Exploring the Use of the Peng-Robinson Equation of State in Python

Sunny Zhao

August 19, 2020

Contents

1 Introduction

The Peng-Robinson equation of state(PREOS) is one of many cubic equations of state that describe the properties of fluids. This project aims to discover methods in which the PREOS can be used to calculate vapor pressure. It is intended to function with all fluids known to have an acentric factor. This document uses propane as an example to examine the capabilities and limitations of the Python program. *cv6Enterprise.py* is written in Python3, and displays all of its results in the terminal of any UNIX operating system and Matplotlib.

1.1 Modules used:

NumPy provided many of the mathematical functions used in the program.
Cmath was used to handle complex arithmetic.
Matplotlib was used to provide custom plots of related functions.
SciPy was used to find roots to a function.
The contents of *cv6Enterprise.py* is included in the repository on GitHub.

2 Equations

2.1 PREOS

The equation of state describes that pressure can be expressed in terms of molar volume:

$$P = \frac{RT}{\underline{V} - b} - \frac{a}{\underline{V}(\underline{V} + b) + b(\underline{V} - b)} \tag{1}$$

By substituting pressure for the compressibility factor, Z, the compressibility factor can also be expressed in an equation where it is cubic in molar volume:

$$Z = \frac{\underline{V}}{\underline{V} - b} - \left(\frac{a\underline{V}}{RT}\right)\left[\frac{1}{\underline{V}(\underline{V} + b) + b(\underline{V} - b)}\right] \tag{2}$$

Unlike the van der Waals equation of state, the parameter, a is a function of temperature. a and b are given by the following equations:

$$\begin{aligned} a &= a_c\alpha \\ a_c &= 0.45724R^2\frac{T_c^2}{P_c} \\ a &= \left[1 + \kappa\left(1 - T_r^{0.5}\right)\right]^2 \\ \kappa &= 0.37464 + 1.524226\omega - 0.269932\omega^2 \\ b &= 0.07780R\frac{T_c}{P_c} \end{aligned}$$

Where ω is the acentric factor.

2.2 Fugacity

Fugacity is derived from the expression for Gibbs free energy. After many substitutions and simplifications, fugacity expressed in terms of the PREOS is:

$$f = P\ \exp\left((Z-1) - \ln(Z-B) - \frac{A}{2B\sqrt{2}} \times \ln\left[\frac{Z + (1+\sqrt{2})B}{Z + (1-\sqrt{2})B}\right]\right) \tag{3}$$

With constants A and B:

$$\begin{aligned} A &= \frac{aP}{R^2T^2} \\ B &= \frac{bP}{RT} \end{aligned}$$

3 Functions in *cv6Enterprise.py*

All properties are computed using a gas constant of,
$R = 8.314462\,MPa \cdot cm^3/mol \cdot K$. Hence, pressure and molar volume will be reported in MPa and cm^3/mol, respectively.
The temperature of interest can be adjusted by changing the globally-defined variable, T. Inputting a pressure is optional; the variable, P, is usually used with the functions $freikugel(P)$ and $fugac(P, V)$. Other constants must be manually changed, like the acentric factor, critical pressure, and critical temperature. The constants for propane were found in *Fundamentals of Chemical Engineering Thermodynamics* by Dahm and Visco. [1] The triple point pressure and temperature was also used in this program. The approximate values for propane were found on NIST. [2]

3.1 $freikugel(P)$

This function systematically solves for the molar volumes of a fluid at a given pressure. Rearranging Equation 1 into the standard form of a cubic equation($rx^3 + sx^2 + tx + u = 0$) gives:

$$(-P)\underline{V}^3 + (RT - P2b + Pb)\,\underline{V}^2 + \left(2bRT - a + 3b^2P\right)\underline{V} + \left(ab - Pb^3\right) = 0$$

Where:

$$\begin{aligned} r &= -P \\ s &= RT - Pb \\ t &= 2RTb - a + 3b^2P \\ u &= ab - RTb^2 - Pb^3 \end{aligned}$$

Let $\underline{V} = x$. Substituting ($x = y - \frac{s}{3r}$) will depress the cubic to a form:

$$y^3 + py + q = 0$$

Where:

[1] Dahm, K. ; Visco, D. Appendix C.1 Critical Point, Enthalpy of Phase Change, and Liquid Molar Volume. *Fundamentals of Chemical Engineering Thermodynamics*, 1st Ed.; Cengage Learning, 2014, pp. 735

[2] NIST Chemistry WebBook. https://webbook.nist.gov/cgi/cbook.cgi?ID=C74986&Mask=4 (Accessed August 19, 2020)

$$
\begin{aligned}
p &= \frac{1}{r}\left(t - \frac{s^3}{3r}\right) \\
q &= \frac{1}{r}\left(u + \frac{2s^3}{27r^2} - \frac{st}{3r}\right)
\end{aligned}
$$

The roots of the cubic equation can be solved for using Vieta's substitution. [3] Substituting $y = w - \frac{p}{3w}$ results:

$$w^3 + q - \frac{p^3}{27w^3} = 0$$

After multiplying by w^3, the equation is now quadratic in w^3:

$$(w^3)^2 + q(w^3) - \frac{1}{27}p^3 = 0 \tag{4}$$

Equation 4 can be solved for using the quadratic formula. It will result in two values for w^3, and theoretically, six values for w. However, the roots from each root of w^3 are identical, leaving only 3 distinct roots for w. These cubic roots can be solved for using De Moivre's theorem:

$$\text{Let } w^3 = W.$$

$$w = \sqrt[3]{W}\left[\cos\left(\frac{\pi + 2\pi k}{n}\right) + i\sin\left(\frac{\pi + 2\pi k}{n}\right)\right] \tag{5}$$

$$\text{For } n = 3 \text{ and } k \text{ is a positive integer less than } n.$$

Once the three values of w are known, molar volume can be back-solved through all the substitutions. The lowest molar volume would correspond to the liquid molar volume, and the highest molar volume would correspond to the vapor molar volume. $freikugel(P)$ will compute molar volumes at any temperature specified between the triple point pressure and critical point pressure. However, its accuracy cannot be measured. For instance, at 90 K—just 5 degrees above the triple point temperature—the liquid molar volume was computed to be 35.7 cubic centimeters/mol. This would raise in error when calculating the fugacity of a liquid, because the logarithmic term in the equation is undefined when the volume is less than b, the molar volume of a substance at maximum compression. For comparison, b for propane is equal to 56.3 cubic centimeters.

[3] https://en.wikipedia.org/wiki/Cubic_equation#Vieta's_substitution

3.2 *plotcubic()*

This function plots the cubic equation of state. It creates an array of 200 values from $b + 1$ to 2000 cubic centimeters/mol and used Matplotlib to plot it against Equation 1. The starting point is $b + 1$ because substituting b for $\underline{V}$ into Equation 1 would result in a divide by zero error. An error would be raised and printed in the terminal output. An example plot is included as Figure 1:

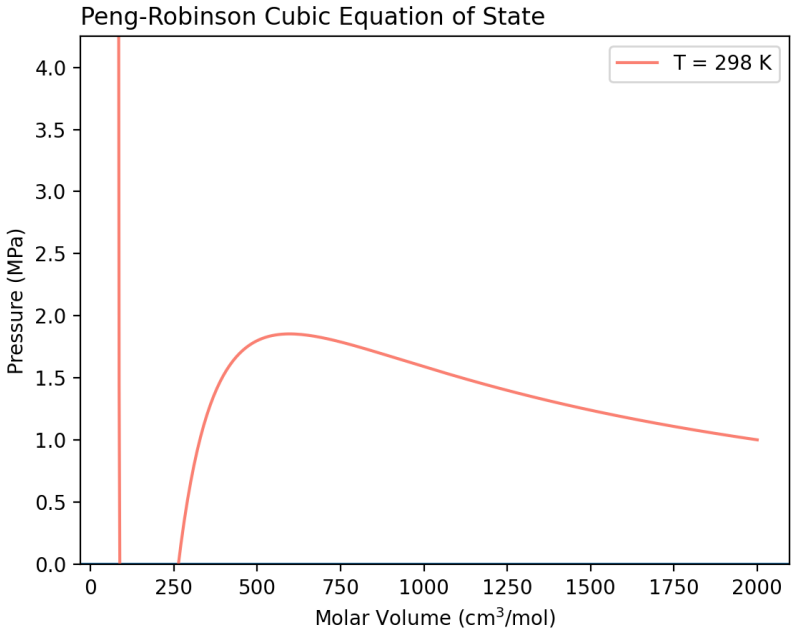

Figure 1: Plot of pressure vs molar volume at 298 K

It can be seen in Figure 1 that the function is asymptotic as it approaches the value of b from the right.

3.3 $fugac(P, V)$

This functions computes the fugacity of a substance given a molar volume and pressure, using Equation 3. It is used in conjunction with $freikugel(P)$, by taking the output of $freikugel(P)$ and using it as an input.

3.4 $plotFugacityDifference()$

This function plots the difference in fugacity: $f_l - fv$ vs pressure. A sample plot is shown in Figure 2 for a temperature of 298 K:

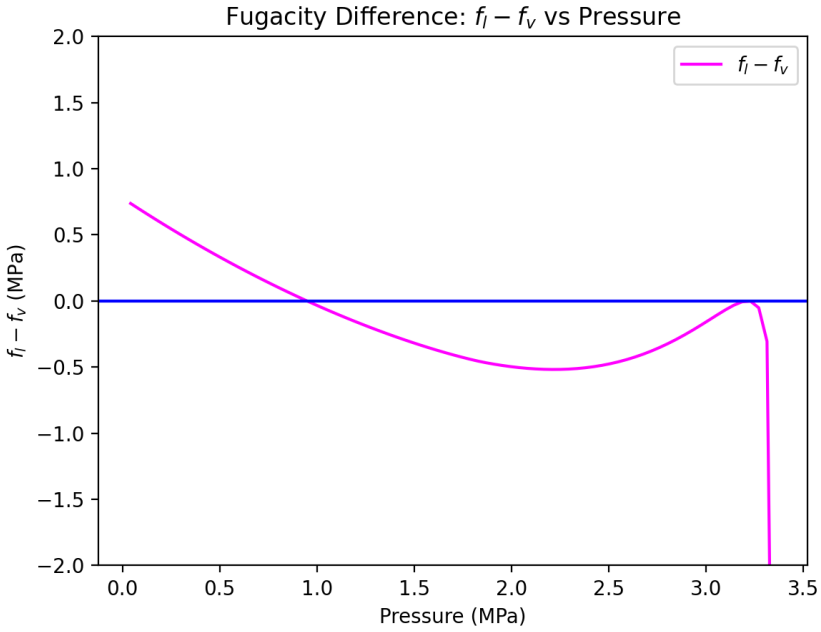

Figure 2: Fugacity difference vs pressure at 298 K

It can be seen in Figure 2 that two roots exist for the graph. One is at 0.949 MPa, and the other at about 3.3 MPa. The first-degree root is the vapor pressure at that temperature. The second-degree root has no physical meaning. When the value of the function is positive, it indicates that the substance is a vapor at that pressure since the fugacity of the vapor is computed to be lower than the fugacity of the liquid. Likewise, when the function is below the x-axis, the substance will be a liquid in that range of pressures.

3.5 $optim(P)$

This function is a combination of $freikugel(P)$ and $fugac(P, V)$. Vapor-liquid equilibrium exists when the difference between the liquid and vapor fugacity is zero.

SciPy.optimize.newton [4] helps to find the first-degree root as seen in Figure 2. The function additionally requires an initial guess as a parameter. A plot of vapor pressure vs temperature for propane found on The Engineering ToolBox[4] looks like it could be linear in log10 scale. It is shown as Figure 3:

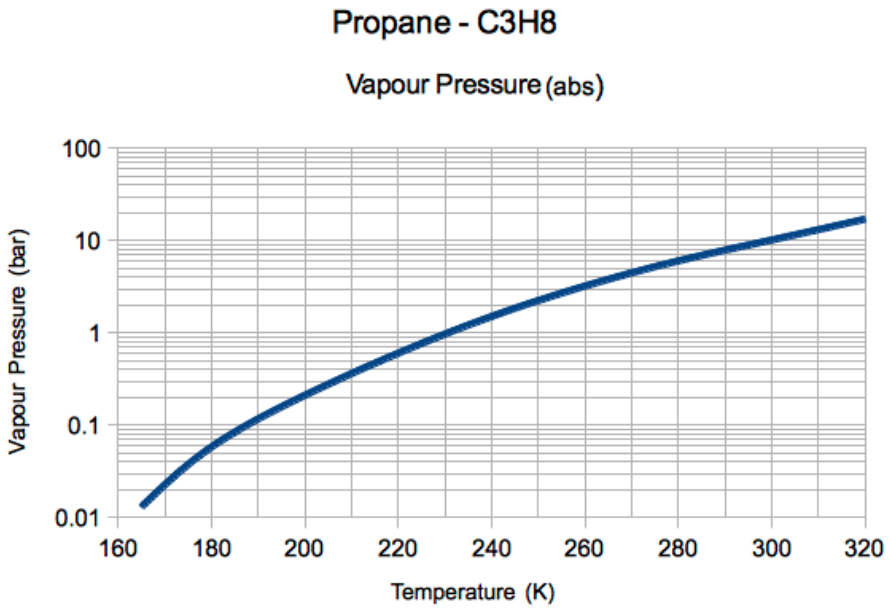

Figure 3: A plot of vapor pressure vs temperature for propane from The Engineering ToolBox[4]

A guess value could be calculated using interpolation. Vapor-liquid equilibrium only exists between the triple point and the critical point. Interpolating between these points may not give the best estimate, but it will provide an input that will scale with temperature. This estimate is actually very far from the actual root. At 298 K, the interpolated guess is 0.01 MPa. The actual vapor pressure is 0.949 MPa.

[4]scipy.optimize.newton https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.newton.html (Accessed Aug 19, 2020)

[4]

[4]Propane-Vapor Pressure https://www.engineeringtoolbox.com/propane-vapor-pressure-d_1020.html (Accessed Aug 19, 2020)

3.6 $antoine(T)$

In order to verify the results obtained from the previous functions, they needed to be compared to real data. The closest was a derivative of the Antoine equation found on the DIPPR Project 809 database. It has an error of less than 1%, and was fitted from a few data sets. [5] It is applicable to all temperatures between the triple point and critical point, and has the form:

$$Y = \exp\left[A + \frac{B}{T} + C\ln T + DT^E\right] \tag{6}$$

3.7 $percentError(x, accepted)$

This function calculates percent error. It provides a measure of accuracy when comparing the result obtained from the PREOS to the Antoine equation.

[5] Goodwin, R.D.; Haynes, W.M. 1982 Thermophysical Properties of Propane from 85 to 700 K at Pressures to 70 MPa National Bureau of Standards Monograph 170, Boulder, Colorado
Younglove, B.A.; Ely, J.F.Thermophysical Properties of Fluids. II. Methane, Ethane, Propane, Isobutane, and Normal Butane 577 J. Phys. Chem. Ref. Data 1987 16 4

4 Limitations

Although the minimum temperature in which vapor-liquid equilibrium can exist for propane is 85 K, the lowest temperature *cv6Enterprise.py* can reasonably calculate is 230 K. According to Figure 4, a root does exist, but *scipy.optimize.newton* fails to converge at that value. According to Figure 4, the vapor pressure at 229 K is

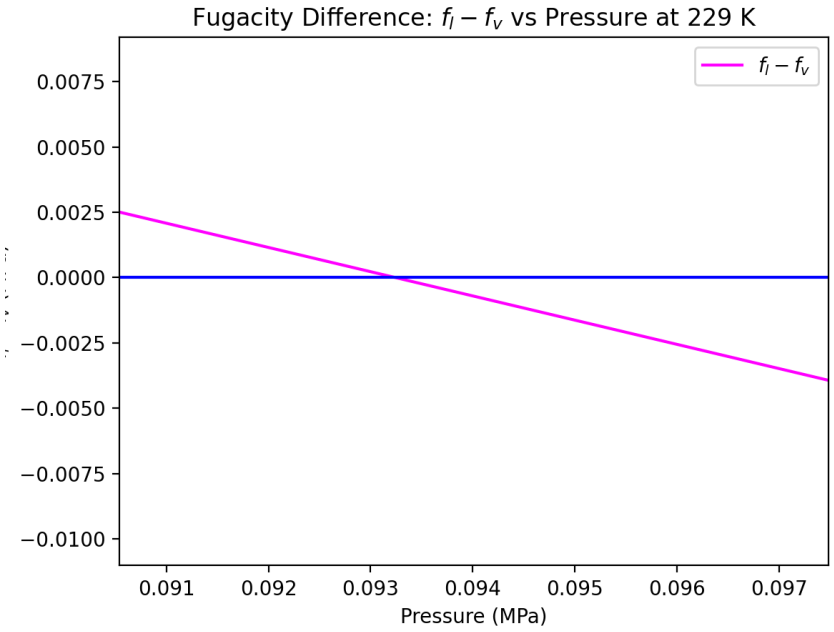

Figure 4: Fugacity difference plot at 229 K

approximately 0.093 MPa. It is additionally known that a root appears near that pressure, because the Antoine equation calculates a corresponding pressure of 0.0923 MPa.

A table of temperatures in which a root did converge is included in Table 1 of the appendix. According to the table, at many of the temperatures, *cv6Enterprise.py* yielded decent estimates for the vapor pressure. Using the vapor pressure obtained from the Antoine equation as real data, the corresponding percent error was usually small, with the exception of two entries at 356 and 357 K. The vapor pressures at these two temperatures were 5.23 and 5.27 MPa, respectively. The calculated pressures each had a percent error greater than 50%. This is peculiar because the

interpolated guesses at those two temperatures were 1.33 and 1.45 MPa, respectively. Additionally, it was only at these two temperatures that there was such a large percent error. The temperatures that preceded and followed produced more accurate values. The Antoine equation calculates a vapor pressure near 3 MPa, indicating that the interpolated guesses were significantly different from what was returned by the optimization function. Extending the domain of the fugacity difference plot, the function reveals this:

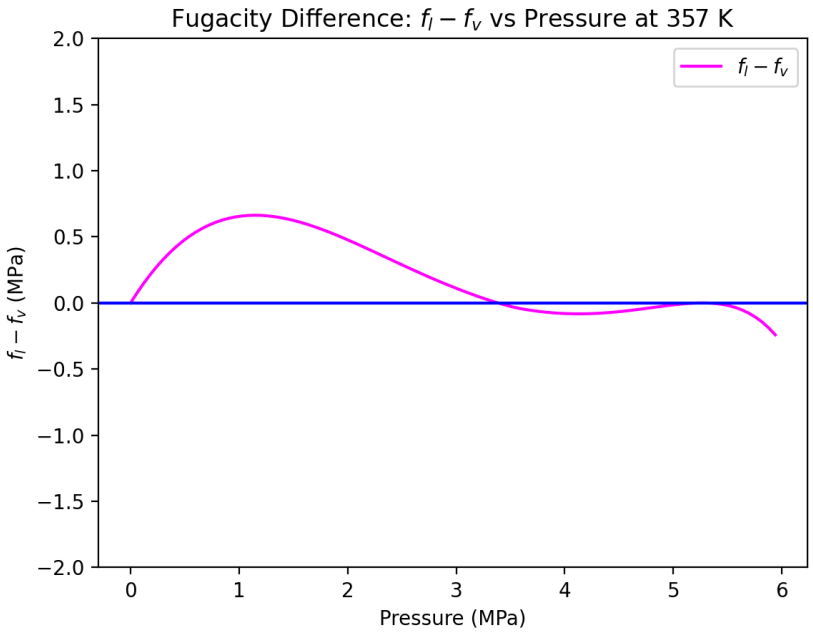

Figure 5: Fugacity difference plot at 357 K

Figure 5 shows a first-order root at the expected vapor pressure and a second-order root near 5 MPa, the vapor pressure that the optimization function had returned. The optimization function had detected the incorrect root at 356 K and 357 K. Thus, at certain temperatures, the optimization function may not be the most optimal to use to optimize a function.

5 Possible Improvements

Many issues were encountered while using the *scipy.optimize.newton* function. The function sometimes does not locate the correct root. If a specific substance were to be studied, it could be useful to provide a better guess to the optimization function by using the Antoine equation. *cv6Enterprise.py* was intended to work universally for all substances with a known acentric factor, and it would require additional research to find Antoine equation constants. The Antoine equation for propane found on a database applied to all possible temperatures for vapor-liquid equilibrium. Other Antoine equations may not be as simple because they would comprise of different sets of Antoine constants. For instance, the Antoine equation found on the NIST WebBook for propane [6] has three sets of constants, all of which apply to different temperature ranges. A piecewise function would need to be constructed to provide an estimate for all temperatures within the VLE range. Using real data will always be the most accurate, but when it is not available, an equation of state would have to suffice.

Instead of relying on Newton's method, perhaps a more specialized method could be developed to approach root-finding for the fugacity difference function. There could be ways to mathematically manipulate the function so that a specialized optimization method can selectively locate these roots.

[6] Propane. https://webbook.nist.gov/cgi/cbook.cgi?ID=C74986&Mask=4&Type=ANTOINE&Plot=on#ANTOINE (Accessed Aug 20, 2020)

6 Appendix

6.1 Table 1.

Table 1: Comparison of Calculated Vapor Pressure and Real Data

Temperature (K)	calculated P_{vap} (MPa)	Antoine P_{vap} (MPa)	percent error (%)
230	0.0974	0.0966	0.821
231	0.1017	0.1010	0.772
232	0.1063	0.1055	0.724
233	0.1110	0.1102	0.678
234	0.1158	0.1151	0.633
235	0.1208	0.1201	0.590
240	0.1484	0.1478	0.396
244	0.1739	0.1734	0.264
245	0.1807	0.1803	0.234
246	0.1878	0.1874	0.206
247	0.1950	0.1947	0.178
249	0.2102	0.2100	0.127
250	0.2181	0.2179	0.104
252	0.2347	0.2345	0.060
253	0.2433	0.2432	0.039
254	0.2522	0.2521	0.020
255	0.2613	0.2613	0.002
256	0.2706	0.2707	0.016
257	0.2802	0.2803	0.032
258	0.2901	0.2902	0.047
259	0.3002	0.3004	0.061
260	0.3106	0.3108	0.074
262	0.3322	0.3325	0.098
263	0.3434	0.3438	0.108
264	0.3549	0.3553	0.118
265	0.3667	0.3671	0.126
266	0.3788	0.3793	0.134
267	0.3911	0.3917	0.141
268	0.4038	0.4044	0.147
269	0.4168	0.4174	0.152

270	0.4301	0.4308	0.156
271	0.4437	0.4444	0.160
272	0.4577	0.4584	0.163
273	0.4719	0.4727	0.165
274	0.4865	0.4874	0.166
275	0.5015	0.5023	0.166
276	0.5168	0.5176	0.166
277	0.5324	0.5333	0.166
278	0.5484	0.5493	0.164
279	0.5647	0.5656	0.162
280	0.5814	0.5823	0.159
281	0.5985	0.5994	0.156
282	0.6159	0.6168	0.152
283	0.6337	0.6346	0.147
284	0.6519	0.6528	0.142
285	0.6705	0.6714	0.136
286	0.6894	0.6903	0.130
287	0.7088	0.7097	0.123
288	0.7286	0.7294	0.115
289	0.7487	0.7495	0.108
290	0.7693	0.7701	0.099
291	0.7903	0.7910	0.090
292	0.8117	0.8124	0.081
293	0.8336	0.8342	0.071
294	0.8559	0.8564	0.061
295	0.8786	0.8790	0.051
296	0.9018	0.9021	0.040
297	0.9254	0.9257	0.028
298	0.9495	0.9496	0.017
299	0.9740	0.9740	0.005
300	0.9990	0.9989	0.008
301	1.0245	1.0243	0.020
302	1.0504	1.0501	0.033
303	1.0769	1.0763	0.047
304	1.1038	1.1031	0.060
305	1.1312	1.1303	0.074
306	1.1591	1.1581	0.088
307	1.1875	1.1863	0.103

308	1.2164	1.2150	0.117
309	1.2459	1.2442	0.132
310	1.2758	1.2740	0.147
311	1.3063	1.3042	0.162
312	1.3374	1.3350	0.177
313	1.3689	1.3663	0.192
314	1.4010	1.3981	0.208
315	1.4337	1.4305	0.224
316	1.4669	1.4634	0.239
317	1.5007	1.4969	0.255
318	1.5350	1.5309	0.271
319	1.5700	1.5655	0.287
320	1.6055	1.6006	0.303
321	1.6416	1.6363	0.319
322	1.6783	1.6726	0.335
323	1.7155	1.7095	0.351
324	1.7534	1.7470	0.368
325	1.7919	1.7851	0.384
326	1.8310	1.8237	0.400
327	1.8708	1.8630	0.416
328	1.9111	1.9029	0.432
329	1.9521	1.9434	0.448
330	1.9938	1.9846	0.463
331	2.0361	2.0264	0.479
332	2.0790	2.0688	0.495
333	2.1227	2.1119	0.510
334	2.1669	2.1556	0.526
335	2.2119	2.2000	0.541
336	2.2575	2.2450	0.556
337	2.3039	2.2908	0.571
338	2.3509	2.3372	0.586
339	2.3986	2.3843	0.600
351	3.0284	3.0056	0.757
355	3.2631	3.2372	0.800
356	5.2323	3.2971	58.696
357	5.2704	3.3578	56.963
358	3.4476	3.4193	0.828
359	3.5108	3.4817	0.837

360	3.5749	3.5449	0.845
361	3.6398	3.6090	0.853
362	3.7056	3.6740	0.861
363	3.7723	3.7398	0.868
364	3.8398	3.8065	0.874
365	3.9082	3.8741	0.881
366	3.9776	3.9426	0.886
367	4.0478	4.0120	0.892
368	4.1190	4.0824	0.897
369	4.1910	4.1536	0.901