MAD 3401 — Introductory Numerical Analysis

Section 1, Spring 1994.

Instructor: Bellenot.

The good doctor's Office: 002-B Love, Office Hours: MWF 9:20–10:00 or by appointment.

Eligibility: A grade of C- or better in Calculus 2 (MAC 3312) and knowledge of a high level programming language such as C, Pascal or FORTRAN.

Text: K. Atkinson *Elementary Numerical Analysis* 2^{nd} Edition.

Coverage: Chapters 1 - 7.

Final: At 10-12 Thursday Apr 28, 1994.

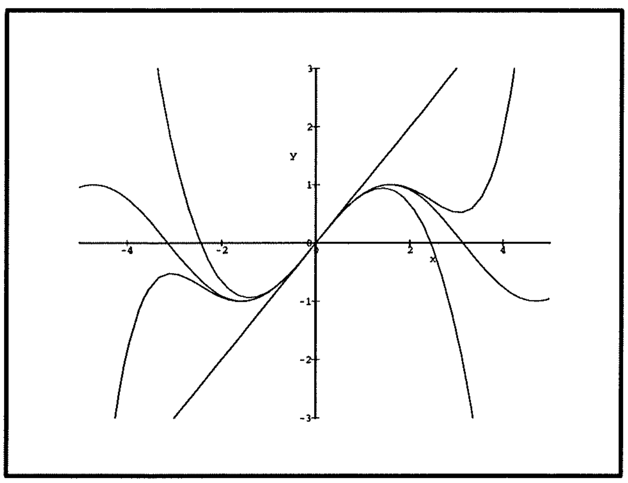
Tests: (3) Tentatively at Jan 26, Feb 23(Mar 2?) and Apr 13. No Makeup tests.

Programming Projects: Due every Wednesday (except test days). Many of the projects will use Maple.

Grades: 90% A, 80% B, 70% C, 60% D.

Relative Weights F = 2T and T = P (F is 1/3, each T is 1/6 and P is 1/6).

Homework and Attendance are required.



Maple Input:

```
p1(x) := convert(taylor ( sin(x), x=0, 2), polynom);
p3(x) := convert(taylor ( sin(x), x=0, 4), polynom);
p5(x) := convert(taylor ( sin(x), x=0, 6), polynom);
plot( {sin(x), p1(x), p3(x), p5(x)}, x=-5..5, y=-3..3);
```

```
> q:=x->b0+b1*x+b2*x^2+b3*x^3;

                                                 2         3
                          q := x -> b0 + b1 x + b2 x  + b3 x
--------------------------------------------------------------------------------
> r:=unapply(diff(q(x), x),x);

                                                           2
                            r := x -> b1 + 2 b2 x + 3 b3 x
--------------------------------------------------------------------------------
> f:=x->E^x;

                                               x
                                     f := x -> E
--------------------------------------------------------------------------------
> g:=unapply(diff(f(x),x),x);

                                               x
                                     g := x -> E
--------------------------------------------------------------------------------
> t:=unapply(convert(taylor(f(x),x=0,4),polynom),x);

                                                  2         3
                           t := x -> 1 + x + 1/2 x  + 1/6 x
--------------------------------------------------------------------------------
> solve({q(0)=f(0),r(0)=g(0),q(2)=f(2),r(2)=g(2)}, {b0,b1,b2,b3});

                                                                    2
                     {b0 = 1, b1 = 1, b3 = 1/2, b2 = - 7/4 + 1/4 E }
--------------------------------------------------------------------------------
> assign(");
--------------------------------------------------------------------------------
> q(x);

                                                 2   2         3
                         1 + x + (- 7/4 + 1/4 E ) x  + 1/2 x
--------------------------------------------------------------------------------
> plot({t(x)-f(x),q(x)-f(x)}, x=-0.5..2.5);
--------------------------------------------------------------------------------
>
```

```
#include <stdio.h>

typedef float Real;

typedef Real DecFun(int);

class DecFunSums
{
private:
        DecFun * fun;
        DecFun * sum;
public:
        DecFunSums ( DecFun * f, DecFun * g ) { fun = f; sum = g; }
        Real sumLS ( int n );
        Real sumSL ( int n );
        Real sumT ( int n );
};

Real DecFunSums::sumSL ( int n )
{
        int m;
        Real answer = 0.0;

        for ( m = n; m > 0; m-- )
                answer +=  fun ( m );

        return answer;
}

Real DecFunSums::sumLS ( int n )
{
        int m;
        Real answer = 0.0;

        for ( m = 1; m <= n; m++ )
                answer +=  fun ( m );

        return answer;
}

Real DecFunSums::sumT ( int n )
{
        return sum ( n );
}

inline Real a ( int n ) { return 1.0/((Real) (n*(n+1))); }
inline Real aAns ( int n ) { return ((Real) n)/((Real) (n+1)); }

int indices[] = {10, 50, 100, 500, 1000};

main()
{
        DecFunSums A(a, aAns);

        for ( int i = 0; i < 5; i++ )
                printf ( "%d    %.91f    %.91f    %.91f    %.91f\n",
                        indices[i], A.sumLS ( indices[i] ),
                        A.sumSL ( indices[i] ), A.sumT ( indices[i] ), a(indices[i]) );

        for ( i = 0; i < 20; i++ )
                printf ( "%d %.91f %.91f %.91f\n", i, A.sumLS ( i ),
                        A.sumSL ( i ), A.sumT ( i ) );
}
```

MAD 3401 Intro Numerical Math **Test 2** 2 Mar 1994

Show **ALL** work for credit; be neat; and use only **ONE** side of each page of paper.

1. Find the error estimates:
   A. Calculate the error in the approximation $x_A \approx x_T$, if $x_T = 28.254$ and $x_A = 28.271$.
   B. Calculate the relative error for the numbers in Part A.
   C. Calculate the number of significant digits for the numbers in Part A.
   D. Suppose x_T and y_T round to $x = 1.223$ and $y = 1.14$, find the smallest interval estimate for the true value of $x * y$.
   E. For the numbers in Part D, find the smallest interval estimate for the true value of $x - y$.

2. Use Newton's method to find the cube root of 2.
   A. Write a polynomial function $f(x)$ so that the root of $f(x)$ is $2^{\frac{1}{3}}$.
   B. Write the Newton's formula for this particular $f(x)$ which yields x_{n+1} in terms of x_n
   C. Starting from $x_0 = 2$ do two iterations of the formula in B.

3. Consider the function $g(x) = (x^2 - 5)/4$
   A. Find both fix points of $g(x)$.
   B. For each fix point α determine if iterates of the form $x_{n+1} = g(x_n)$ will converge to α provided x_0 starts close enough to α.

4. The initial interval for the bisection method has length $b - a = 2^{-1}$, how many iterations are necessary to get the interval to length 10^{-9}? Suppose the same problem (which has root α,) when using Newton's method, starts with $|\alpha - x_0| = 2^{-1}$, and the estimate $\alpha - x_{n+1} = (\alpha - x_n)^2$ is true. How many iterations of Newtons method are needed to get within 10^{-9}?

5. Derive the secant method, let $f(x)$ be given.
   A. Given x_n and x_{n-1} find the equation of the secant line passing through the points $(x_n, f(x_n))$ and $(x_{n-1}, f(x_{n-1}))$.
   B. Solve your equation above for x_{n+1}.

6. Suppose your computer does '4 digit decimal arithmetic' and it rounds. Suppose $x_1 = y_1 = 0.2; x_i = 0.00004$ for $i > 1$ and $y_i = 0.00007$ for $i > 1$. Compute $\sum_1^{1001} x_i$ and $\sum_1^{1001} y_i$ using your computer both by LS (Largest to Smallest) and SL. Give the errors from the true sums.

7. Suppose $f(x)$ is a fuction with a zero at $x = \alpha$ and $f'(\alpha) \neq 0$. Suppose x_n is obtained by Newton's method starting from some x_0 which is near α. Show $(\alpha - x_{n+1}) = M(\alpha - x_n)^2$ by doing the following.
   A. First expand $f(x)$ as a Taylor polynomial (of degree one) with remainder about $x = x_n$.
   B. Substitute $x = \alpha$, use $f(\alpha) = 0$, and divide by $f'(x_n)$.
   C. Use Newton's formula for x_{n+1} to obtain your answer.
   D. Find M.

8. Write a Maple procedure `bisect` which uses the bisection method to find a root. The procedure repeats the bisection until it find an answer within 10^{-6} of the root. The procedure bisect is called with parameters (f, a, b) where $f(x)$ is the function in question, $a < b$ are real numbers so that $f(a)*f(b) < 0$. If you need it, the syntax of the Maple if statement is given below, (statements between [ ] are optional).

```
if conditional-expression then statement-sequence
[ elif conditional-expression then statement-sequence ]
[ else statement-sequence ]
fi
```

MAD 3401 Intro Numerical Math **Test 3** 13 Apr 1994

Show **ALL** work for credit; be neat; and use only **ONE** side of each page of paper.

1. Consider the integral $I(f) = \int_0^1 f(x)dx$.
   A. Find the estimate $T_4(f)$ for $I(f)$.
   B. Find the estimate $S_4(f)$ for $I(f)$.

2. Consider the data points $\{(1,2),(2,1),(3,3)\}$
   A. Find the piecewise linear interpolating polynomial for the data.
   B. Find the quadratic interpolating polynomial for the data.

3. Write the requested polynomial.
   A. The Lagrange interpolation basis polynomial which is zero at $x = 2, x = 3$ and $x = 5$ but one at $x = 4$.
   B. The Chebyshev polynomials $T_n(x)$ for $n = 0, 1, 2$ and 3 in terms of x.

4. Consider the integral $I(f) = \int_0^1 f(x)dx$.
   A. Bound the error $E_n^T(f)$ if the trapezoidial rule with $n = 10$ is used to estimate $I(f)$.
   B. Bound the error $E_n^S(f)$ if Simpson's rule with $n = 10$ is used to estimate $I(f)$.

5. A minimax polynomial $p(x)$ of degree 5 is use to approximate $\sin x$ for $0 \le x \le \pi/4$. Estimate the maximum error $|\sin x - p(x)|$. The estimates $\pi^2 < 10$ and $2^{10} \doteq 10^3$ may be used.

6. Suppose $t(x) = -t(-x)$ and for positive $x, t(x) = \pi/2 - t(1/x)$. Using these identites what smaller interval can be used to approximate $t(x), -\infty < x < \infty$? Give an algorithm for reducing the evaluation of $t(x)$ to this smaller interval, assume the polynomial $p(x)$ is a good approximation to $t(x)$ on this smaller interval.

7. Find the cubic polynomial $p(x)$ which satisfies $p(x_0) = p(x_1) = p'(x_1) = 0$, but $p'(x_0) = 1$.

8. Show $f[x_0, x_1] - f'((x_0 + x_1)/2) \doteq h^2 f'''(z)/6$ Hints: Let z be the midpoint and let $z + h = x_1, z - h = x_0$; and expand f in a Taylors series for $f(z \pm h)$ about z.

```c
#include <stdio.h>

union {
int i;
char c[4];
} j;

union {
float f;
char c[4];
} g;

union {
double d;
char c[8];
} x;

void byteInHex(char c)
{
        char top = '0' + ((c >> 4) & 0xf);
        char bot = '0' + (c & 0xf);

        if ( top >  '9' )
                top = top - ':' + 'a';

        if ( bot >  '9' )
                bot = bot - ':' + 'a';

        printf ( " %c%c", top, bot );
}

void print_j(void )
{
        int k;

        printf ( "%d", j.i );
        for ( k = 0; k < 4; k++ )
                byteInHex ( j.c[k] );
        printf ( "\n" );
}

void print_g(void )
{
        int k;

        printf ( "%f", g.f );
        for ( k = 0; k < 4; k++ )
                byteInHex ( g.c[k] );
        printf ( "\n" );
}

void print_x(void )
{
        int k;

        printf ( "%lf", x.d );
        for ( k = 0; k < 8; k++ )
                byteInHex ( x.c[k] );
        printf ( "\n" );
}


main()
{
        int i;
```

```
        float half = 1.0/2.0;
/*
        printf ( "Integer\n" );
        for ( i = 0; i < 32; i++ )
        {
                j.i = 1 << i;
                print_j();
        }
*/
        printf ( "Float\n" );
        g.f = 1.55;
        print_g();
        g.f = 3.55;
        print_g();
        printf ( "Double\n" );
        x.d = 1.55;
        print_x();
        x.d = 3.55;
        print_x();
/*
        for ( i = 0; i < 32; i++ )
        {
                g.f += half;
                print_g();
                half /= 2.0;
        }
*/
/*
        printf ( "Double\n" );
        for ( i = 0; i < 32; i++ )
        {
                x.d = 1 << i;
                print_x();
        }
*/
}
```