HW1B (1.3 – 1.5) Due by Beginning of Class, **Friday** January 19

1.3 Basic MATLAB function using VECTORIZATION, and ERROR definitions

6 pts You probably already know from calculus that you can write $\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \cdots$

a) Create a MATLAB function called **`FUNC1_3.m`** to calculate the sum of the terms up to the power x^N. That is, evaluate $S(x,N) = \sum_{k=1,3,5,\ldots}^{N} (-1)^{(k-1)/2} \left(\frac{x^k}{k!}\right)$

For example, $S(2,11) = 2 - \frac{2^3}{3!} + \frac{2^5}{5!} - \frac{2^7}{7!} + \frac{2^9}{9!} - \frac{2^{11}}{11!} = 0.909296136\ldots$

Your function `FUNC1_3` must accept two inputs (*x* and *N*) and output one value representing *S*(*x*,*N*).

What's the catch? **NOWHERE in your function can you use `FOR` or `WHILE` loops, or `IF` commands!!!** You must think how to do all calculations in a "vectorized" way, as discussed in class.

(*Hint:* maybe create a vector of just the odd values of *k* from 1 to *N*; then use MATLAB's element-by-element math operators to create a vector in which the elements are $(-1)^{(k-1)/2} x^k / k!$; then use the `sum` function to add up the terms of the vector. Or, maybe you have another way of doing it. Just don't use `for-while-if` !!)

b) Use your function to demonstrate to yourself that the series converges (as $N \rightarrow \infty$) to the limit **1** for $x = \pi/2$. (This should make sense to you because you know $\sin(\pi/2) = 1$.)

Do the demonstration by using your new MATLAB function to calculate *S*(*x*,*N*) for *x* = pi/2 and *N* = 5, 11 and 17, and you should see the result getting closer to 1 as *N* gets bigger. Let's define the "error" as the difference between the "exact" sin(*x*) and the "approximate" *S*(*x*,*N*).

Now calculate the "errors" (*i.e.* differences between 1 and $S(\pi/2, N)$) for *N* = 5, 11 and 17. Use `format long` to be sure to capture each of the three "errors" to at least three significant figures. These are the three values you're going to enter into Carmen. You should see these "errors" get closer to 0 as *N* gets larger.

For this problem, please submit the following:

- ONLINE (in HW1B assignment folder): **Four** things! Your *documented* function **`FUNC1_3.m`**, and your **three** values of error for $S(\pi/2, 5)$, $S(\pi/2, 11)$, $S(\pi/2, 17)$, in the **comment** section (clearly labeled).

  By "Comment Section" I mean type it right into the comment field beside where you upload files for your assignment on Canvas, just like you did in HW1.1. It is NOT *sufficient to just type it as a part of your documentation inside your function FUNC1_3.m. I want the graders to be able to see your results before they even open up and run your code.*

  Also – don't forget! You need to submit ALL the files for problems 1.3, 1.4 and 1.5 ALL AT ONCE to your HW1B assignment on Canvas. Like we talked about in the recitations, if you submit files for 1.3 first, then try to add the files for 1.4 later, you will ERASE the files from 1.3, and you'll end up with a grade of 0!

HW1B (1.3 – 1.5) Due by Beginning of Class, **Friday** January 19

1.4 MATLAB script using FOR loops and IF-THEN statements

7 pts Create a MATLAB function called **FUNC1_4.m** which starts with the following line:

```
function C = FUNC1_4(A,B,option)
```

This function will take inputs `A` and `B` (vectors or matrices of any size) and `option` (scalar number from 1 to 3) and create the output matrix `C`, where

- `C = A + B`, if `option` = 1,
- `C = A .* B`, if `option` = 2,
- `C = A * B`, if `option` = 3.

However, this isn't quite as easy at it sounds because you must follow the following two rules:

1. You can **NOT** use vectorized MATLAB operations, like A+B, A.*B, A*B as shown above, which do all the additions and multiplication in one command. You **MUST** use multiple, nested `for` loops to break the problem down into the individual multiplications and/or additions performed on just one **pair of scalar numbers** from A and B at a time (sort of like how we *first* calculated the cannon-ball energies in Class 2).
2. You must **check** at the beginning of the function that the input matrices `A` and `B` have consistent sizes (depending on the value of option) to allow for a mathematically valid `C`. For example, if option = 2 then A and B must have identical size (since A .* B only makes sense if A and B have the same size), but if option = 3 then there's a different requirement on the relative size between A and B to ensure A*B makes sense.
   If you detect an inconsistency in the size of A and B, such that C can not be validly evaluated, then have your function indicate that by setting `C = 'No good! Garbage.'` instead of calculating an actual value for C.

Finally, it's OK to assume that A and B will *never* be scalars, but rather true vectors or matrices, where at least one of the dimensions is greater than 1.

Be sure to test your function out to make sure it works!

HW1.4: Please upload your final **documented** function **`FUNC1_4.m`**, to HW1B in Carmen.

HW1B (1.3 – 1.5) Due by Beginning of Class, **Friday** January 19

1.5 More advanced MATLAB script using VECTORIZATION and 3-D PLOTTING

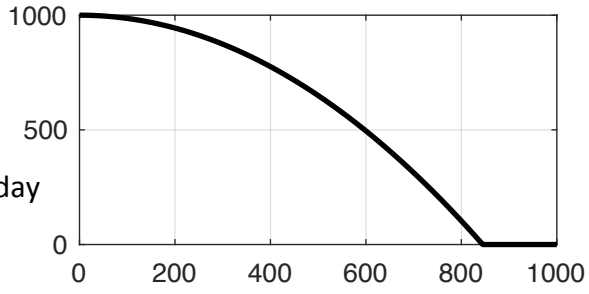
7 pts A systems-engineering assessment of a doll-head making machine has determined that the daily production of heads can be modeled by the equation $P = 10\,s - \left(\frac{s}{2\times10^6}\, e^{s/30}\right)(t-1)^2$ where

- ***P*** is the **number** of doll heads produced on day ***t***.
- ***t*** is the **day** of operation (starting at day t = **1** when it's first turned on),
- ***s*** is the machine **setting** (percent of max operation, between **0** = off and **100** = max operation),

Assume P can be any number ≥ 0. Even fractions are OK. But if you ever calculate $P < 0$ ("negative production??") just assume $P = 0$ that day.

Aside: Just to make sure you understand the equation, first check that you agree with all the following:

- If $s = 0$ (0% max operation) then $P = 0$ for all time t (*i.e.* no dolls are ever made). This makes sense because the machine is always off!
- At $t = 1$ (the first day you start operating the machine) $P = 10\,s$, so the machine can produce between 0 to 1000 heads that first day depending on the machine setting from $s = 0$ to 100.
- For a fixed setting s, as time goes on, the daily rate of head production $P(t)$ drops, presumably because the machine gets old and doesn't operate as efficiently. In fact, for higher settings (as s goes to 100) the machine overheats faster and the rate of efficiency drop-off can get so big that P drops to 0! (See my plot of $P(t)$ at right for $s = 100$)
- Once P hits 0 or goes negative, assume the machine is now broken, and can no longer product doll heads for later times.
- The plot at right is of $P(t)$ for $s = 100$, and shows the trends I described above: See P starts at 1000 dolls/day at $t = 1$, then drops off rapidly over time, finally "breaking" and producing no more heads ($P = 0$) for $t \geq 846$.



Create a MATLAB script called **`DOLLHEAD.m`** that does all the following:

a) Creates a column vector `s` of all settings from 0 to 100, skipping by 5, so $s = \begin{bmatrix} 0 \\ 5 \\ 10 \\ \vdots \\ 100 \end{bmatrix}$.

b) Creates a vector `t` of days from 1 to 1200, so t = [1 2 3 … 1200].

c) Uses a *single, vectorized* equation to create a matrix of values `P` for all the s and t values in the vectors above. You can **NOT** use any **for** or **while** loops to do this calculation! The equation must be cleverly *vectorized* to calculate P with a *single* equation using the entire s and t vectors at once.

d) Overrides all values of $P < 0$ to be equal to zero (*i.e.* no production when the machine is broken).

Now you should have a [21 x 120] matrix for P, where each row represents the daily production of heads for one particular setting. You should double-check that the row corresponding to $s = 100$ (max operation) has exactly the shape of the plot I show above before moving on.

(If you choose to continue this adventure, please go to next page …)

HW1B (1.3 – 1.5) Due by Beginning of Class, **Friday** January 19

1.5 continued …

e) Make a plot of all 21 $P(t)$ lines, one for each setting s, on top of each other in one graph. I expect to see 21 lines on the same plot. This might be as easy as just using the command `plot(t,P)`. **Save** this plot in pdf form called `DHplot.pdf`.

Now I want you to figure out WHICH setting s (between 0 and 100) produces the maximum number of doll heads over the machine's lifetime. For example, if you look at the plot you made in (e), running at a *low* setting (like s = 10) doesn't produce very many doll heads each day, but at least the machine works for all 1200 days. But running at *max* setting (s = 100) produces the most doll heads *initially*, but the machine breaks the soonest, so maybe there's a middle setting s that is the best "overall". To determine this *optimal* setting do the following:

f) In your code, make a new column vector called `TOTAL` , for which each element in `TOTAL` represents the total number of dolls produced over the 1200 day lifetime of the machine for each setting s. In otherwise, TOTAL should be a vector of length 21, with the answer for each setting in the vector `s`.

g) Look at the vectors s and TOTAL to figure out which value of the setting s produced the maximum number of doll heads. It's fine to do this by eye outside your code, or you can try to automate this in MATLAB too if you like (there's no bonus points for doing so, but maybe you like a challenge!).

HW1.5: Please submit the following **3 things** ONLINE in the HW1B folder on Carmen:

- Upload your final **documented** main script **`DOLLHEAD.m`**,
- Upload your labeled plot **`DHplot.pdf`**,
- Enter your value for the optimal setting s from (g) in the COMMENT section for this HW in Carmen. Follow the format below:

  "HW1.5: The optimal setting for s is 20" (but replace 20 with whatever your # is)

HW1B Summary:

Since there's three problems-worth of material (1.3, 1.4, 1.5) you're uploading all at once, here's a summary of everything you're submitting to HW1B on January 19:

HW1.3:

- your *documented* function **`FUNC1_3.m`**
- your **three** values of error for $S(\pi/2, 5)$, $S(\pi/2, 11)$, $S(\pi/2, 17)$, in the **comment** section (clearly labeled).

HW1.4:

- your *documented* function **`FUNC1_4.m`**.

HW1.5:

- your *documented* main script **`DOLLHEAD.m`**,
- your labeled plot **`DHplot.pdf`**,
- your value for the optimal machine setting s in the **comment** section