

```

1. # superbubbles.py (python3)
2.
3. from numpy import pi, sqrt, arccos, arcsin, exp, log
4. from tqdm import tqdm
5. import scipy.integrate as integrate
6. import numpy as np
7. import os, plawt
8.
9. figdir = 'figures'
10. if not os.path.exists(figdir):
11.     os.mkdir(figdir)
12.
13. # Dimensionless Scaling
14. H = 1 # [L] = H
15. gamma = 5/3
16. L_not = 1
17. rho_not = 1
18. P = 1
19.
20. def r(z,y):
21.     """ Get the shape of the shockfront """
22.     arg = 1 - y**2/(4*H**2) + exp(-z/H)
23.     arg *= exp(z/(2*H))/2
24.     return 2 * H * arccos(arg)
25.
26. def z12(y):
27.     """
28.     Get the edges of the shockfront
29.     returns tuple (z1, z2)
30.     """
31.     return (-2*H*log(1 - y/(2*H)), -2*H*log(1 + y/(2*H)))
32. z12 = np.vectorize(z12)
33.
34. def rmax(y):
35.     """ Get max radius of the bubble """
36.     return 2*H*arcsin(y/(2*H))
37.
38.
39. def shockfronts():
40.     import imageio
41.

```

```

42. z = np.arange(-2, 10, 0.00001)
43. y = [0.1, 0.5, 1, 1.4, 1.7, 1.9, 1.98, 2.0]
44. figure1 = {
45.     'ylabel': 'z/H', 'xlabel': 'r/H',
46.     'filename': 'shockfront.png',
47.     'ylim': (-2, 10), 'xlim': (-6, 6),
48.     'figsize': (6/1.3, 6.5/1.3),
49.     'show': False,
50.     # 'legend': {'loc':4}
51. }
52.
53. for i, yi in enumerate(tqdm(y)):
54.     figure1[i] = {'x': np.concatenate((r(z, y[i]), -r(z, y[i]))), 'y': np.concatenate((z, z)), 'label': '$y=$'+str(yi)}
55. plawt.plot(figure1)
56.
57. animation = {
58.     'ylabel': 'z/H', 'xlabel': 'r/H',
59.     'ylim': (-2, 10), 'xlim': (-6, 6),
60.     'figsize': (6/1.3, 6.5/1.3),
61.     'title': 'Likely how W4 expanded',
62.     'show': False,
63.     'keepOpen': True,
64.     'legend': {'loc':4}
65. }
66. y = np.arange(0.01, 2.05, 0.05)
67. with imageio.get_writer('blast.gif', mode='I', fps=24) as writer:
68.     for i, t in enumerate(tqdm(y)):
69.         animation[0] = {'x': np.concatenate((r(z, y[i]), -r(z, y[i]))), 'y': np.concatenate((z, z)),
70.             'line':'k-', 'label': '$y=$'+str(y[i])}
71.         plt = plawt.plot(animation)
72.         fig = plt.gcf()
73.         fig.canvas.draw()
74.         data = fig.canvas.tostring_rgb()
75.         row, col = fig.canvas.get_width_height()
76.         image = np.fromstring(data, dtype=np.uint8).reshape(col, row, 3)
77.         writer.append_data(image)
78.         plt.close()
79.
80. shockfronts()
81.
82. ### Math Helpers ###

```

```

83.
84. # Derivatives of stuff
85. dy = lambda Eth, Omega: sqrt((gamma**2 - 1)*Eth / 2 / (rho_not * Omega))
86. drdy = lambda z, y: y / ( 2*sqrt(1 - 1/4*exp(z/H)*(1-y**2/(4*H**2)+exp(-z/H))**2) )
87. dOmega = lambda y, dy: 2 * pi * integrate.quad(lambda z: r(z, y) * drdy(z, y) * dy, z12(y)[1], z12(y)[0])[0]
88. dEth = lambda y, dy, P: L_not - P * dOmega(y, dy)
89.
90. # Equations from paper
91. PFunc = lambda E, O: (gamma - 1)*E/O
92. PFunc = np.vectorize(PFunc)
93. OmegaFunc = lambda y: pi * integrate.quad(lambda z: r(z, y)**2, z12(y)[1], z12(y)[0])[0]
94. EnergyFunc = lambda oprev, onext, E: L_not*dt - (gamma-1)*E*(onext-oprev)/oprev+E
95.
96. dzsdt = lambda y, E, O: ( dy(E, O)/(1-y/(2*H)), -dy(E, O)/(1+y/(2*H)) )
97. dzsdt = np.vectorize(dzsdt)
98.
99. ###
100.
101. # initial conditions
102. dt = 0.0001 # only seems to work with this dt
103. time = np.arange(0.005, 10, dt)
104. yi = 0.01
105. Omegai = OmegaFunc(yi)
106. Ethi = P/(gamma-1)*Omegai
107.
108. initialstate = [yi, Omegai, Ethi]
109. ys = [yi]
110. Omegas = [Omegai]
111. Es = [Ethi]
112.
113. # Integrate
114. for t in tqdm(time):
115.     ynext = ys[-1] + dy(Es[-1], Omegas[-1])*dt
116.     omeganext = OmegaFunc(ynext)
117.     energynext = EnergyFunc(Omegas[-1], omeganext, Es[-1])
118.
119.     ys.append(ynext)
120.     Omegas.append(omeganext)
121.     Es.append(energynext)
122.     if ynext > 1.99999:
123.         break

```

```

124.
125. # Calculate extras
126. z12s = z12(ys)
127. r = np.vectorize(r) # vectorize after we're done integrating because it makes it really slow otherwise
128. Ps = PFunc(Es, Omegas)
129. dz1sdt = dzsdt(ys, Es, Omegas)[0]
130.
131. # Plot
132. plawt.plot({
133.     0: {'x': time[:len(ys)], 'y': ys, 'line':'k-'},
134.     'show':False,
135.     'filename': os.path.join(figdir, 'y.png'),
136.     'title': "(a)  $y$  vs Time",
137.     'xlabel': ' $\tilde{t}$ ',
138.     'ylabel': ' $\tilde{y}$ ',
139.     'set_yscale': 'log', 'set_xscale': 'log',
140.     'xlim': (0.01, 10), 'ylim': (0.1, 10.0),
141.     'grid':True
142. })
143. plawt.plot({
144.     0: {'x': time[:len(ys)], 'y': Es, 'line':'k-'},
145.     'show':False,
146.     'filename': os.path.join(figdir, 'energy.png'),
147.     'title': "Thermal Energy vs Time",
148.     'xlabel': ' $\tilde{t}$ ',
149.     'ylabel': ' $\tilde{E}_{th}$ ',
150.     'set_yscale': 'log', 'set_xscale': 'log',
151.     'xlim': (0.01, 10), 'ylim': (0.01, 10.0),
152.     'grid':True
153. })
154. plawt.plot({
155.     0: {'x': time[:len(ys)], 'y': z12s[0], 'label': ' $\tilde{z}_1$ ', 'line':'k-'},
156.     1: {'x': time[:len(ys)], 'y': -z12s[1], 'label': ' $\tilde{z}_2$ '},
157.     2: {'x': time[:len(ys)], 'y': r(0, ys), 'label': ' $\tilde{r}(z=0, y)$ ', 'line': 'k--'},
158.     'filename': os.path.join(figdir, 'blastedges.png'),
159.     'title': 'Blast Edges vs. Time',
160.     'xlabel': ' $\tilde{t}$ ',
161.     'ylabel': 'Distance',
162.     'legend': {'loc': 4},
163.     'set_yscale': 'log', 'set_xscale': 'log',
164.     'xlim': (0.01, 10), 'ylim': (0.1, 10.0),

```

```

165.     'grid':True
166. })
167. plawt.plot({
168.     0: {'x': time[:len(ys)], 'y': Ps, 'line':'k-'},
169.     'show':False,
170.     'filename': os.path.join(figdir, 'pressure.png'),
171.     'title': "Pressure vs. Time",
172.     'xlabel': '$\\tilde{t}$',
173.     'ylabel': '$\\tilde{P}$',
174.     'set_yscale': 'log', 'set_xscale': 'log',
175.     'xlim': (0.01, 10), 'ylim': (0.01, 10.0),
176.     'grid':True
177. })
178. plawt.plot({
179.     0: {'x': time[:len(ys)], 'y': dz1sdt, 'line':'k-'},
180.     'show':False,
181.     'filename': os.path.join(figdir, 'blastedgespeed.png'),
182.     'title': "Blast Edge Speed",
183.     'xlabel': '$\\tilde{t}$',
184.     'ylabel': '$d\\tilde{z}_1/dt$',
185.     'set_yscale': 'log', 'set_xscale': 'log',
186.     'xlim': (0.01, 10), 'ylim': (0.01, 10.0),
187.     'grid':True
188. })
189. plawt.plot({
190.     0: {'x': z12s[0], 'y': dz1sdt, 'line':'k-'},
191.     'show':False,
192.     'filename': os.path.join(figdir, 'blastedgeSpeedvsPos.png'),
193.     'title': "Blast Edge Speed vs. Position",
194.     'xlabel': '$\\tilde{z}_1$',
195.     'ylabel': '$d\\tilde{z}_1/dt$',
196.     'set_yscale': 'log', 'set_xscale': 'log',
197.     'xlim': (0.1, 10), 'ylim': (0.1, 10.0),
198.     'grid':True
199. })

```