HW3B (3.4, 3.5, 3.6 on paper, 3.7 online) due Friday Feb 1

3.4 **PRETEND** you are trying to computationally solve for the root of the equation $f(x) = 10^x - 100 = 0$ using the bisection method starting with the initial "bracket" $a = 0$, $b = 3$.

8 pts

(a) Are these acceptable values for a and b? Explain. (Write out the general criteria that determine whether an initial a and b are acceptable for the bisection method, and use that to explain whether these values in particular are acceptable for this function $f(x)$.)

(b) Calculate the values of tol_x and tol_f you would use for the two convergence criteria, $|x_k - x_{k-1}| < tol_x$ and $|f(x_k)| < tol_f$, if you were to write your own bisection method in MATLAB.

Now (pretend) you've completed 33 iterations with the last two iterations having values x_{32} = **2**.000**1**0000000000**1** and x_{33} = **2**.000**1**00000000000. Knowing (by inspection) the exact solution is x = **2**, answer the following (*and SHOW YOUR WORK!!*):

(c) What is the (just plain old) *error* in the last iteration (x_{33})?

(d) What is the "proxy" error for the last iteration?

(e) What is the "residual" error for the last iteration?

(f) Based on all your values above, explain whether the process has "converged" yet or not.

3.5 Use the `fzero` command in MATLAB to solve problem 3.3 for the weight of the hog, w, at t = 30 days. If you weren't sure of the right $f(x) = 0$ equation to use in problem 3.3, be sure to check my solutions posted after classes Monday.

4 pts

Do NOT submit your MATLAB routines for this problem. But as proof you accomplished this task correctly, please write out all the following on paper:

(a) The entire function ("something".m) you needed to create for `fzero` to call.

(b) How you called the fzero command in your command window. It should be something like:

```
...something... = fzero(...something...)
```

(c) Your final answer for the hog weight w, in pounds, rounded to four decimals (*not* 4 sig. figs; I do mean 4 *decimals*).

3.6 Assume the Newton-Raphson method to solve $f(x) = 0$ converges according to $\dfrac{|e_{k+1}|}{|e_k|^r} \approx C$ for $C = 0.1$, and assume the error for the first iteration is $e_1 = 0.1$.

4 pts

(a) Calculate the absolute value of the errors for the first 5 iterations $|e_1|$ through $|e_5|$, and plot these errors on a "convergence plot" of k vs. $\log_{10}|e_k|$, (by-hand, on paper).

(b) Explain by which iteration k should the value of x_k be accurate to within 8 decimals.

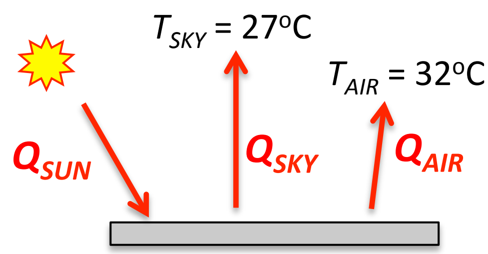
Remember: For every question *show your work*! You'll be very sad when you aren't given credit when you write down the correct answer but don't show me your thought process or assumptions behind that answer.

HW3B (3.4, 3.5, 3.6 on paper, 3.7 online) due Friday Feb 1

3.7 13 pts — Ever wanted to fry an egg on hot pavement? The Imagination™ Station in Toledo says you've got to get the egg above 70°C to coagulate the proteins. Lets create and solve a physics-based model to see if the surface of a road can really get that hot.

Here's a realistic model of the heats (*Q*, in Watts) entering and leaving the surface of a road at (unknown) temperature **T** (in Celsius). You'll learn more about this in Heat Transfer later:

- Q_{SUN} = 650 is solar heat directly hitting the pavement at noon on a very sunny day.
- Q_{AIR} = (15) (**T** – 32) is the heat convected away from the pavement by slow moving air at 32°C.
- Q_{SKY} = (5.103 x 10^{-8}) { (**T**+273)^4 – (27+273)^4 } is the heat radiating off the pavement to the upper sky at 27°C. The "+273" in the equation is converting °C to Kelvin.



Pavement at unknown temperature **T** (in °C)

The temperature **T** (in Celsius) of the pavement is obtained when all the incoming heat balances the outgoing heat, *i.e.* when $Q_{SUN} = Q_{AIR} + Q_{SKY}$.

Following the instructions below, you'll solve for **T** *two* ways:

1. Writing your *own* <u>BISECTION</u> routine, which calls your *own* function `pavement(T)`,
2. Using MATLAB's built-in `fzero` function

and then compare how they did (and what's easier to do!).

[A] Create the "f(x)" function

First, look at the model above, and think about the function $f(T)$ needed so that the solution to the root of $f(T) = 0$ gives you the desired pavement temperature **T**.

Hint: Since **T** is the solution to $Q_{SUN} = Q_{AIR} + Q_{SKY}$, then a great idea is to make $f(T) = Q_{SUN} - Q_{AIR} - Q_{SKY}$, where each Q has the formula above.

[B] Turn *f*(*T*) into a MATLAB function

Create a function called **pavement.m** that starts with this *exact* line: `function f = pavement(T)` .
Complete the function so that it outputs the value of the function $f(T)$ from part [A] for an input `T` in °C.

[C] Create a plot of $f(T)$ over T = 0 to 100°C to help you visualize the problem. From the plot, what do *you* think would be a good first bracket [*a*, *b*] for the bisection method?

(You do **not** submit this plot to me – it's just for you to see the problem before you get started.)

[D] Start with my "code fragment" (provided online) for the script **HW3_7.m**. Complete the code by writing your *own* <u>bisection</u> method to converge on the solution for *T*:

- Use the **given** initial bracket of [*a*, *b*] = [0, 100] Celsius. Do ***not*** change those values please!
- Develop appropriate convergence tolerances (tol_x, tol_f) based on the initial bracket just like we discussed in class.
- Use a **while** loop to keep iterating until *both* convergence criteria are met (*i.e. keep* looping if either or both criteria are *not* met). Remember: you *don't know* the "real" solution for temperature *T* yet when comparing to tol_x, so use the "proxy" error $|T_k - T_{k-1}|$.
- Be sure to store the value of each iteration T_k in a vector called `T_history` so you can plot the method's convergence history later.

HW3B (3.4, 3.5, 3.6 on paper, 3.7 online) due Friday Feb 1

3.7 Continued …

- For the purposes of keeping track of the iterations in the “history” vector, define $T_1 = a$ and $T_2 = b$ as the “first two” iterations, and then define T_3 to be the first midpoint of *a* and *b*.

[E] After the routine converges, add even more code to your HW3_7.m script to do the following:

- Run `fzero` with iterations displayed, using the *same* initial bracket [0, 100].
- Assume this result from `fzero` is the “true” answer for temperature $\hat{T}$, and use it to back-calculate a vector of all the “true” errors $e_k = \left|\hat{T} - T_k\right|$ for all the iterations in your Bisection routine you stored in the *T_history* vector.
- Make & label a *convergence history plot* of your bisection method: plot $\log_{10}(e_k)$ as a function of iteration *k*, up until convergence. Save this plot as a pdf called **PLOT3_7.pdf**.

That’s it! Admire how well your code did converging compared to the built-in MATLAB routine. What kind of shape do you see in your “convergence history” plot? What did you think was easier (writing your own code [D] or using `fzero` [E])? Did you ever actually fry the egg (was the pavement above 70°C)?

Finally, here’s exactly what I need you to submit ONLINE:

- Your convergence history plot **PLOT3_7.pdf**.
- Your **two m-files**: your completed, documented script **HW3_7.m** and function **pavement.m**.
- Enter **answers** to the following three questions (in order) in the comment section on Carmen:
  a) From your Bisection routine [D], what is the final (converged) **value** for temperature *T* to four decimals (*e.g.* 43.1234 °C), and **how many** iterations did it take?
  b) From the `fzero` routine [E], what is the **value** of $\hat{T}$ to four decimals, and **how many** iterations did it take?
  c) Does the **shape** of your convergence plot ($\log_{10}$(error) vs. *k*) indicate a *linear*- or *quadratic*-type convergence rate? Explain (briefly).
- There is nothing to submit on paper for 3.7. Only the things above in Carmen.