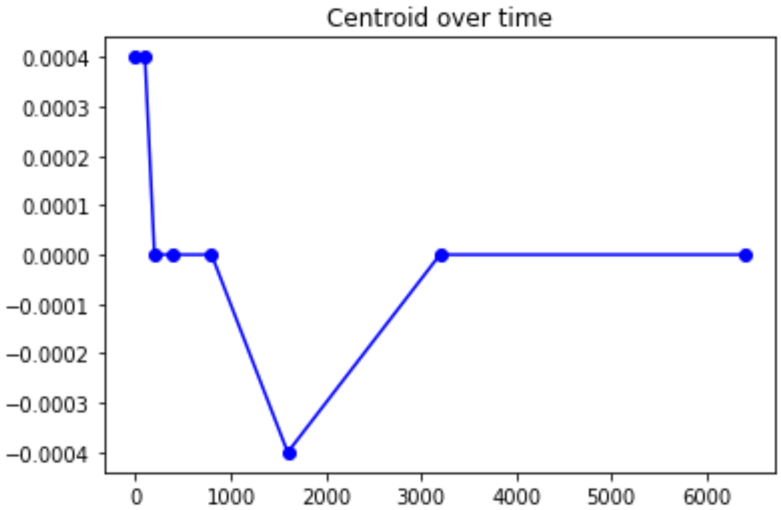
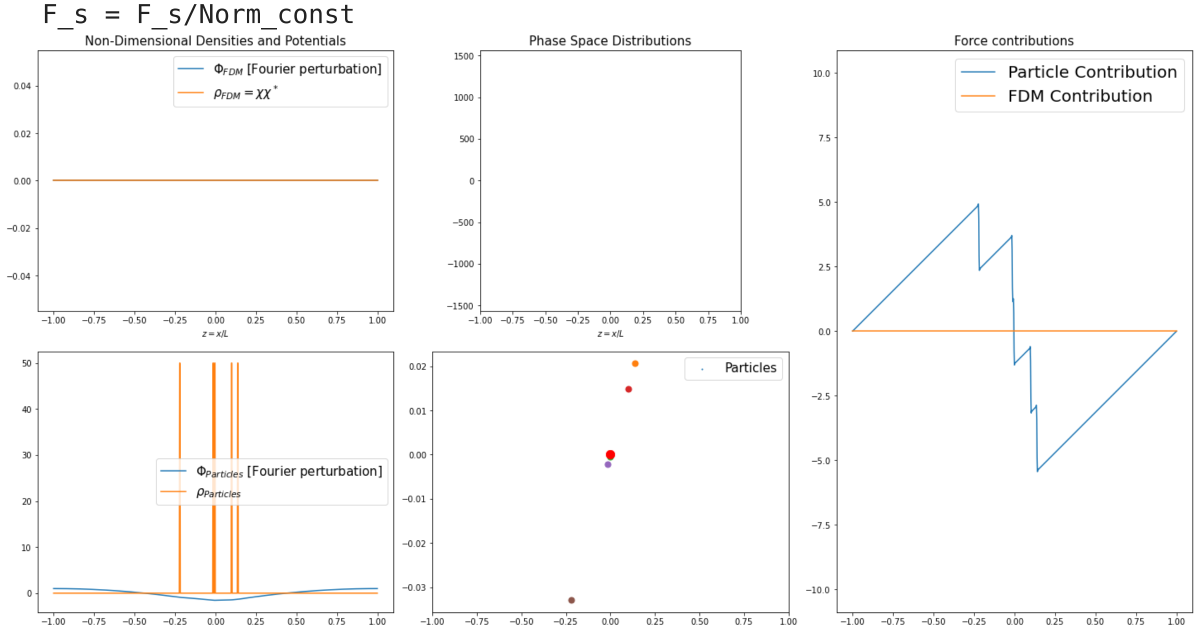
In [ ]:
```
# import Analysis

# Args = [
#     [1,1,0,0.2,5],
#     ]

# for args in Args:
#     print("----------New Analysis--------")
#     print(
#         f"r = {args[0]}",
#         f"mu = {args[1]}",
#         f"Num_bosons = {args[2]}",
#         f"sigma = {args[3]}",
#         f"Num_stars = {args[4]}"
#     )
#     Analysis.analysis(*args)
```

In [ ]:
```
import Analysis

#[ ..., [r,m,Num_bosons,sigma,Num_stars],...]
# Args = [
#     [0.5,1.0,0,1,10000],
#     [0.5,1.0,10000,1,10000],
#     [1,0.5,20000,1,10000],
#     [5,0.1,100000,1,10000],
#     [10,0.05,200000,1,10000],
#     [50,0.01,1000000,1,10000],
#     [0.5,1.0,10000,1,0]
# ]

Args = [
    [1,1,0,0.2,5],
    [1,1,0,0.002,500],
    [1,1,0,0.001,1000],
    [1,1,0,0.0002,5000],
    [1,1,0,0.0001,10000],
    [1,1,0,0.00002,50000],
    [1,1,0,0.00001,100000]
]

z_rms_s = []
v_rms_s = []
for args in Args:
    print("----------New Analysis--------")
    print(
        f"r = {args[0]}",
        f"mu = {args[1]}",
        f"Num_bosons = {args[2]}",
        f"sigma = {args[3]}",
        f"Num_stars = {args[4]}"
    )
    z_rms, v_rms = Analysis.analysis(*args)
    z_rms_s.append(z_rms)
    v_rms_s.append(v_rms)
```

```
----------New Analysis--------
r = 1 mu = 1 Num_bosons = 0 sigma = 0.2 Num_stars = 5
```

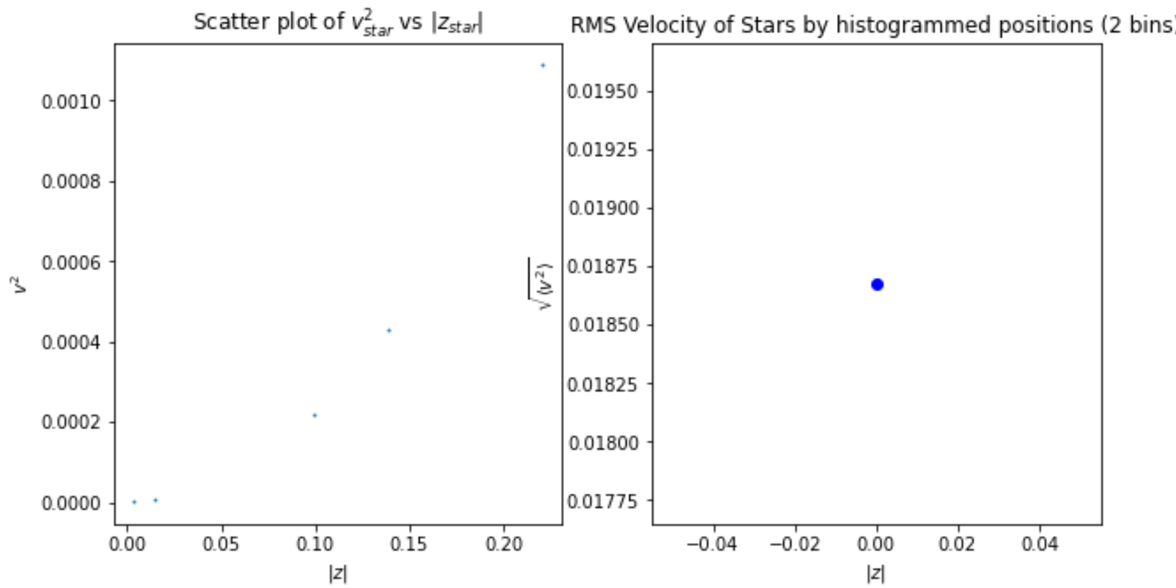


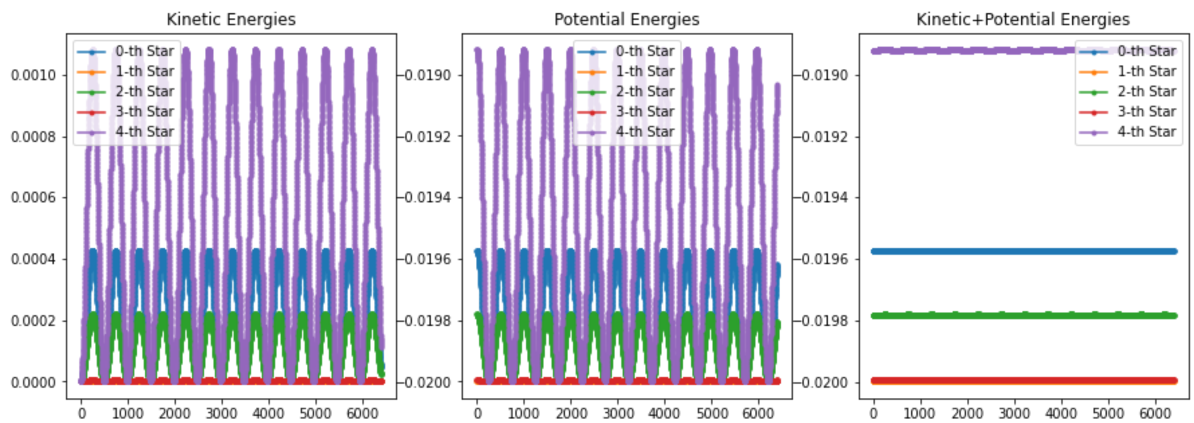

```
/home/boris/Documents/Research/FDM_n_Bodies/OneD/WaveNonDim.py:129: Runtime
Warning: invalid value encountered in true_divide
  F_s = F_s/Norm_const
```

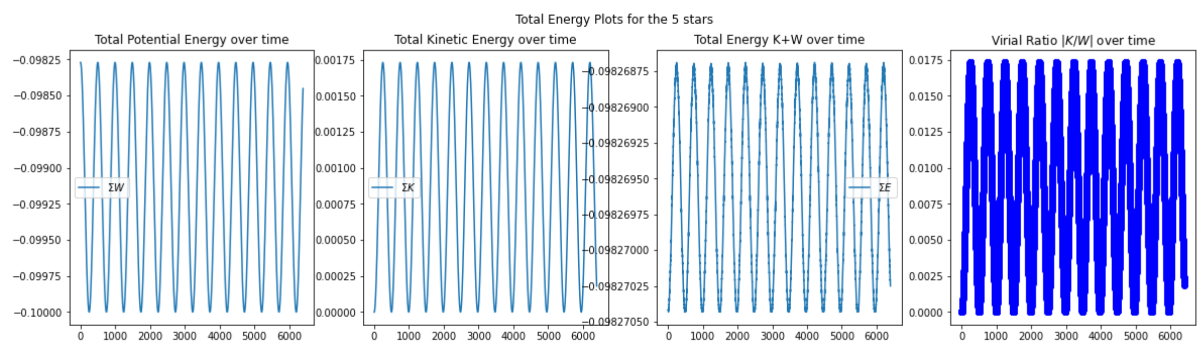


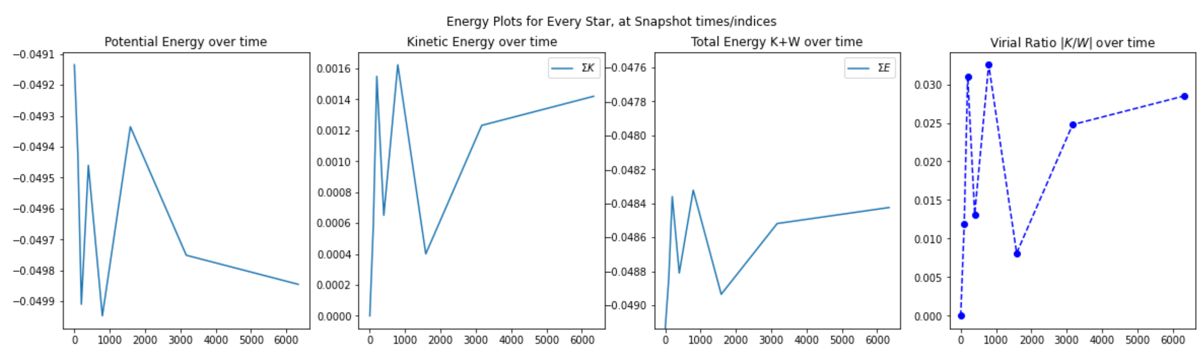

```
v_rms = 0.018675629975302l3
z_rms = 0.1247108499035957
K_avg = 0.5*m*v_rms^2 = 0.00017438957748720172 (m=1)
=> 2*K_avg = 0.00034877915497440344
W_avg = 0.6235542495179784
------------------
K_tot = 0.00017438957748720172
K_avg = 3.487791549744035e-05
W_tot = -0.09752468531550332
W_avg = -0.019504937063100664
```

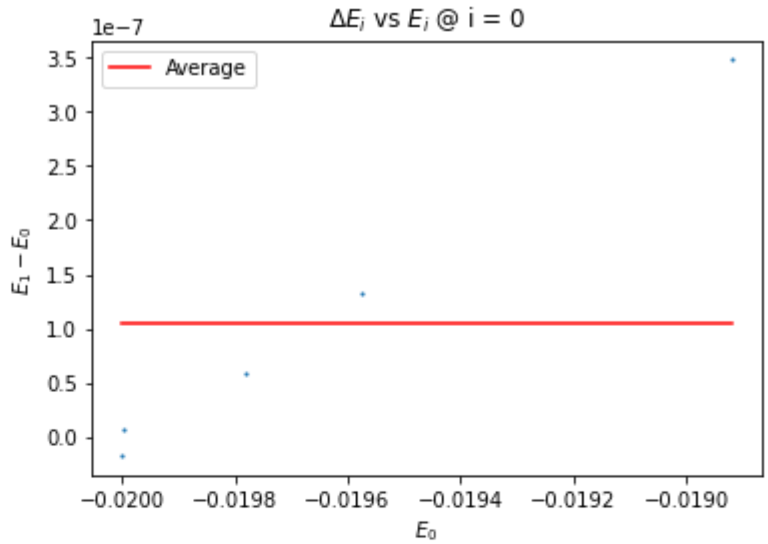
```
/home/boris/Documents/Research/FDM_n_Bodies/1D_Codes/Non-Dim/Analysis/Analy
sis.py:277: RuntimeWarning: invalid value encountered in true_divide
  v_rms_array = bins/bins_counts
```


Scatter plot of v_{star}^2 vs $|z_{star}|$
v^2
$|z|$
RMS Velocity of Stars by histogrammed positions (2 bins)
$\sqrt{\langle v^2 \rangle}$
$|z|$

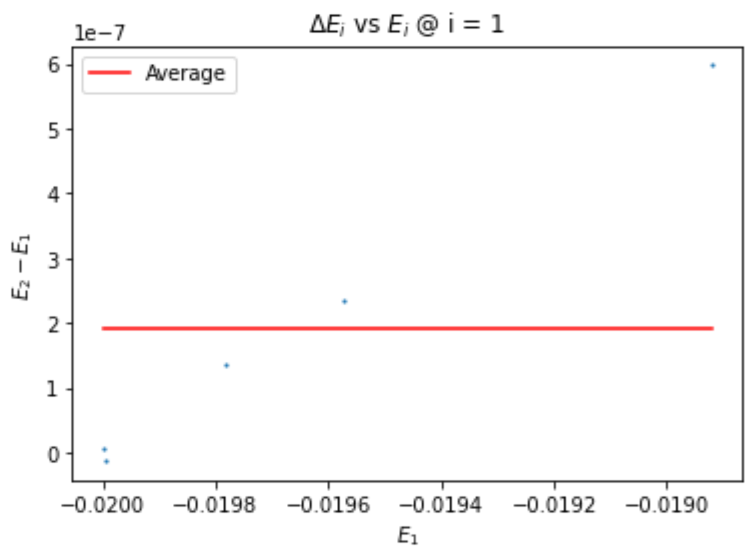

Kinetic Energies
Potential Energies
Kinetic+Potential Energies
0-th Star
1-th Star
2-th Star
3-th Star
4-th Star


Total Energy Plots for the 5 stars
Total Potential Energy over time
ΣW
Total Kinetic Energy over time
ΣK
Total Energy K+W over time
ΣE
Virial Ratio $|K/W|$ over time

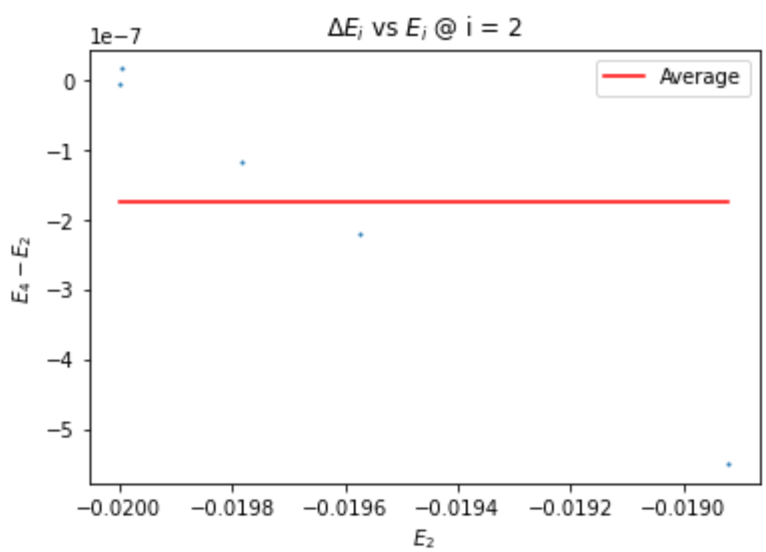

Energy Plots for Every Star, at Snapshot times/indices
Potential Energy over time
Kinetic Energy over time
ΣK
Total Energy K+W over time
ΣE
Virial Ratio $|K/W|$ over time



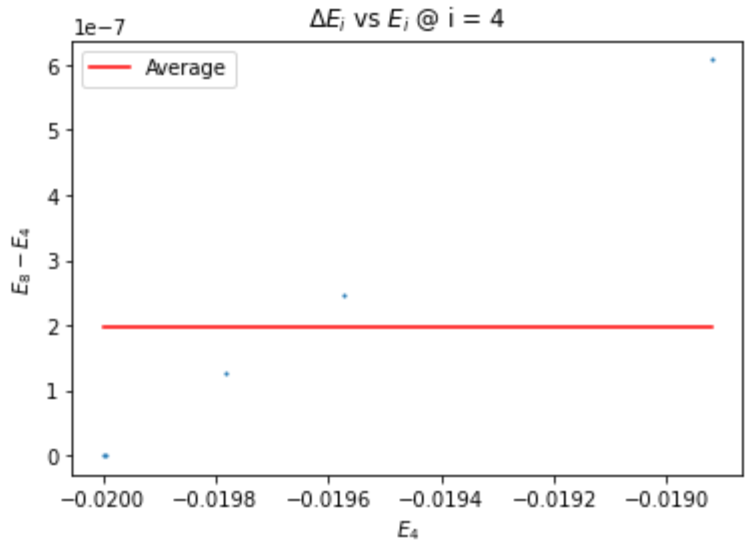
ΔE_i vs E_i @ i = 0
1e−7
Average
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
$E_1 - E_0$
−0.0200
−0.0198
−0.0196
−0.0194
−0.0192
−0.0190
E_0

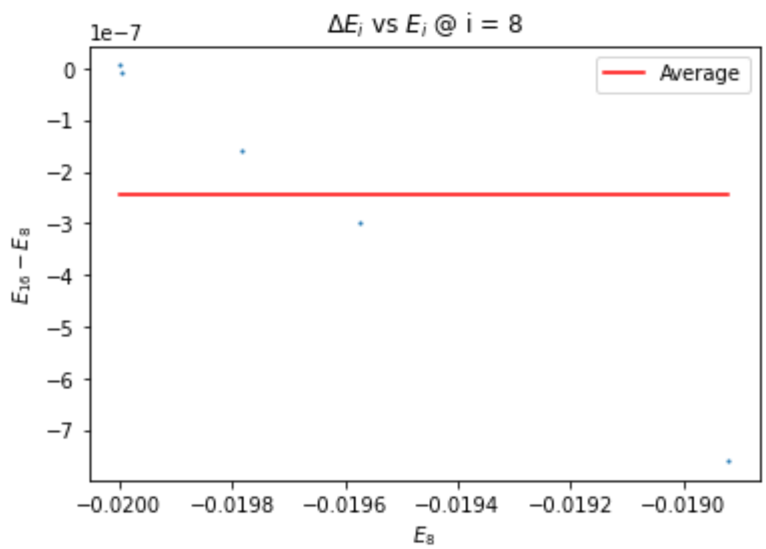


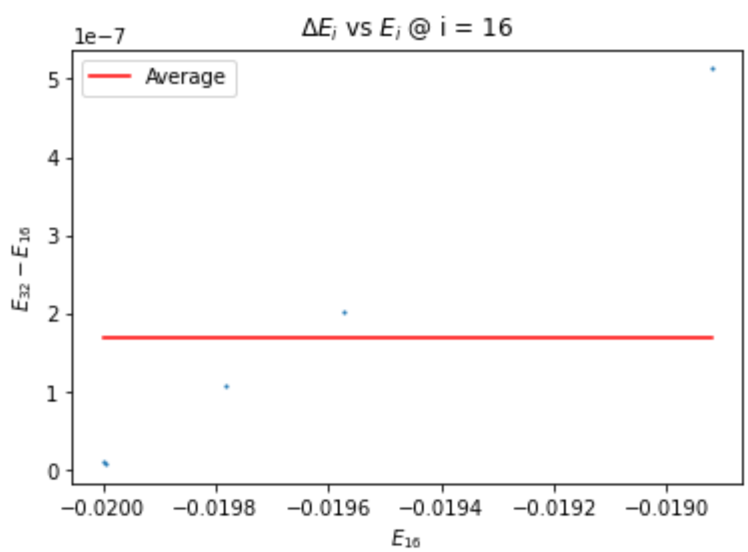
ΔE_i vs E_i @ i = 1
1e−7
Average
6
5
4
3
2
1
0
$E_2 - E_1$
−0.0200
−0.0198
−0.0196
−0.0194
−0.0192
−0.0190
E_1

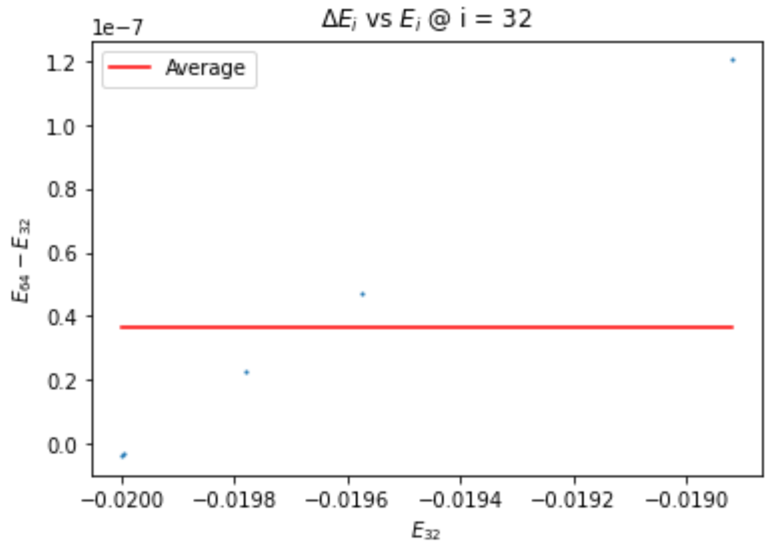


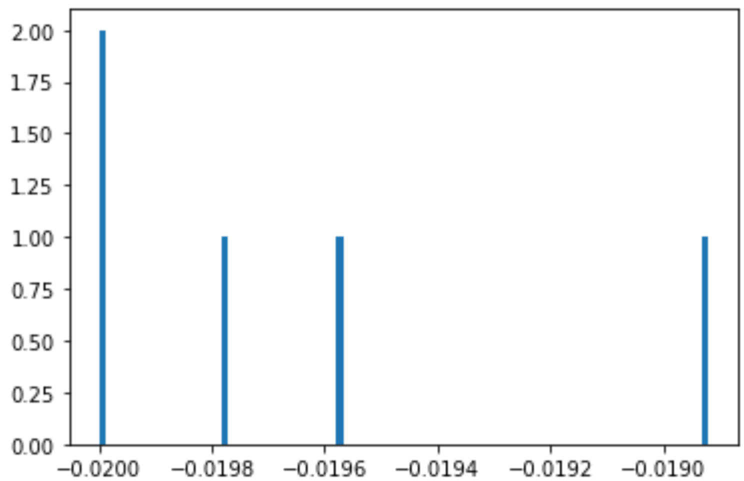
ΔE_i vs E_i @ i = 2
1e−7
Average
0
−1
−2
−3
−4
−5
$E_4 - E_2$
−0.0200
−0.0198
−0.0196
−0.0194
−0.0192
−0.0190
E_2

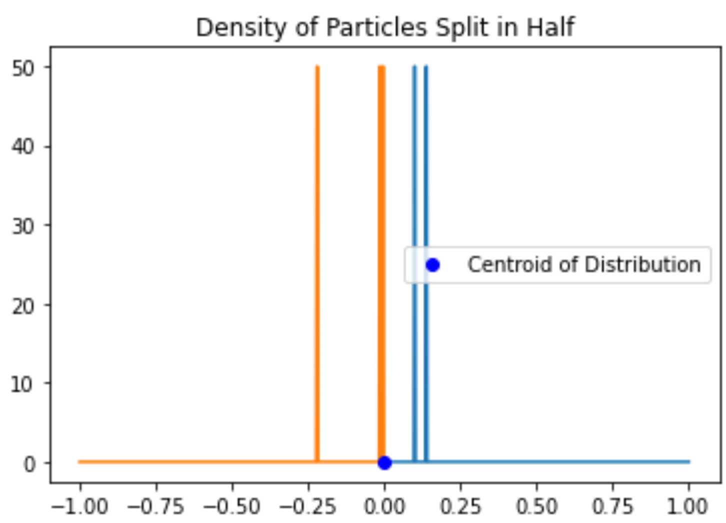

ΔE_i vs E_i @ i = 4
1e−7
Average
$E_8 - E_4$
E_4
−0.0200 −0.0198 −0.0196 −0.0194 −0.0192 −0.0190
0 1 2 3 4 5 6


ΔE_i vs E_i @ i = 8
1e−7
Average
$E_{16} - E_8$
E_8
−0.0200 −0.0198 −0.0196 −0.0194 −0.0192 −0.0190
0 −1 −2 −3 −4 −5 −6 −7

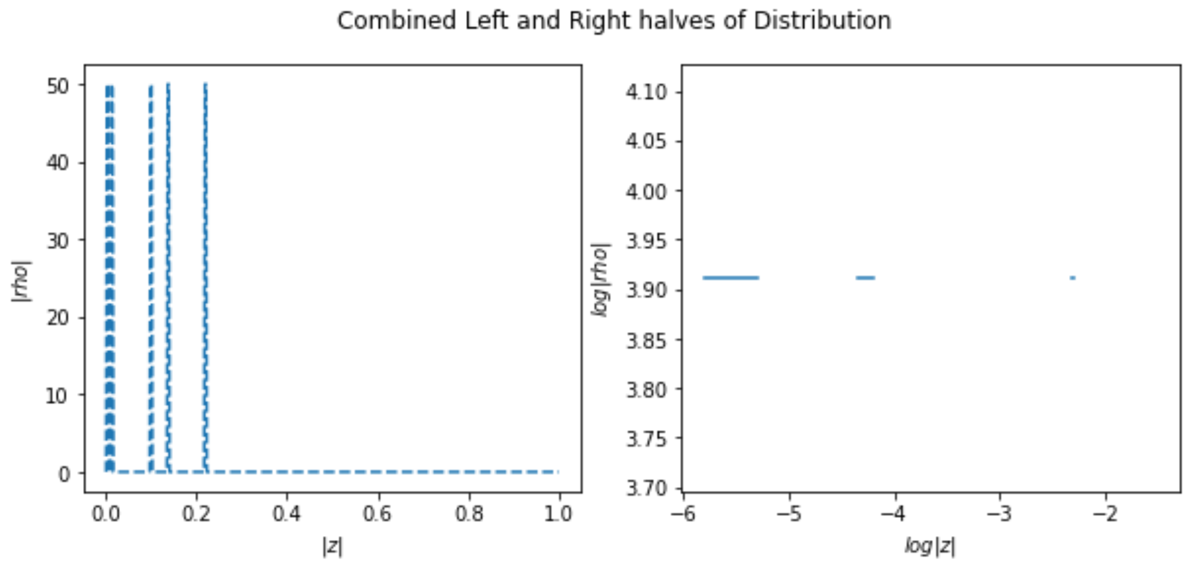

ΔE_i vs E_i @ i = 16
1e−7
Average
$E_{32} - E_{16}$
E_{16}
−0.0200 −0.0198 −0.0196 −0.0194 −0.0192 −0.0190
0 1 2 3 4 5



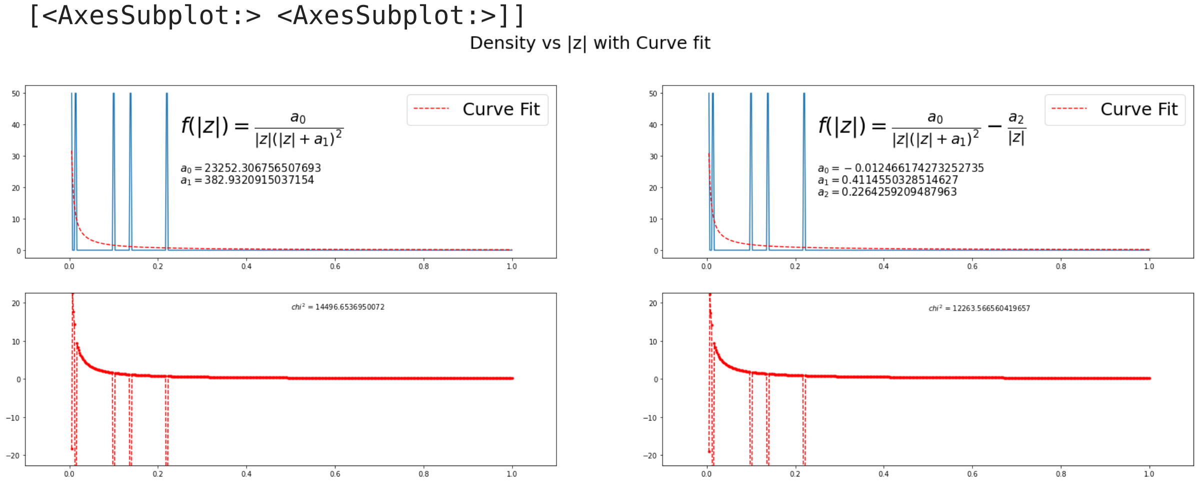


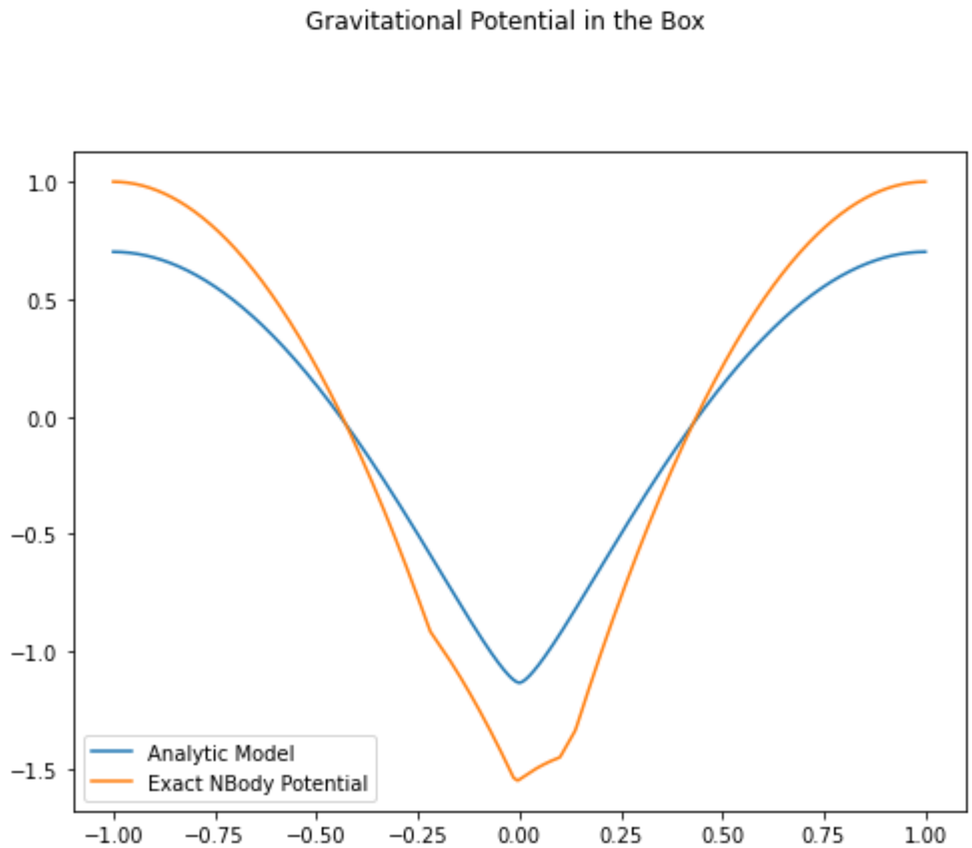


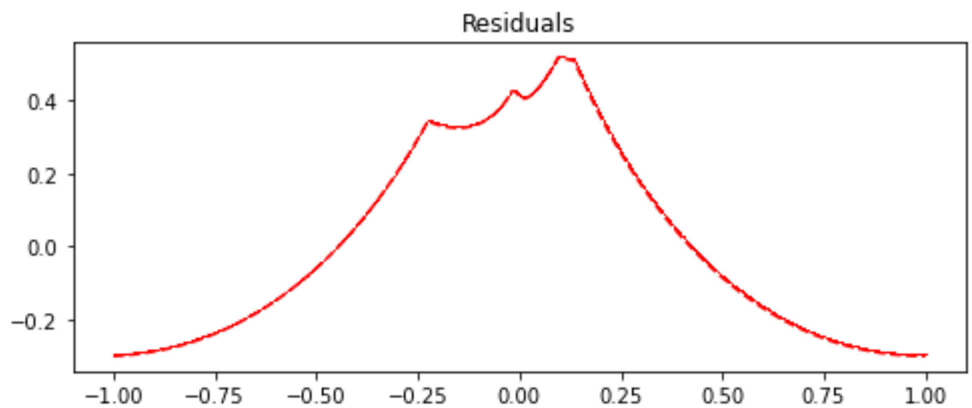

```
/home/boris/Documents/Research/FDM_n_Bodies/1D_Codes/Non-Dim/Analysis/Analy
sis.py:487: RuntimeWarning: divide by zero encountered in log
  ax[1].plot(np.log(z_right),np.log(rho_whole))
```



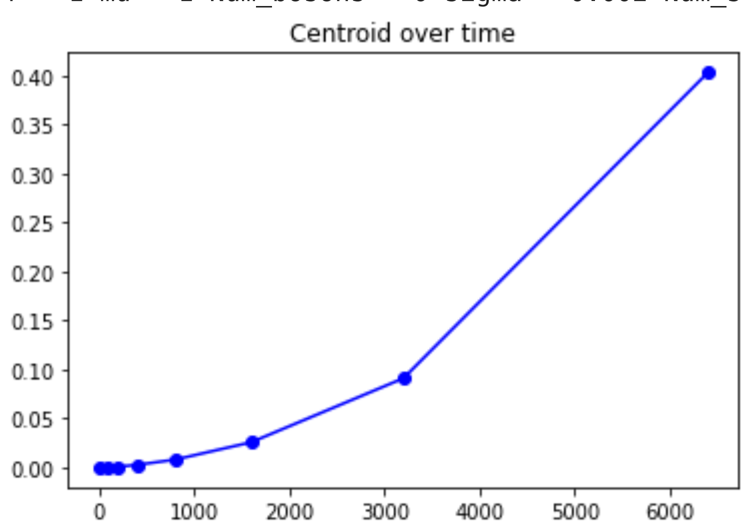

```
Check
Check
#columns = 2
[[<AxesSubplot:> <AxesSubplot:>]
 [<AxesSubplot:> <AxesSubplot:>]]
```

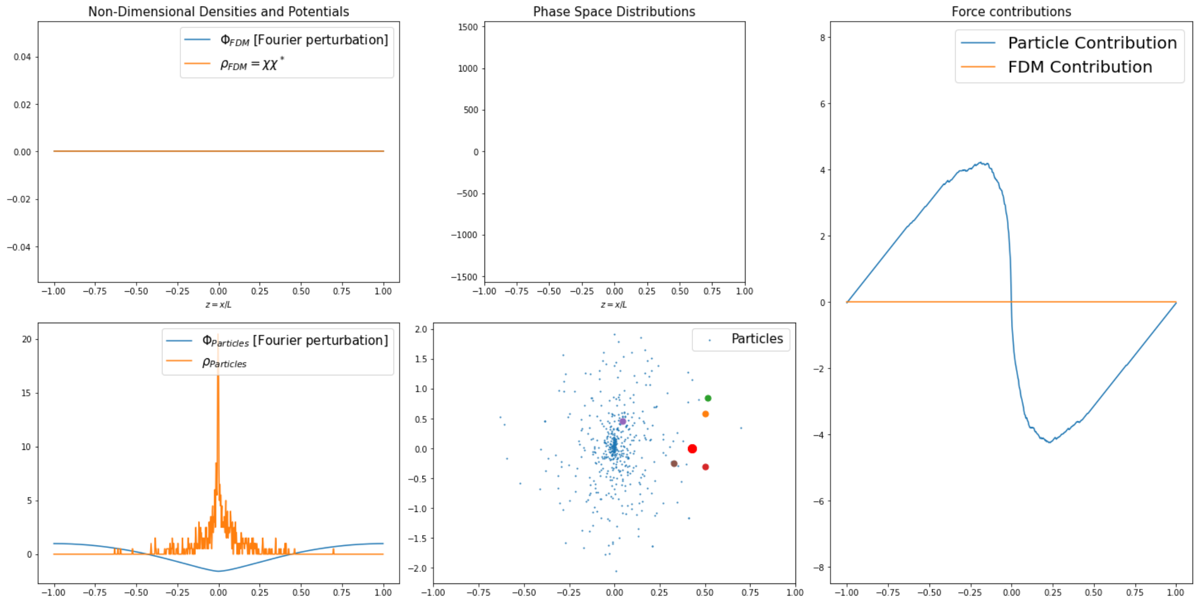




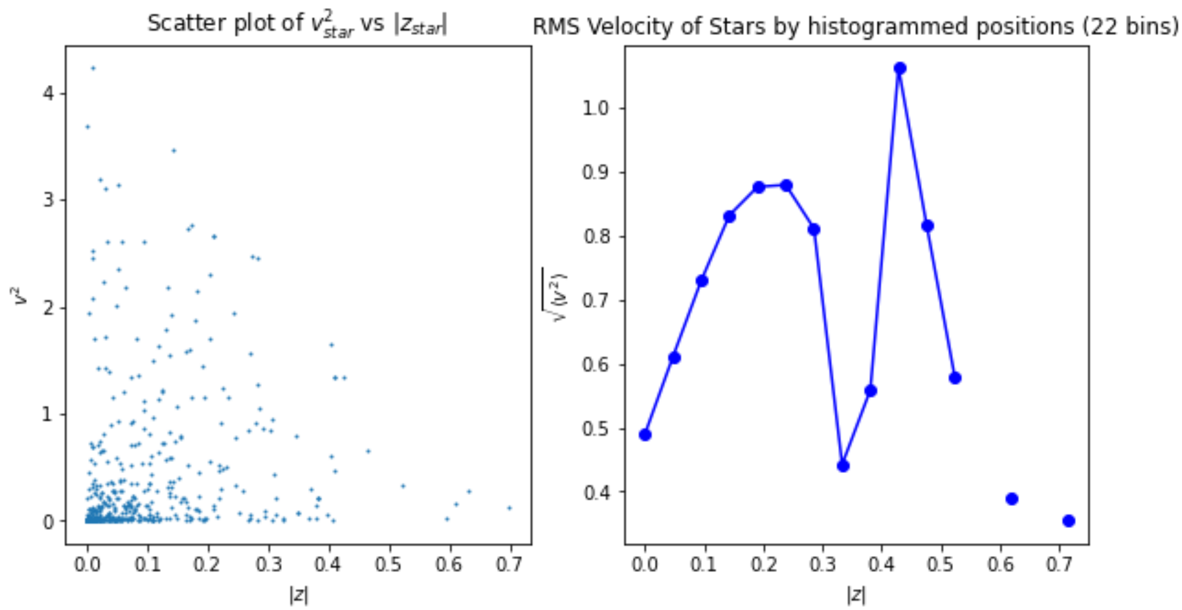


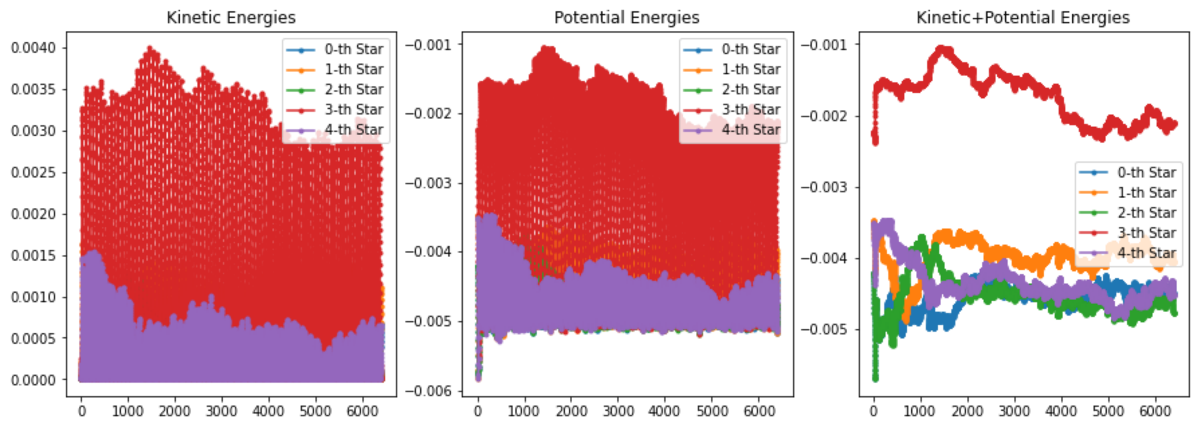

```
----------New Analysis--------
r = 1 mu = 1 Num_bosons = 0 sigma = 0.002 Num_stars = 500
```

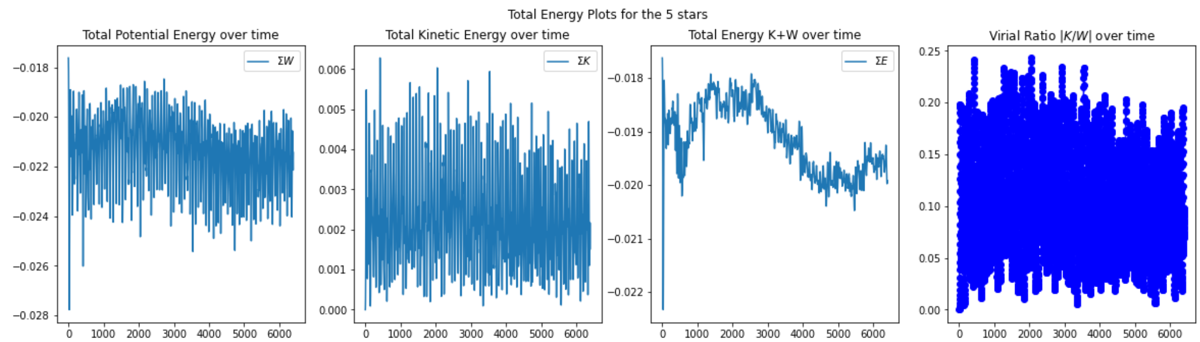


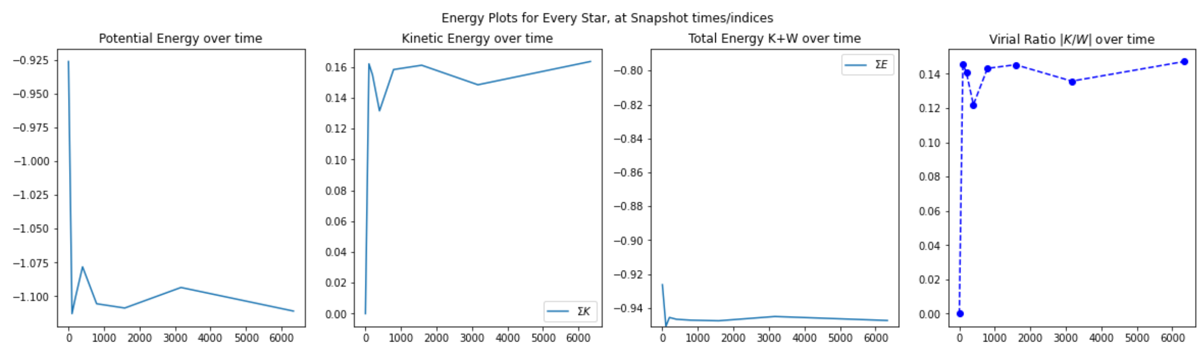


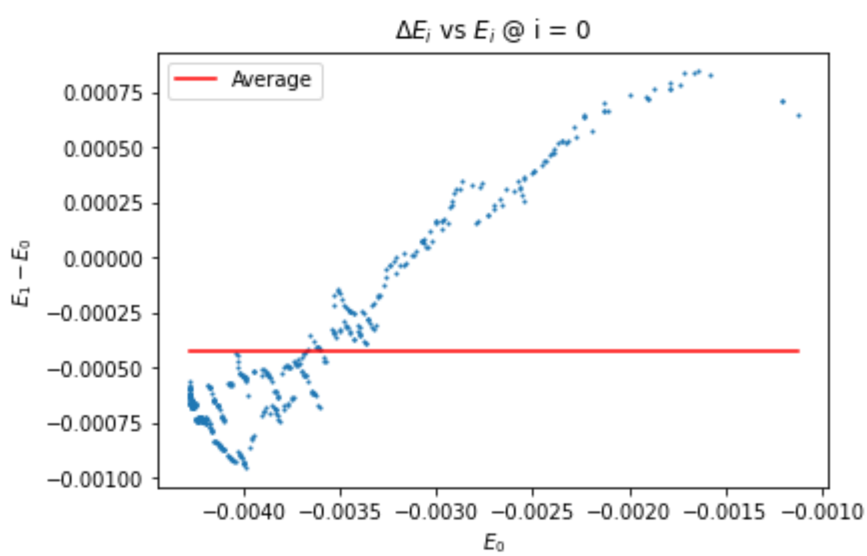

```
v_rms = 0.6232813792609679
z_rms = 0.14315501233400055
K_avg = 0.5*m*v_rms^2 = 0.19423983886672727 (m=1)
=> 2*K_avg = 0.38847967773345454
W_avg = 71.57750616700028
------------------
K_tot = 0.1942398388667273
K_avg = 0.00038847967773345
W_tot = -0.12655169196887073
W_avg = -0.0002531033839377415
```

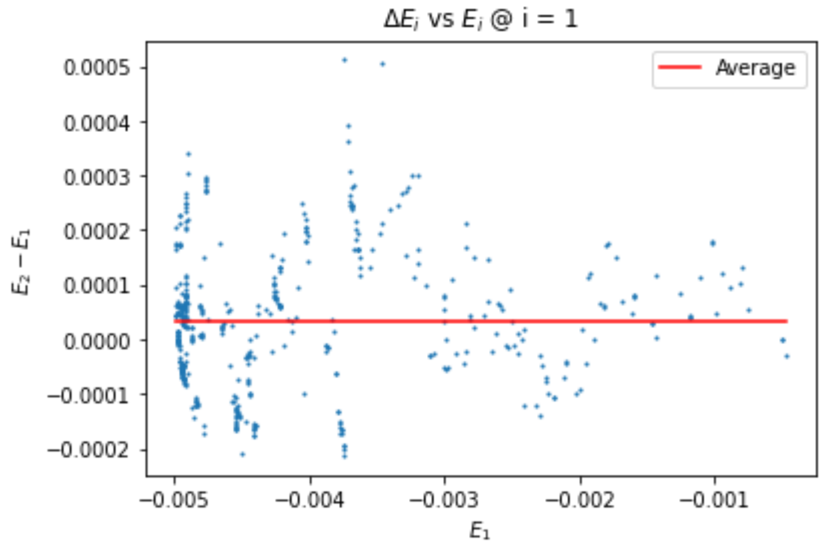


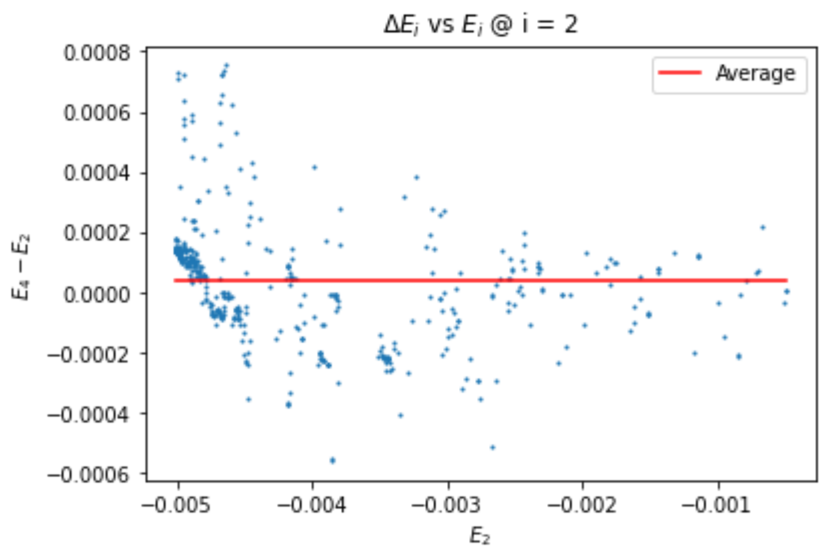

Kinetic Energies
Potential Energies
Kinetic+Potential Energies
0-th Star
1-th Star
2-th Star
3-th Star
4-th Star

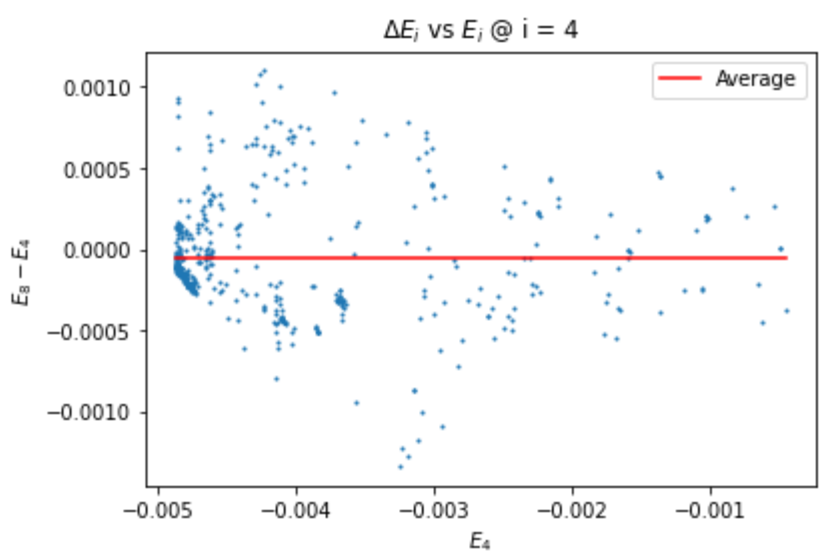

Total Energy Plots for the 5 stars
Total Potential Energy over time
ΣW
Total Kinetic Energy over time
ΣK
Total Energy K+W over time
ΣE
Virial Ratio |K/W| over time

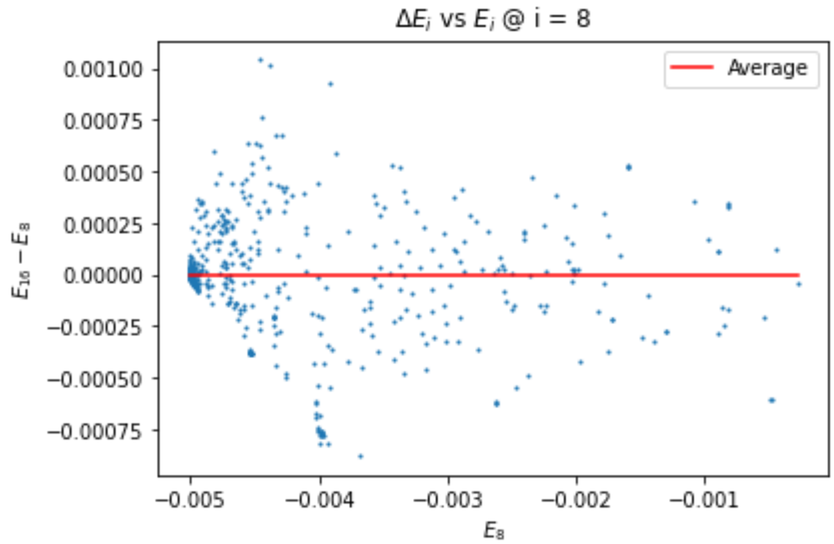

Energy Plots for Every Star, at Snapshot times/indices
Potential Energy over time
Kinetic Energy over time
ΣK
Total Energy K+W over time
ΣE
Virial Ratio |K/W| over time

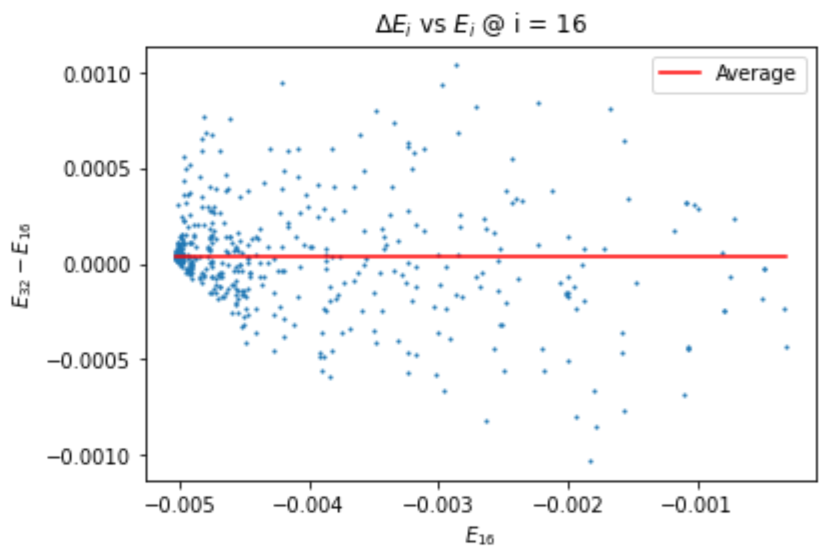

ΔE_i vs E_i @ i = 0
Average
E_1 − E_0
E_0

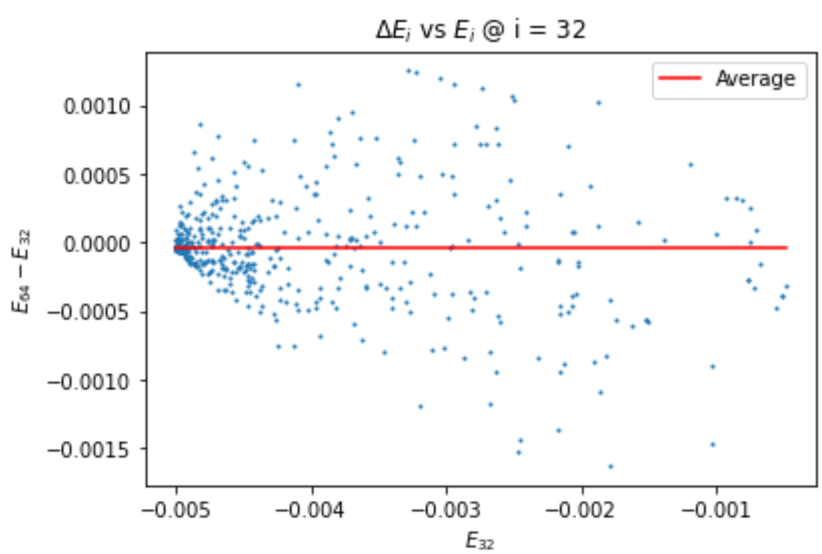

ΔE_i vs E_i @ i = 1
Average
$E_2 - E_1$
E_1
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
−0.0001
−0.0002
−0.005
−0.004
−0.003
−0.002
−0.001

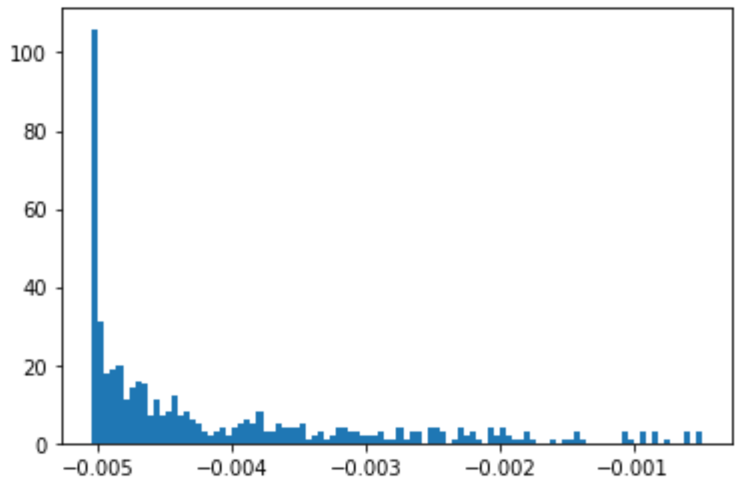

ΔE_i vs E_i @ i = 2
Average
$E_4 - E_2$
E_2
0.0008
0.0006
0.0004
0.0002
0.0000
−0.0002
−0.0004
−0.0006
−0.005
−0.004
−0.003
−0.002
−0.001


ΔE_i vs E_i @ i = 4
Average
$E_8 - E_4$
E_4
0.0010
0.0005
0.0000
−0.0005
−0.0010
−0.005
−0.004
−0.003
−0.002
−0.001

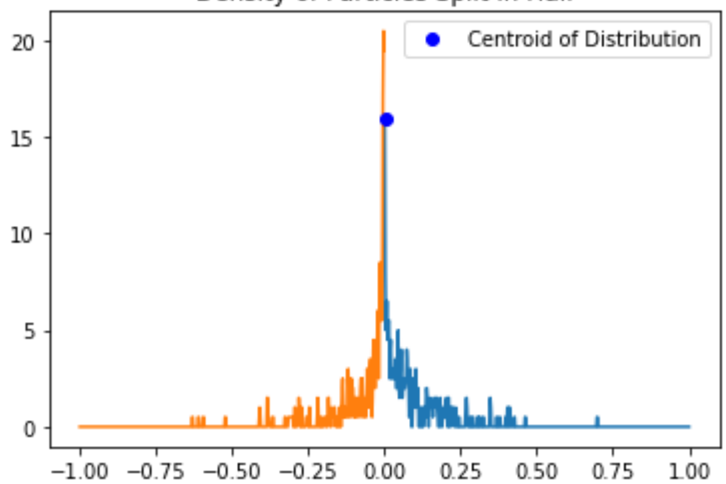

ΔE_i vs E_i @ i = 8
Average
$E_{16} - E_8$
E_8


ΔE_i vs E_i @ i = 16
Average
$E_{32} - E_{16}$
E_{16}

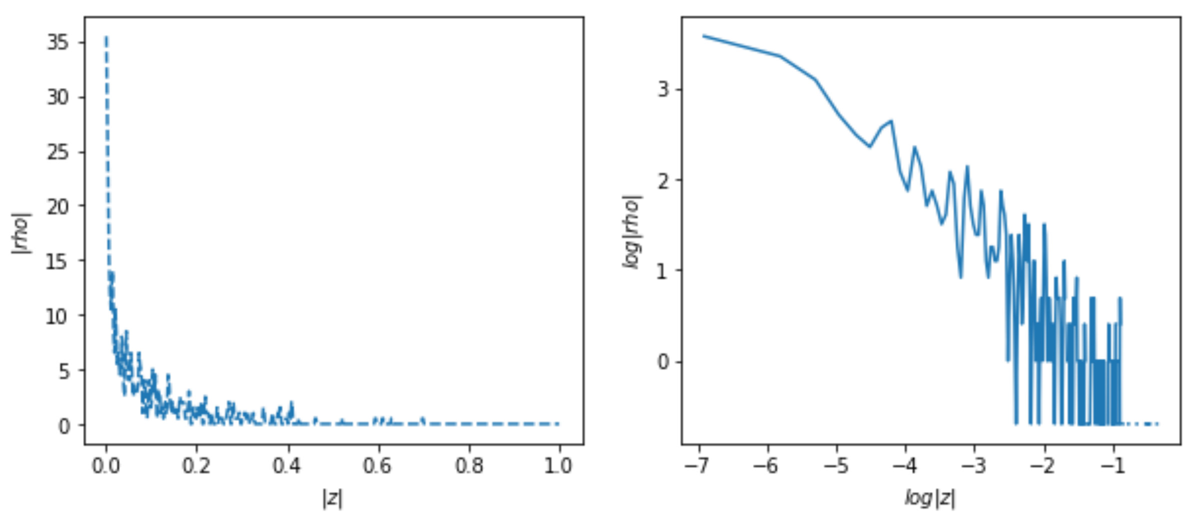

ΔE_i vs E_i @ i = 32
Average
$E_{64} - E_{32}$
E_{32}



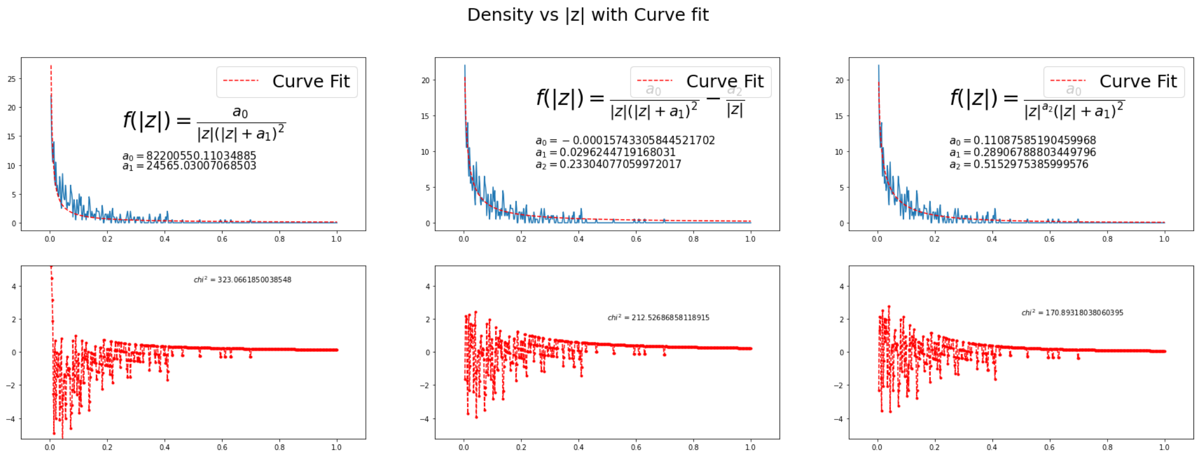
Density of Particles Split in Half

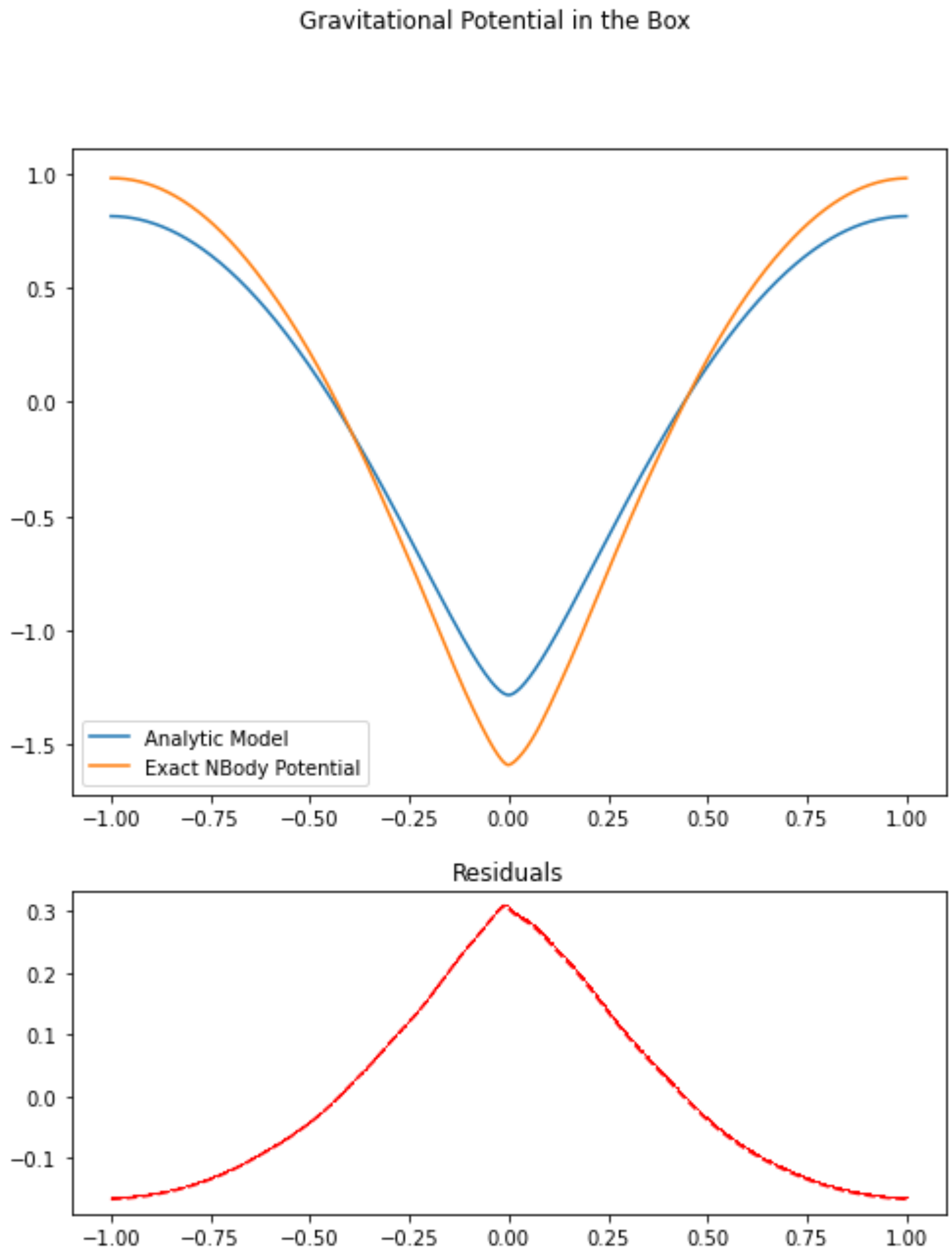


Combined Left and Right halves of Distribution

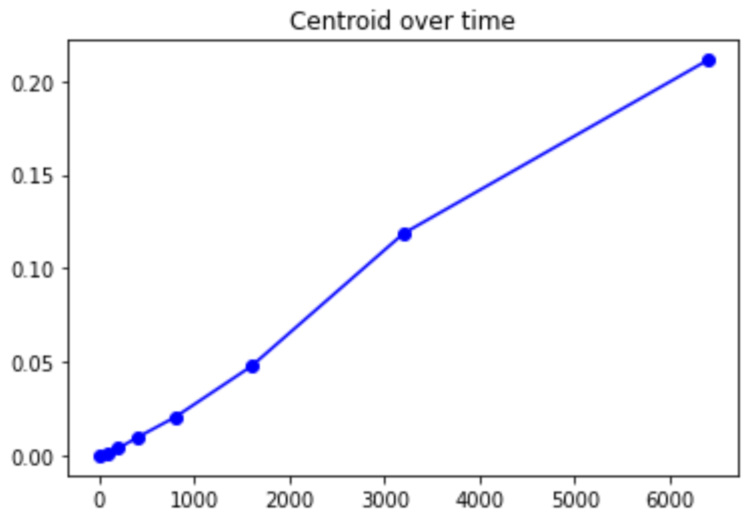


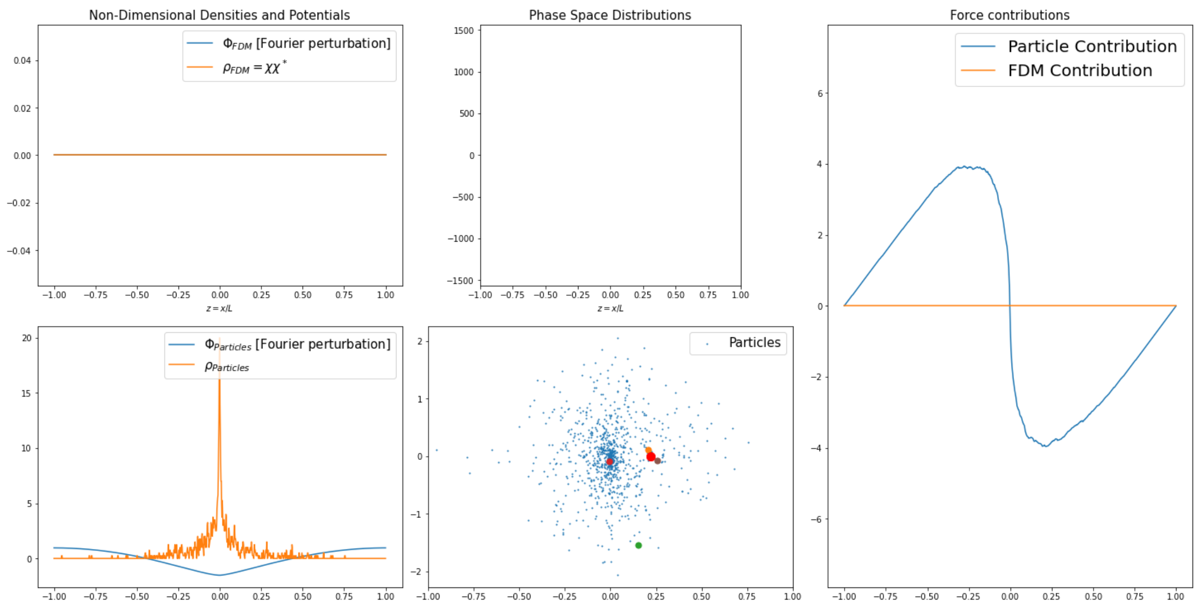

```
Check
Check
Check
#columns = 3
[[<AxesSubplot:> <AxesSubplot:> <AxesSubplot:>]
 [<AxesSubplot:> <AxesSubplot:> <AxesSubplot:>]]
```



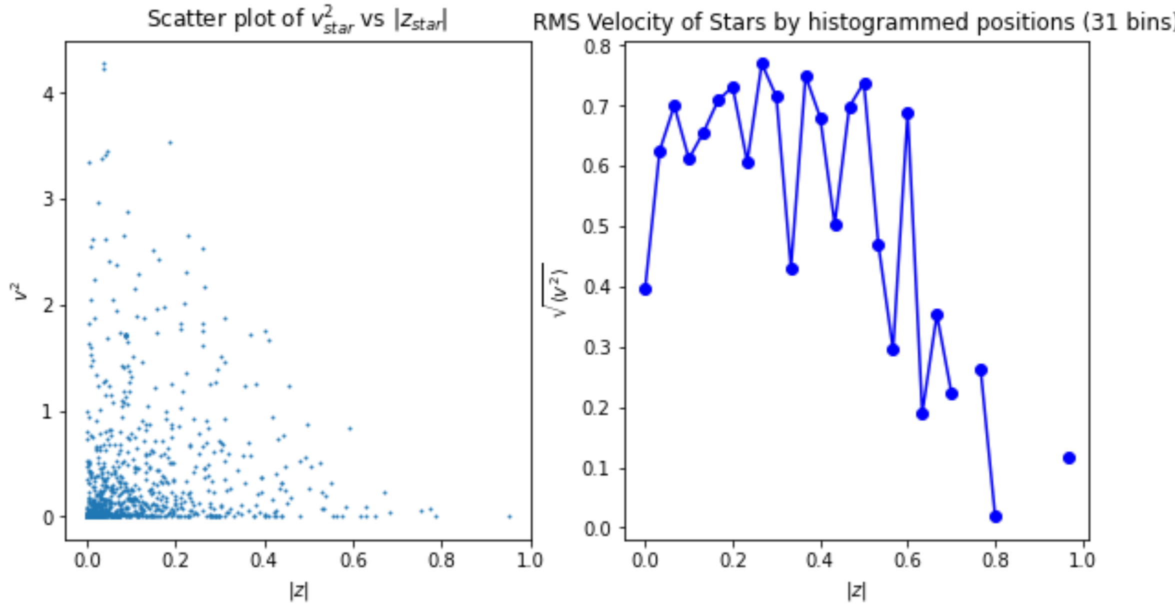


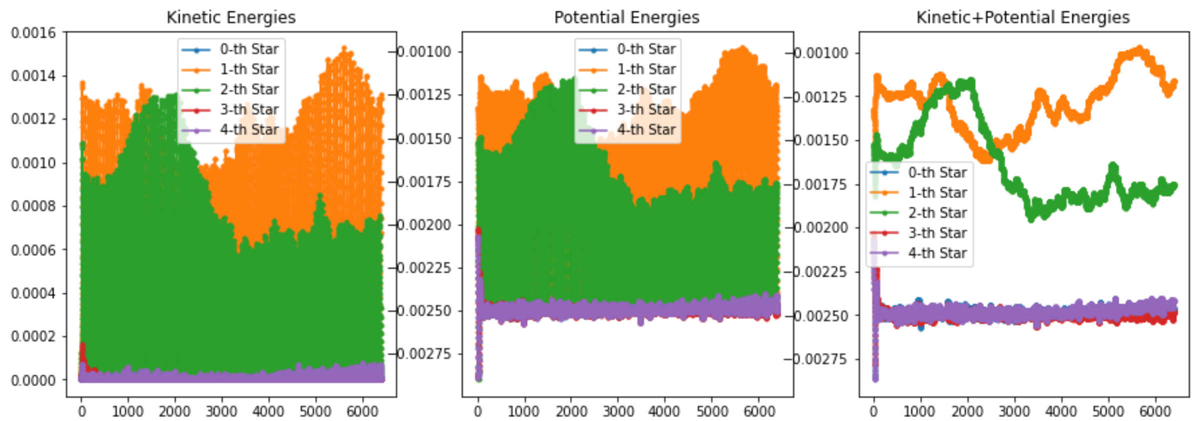

```
----------New Analysis--------
r = 1 mu = 1 Num_bosons = 0 sigma = 0.001 Num_stars = 1000
```



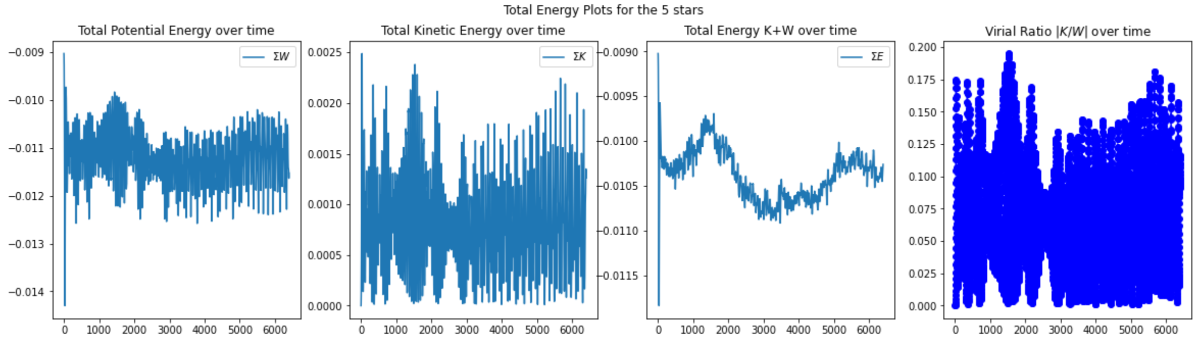



```
v_rms = 0.5695975824840549
z_rms = 0.17100296207179705
K_avg = 0.5*m*v_rms^2 = 0.16222070298583985 (m=1)
=> 2*K_avg = 0.3244414059716797
W_avg = 171.00296207179704
------------------
K_tot = 0.16222070298583985
K_avg = 0.00016222070298583984
W_tot = -0.1737126516320072
W_avg = -0.0001737126516320072
```

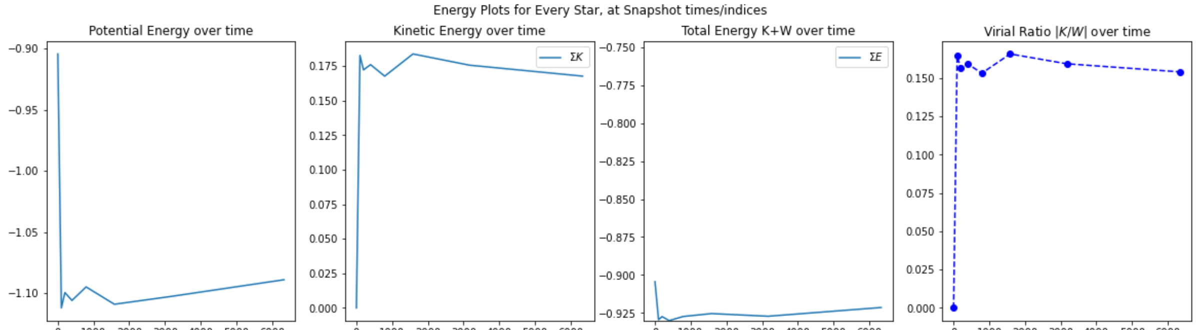


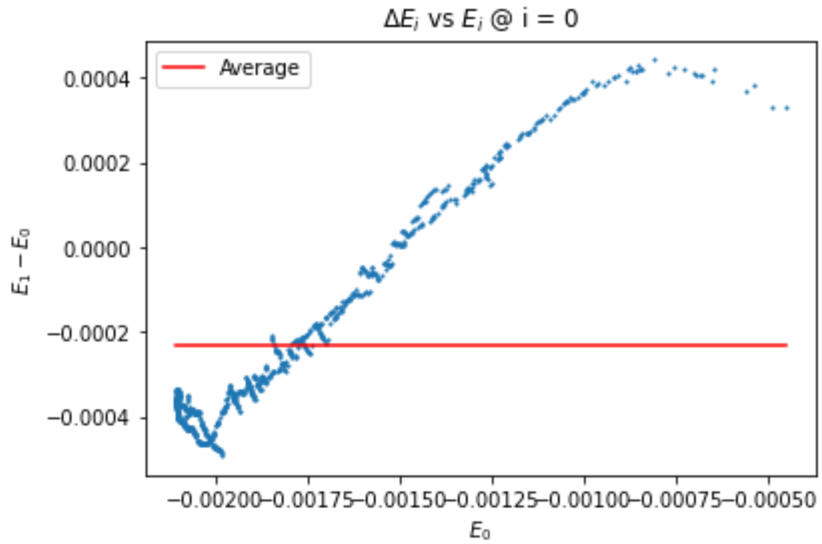


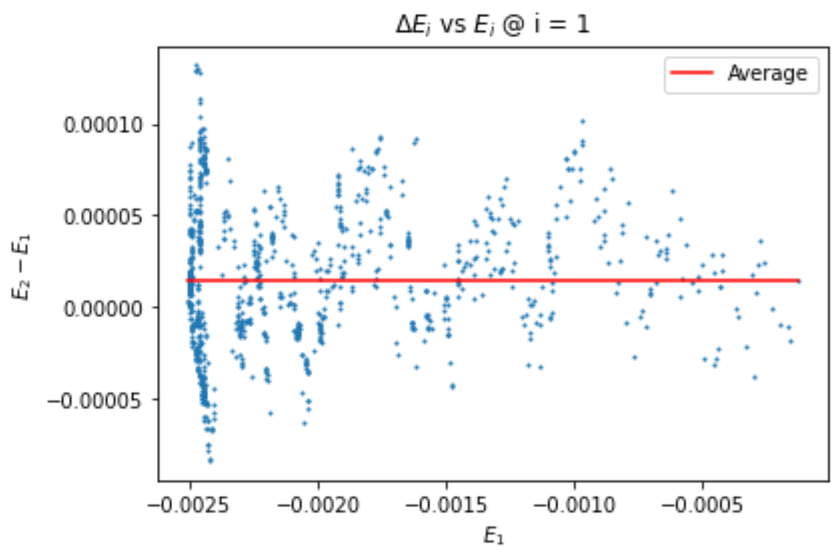
Total Energy Plots for the 5 stars

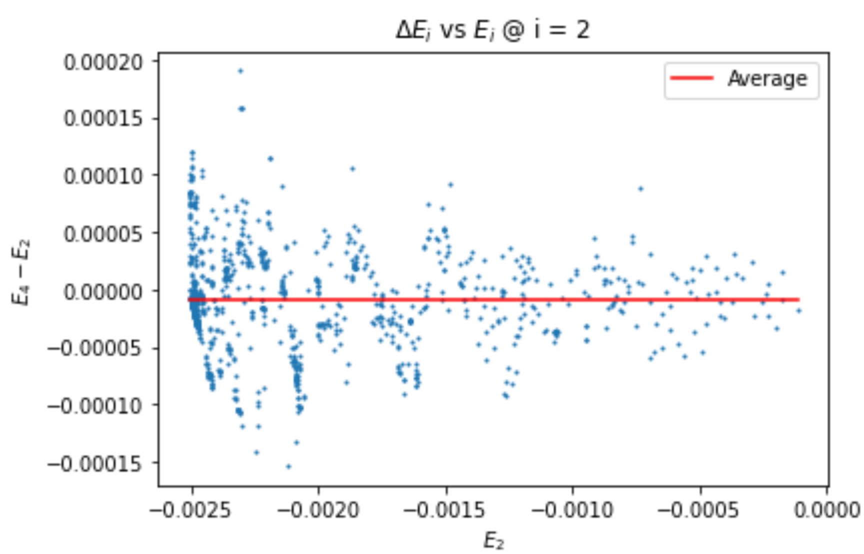


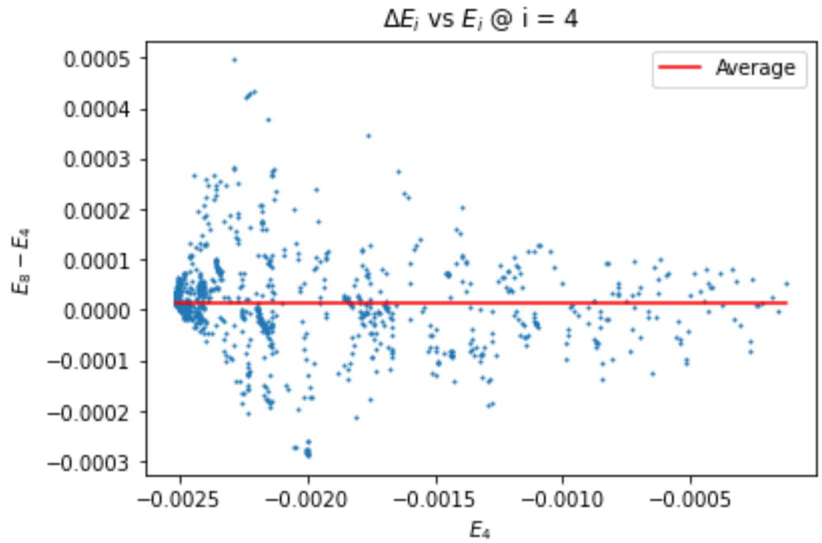
Energy Plots for Every Star, at Snapshot times/indices



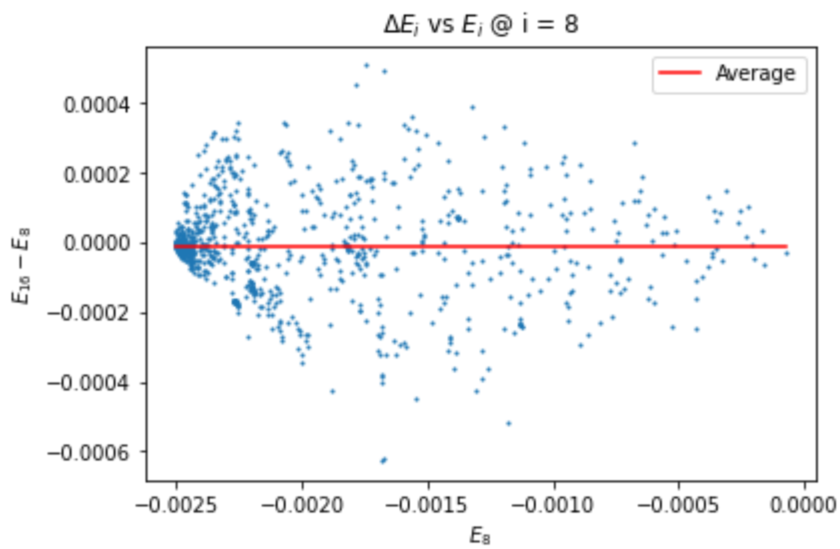

ΔE_i vs E_i @ i = 0
Average
$E_1 - E_0$
E_0
0.0004
0.0002
0.0000
−0.0002
−0.0004
−0.00200 −0.00175 −0.00150 −0.00125 −0.00100 −0.00075 −0.00050


ΔE_i vs E_i @ i = 1
Average
$E_2 - E_1$
E_1
0.00010
0.00005
0.00000
−0.00005
−0.0025 −0.0020 −0.0015 −0.0010 −0.0005

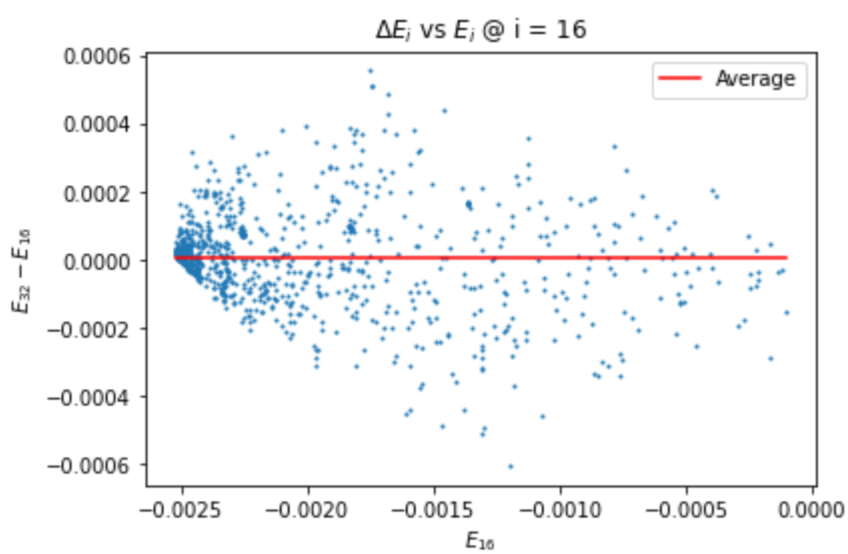

ΔE_i vs E_i @ i = 2
Average
$E_4 - E_2$
E_2
0.00020
0.00015
0.00010
0.00005
0.00000
−0.00005
−0.00010
−0.00015
−0.0025 −0.0020 −0.0015 −0.0010 −0.0005 0.0000



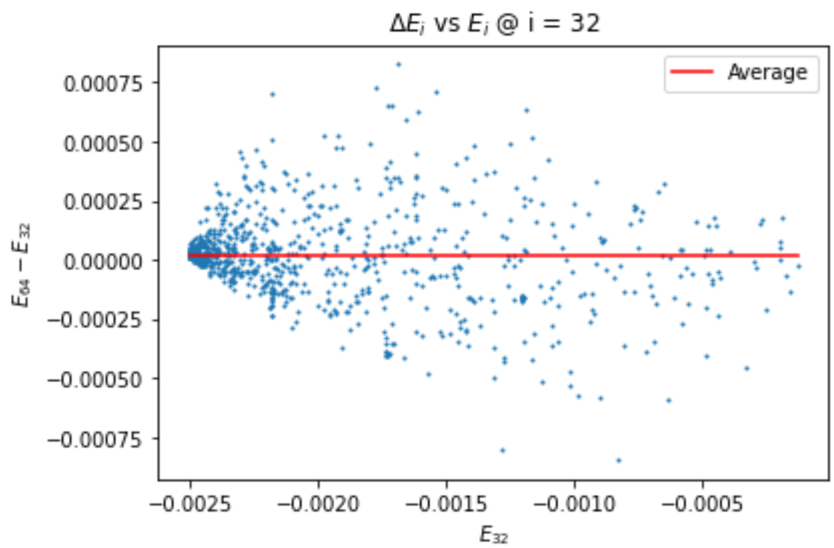
ΔE_i vs E_i @ i = 4
Average
$E_8 - E_4$
E_4
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
−0.0001
−0.0002
−0.0003
−0.0025
−0.0020
−0.0015
−0.0010
−0.0005

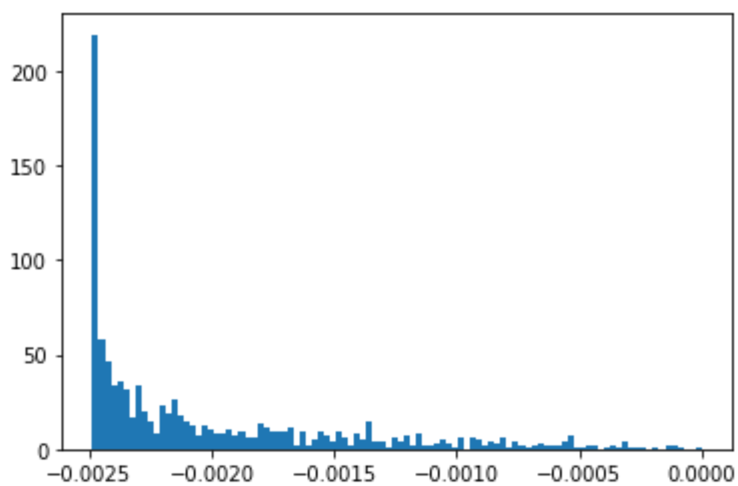


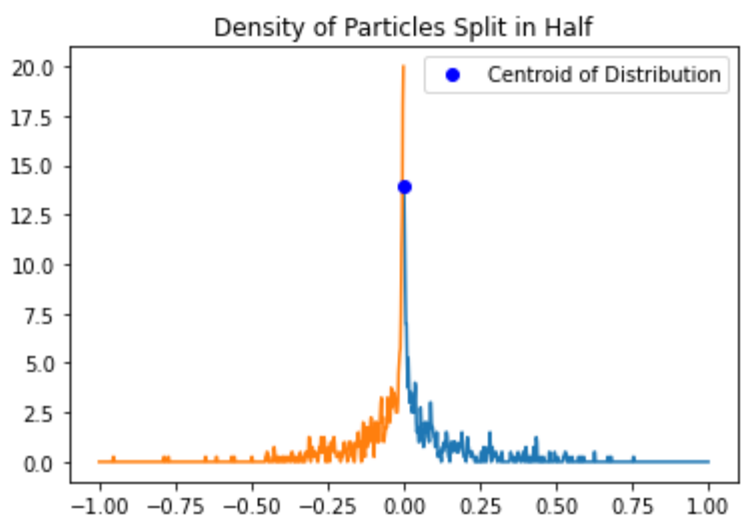
ΔE_i vs E_i @ i = 8
Average
$E_{16} - E_8$
E_8
0.0004
0.0002
0.0000
−0.0002
−0.0004
−0.0006
−0.0025
−0.0020
−0.0015
−0.0010
−0.0005
0.0000

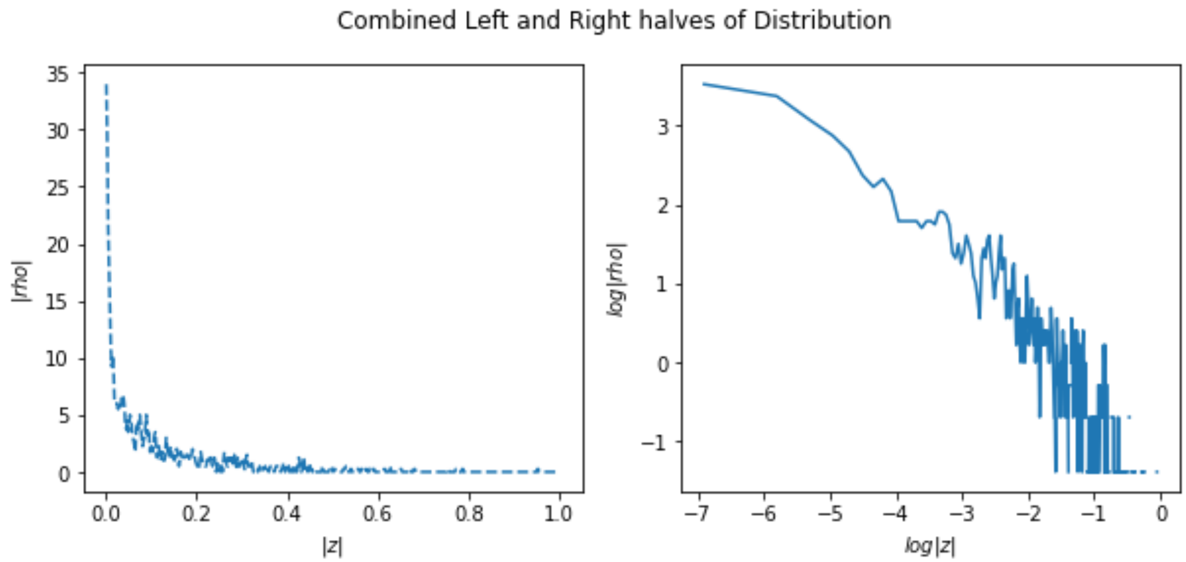


ΔE_i vs E_i @ i = 16
Average
$E_{32} - E_{16}$
E_{16}
0.0006
0.0004
0.0002
0.0000
−0.0002
−0.0004
−0.0006
−0.0025
−0.0020
−0.0015
−0.0010
−0.0005
0.0000

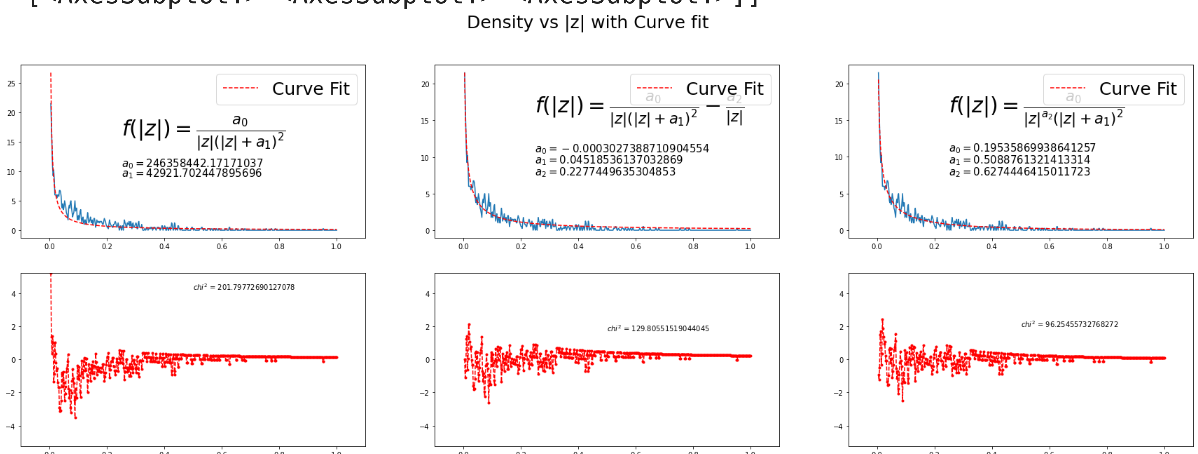

ΔE_i vs E_i @ i = 32
Average
0.00075
0.00050
0.00025
0.00000
−0.00025
−0.00050
−0.00075
$E_{64} - E_{32}$
−0.0025
−0.0020
−0.0015
−0.0010
−0.0005
E_{32}

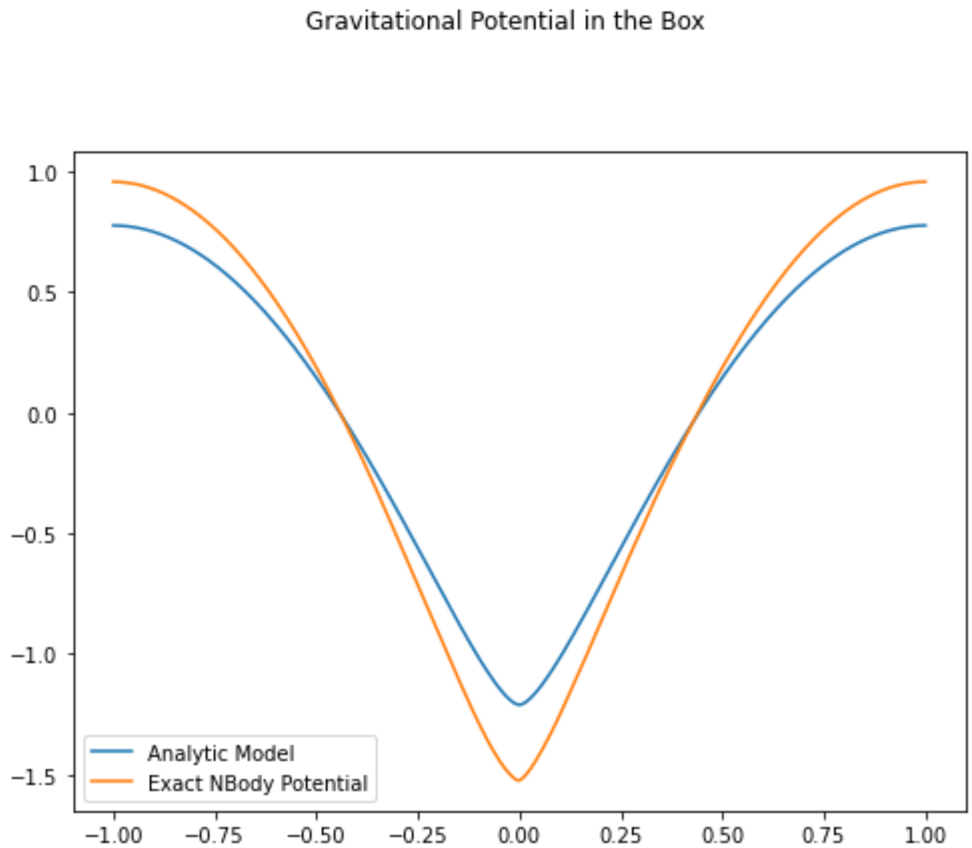

200
150
100
50
0
−0.0025
−0.0020
−0.0015
−0.0010
−0.0005
0.0000

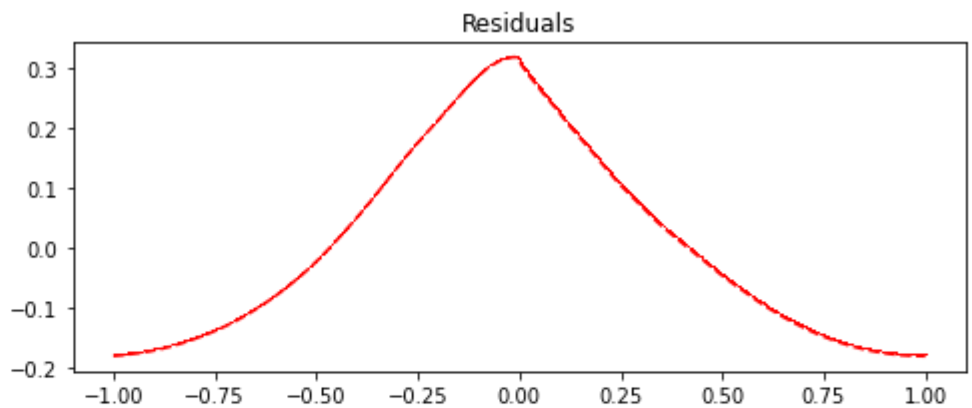

Density of Particles Split in Half
Centroid of Distribution
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00



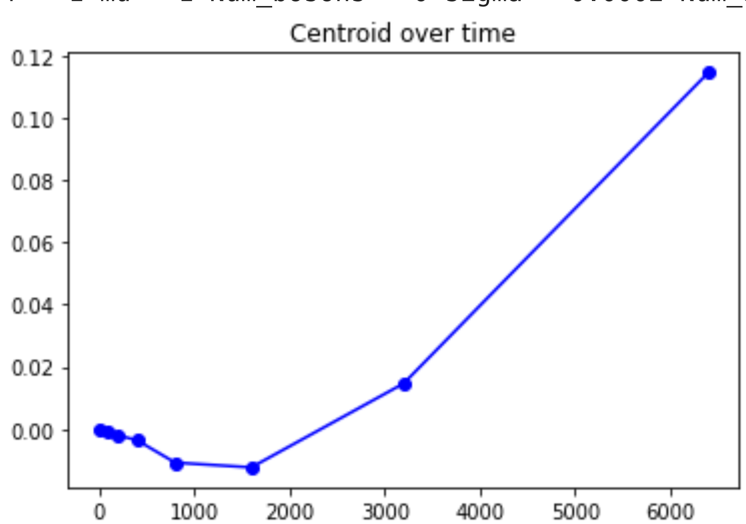

```
Check
Check
Check
#columns = 3
[[<AxesSubplot:> <AxesSubplot:> <AxesSubplot:>]
 [<AxesSubplot:> <AxesSubplot:> <AxesSubplot:>]]
```

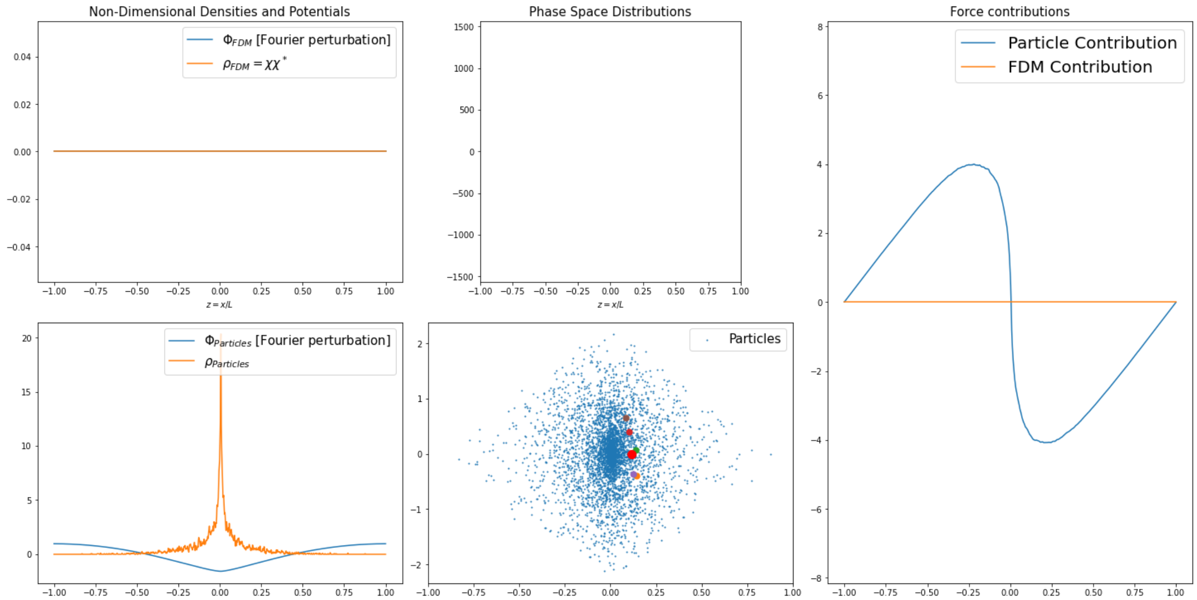




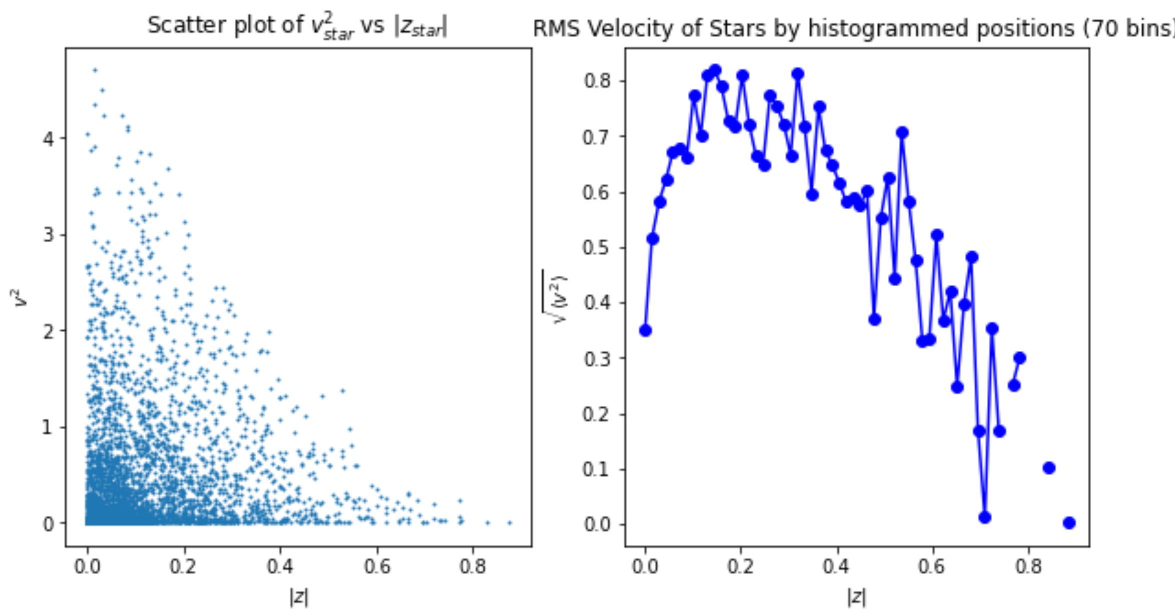


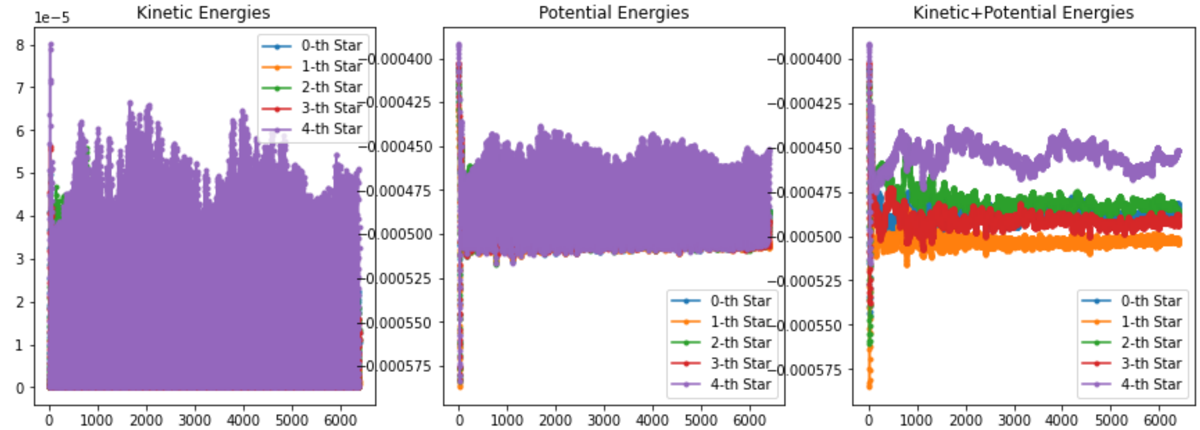

```
----------New Analysis--------
r = 1 mu = 1 Num_bosons = 0 sigma = 0.0002 Num_stars = 5000
```

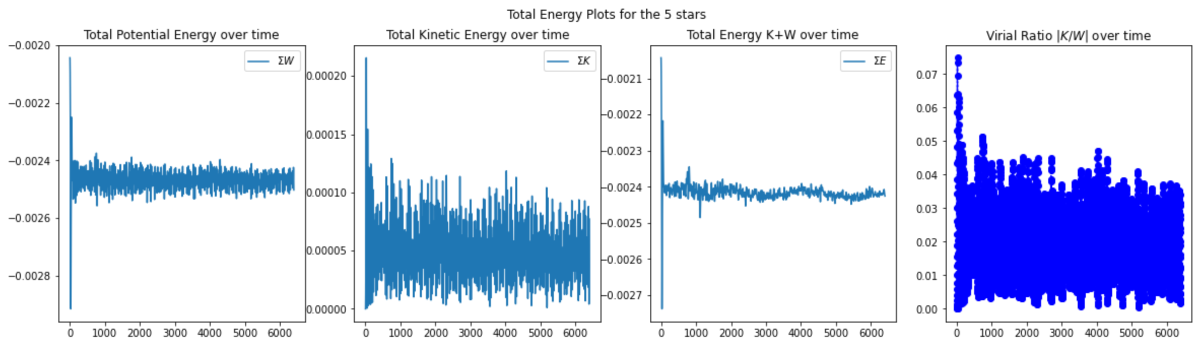


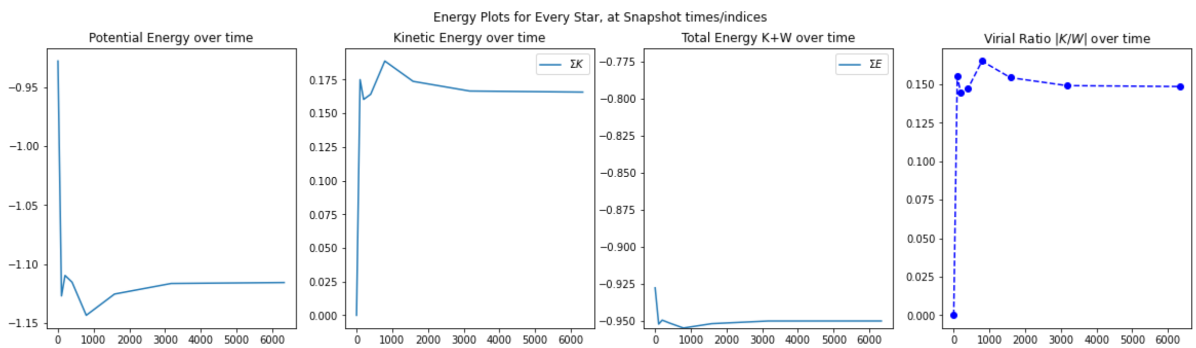


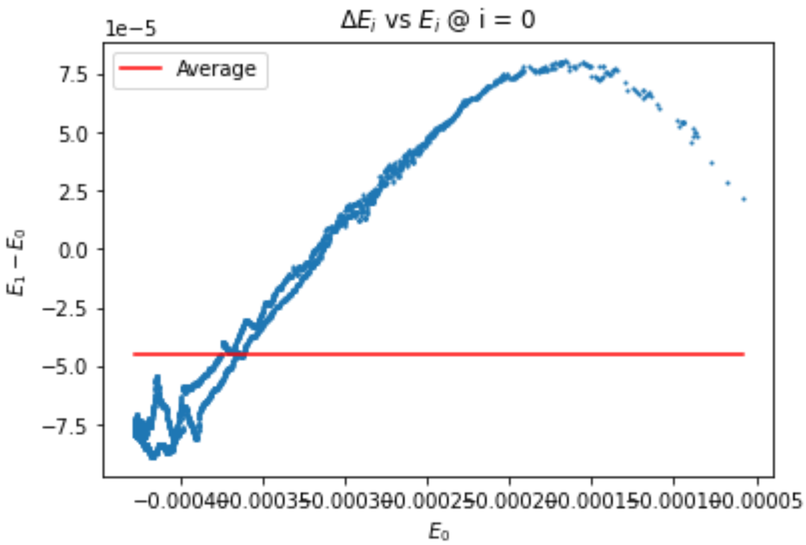

```
v_rms = 0.5878237987107265
z_rms = 0.15515828199421808
K_avg = 0.5*m*v_rms^2 = 0.17276840916535438 (m=1)
=> 2*K_avg = 0.34553681833070876
W_avg = 775.7914099710904
------------------
K_tot = 0.17276840916535405
K_avg = 3.455368183307081e-05
W_tot = -0.1467679478017614
W_avg = -2.9353589560352277e-05
```

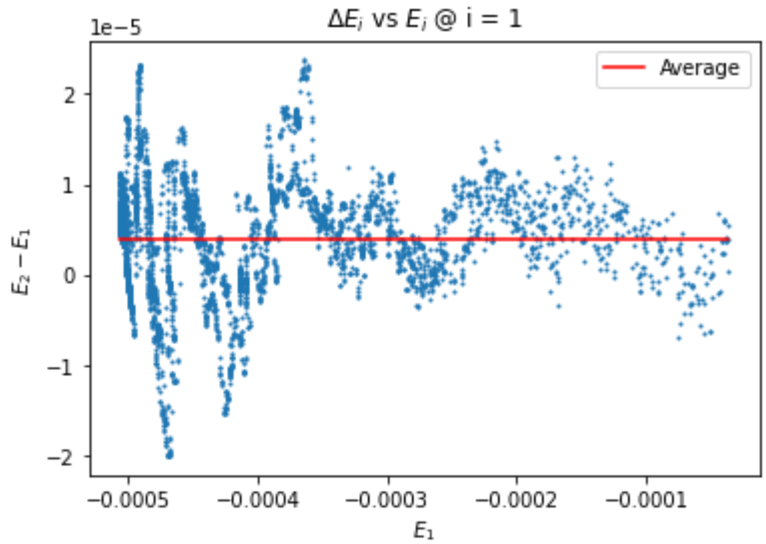


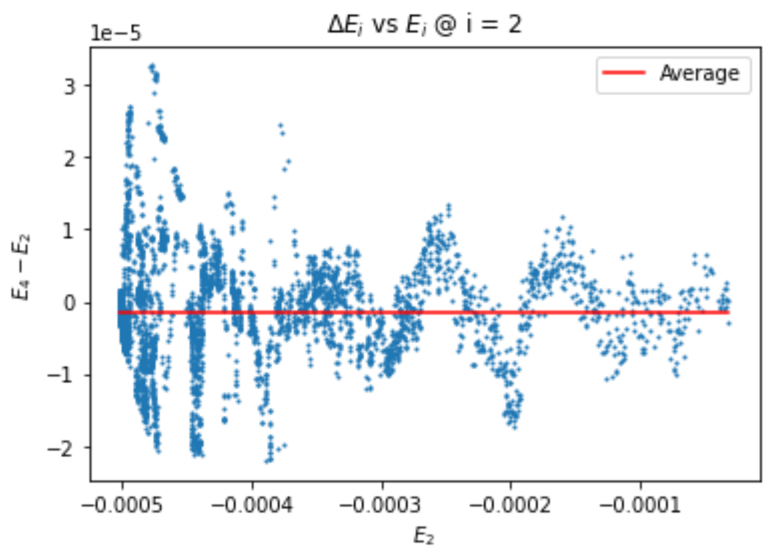

Kinetic Energies
Potential Energies
Kinetic+Potential Energies
0-th Star
1-th Star
2-th Star
3-th Star
4-th Star

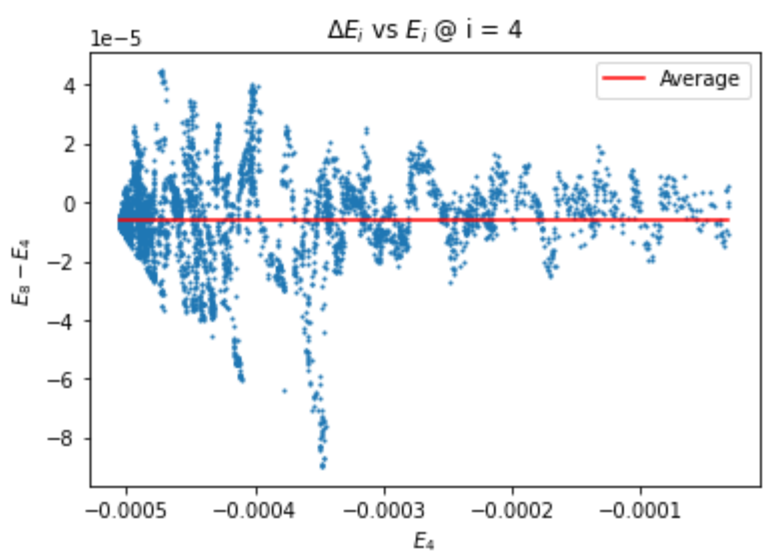

Total Energy Plots for the 5 stars
Total Potential Energy over time
Total Kinetic Energy over time
Total Energy K+W over time
Virial Ratio |K/W| over time
ΣW
ΣK
ΣE

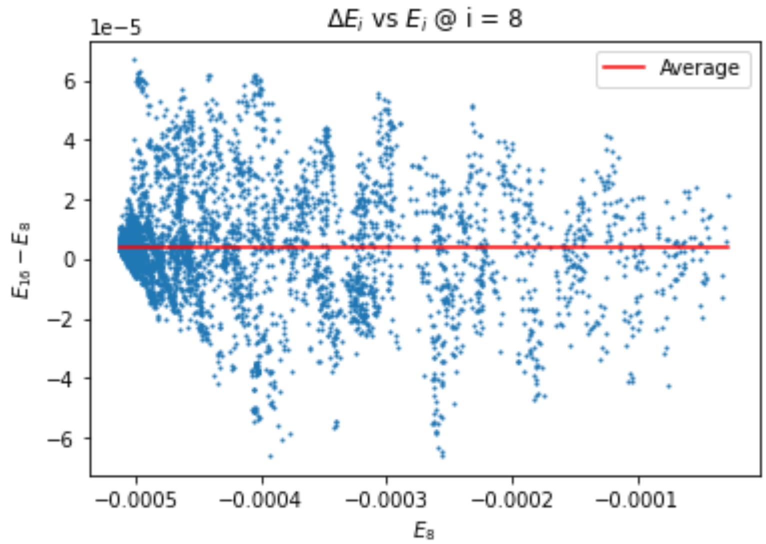

Energy Plots for Every Star, at Snapshot times/indices
Potential Energy over time
Kinetic Energy over time
Total Energy K+W over time
Virial Ratio |K/W| over time
ΣK
ΣE

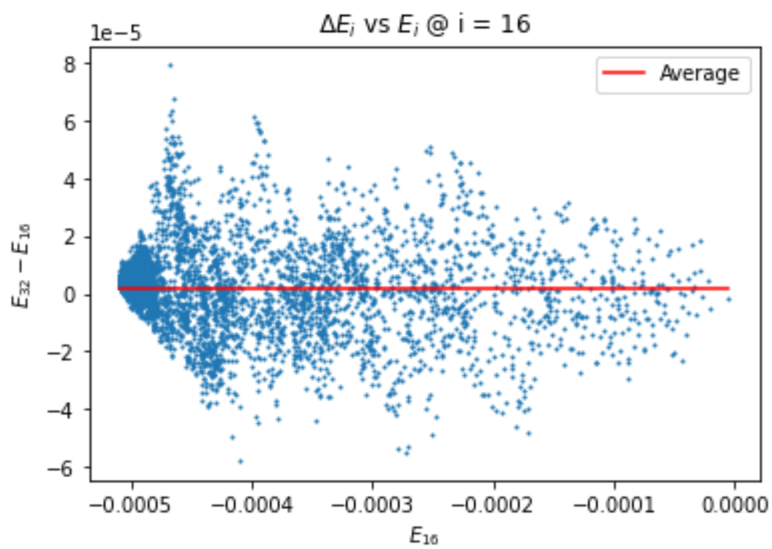

ΔE_i vs E_i @ i = 0
Average
E_1 − E_0
E_0

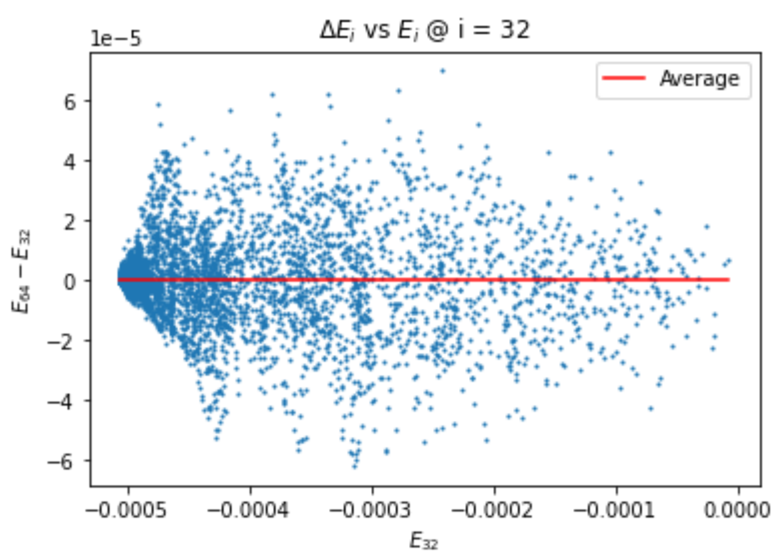

ΔE_i vs E_i @ i = 1
1e−5
Average
$E_2 - E_1$
E_1
2
1
0
−1
−2
−0.0005
−0.0004
−0.0003
−0.0002
−0.0001

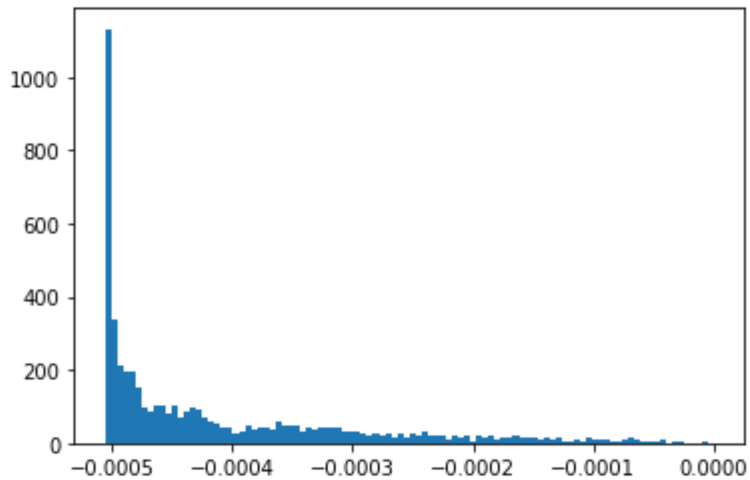

ΔE_i vs E_i @ i = 2
1e−5
Average
$E_4 - E_2$
E_2
3
2
1
0
−1
−2
−0.0005
−0.0004
−0.0003
−0.0002
−0.0001

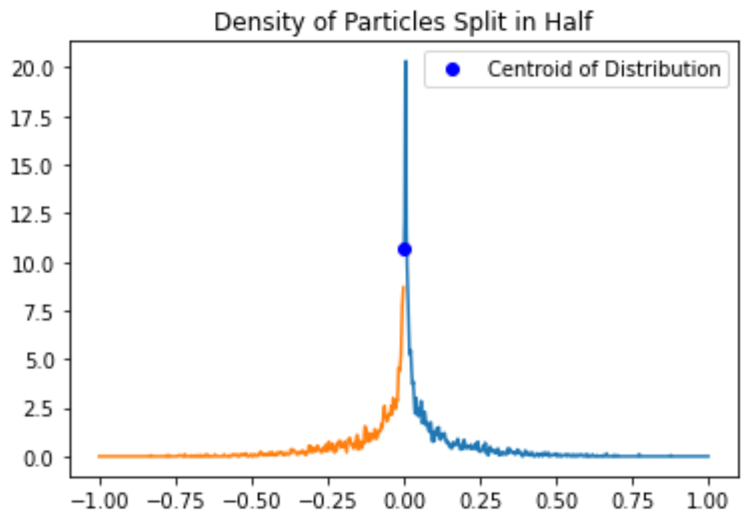

ΔE_i vs E_i @ i = 4
1e−5
Average
$E_8 - E_4$
E_4
4
2
0
−2
−4
−6
−8
−0.0005
−0.0004
−0.0003
−0.0002
−0.0001

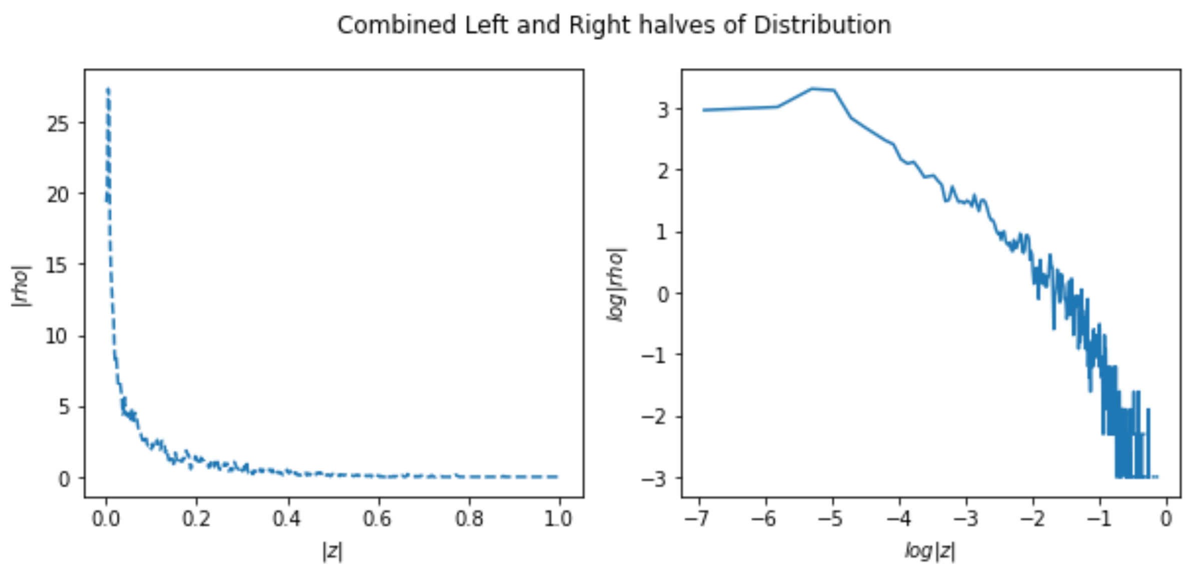

ΔE_i vs E_i @ i = 8
1e−5
Average
$E_{16} - E_8$
E_8
6
4
2
0
−2
−4
−6
−0.0005
−0.0004
−0.0003
−0.0002
−0.0001


ΔE_i vs E_i @ i = 16
1e−5
Average
$E_{32} - E_{16}$
E_{16}
8
6
4
2
0
−2
−4
−6
−0.0005
−0.0004
−0.0003
−0.0002
−0.0001
0.0000

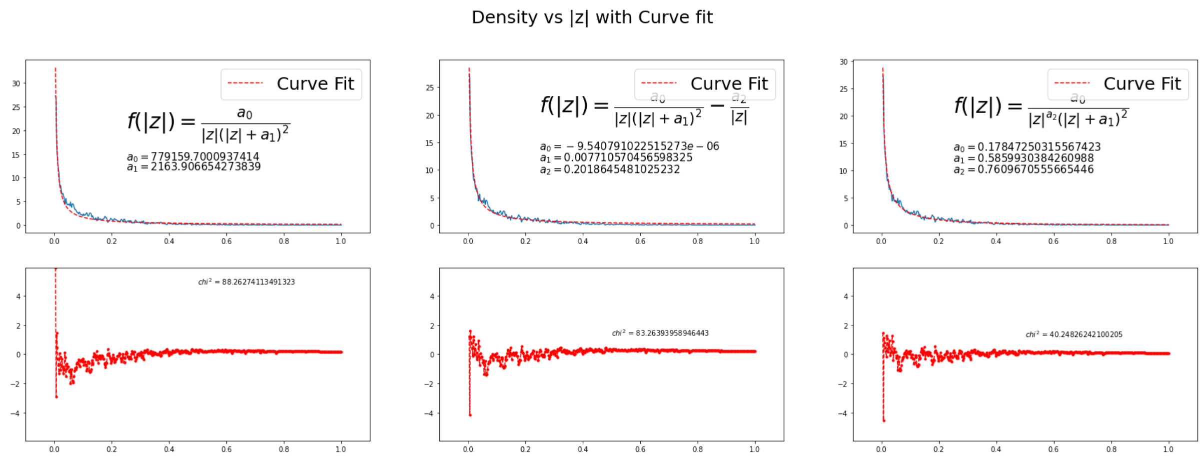

ΔE_i vs E_i @ i = 32
1e−5
Average
$E_{64} - E_{32}$
E_{32}
6
4
2
0
−2
−4
−6
−0.0005
−0.0004
−0.0003
−0.0002
−0.0001
0.0000

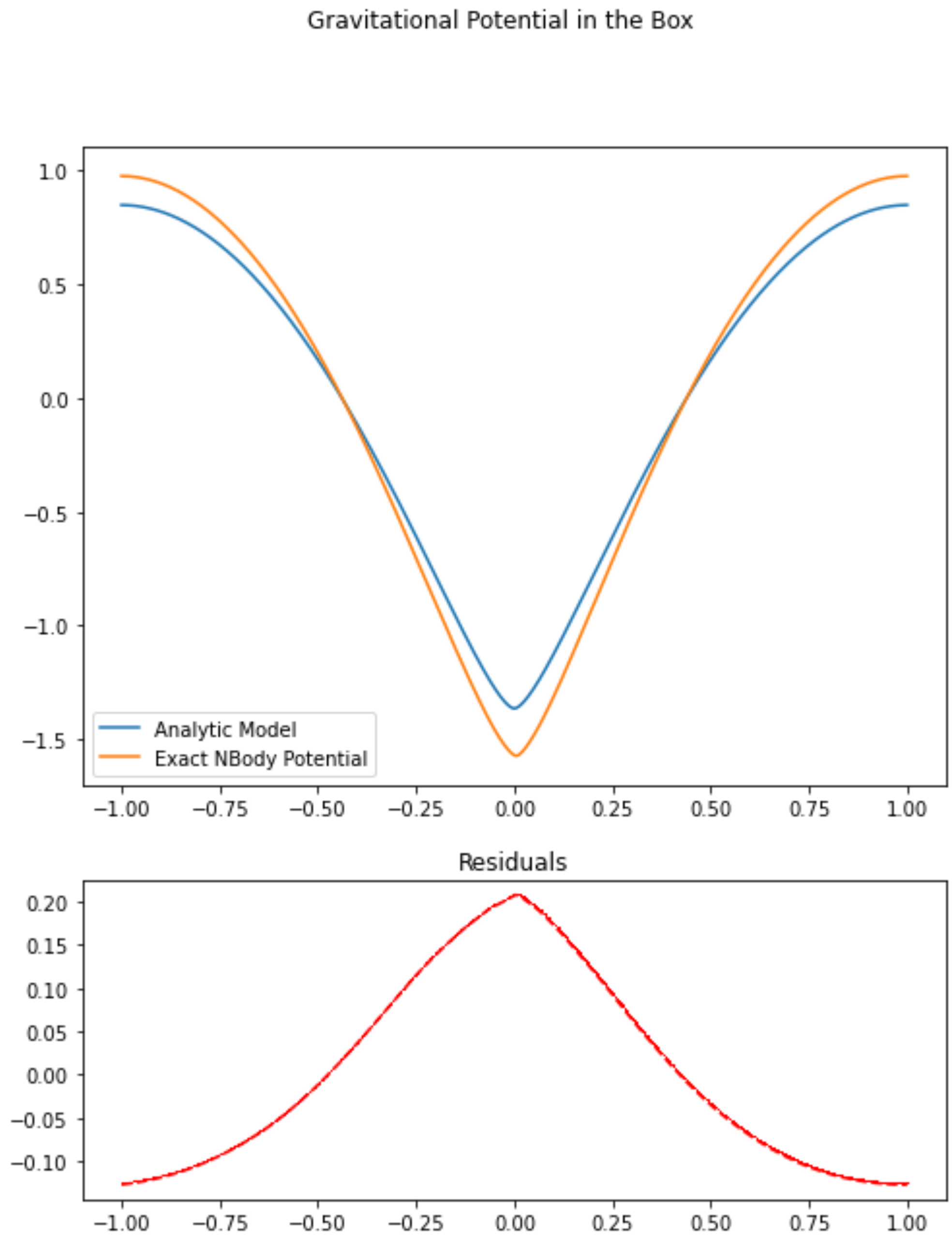




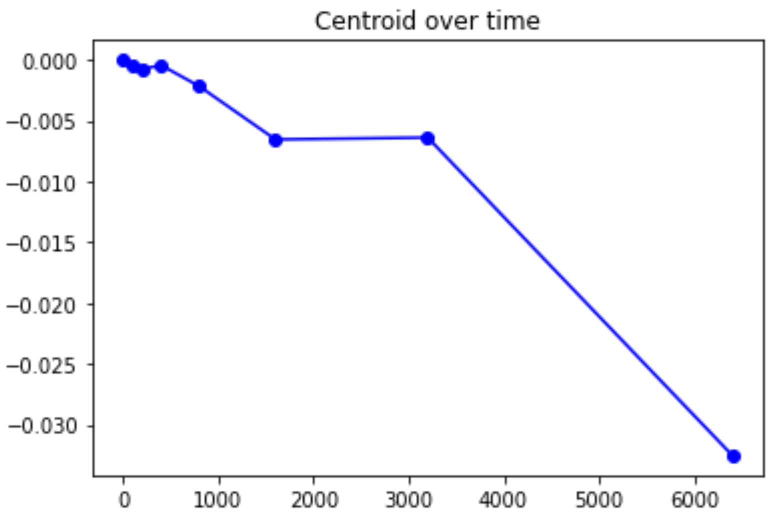


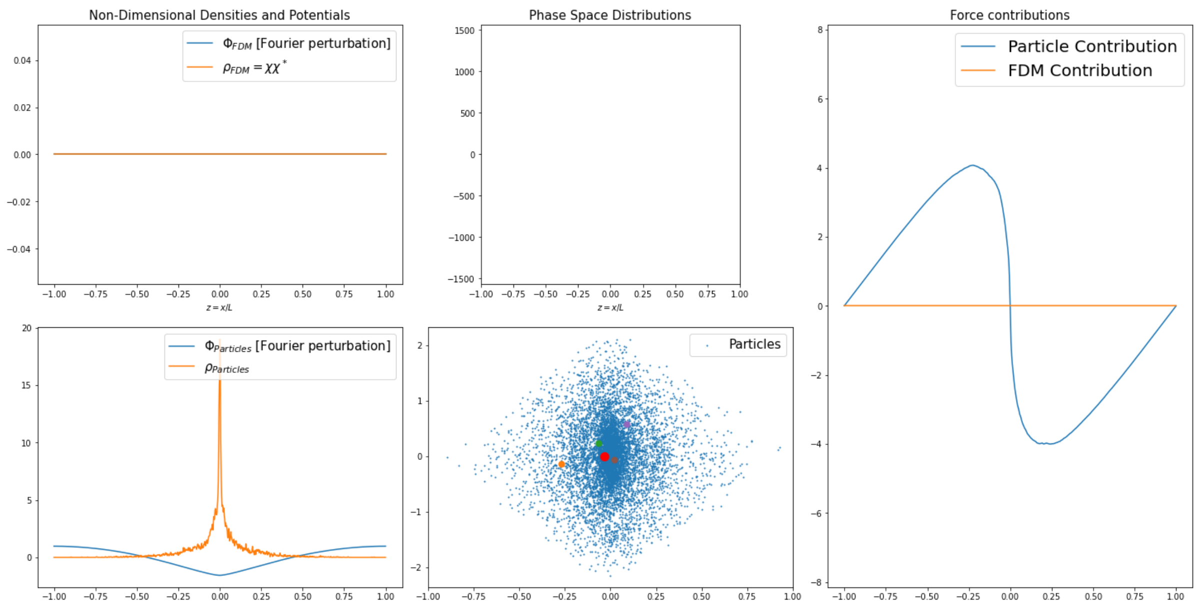

```
Check
Check
Check
#columns = 3
[[<AxesSubplot:> <AxesSubplot:> <AxesSubplot:>]
 [<AxesSubplot:> <AxesSubplot:> <AxesSubplot:>]]
```



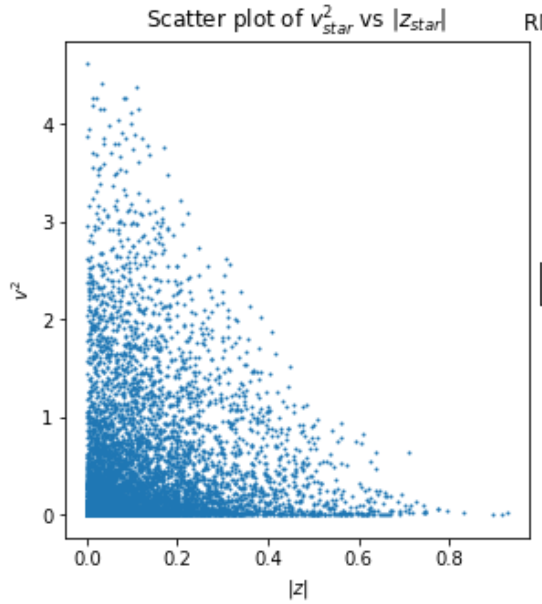


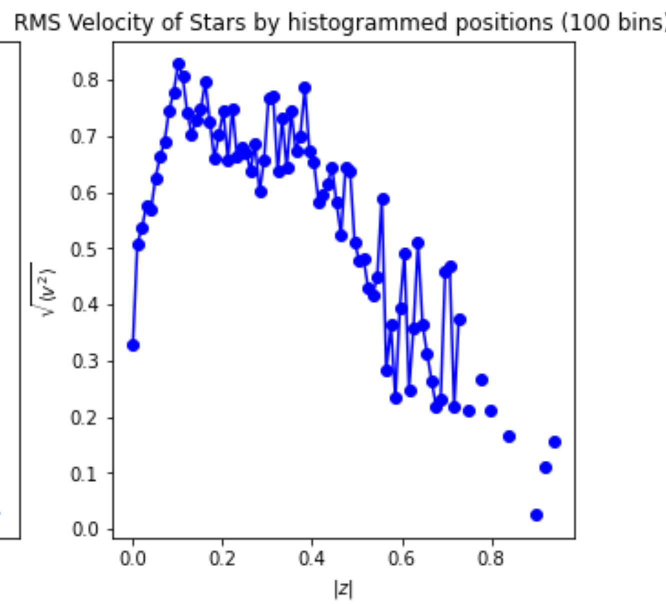

```
----------New Analysis--------
r = 1 mu = 1 Num_bosons = 0 sigma = 0.0001 Num_stars = 10000
```

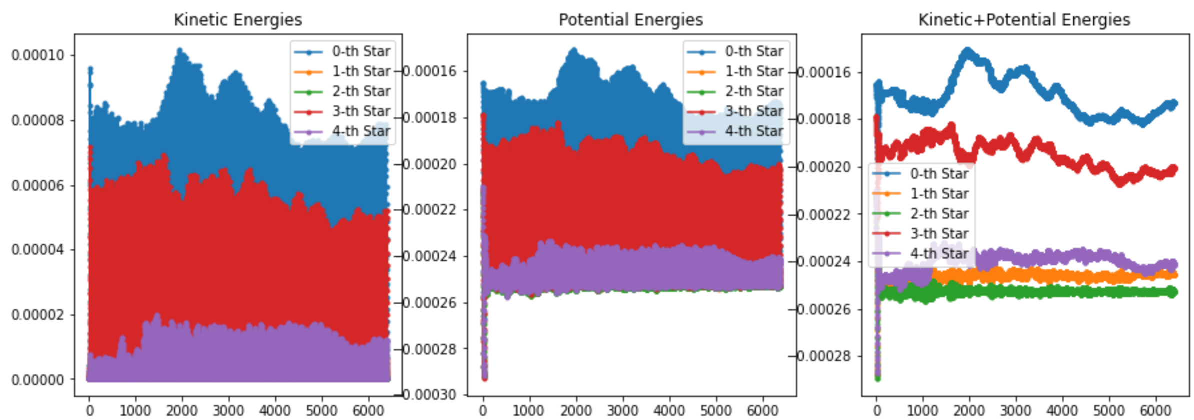


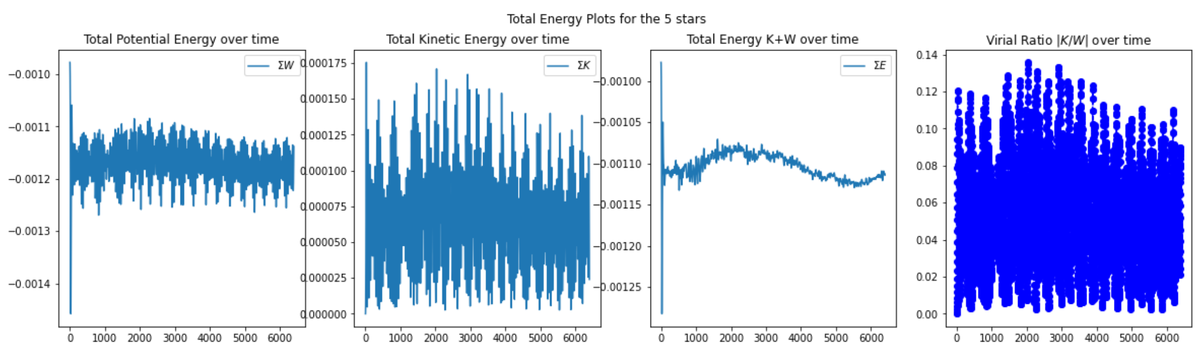


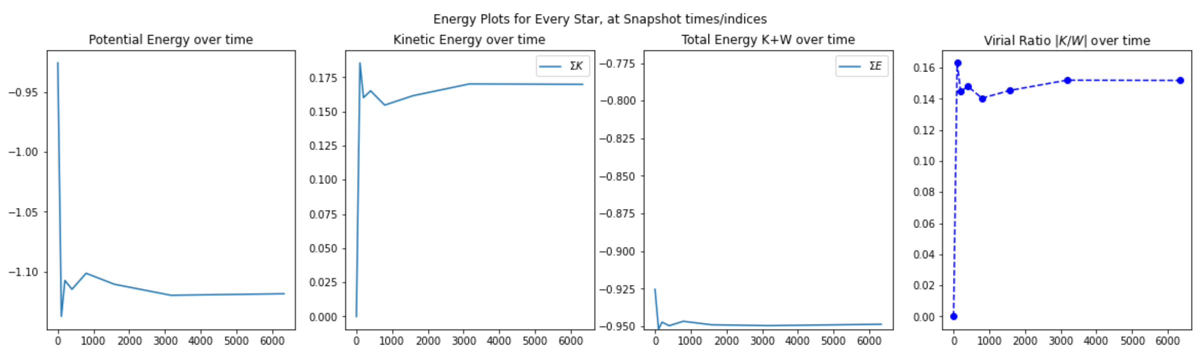

```
v_rms = 0.5846193062837114
z_rms = 0.1571204630460839
K_avg = 0.5*m*v_rms^2 = 0.17088986663982397 (m=1)
=> 2*K_avg = 0.34177973327964795
W_avg = 1571.204630460839
------------------
K_tot = 0.17088986663982333
K_avg = 1.7088986663982334e-05
W_tot = -0.15020247780808277
W_avg = -1.5020247780808277e-05
```

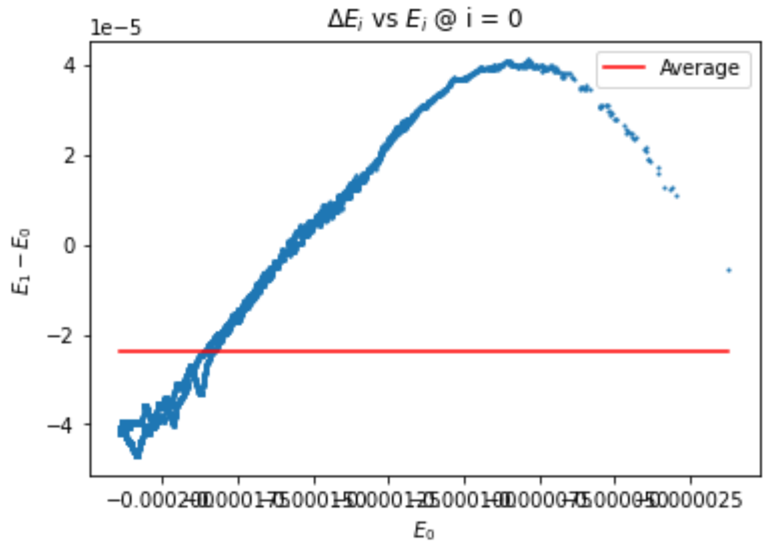

Scatter plot of v_{star}^2 vs $|z_{star}|$
v^2
$|z|$

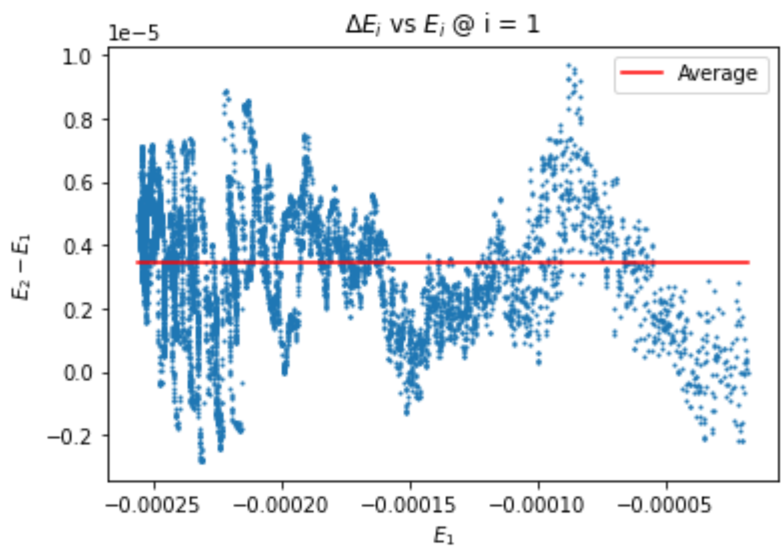

RMS Velocity of Stars by histogrammed positions (100 bins)
$\sqrt{\langle v^2 \rangle}$
$|z|$

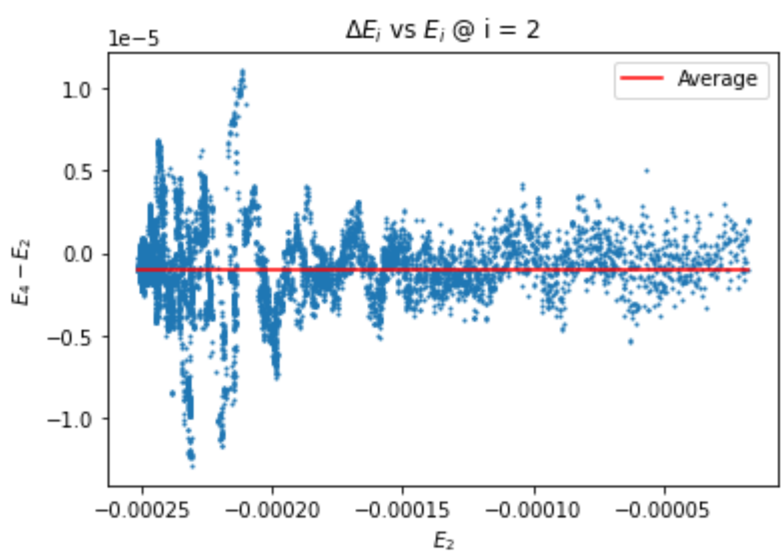

Kinetic Energies
Potential Energies
Kinetic+Potential Energies
0-th Star
1-th Star
2-th Star
3-th Star
4-th Star

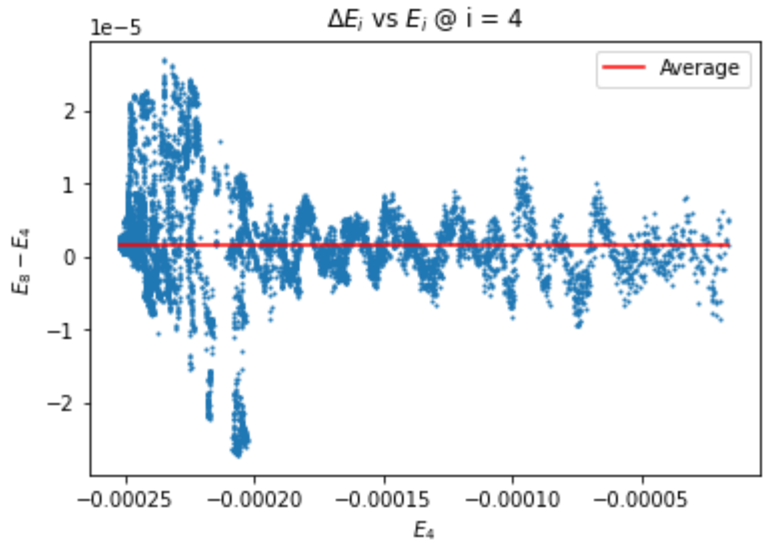

Total Energy Plots for the 5 stars
Total Potential Energy over time
ΣW
Total Kinetic Energy over time
ΣK
Total Energy K+W over time
ΣE
Virial Ratio $|K/W|$ over time


Energy Plots for Every Star, at Snapshot times/indices
Potential Energy over time
Kinetic Energy over time
ΣK
Total Energy K+W over time
ΣE
Virial Ratio $|K/W|$ over time

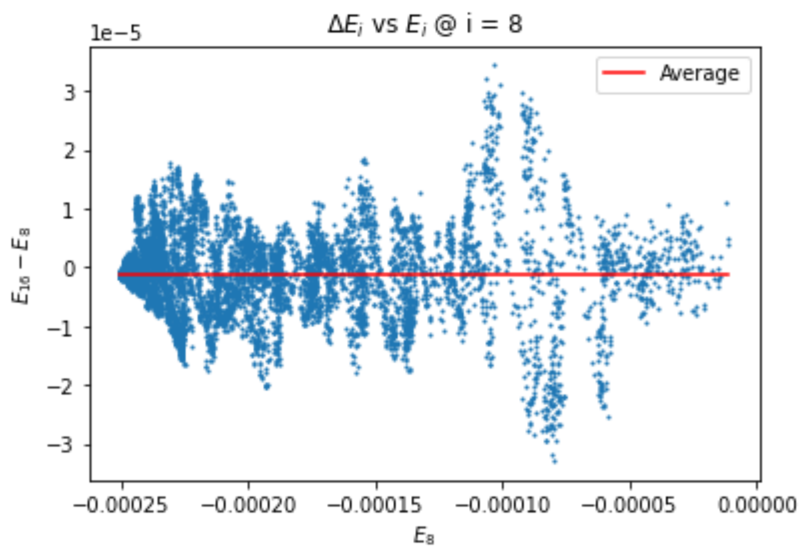

ΔE_i vs E_i @ i = 0
1e−5
Average
$E_1 - E_0$
E_0


ΔE_i vs E_i @ i = 1
1e−5
Average
$E_2 - E_1$
E_1

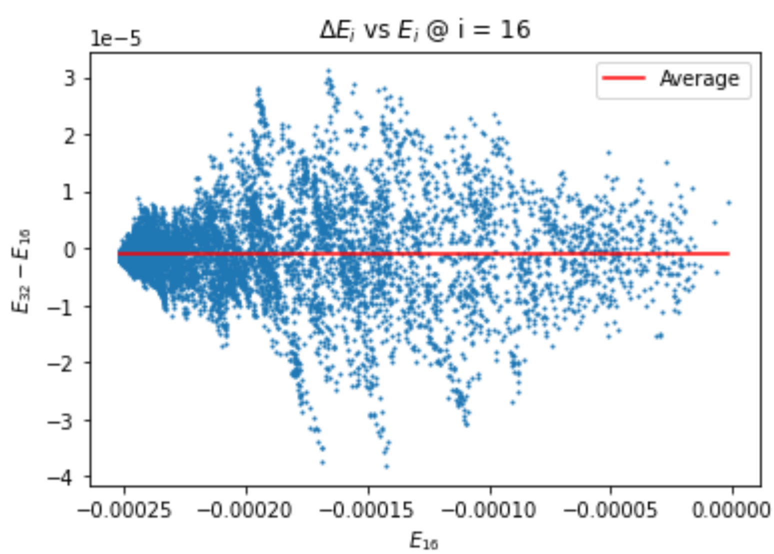

ΔE_i vs E_i @ i = 2
1e−5
Average
$E_4 - E_2$
E_2



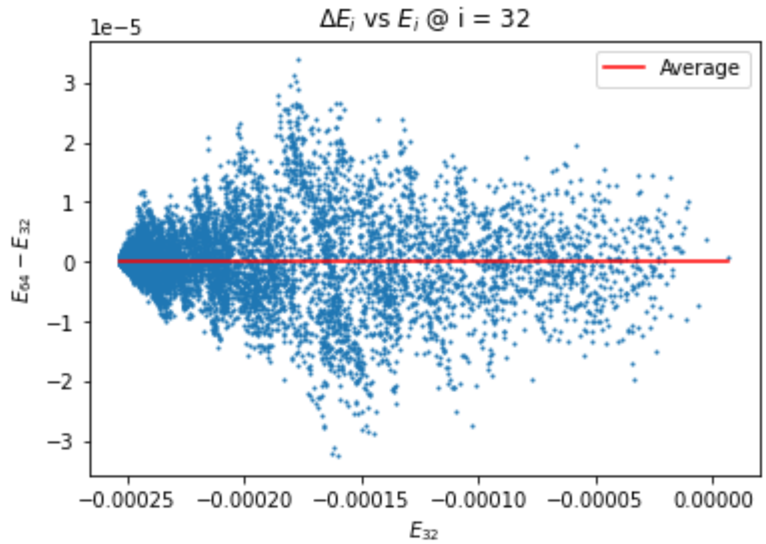
ΔE_i vs E_i @ i = 4
1e−5
Average
$E_8 - E_4$
E_4

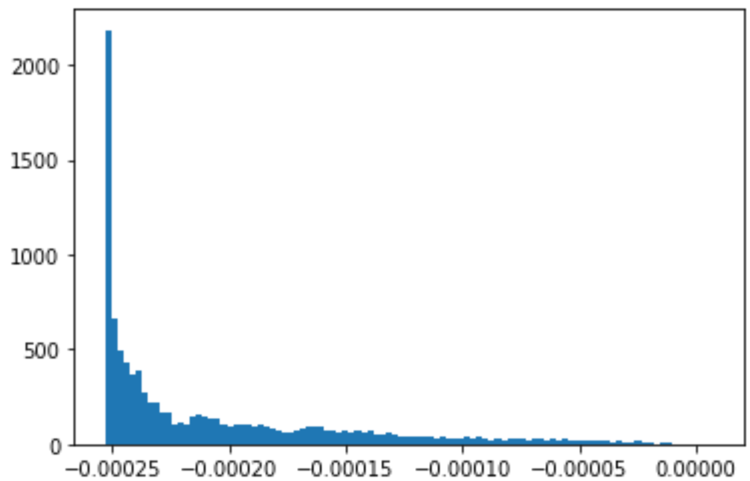


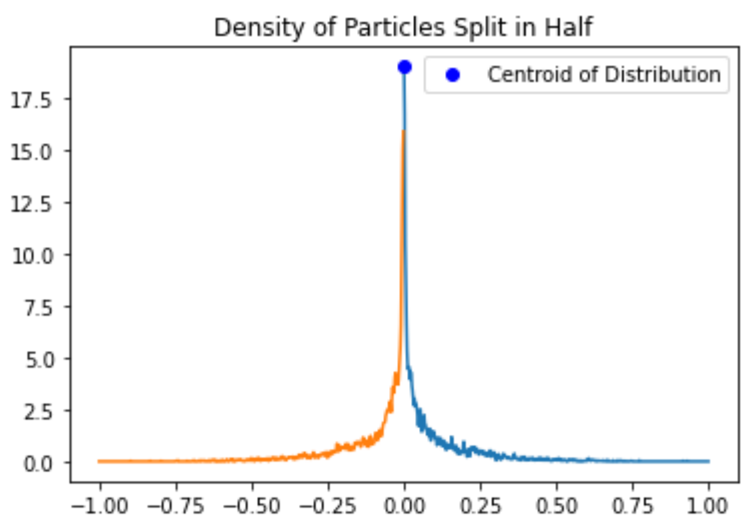
ΔE_i vs E_i @ i = 8
1e−5
Average
$E_{16} - E_8$
E_8

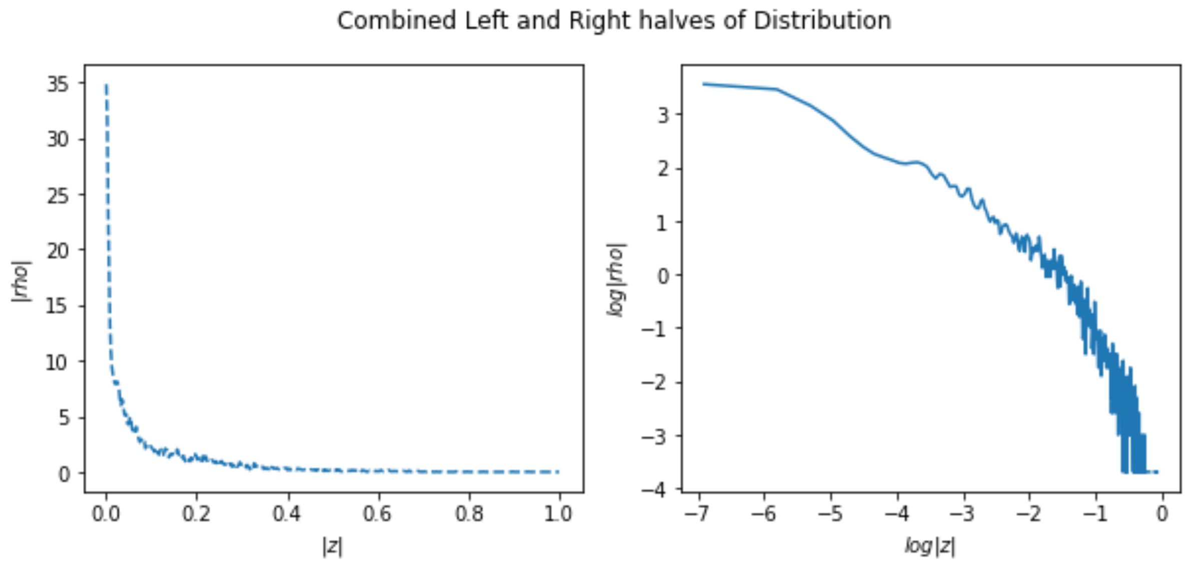


ΔE_i vs E_i @ i = 16
1e−5
Average
$E_{32} - E_{16}$
E_{16}

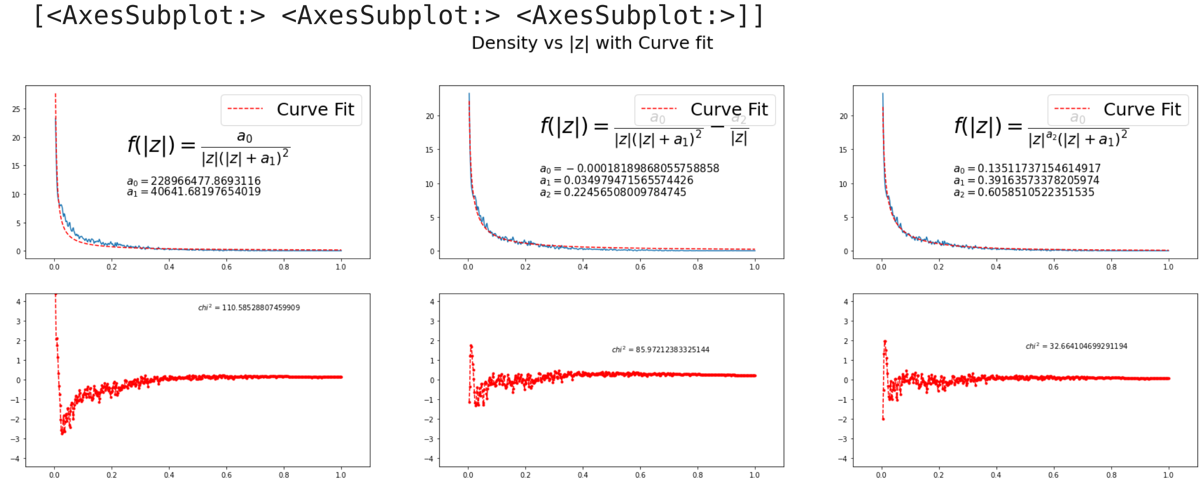

ΔE_i vs E_i @ i = 32
1e−5
Average
$E_{64} - E_{32}$
E_{32}
−0.00025 −0.00020 −0.00015 −0.00010 −0.00005 0.00000

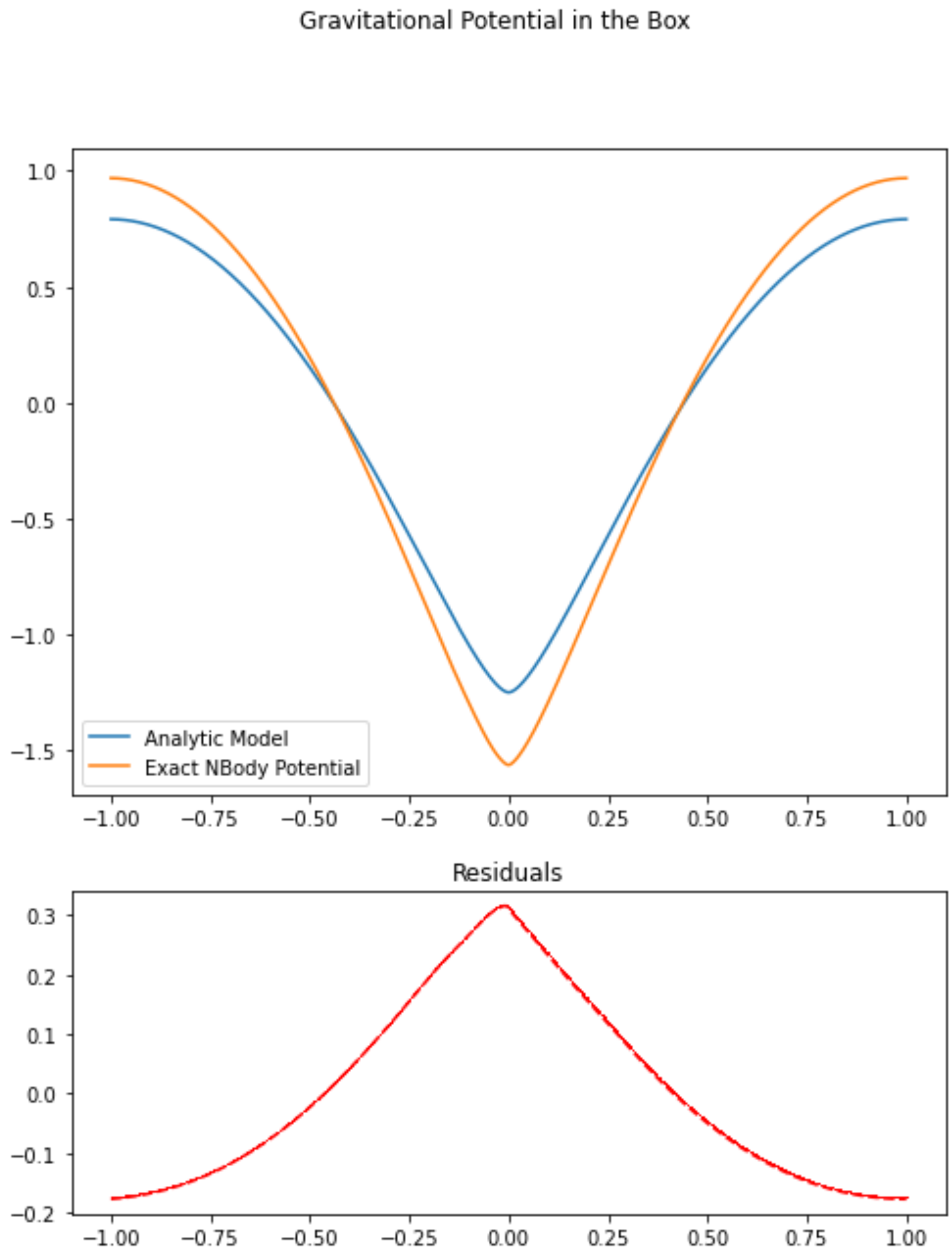

2000
1500
1000
500
0
−0.00025 −0.00020 −0.00015 −0.00010 −0.00005 0.00000


Density of Particles Split in Half
Centroid of Distribution
−1.00 −0.75 −0.50 −0.25 0.00 0.25 0.50 0.75 1.00

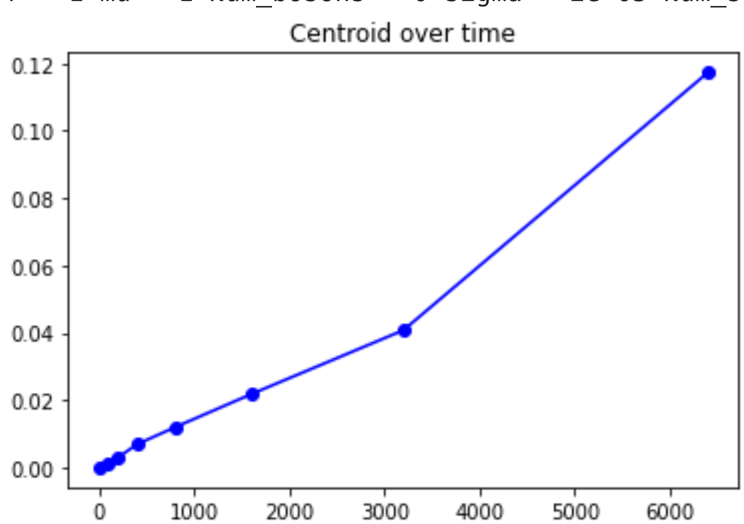


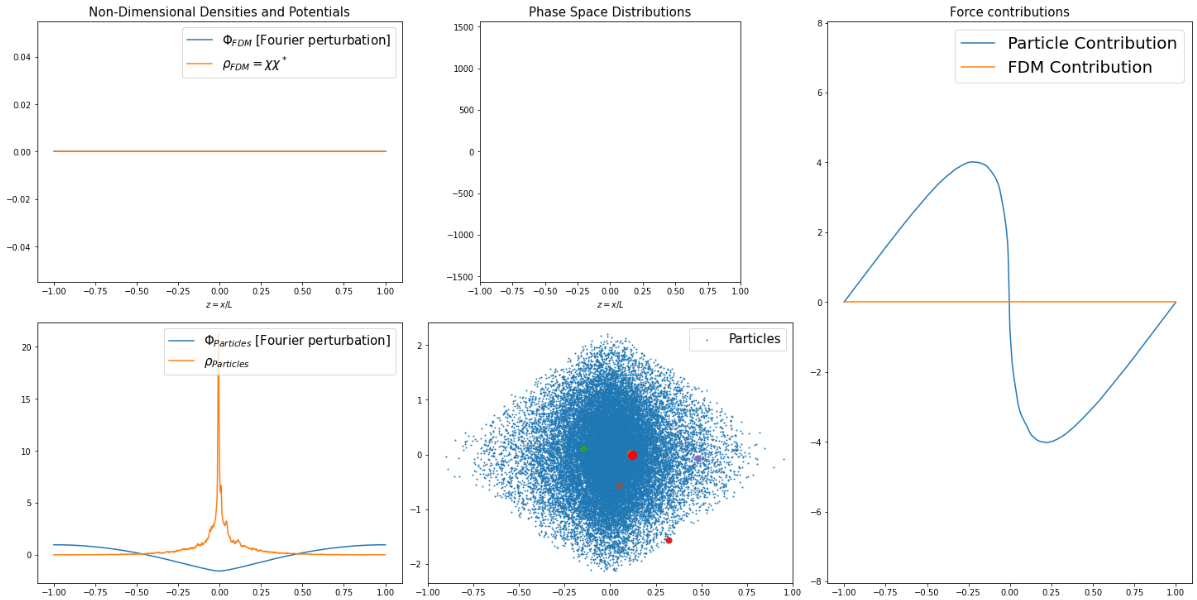

```
Check
Check
Check
#columns = 3
[[<AxesSubplot:> <AxesSubplot:> <AxesSubplot:>]
 [<AxesSubplot:> <AxesSubplot:> <AxesSubplot:>]]
```



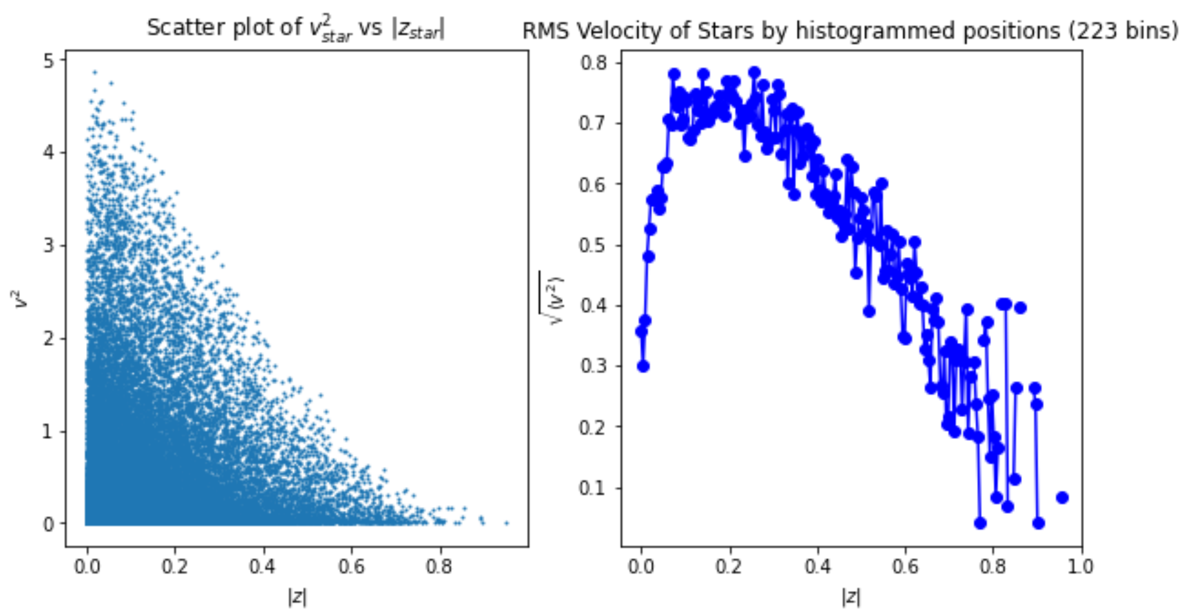


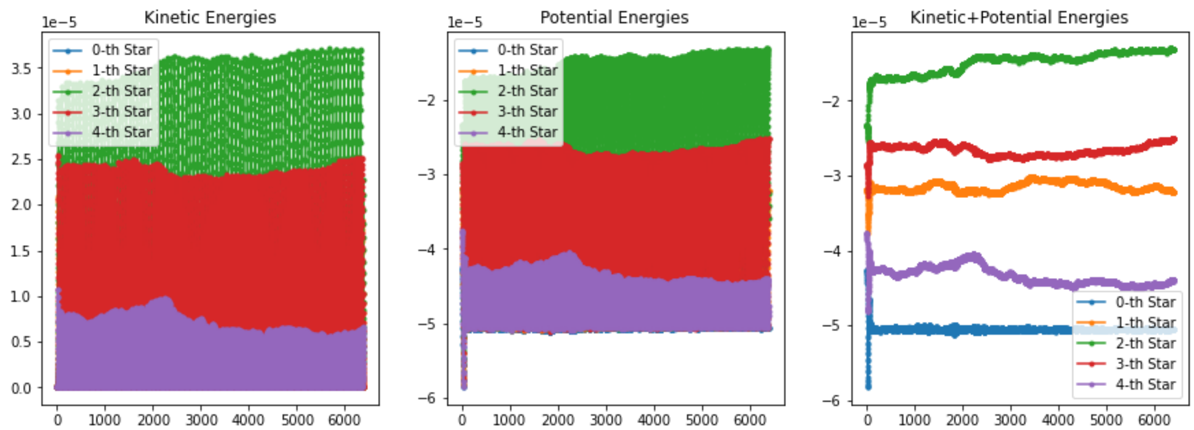

```
----------New Analysis--------
r = 1 mu = 1 Num_bosons = 0 sigma = 2e-05 Num_stars = 50000
```

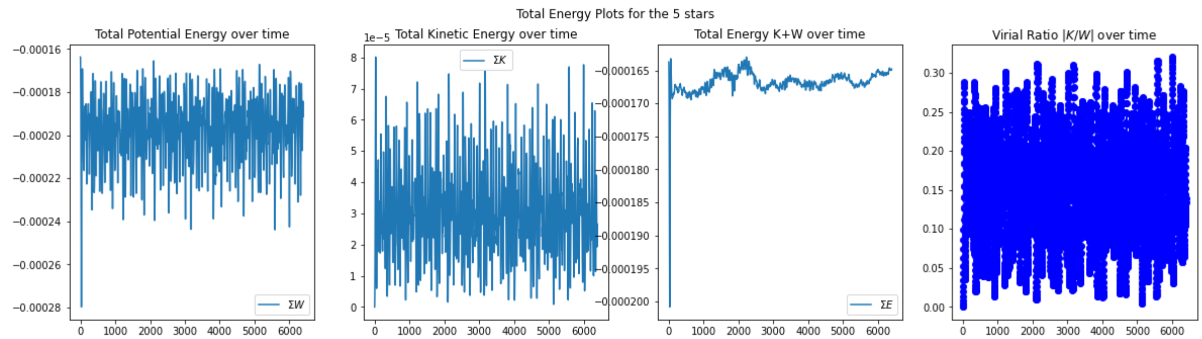


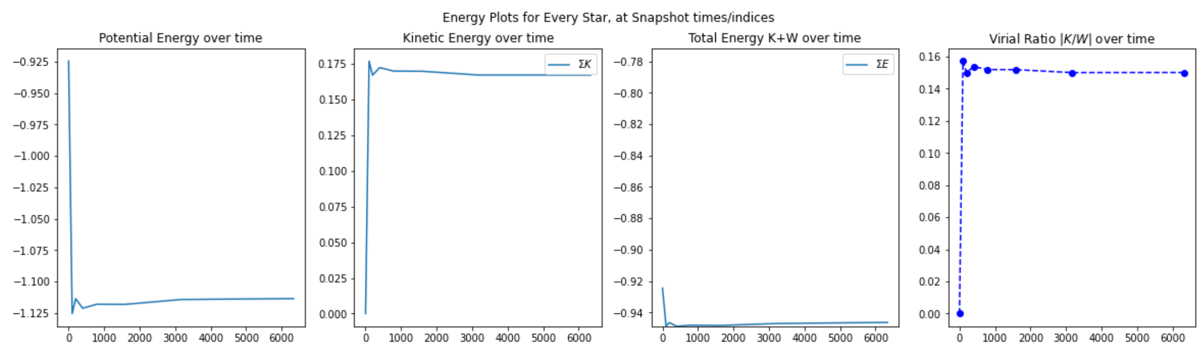


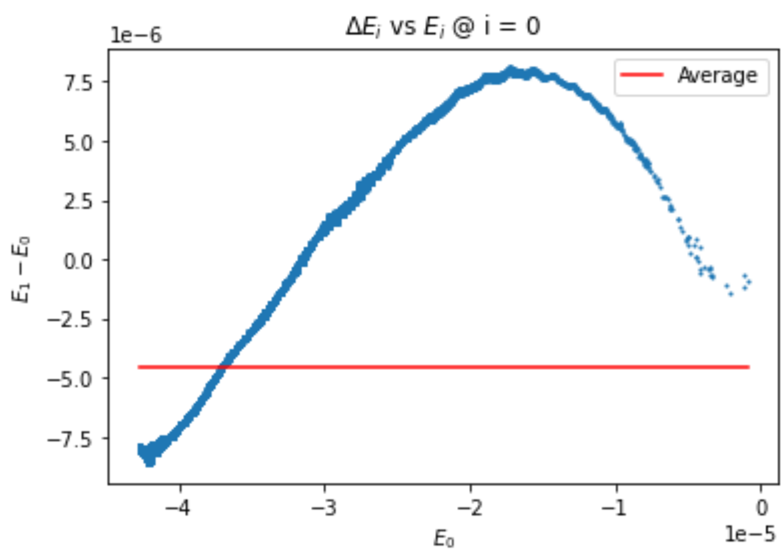

```
v_rms = 0.580363499769716
z_rms = 0.1584492073892115
K_avg = 0.5*m*v_rms^2 = 0.16841089593247657 (m=1)
=> 2*K_avg = 0.33682179186495315
W_avg = 7922.460369460575
------------------
K_tot = 0.16841089593247574
K_avg = 3.368217918649515e-06
W_tot = -0.1530561422080706
W_avg = -3.061122844161412e-06
```

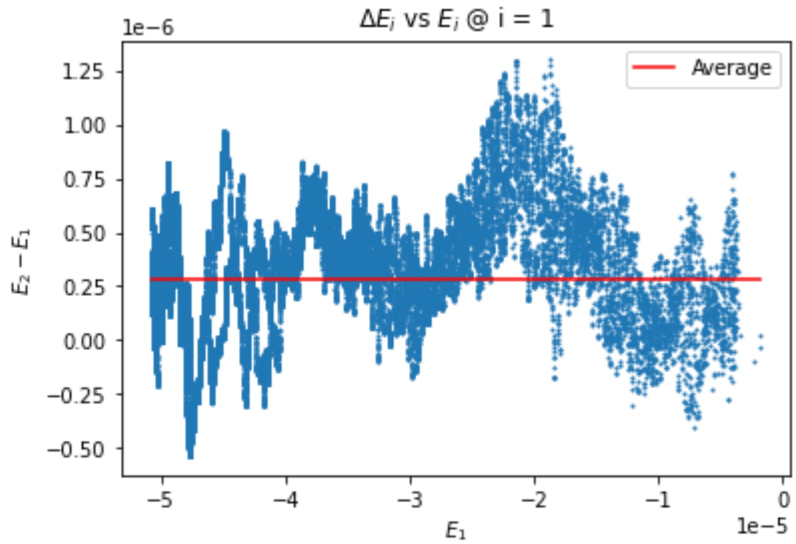



Kinetic Energies
Potential Energies
Kinetic+Potential Energies
0-th Star
1-th Star
2-th Star
3-th Star
4-th Star

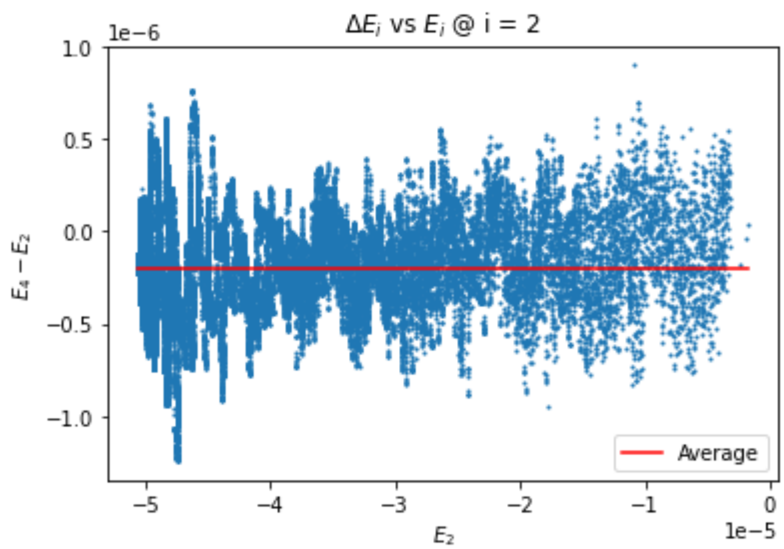

Total Energy Plots for the 5 stars
Total Potential Energy over time
Total Kinetic Energy over time
Total Energy K+W over time
Virial Ratio |K/W| over time
ΣW
ΣK
ΣE


Energy Plots for Every Star, at Snapshot times/indices
Potential Energy over time
Kinetic Energy over time
Total Energy K+W over time
Virial Ratio |K/W| over time
ΣK
ΣE

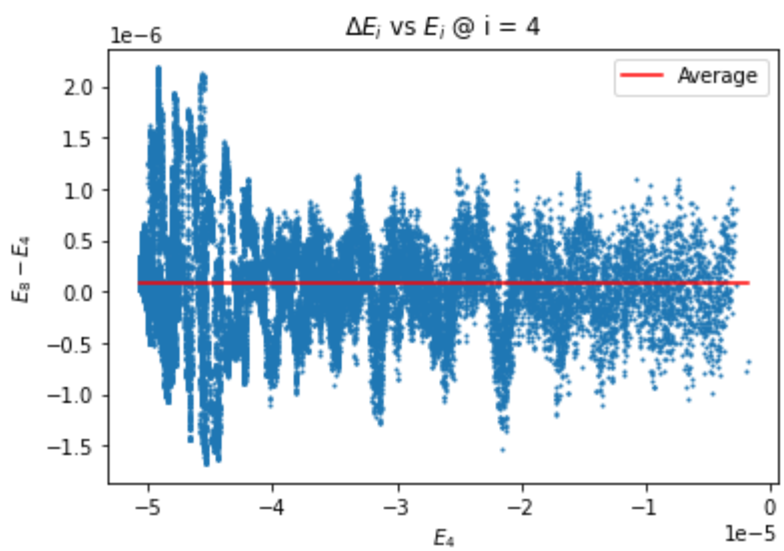

ΔE_i vs E_i @ i = 0
Average
E_1 − E_0
E_0



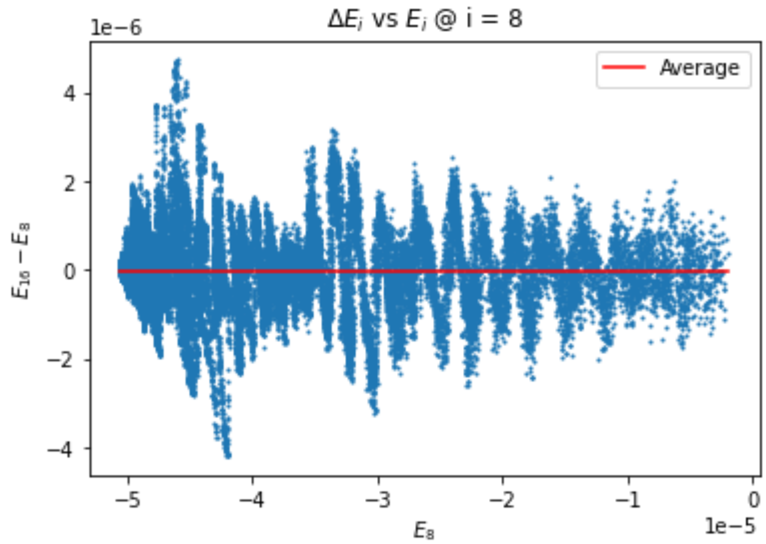
ΔE_i vs E_i @ i = 1
1e−6
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
$E_2 - E_1$
Average
−5
−4
−3
−2
−1
0
1e−5
E_1

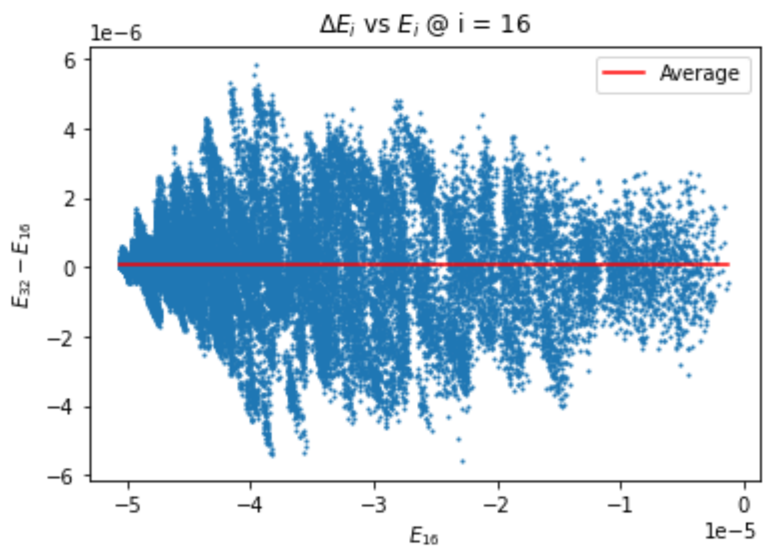


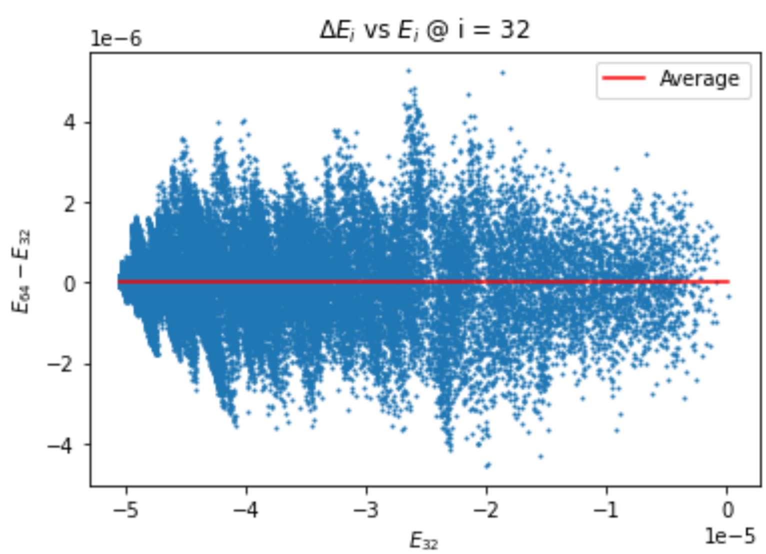
ΔE_i vs E_i @ i = 2
1e−6
1.0
0.5
0.0
−0.5
−1.0
$E_4 - E_2$
Average
−5
−4
−3
−2
−1
0
1e−5
E_2

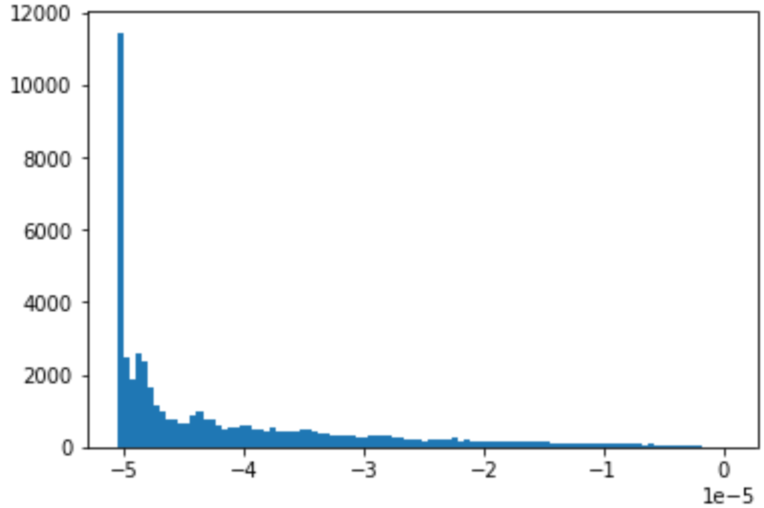


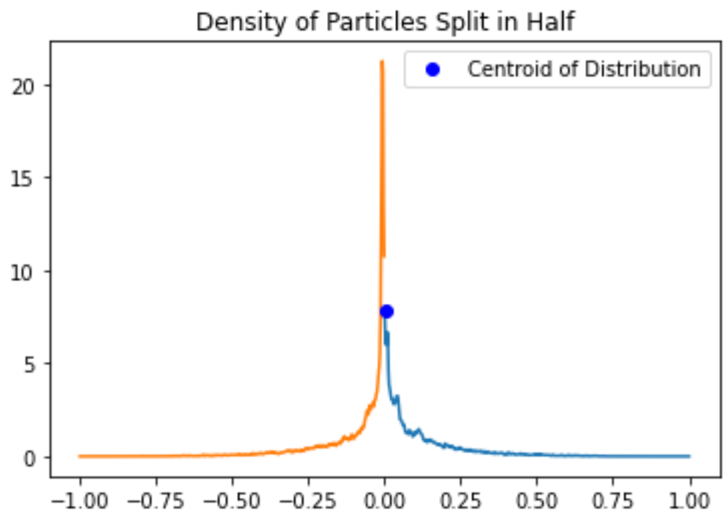
ΔE_i vs E_i @ i = 4
1e−6
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
$E_8 - E_4$
Average
−5
−4
−3
−2
−1
0
1e−5
E_4

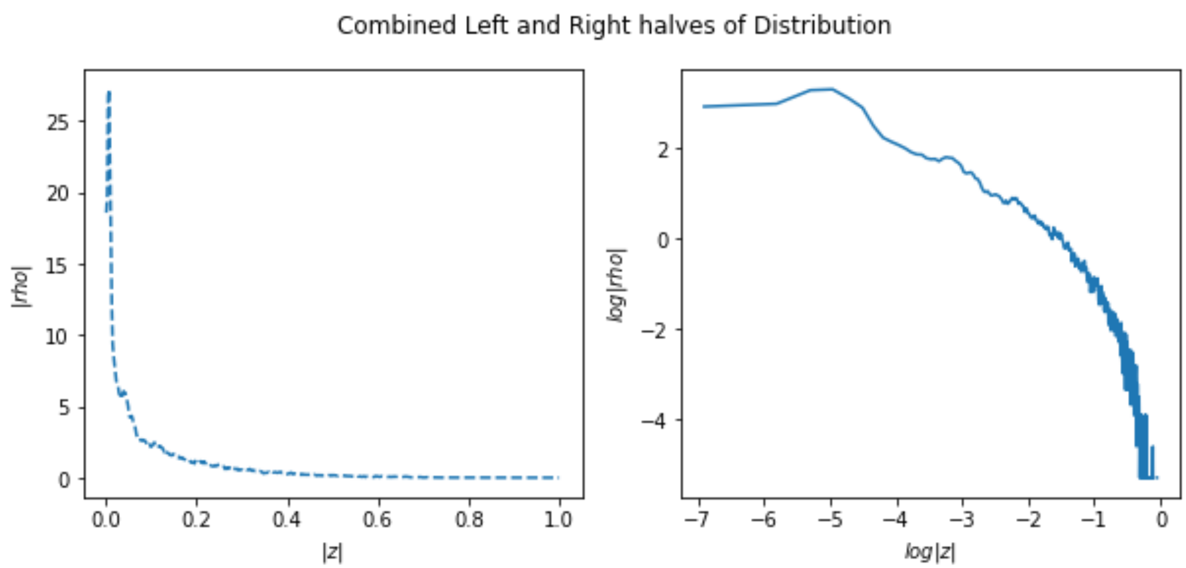

ΔE_i vs E_i @ i = 8
1e−6
Average
$E_{16} - E_8$
E_8
1e−5


ΔE_i vs E_i @ i = 16
1e−6
Average
$E_{32} - E_{16}$
E_{16}
1e−5


ΔE_i vs E_i @ i = 32
1e−6
Average
$E_{64} - E_{32}$
E_{32}
1e−5

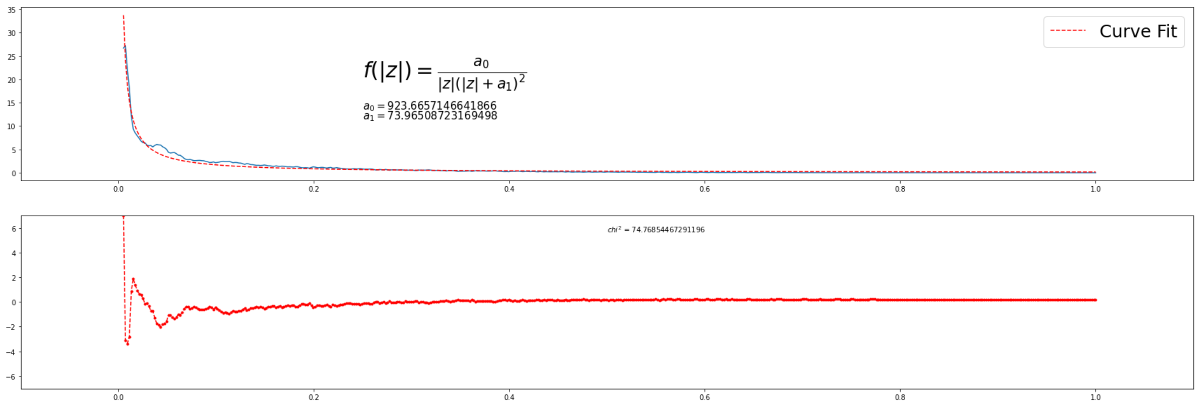




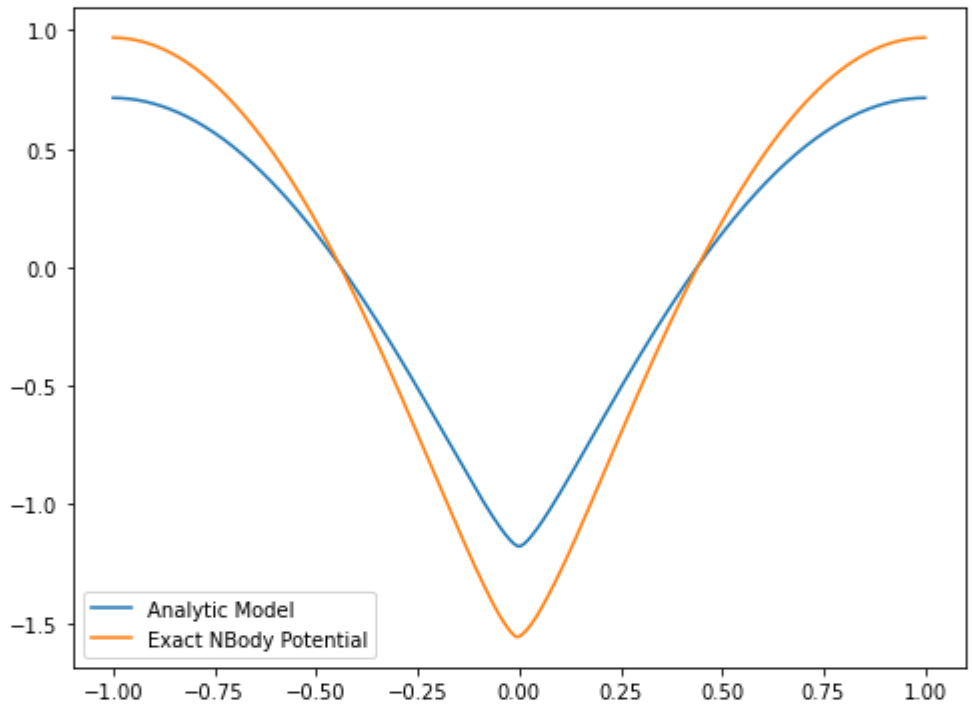



```
Check
#columns = 1
[<AxesSubplot:> <AxesSubplot:>]
```

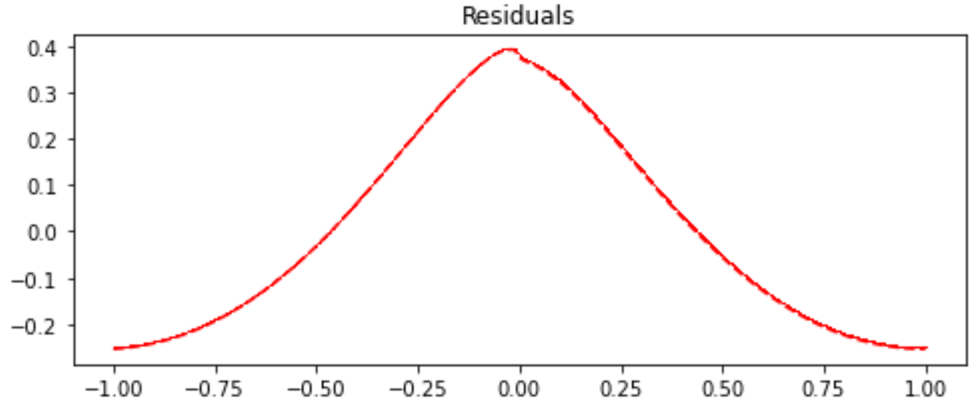
Density vs |z| with Curve fit



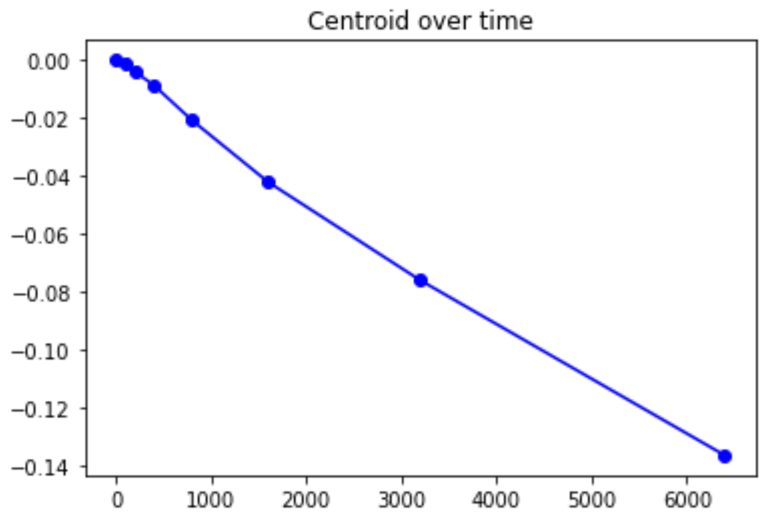
Gravitational Potential in the Box

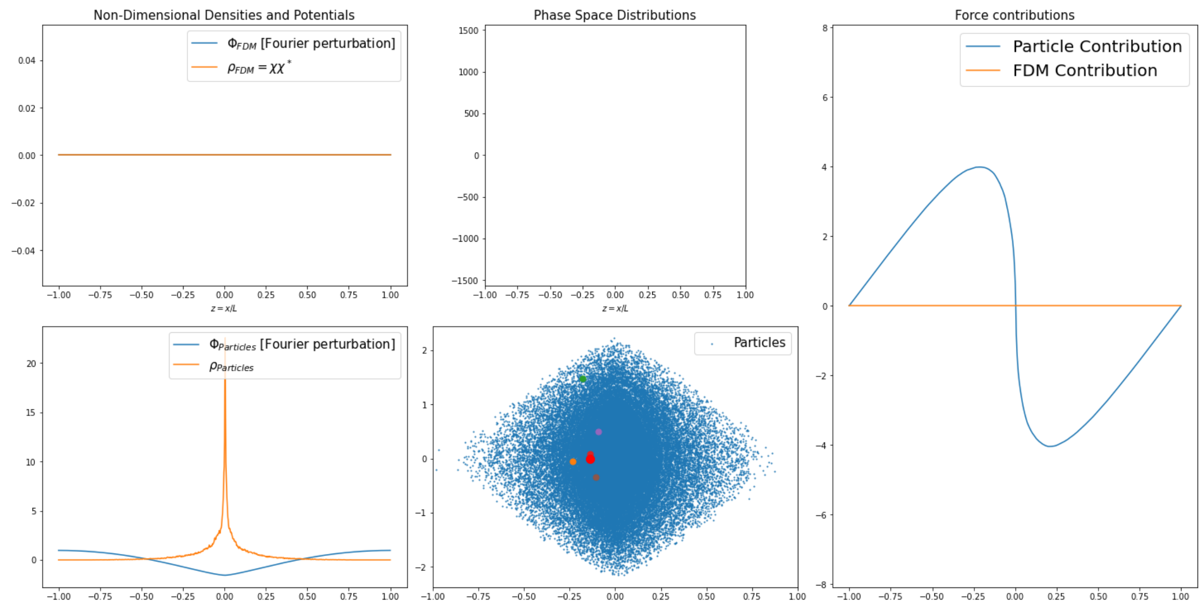


Residuals

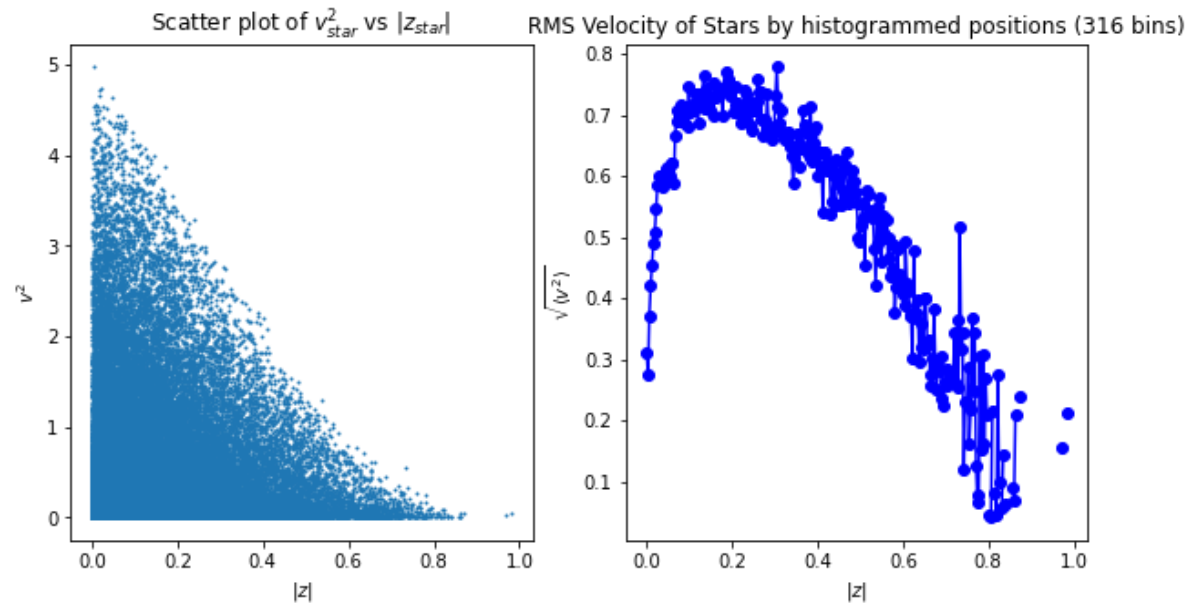


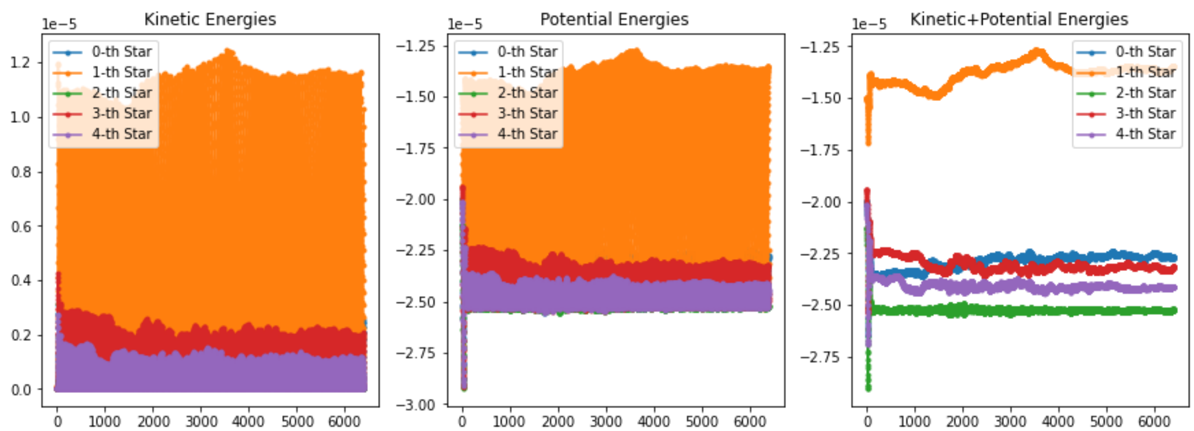
```
----------New Analysis--------
r = 1 mu = 1 Num_bosons = 0 sigma = 1e-05 Num_stars = 100000
```

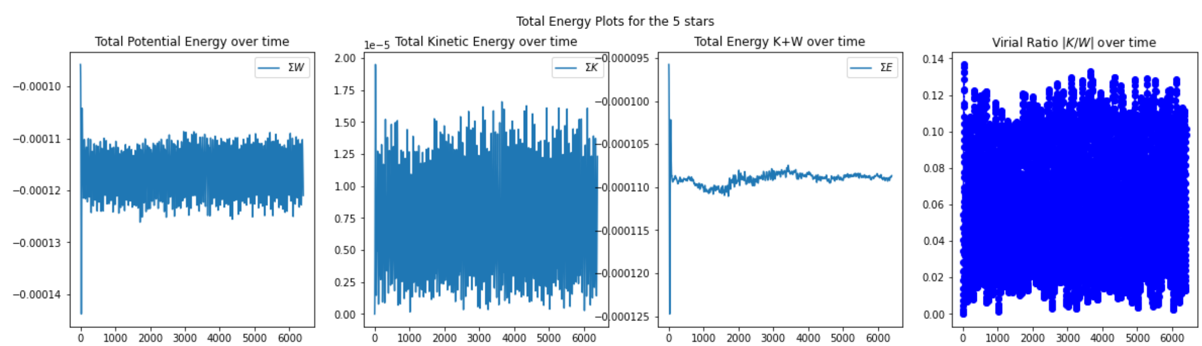


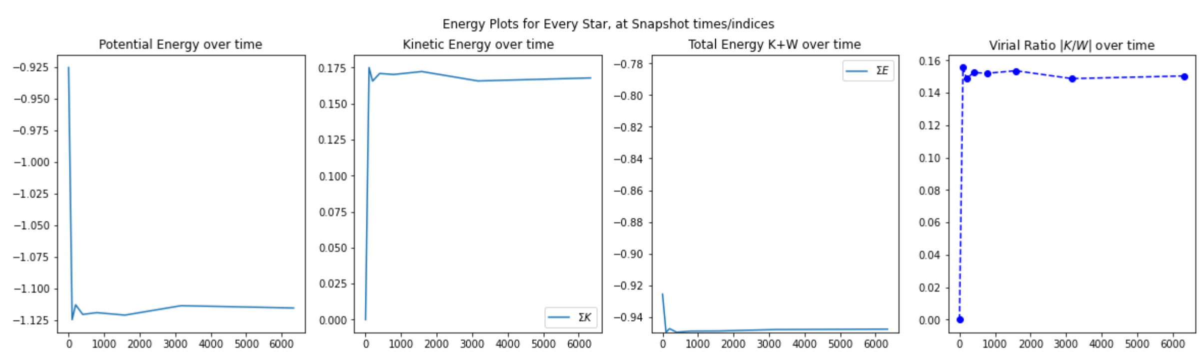


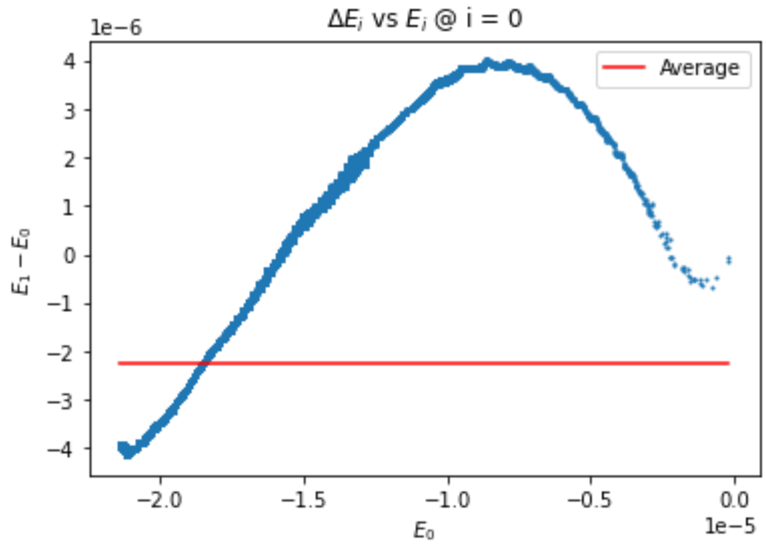

```
v_rms = 0.5755103792943796
z_rms = 0.15924094420910595
K_avg = 0.5*m*v_rms^2 = 0.16560609833778037 (m=1)
=> 2*K_avg = 0.33121219667556073
W_avg = 15924.094420910595
------------------
K_tot = 0.16560609833777962
K_avg = 1.6560609833777961e-06
W_tot = -0.15448555885359844
W_avg = -1.5448555885359843e-06
```

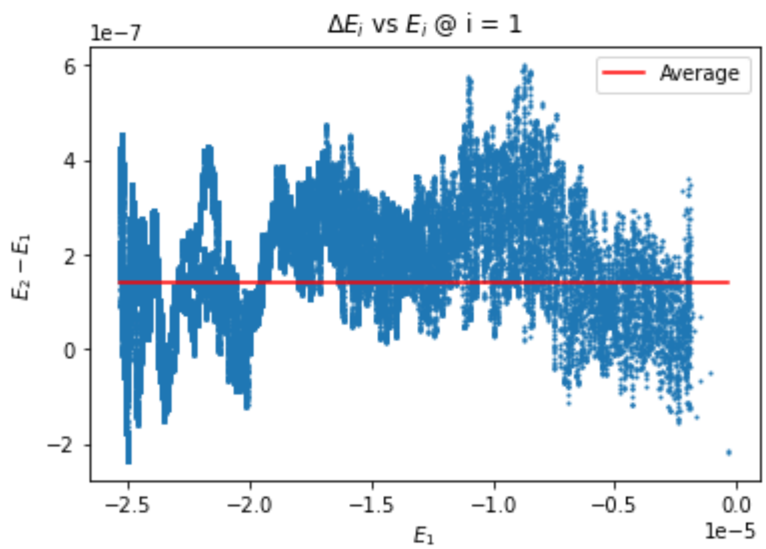

Scatter plot of v_{star}^2 vs $|z_{star}|$
v^2
$|z|$
RMS Velocity of Stars by histogrammed positions (316 bins)
$\sqrt{\langle v^2 \rangle}$
$|z|$

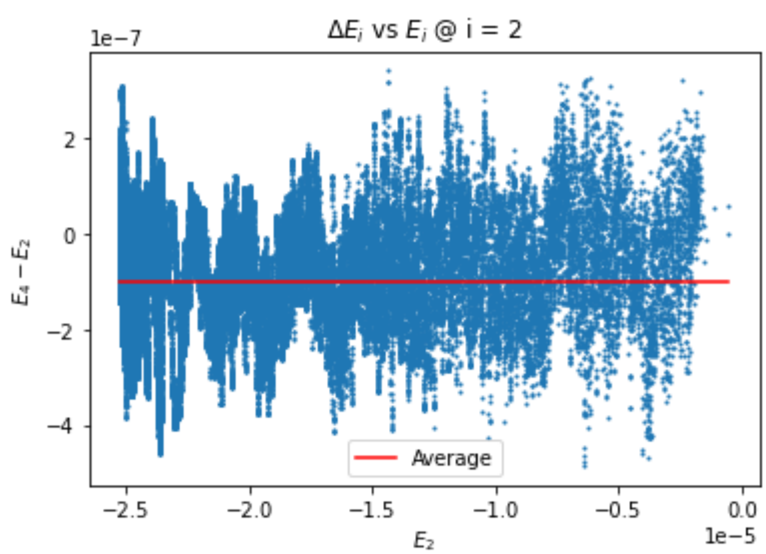

Kinetic Energies
Potential Energies
Kinetic+Potential Energies
1e−5
0-th Star
1-th Star
2-th Star
3-th Star
4-th Star

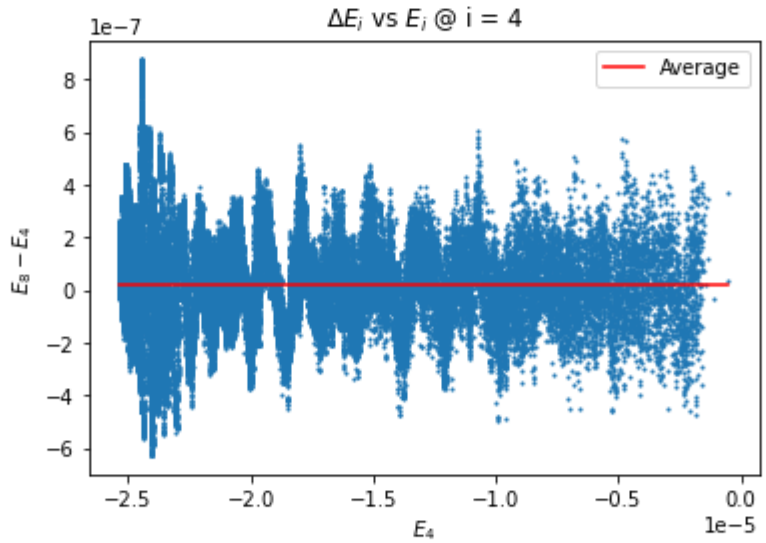

Total Energy Plots for the 5 stars
Total Potential Energy over time
ΣW
Total Kinetic Energy over time
ΣK
Total Energy K+W over time
ΣE
Virial Ratio $|K/W|$ over time

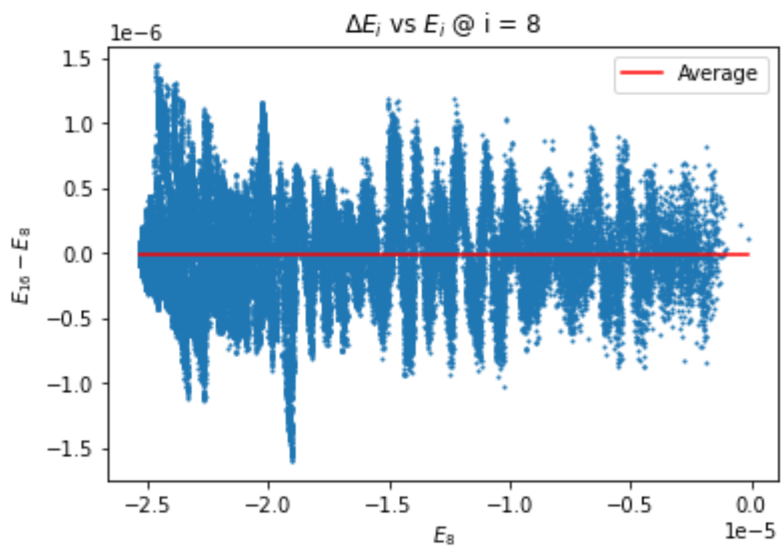

Energy Plots for Every Star, at Snapshot times/indices
Potential Energy over time
Kinetic Energy over time
ΣK
Total Energy K+W over time
ΣE
Virial Ratio $|K/W|$ over time

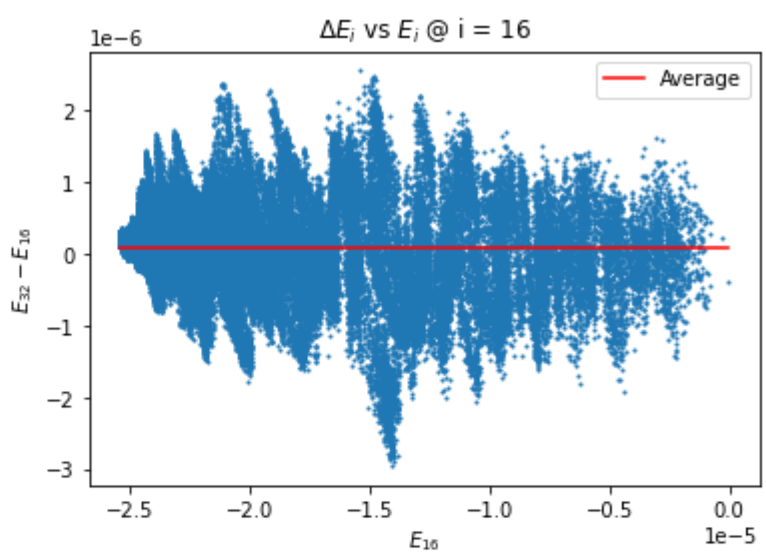

ΔE_i vs E_i @ i = 0
1e−6
Average
$E_1 - E_0$
E_0
1e−5

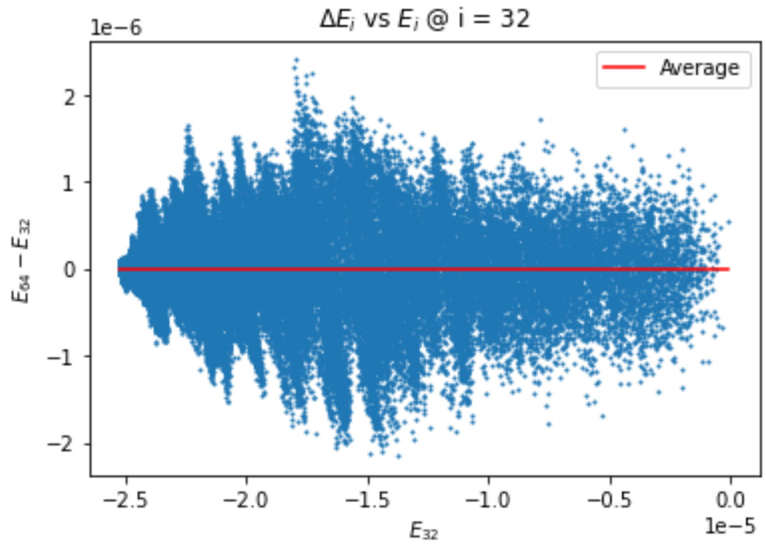

ΔE_i vs E_i @ i = 1
1e−7
Average
$E_2 - E_1$
E_1
1e−5


ΔE_i vs E_i @ i = 2
1e−7
Average
$E_4 - E_2$
E_2
1e−5

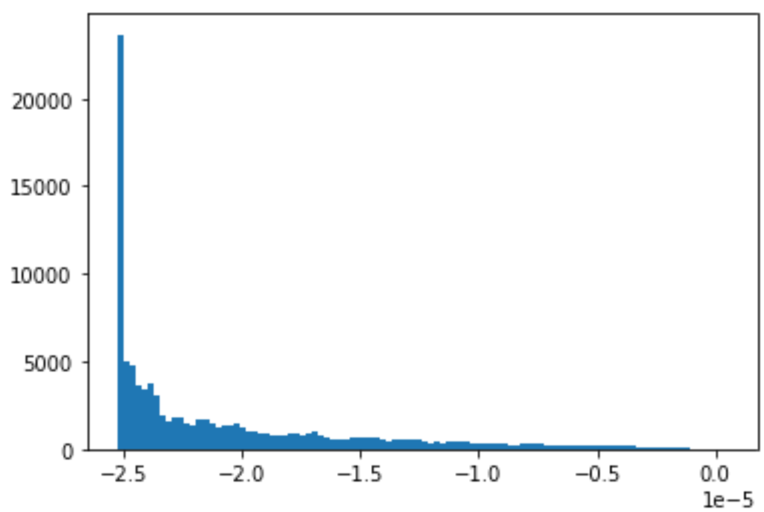

ΔE_i vs E_i @ i = 4
1e−7
Average
E_8 − E_4
E_4
1e−5


ΔE_i vs E_i @ i = 8
1e−6
Average
E_16 − E_8
E_8
1e−5

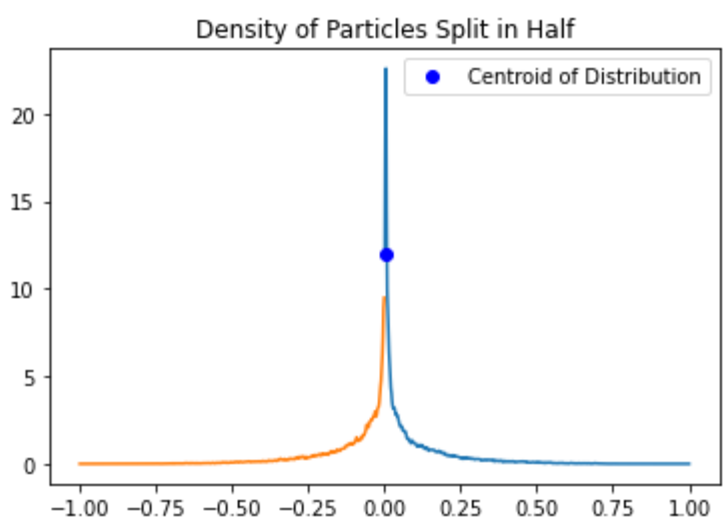

ΔE_i vs E_i @ i = 16
1e−6
Average
E_32 − E_16
E_16
1e−5



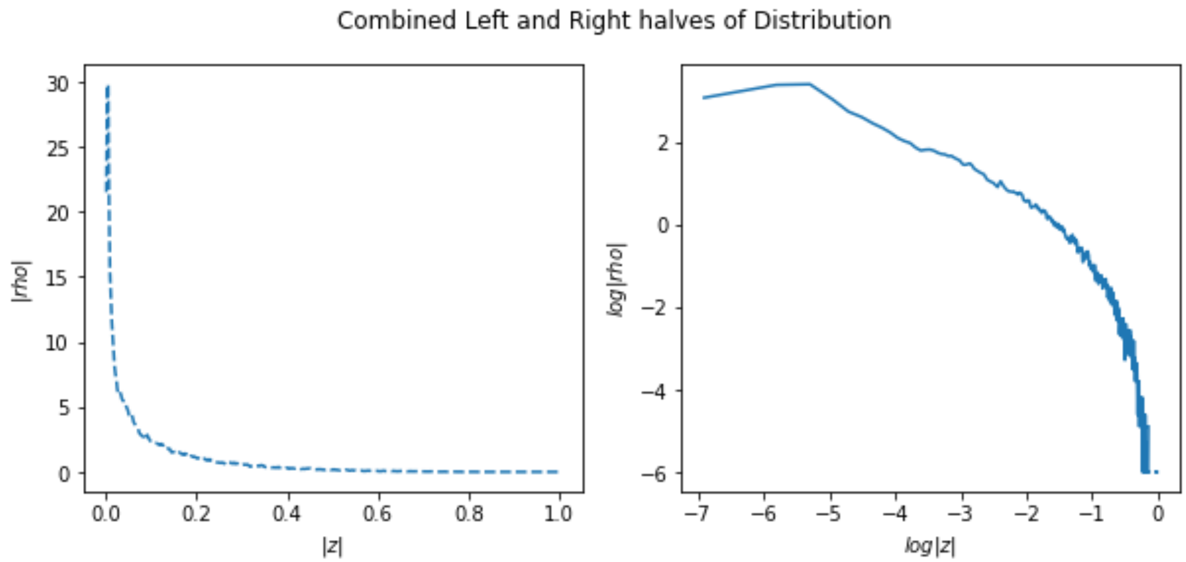
ΔE_i vs E_i @ i = 32
1e−6
Average
$E_{64} - E_{32}$
E_{32}
1e−5



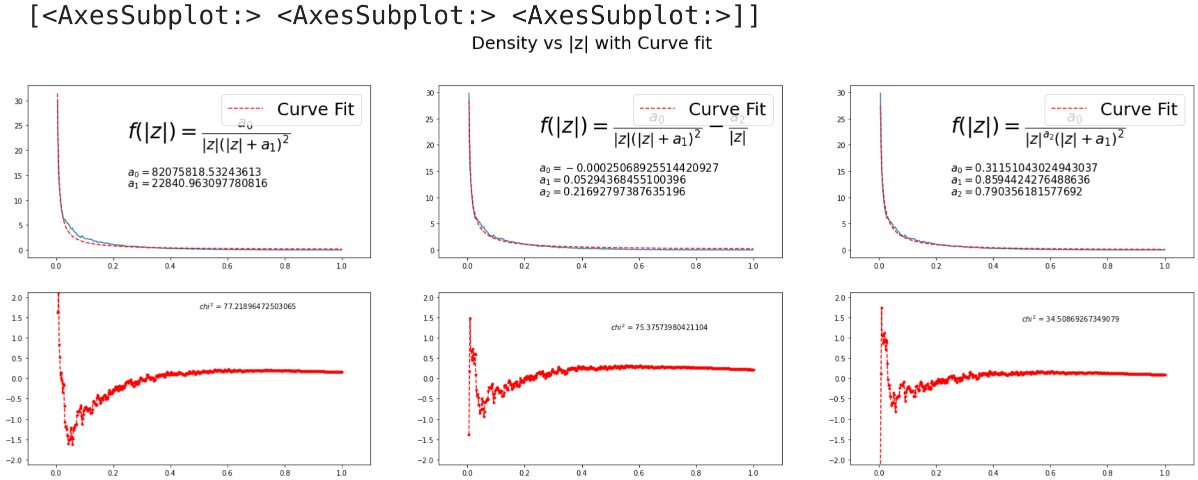
1e−5

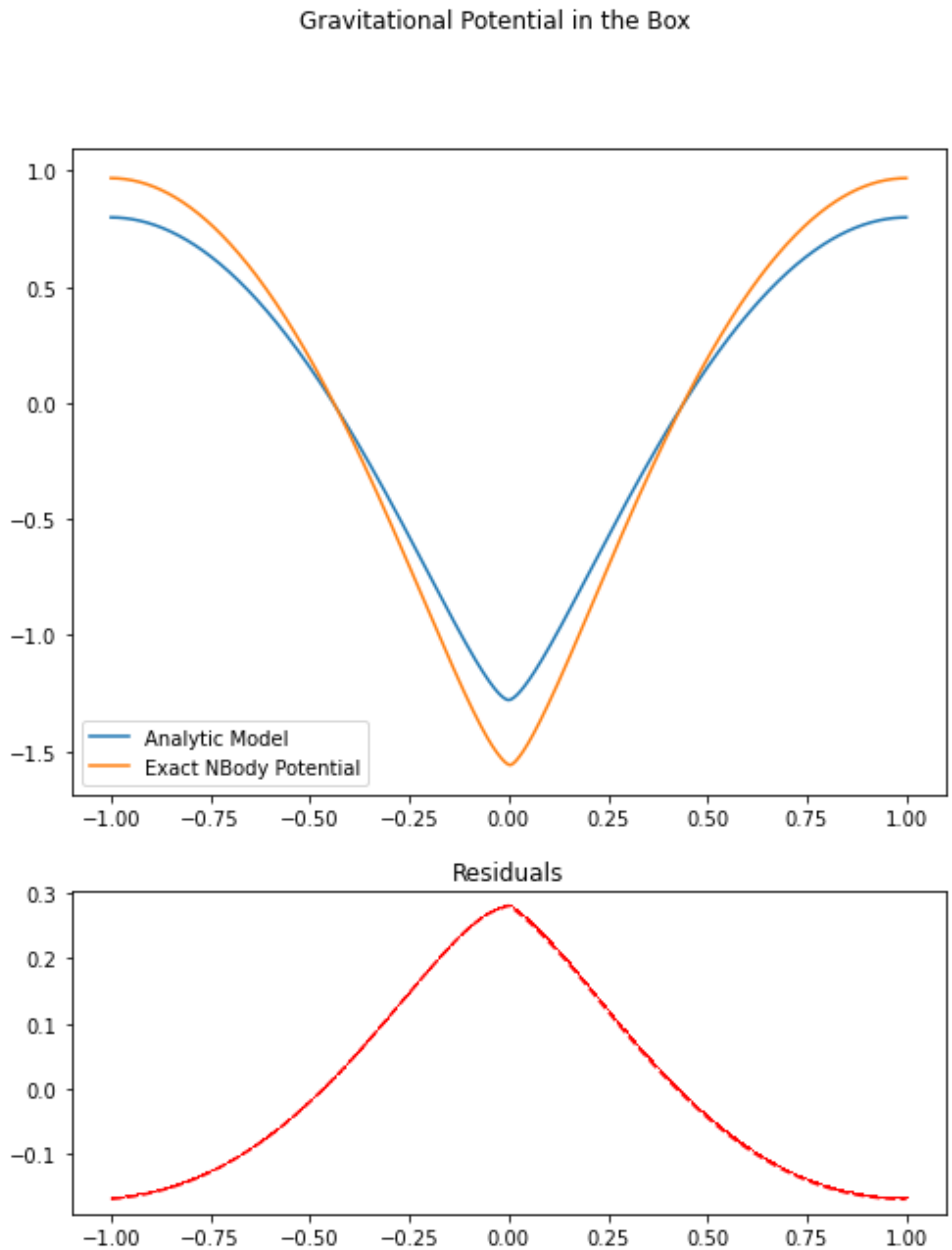


Density of Particles Split in Half
Centroid of Distribution



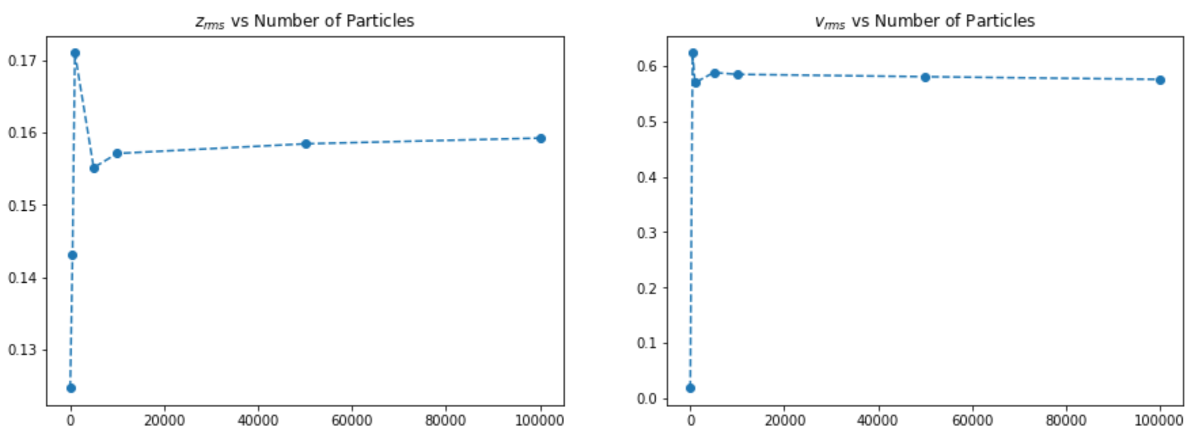

```
Check
Check
Check
#columns = 3
[[<AxesSubplot:> <AxesSubplot:> <AxesSubplot:>]
 [<AxesSubplot:> <AxesSubplot:> <AxesSubplot:>]]
```

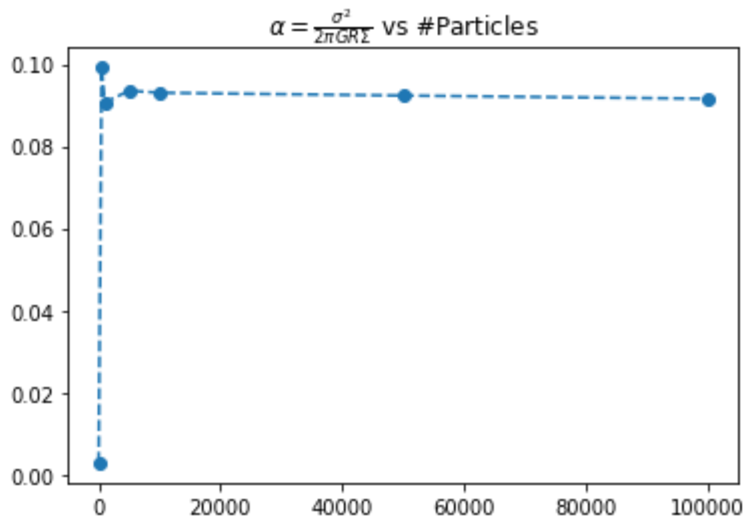



Gravitational Potential in the Box
1.0
0.5
0.0
−0.5
−1.0
−1.5
Analytic Model
Exact NBody Potential
−1.00 −0.75 −0.50 −0.25 0.00 0.25 0.50 0.75 1.00
Residuals
0.3
0.2
0.1
0.0
−0.1
−1.00 −0.75 −0.50 −0.25 0.00 0.25 0.50 0.75 1.00

In [ ]:
```
import matplotlib.pyplot as plt
import numpy as np
V_rms_s = np.copy(v_rms_s)
Z_rms_s  = np.copy(v_rms_s)
Num_p_s = [5,500,1000,5000,10000,50000,100000]


Z_rms_s = Z_rms_s[0:]
V_rms_s = V_rms_s[0:]
Num_p_s = Num_p_s[0:]

print(Z_rms_s,V_rms_s,Num_p_s)

fig,ax=plt.subplots(1,2,figsize= (15,5))
ax[0].plot(Num_p_s,z_rms_s,'o--')
ax[0].set_title("$z_{rms}$ vs Number of Particles")
#ax[0].set_ylim(0,0.5)

ax[1].plot(Num_p_s,v_rms_s,'o--')
ax[1].set_title("$v_{rms}$ vs Number of Particles")
plt.show()

alpha_s = V_rms_s**2 / (2*np.pi*Z_rms_s) #divided by total mass (= 1)
plt.plot(Num_p_s,alpha_s,'o--')
plt.title("$\\alpha = \\frac{\\sigma^2}{2\\pi G R \\Sigma}$ vs #Particles")
plt.show()
```

```
[0.01867563 0.62328138 0.56959758 0.5878238  0.58461931 0.5803635
 0.57551038] [0.01867563 0.62328138 0.56959758 0.5878238  0.58461931 0.5803
635
 0.57551038] [5, 500, 1000, 5000, 10000, 50000, 100000]
```





In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt

My_Package_PATH = "/home/boris/Documents/Research/Coding"
import sys
sys.path.insert(1, My_Package_PATH)
import OneD.NBody as NB

z = np.linspace(-1,1)

x = 0
v = 1
star = NB.star(0,1,x,v)

dt = 0.1
t = 0
i = 0
while t < 2:
    plt.plot(star.x,0,'ro')
    plt.xlim(-1,1)
    plt.show()

    star.x -= v*dt
    star.reposition(2)
    t += dt
```

In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.cm as cm
from matplotlib.colors import LogNorm, Normalize
import os
import subprocess
import cv2
from PIL import Image
import scipy.optimize as opt

#Import My Library
My_Package_PATH = "/home/boris/Documents/Research/Coding"
import sys
sys.path.insert(1, My_Package_PATH)
import OneD.WaveNonDim as ND
import OneD.NBody as NB
import OneD.GlobalFuncs as GF

#Set up Directory for saving files/images/videos
# Will not rename this again
dirExtension = "1D_Codes/Non-Dim/Analysis"
Directory = os.getcwd()#+"/"+dirExtension #os.curdir() #"/home/boris/Documen
print(Directory)

r,m,Num_bosons,sigma,Num_stars = [0.5,1.0,0,1,10000]

mu = m #M_scale = 1

L = 2
N = 10**3
z = np.linspace(-L/2,L/2,N)
dz = z[1]-z[0]

folder = "ParticlesOnly_Snapshots"
stars_x = np.loadtxt(folder+"/"+f"StarsOnly_Pos.csv", dtype = float, delimit
stars_v = np.loadtxt(folder+"/"+f"StarsOnly_Vel.csv", dtype = float, delimit
Energies = np.loadtxt(folder+"/"+"Energies.csv", dtype = float,delimiter = '
#chi = np.loadtxt(folder+"/"+f"Chi.csv", dtype = complex, delimiter=",")
chi = np.zeros_like(z)
centroids = np.loadtxt(folder+"/"+"Centroids.csv",dtype = float, delimiter='

stars = [NB.star(i,sigma,stars_x[i],stars_v[i]) for i in range(len(stars_x))

grid_counts = NB.grid_count(stars,L,z)

rho = (grid_counts/dz)*sigma


i = 0
max_bool = False
while max_bool == False:
    for j in range(len(rho)):
        if rho[j] > rho[i]: #if you come across an index j that points to a
            #then set i equal to j
            i = j
            #break
        else:
            max_index = i
```

```
                max_bool = True
max_rho = rho[max_index]

#Other method to accumulate left and right sides:
for star in stars:
    star.x = star.x - z[max_index] #shift
    star.reposition(L) #reposition

grid_counts = NB.grid_count(stars,L,z)
rho_part = (grid_counts/dz)*sigma
#Add the density from the FDM
rho_FDM = mu*np.absolute(chi)**2
rho = rho_FDM + rho_part

centroid_z = 0
for j in range(len(grid_counts)):
    centroid_z += z[j]*grid_counts[j]
centroid_z = centroid_z / Num_stars

stars_x = [star.x for star in stars]

std = np.std(stars_v)
mean_x = np.mean(stars_x)


R = 0
while True:
    R += dz
    mass_enclosed = 0
    star_collection = []
    for star in stars:
        if np.abs(star.x-mean_x) <= R:
            mass_enclosed += 1
            star_collection.append(star)
    print(R,mass_enclosed)
    if mass_enclosed >= 0.5*Num_stars:
        break

print(R)
plt.figure()
plt.scatter(stars_x,stars_v,s=1)
xx = np.linspace(-R,R,100)
plt.plot(xx,np.sqrt(R-xx**2))
plt.plot(xx,-np.sqrt(R-xx**2))
plt.scatter([star.x for star in star_collection],[star.v for star in star_cc
plt.show()

Sigma = std**2 / R
print(Sigma)
```

In [ ]:

```
G = 6.67E-11
print(R)
print("--------------------")
print("")

Sigma = std**2 / (np.pi* R**(3/2))
print(Sigma)

print(10000/R)
```

```
print(std**2)
```

In [ ]:
```
v_rms = np.sqrt(np.mean([star.v**2 for star in stars]))
print(v_rms)
print(std**2 * R)
```

In [ ]:
```
v_mean = np.mean([star.v for star in stars])
std = np.sqrt(np.sum([(star.v - v_mean)**2 for star in stars])/(len(stars)-1
print(std)

Sigma = std**2 / (np.pi* R**(3/2))
print(Sigma)

print(10000/R)
print(std**2)
print(10000/(np.pi*R**2))

print(std**2 * R)
```

In [ ]:
```
phi_part = GF.fourier_potentialV2(rho_part,L)
phi_part = phi_part - np.mean(phi_part)
print(np.mean(phi_part))

phi_part = phi_part - np.max(phi_part)

# Compute Chandrasekhar's potential energy tensor:
a_part = NB.acceleration(phi_part,L)
W = 0
for i in range(len(z)):
    dW = rho_part[i]*z[i]*a_part[i]
    W += dW
print(W)

a_part = NB.acceleration(phi_part,L)
W = 0
for i in range(len(z)):
    dW = -0.5*rho_part[i]*phi_part[i]
    W += dW
print(W)

# Compute only for the stars that exist:
a_part = NB.acceleration(phi_part,L)
W = 0
for star in stars:
    g = NB.g(star,a_part,dz)

    dW = - star.x*g
    W += dW / Num_stars
print(W)

# phi_part = GF.fourier_potentialV2(rho_part,L)
# a_part = NB.acceleration(phi_part,L)
# W = 0
# for i in range(len(z)):
#     dW = - dz*a_part[i]**2 / (8*np.pi)
#     W += dW
# print(W)
#W = np.sum(phi_part)
#print(W)

# Compute only for the stars that exist:
W = 0
for star in stars:
    #g = NB.g(star,a_part,dz)
    i = int(star.x//dz)
    rem = star.x % dz

    if i != len(phi_part)-1:
        value = phi_part[i] + rem*(phi_part[i+1]-phi_part[i])/dz
    elif i == len(phi_part)-1:
        # then i+1 <=> 0
        value = phi_part[i] + rem*(phi_part[0]-phi_part[i])/dz

    phi_star = value
    dW = phi_star
    W += dW
print(W)
```

Compute Total KE and Total Potential Energy of Stars

In [ ]:
```
# Compute total KE of stars:
K = 0
for star in stars:
    dK = 0.5*sigma*star.v**2
    K += dK
print(K)
#average KE:
print(K/Num_stars)

# #Compute Total Potential
# W = 0
# for star in stars:
#     #g = NB.g(star,a_part,dz)
#     i = int(star.x//dz)
#     rem = star.x % dz

#     if i != len(phi_part)-1:
#         value = phi_part[i] + rem*(phi_part[i+1]-phi_part[i])/dz
#     elif i == len(phi_part)-1:
#         # then i+1 <=> 0
#         value = phi_part[i] + rem*(phi_part[0]-phi_part[i])/dz

#     phi_star = value
#     dW = phi_star
#     W += dW
# print(W)
# #average W:
# print(W/Num_stars)

# Compute only for the stars that exist:
a_part = NB.acceleration(phi_part,L)
W = 0
for star in stars:
    g = NB.g(star,a_part,dz)

    dW = - sigma*star.x*g
    W += dW
print(W)
print(W/Num_stars)
```

Calculate v_{rms} and R_{syst}

Want to verify

$$\langle v^2 \rangle = \frac{GM}{R_{syst}}$$

In [ ]:
```
v_rms = np.sqrt(np.mean([star.v**2 for star in stars]))
z_rms = np.sqrt(np.mean([star.x**2 for star in stars]))
print(f"v_rms = {v_rms}")
print(z_rms)
#v_rms = np.sqrt(np.sum([star.v**2 for star in stars])/Num_stars)

K = 0.5 * v_rms**2
print(f"K_avg = 0.5*m*v_rms^2 = {K} (m=1)")
print(F"=> 2*K_avg = {2*K}")

print(z_rms*Num_stars)

print("-----------------------")




R_syst = Num_stars / v_rms**2
print(R_syst)



rho_0 = np.mean(rho_part)
print(4*rho_0*z_rms)

print(v_rms**2 / (2*np.pi*z_rms))

print(16*np.pi*rho_0**2*z_rms**3 / Num_stars)
```

In [ ]:
```
plt.plot(z,phi_part)
plt.plot(z,-Num_stars/np.abs(z))
plt.ylim(5*np.min(phi_part),-np.min(phi_part))
```

In [ ]:
```
phi_part = phi_part - (np.max(phi_part)-np.max(-Num_stars/np.abs(z)))

plt.plot(z,phi_part)
plt.plot(z,-Num_stars/np.abs(z))
plt.ylim(5*np.min(phi_part),-np.min(phi_part))
plt.show()

# Compute total KE of stars:
K = 0
for star in stars:
    dK = 0.5*star.v**2
    K += dK
print(K)
#average KE:
print(K/Num_stars)

#Compute Total Potential
W = 0
for star in stars:
    #g = NB.g(star,a_part,dz)
    i = int(star.x//dz)
    rem = star.x % dz

    if i != len(phi_part)-1:
        value = phi_part[i] + rem*(phi_part[i+1]-phi_part[i])/dz
    elif i == len(phi_part)-1:
        # then i+1 <=> 0
        value = phi_part[i] + rem*(phi_part[0]-phi_part[i])/dz

    phi_star = value
    dW = phi_star
    W += dW
print(W)
#average W:
print(W/Num_stars)
```

In [ ]:

```
def f(z,*p):
    u_0 = p[0]
    z_0 = p[1]
    return u_0 / np.cosh(0.5*z/z_0)**2

guess = [rho_0,z_0]
popt,pcov = opt.curve_fit(f,z,grid_counts,p0 = guess)
plt.plot(z,grid_counts)
plt.plot(z,f(z,*popt))
plt.show()

guess = [rho_0,z_0]
popt,pcov = opt.curve_fit(f,z,phi_part,p0 = guess)
plt.plot(z,phi_part)
plt.plot(z,f(z,*popt))
plt.show()

def g(z,*p):
    return p[0]*np.exp(-z**2 / p[1])

guess = [-rho_0,z_0]
popt,pcov = opt.curve_fit(g,z,phi_part,p0 = guess)
plt.plot(z,phi_part)
plt.plot(z,g(z,*popt))
plt.show()
```

In [ ]: