1. Reading the data of csv file using numpy and importing the libraries required

In [5]:
```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
from scipy.optimize import curve_fit as cf
Time, Distance, Angle = np.loadtxt(r"Comet_Data_1.csv",delimiter=',
import math
```

2. Calculation of the distance of the comet from stationary earth reference

In [6]:
```
New_Distance = []
for h in range( len(Time)):
    if(abs(Angle[h]) < 1.57079633):
        Distance_square = np.square(Distance[h])
        s = np.sin(Angle[h])
        o = np.cos(Angle[h])
        sine = np.square(s)
        cosine = np.square(o)
        Distance_times_cos = np.multiply(Distance[h], o)
        c = 2*Distance_times_cos
        k = np.multiply(sine, Distance_square)
        a = np.add(k, Distance_square)
        b = np.add(c, cosine)
        d = np.add(a, b)
        f = np.sqrt(d)
        New_Distance.append(f)
    elif(abs(Angle[h]) > 1.57079633):
        w = 1/math.sin(Angle[h])
        v = 1/math.tan(Angle[h])
        t = math.atan((w/Distance[h]) - v)
        l = math.cos(t)
        Distance_square = np.square(Distance[h])
        g = np.sqrt(Distance_square - l**2)
        n = np.sqrt(1 - l**2)
        m = g + n
        New_Distance.append(f)
New_Distance = np.array(New_Distance)
print(New_Distance) #this is the final distance of the comet to sun
```

```
[ 1.97695757  2.37934504  2.91972202  3.55404808  4.15434468  4.59
60192
  4.89507862  5.18935566  5.5788724   5.8628259   5.68386103  5.68
```

```
386103
  5.68386103  5.68386103  5.68386103  5.68386103  8.23161829  8.59
496407
  8.32517492  8.39510703  9.23709205 10.02773301 10.02773301 10.02
773301
 10.02773301 10.02773301 10.02773301 10.02773301 12.12995484 11.53
365254
 10.85545556 11.55181891 13.01364935 13.48604502 13.48604502 13.48
604502
 13.48604502 13.48604502 13.48604502 15.191553   14.62282085 13.22
028747
 13.41695927 15.2243456  16.45535844 16.45535844 16.45535844 16.45
535844
 16.45535844 16.45535844 17.84313974 17.46973275 15.63955622 15.19
876833
 17.05521035 18.96775666 18.96775666 18.96775666 18.96775666 18.96
775666
 18.96775666 18.96775666 20.23536249 18.04193096 16.80170788 18.54
93426
 21.0228849  21.0228849  21.0228849  21.0228849  21.0228849  21.02
28849
 21.0228849  22.68085995 20.35672522 18.45744341 19.770327   22.80
018219
 23.77777443 23.77777443 23.77777443 23.77777443 23.77777443 23.77
777443
 24.79976749 22.65773239 20.07694839 20.87105821 24.17583598 25.86
259701
 25.86259701 25.86259701 25.86259701 25.86259701 25.86259701 26.86
419582
 25.03852145 21.8617517  21.89736376 25.2988623 ]
```

3. Calculation of the conic section plot with its eccentricities

In [7]:
```
j = 2*(np.pi)
Angle_Earth= []
for i in Time:
    Angle_Swept = i*j
    Angle_Earth.append(Angle_Swept)
Angle_Earth = np.array(Angle_Earth)
def conic_Section_Co0ordinates(Angle, Distance, Angle_Earth):
    x = []
    y = []
    for i in range(len(Angle)):
        u = Distance[i] * np.cos(Angle[i])
        v = 1-Distance[i] * np.sin(Angle[i])
        Earths_Rotation_Matrix = [[np.cos(Angle_Earth[i]), -np.sin(
        Comet_Cordinates = [u,v]
        New_Comet_Cordinates = np.matmul(Earths_Rotation_Matrix, Co
        x.append(New_Comet_Cordinates[0])
        y.append(New_Comet_Cordinates[1])
    x = np.array(x)
    y = np.array(y)
    return x,y
```

4. Spacial co-ordinates of the comet

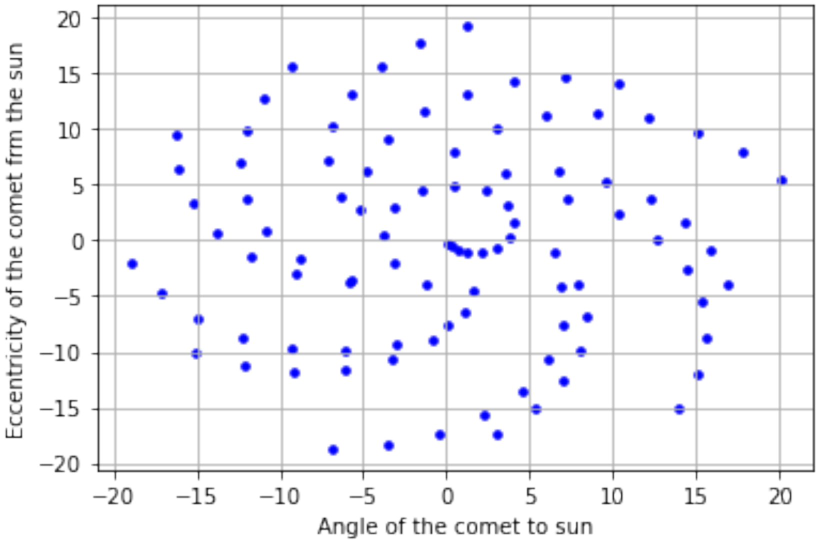
In [17]:
```

nates(Angle, Distance, Angle_Earth):


ge(len(Angle)):
stance[i] * np.cos(Angle[i])
ance[i] * np.sin(Angle[i])
otation_Matrix = [[np.cos(Angle_Earth[i]), -np.sin(Angle_Earth[i])],
rdinates = [u,v]
t_Cordinates = np.matmul(Comet_Cordinates, Earths_Rotation_Matrix)
(New_Comet_Cordinates[0])
(New_Comet_Cordinates[1])
(x)
(y)
```

5. Plot of the conic section curve

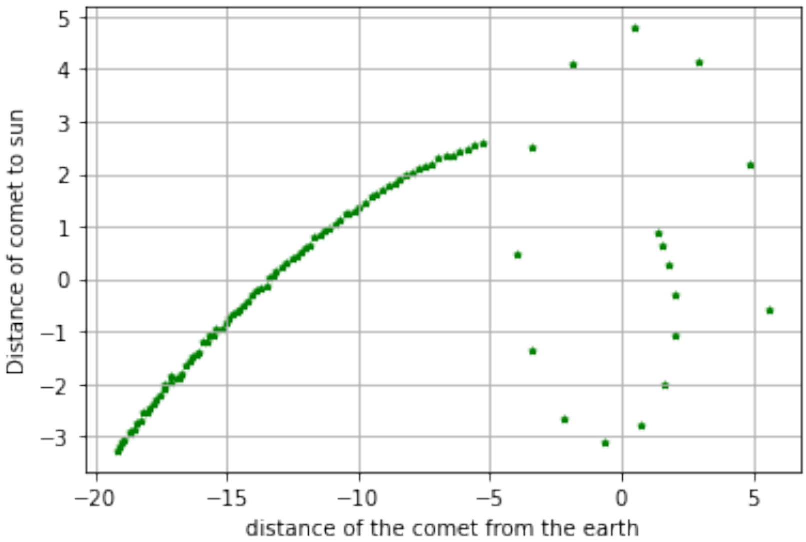
In [9]:
```
x,y = conic_Section_Co0ordinates(Angle, Distance, Angle_Earth)
plt.scatter(x,y,s=50, color='b', marker='.')
plt.ylabel('Eccentricity of the comet frm the sun')
plt.xlabel('Angle of the comet to sun')
plt.grid()
plt.show()
```



In [10]:
```
# SINCE THIS WHIRLPOOL IS A CLOSD FIGURE ORIGINATING FROM ONE POINT
# REPRESENTS THE CONIC SECTIONS THAT CAN BE FORMED ARE EITHER ELLIP
```

6. Plot of the path of the comet with respect to the sun

In [11]:
```
y,x = Space_Coordinates(Angle, Distance, Angle_Earth)
plt.scatter(x,y,s=10, color='g', marker='*')
plt.ylabel('Distance of comet to sun')
plt.xlabel('distance of the comet from the earth')
plt.grid()
plt.show()
```



7. Figuring out the conic section equation of the trajectory of the comet with sun at one foci

In [12]:
```
#curve fitting function using SciPy optimization
#Deterintion of the conic equation
X=[]
Y=[]
for i in range(len(x)):
    X.append([x[i]])
    Y.append([y[i]])
X = np.array(X)
Y = np.array(Y)
A = np.hstack([X**2, X * Y, Y**2, X, Y])
b = np.ones_like(X)
m = np.linalg.lstsq(A,b)[0] #minimizes using least squares
print('The conic is {}x^2 - {}xy + {}y^2 + {}x +{}y - 1 = 0'.format
```

```
The conic is [0.01266896]x^2 - [0.04938894]xy + [0.08877545]y^2 +
[0.09406674]x +[-0.10544793]y - 1 = 0

/var/folders/dd/mm0bjyy925nbd_28cjkrchgr0000gn/T/ipykernel_65968/3
279779048.py:12: FutureWarning: `rcond` parameter will change to t
he default of machine precision times ``max(M, N)`` where M and N
are the input matrix dimensions.
To use the future default and silence this warning we advise to pa
ss `rcond=None`, to keep using the old, explicitly pass `rcond=-1`
.
  m = np.linalg.lstsq(A,b)[0] #minimizes using least squares
```

8. Calculation of the distance of the comet from the sun and sorting out the array to calculate the closest approach

In [15]:
```
y_array = np.array(y)
modulus_y_array = np.abs(y_array)
Sorted_y_array = np.sort(modulus_y_array)
print(Sorted_y_array)
```

```
[0.03880427 0.07546963 0.12243524 0.15827495 0.19371884 0.23392619
 0.23949326 0.26481509 0.29132251 0.30216573 0.30784465 0.39317923
 0.41311087 0.41806002 0.48221788 0.49718878 0.51723439 0.60419907
 0.60452987 0.6099579  0.61110239 0.62320784 0.6325148  0.68161708
 0.75861268 0.78056912 0.83039897 0.85779852 0.86415526 0.90074857
 0.96458552 0.96602671 0.97941835 1.04453064 1.06739373 1.08139036
 1.09316737 1.11932477 1.19406235 1.20603964 1.23183482 1.26321656
 1.29122684 1.363854   1.37672519 1.38369944 1.42814303 1.45406409
 1.50149028 1.57284546 1.59139025 1.62561527 1.64804747 1.69160581
 1.76188761 1.79151692 1.82689863 1.86171574 1.87381865 1.89569307
 1.89670165 1.9689522  1.98511433 2.01111609 2.01444652 2.02582555
 2.08818269 2.10718349 2.14616938 2.18604682 2.18701453 2.23571323
 2.2891358  2.31651841 2.3359515  2.35081145 2.39222978 2.43102436
 2.45575057 2.47981023 2.49204578 2.52326901 2.53166424 2.55599698
 2.57609048 2.66021888 2.68927256 2.76358313 2.793422   2.8698554
 2.9217751  2.96042546 3.06373264 3.10752621 3.13072653 3.20498537
 3.27314122 4.11907506 4.14159001 4.78186969]
```

9. Conclusion

In [16]:
```
print("Answer: The comet follows the trajectory of a {} and it's cl
```

```
Answer: The comet follows the trajectory of a Hyperbola and it's c
losest approach to the sun is 0.038804266924794106Au
```