NAME:

MATH 353: Engineering Mathematics III

Spring'13 Midterm exam #1 (solutions) March 6

1. (10 points) We run the following lines of code

```
>> a=2:0.2:6;
>> b=a([3 5 6]);
```

What is in `b`?

The array `a` is

```
2   2.2   2.4   2.6   2.8   3   3.2   3.4    3.6    ... 6
```

We take the third, fifth and sixth components to build `b`, that is

```
b = [ 2.4   2.8   3 ]
```

2. (10 points) Explain the following two lines of Matlab

```
>> 1e320
ans =
   Inf
>> Inf/Inf
ans =
   NaN
```

Use them to explain why the following two lines give different results. What are they?

```
>> a=(1e120*3e200)/(2e100*1e210)
>> b=(1e120/2e100)*(3e200/1e210)
```

The number 10^{320} exceeds the maximum exponent that is allowed in double precision arithmetic. Therefore it is stored by Matlab as ∞. The division ∞/∞ is an indeterminate and Matlab returns *not a number.*

The first computation gives `Inf/Inf` and therefore the result is `NaN`. The second computation has better balanced parentheses, avoiding multiplication of very large numbers. The result is $0.5 \times 10^{20}/(3 \times 10^{10}) = 1.5 \times 10^{10}$.

3. (10 points) We run the following lines of code

```
>> f = @(x) x.^2./(1+x.^3);
>> f([1 3])
```

What is the result?

We have written a function that allows us to evaluate

$$f(x) = \frac{x^2}{1+x^3}$$

simultaneously in many points (it is vectorized). We then evaluate it in 1 and 3. The result is the vector with components $(\frac{1}{2}, \frac{9}{28})$.

4. (10 points) Consider the following lines that we have taken from a Matlab script

```
x=1;
for it=1:2e3
    x=0.5*(x+3/x);
end
```

Write with mathematical formulas what we just did (not the result!). How many iterations have we performed?

This is the iteration

$$x_1 = 1, \qquad x_{i+1} = \frac{1}{2}\left(x_i + \frac{3}{x_i}\right) \qquad i = 1, \ldots, 2000.$$

5. (10 points) Write the mathematical expression of the following line of code (assume that `n` stores a positive integer)

```
>> x=a+(b-a)/n*(0:n);
```

How many entries does the vector `x` have?

We have built a vector with components

$$a + \frac{b-a}{n}\, i \qquad i = 0, 1, \ldots, n.$$

We have $n = 1$ components in the array.

6. (10 points) We run the following lines of code.

```
>> err =[1.354248688935409,...
         0.229248688935409,...
         0.009139993283235,...
         0.000015733125699,...
         0.000000000046779];
>> trend=err(2:end)./err(1:end-1).^2
trend =
    0.1250    0.1739    0.1883    0.1890
```

In `err` we have stored some errors $|x_i - r|$ for an iteration trying to compute a quantity r. What can you say about convergence of that iteration? If `err(i)` stores

$e_i = |x_i - r|$, the line where we compute the array `trend` corresponds to the mathematical expression

$$\frac{e_{i+1}}{e_i^2}.$$

This quantity seems to converge to a number (its value is not important), which shows *quadratic* convergence of the corresponding iteration.

7. (15 points) An iteration

$$x_0 \text{ given}, \qquad x_{i+1} = g(x_i), \qquad i = 1, 2, 3, 4$$

has been applied to compute a fixed point $r = g(r)$. We have also computed the errors

$$e_i = |x_i - r|, \qquad i = 0, \ldots, 4.$$

These are the quantities we obtained

```
err1 =[0.020746217944410,...
    0.006076689632185,...
    0.001779827797216,...
    0.000521299658575,...
    0.000152685139140];
```

Show that the iteration is converging linearly. (You'll need to compute four divisions for this.) Assuming that g is differentiable, what can we say about $|g'(r)|$?

If we compute the quotients of consecutive errors as follows

$$\frac{0.006076689632185}{0.020746217944410} \approx 0.292906 \qquad \frac{0.001779827797216}{0.006076689632185} \approx 0.292894$$

$$\frac{0.000521299658575}{0.001779827797216} \approx 0.292893 \qquad \frac{0.000152685139140}{0.000521299658575} \approx 0.292893$$

we can see that the iteration is converging *linearly at a rate* ≈ 0.29289. If this is a fixed point iteration for the function g and r is the fixed point, then $|g'(r)| \approx 0.29289$.

8. (15 points) Compute two iterations of the *secant method* applied to the function

$$f(x) = x^2 - 5$$

starting at $x_0 = 0$ and $x_1 = 1$. Write the result of the final computation in *floating point* form $0.m_1....10^n$, with five significant digits.

The formula for an iteration of the secant method is

$$x_{i+1} = x_i - f(x_i)\frac{x_i - x_{i-1}}{f(x_i) - f(x_{i-1})}.$$

The first iteration gives $x_2 = 5$. The second one, computed with x_1 and x_2, gives $x_3 = \frac{10}{6} \approx 0.16667 \times 10^1$.

9. (20 points) Here's a piece of code that is almost finished

```
function [r,hist]=unknown(f,fp,x0,tol,itmax)

% [r,hist]=unknown(f,fp,x0,tol,itmax)
% Input:
%    Explanations
% Output:
%     Explanations

r = x0; hist = r;
for i=1:itmax
    d=f(r)/fp(r);
    r=r-d;
    hist=[hist r];
    if abs(d) < tol
        return
    end
end
display('Message')    % Line to be completed
return
```

Three questions on this piece of code:

(a) What method does this function correspond to? What is it that we are trying to solve?

(b) What (exactly) would happen if we type `help unknown`?

(c) The `Message'` line is to be completed. What is the message you should write instead on the word `Message`. It has to make sense in the context that we are trying to use this.

(a) It is Newton's method to compute roots of functions. (b) We would get all the commented lines right after the header of the function

```
% [r,hist]=unknown(f,fp,x0,tol,itmax)
% Input:
%    Explanations
% Output:
%     Explanations
```

(c) If we have reached this line, then we have iterated all the way to the maximum number of iterations but the stopping criterion has never been satisfied. The program has to wanr the user that this is so:

```
Maximum number of iterations reached without convergence
```

10. (20 points) We want to interpolate in the following points

$$(1, \tfrac{1}{2}), \qquad (2, -\tfrac{1}{4}), \qquad (5, 1)$$

with a polynomial of degree at most 2.

(a) Write down (in all detail, but without simplifying) the Lagrange formula to interpolate the points.

Here's our code to evaluate the interpolating polynomial in a collection of points.

```
function Y=evaluatelagrange(x,y,X)

% Y=evaluatelagrange(x,y,X)
%
% Input:
%      x    : vector with n+1 entries (points x_i)
%      y    : vector with n=1 entires (points y_i)
%      X    : any vector where we want to evaluate the interpolant
% Output:
%      Y    : values of the interpolant P at the points X

Y=0;
n=length(x)-1;
for k=1:n+1
    c=y(k);
    for j=[1:k-1 , k+1:n+1]      % list of 1 to n=1 w/o k
        c=c.*(X-x(j))./(x(k)-x(j));
    end
    Y=Y+c;
end
return
```

(b) Using `evaluatelagrange`, write down what would have to do to evaluate in the points $\{1, 1.1, 1.2, 1.3, ..., 2\}$ the polynomial that interpolates $(1, \frac{1}{2}), (2, -\frac{1}{4}), (5, 1)$.

For (a), the solution is:

$$p(x) = \tfrac{1}{2}\,\frac{(x-2)(x-5)}{(1-2)(1-5)} + (-\tfrac{1}{4})\frac{(x-1)(x-5)}{(2-1)(2-5)} + 1\,\frac{(x-1)(x-2)}{(5-1)(5-2)}$$

For (b) we just need to use the program in the following way

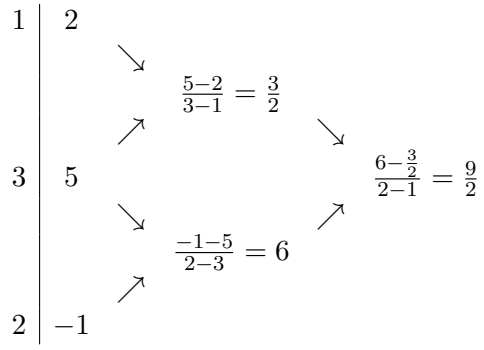
```
evaluatelagrange([1 2 5],[1/2 -1/4 1],1:0.1:2)
```

11. (20 points) Compute the Newton divided differences and write down the Newton formula for the interpolating polynomial for the following points

$$(1, 2), \qquad (3, 5), \qquad (2, -1).$$

(The points have to be used in the given order.)

This is the table of divided differences:



This is the interpolation polynomial given in Newton's form

$$p(x) = 2 + \tfrac{3}{2}(x-1) + \tfrac{9}{2}(x-1)(x-3).$$