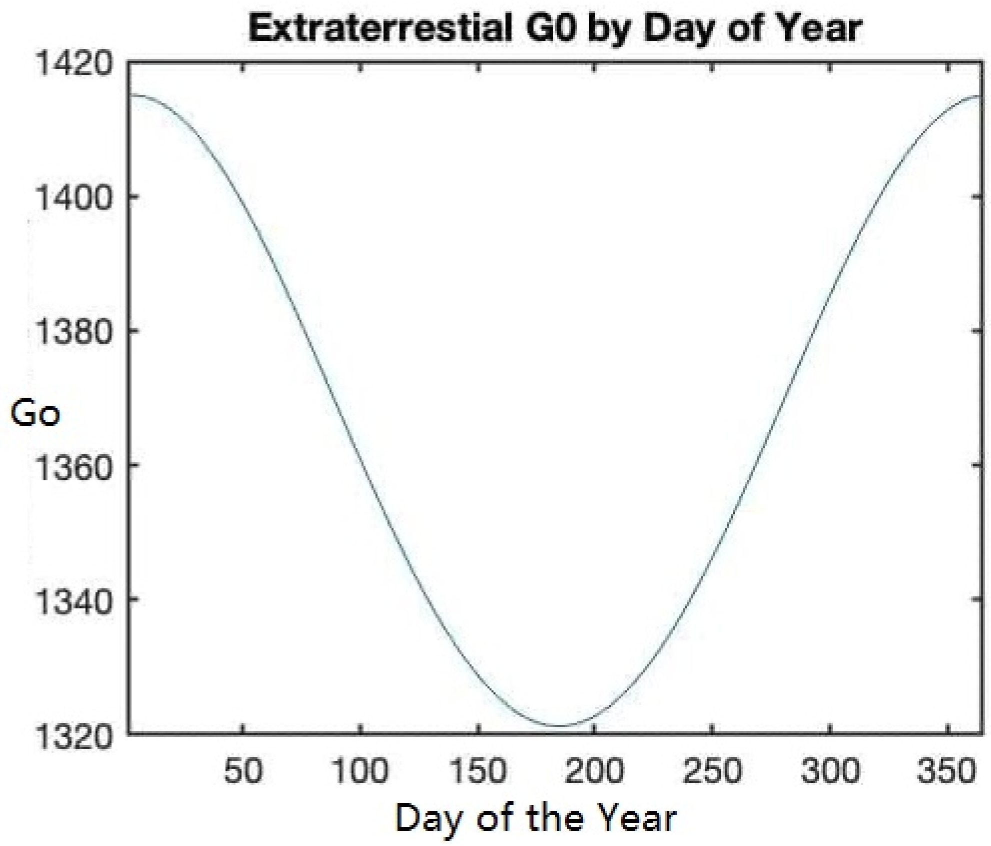
Problem 1:

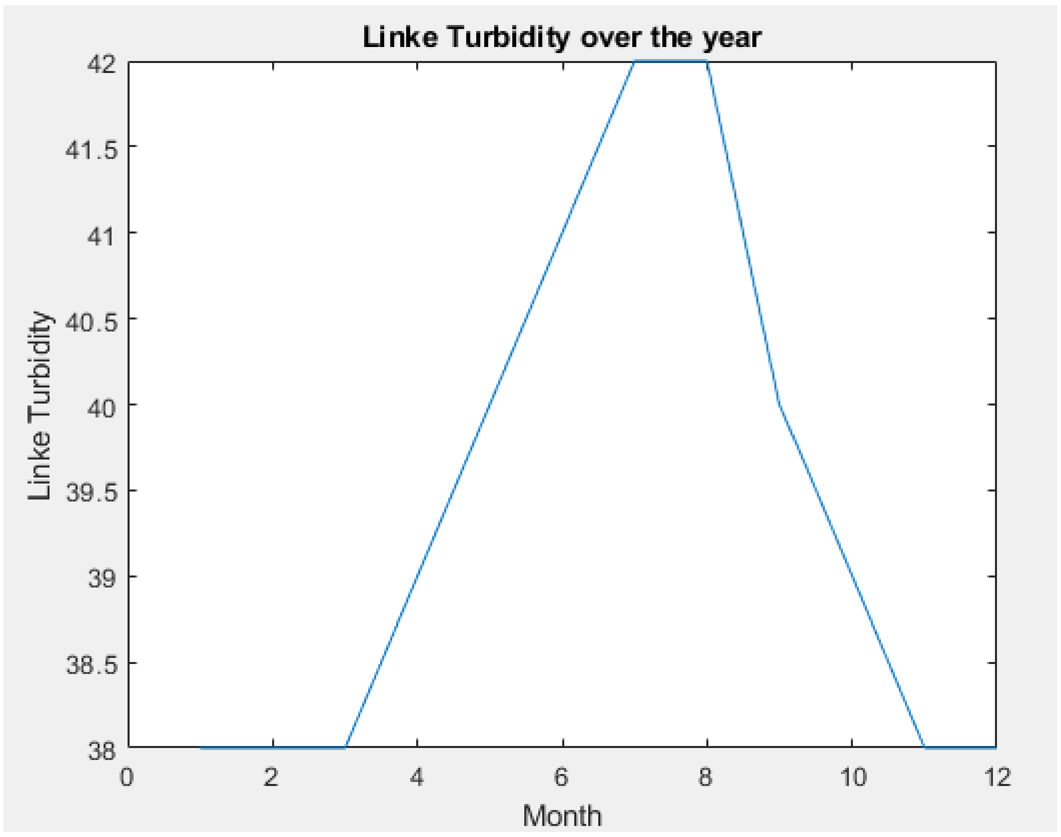
1. Extraterrestrial G0 over the year:



2. Execute the code below:

```
% Load the Linke Turbidity matrix
lk_list = [];
for i = LinkeTurbidity(90-33,180-117,:)
lk_list = [lk_list,i];
end
plot(1:12, lk_list)
xlabel("Month")
ylabel("Linke Turbidity")
title("Linke Turbidity over the year")
% lk_list = [38    38    38    39    40    41    42    42    40    39    38    38]
```

Plot:

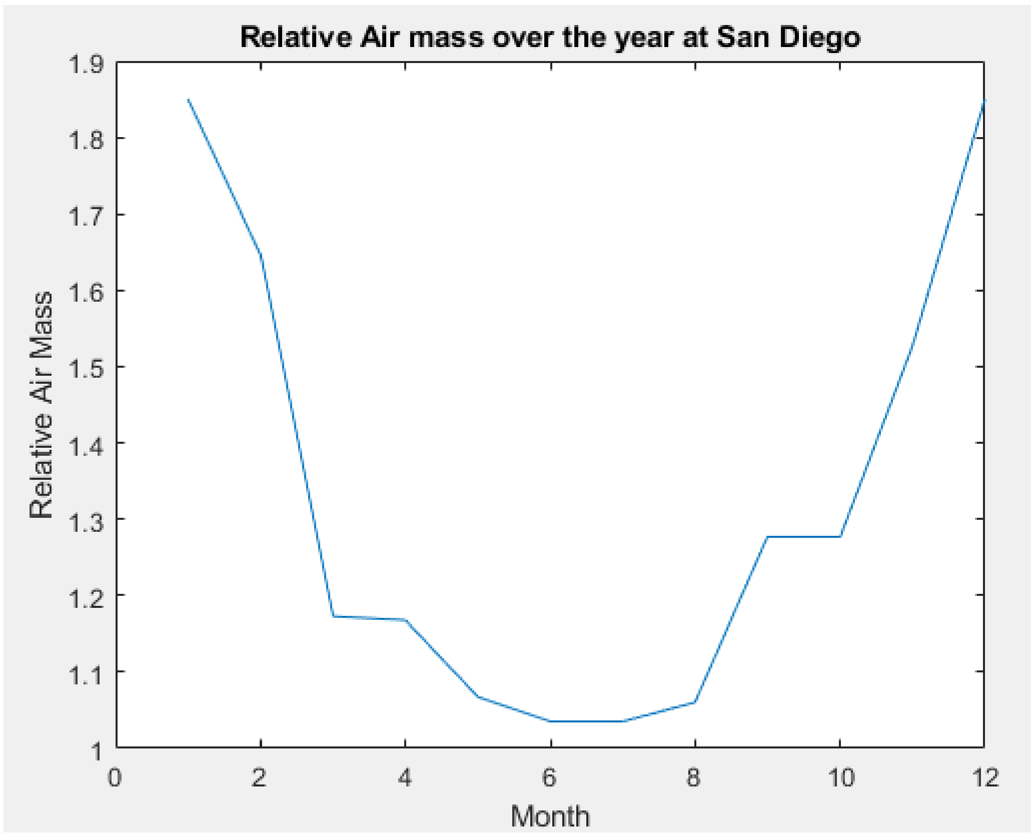


3. Airmass can be calculated using the code:

```
close all

% The following examples show different uses of datetime and DST for
% Golden, Colorado.
lat = 33; % [arc-degrees] latitude
long = -117; % [arc-degrees] longitude
TZ = -8; % [hrs] offset from UTC, during standard time
rot = 0; % [arc-degrees] rotation clockwise from north

% cell array of local date-times during DST
datetimes = {'1/1/2020 12:00 PM', '2/1/2020 12:00 PM', '3/31/2020 12:00
PM',...
    '4/1/2020 12:00 PM', '5/1/2020 12:00 PM', '6/31/2020 12:00 PM',...
    '7/1/2020 12:00 PM', '8/1/2020 12:00 PM', '9/31/2020 12:00 PM',...
    '10/1/2020 12:00 PM', '11/1/2020 12:00 PM', '12/31/2020 12:00 PM'};

DST = true; % local time is daylight savings time
% calculate solar position
a = solarPosition(datetimes,lat,long,TZ,rot,DST); % [arc-degrees]
a = a(:,1);
am = pvl_relativeairmass(a);
plot(1:12, am)
title('Relative Air mass over the year at San Diego')
xlabel('Month')
ylabel('Relative Air Mass')
function [angles,projection] = solarPosition(datetime,latitude,longitude,
...
                                             time_zone,rotation,dst)
%SOLARPOSITION Calculate solar position using most basic algorithm
%   This is the most basic algorithm. It is documented in Seinfeld &
%   Pandis, Duffie & Beckman and Wikipedia.
%
```

```
% [ANGLES,PROJECTION] =
SOLARPOSITION(DATE,TIME,LATITUDE,LONGITUDE,TIME_ZONE)
% returns ZENITH & AZIMUTH for all DATE & TIME pairs at LATITUDE, LONGITUDE.
% ANGLES = [ZENITH,AZIMUTH] and PROJECTION = [PHI_X, PHI_Y]
% PHI_X is projection on x-z plane & PHI_Y is projection on y-z plane.
% DATETIME can be string, vector [YEAR, MONTH, DAY, HOURS, MINUTES,
SECONDS],
%   cellstring or matrix N x [YEAR, MONTH, DAY, HOURS, MINUTES, SECONDS] for
N
%   times.
% LATITUDE [degrees] and LONGITUDE [degrees] are the coordinates of the
site.
% TIME_ZONE [hours] of the site.
% ROTATION [degrees] clockwise rotation of system relative to north.
% DST [logical] flag for daylight savings time, typ. from March to November
%   in the northern hemisphere.
%
% References:
% http://en.wikipedia.org/wiki/Solar_azimuth_angle
% http://en.wikipedia.org/wiki/Solar_elevation_angle
%
% Mark A. Mikofski
% Copyright (c) 2013
%
%% datetime
if iscellstr(datetime) || ~isvector(datetime)
    datetime = datenum(datetime); % [days] dates & times
else
    datetime = datetime(:); % convert datenums to row
end
date = floor(datetime); % [days]
[year,~,~] = datevec(date);
time = datetime - date; % [days]
%% constants
toRadians = @(x)x*pi/180; % convert degrees to radians
toDegrees = @(x)x*180/pi; % convert radians to degrees
%% Equation of time
d_n = mod(date-datenum(year,1,1)+1,365); % day number
B = 2*pi*(d_n-81)/365; % ET parameter
ET = 9.87*sin(2*B)-7.53*cos(B)-1.5*sin(B); % [minutes] equation of time
% approximate solar time
solarTime = ((time*24-double(dst))*60+4*(longitude-time_zone*15)+ET)/60/24;
latitude_rad = toRadians(latitude); % [radians] latitude
rotation_rad = toRadians(rotation); % [radians] field rotation
t_h = (solarTime*24-12)*15; % [degrees] hour angle
t_h_rad = toRadians(t_h); % [radians]
delta = -23.45 * cos(2*pi*(d_n+10)/365); % [degrees] declination
delta_rad = toRadians(delta); % [radians]
theta_rad = acos(sin(latitude_rad)*sin(delta_rad)+ ...
    cos(latitude_rad)*cos(delta_rad).*cos(t_h_rad)); % [radians] zenith
theta = toDegrees(theta_rad); % [degrees] zenith
elevation = 90 - theta; % elevation
day = elevation>0; % day or night?
cos_phi = (cos(theta_rad)*sin(latitude_rad)- ...
    sin(delta_rad))./(sin(theta_rad)*cos(latitude_rad)); % cosine(azimuth)
% azimuth [0, 180], absolute value measured from due south, so east = west =
90,
% south = 0, north = 180
```

```
phi_south = acos(min(1,max(-1,cos_phi)));
% azimuth [0, 360], measured clockwise from due north, so east = 90,
% south = 180, and west = 270 degrees
phi_rad = NaN(size(phi_south)); % night azimuth is NaN
% shift from ATAN to ATAN2, IE: use domain from 0 to 360 degrees instead of
% from -180 to 180
phi_rad(day) = pi + sign(t_h(day)).*phi_south(day); % Shift domain to 0-360
deg
% projection of sun angle on x-z plane, measured from z-direction (up)
phi_x = toDegrees(atan2(sin(phi_rad-rotation_rad).*sin(theta_rad), ...
    cos(theta_rad))); % [degrees]
% projection of sun angle on y-z plane, measured from z-direction (up)
phi_y = toDegrees(atan2(cos(phi_rad-rotation_rad).*sin(theta_rad), ...
    cos(theta_rad))); % [degrees]
phi = toDegrees(phi_rad); % [degrees] azimuth
angles = [theta, phi]; % [degrees] zenith, azimuth
projection = [phi_x,phi_y]; % [degrees] x-z plane, y-z plane
end
```



4. Clear sky GHI can be calculated using the code:

```
%% Add pvlib to path
addpath(genpath('pvlib'));
%% Setup location and time to analyze
today_time=datetime(2019,1:12,7,11,0,0);
Time = pvl_maketimestruct(datenum(today_time),-8);
Location = pvl_makelocationstruct(33,-117,0);
%% How can we get when is solar noon?
[SunAz, SunEl, AppSunEl, SolarTime] = pvl_ephemeris(Time,Location);
[~,isolnoon]=min(abs(SolarTime-12)); % find when solartime is closer to 12
tsolnoon=today_time(isolnoon);
Time = pvl_maketimestruct(datenum(tsolnoon),-8); %redefine time to be solar
noon
```

```
%% Get clear sky GHI using Ineichen-Perez clear sky model
[ClearSkyGHI_1, ClearSkyDNI_1, ClearSkyDIF_1]=
pvl_clearsky_ineichen(Time,Location,1);
% [ClearSkyGHI_7, ClearSkyDNI_7, ClearSkyDIF_7]=
pvl_clearsky_ineichen(Time,Location,7);
% fprintf(['GHI for TL=1 is ',num2str(ClearSkyGHI_1),',
kt=',num2str(840/ClearSkyGHI_1),'\n'])
% fprintf(['GHI for TL=7 is ',num2str(ClearSkyGHI_7),',
kt=',num2str(840/ClearSkyGHI_7),'\n'])
% %How should DNI and DHI change with TL?
% fprintf(['DNI for TL=1 is ',num2str(ClearSkyDNI_1),'\n'])
% fprintf(['DNI for TL=7 is ',num2str(ClearSkyDNI_7),'\n'])
% fprintf(['DIF for TL=1 is ',num2str(ClearSkyDIF_1),'\n'])
% fprintf(['DIF for TL=7 is ',num2str(ClearSkyDIF_7),'\n'])
ClearSkyGHI_1
```

GHI tends to increase with respect to increasing altitude and it tends to decrease with increasing T_l or airmass

5. Actual GHI

Problem 2

1. Using the code:

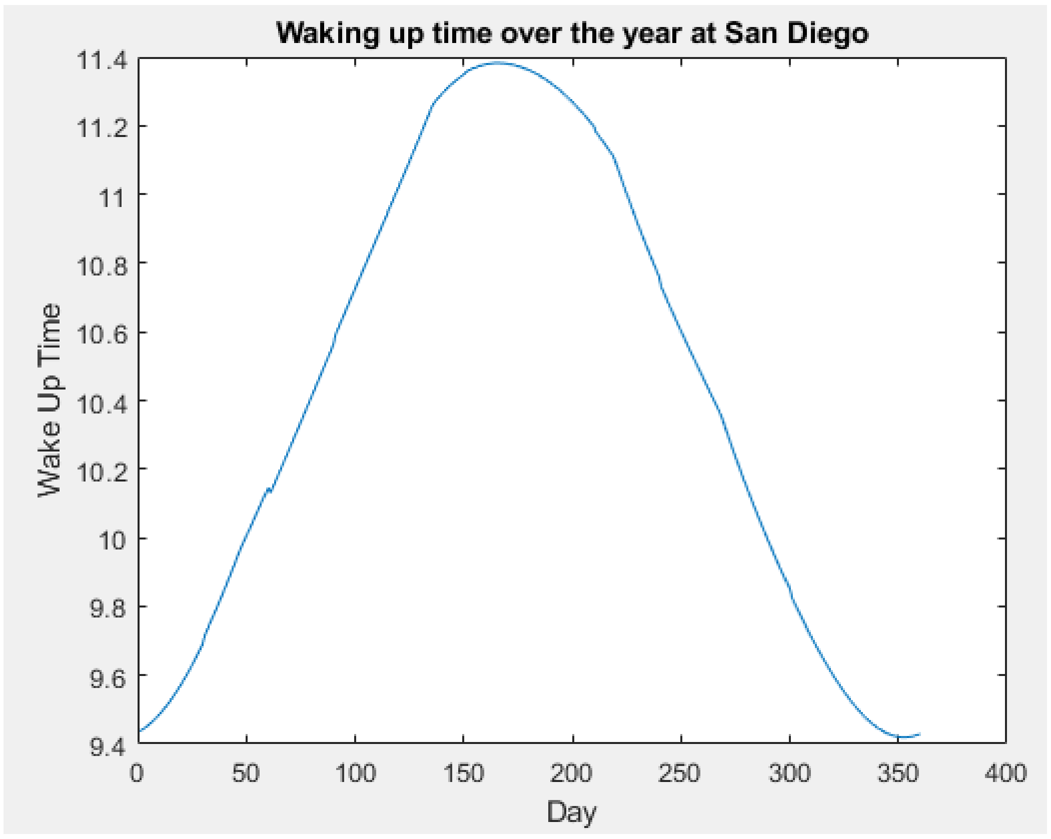
```
clc; clear;
Y = 12 * cosd(10);
x = 180 - atand(Y/15);
Phi = 33;
today_time = datetime(2019,2,1,0:23,0,0);
Time = pvl_maketimestruct(datenum(today_time), -8);
Location = pvl_makelocationstruct(Phi, -117);
[SunAz, SunEl, AppSunEl, SolarTime] = pvl_ephemeris(Time, Location);
WSt = interp1(SunAz, SolarTime, x);


% WSt = 9.7177
% Sun hits window at 9:43 a.m
```

The UCSD student wakes up at 9:43 A.M.

2. Using the code:

```
clc; clear;
Y = 12 * cosd(10);
x = 180 - atand(Y/15);
Phi = 33;

WSt_mat = [];
for d = 1:30
    WSt_list = [];
    for i = 1:12
        today_time = datetime(2019,i,d,0:23,0,0);
        Time = pvl_maketimestruct(datenum(today_time), -8);
        Location = pvl_makelocationstruct(Phi, -117);
        [SunAz, SunEl, AppSunEl, SolarTime] = pvl_ephemeris(Time, Location);
        wSt = interp1(SunAz, SolarTime, x);
        Wst_list = [Wst_list, wSt];
    end
    Wst_mat = [Wst_mat; Wst_list];
end

tlist = [];
for i = 1:12
   tlist = [tlist; Wst_mat(:,i)];
end

plot(1:360, tlist)
title('Waking up time over the year at San Diego')
xlabel('Day')
ylabel('Wake Up Time')
```

Plot:



As the plot suggests, he tends to wake up later during the summer but later during the winter.