Facultatea de Electronica, Telecomunicatii si Tehnologia informatiei

The sunset and sunrise on Earth around the year

Student: Gligor Catalin Andrei
Teacher: Cirlugea Mihaela Cristina

Year of elaboration: 2022-2023

Table of Contents

Introduction

MATLAB is a proprietary multi-paradigm programming language and numeric computing environment developed by MathWorks. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages.

The project's theme is about visualizing how the sunset and sunrise vary around the year in multiple locations on Earth, using global coordinates and time zone, as well as visualizing the time equation.

The Earth's orbit around the Sun is elliptical, rather than circular, and the Earth's axis of rotation is not perpendicular to the plane of the orbit. This non-circularity of the orbit and the tilt of the Earth's axis of rotation both contribute to the uneven changes in the times of sunrise and sunset. For example, as you noticed, the Sun rises only a little earlier each day in January, but sets noticeably later each day.

Sunrises and sunsets are different from one part of the world to another, that's because sunrise and sunset times depend on a number of variables, including longitude, latitude, altitude, and location in relationship to your specific Time Zone. Here's a look at how each of these factors affects your area's sunrise and sunset times.

I believe the project is unique in its category because apart from the plot regarding sunrise and sunset and the estimated times, it also displays a plot with the equation of time, it is relatively accurate, with a maximum of about 20 minutes error, the GUI has a pleasant design, it contains a photo for a better understanding of the latitude and longitude and also the coordinates, and it also contains a nice little animation at the beginning with the sun rising and setting.

Theoretical aspect

To figure out the sunset and sunrise times knowing the coordinates and the time zone is not as hard as it sounds, thankfully. That's because for thousands of years, humans have been fascinated by the Sun and its movement and have studied these aspects since the Greek mathematicians. Thus, in today's day and age you have all the formulas you need at a couple of clicks away. For this project I have used **NOAA Global Monitoring Division** formulas, publically available as a PDF provided by them, named **General Solar Position Calculations.**

First, the fractional year (γ) is calculated, in radians.

$$\gamma = \frac{2\pi}{365} * (\text{day\_of\_year} - 1 + \frac{hour - 12}{24})$$

(For leap years, use 366 instead of 365 in the denominator.)

From γ, we can estimate the equation of time (in minutes) and the solar declination angle (in radians).

$$eqtime = 229.18*(0.000075 + 0.001868\cos(\gamma) - 0.032077\sin(\gamma) - 0.014615\cos(2\gamma) - 0.040849\sin(2\gamma))$$

$$decl = 0.006918 - 0.399912\cos(\gamma) + 0.070257\sin(\gamma) - 0.006758\cos(2\gamma) + 0.000907\sin(2\gamma) - 0.002697\cos(3\gamma) + 0.00148\sin(3\gamma)$$

For the special case of sunrise or sunset, the zenith is set to 90.833° (the approximate correction for atmospheric refraction at sunrise and sunset, and the size of the solar disk), and the hour angle becomes:

$$ha = \pm arccos\left\{\frac{\cos(90.833)}{\cos(lat)\cos(decl)} - \tan(lat)\tan(decl)\right\}$$

where the positive number corresponds to sunrise, negative to sunset.

Then the UTC time of sunrise (or sunset) in minutes is:

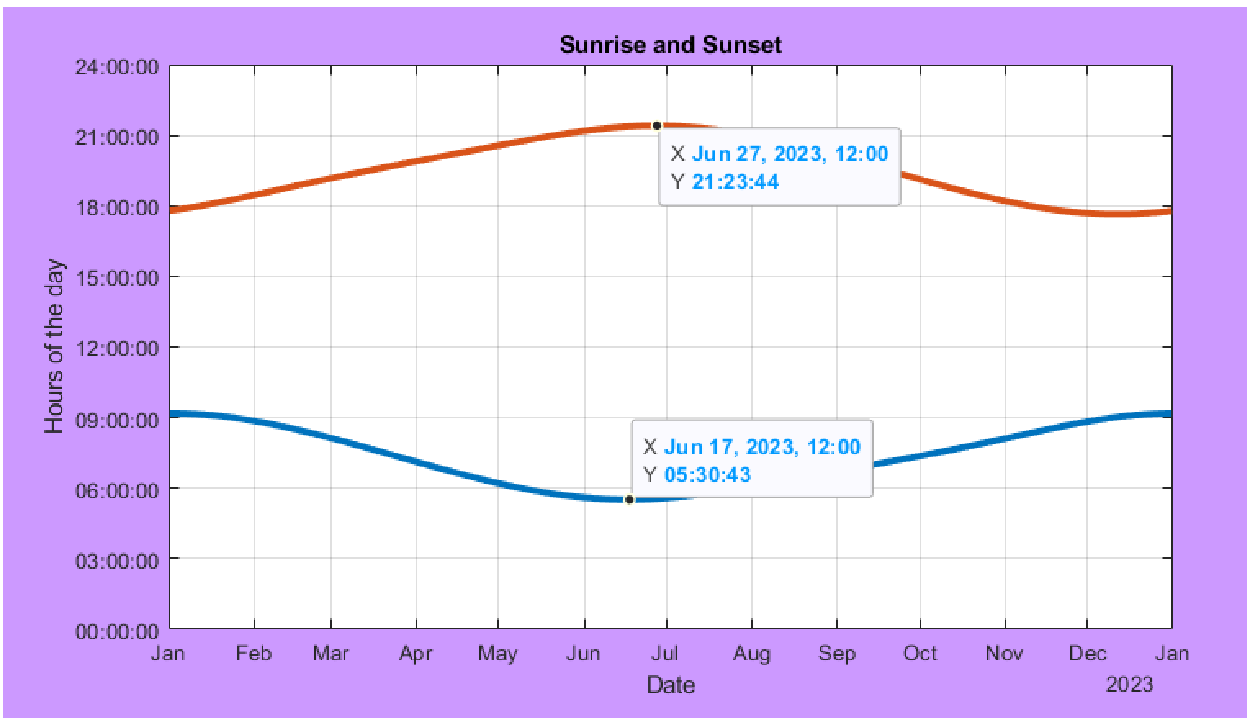
$$sunrise = 720 - 4*(longitude + ha) - eqtime$$

It looks like a big headache, but they are most definitely correct and they provide the right answers.

Experimental part

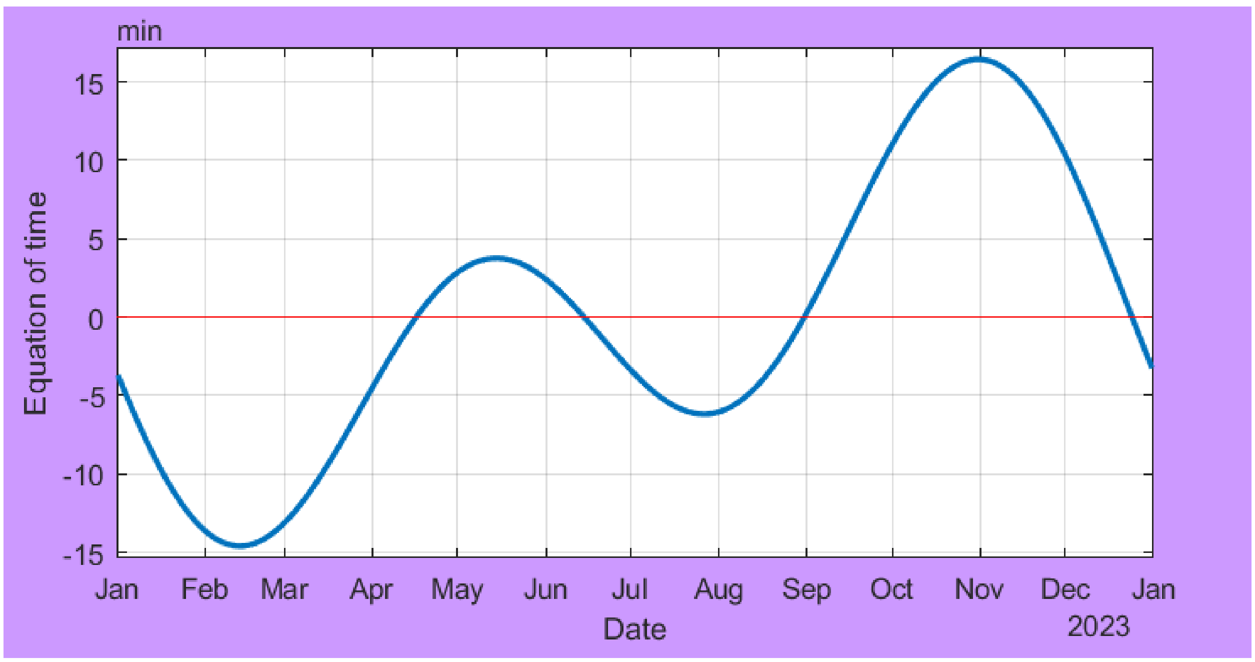
Regarding the experimental part, the default values for the latitude and longitude are set in Romania, more specifically in Cluj-Napoca, 46.77 N, 23.58 E and the time zone is set to GMT+03:00. Usually the time in Romania is GMT+02:00, but between March and October it is GMT+03:00, thus the earliest sunrise and the latest sunset can be more accurate being in the summer.

For these specific values, according to more advanced and precise calculations (not done by me, but found on the internet), the earliest sunrise in 2023 will be at **05:30:24 AM** and the latest sunset will be at **09:04:06 PM** (I couldn't find the accurate times for Cluj-Napoca, these are for Bucharest, but it's not that far off, maybe a couple of minutes maximum).



Now let's see what values does the program generate. For the earliest sunrise, our graph tells us that it will be at **05:30:43 AM**. That's a difference of only **19 seconds**, pretty impressive, very little error. Now, for the latest sunrise, the graph shows that it will be **09:23:44 PM,** the difference from the actual latest sunset being **about 19 minutes**. It's not as accurate here as it is for the sunrise, but as I mentioned above and in the project already, the calculations are not at the highest level possible, they are pretty standard, not too easy but not too complicated, and they will give results with an error of maximum 20 minutes. Calculations at a higher level can be done for more accurate results, but they are harder, longer and more complex. The results for the default values have been below the 20 minute error margin, thus I can say they were a great success, even more because the earliest sunrise was just a couple of seconds off.

This graph shows exactly how sunrises, represented by the blue curve, and sunsets, represented by the orange curve, vary around the year, depending on the season, month, hour. We can also see how the day gets bigger and bigger, precisely starting from the 22nd of December, until 22nd of June.



This graph represents the equation of time values around the year. This graph is not supposed to change when coordinates or time zone are modified. It is the same for every location, every year, because the Earth follows the same path (approximately) around the Sun. For a better understanding of the equation of time, it is defined as the difference between your local apparent solar time, when the Sun crosses your meridian, and local mean solar time, which is the average rate of solar time used in clocks.

Conclusion

The purpose of this research was to provide an intuitive and accurate graph of the sunrise and sunset around the year and I find it to be a great success. Its accuracy is not the best, but it is specified multiple times above that the times can have a maximum error of 20 minutes, due to the fact that the calculations are not at the highest level and better accuracy can be achieved by performing higher level calculations, that take more time and are more complex. However, the smallest error the tests have shown was 19 seconds, while the biggest was 19 minutes, within the error boundary mentioned, thus I can say that the results were a complete success.

For better accuracy regarding the results, that would take a bit more research. It is very much possible, the necessary data definitely is available, whether on the internet or in books, but it’s much more complicated and complex, but that’s what it takes for pinpoint accuracy.

This research of mine is no life changing discovery for sure, but I find it to be very interesting, to have a bit more understanding of the Earth’s rotation around the Sun. It gets even more interesting knowing that people were also calculating these things in the past, as well as the circumference of Earth, its axial tilt, the circumference of the Sun and the distance from Earth to the Sun, more than 2000 years ago, with no advanced technology to their hands, just geography and math.

References

- https://gml.noaa.gov/grad/solcalc/solareqns.PDF
- https://www.mathworks.com/matlabcentral/fileexchange/55509-sunrise-sunset?s_tid=FX_rc1_behav
- https://www.merriam-webster.com/dictionary/equation%20of%20time
- https://public.nrao.edu/ask/why-dont-sunrise-and-sunset-times-change-by-the-same-amount-each-day/
- https://www.farmersalmanac.com/not-all-sunrises-are-created-equal-11611
- https://www.youtube.com/watch?v=bnbLVNRj0II&t=281s&ab_channel=PranavBhounsule

The code

%main.m

```
figure('Name','Sun animation','NumberTitle','off');
y_sun = [linspace(-1,1) linspace(1, -1)];
x_sun = zeros(1,length(y_sun));
for i=1:length(y_sun)
    pause(0.001);
    plot(x_sun(i),y_sun(i),'yo','Markersize',70,'MarkerFaceColor','y');%yo means cir-
cle
    set(gca,'Color','c');
    x = [-2 -2 2 2];
    y = [0 -2 -2 0];
    patch(x,y,[0.4660 0.6740 0.1880])
    axis([-2 2 -2 2]);
end
```

```
latitude=46.77;
longitude=23.58;%coordinates of Cluj Napoca
today = datetime("today","TimeZone","UTC");%gets the current date
first_midnight = dateshift(today,"start","year");%the first midnight of the year
first_midnight_next_year = dateshift(today,"start","year","next");%first midnight of
the next year
middays = (first_midnight+hours(12)):caldays(1):first_midnight_next_year;%sets date
to the midday of every day of the year

%Those dates and times are needed for the calculations next

y = 360*(day(middays,'dayofyear')-81)/365;%this is the formula for the solar declina-
tion
eq_of_time = minutes(9.87*sind(2*y) - 7.53*cosd(y) - 1.5*sind(y));%this is the value
in minutes for the equation of time, plotted below
%figure(1);
%figure('Name','The value of the equation of time','NumberTitle','off');
%plot(middays,eq_of_time, "LineWidth", 2);%plot showing how the value of the equation
of time changes around the year
%hold on;
%xlabel('Date')
%ylabel('Equation of time')
%line(middays,minutes(zeros(size(middays))),'Color','r');


year_fraction = (middays - first_midnight) ./ (first_midnight_next_year - first_mid-
night);%year fraction at midday every day around the year
gamma = 2*3.1415*year_fraction;%transformed in radians
solar_decl = 0.006918-0.399912*cos(gamma)+0.070257*sin(gamma)-
0.006758*cos(2*gamma)+0.000907*sin(2*gamma)-
0.002697*cos(3*gamma)+0.00148*sin(3*gamma);
```

```
hour_angle = acosd((cosd(90.833)./(cosd(latitude).*cos(solar_decl))) - tand(lati-
tude).*tan(solar_decl));%for the special case of sunrise or sunset, the zenith is set
to 90.833°


%sunrise = middays - minutes(4*(longitude + hour_angle)) - eq_of_time;
%sunset  = middays - minutes(4*(longitude - hour_angle)) - eq_of_time;%all 4 of these
calculations are from the PDF with solar equations provided in the documentation
time_zone="+03:00";
sunrise.TimeZone = time_zone;
sunset.TimeZone = time_zone;%we adjust the timezone to Romania's timezone which is
GMT+2, but because of daylight saving time, used ...
%between March and October, we use +3 for accurate earliest sunrise and latest sunset
%figure(2);
%figure('Name','The sunset and sunrise around the year','NumberTitle','off');
%plot(middays,timeofday(sunrise),middays,timeofday(sunset), "LineWidth", 3);%plot
showing how the sunrise and sunset times vary around the year with error of about 20
minutes
%hold off;
%title("Sunrise and Sunset")
%xlabel("Date"); ylabel("Hours of the day")
%ylim(hours([0 24]));

Fig=figure('Name','Project',...
'Units','normalized',...
'NumberTitle','off',...
'Position',[0.1 0.1 0.8 0.8],...
'Color',[204 153 255]./255);
h = uimenu('Label','User Menu');
%creates the menu item that displays the JPG image for LPF
%with accelerator Ctrl+L
uimenu(h,'Label','Geospatial data','Callback','geospatialdata' ,...
 'Accelerator','L');
uimenu(h,'Label','Documentation','Callback','docu' ,...
 'Accelerator','D');
%creates the menu item that closes the window
%with accelerator Ctrl+Q
uimenu(h,'Label','Save','Callback','save');
uimenu(h,'Label','Close','Callback','close',...
 'Separator','on','Accelerator','Q');

% button indicating default values
% if the initial values are changed, the text in this button
% should be modified too!!!
str = "Default values: Latitude=46.77";
str2="!!Small disclaimer: In reality, latitude coordinates go from -90 to 90, but be-
cause these calculations are not 100% accurate and have an error of maximum 20
minutes, please only enter latitudes between -65 and 65, or else the program won't
work :)"
str=str+"                            Longitude=23.58";
str=str+newline+"                             Time zone=+03:00";
str2=str2+newline+"!!Also, when changing coordinates and the time zone, please click
outside of the edit box first for the value to be modified and only then modify an-
other value :)";
%str = str + newline + "A stately pleasure-dome decree"
```

```
uicontrol('Style','text',...
'Units','normalized',...
'Position',[0.55 0.83 0.24 .12],...
'Backgroundcolor',[204 153 255]./255,...
'FontName','Arial',...
'FontWeight','bold',...
'FontSize',16,...
'String',str);
uicontrol('Style','text',...
'Units','normalized',...
'Position',[0.55 0.08 0.29 .1],...
'Backgroundcolor',[204 153 255]./255,...
'FontName','Arial',...
'FontWeight','bold',...
'FontSize',8,...
'String',str2);
%call of the function file creating the graphics objects
%inside the interface


interfata(latitude,longitude,time_zone);
```

//

%interfata.m

```
% interfata.m function file
function interfata(latitude, longitude, time_zone)



today = datetime("today","TimeZone","UTC");%gets the current date
first_midnight = dateshift(today,"start","year");%the first midnight of the year
first_midnight_next_year = dateshift(today,"start","year","next");%first midnight of
the next year
middays = (first_midnight+hours(12)):caldays(1):first_midnight_next_year;%sets date
to the midday of every day of the year
%Those dates and times are needed for the calculations next

y = 360*(day(middays,'dayofyear')-81)/365;%this is the formula for the solar declina-
tion
eq_of_time = minutes(9.87*sind(2*y) - 7.53*cosd(y) - 1.5*sind(y));%this is the value
in minutes for the equation of time, plotted below


year_fraction = (middays - first_midnight) ./ (first_midnight_next_year - first_mid-
night);%year fraction at midday every day around the year
gamma = 2*3.1415*year_fraction;%transformed in radians
solar_decl = 0.006918-0.399912*cos(gamma)+0.070257*sin(gamma)-
0.006758*cos(2*gamma)+0.000907*sin(2*gamma)-
0.002697*cos(3*gamma)+0.00148*sin(3*gamma);
```

```
hour_angle = acosd((cosd(90.833)./(cosd(latitude).*cos(solar_decl))) - tand(lati-
tude).*tan(solar_decl));%for the special case of sunrise or sunset, the zenith is set
to 90.833°


sunrise = middays - minutes(4*(longitude + hour_angle)) - eq_of_time;
sunset  = middays - minutes(4*(longitude - hour_angle)) - eq_of_time;%all 4 of these
calculations are from the PDF with solar equations provided in the documentation
%time_zone='+03:00';
sunrise.TimeZone = time_zone;
sunset.TimeZone = time_zone;%we adjust the timezone to Romania's timezone which is
GMT+2, but because of daylight saving time, used ...
%between March and October, we use +3 for accurate earliest sunrise and latest sunset


% definitions as indicated in the problem3 specs
% defining the cycle of the signal
%T=1/f;
% representation range in time
%t=0:T/100:N*T;
% first function
%x1=A*sin(2*pi*f*t);
% second function
%x2=B*cos(5*2*pi*f*t);
% sum of the two functions sine and cosine
%x3=x1+x2;
% pushbuton CLOSE
uicontrol('Style','pushbutton',...
'Units','normalized',...
'Position',[0.9 0.9 0.08 .05],...
'string','CLOSE',...
'FontName','Arial',...
'FontWeight','bold',...
'FontSize',12,...
'Callback','close');
% Text Button for variable latitude
uicontrol('Style','text',...
'Units','normalized',...
'Position',[0.9 0.65 0.08 .05],...
'Backgroundcolor',[204 153 255]./255,...
'FontSize',13,...
'FontWeight','bold',...
'String','Latitude');
% Edit Button for latitude
uicontrol('Style','edit',...
'Units','normalized',...
'Position',[0.9 0.62 0.08 .05],...
'String',[latitude],...
'Callback',['latitude=','str2num(get(gco,''String''));interfata(latitude, longitude,
time_zone);']);
% Text Button for variable longitude
uicontrol('Style','text',...
'Units','normalized',...
'Position',[0.9 0.55 0.08 .05],...
'backgroundcolor',[204 153 255]./255,...
```

```
'FontWeight','bold',...
'FontSize',13,...
'string','Longitude');
% Edit Button for longitude
uicontrol('Style','edit',...
'Units','normalized',...
'Position',[0.9 0.52 0.08 .05],...
'String',[longitude],...
'Callback',['longitude=','str2num(get(gco,''String''));interfata(latitude, longitude,
time_zone);']);
%Text Button for time_zone
uicontrol('Style','text',...
'Units','normalized',...
'Position',[0.9 0.45 0.08 .05],...
'Backgroundcolor',[204 153 255]./255,...
'FontWeight','bold',...
'FontSize',13,...
'String','Time zone');
% Edit Button for time_zone
uicontrol('Style','edit',...
'Units','normalized',...
'Position',[0.9 0.42 0.08 .05],...
'String',[time_zone],...
'Callback',['time_zone=','get(gco,''String'');interfata(latitude, longitude,
time_zone);']);

uicontrol('Style','pushbutton',...
'Units','normalized',...
'Position',[0.9 0.14 0.08 .05],...
'String','Reset to Initial Values',...
'FontName','Arial',...
'Callback',['latitude=','[44.42] ;interfata(46.77,23.58,"+03:00");'],...
'Callback',['time_zone=','"+03:00" ;interfata(46.77,23.58,"+03:00");'],...
'Callback',['longitude=','[26.02] ;interfata(46.77,23.58,"+03:00");']);
%setting the plot windows
%figure('Name','The value of the equation of time','NumberTitle','off');
subplot('position',[0.1 0.62 0.4 0.35]);
plot(middays,eq_of_time, "LineWidth", 2);%plot showing how the value of the equation
of time changes around the year
grid on;
hold on;
xlabel('Date')
ylabel('Equation of time')
line(middays,minutes(zeros(size(middays))),'Color','r');

%figure('Name','The sunset and sunrise around the year','NumberTitle','off');
subplot('position',[0.1 0.12 0.4 0.4]);
plot(middays,timeofday(sunrise),middays,timeofday(sunset), "LineWidth", 3);%plot
showing how the sunrise and sunset times vary around the year with error of about 20
minutes
grid on;
hold off;
title("Sunrise and Sunset")
xlabel("Date"); ylabel("Hours of the day")
ylim(hours([0 24]));
```

```
end
```

//

%geospatialdata.m

```
function geospatialdata
i=imread('geospatialdata.jpg');
%displays the image
figure('Name', 'Geospatial data of Earth', 'NumberTitle','off');
imshow(i)
title('Geospatial data');
```

//

%docu.m

```
function docu
open('documentation.docx');
```