Numerical Analysis — Final Exam

Dec. 6, 2022. During this exam, you are welcome to use your copies of the HP-25 Owner's Handbook and the HP-25 Applications Programs book, and your notes, but not the internet.

Problem 1 — Statistics — Power Law Fit

Perhaps the most famous power law relationship of all time is captured in this table:

In[25]:=
```
planetList = AstronomicalData["Planet"];

earthSemimajorAxis = AstronomicalData["Earth", "SemimajorAxis"];

earthPeriod = AstronomicalData["Earth", "OrbitPeriod"];

aList = AstronomicalData[#, "SemimajorAxis"] / earthSemimajorAxis & /@ planetList;

pList = AstronomicalData[#, "OrbitPeriod"] / earthPeriod & /@ planetList;

NumberForm[TableForm[{planetList, aList, pList}, TableHeadings → {{"", "Distance", "Period"}, None}], 4]
```

Out[30]//NumberForm=

	Mercury	Venus	Earth	Mars	Jupiter	Saturn	Uranus	Neptu
Distance	0.3871	0.7233	1.000	1.524	5.203	9.537	19.19	30.07
Period	0.2408	0.6152	1.000	1.881	11.86	29.45	84.02	164.8

The upper row of data is the planet's distance from the Sun. This plays the role of x.

The lower row is how long the planet takes to go around the Sun. This is called the period. It plays the role of y.

You might find it odd that the distance and period are both 1.000 for Earth, but that's because we have chosen(!) to measure distances using the Earth's distance from the Sun, and we measure times in Earth years.

(a) Use the program on p. 99 of the Applications Programs book to enter this data (just as was done with the example data on p. 100). What does the program give for the a and b values in the formula

$$y = ax^b$$

HINT/NOTE: Since Earth has 1.000, 1.000 as its values, a good cross-check on your data entry is that a should come out extremely close — if not identically equal — to 1.000. b is what is really interesting. It is Kepler's Third Law.

(b) If an asteroid was found to be orbiting at a distance from the Sun of 3.700 in these units. What would its period be?

NOTE: Asteroids orbit the Sun following the same laws as planets. Asteroids are just smaller.

Problem 2 — Differential Equations — Euler's Method

There is a differential equation for approach of temperature T to an equilibrium temperature S called the Stefan-Boltzmann Law. It is:

$$T'(t) \equiv \frac{dT}{dt} \equiv \underset{\text{as teeny as } \Delta t \text{ can be made}}{\text{limit}} \frac{\Delta T}{\Delta t} = -k\left(T^4 - S^4\right)$$

In this equation, k and S are just numbers. We will choose $k = 1.20 \times 10^{-10}$ and $S = 300$. For this problem, use $h = 0.25$, and $T(0) = 2000$.

(a) For your program, put k in R5 and S in R6, and then write the program (with your starting point being the program on p. 83) that solves this problem using Euler's method.

(b) Fill in the table below.

NOTES: T is playing the role of y. Also, you should not use R2 for y, even though that is documented to contain y! (I could explain why, but we went through this program very carefully in class, so let's not go back into it.) Instead, for y, you must use the y that is in stack register Y.

In[32]:=
```
TableForm[Table[{t, If[t == 0, 2000, "    "]}, {t, 0, 5, 0.25}], TableHeadings → {None, {"t", "T"}}]
```

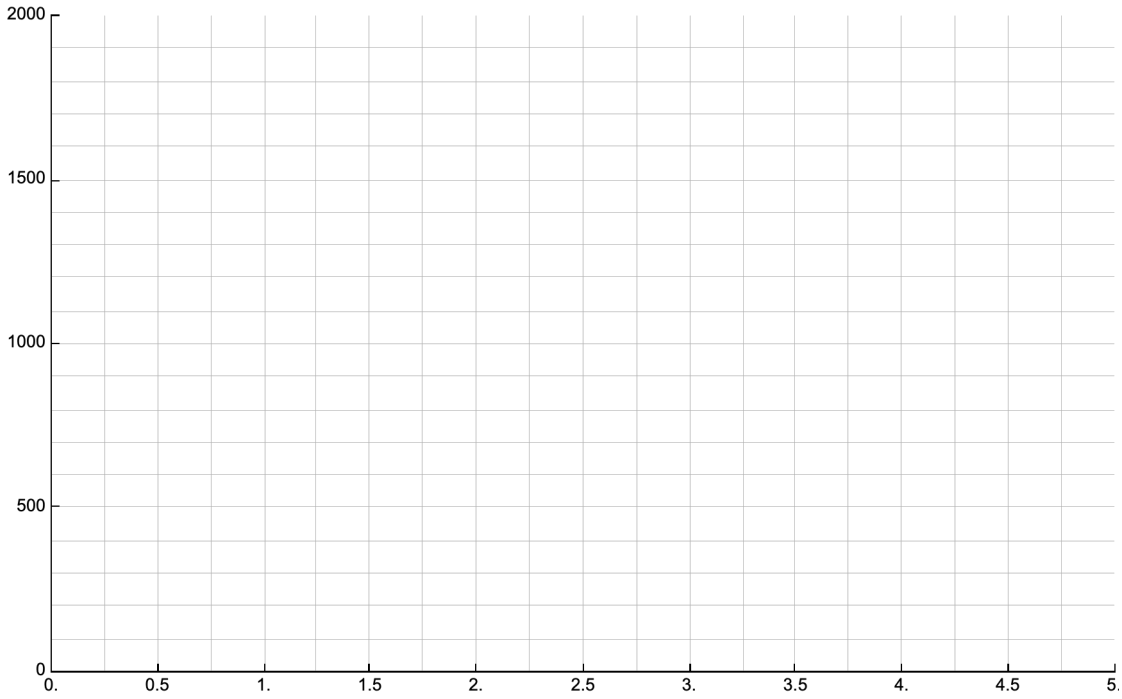
Out[32]//TableForm=

t	T
0.	2000
0.25	
0.5	
0.75	
1.	
1.25	
1.5	
1.75	
2.	
2.25	
2.5	
2.75	
3.	
3.25	
3.5	
3.75	
4.	
4.25	
4.5	
4.75	
5.	

(c) Graph your results:

In[54]:=
```
Plot[{}, {x, 0, 5}, PlotRange → {{0, 5}, {0, 2000}},
 Ticks → {Range[0, 5, 0.5], Range[0, 2000, 500]}, GridLines → {Range[0, 5, 1/4], Range[0, 2000, 100]}]
```

Out[54]=



Congratulations! You just found how something pulled out of a forge at 2000K (that's 1727 Centigrade or 3140° Fahrenheit) would cool towards room temperature in its first five minutes.

Problem 3 — Interpolation — Using Quadratic Functions

Suppose you have three points with x-values at -1, 0, and 1, but the y-values can be anything. I'll write the three points as:

$(-1,\ y_{-1}),\ (0,\ y_0),\ \text{and}\ (1,\ y_1).$

There are formulas for the parabola, $y = ax^2 + bx + c$, that goes through these three points.

The formulas for $a,\ b,\ \text{and}\ c$ are:

$a = \frac{1}{2}(y_1 + y_{-1}) - y_0$

$b = \frac{1}{2}(y_1 - y_{-1})$

$c = y_0$

Write a program that does two things:

(a) Assuming that y_{-1} is in R0, y_0 is in R1, and y_1 is in R2, have your program compute a and put it in R3 and b and put it in R4. You don't need to compute c, because it is just y_0 which is already in R1.

For part (a), your program just stops after it has done its work, but if the user puts an x-value into the X-register and hits R/S again, ...

(b) Your program should then compute $a\,x^2 + bx + c$ and then return to the same spot that it stopped before (not the beginning of the program) so that it can be ready to do this again with another x-value.

If you run out of time to do (b), bummer, but at least document what you wrote for part (a) so I can give you partial credit.

Then use your program to work through the following example:

(c) With the values:

In[•]:=
```
y₋₁ = 0.60653;
y₀ = 1.00000;
y₁ = 1.64872;
```

run your program and write down the a and b, it has calculated.

(d) Put your calculator into f FIX 5 mode (so that it shows 5 digits after the decimal place). Use the program you wrote in part (b) to fill in the following table:

In[•]:=
```
TableForm[Table[{x, If[x == -1, y₋₁, If[x == 0, "1.00000", If[x == 1, y₁, "    "]]]}, {x, Range[-1, 1, 0.2]}]]
```

Out[•]//TableForm=

−1.	0.60653
−0.8	
−0.6	
−0.4	
−0.2	
0.	1.00000
0.2	
0.4	
0.6	
0.8	
1.	1.64872

HINT/NOTE: Of course, a good test of your program is that it works for the three values that were originally given, so don't skip them. As a note in passing, I'll say that the points I just had you do were to approximate the exponential $e^{x/2}$ by a parabola, which of course doesn't work perfectly, because an exponential is not a parabola. I checked all the values and the approximation works to better than 1%.