Matlab Homework 10*a*

In the online book:

- Do the "Challenge Activities" of: 9.1,2, 12.6
- Do the "Participation Activities" of: 16.1-6,9

Matlab Homework 10*b*

The same general requirements as for homework 4b apply. And you must study the posted lesson(s) and have done the online book part above before you can ask a TA or the instructor for help.

1. In the previous homework, you printed out the roots ω_1 through ω_9 of the equation:

$$J_0(\omega) - k\omega J_1(\omega) = 0$$

   using two `for` loops over the root counter n, going from 1 to a value $n_{\max}$. Repeat this, but now make Matlab stop printing out roots when it sees that the error in the guessed value has become less than 0.03. Take $k = 2$ again. Make sure the output is lined up in columns and neat; this time output the frequencies with 2 digits behind the decimal point.

   The correct *old* solution is already in the `q1.m` file. Just modify it as requested.

2. In the previous homework, you created matrices like

$$A = \begin{pmatrix} -2 & 1 & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 \\ 0 & 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 1 & -2 \end{pmatrix}$$

   by initializing a zero matrix, putting in the nonzero elements in rows 2 to $n-1$, (with n the size of the matrix) in a `for` loop, and manually adding the nonzero elements in rows 1 and n.

   This time do the same, but put in *all* nonzero elements in a `for` loop over *all* n rows. Use `if` statements to ensure that you do not write nonzero values outside the boundaries of the matrix in rows 1 and n, as you would normally do. Check it for $n = 4$, 5, and 6.

   The correct *old* solution is already in the `q2.m` file and `vibMatrix`. Just modify them as requested.

3. Some mathematician claims that the sum

$$\sum_{i=1}^{\infty} \frac{1}{i}$$

or in other words

$$\frac{1}{1}+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\ldots$$

is infinite. Let's check that out.

Create a script `sum1Tol.m` that sums the above sum until Matlab finds that the estimated error has become less than 0.01, or up to say 100,000 terms, as many terms as Matlab will sum in a reasonable time. Take the estimated error to be the one appropriate for a non-alternating series (see the posted lesson). If the estimated error does become less than 0.01, print a message

```
The mathematician seems to be wrong.  The sum
seems to be *.12 to about *.12.
```

If not, print a message

```
The mathematician seems to be right.
The error remains *.12 with * terms.
```

In both cases, print the final line with `fprintf` without data numbers in FORMATSTRING.

Warning: Students who end up with frozen homework programs, or messages that Java and/or Adobe are misbehaving, have incorrectly implemented the `break` command, or even omitted it completely. Trying to `publish` 100,000 message lines is a sure recipe for crashing something. Please check operation your `break` command *before* seeing TA or instructor.

4. The same mathematician claims that the sum

$$\sum_{i=1}^{\infty}(-1)^{i-1}\frac{1}{i}$$

or in other words

$$\frac{1}{1}-\frac{1}{2}+\frac{1}{3}-\frac{1}{4}+\frac{1}{5}-\ldots$$

is finite, and in fact equal to ln(2). To check this, copy script `sum1Tol.m` over into `sum1AltTol.m`. Then in `sum1AltTol.m` make the following changes: (a) Change the estimated error into the one appropriate for an alternating series like this. (b) Before the `for` loop initialize a variable `sgn=-1;`. Then inside the loop, use a statement `sgn=-sgn;` and multiply your current term by sgn. Since `sgn` will alternatingly be $+1$ and -1, this produces the $(-1)^{i-1}$ in the terms in an efficient way. At the end, print a message

```
The mathematician seems to be right. The sum
seems to be *.12 to about *.12
after summing * terms.
The deviation from ln(2) is *.1E12.
```

The mathematician is not wrong, as you should have learned in calculus, so do not worry about that possibility.

5. Some mathematician claims that the sum

$$\sum_{\substack{i=1 \\ i \text{ odd}}}^{\infty} \frac{1}{i^2}$$

or in other words

$$\frac{1}{1^2} + \frac{1}{3^2} + \frac{1}{5^2} + \frac{1}{7^2} + \ldots$$

equals $\pi^2/8$

Create a script `sum2OddTol.m` that sums the above sum until Matlab finds that the estimated error has become less than 0.00001, or up to say 100,000 terms, as many terms as Matlab will sum in a reasonable time. Take the estimated error to be the one appropriate for a series like the one above. Print the results out as

```
The mathematician seems to be right:
Found: *.12345
Exact: *.12345
Estimated error: *.1E12
Actual    error: *.1E12
```

6. In the previous homework you summed the Taylor series of the sine integral,

$$\mathrm{Si}(x) = \frac{x}{1!1} - \frac{x^3}{3!3} + \frac{x^5}{5!5} - \frac{x^7}{7!7} + \ldots$$

at $x = 5\pi$ to get the correct value 1.6339648 to that many digits.

Repeat this in a script `siTol.m`, but this time let Matlab itself decide when to finish summing based on an allowed tolerance of 0.0000001 that you provide.

Warning: this is an alternating series; use the appropriate error estimate for that.

Also evaluate the terms in a more efficient way this time, by using

$$t_i = -t_{i-2} \frac{x^2}{(i-1)i} \frac{i-2}{i}$$

which allows you to compute the current term `ti` from the previous one.

In this case, print out how many terms Matlab ended up doing, and the approximate and exact values for $\mathrm{Si}(5\pi)$ to 7 decimals. Check that they agree to about 0.0000001. Also print out the number of terms Matlab needed to sum. And also print out the actual error using the exact value 1.633964846102835 (as obtained from `sinint`).

The correct *old* solution is already in the `q6.m` file and `si.m`. Just modify it as requested. (The first $i = 0$ in `si.m` should have been $i = 1$, but it does not make a difference.)

7. Next copy `siTol.m` into `siBest.m` and in `si_best.m` make Matlab keep summing until the accuracy no longer improves. Show exact and approximate values, both up to 16 decimals now, number of terms summed, and the error.

Since the Taylor series result is not as accurate as you would expect, comment on what you think is the reason. Remember that normal Matlab numbers have a relative error of about 10^{-16}. In other words, there are about 16 good digits starting from the first nonzero digit. Also take into account that, since 5π is about 16, the largest term in the series is $(5\pi)^{15}/15!15$ or about 45,000.