Applied Numerical Methods I, 3450:427/527:001, Kreider, Fall 2012

MiniProject 2 – Avoiding Overflow (40 points)
Due date: Tuesday 18 September

Let's investigate the computation of the potential in the bulk electrolyte for our crevice corrosion problem. The potential is given by

$$E(x) = \frac{wE_B + NE_A}{w+N} + \frac{2(E_B - E_A)}{w(w+N)} \sum_{n=1}^{\infty} \frac{\sin^2(\lambda_n N)\left(e^{\lambda_n(x-2L)} + e^{-\lambda)nx}\right)}{\lambda_n^2\left(e^{\lambda_n(M-2L)} + e^{-\lambda)nM}\right)}$$

$$\lambda_n = \frac{n\pi}{w+N}$$

Parameter values are: $w = 0.03$ cm, $M = 1$ cm, $L = 1.2$ cm, $N = 10$ cm, $E_A = -0.3$ V and $E_B = -0.29$ V. The domain for x is $M \le x \le L$.

There are 2 numerical issues. First, the exponentials all have negative argument, so as λ_n gets big (as n increases), they go to zero rapidly, so the terms in the series look like zero over zero. This would be ok only if the series converged rapidly so that you wouldn't need large n values, but it's not clear at the start if this is true.

Part A. Underflow issue. Write a MATLAB code, called CP2A.m, that prints the individual terms of the series from 1 to 2400. With complicated expressions, you should ALWAYS break the computation into intermediate steps:

```
{set parameter values}
for n=1:2400
    lambda =
    num = {exponentials go here}
    num = num*sin(lambda*N)^2;
    den = {exponentials go here}
    den = den*lambda^2;
    frac = num/den;
    fprintf('%d %e %e %e\n',n,num,den,frac)
end
```

Scroll through the results, noting especially the lines around $n = 2162$ and $n = 2378$. Underflow hits the numerator and denominator at different n values. **Submit a copy of your code and the output for lines 2156 through 2162**. You can cut and paste them into a Word document. Do NOT print the entire set of outputs. I use that subset of lines as a quick check to help with grading.

Part B. Fixing the underflow issue. Multiply the numerator and denominator by $\exp(\lambda_n M)$ to recast the potential as

$$E(x) = \frac{wE_B + NE_A}{w+N} + \frac{2(E_B - E_A)}{w(w+N)} \sum_{n=1}^{\infty} \frac{\sin^2(\lambda_n N)\left(e^{\lambda_n(x+M-2L)} + e^{-\lambda_n(x+M)}\right)}{\lambda_n^2\left(e^{\lambda_n(2M-2L)} + 1\right)}$$

Copy CP2A.m into CP2B.m, adjust the expressions for num and den, and rerun. Note that the underflow issue is now resolved for this range of n values. Note that the numerator

likely would still run into underflow, but only for much larger n values that are probably not needed in the summation. The denominator will not have underflow issues. **Again, cut and paste the output for lines 2156 through 2162 into a Word document and submit them with a copy of this code.**

Part C. Convergence issue. Now that we have an appropriate computational form for the terms of the series, let's see what happens when we add them. Make a copy of CP2B.m and call it CP2C.m. Change E_B and x to $E_B = -0.1$, $x = 1$ to exaggerate the effects. The series must be truncated; we'll use Nmax terms, and vary Nmax to see how many terms we might need. Again, we'll build the sum in stages:

```
{set parameter values}
for Nmax=1000:1000:20000
    sum = 0;
    for n=1:Nmax
        lambda =
        num = {exponentials go here}
        num = num*sin(lambda*N)^2;
        den = {exponentials go here}
        den = den*lambda^2;
        frac = num/den;
        sum = sum + frac;
    end
    E = 2*(EB-EA)*sum/w/(w+N) + (w*EB+N*EA)/(w+N);
    fprintf('%d %20.14e\n',Nmax,E)
end
```

Note the format for printing E – we need to see the significant digits to see if it's converged. Run the code with Nmax going up to 20000. The value printed is $E(1)$; it's not quite converged. **Submit a copy of your code and cut and paste the output for Nmax=20000**. Now run the code for higher values of Nmax, say up to 200000. We expect $E(M) = E_A$; notice how many terms it takes to get the series to match -0.1. Determine how many terms (to the nearest thousand) it takes to get the computed value accurate to 4 significant digits (ie, the computation rounds to -1.000). **Put the answer on the same page as your code and the Nmax=20000 result.**