HBRP PUBLICATION

Research and Applications of Thermal Engineering
Volume 7 Issue 1

Modelling of Solar Radiation using Python

[1]Adarsh Deo, [2]Arnav Raj Joshi, [3*]Pankaj Dumka, [4]Dhananjay R. Mishra
Department of Mechanical Engineering, Jaypee University of Engineering and Technology, Guna, Madhya Pradesh, India

***Corresponding Author**
E-Mail Id: p.dumka.ipec@gmail.com

ABSTRACT

This short communication describes how a Python module has been developed to understand solar radiation. Through the use of the python module, functions have been developed for zenith angle, hour angle, angle, solar declination angle and solar intensity of extraterrestrial radiation. To ensure that the developed modules were accurate, four problems were selected. The developed codes were tested on these four problems. Correspondingly, the result has shown that the functions generated have helped in a better interpretation and understanding of solar geometry and sun-earth angles.

Keywords: *Solar radiation, python programming, Sun earth angles*

NOMENCLATURE

δ Solar declination angle
ω Hour angle
θ_z Zenith angle
I_{SC} Solar constant
I_{ON} Intensity of extraterrestrial radiation
φ Latitude

INTRODUCTION

Solar radiation, also known as sunlight, emanates from the sun. Numerous technologies exist to harness this radiation, transforming it into practical forms of energy like heat and electricity. Nevertheless, the viability and economic viability of such technologies depend significantly on the solar energy resources accessible at a particular location [1].

Solar technologies transform sunlight into electrical energy through either a photovoltaic panel or by means of mirrors to concentrate solar radiation [2]. The sun emits electromagnetic radiation across various wavelengths, including infrared. This spectrum allows effective transfer of thermal energy to bodies capable of absorbing it. Materials known as 'black bodies' efficiently absorb thermal electromagnetic energy, as the color black absorbs all wavelengths visible to the human eye [3–6].

Solar radiation varies at different locations on Earth's surface due to several factors:

- Geographic location
- Local landscape
- Local weather conditions
- Time of day

The level of solar radiation absorbed hinges on the angle of incidence between the sun and the Earth's surface, which varies from 0° (near the horizon) to 90° (directly above). At a perpendicular stance, solar rays directly target the Earth's surface, maximizing energy absorption. However, as the angle deviates, rays traverse a longer atmospheric path, causing heightened scattering and dispersion of radiation.

Modelling solar radiation and at the same time graphing it is difficult using pen and paper. Here comes the role of Python programming as a programming language that can perform numerical computations and graphing very easily [7–13]. Python's true power resides in its modules such as Numpy [14–17], SymPy [18–21], Matplotlib [22,23], etc. In this article the strength of Python programming has been used to model solar radiation.

SUN-EARTH ANGLES

Since the radiation beam hitting the Earth's surface can be oriented in any direction, different angles between the sun and Earth are necessary to interpret the solar energy received.

Solar Declination Angle (δ)

The declination angle delineates the angle between the sun's rays and the equatorial plane. This variation primarily arises from the Earth's rotation around its axis. Its peak reaches 23.45° on December 21st, while its nadir is -23.45° on June 21st, a calculation derived from the following relation. It is represented in Eqn. (1)

$$\delta = 23.45 \times sin\ (284 + n) \times 360 \qquad (1)$$

where, 'n' is the n^{th} of the year

Azimuth Angle

The azimuth angle shows the compass direction from which sunlight emanates. It's essential to note that in the northern hemisphere, the sun constantly look as if directly south at solar noon, while in the southern hemisphere, it look as if straight north. Furthermore, sunrise and sunset are associated with azimuth angles of 90° and 270°, correspondingly, which align with the equinoxes.

Hour Angle (ω)

The position of a meridian on Earth is established by the angle needed for the Earth to rotate and align it with its current location. When the angle becomes positive, it will increase, as it continues to decrease from sunrise to noon. At noon, it will become zero and then begin to increase after that time. Eqn. (2) shows the formula to evaluate it.

$$\omega = (ST - 12) \times 15 \qquad (2)$$

where, ST is the standard time.

Zenith Angle (θ_z)

An angle formed by the sun's rays angled at its vertical direction is termed the solar zenith angle. The altitude angle is also known as the elevation angle. It serves as a complement to the solar altitude or solar elevation. The expression used for obtaining (θ_z) is shown in Eqn. (3)

$$Cos(\theta_z) = sin\varphi \times sin\delta + cos\varphi \times cos\delta \times cos\omega \qquad (3)$$

SOLAR INTENSITY OF EXTRATERRESTRIAL RADIATION

A unit area on an extraterrestrial surface can accept a consistent quantity of radiant energy per second when positioned perpendicular to the sun's rays at an average Earth-Sun distance. Due to Earth's orbit being elliptical with the sun at one focus, rather than circular, the extraterrestrial radiation experiences fluctuations. These variations in the intensity of extraterrestrial radiation on the nth day of the year, measured on a plane perpendicular to the radiation, are depicted by Eqn. (4).

$$I_{ON} = I_{SC}\,(1 + 0.033 \times cos\,(\frac{360\text{ n}}{365})) \qquad (4)$$

IMPLEMENTATIONS OF SOLAR RADIATION IN PYTHON [Table 1]

Table 1: *Functions developed in python for evaluating sun-earth angles.*

<table>
<tr><th>EXPLAINATION</th><th>CODE</th></tr>
<tr><td>Libraries imported</td><td><pre>from math import*
from numpy import*
from pylab import*</pre></td></tr>
<tr><td>Function for Solar declination angle</td><td><pre> # Function- s_dec
Input- day (d), month (m), year (y)
Output- Solar declination angle

def s_dec(d,m,y):
    arr = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

    if(((y%400==0) or ((y%100!=0) and (y%4==0)))) :
        arr[1] = 29
    else :
        arr[1] = 28; n_days=int(sum(arr[:m-1]) + d)

    x=array(list(range(1,367)))
    δ=zeros(n_days)
    for i in range (0,n_days):
        δ[i]= 23.45*sin((284+x[i])*(360/365)*(pi/180))
        print(x[:n_days])
        plot(x[:n_days],δ,'r-')
        xlabel('NUMBERS OF DAY')
        ylabel('SOLAR DECLINATION ANGLE')
        show()
    return δ</pre></td></tr>
<tr><td>Function for Hour angle</td><td><pre> # Function- h_ang
Input- Sunshine hours (6am to 6pm)
Output- Hour angle

def h_ang(z,t):
    hour=array(range(z,t+1))
    h=zeros(len(hour))
    for i in range(0,len(hour)):
        h[i]=(hour[i]-12)*15
    ylim(-110,110)
    plot(hour,h,'g-o')
    xlabel('SOLAR TIME')
    ylabel('HOUR ANGLE')
    show()
    return h</pre></td></tr>
</table>

<table>
<tr>
<td>Function for Solar intensity of extraterrestrial radiation</td>
<td><pre><code># Function- s_int
Input- day (d), month (m), year (y), Solar constant (1367 W/m^2)
Output- Intensity of extraterrestrial radiation

def s_int(d,m,y,i_sc):
    arr = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

    if(((y%400==0) or ((y%100!=0) and (y%4==0)))) :
        arr[1] = 29
    else :
        arr[1] = 28;[]
    n_days=int(sum(arr[:m-1]) + d)

    x=array(list(range(1,367)))
    i_on=zeros(n_days)
    for i in range (0,n_days):
        i_on[i]= i_sc*(1 + 0.0033*cos((x[i]*360)/365))
    print('n=',n_days)
    plot(x[:n_days],i_on,'g-')
    xlabel('NUMBERS OF DAY')
    ylabel('INTENSITY OF EXTRATERRESTRIAL RADIATION')
    show()
    return i_on</code></pre></td>
</tr>
</table>

Example 1: Plot the variation of solar declination angle (δ) with n^{th} day of the year.

Python Approach

<table>
<tr><th>PYTHON CODE</th><th>PROGRAM OUTPUT</th></tr>
<tr>
<td><pre><code>#INPUT DATA

d=31
m=12
y=2022

#CALLING THE FUNCTION

s_dec(d,m,y)</code></pre></td>
<td>Solar declination angles till ‘nth’ day of the year at an interval of 12 days:

array([-23.01163673, -21.43630132,
-18.79191752, -15.21036321,
-10.87025385, -5.98803477,
-0.80718679, 4.41391635,
9.41489335, 13.94634081,
17.78227121, 20.73138311,
22.64660154, 23.43241276,
23.04962764, 21.51733603,
18.91195474, 15.36341658,
11.04869045, 6.1829558 ,
1.00887136, -4.21552644,
-9.22969199, -13.78356417,
-17.65003711, -20.63628618,
-22.59338436, -23.42372933,
-23.085911])</td>
</tr>
</table>

Graph

The graph below (Fig. 1) illustrates the fluctuation of the solar declination angle over the course of 365 days. It reaches its minimum value of -23.449 degrees on December 21st and its maximum value of 23.449 degrees on June 21st.

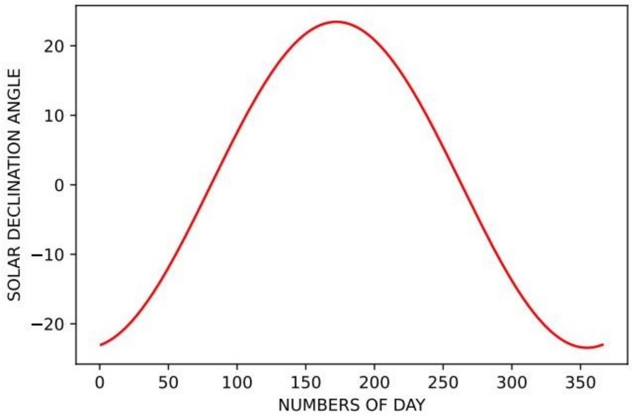

Fig. 1: *Variation of solar declination angle with days.*

Example 2: Plot the variation of hour angle (ω) with solar time for all sunshine hours from 6 am to 6 pm.

Python Approach

PYTHON CODE	PROGRAM OUTPUT
#INPUT DATA z=6 t=18 *#24HR FORMAT* *#CALLING FUNCTION* h_ang(z,t)	[-90., -75., -60., -45., -30., -15., 0. 15., 30., 45., 60., 75., 90.]

Graph

The presented figure (Fig. 2) illustrates the fluctuation of the hour angle with solar time (ST), ranging from -90 to 90 degrees, covering the period from 6 am to 6 pm.

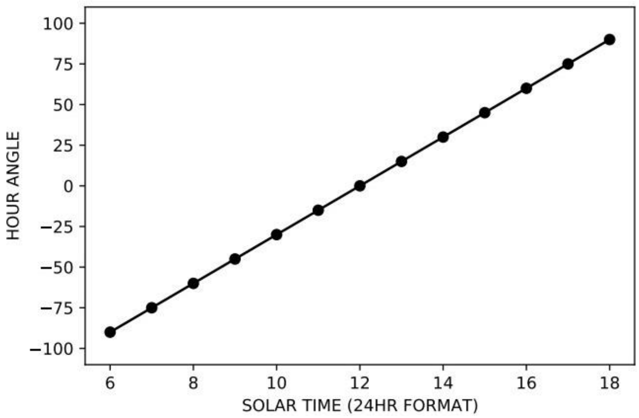

Fig. 2: *Variation of Hour angle with solar time.*

Example 3: Plot the zenith angle (θ_z) variation against the hour angle for New Delhi ($\phi =$ 28.45°) on October 25, 2022.

Python Approach

PYTHON CODE	PROFRAM OUTPUT

<pre>#INPUT DATA
 	
d=25
m=10
y=2022
 	
z=6
t=18
 	
φ =28.58*(pi/180)
 	
#CALLING HOUR ANGLE FUNCTION
 	
ω=h_ang(z,t)*(pi/180)
 	
#CALLING SOLAR DECLINATION ANGLE FUNCTION
 	
δ =s_de(d,m,y)*(pi/180)
 	
#ZENITH ANGLE
 	
θ_z=zeros(len( ))
for i in range(0,len( )):
θ_z[i]= acos(cos(φ)*cos(δ)*cos(ω[i])+sin(φ)*sin(δ))
 print(' θ_z=', θ_z*(180/pi))
 	
#PLOTTING GRAPH OF VARIATION OF ZENITH ANGLE WITH HOUR ANGLE
 	
plot(ω*(180/pi), θ_z*(180/pi),'b-o')
xlabel('HOUR ANGLE')
ylabel('ZENITH ANGLE')
show()</pre>
	θ_z= [94.40060347 81.51156007 69.1047287 57.57622602 47.62929677 40.49024901 37.80969199 40.49024901 47.62929677 57.57622602 69.1047287 81.51156007 94.40060347]

Below is the obtained plot (Fig. 3) which illustrates the variation of the zenith angle with the hour angle for October 25, 2022. Notably, the zenith angle reaches its minimum value of 37.09 degrees at an hour angle of 0 degrees, while it attains its maximum values of 94.40 degrees at hour angles of -90 and 90 degrees.

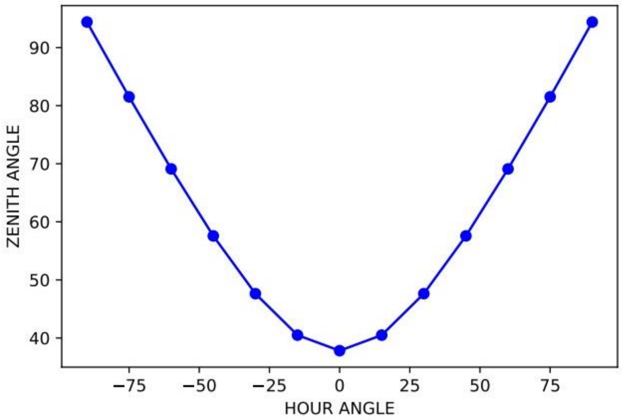

Fig. 3: *Variation of zenith angle with hour angle.*

Example 4

Plot the extraterrestrial radiation intensity variation throughout the year.

Python Approach

PYTHON CODE	PROGRAM OUTPUT
#INPUT DATA d=31 m=12 y=2022 i_sc=1367 *#CALLING THE FUNCTION* s_int(d,m,y,i_sc)	Intensity of extraterrestrial radiation till n=365 days of the year at an interval of 12 days: array([1369.48912687, 1368.45729907, 1367.33082006, 1366.18285437, 1365.08796209, 1364.11725619, 1363.33378379, 1362.78843123, 1362.51661902, 1362.53600129, 1362.84531918, 1363.42448255, 1364.2358749 , 1365.22679649, 1366.33288721, 1367.48230674, 1368.60040057, 1369.61454878, 1370.45888269, 1371.07856302, 1371.4333417 , 1371.50017593, 1371.27472484, 1370.77163144, 1370.0235715 , 1369.07913135, 1367.99965212, 1366.85524574, 1365.72024112])

The graph (Fig. 4) below shows the variation of intensity of extraterrestrial radiation with the nth day of the year (‘n’ varies from 1 to 365).

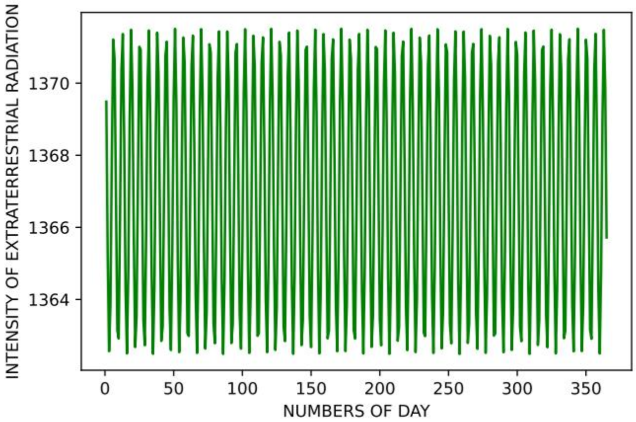

Fig. 4: *Variation of intensity of extraterrestrial radiation with days.*

CONCLUSION

In this research article, the graphical evaluation of solar-earth angles was successfully performed using Python programming. For this study, four problems were taken, and functions were developed accordingly with the help of different modules. The results obtained were accurate and can help beginners to increase their basic understanding and interpretation of solar geometry.

REFERENCES

1. Mahian, O., Kianifar, A., Jumpholkul, C., Thiangtham, P., Wongwises, S., & Srisomba, R. (2015). Solar distillation practice for water desalination systems. *Journal of Thermal Engineering*, *1*(4), 287-288.
2. Katekar, V. P., & Deshmukh, S. S. (2020). A review on research trends in solar still designs for domestic and industrial applications. *Journal of cleaner production*, *257*, 120544.
3. Dumka, P., & Mishra, D. R. (2020). Performance evaluation of single slope solar still augmented with the ultrasonic fogger. *Energy*, *190*, 116398.
4. Dumka, P., Jain, A., & Mishra, D. R. (2020). Energy, exergy, and economic analysis of single slope conventional solar still augmented with an ultrasonic fogger and a cotton cloth. *Journal of Energy Storage*, *30*, 101541.
5. Chauhan, R., Dumka, P., & Mishra, D. R. (2022). Modelling conventional and solar earth still by using the LM algorithm-based artificial neural network. *International Journal of Ambient Energy*, *43*(1), 1389-1396.
6. Ahmad, M.J., Tiwari, G.N. (2010). Solar radiation models-review. *Int. J. ENERGY Environ, 1*, 2076–2909.
7. Pawar, P. S., Mishra, D. R., & Dumka, P. (2022). Solving first order ordinary differential equations using least square method: a comparative study. *Int J Innov Sci Res Technol*, *7*(3), 857-864.
8. Deo, A., Joshi, A. R., Parashar, A., Mishra, D. R., & Dumka, P. (2022). Analysing one dimensional tapered pin-fin using finite difference. *Res Appl Therm Eng*, *5*(1), 1-6.
9. Rocklin, M. (2012). Uncertainty modeling with SymPy stats. In *Proc 11th Python Sci Conf* (pp. 51-5).
10. Pawar, P. S., Mishra, D. R., Dumka, P., & Pradesh, M. (2022). Obtaining exact solutions of viscoincompressible parallel flows using python. *Int J Eng Appl Sci Technol*, *6*(11), 213-217.
11. Huei, Y. C. (2014, December). Benefits and introduction to python programming for freshmore students using inexpensive robots. In *2014 IEEE International Conference on Teaching, Assessment and Learning*

for Engineering (TALE) (pp. 12-17). IEEE.
12. Joshi, A. R., Deo, A., Parashar, A., Mishra, D. R., & Dumka, P. (2023). Modelling Steam Power Cycle using Python.
13. Varsha, M., Yashashree, S., Ramdas, D. K., & Alex, S. A. (2019). A Review of Existing Approaches to Increase the Computational Speed of the Python. *International Journal of Research in Engineering, Science and Management*, *2*(4).
14. Hoyer, S., & Hamman, J. (2017). xarray: ND labeled arrays and datasets in Python. *Journal of Open Research Software*, *5*(1).
15. Johansson, R., Johansson, R., & John, S. (2019). *Numerical python* (Vol. 1). New York: Apress.
16. Dumka, P., Dumka, R., & Mishra, D. R. (2022). *Numerical Methods using Python (For scientists and Engineers)*. Blue Rose Publishers.
17. Dumka, P., Rana, K., Tomar, S. P. S., Pawar, P. S., & Mishra, D. R. (2022). Modelling air standard thermodynamic cycles using python. *Advances in Engineering Software*, *172*, 103186.
18. Dumka, P., Chauhan, R., Singh, A., Singh, G., & Mishra, D. (2022). Implementation of Buckingham's Pi theorem using Python. *Advances in Engineering Software*, *173*, 103232.
19. Rocklin, M., & Terrel, A. R. (2012). Symbolic statistics with SymPy. *Computing in Science & Engineering*, *14*(3), 88-93.
20. Cywiak, M., & Cywiak, D. (2021). Two-Dimensional Fourier Transform. In *Multi-Platform Graphics Programming with Kivy: Basic Analytical Programming for 2D, 3D, and Stereoscopic Design* (pp. 313-346). Berkeley, CA: Apress.
21. Meurer, A., Smith, C. P., Paprocki, M., Čertík, O., Kirpichev, S. B., Rocklin, M., ... & Scopatz, A. (2017). SymPy: symbolic computing in Python. *PeerJ Computer Science*, *3*, e103.
22. Kanagachidambaresan, G. R., & Manohar Vinoothna, G. (2021). Visualizations. *Programming with TensorFlow: Solution for Edge Computing Applications*, 15-21.
23. Bisong, E. (2019). *Building machine learning and deep learning models on Google cloud platform* (pp. 59-64). Berkeley, CA: Apress.