must correct both the longitude and latitude. Add this to Saturn's heliocentric longitude:

+0.812\_deg \* sin(2\*Mj - 5\*Ms - 67.6\_deg)

-0.229\_deg \* cos(2\*Mj - 4\*Ms - 2\_deg)

+0.119\_deg \* sin(Mj - 2\*Ms - 3\_deg)

+0.046\_deg \* sin(2\*Mj - 6\*Ms - 69\_deg)

+0.014\_deg \* sin(Mj - 3\*Ms + 32\_deg)

and to Saturn's heliocentric latitude these terms should be added:

-0.020\_deg \* cos(2\*Mj - 4\*Ms - 2\_deg)

+0.018\_deg \* sin(2\*Mj - 6\*Ms - 49\_deg)

Finally, add this to Uranus heliocentric longitude:

+0.040\_deg \* sin(Ms - 2\*Mu + 6\_deg)

+0.035\_deg \* sin(Ms - 3\*Mu + 33\_deg)

-0.015\_deg \* sin(Mj - Mu + 20\_deg)

The perturbation terms above are all terms having an amplitude of 0.01 degrees or more. We ignore all perturbations in the distances of the planets, since a modest perturbation in distance won't affect the apparent position very much.  
  
The largest perturbation term, "the grand Jupiter-Saturn term" is the perturbation in longitude with the largest amplitude in both Jupiter and Saturn. Its period is 918 years, and its amplitude is a large part of a degree for both Jupiter and Saturn. There is also a "grand Saturn-Uranus term", which has a period of 560 years and an amplitude of 0.035 degrees for Uranus, but less than 0.01 degrees for Saturn. Other included terms have periods between 14 and 100 years. Finally we should mention the "grand Uranus-Neptune term", which has a period if 4200 years and an amplitude of almost one degree. It's not included in our perturbation terms here, instead its effects have been included in the orbital elements for Uranus and Neptune. This is why the mean distances of Uranus and Neptune are varying.  
  
If we compute the perturbations for our test date, we get:

Mj = 85.5238\_deg Ms = 198.4741\_deg Mu = 101.0460:

Perturbations in Jupiter's longitude:

+ 0.0637\_deg - 0.0236\_deg + 0.0038\_deg - 0.0270\_deg - 0.0086\_deg

- 0.0049\_deg - 0.0155\_deg = -0.0120\_deg

Jupiter's heliocentric longitude, with perturbations: 105.2423\_deg

The Astronomical Almanac says: 105.2603\_deg

Perturbations in Saturn's longitude:

-0.1560\_deg + 0.0206\_deg + 0.0850\_deg - 0.0070\_deg - 0.0124\_deg

= - 0.0699\_deg

Perturbations in Saturn's latitude:

+0.0018\_deg + 0.0035\_deg = +0.0053\_deg

Saturns position, with perturbations: 289.3824\_deg +0.1845\_deg

The Astronomical Almanac says: 289.3864\_deg +0.1816\_deg

Perturbations in Uranus' longitude:

+0.0017\_deg - 0.0332\_deg - 0.0012\_deg = -0.0327\_deg

Uranus heliocentric longitude, with perturbations: 276.7672\_deg

The Astronomical Almanac says: 276.7706\_deg

**13. Precession**

The planetary positions computed here are for "the epoch of the day", i.e. relative to the celestial equator and ecliptic at the moment. Sometimes you need to use some other epoch, e.g. some standard epoch like 1950.0 or 2000.0. Due to our modest accuracy requirement of 1-2 arc minutes, we need not distinguish J2000.0 from B2000.0, it's enough to simply use 2000.0.  
  
We will simplify the precession correction further by doing it in eliptic coordinates: the correction is simply done by adding

3.82394E-5\_deg \* ( 365.2422 \* ( epoch - 2000.0 ) - d )

to the ecliptic longitude. We ignore precession in ecliptic latitude. "epoch" is the epoch we wish to precess to, and "d" is the "day number" we used when computing our planetary positions.  
  
Example: if we wish to precess computations done at our test date 19 April 1990, when d = -3543, we add the quantity below (degrees) to the ecliptic longitude:

3.82394E-5\_deg \* ( 365.2422 \* ( 2000.0 - 2000.0 ) - (-3543) ) =

= 0.1355\_deg

So we simply add 0.1355\_deg to our ecliptic longitude to get the position at 2000.0.

**14. Geocentric positions of the planets**

To convert the planets' heliocentric positions to geocentric positions, we simply add the Sun's rectangular (x,y,z) coordinates to the rectangular (x,y,z) heliocentric coordinates of the planet:  
  
Let's do this for Mercury on our test date - we add the x, y and z coordinates separately:

xsun = +0.881048 ysun = +0.482098 zsun = 0.0

xplan = -0.367821 yplan = +0.061084 zplan = +0.038699

-----------------------------------------------------------------

xgeoc = +0.513227 ygeoc = +0.543182 zgeoc = +0.038699

Now we have rectangular geocentric coordinates of Mercury. If we wish, we can convert this to spherical coordinates - then we get geocentric ecliptic longitude and latitude. This is useful if we want to precess the position to some other epoch: we then simply add the approproate precessional value to the longitude. Then we can convert back to rectangular coordinates.  
  
But for the moment we want the "epoch of the day": let's rotate the x,y,z, coordinates around the X axis, as described earlier. Then we'll get equatorial rectangular geocentric (whew!) coordinates:

xequat = +0.513227 yequat = +0.482961 zequat = 0.251582

We can convert these coordinates to spherical coordinates, and then we'll (finally!) get geocentric Right Ascension, Declination and distance for Mercury:

RA = 43.2598\_deg Decl = +19.6460\_deg R = 0.748296

Note that the distance now is different. This is quite natural since the distance now is from the Earth and not, as earlier, from the Sun.  
  
Let's conclude by checking the values given by the Astronomical Almanac:

RA = 43.2535\_deg Decl = +19.6458\_deg R = 0.748262

**15. The elongation and physical ephemerides of the planets**

When we finally have completed our computation of the heliocentric and geocentric coordinates of the planets, it could also be interesting to know what the planet will look like. How large will it appear? What are its phase and magnitude (brightness)? These computations are much simpler than the computations of the positions.  
  
Let's start by computing the apparent diameter of the planet:

d = d0 / R

R is the planet's geocentric distance in astronomical units, and d is the planet's apparent diameter at a distance of 1 astronomical unit. d0 is of course different for each planet. The values below are given in seconds of arc. Some planets have different equatorial and polar diameters:

Mercury 6.74"

Venus 16.92"

Earth 17.59" equ 17.53" pol

Mars 9.36" equ 9.28" pol

Jupiter 196.94" equ 185.08" pol

Saturn 165.6" equ 150.8" pol

Uranus 65.8" equ 62.1" pol

Neptune 62.2" equ 60.9" pol

The Sun's apparent diameter at 1 astronomical unit is 1919.26". The Moon's apparent diameter is:

d = 1873.7" \* 60 / r

where r is the Moon's distance in Earth radii.  
  
Two other quantities we'd like to know are the phase angle and the elongation.  
  
The phase angle tells us the phase: if it's zero the planet appears "full", if it's 90 degrees it appears "half", and if it's 180 degrees it appears "new". Only the Moon and the inferior planets (i.e. Mercury and Venus) can have phase angles exceeding about 50 degrees.  
  
The elongation is the apparent angular distance of the planet from the Sun. If the elongation is smaller than about 20 degrees, the planet is hard to observe, and if it's smaller than about 10 degrees it's usually not possible to observe the planet.  
  
To compute phase angle and elongation we need to know the planet's heliocentric distance, r, its geocentric distance, R, and the distance to the Sun, s. Now we can compute the phase angle, FV, and the elongation, elong:

elong = acos( ( s\*s + R\*R - r\*r ) / (2\*s\*R) )

FV = acos( ( r\*r + R\*R - s\*s ) / (2\*r\*R) )

When we know the phase angle, we can easily compute the phase:

phase = ( 1 + cos(FV) ) / 2 = hav(180\_deg - FV)

hav is the "haversine" function. The "haversine" (or "half versine") is an old and now obsolete trigonometric function. It's defined as:

hav(x) = ( 1 - cos(x) ) / 2 = sin^2 (x/2)

As usual we must use a different procedure for the Moon. Since the Moon is so close to the Earth, the procedure above would introduce too big errors. Instead we use the Moon's ecliptic longitude and latitude, mlon and mlat, and the Sun's ecliptic longitude, mlon, to compute first the elongation, then the phase angle, of the Moon:

elong = acos( cos(slon - mlon) \* cos(mlat) )

FV = 180\_deg – elong

Finally we'll compute the magnitude (or brightness) of the planets. Here we need to use a formula that's different for each planet. The phase angle, FV, is in degrees:

Mercury: -0.36 + 5\*log10(r\*R) + 0.027 \* FV + 2.2E-13 \* FV\*\*6

Venus: -4.34 + 5\*log10(r\*R) + 0.013 \* FV + 4.2E-7 \* FV\*\*3

Mars: -1.51 + 5\*log10(r\*R) + 0.016 \* FV

Jupiter: -9.25 + 5\*log10(r\*R) + 0.014 \* FV

Saturn: -9.0 + 5\*log10(r\*R) + 0.044 \* FV + ring\_magn

Uranus: -7.15 + 5\*log10(r\*R) + 0.001 \* FV

Neptune: -6.90 + 5\*log10(r\*R) + 0.001 \* FV

\*\* is the power operator, thus FV\*\*6 is the phase angle (in degrees) raised to the sixth power. If FV is 150 degrees, then FV\*\*6 becomes ca 1.14E+13, which is a quite large number.  
  
Saturn needs special treatment due to its rings: when Saturn's rings are "open" then Saturn will appear much brighter than when we view Saturn's rings edgewise. We'll compute ring\_mang like this:

ring\_magn = -2.6 \* sin(abs(B)) + 1.2 \* (sin(B))\*\*2

Here B is the tilt of Saturn's rings which we also need to compute. Then we start with Saturn's geocentric ecliptic longitude and latitude (los, las) which we've already computed. We also need the tilt of the rings to the ecliptic, ir, and the "ascending node" of the plane of the rings, Nr:

ir = 28.06\_deg

Nr = 169.51\_deg + 3.82E-5\_deg \* d

Here d is our "day number" which we've used so many times before. For our test date d = -3543. Now let's compute the tilt of the rings:

B = asin( sin(las) \* cos(ir) - cos(las) \* sin(ir) \* sin(los-Nr) )

This concludes our computation of the magnitudes of the planets.

**16. The positions of comets. Comet Encke and Levy.**

If you want to compute the position of a comet or an asteroid, you must have access to orbital elements that still are valid. One set of orbital elements isn't valid forever. For instance if you try to use the 1986 orbital elements of comet Halley to compute its appearance in either 1910 or 2061, you'll get very large errors in your computed positions - sometimes the errors will be 90 degrees or more.  
  
Comets will usually have a new set of orbital elements computed for each perihelion. The comets are perturbed most severely when they're close to aphelion, far away from the gravity of the Sun but maybe much closer to Jupiter, Saturn, Uranus or Neptune. When the comet is passing through the inner solar system, the perturbations are usually so small that the same set of orbital elements can be used for the entire apparition.  
  
Orbital elements for an asteroid should preferably not be more than about one year old. If your accuracy requirements are lower, you can of course use older elements. If you use orbital elements that are five years old for a main-belt asteroid, then your computed positions can be several degrees in error. If the orbital elements are less than one year old, the errors usually stay below approximately one arc minute, for a main-belt asteroid.  
  
If you have access to valid orbital elements for a comet or an asteroid, proceed as below to compute its position at some date:  
  
1. If necessary, precess the angular elements N,w,i to the epoch of today. The simples way to do this is to add the precession angle to N, the longitude of the ascending node. This method is approximate, but it's good enough for our accuracy aim of 1-2 arc minutes.  
  
2. Compute the day number for the time or perihelion, call it D. Then compute the number of days since perihelion, d - D (before perihelion this number is of course negative).  
  
3. If the orbit is elliptical, compute the Mean Anomaly, M. Then compute r, the heliocentric distance, and v, the true anomaly.  
  
4. If the orbit is a parabola, or close to a parabola (the eccentricity is 1.0 or nearly 1.0), then the algorithms for elliptical orbits will break down. Then use another algorithm, presented below, to compute r, the heliocentric distance, and v, the true anomaly, for near-parabolic orbits.  
  
5. When you know r and v, proceed as with the planets: compute first the heliocentric, then the geocentric, position.  
  
6. If needed, precess the final position to the desired epoch, e.g. 2000.0  
  
A quantity we'll encounter here is Gauss' gravitational constant, k. This constant links the Sun's mass with our time unit (the day) and the length unit (the astronomical unit). The EXACT value of Gauss' gravitational constant k is:

k = 0.01720209895 (exactly!)

If the orbit is elliptical, and if the perihelion distance, q, is given instead of the mean distance, a, we start by computing the mean distance a from the perihelion distance q and the eccentricity e:

a = q / (1 - e)

Now we compute the Mean Anomaly, M:

M = (180\_deg/pi) \* (d - D) \* k / (a \*\* 1.5)

a \*\* 1.5 is most easily computed as: sqrt(a\*a\*a)

Now we know the Mean Anomaly, M. We proceed as for a planetary orbit by computing E, the eccentric anomaly. Since comet and asteroid orbits often have high eccentricities, we must use the iteration formula given earlier, and be sure to iterate until we get convergence for the value of E.  
  
The orbital period for a comet or an asteroid in elliptic orbit is (P in days):

P = 2 \* pi \* (a \*\* 1.5) / k

If the comet's orbit is a parabola, the algorithm for elliptic orbits will break down: the semi-major axis and the orbital period will be infinite, and the Mean Anomaly will be zero. Then we must proceed in a different way. For a parabolic orbit we start by computing the quantities a, b and w (where a is not at all related to a for an elliptic orbit):

a = 1.5 \* (d - D) \* k / sqrt(2 \* q\*q\*q)

b = sqrt( 1 + a\*a )

w = cbrt(b + a) - cbrt(b - a)

cbrt is the Cubic Root function. Finally we compute the true anomaly, v, and the heliocentric distiance, r:

v = 2 \* atan(w)

r = q \* ( 1 + w\*w )

From here we can proceed as usual.  
  
Finally we have the case that's most common for newly discovered comets: the orbit isn't an exact parabola, but very nearly so. It's eccentricity is slightly below, or slightly above, one. The algorithm presented here can be used for eccentricities between about 0.98 and 1.02. If the eccentricity is smaller than 0.98 the elliptic algorithm should be used instead. No known comet has an eccentricity exceeding 1.02.  
  
As for the purely parabolic orbit, we start by computing the time since perihelion in days, d - D, and the perihelion distance, q. We also need to know the eccentricity, e. Then we can proceed as:

a = 0.75 \* (d - D) \* k \* sqrt( (1 + e) / (q\*q\*q) )

b = sqrt( 1 + a\*a )

W = cbrt(b + a) - cbrt(b - a)

f = (1 - e) / (1 + e)

a1 = (2/3) + (2/5) \* W\*W

a2 = (7/5) + (33/35) \* W\*W + (37/175) \* W\*\*4

a3 = W\*W \* ( (432/175) + (956/1125) \* W\*W + (84/1575) \* W\*\*4 )

C = W\*W / (1 + W\*W)

g = f \* C\*C

w = W \* ( 1 + f \* C \* ( a1 + a2\*g + a3\*g\*g ) )

v = 2 \* atan(w)

r = q \* ( 1 + w\*w ) / ( 1 + w\*w \* f )

This algorithm yields the true anomaly, v, and the heliocentric distance, r, for a nearly-parabolic orbit.  
  
Now it's time for a practical example. Let's select two of the comets that were seen in the autumn of 1990: Comet Encke, a well-known periodic comet, and comet Levy, which was easily seen towards a dark sky in the autumn of 1990. When passing the inner solar system, the orbit of comet Levy was slightly hyperbolic.  
  
According the the Handbook of the British Astronomical Association the orbital elements for comet Encke in 1990 are:

T = 1990 Oct 28.54502 TDT

e = 0.8502196

q = 0.3308858

w = 186.24444\_deg

N = 334.04096\_deg 1950.0

i = 11.93911\_deg

The orbital elements for comet Levy are (BAA Circular 704):

T = 1990 Oct 24.6954 ET

e = 1.000270

q = 0.93858

w = 242.6797\_deg

N = 138.6637\_deg 1950.0

i = 131.5856\_deg

Let's also choose another test date, when both these comets were visible: 1990 Aug 22, 0t UT, which yields a "day number" d = -3418.0  
  
Now we compute the day numbers at perihelion for these two comets. We get for comet Encke:

D = -3350.45498 d - D = -67.54502

and for comet Levy:

D = -3354.3046 d - D = -63.6954

We'll continue by computing the Mean Anomaly for comet Encke:

M = -20.2751\_deg = 339.7249\_deg

The first approximation plus successive approximation for the Eccentric anomaly, E, becomes (degrees):

E = 309.3811 293.5105 295.8474 295.9061 295.9061\_deg ....

Here we clearly see the great need for iteration: the initial approximation differs from the final value by 14 degrees. Finally we compute the true anomaly, v, and heliocentric distance, r, for comet Encke:

v = 228.8837\_deg

r = 1.3885

Now it's time for comet Levy: we'll compute the true anomaly, v, and the heliocentric distance, r, for Levy in two different ways. First we'll pretend that the orbit of Levy is an exact parabola. We get:

a = -1.2780823 b = 1.6228045 w = -0.7250189

v = -71.8856\_deg

r = 1.431947

Then we repeat the computation but accounts for the fact that Levy's orbit deviates slightly from a parabola. We get:

a = -1.2781686 b = 1.6228724 W = -0.7250566

c = 2.9022000 f = -1.3498E-4 g = -1.60258E-5

a1= 0.8769495 a2= 1.9540987 a3= 1.5403455

w = -0.7250270

v = -71.8863\_deg

r = 1.432059

The difference is small in this case - only 0.0007 degrees or 2.5 arc seconds in true anomaly, and 0.000112 a.u. in heliocentric distance. Here it would have been sufficient to treat Levy's orbit as an exact parabola.  
  
Now we know the true anomaly, v, and the heliocentric distance, r, for both Encke and Levy. Next we proceed by precessing N, the longitude of the ascending node, from 1950.0 to the "epoch of the day". Let's compute the precession angle from 1950.0 to 1990 Aug 22:

prec = 3.82394E-5\_deg \* ( 365.2422 \* ( 1950.0 - 2000.0 ) - (-3418) )

prec = -0.5676\_deg

To precess from 1990 Aug 22 to 1950.0, we should add this angle to N. But now we want to do the opposite: precess from 1950.0 to 1990 Aug 22, therefore we must instead subtract this angle:  
  
For comet Encke we get:

N = 334.04096\_deg - (-0.5676\_deg) = 334.60856\_deg

and for comet Levy we get:

N = 138.6637\_deg - (-0.5676\_deg) = 139.2313\_deg

Using this modified value for N we proceed just like for the planets. I won't repeat the details, but merely state some intermediate and final results:

Sun's position: x = -0.863890 y = +0.526123

Heliocentric: Encke Levy

x +1.195087 +1.169908

y +0.666455 -0.807922

z +0.235663 +0.171375

Geoc., eclipt.: Encke Levy

x +0.331197 +0.306018

y +1.192579 -0.281799

z +0.235663 +0.171375

Geoc., equat.: Encke Levy

x +0.331197 +0.306018

y +1.000414 -0.326716

z +0.690619 +0.045133

RA 71.6824\_deg 313.1264\_deg

Decl +33.2390\_deg +5.7572\_deg

R 1.259950 0.449919

These positions are for the "epoch of the day". If you want positions for some standard epoch, e.g. 2000.0, these positions must be precessed to that epoch.  
  
Finally some notes about computing the magnitude of a comet. To accurately predict a comet's magnitude is usually hard and sometimes impossible. It's fairly common that a magnitude prediction is off by 1-2 magnitudes or even more. For comet Levy the magnitude formula looked like this:

m = 4.0 \* 5\*log10(R) + 10\*log10(r)

where R is the geocentric distance and r the heliocentric distance. The general case is:

m = G \* 5\*log10(R) + H\*log10(r)

where H usually is around 10. If H is unknown, it's usually assumed to be 10. Each comet has it's own G and H.  
  
Some comets have a different magnitude formula. One good example is comet Encke, where the magnitude formula looks like this:

m1 = 10.8 + 5\*log10(R) + 3.55 \* ( r\*\*1.8 - 1 )

"m1" refers to the total magnitude of the comet. There is another cometary magnitude, "m2", which refers to the magnitude of the nucleus of the comet. The magnitude formula for Encke's m2 magnitude looks like this:

m2 = 14.5 + 5\*log10(R) + 5\*log10(r) + 0.03\*FV

Here FV is the phase angle. This kind of magnitude formula looks very much like the magnitude formula of asteroids, for a very good reason: when a comet is far away form the Sun, no gases are evaporated from the surface of the comet. Then the comet has no tail (of course) and no coma, only a nucleus. Which means the comet then behaves much like an asteroid.  
  
During the last few years it's become increasingly obvious that comets and asteroids often are similar kinds of solar-system objects. The asteroid (2060) Chiron has displayed cometary activity and is now also considered a comet. And in some cases comets that have "disappeared" have been re-discovered as asteroids! Apparently they "ran out of gas" and what remains of the former comet is only rock, i.e. an asteroid.