

CISC271
Fall 2005
Homework for week 4
in preparation for quiz 2
Solutions

This homework will give you some practice with Taylor series. Doing these exercises should clarify some of the confusion distinguishing between floating point roundoff error and truncation error.

Let:

$$P_n(x) = \sum_{k=0}^n \frac{f^{(k)}(x_0)}{k!} (x - x_0)^k$$

be the n th Taylor polynomial for f about x_0 . The remainder term or the truncation error will be denoted by $R_n(x)$ and is just the difference $f(x) - P_n(x)$.

1. The infinite series expansion for e^x about 0 is:

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!}$$

- (a) Write a Matlab function that computes the n th Taylor polynomial for e^x

Solution: See expSeriesPlot in Rec p.220

- (b) Matlab has a built in function exp(x) which returns the value e^x . Experiment with your function and determine the value for n where your the absolute difference between exp(1) and your Taylor approximation is equal to zero.

Solution: Using expSeriesPlot and a tolerance of `eps` it takes 19 iterations. Using a tolerance of `realmin` it takes 172 iterations. I could never find a sum of this series where the absolute difference is exactly 0.

2. Consider the cubic polynomial $f(x) = x^3$.

(a) Determine the Taylor polynomial $P_2(x)$ about $x_0 = 0$. What is the value of $P_2(1)$?

Solution: The answer is 0 which is not a very good approximation of 1^3 .

(b) Determine the Taylor polynomial $P_3(x)$ and $P_n(x)$ for all values of n greater than n . What is the value of $P_3(1)$? How does $P_3(x)$ compare to x^3 for any value of x .

Solution: For x^3 at 1 we get 1. In fact for any x the values will always x^3 .

3. The Taylor series expansion of $\cos(x)$ at $x_0 = 0$ is

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots + (-1)^k \frac{x^{2k}}{2k!}$$

I wrote a MATLAB m-file that would obtain an approximation for $\cos(x)$ by expanding the Taylor series to 10 terms. I tried it with a value of π and obtained the answer - 1.0000. I then tried the same function with a value of 13π and got a different answer. However, I know that $\cos(\pi) = \cos(13\pi) = -1$. Try to reproduce these results by writing your own version of this function. Can you explain the strange behaviour?

Solution: The series oscillates about the solutions. The values of $13\pi^{2k}$ get really big and the oscillations get to be "miles" away from the right answer.

4. The fidiff Matlab m-file shown in listing 5.9 of Rec is an interesting tool for exploring the connections between roundoff and truncation errors. I took this m-file as a starting point and made modifications until I could pinpoint the value that gave the smallest relative error. I replaced the logspace line with linspace to get a finer resolution. In the end it appears that the smallest relative error is somewhere around 2.3338×10^{-8} . It's also interesting to look at the plot of values. Depending on the interval and the number of points the plot varies from seemingly random points to points following a definite pattern. You should try this yourselves, and report your findings.

Solution: Here is my fidiff m-file

```
function fidiff(x)
% fidiff First order finite-difference approximation to d/dx of exp(x)
%
% Synopsis: fidiff(x)
%
% Input: x = (optional) value at which the derivative is to be evaluated
%         Default: x = 1
%
%
if nargin<1, x=1; end
fp = exp(x); % Exact value of fprime
h = linspace(2.3336*10^-8,2.3338*10^-8,100); % Column vector of stepsizes
fpfd = (exp(x+h) - exp(x))./h; % Vectorized calculation of fd approximation
```

```

Erel = abs(fpfd - fp)./fp;          % and relative errors
Eabs = abs(fpfd - fp);
fprintf('      h      fp      fpfd      Erel      Eabs\n');
minrel = 1;
for k=1:length(h)
    fprintf('%10.1e %9.5f %9.5f %12.2e %12.2e \n',h(k),fp,fpfd(k),Erel(k), Eabs(k));
    if Erel(k) < minrel, minrel = Erel(k); bestk = k; end;
end
h(bestk)
Erel(bestk)
Eabs(bestk)
%loglog(h,Erel,'+');
plot(h,Eabs,'+');
xlabel('Stepsize, h');    ylabel('Relative error');

```

Here is the plot.

