Table of Contents

3B

```
%Problem 4.15 from textbook- Problem 3 from HW 1- Using a Taylor
 Series
%approximation to predict f(2) for f(x)=ln(x) using base point at x=1

%0th order taylor series term

x0 = 1;
h=1;
taylorSeriesTerms = [];

taylorSeriesTerms(1)= log(x0);
%1st order taylor series term
taylorSeriesTerms(2) = 1/x0;
%2nd order taylor series term
taylorSeriesTerms(3) = -1/(2*x0^2);
%3rd order taylor series term
taylorSeriesTerms(4) = 1/(3*x0^3);
taylorSeriesTerms(5) = -1/(4*x0^4);

%Calculating Errors
errors = [];

errors(1) = 100*(log(2) - taylorSeriesTerms(1) )/log(2);
errors(2) = 100*(log(2) - sum(taylorSeriesTerms(1:2)))/log(2);

for i=1:length(taylorSeriesTerms)
   errors(i) = 100*(log(2) - sum(taylorSeriesTerms(1:i)))/log(2);
end


termNumber = 5;
seriesTerms = [];

for i = 1:termNumber


end
```

Expansion

```
error = [];
errorReduction = [];
```

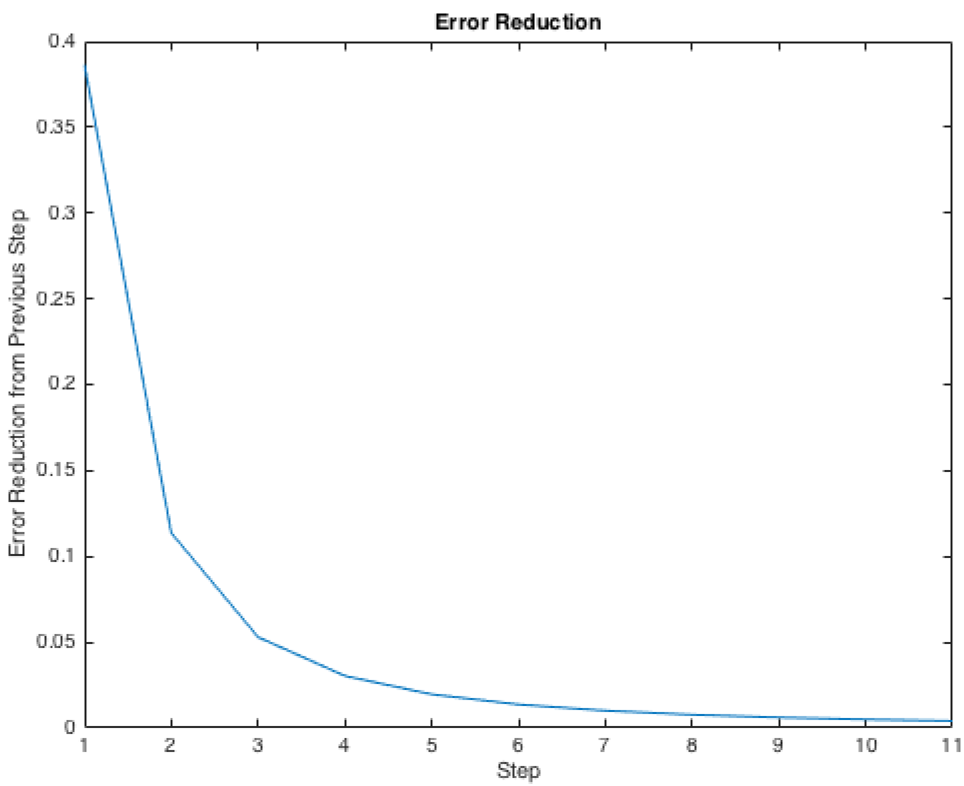
```
approximations = [];

for i = 1:12
    [percentError, approximations(i)] = taylorSeriesLn2fromLn1(i-1);
    error(i) = abs(approximations(i) - log(2) );
end

for i = 1:11
    errorReduction(i) = abs(error(i+1) - error(i));
end

plot(errorReduction)
title('Error Reduction')
ylabel('Error Reduction from Previous Step')
xlabel('Step')

%Notice that as you add terms, the error reduction goes down.
 Therefore,
%adding more and more terms becomes arbitrary at some point.
```



Percent Error Analysis1.3

```
percentError = 10;
order = 0;

while (percentError > 1)
```

```
    [percentError,~]=taylorSeriesLn2fromLn1(order);
    order = order+1;
end

order

%the order above is the necessary order to reduce percent error to 1

while (percentError > 0.1)
    [percentError,~]=taylorSeriesLn2fromLn1(order);
    order = order+1;
end

order
%the order above is the necessary order to reduce percent error to 0.1
```

```
order =

    73


order =

   722
```

Published with MATLAB® R2015b