APPENDIX

The following Maple code solves nonlinear boundary value problems of the form

$$y''(x) = f(x, y), \quad a \le x \le b,$$
$$y(a) = 0, \ y(b) = 0.$$

Usage : nsm().

Input : Nonlinear function $f(x, y)$; endpoints a, b; number of subintervals N; initial guess TK; tolerance TOL; maximum number of iterations NN.

Output : Approximations y(x), y′(x); graph of y(x).

```
nsm := proc()
Digits :=100;
global  XX, Y, YP, sequence, graph;
local   X, F, FY, FYP, A, B, TK, N, TOL, NN, OK, H, K, U1, U2,T, K11,
        K12, K21, K22, K31, K32, K41, K42, i;
printf("\ n This is the Nonlinear Shooting Method.\ n");
printf("Use for solve the BVP in the form\ n");
printf("          y'' = f(x, y)\ n");
printf("       y(a) = 0, y(b) = 0.\ n\ n");
printf("Input the function f(x, y) in terms of x, y.\n");
printf("For example:      (32+2 * x^3-y)/8\ n");
        F := scanf("%a")[1];
        FY := diff(F, y);
        FYP := diff(F, z);
        F := unapply(F, x, y, z);
        FY := unapply(FY, x, y, z);
```

```
        FYP := unapply(FYP, x, y, z);
OK := FALSE;
while OK = FALSE do
        printf("Input left and right endpoints separated by blank.\ n");
                A := scanf("%e")[1];
                B := scanf("%e")[1];
        if A ≥ B then
                printf("Left endpoint must be less than right endpoint.\ n");
        else
                OK := TRUE;
        fi;
od;
printf("Input initial guess TK\ n");
        TK := scanf("%f")[1];
OK := FALSE;
while OK = FALSE do
        printf("Input an integer > 1 for the number of subintervals.\ n");
                N := scanf("%d")[1];
        if N ≤ 1 then
                printf("Number must exceed 1.\ n");
        else
                OK := TRUE;
        fi;
od;
OK := FALSE;
while OK = FALSE do
        printf("Input positive tolerance.\ n");
                TOL := scanf("%f")[1];
        if TOL ≤ 0 then
                printf("Tolerance must be positive.\ n");
        else
```

```
                OK := TRUE;
        fi;
od;
OK := FALSE;
while OK = FALSE do
        printf("Input maximum number of iterations.\ n");
                NN := scanf("%d")[1];
        if NN ≤ 0 then
                printf("Must be positive integer.\ n");
        else
                OK := TRUE;
        fi;
od;
H := (B−A)/N;
K := 1;
OK := FALSE;
while K ≤ NN and OK = FALSE do
        Y[0] := 0;
        YP[0] := TK;
        U1 := 0;
        U2 := 1;
        for i from 1 to N/2 do
            X := A+(i−1)*H;
            T := X+0.5*H;
            K11 := H*YP[i−1];
            K12 := H*F(X, Y[i−1], YP[i−1]);
            K21 := H*(YP[i−1]+0.5*K12);
            K22 := H*F(T, Y[i−1]+0.5*K11, YP[i−1]+0.5*K12);
            K31 := H*(YP[i−1]+0.5*K22);
            K32 := H*F(T, Y[i−1]+0.5*K21, YP[i−1]+0.5*K22);
            K41 := H*(YP[i−1]+K32);
```

```
        K42 := H*F(X+H, Y[i−1]+K31, YP[i−1]+K32);
        Y[i] := Y[i−1]+(K11+2*(K21+K31)+K41)/6;
        YP[i] := YP[i−1]+(K12+2*(K22+K32)+K42)/6;
        K11 := H*U2;
        K12 := H*(FY(X,Y[i−1],YP[i−1])*U1+FYP(X, Y[i−1], YP[i−1])*U2);
        K21 := H*(U2+0.5*K12);
        K22 := H*(FY(T, Y[i−1], YP[i−1])*(U1+0.5*K11)+
                FYP(T, Y[i−1], YP[i−1])*(U2+0.5*K21));
        K31 := H*(U2+0.5*K22);
        K32 := H*(FY(T, Y[i−1], YP[i−1])*(U1+0.5*K21)+
                FYP(T, Y[i−1], YP[i−1])*(U2+0.5*K22));
        K41 := H*(U2+K32);
        K42 := H*(FY(X+H, Y[i−1], YP[i−1])*(U1+K31)+
                FYP(X+H, Y[i−1], YP[i−1])*(U2+K32));
        U1 := U1+(K11+2*(K21+K31)+K41)/6;
        U2 := U2+(K12+2*(K22+K32)+K42)/6;
od;
if abs(YP[N/2]) < TOL then
        for i from N/2+1 to N do
                X := A+(i−1)*H;
                T := X+0.5*H;
                K11 := H*YP[i−1];
                K12 := H*F(X, Y[i−1], YP[i−1]);
                K21 := H*(YP[i−1]+0.5*K12);
                K22 := H*F(T, Y[i−1]+0.5*K11, YP[i−1]+0.5*K12);
                K31 := H*(YP[i−1]+0.5*K22);
                K32 := H*F(T, Y[i−1]+0.5*K21, YP[i−1]+0.5*K22);
                K41 := H*(YP[i−1]+K32);
                K42 := H*F(X+H, Y[i−1]+K31, YP[i−1]+K32);
                Y[i] := Y[i−1]+(K11+2*(K21+K31)+K41)/6;
                YP[i] := YP[i−1]+(K12+2*(K22+K32)+K42)/6;
```

```
        od;
        OK := TRUE;
else
        TK := TK-(YP[N/2]/U2);
        K := K+1;
fi;
od;
if OK = TRUE then
        printf("NONLINEAR SHOOTING METHOD\ n\ n");
        printf("        i        x(i)        y(x)        y'(x)\ n");
        for i from 0 to N do
            XX[i] := A+i*H;
            printf("%3d %5.5f %10.16f %10.16f\ n", i, XX[i], Y[i], YP[i]);
        od;
        printf("\ n Convergence in %d iterations\ n", K);
        printf("with t = %14.16f\ n\ n", TK);
        sequence := [seq([XX[i], Y[i]], i=0..20)];
        graph := plot([sequence], title="Graph of solution y(x)");
        return graph;
else
        printf("Method failed after %d iterations\ n", NN);
        return OK;
fi;
end proc:
```

VITA

Name	:	Mr. Somkid Inthap
Date of Birth	:	February 8, 1977
Place of Birth	:	Chiang Rai, Thailand
Institutions Attended	:	B.Ed. (Mathematics) from Rajabhat Institute Chiang Mai in 2000, Chiang Mai, Thailand
Experience	:	Teacher at Rajamangala Institute of Technology, Chiang Rai Campus since 2000
Scholarship	:	U.D.C. from Burapa University, 2002