Appendix A

Code from Maple, MATLAB, and Fortran for Example 1 from "Extending Explicit and Linearly Implicit ODE Solvers for Index-1 DAEs."

1. Maple Code (using dsolve's rkf45)

Use y1, y2, etc. for all differential variables and z1, z2, etc. for all algebraic variables

```
> restart;
> with(plots):
```

Enter all ODEs in eqode

```
> eqode:=[diff(y1(t),t)=-y1(t)^2+z1(t)];
```

$$eqode := \left[\frac{d}{dt} \mathrm{y1}(t) = -\mathrm{y1}(t)^2 + \mathrm{z1}(t) \right]$$

Enter all AEs in eqae

```
> eqae:=[cos(y1(t))-z1(t)^0.5=0];
```

$$eqae := [\cos(\mathrm{y1}(t)) - \mathrm{z1}(t)^{0.5} = 0]$$

Enter all initial conditions for differential variables in icodes

```
> icodes:=[y1(0)=0.25];
```

$$icodes := [\mathrm{y1}(0) = 0.25]$$

Enter all intial conditions for algebraic variables in icaes

```
> icaes:=[z1(0)=0.8];
```

$$icaes := [\mathrm{z1}(0) = 0.8]$$

Enter parameters for perturbation value (epsilon), switch function (q and tint), and runtime (tf)

```
> pars:=[epsilon=0.1,q=1000,tint=1,tf=5];
```

$$pars := [\varepsilon = 0.1, q = 1000, tint = 1, tf = 5]$$

Choose solving method (1 for explicit, 0 for implicit)

```
> Xexplicit:=1:
```

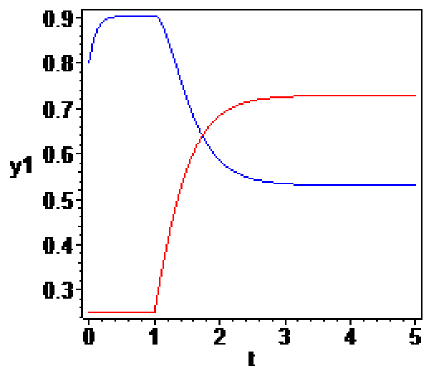
Standard solver requires IC z(0)=0.938791 or else it will fail

```
> solx:=dsolve({eqode[1],eqae[1],icodes[1],icaes[1]},numeric):
Error, (in dsolve/numeric/DAE/checkconstraints) the initial conditions do not satisfy the algebraic
constraints error = .745e-1, tolerance = .559e-6, constraint = cos(y1(t))-z1(t)^.5000000000000000000000
> ff:=subs(pars,1/2+1/2*tanh(q*(t-tint)));
```

$$ff := \frac{1}{2} + \frac{1}{2} \tanh(1000\, t - 1000)$$

```
> NODE:=nops(eqode):NAE:=nops(eqae):
> for XX from 1 to NODE do
> EQODE||XX:=lhs(eqode[XX])=rhs(eqode[XX])*ff:
> end do:
> for XX from 1 to NAE do
> EQAE||XX:=subs(pars,-epsilon*(diff(rhs(eqae[XX])-lhs(eqae[XX]),t))=rhs(eqae
> [XX])-lhs(eqae[XX])):
> end do:
> Dvars1:={seq(diff(z||x(t),t)=D||x,x=1..NAE)}:
> Dvars2:={seq(rhs(Dvars1[x])=lhs(Dvars1[x]),x=1..NAE)}:
> icsn:=seq(subs(y||x(0)=y||x(t),icodes[x]),x=1..NODE),seq(subs(z||x(0)=
> z||x(t),icaes[x]),x=1..NAE):
> for j from 1 to NAE do
> EQAEX||j:=subs(Dvars1,eqode,icsn,Dvars2,lhs(EQAE||j))=rhs(EQAE||j):
```

```
> end do:
> Sys:={seq(EQODE||x,x=1..NODE),seq(EQAEX||x,x=1..NAE),seq(icodes[x],x=1
> ..NODE),seq(icaes[x],x=1..NAE)}:
> if Xexplicit=1 then
> sol:=dsolve(Sys,numeric):
> else
> sol:=dsolve(Sys,numeric,stiff=true,implicit=true):
> end if:
# Plotting Results
> for XX from 1 to NODE do
> a||XX:=odeplot(sol,[t,y||XX(t)],0..subs(pars,tf),color=red):
> end do:
> for XX from NODE+1 to NODE+NAE do
> a||XX:=odeplot(sol,[t,z||(XX-NODE)(t)],0..subs(pars,tf),color=blue):
> end do:
> display(seq(a||x,x=1..NODE+NAE),axes=boxed);
```



End Maple Code

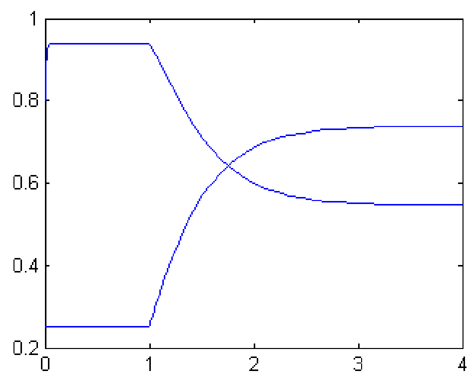
2. MATLAB code (using ode15s)

Example 1 has been converted into a useable form for ode15s

```
function CCS3s
clear
clf
clc
hold off
tsp = 4;
tspan=[0 tsp];
%Extra dummy variable y(2)=dy/dt has been added so that y(1)=y; y(2)=dy/dt;
%y(3)=z
y0 = [0.25,0,0.8];
Nels = 3;
M = [1 0 0;0 0 0;0 0 1;];
options=odeset('Mass',M);
[T,Y]=ode15s(@MEQS,tspan,y0,options);
hold on
for i=1:2:Nels
plot(T,Y(:,i));
end
return

function [res]=MEQS(t,y)
tj=1;
```

```
q=1000;
epsilon=0.01;
ff=(1/2+1/2*tanh(q*(t-tj)));
%Converting the variables from Example 1, y->y(1), z->y(3), and dy/dt->y(2)
%y(2) must be used because ode15s must be of the form Mf(t,y')=f(t,y)
%Based on the Mass Function, M, Eq1 will equal the derivative of y(1), Eq2
%will equal zero, and Eq3 will equal the derivative of y(3)
Eq1=y(2);
Eq2=y(2)-(-y(1)^2+y(3))*ff;
Eq3=-2*y(2)*(y(3)^0.5)*sin(y(1))+2*cos(y(1))*(y(3)^0.5)/epsilon-2*y(3)/epsilon;
res = [Eq1;Eq2;Eq3;];
return
```



End Matlab Code

3. Fortran Code (using RODAS solver which is a Rosenbrock method solver)

In order to run the Fortran driver, you will also need download the RODAS solver, DECSOL linear algebra routines, and DC_DECSOL subroutines which are available free at
http://www.unige.ch/~hairer/software.html
This code was compile using Compaq Visual Fortran 6

```
C * * * * * * * * * * * * * * * * * * * * * * * * *
C --- DRIVER FOR ROSENBROCK CODE RODAS
C * * * * * * * * * * * * * * * * * * * * * * * * *
c link dr_rodas rodas decsol dc_decsol
c link dr_rodas rodas lapack lapackc dc_lapack
        IMPLICIT REAL*8 (A-H,O-Z)
C --- PARAMETERS FOR RODAS (FULL JACOBIAN)
        PARAMETER (ND=2,LWORK=6*ND*ND+14*ND+20,LIWORK=3*ND+20)
C --- DECLARATIONS
        DIMENSION Y(ND),WORK(LWORK),IWORK(LIWORK)
        EXTERNAL FEQN,JAC,SOLOUT,MAS,FIC
C --- DIMENSION OF THE SYSTEM
        N=2
C --- PROBLEM IS AUTONOMOUS
        IFCN=0
C --- COMPUTE THE JACOBIAN ANALYTICALLY
        IJAC=1
C --- JACOBIAN IS A FULL MATRIX
        MLJAC=N
C --- DIFFERENTIAL EQUATION IS IN EXPLICIT FORM
        IMAS=1
```

```fortran
      MLMAS=N
C --- OUTPUT ROUTINE IS USED DURING INTEGRATION
      IOUT=1
C --- INITIAL VALUES
      X=0.0D0
      CALL FIC(N,Y)
C --- ENDPOINT OF INTEGRATION
      XEND=5.0D0
C --- REQUIRED TOLERANCE
      RTOL=1.0D-6
      ATOL=1.0D-6
      ITOL=0
C --- INITIAL STEP SIZE
      H=1.0D-6
C --- SET DEFAULT VALUES
      DO 10 I=1,20
      IWORK(I)=0
  10  WORK(I)=0.D0
C --- CALL OF THE SUBROUTINE RODAS
      CALL RODAS(N,FEQN,IFCN,X,Y,XEND,H,
     &                  RTOL,ATOL,ITOL,
     &                  JAC,IJAC,MLJAC,MUJAC,FVPOL,IDFX,
     &                  MAS,IMAS,MLMAS,MUMAS,
     &                  SOLOUT,IOUT,
     &                  WORK,LWORK,IWORK,LIWORK,RPAR,IPAR,IDID)
C --- PRINT FINAL SOLUTION
      WRITE (6,99) X,Y(1),Y(2)
 99   FORMAT(1X,'X =',F5.2,'    Y =',2E18.10)
C --- PRINT STATISTICS
      WRITE (6,90) RTOL
 90   FORMAT('       rtol=',D8.2)
      WRITE (6,91) (IWORK(J),J=14,20)
 91   FORMAT(' fcn=',I5,' jac=',I4,' step=',I4,
     &        ' accpt=',I4,' rejct=',I3,' dec=',I4,
     &        ' sol=',I5)
      STOP
      END
C
      SUBROUTINE SOLOUT (NR,XOLD,X,Y,CONT,LRC,N,RPAR,IPAR,IRTRN)
C --- PRINTS SOLUTION
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION Y(N),CONT(LRC)
      COMMON /INTERN/XOUT
      IF (NR.EQ.1) THEN
         WRITE (6,99) X,Y(1),Y(2),NR-1
         XOUT=0.2D0
      ELSE
         IF (X.GE.XOUT) THEN
            Y1=CONTRO(1,XOUT,CONT,LRC)
            Y2=CONTRO(2,XOUT,CONT,LRC)
            WRITE (6,99) XOUT,Y1,Y2,NR-1
            XOUT=XOUT+0.2D0
         END IF
      END IF
 99   FORMAT(1X,'X =',F5.2,'    Y =',2E18.10,'    NSTEP =',I4)
      RETURN
      END
```

```fortran
C

      SUBROUTINE FEQN(N,X,Y,F,RPAR,IPAR)
!================================================================
      IMPLICIT REAL*8 (A-H,O-Z)

      DIMENSION Y(N),F(N)

      F(1) = (0.5D0+0.5D0*tanh(1000.D0*X-1000.D0))*(-1.D0*Y(1)**2+Y(2))
      F(2) = cos(Y(1))-1.D0*Y(2)**0.5D0

      RETURN
      END


      SUBROUTINE JAC(N,X,Y,DFY,LDFY,RPAR,IPAR)
!================================================================
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION Y(N),DFY(LDFY,N)

      DFY(1,1) = -2.D0*(0.5D0+0.5D0*tanh(1000.D0*X-1000.D0))*Y(1)
      DFY(1,2) = 0.5D0+0.5D0*tanh(1000.D0*X-1000.D0)
      DFY(2,1) = -sin(Y(1))
      DFY(2,2) = -0.5D0/Y(2)**0.5D0

      RETURN
      END


      SUBROUTINE MAS(N,AM,LMAS,RPAR,IPAR)
!================================================================
      IMPLICIT REAL*8 (A-H,O-Z)
      DOUBLE PRECISION AM(LMAS,N),Y(N)

      AM(1,1) = 1
      AM(1,2) = 0
      AM(2,1) = 0.247403959254523D-5
      AM(2,2) = 0.52704627669473D-5

      RETURN
      END


      SUBROUTINE FIC(N,Y)
!================================================================
      IMPLICIT REAL*8 (A-H,O-Z)
      DOUBLE PRECISION Y(N)

      Y(1) = 0.25D0
      Y(2) = 0.8D0

      RETURN
      END
```

```
X = 0.00    Y =  0.2500000000E+00  0.8000000000E+00    NSTEP =   0
X = 0.20    Y =  0.2500000000E+00  0.9387912809E+00    NSTEP =  25
X = 0.40    Y =  0.2500000000E+00  0.9387912809E+00    NSTEP =  26
X = 0.60    Y =  0.2500000000E+00  0.9387912809E+00    NSTEP =  27
X = 0.80    Y =  0.2500000000E+00  0.9387912809E+00    NSTEP =  28
X = 1.00    Y =  0.2503026871E+00  0.9386461326E+00    NSTEP =  55
X = 1.20    Y =  0.4061356666E+00  0.8439243822E+00    NSTEP =  87
X = 1.40    Y =  0.5215630910E+00  0.7517666333E+00    NSTEP =  91
X = 1.60    Y =  0.6007188478E+00  0.6805114595E+00    NSTEP =  94
X = 1.80    Y =  0.6525172790E+00  0.6313223198E+00    NSTEP =  96
X = 2.00    Y =  0.6854701462E+00  0.5992640819E+00    NSTEP =  98
X = 2.20    Y =  0.7060826415E+00  0.5789833505E+00    NSTEP = 100
X = 2.40    Y =  0.7188464484E+00  0.5663555956E+00    NSTEP = 102
X = 2.60    Y =  0.7267018158E+00  0.5585617516E+00    NSTEP = 104
X = 2.80    Y =  0.7315187328E+00  0.5537753161E+00    NSTEP = 105
X = 3.00    Y =  0.7344657681E+00  0.5508444084E+00    NSTEP = 106
X = 3.20    Y =  0.7362663926E+00  0.5490527710E+00    NSTEP = 107
X = 3.40    Y =  0.7373657096E+00  0.5479586296E+00    NSTEP = 108
X = 3.60    Y =  0.7380364367E+00  0.5472909451E+00    NSTEP = 109
X = 3.80    Y =  0.7384455256E+00  0.5468836683E+00    NSTEP = 110
X = 4.00    Y =  0.7386964050E+00  0.5466338846E+00    NSTEP = 111
X = 4.20    Y =  0.7388551462E+00  0.5464758318E+00    NSTEP = 112
X = 5.00    Y =  0.7390521643E+00  0.5462796619E+00
       rtol=0.10D-05
fcn=  712 jac= 112 step= 120 accpt= 112 rejct=  8 dec= 120 sol=  720
```

End Fortran Code

End of Appendix A