```
> restart;
```

```
> ### This worksheet was written for Maple 16.01 Standard.
  ### May need tweaking for earlier versions of Maple or for Maple
  Classic.
  ### Last Revised 2012-10-01
  ### Report problems: contact@patricktoche.com
```

```
> ### Set display option
  mydisplayprecision:=3:
  interface(displayprecision=mydisplayprecision):
```

```
> ### Procedure to export plots

  MakePlot := proc(p::evaln, {[x,ext,extension]:=ps})
      local thename, theplace, opts:
      global N;
      thename := cat(convert(p,string),"_",convert(N,string),".",
  convert(x,string)):
      theplace := cat(currentdir(),kernelopts(dirsep),convert(N,
  string),kernelopts(dirsep)):
      if x = gif then
          opts := `color,portrait,noborder,transparent,height=512,
  width=512`: #default jpeg: height=360,width=480
      else
          #default gif : height=512,width=512
          opts := `color,portrait,noborder,transparent,height=360,
  width=480`:
      end if:
      plotsetup('x', 'plotoutput'=cat(theplace,thename),
  'plotoptions'=opts):
      print( plots:-display( eval(p), 'axesfont' = [ TIMES, 10 ],
  'labelfont' = [ TIMES, ROMAN, 10] ) ):
      plotsetup(default):
  end proc:
```

```
> ### Tractable Model Parameter Definitions
  ###   rho   : coefficient of relative risk aversion, CRRA
  ###   mu    : probability of job loss
  ###   R     : interest factor on financial wealth, i.e. R = 1+r
  ###   beta  : patience factor, i.e. inverse of discount factor
  ###   G     : growth factor of labor income
  ###   Gamma : Gamma = G/(1-mu)
```

```
> ############################## Incomplete
  ############################
  ### The Selection of Parameter Values is at the experimental
  stage ###
  ### Choices subject to change
   ###
  ### Not all figures have been tweaked or optimized
   ###
  ##################################################################
  ####
```

```
> ### Parameter values for ctdiscrete, fixing Gamma=1 (Zero Growth)
  ### To use this parameter configuration set N:=1;
  parameters[1] := [ R = 103/100, beta = 100/110, Gamma = 1 ]:
      'parameters[1]' = evalf(%);
      'R*beta' = evalf(eval(R*beta,parameters[1]));
```

$$parameters_1 = [R = 1.03, \beta = 0.909, \Gamma = 1.]$$

$$R\,\beta = 0.936 \tag{1}$$

```
> ### Parameter values for ctdiscrete, fixing G=1 (Zero Growth)
  ### To use this parameter configuration set N:=2;

  parameters[2] := [ R = 103/100, beta = 100/110, Gamma = 1/(1-mu)
  ]:
      'parameters[2]' = evalf(%);
      'R*beta' = evalf(eval(R*beta,parameters[2]));
```

$$parameters_2 = \left[R = 1.03, \beta = 0.909, \Gamma = \frac{1}{1-\mu}\right]$$

$$R\,\beta = 0.936 \tag{2}$$

```
> ### Parameter values from cssUSsaving, 16 March 2012, section 5.2
  ### To use this parameter configuration set N:=3;
  ### R=1.04 and beta=0.975=10000/10256,e at annual frequency.
  ### R=1.01 and beta=1-0.0064=0.994, at quarterly frequency

  parameters[3] := [ R = 104/100, beta = 10000/10256, Gamma =
  101/100/(1-mu) ]:
      'parameters[3]' = evalf(%);
      'R*beta' = evalf(eval(R*beta,parameters[3]));
```

$$parameters_3 = \left[R = 1.04, \beta = 0.975, \Gamma = \frac{1.01}{1-\mu}\right]$$

$$R\,\beta = 1.01 \tag{3}$$

```
> ### Parameter values, fixing Gamma=101/100 (Positive Growth)
  ### To use this parameter configuration set N:=4;

  parameters[4] := [ R = 103/100, beta = 100/110, Gamma = 101/100 ]
  :
      'parameters[4]' = evalf(%);
      'R*beta' = evalf(eval(R*beta,parameters[4]));
```

$$parameters_4 = [R = 1.03, \beta = 0.909, \Gamma = 1.01]$$

$$R\,\beta = 0.936 \tag{4}$$

```
> ### Parameter values, fixing Gamma=101/100  (Positive Growth, R*
  beta=1)
  ### To use this parameter configuration set N:=5;

  parameters[5] := [ R = 103/100, beta = 100/103, Gamma = 101/100 ]
  :
      'parameters[5]' = evalf(%);
      'R*beta' = evalf(eval(R*beta,parameters[5]));
```

$$parameters_5 = [R = 1.03, \beta = 0.971, \Gamma = 1.01]$$

$$R\,\beta = 1. \tag{5}$$

```
> ### Set parameter values from the configurations above
  ### Select a value for N below, save, and  Edit -> Execute ->
  Worksheet

  N := 5:  # Parameter lists are numbered: N = 1,2,3...
      params := parameters[N]:
      'params' = evalf(params);
```

$$params = \left[R = 1.03,\ \beta = 0.971,\ \Gamma = 1.01\right] \tag{6}$$

```
> ### Store selected individual parameters for convenience

  Rf := subs(params,R):
  betaf := subs(params,beta):
  Gammaf := subs(params,Gamma):
```

```
> ### Marginal propensity to consume in unemployment

  mpcu := (R,beta,rho) -> 1-(R*beta)^(1/rho)/R:
      'mpcu' = mpcu(R,beta,rho);
```

$$mpcu = 1 - \frac{(R\,\beta)^{\frac{1}{\rho}}}{R} \tag{7}$$

```
> ### Target wealth-income ratio

  m := (R,beta,Gamma,rho,mu) -> 1 + 1 / ( Gamma/R - 1 + mpcu(R,
  beta,rho) * ( 1 + ( ((R*beta)^(1/rho)/Gamma)^(-rho)-1 ) / mu )^
  (1/rho) ):
      'm' = m(R,beta,Gamma,rho,mu);
```

$$m = 1 + \frac{1}{\frac{\Gamma}{R} - 1 + \left(1 - \frac{(R\,\beta)^{\frac{1}{\rho}}}{R}\right)\left(1 + \frac{\left(\frac{(R\,\beta)^{\frac{1}{\rho}}}{\Gamma}\right)^{-\rho} - 1}{\mu}\right)^{\frac{1}{\rho}}} \tag{8}$$

```
> ### Target saving rate
  ### from pi/(1-pi)=rhs (c.f. equation in the text), we have pi=
  rhs/(1+rhs), so we have s=1-pi=1/(1+rhs)

  s := (R,beta,Gamma,rho,mu) -> 1 / (1 + mpcu(R,beta,rho)*(R/Gamma)
  *((((R*beta)^(1/rho)/Gamma)^(-rho)-(1-mu))/mu)^(1/rho) ):
      's' = s(R,beta,Gamma,rho,mu);
```

(9)

$$s = \frac{1}{1 + \frac{\left(1 - \frac{(R\,\beta)^{\frac{1}{\rho}}}{R}\right) R \left(\frac{\left(\frac{(R\,\beta)^{\frac{1}{\rho}}}{\Gamma}\right)^{-\rho} - 1 + \mu}{\mu}\right)^{\frac{1}{\rho}}}{\Gamma}} \tag{9}$$

```
> ### Create a list of values for rho

  rholist := [ seq(k, k = 1 .. 20) ]:
      'rho' = rholist[1..10];
```

$$\rho = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] \tag{10}$$

```
> ### Create a list of values for mu

  mulist := [ 0, seq(2^k/100, k = 0 .. 20) ]:
      'mu' = evalf(%)[1..10];
```

$$\mu = [0., 0.0100, 0.0200, 0.0400, 0.0800, 0.160, 0.320, 0.640, 1.28, 2.56] \tag{11}$$

```
> ### Check RIC and GIC Conditions

  RIC := (R,beta,rho) -> (R*beta)^(1/rho)/R:
  RICf := rho -> RIC(subs(params,R),subs(params,beta),rho):
  GIC := (R,beta,rho,Gamma) -> (R*beta)^(1/rho)/Gamma:
  GICf := (rho,mu) -> GIC(subs(params,R),subs(params,beta),rho,subs
  (params,Gamma)):

  ### Check the RIC
  Matrix([seq( [seq( is(RICf(rho)<1), mu=mulist[2..8])],rho=rholist
  [1..10])]):
      LinearAlgebra:-Transpose(%);

  ### Check the GIC
  Matrix([seq( [seq( is(GICf(rho,mu)<1), mu=mulist[2..8])],rho=
  rholist[1..10])]):
      LinearAlgebra:-Transpose(%);

  ### Check the strong GIC
  Matrix([seq( [seq( is(GICf(rho,mu)<(1-mu)^(-1/rho)), mu=mulist[2.
  .8])],rho=rholist[1..10])]):
      LinearAlgebra:-Transpose(%);
```

$$\begin{bmatrix} true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \end{bmatrix}$$

$$\begin{bmatrix} true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \end{bmatrix}$$

$$\begin{bmatrix} true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \end{bmatrix} \tag{12}$$

```
> ### Target wealth-income ratio for fixed values of R,Gamma,beta

  eval(m(R,beta,Gamma,rho,mu),params):
  mf := unapply(%,(rho,mu)):
  interface(displayprecision=3):
      'm' = evalf(mf(rho,mu));
  interface(displayprecision=mydisplayprecision):
```

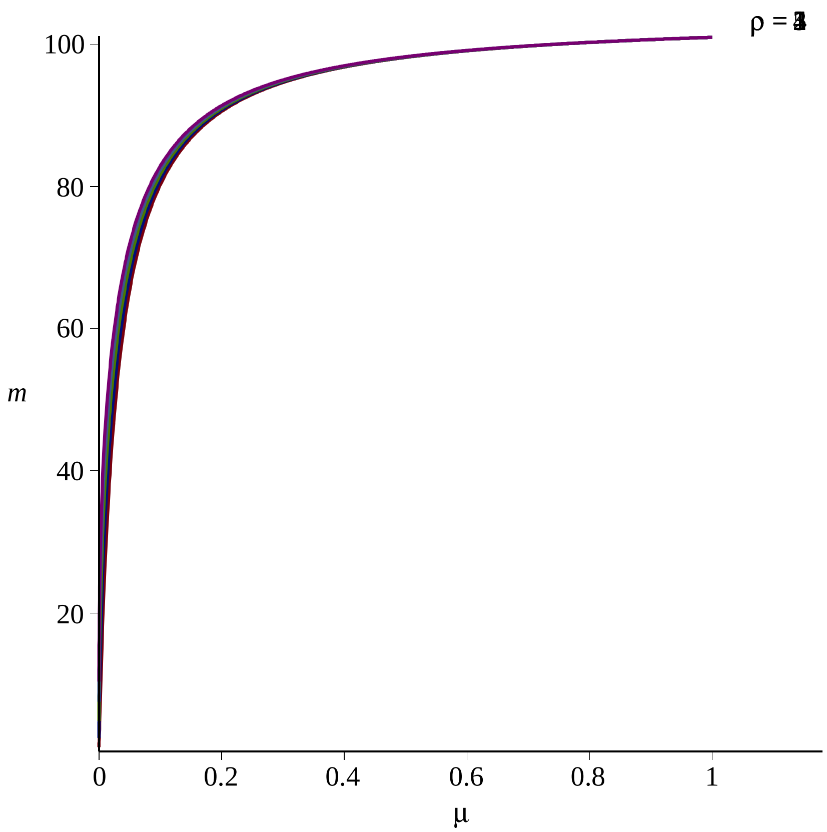
$$m = 1 + \frac{1}{-0.0194 + 0.0291\left(1 + \frac{0.990^{-\rho} - 1}{\mu}\right)^{\frac{1}{\rho}}} \tag{13}$$

```
> ### Plot of m as rho and mu vary

  mTargetUrateVariesCRRAVaries := plots:-display( plot3d(mf(rho,
  mu), rho = 1..5, mu = 0..1)
      , 'axes' = normal
      , 'style' = surfacecontour
      , 'shading' = zhue
      , 'lightmodel' = light1
      , 'tickmarks' = [ 6, 6, 4 ]
      , 'labels' = [ rho, mu, 'm' ]
      , 'view' = [ 1 .. 5, 0 .. 1, default ]
      , 'orientation' = [ -10, 50 ]
    ) :  # % ;
```

```
> ### Animated plot of m as rho and mu vary

  mTargetUrateVariesCRRAVariesAnimation :=  plots:-display(
  mTargetUrateVariesCRRAVaries
      , 'viewpoint' = ["circleright", frames=200]
    ) : # % ;
```

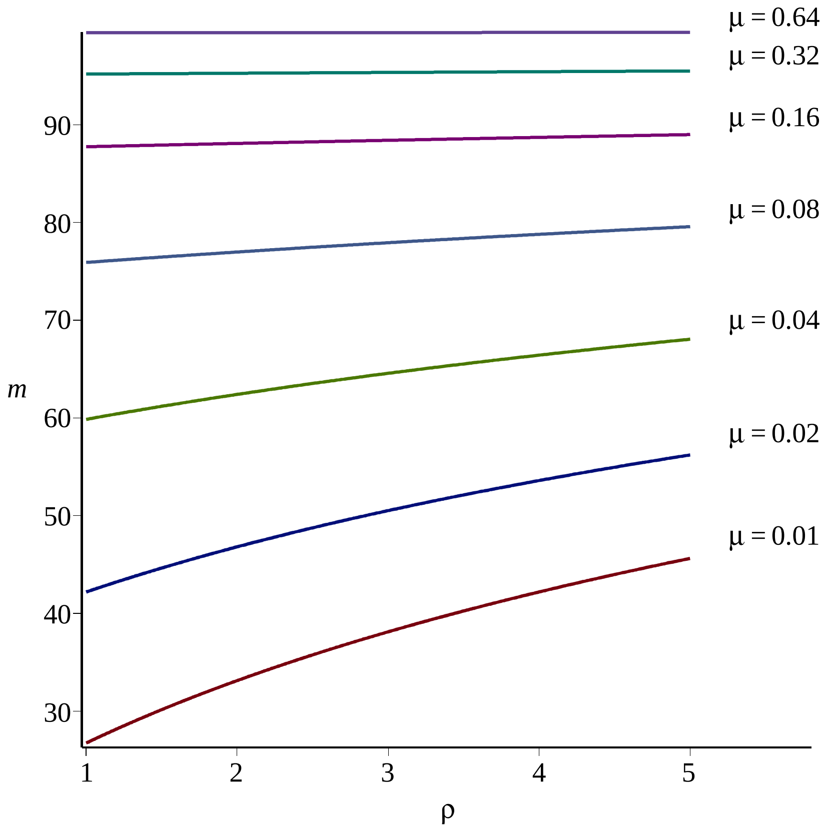
```
> ### Set position of the plot labels, tweaked for stated parameter
  values

  if N=2 then
      xmu:=rho->0.2/rho:  ymu:=rho->1.4*mf(rho,xmu(rho)): # fix x-
  value, vary y-value
      xrho:=mu->5.2:      yrho:=mu->mf(xrho(mu),mu):  # fix x-
  value, vary y-value
  else
      xmu:=rho->1.05: ymu:=rho->mf(rho,xmu(rho)): # fix x-value,
  vary y-value
      xrho:=mu->5.2:  yrho:=mu->mf(xrho(mu),mu):  # fix x-value,
  vary y-value
  end if:
```

```
> ### Plot of m as mu varies for fixed values of rho

  plot_m_mu := plot( [ seq( mf(rho,mu) , rho=rholist[1..5] ) ]
      , mu = 0 .. 1
      , 'numpoints' = 1000
      , 'tickmarks' = [ 6, 6 ]
      , 'labels' = [ mu, 'm' ]
  #    , 'legend' = [ seq( 'rho' = k, k = rholist[1..5] ) ]
  #    , 'legendstyle' = [ 'font' = [TIMES,ROMAN,8], 'location' =
  bottom ]
      , 'view' = [ 0 .. 1.18, default ]
    ) :

  #### plot labels

  ptxt := seq( plots:-textplot([xmu(rho),ymu(rho),'typeset'('rho',
  " = ", rho)], 'align'={'above','right'}), rho=rholist[1..5]):

  mTargetCRRAFixedUrateVaries := plots:-display([plot_m_mu,ptxt]):
  %;
```



```
> ### Plot of m as rho varies for fixed values of mu

  interface(displayprecision=2):

  plot_m_rho := plot( [seq(mf(rho,mu),mu=mulist[2..8])]
      , rho = 1 .. 5
      , 'numpoints' = 1000
      , 'tickmarks' = [ 6, 6 ]
      , 'labels' = [ rho, 'm' ]
  #     , 'legend' = [ seq( 'mu' = evalf(k), k = mulist[2..8] ) ]
  #     , 'legendstyle' = [ 'font' = [TIMES,ROMAN,8], 'location' =
  bottom ]
      , 'view' = [ 1 .. 5.8, default ]
    ) :

  #### plot labels

  ptxt := seq( plots:-textplot([xrho(mu),yrho(mu),'typeset'('mu', "
  = ", evalf(mu))], 'align'={'above','right'}), mu=mulist[2..8]):

  mTargetUrateFixedCRRAVaries := plots:-display([plot_m_rho,ptxt]):
  %;
```

```
interface(displayprecision=mydisplayprecision):
```



```
> ### Table of target values m as rho and mu run through lists

  interface(displayprecision=6):
  mvalues := Matrix([seq( [seq(mf(rho,mu), rho=rholist[1..8])],mu=
  mulist[2..8])]):
  mvalues := ArrayTools:-Concatenate(2,Vector[column](evalf[2]
  (mulist[2..8])),mvalues):
  mvalues := ArrayTools:-Concatenate(1,Vector[row]([0,op(rholist[1.
  .8])]),mvalues):
      'mvalues' = evalf(%);
  interface(displayprecision=mydisplayprecision):
```

mvalues **(14)**

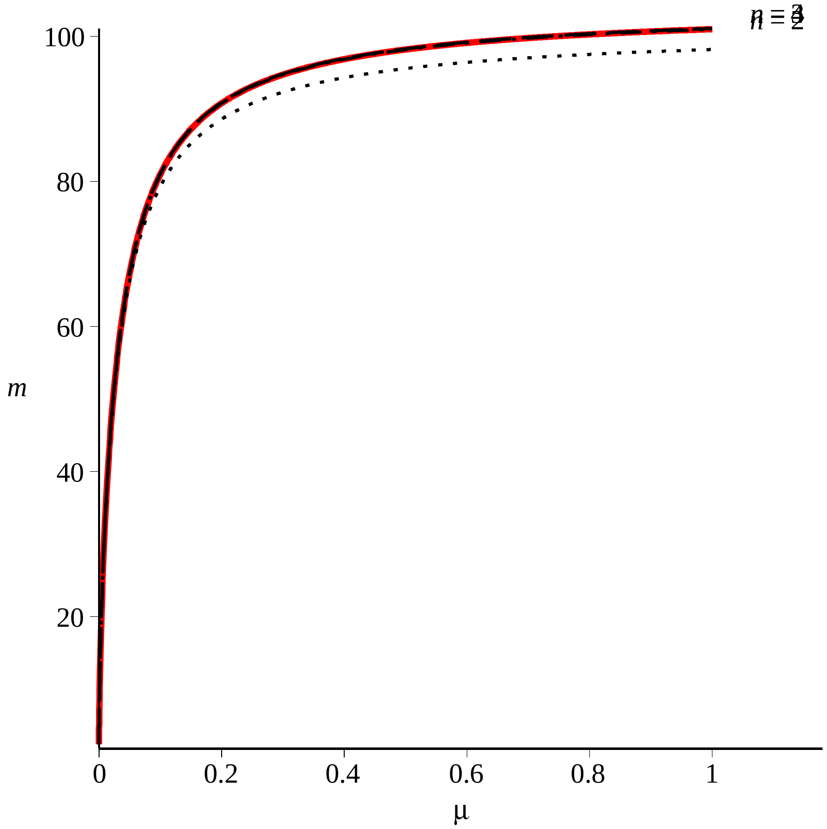
$$
= \left[\begin{array}{ccccccccc}
0. & 1. & 2. & 3. & 4. & 5. & 6. & 7. & 8. \\
0.010 & 26.7500 & 33.1392 & 38.1289 & 42.2077 & 45.6416 & 48.5938 & 51.1723 & 53.4527 \\
0.020 & 42.2000 & 46.8197 & 50.5245 & 53.6014 & 56.2201 & 58.4892 & 60.4834 & 62.2558 \\
0.040 & 59.8571 & 62.4084 & 64.5650 & 66.4244 & 68.0520 & 69.4941 & 70.7847 & 71.9491 \\
0.080 & 75.9091 & 76.9711 & 77.9225 & 78.7819 & 79.5637 & 80.2793 & 80.9376 & 81.5463 \\
0.16 & 87.7368 & 88.0768 & 88.3959 & 88.6964 & 88.9800 & 89.2483 & 89.5026 & 89.7441 \\
0.32 & 95.1714 & 95.2547 & 95.3353 & 95.4135 & 95.4893 & 95.5629 & 95.6343 & 95.7037 \\
0.64 & 99.3881 & 99.4003 & 99.4123 & 99.4242 & 99.4359 & 99.4474 & 99.4589 & 99.4701
\end{array}\right]
$$

```
> ### Check of the accuracy of various approximations
  ### The plot shows that n>3 is needed for decent approximation

  Rho := 2: # Fix a value of rho = Rho

  mfn := (rho,mu,n) -> evalf[n](mf(rho,mu)):
      'mfn' = [mfn(Rho,mu,1),mfn(Rho,mu,2),mfn(Rho,mu,3),mfn(Rho,
  mu,4),mfn(Rho,mu,5)];

  plot_mff_mu := plot( mf(Rho,mu)
      , mu = 0 .. 1
      , 'numpoints' = 1000
      , 'color' = red
      , 'thickness' = 3
      , 'linestyle' = solid
    ) :

  plot_mfn_mu := n -> plot( mfn(Rho,mu,n)
      , mu = 0 .. 1
      , 'numpoints' = 1000
      , 'color' = black
      , 'thickness' = 1
      , 'linestyle' = n
    ) :


  ### plot labels
  xmu:=n->1.05: ymu:=n->mfn(Rho,1,n): # fix x-value, vary y-value
  ptxt := seq( plots:-textplot([xmu(n),ymu(n),'typeset'('n', " = ",
  n)], 'align'={'above','right'}), n=2..4):

  mTargetCRRAFixedUrateVariesApproximations :=
      plots:-display([plot_mff_mu,plot_mfn_mu(2),plot_mfn_mu(3),
  plot_mfn_mu(4),ptxt]
          , 'tickmarks' = [ 6, 6 ]
          , 'labels' = [ mu, 'm' ]
          , 'view' = [ 0 .. 1.18, default ]
        ) : %;
```

$$
mfn = \left[1 + \frac{1}{-0.02 + 0.03\sqrt{1 + \frac{0.02}{\mu}}}, 1 + \frac{1}{-0.019 + 0.029\sqrt{1 + \frac{0.020}{\mu}}}, 1\right.
$$

$$+ \frac{1}{-0.0194 + 0.0291\,\sqrt{1 + \frac{0.0201}{\mu}}}, 1 + \frac{1}{-0.0194 + 0.0291\,\sqrt{1 + \frac{0.0201}{\mu}}}, 1$$

$$+ \frac{1}{-0.0194 + 0.0291\,\sqrt{1 + \frac{0.0201}{\mu}}}\Bigg]$$



```
> ##########################
> ### Asymptotic values of m as risk-aversion rho becomes
  arbitrarily large

  asymptotic_m_mu := [seq(limit(mf(rho,mu),rho=infinity), mu=mulist
  [2..20])];
```

asymptotic_m_mu := [101, 101, 101, 101, 101, 101, 101, 101, 101, 101, 101, 101, 101, 101, 101, 101, 101, 101, 101] **(15)**

```
> ### Derivative of m with respect to R

  dm := (R,beta,Gamma,rho,mu) -> diff(m(R,beta,Gamma,rho,mu),R):
  eval(dm(R,beta,Gamma,rho,mu),params):
```

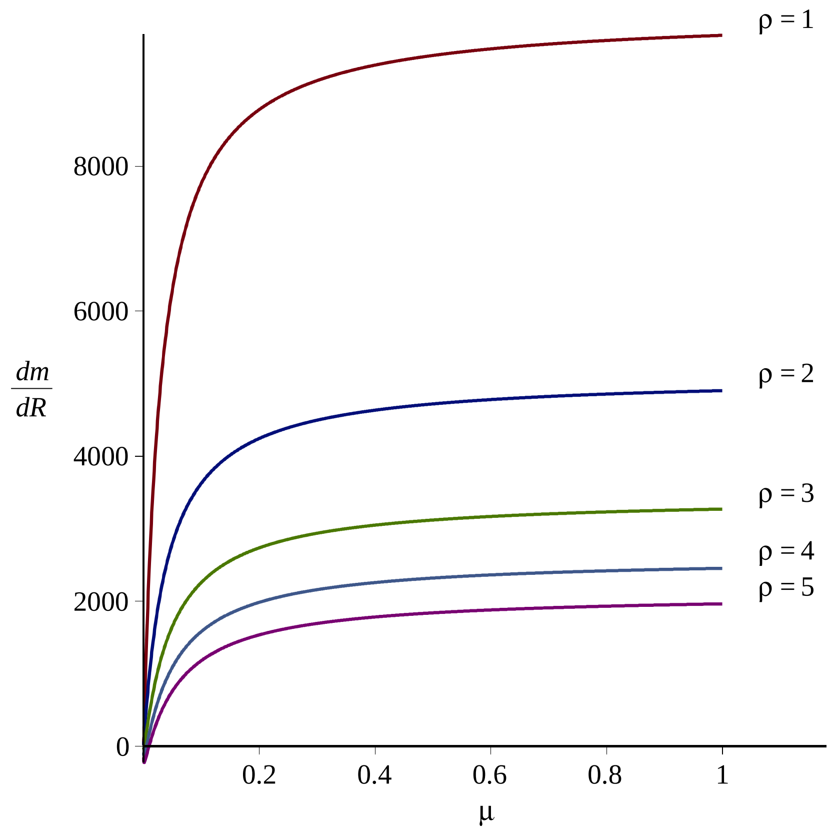
```
  dmf := unapply(%,(rho,mu)):
  interface(displayprecision=4):
      'dm' = evalf(dmf(rho,mu));
  interface(displayprecision=mydisplayprecision):
```

$$dm = -\left(-0.9520 + \left(-\frac{0.9426}{\rho} + 0.9426 \right) \left(1 + \frac{0.9901^{-\rho} - 1}{\mu} \right)^{\frac{1}{\rho}} - \frac{0.02828 \left(1 + \frac{0.9901^{-\rho} - 1}{\mu} \right)^{\frac{1}{\rho}} 0.9901^{-\rho}}{\rho\, \mu \left(1 + \frac{0.9901^{-\rho} - 1}{\mu} \right)} \right) \Bigg/ \left(-0.01942 + 0.02913 \left(1 + \frac{0.9901^{-\rho} - 1}{\mu} \right)^{\frac{1}{\rho}} \right)^{2} \qquad \textbf{(16)}$$

```
> ### Set position of the plot labels, tweaked for stated parameter
  values

  if N=2 then
      xmu:=rho->0.12:  ymu:=rho->-4+1.6*dmf(rho,xmu(rho)): # fix x-
  value, vary y-value
      xrho:=mu->5.2:      yrho:=mu->dmf(xrho(mu),mu):  # fix x-
  value, vary y-value
  else
      xmu:=rho->1.05: ymu:=rho->dmf(rho,xmu(rho)): # fix x-value,
  vary y-value
      xrho:=mu->5.2:  yrho:=mu->dmf(xrho(mu),mu)+20:  # fix x-
  value, vary y-value
  end if:
```

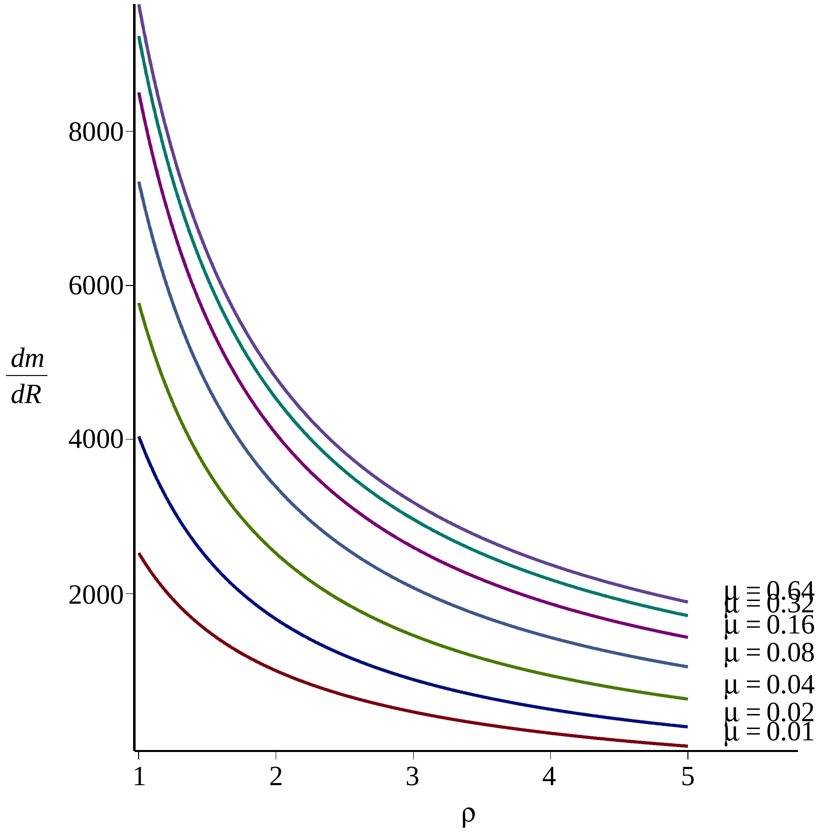
```
> ### Plot of derivative of m with respect to R, for fixed values
  of rho

  plot_dmdR_mu := plot( [ seq( dmf(rho,mu) , rho=rholist[1..5] ) ]
      , mu = 0 .. 1
      , 'numpoints' = 1000
      , 'tickmarks' = [ 6, 6 ]
      , 'labels' = [ mu, 'dm/dR' ]
      , 'view' = [ 0 .. 1.18, default ]
    ) :

  #### plot labels

  ptxt := seq( plots:-textplot([xmu(rho),ymu(rho),'typeset'('rho',
  " = ", rho)], 'align'={'above','right'}), rho=rholist[1..5]):
```

```
if N = 2 then
    theview := [ 0 .. 1, -10 .. 28 ] :
else
    theview := default :
end if:

mSlopeCRRAFixedUrateVaries := plots:-display( [plot_dmdR_mu,
ptxt], 'view' = theview ): %;
```



```
> ### Plot of derivative of m with respect to R, for fixed values
  of mu

  interface(displayprecision=2):

  plot_dmdR_rho := plot( [ seq( dmf(rho,mu) , mu=mulist[2..8] ) ]
      , rho = 1 .. 5
      , 'numpoints' = 1000
      , 'tickmarks' = [ 6, 6 ]
      , 'labels' = [ rho, 'dm/dR' ]
      , 'view' = [ 1 .. 5.8, default ]
    ) :

  #### plot labels
```

```
ptxt := seq( plots:-textplot([xrho(mu),yrho(mu),'typeset'('mu', "
= ", evalf(mu))], 'align'={'above','right'}), mu=mulist[2..8]):

mSlopeUrateFixedCRRAVaries := plots:-display([plot_dmdR_rho,ptxt]
): %;

interface(displayprecision=mydisplayprecision):
```



```
> ### Table of percentage change in target values m after 1% Change
  in After-Tax Interest Rate
  ### Mid-Point Formula

  interface(displayprecision=6):
  mchanges := Matrix([seq( [seq( 100*(m(Rf,betaf,Gammaf,rho,mu)-m
  (Rf-1/100,betaf,Gammaf,rho,mu))/((m(Rf,betaf,Gammaf,rho,mu)+m
  (Rf-1/100,betaf,Gammaf,rho,mu))/2) ,rho=rholist[1..8] )],mu=
  mulist[2..8])]):
  mchanges := ArrayTools:-Concatenate(2,Vector[column](evalf[2]
  (mulist[2..8])),mchanges):
  mchanges := ArrayTools:-Concatenate(1,Vector[row]([0,op(rholist
  [1..8])]),mchanges):
      'mchanges' = evalf(%);
  interface(displayprecision=mydisplayprecision):
```

```
mchanges                                                                    (17)
    = [[0., 1., 2., 3., 4., 5., 6., 7., 8.],
    [0.010, 62.9741, 21.7765, 8.09606, 1.87087, −1.44786, −3.38485, −4.57881, −5.33844
    ],
    [0.020, 64.1566, 26.8263, 13.3173, 6.76090, 3.06288, 0.780102, −0.714998, −1.73496],
    [0.040, 64.7648, 31.3271, 18.2897, 11.5799, 7.59753, 5.01896, 3.24838, 1.98030],
    [0.080, 65.0731, 34.7131, 22.2890, 15.6308, 11.5319, 8.78470, 6.83425, 5.39056],
    [0.16, 65.2284, 36.9086, 25.0243, 18.5187, 14.4327, 11.6397, 9.61784, 8.09194],
    [0.32, 65.3064, 38.1858, 26.6730, 20.3135, 16.2850, 13.5076, 11.4793, 9.93460],
    [0.64, 65.3454, 38.8794, 27.5873, 21.3281, 17.3511, 14.6010, 12.5863, 11.0472]]
```

```
> #########################
```

```
> ### Target saving rate for fixed values of R,Gamma,beta

  eval(s(R,beta,Gamma,rho,mu),params):
  sf := unapply(%,(rho,mu)):
  interface(displayprecision=4):
      's' = evalf(sf(rho,mu));
  interface(displayprecision=mydisplayprecision):
```

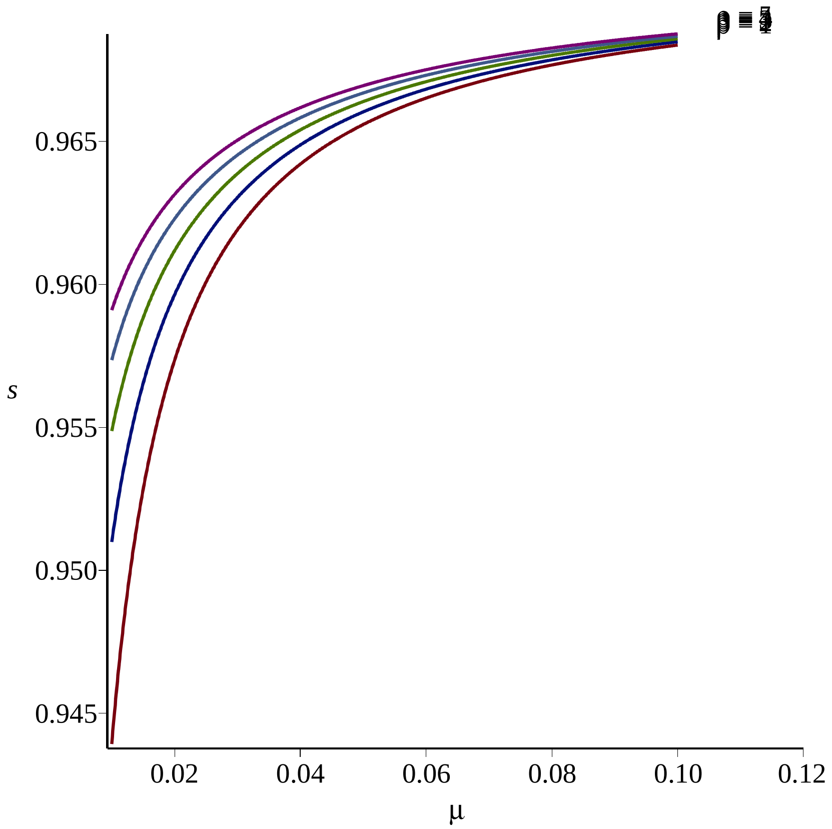
$$s = \frac{1}{1 + 0.02970 \left(\frac{0.9901^{-\rho} - 1 + \mu}{\mu} \right)^{\frac{1}{\rho}}} \tag{18}$$

```
> ### Plot of s as rho and mu vary

  sTargetUrateVariesCRRAVaries := plots:-display( plot3d(sf(rho,
  mu), rho = 1..5, mu = 0..1)
      , 'axes' = normal
      , 'style' = surfacecontour
      , 'shading' = zhue
      , 'lightmodel' = light1
      , 'tickmarks' = [ 6, 6, 4 ]
      , 'labels' = [ rho, mu, 's' ]
      , 'view' = [ 1 .. 5, 0 .. 1, 0.5 .. 1 ]
      , 'orientation' = [ -10, 50 ]
    ) :

  plot_s_rho_mu;
```

$$plot\_s\_rho\_mu \tag{19}$$

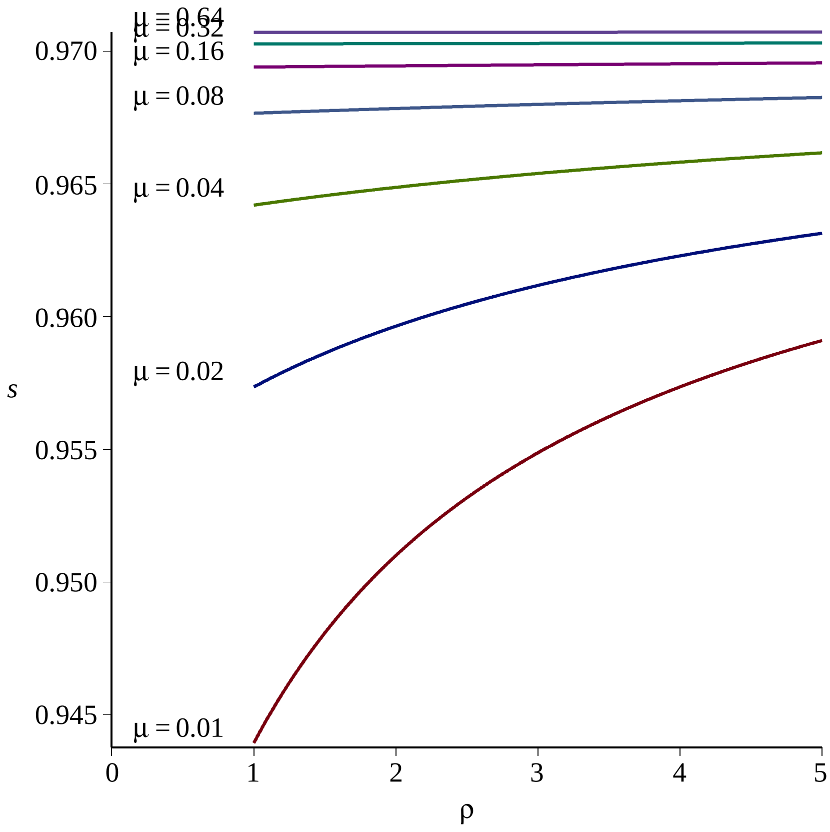
```
> ### Animated plot of m as rho and mu vary

  sTargetUrateVariesCRRAVariesAnimation :=  plots:-display(
  sTargetUrateVariesCRRAVaries
      , 'viewpoint' = ["circleright", frames=200]
    ) : # % ;
```

```
> ### Set position of the plot labels, tweaked for stated parameter
  values
  mumin := 0.01:
  mumax := 0.1:
  rhomin := 1:
  rhomax := 5:

  if N=2 then
      xmu:=rho->0.2/rho:      ymu:=rho->1.4*sf(rho,xmu(rho)): # fix
  x-value, vary y-value
      xrho:=mu->1.05*rhomax:  yrho:=mu->sf(xrho(mu),mu):  # fix x-
  value, vary y-value
  elif N=4 or N=5 then
      xmu:=rho->1.05*mumax:   ymu:=rho->sf(rho,xmu(rho)): # fix x-
  value, vary y-value
      xrho:=mu->1:            yrho:=mu->sf(xrho(mu),mu):  # fix x-
  value, vary y-value
  else
      xmu:=rho->1.05*mumax:   ymu:=rho->sf(rho,xmu(rho)): # fix x-
  value, vary y-value
      xrho:=mu->1.05*rhomax:  yrho:=mu->sf(xrho(mu),mu):  # fix x-
  value, vary y-value
  end if:

> ### Plot of s as mu varies for fixed values of rho

  plot_s_mu := plot( [ seq( sf(rho,mu) , rho=rholist[1..rhomax] ) ]
      , mu = mumin .. mumax
      , 'numpoints' = 1000
      , 'tickmarks' = [ 6, 6 ]
      , 'labels' = [ mu, 's' ]
  #    , 'legend' = [ seq( 'rho' = k, k = rholist[rhomin..rhomax] )
  ]
  #    , 'legendstyle' = [ 'font' = [TIMES,ROMAN,8], 'location' =
  bottom ]
  #    , 'view' = [ mumin .. 1.2*mumax, 0.85 .. max([seq(evalf(sf
  (rho,mumax)),rho=rholist[rhomin..rhomax])]) ]
      , 'view' = [ mumin .. 1.2*mumax
          , min([seq(evalf(sf(rho,mumin)),rho=rholist[rhomin..
  rhomax])]) .. max([seq(evalf(sf(rho,mumax)),rho=rholist[rhomin..
  rhomax])]) ]
    ) :

  #### plot labels

  ptxt := seq( plots:-textplot([xmu(rho),ymu(rho),'typeset'('rho',
  " = ", rho)], 'align'={'above','right'}), rho=rholist[rhomin..
  rhomax]):

  sTargetCRRAFixedUrateVaries := plots:-display([plot_s_mu,ptxt]):
  %;
```



```
> ### Plot of s as rho varies for fixed values of mu

  interface(displayprecision=2):

  plot_s_rho := plot( [seq(sf(rho,mu),mu=mulist[2..8])]
      , rho = 1 .. 5
      , 'numpoints' = 1000
      , 'tickmarks' = [ 6, 6 ]
      , 'labels' = [ rho, 's' ]
  #     , 'legend' = [ seq( 'mu' = evalf(k), k = mulist[2..8] ) ]
  #     , 'legendstyle' = [ 'font' = [TIMES,ROMAN,8], 'location' =
  bottom ]
      , 'view' = [ 0 .. 5, default ]
    ) :

  #### plot labels

  if N=4 or N=5 then  # specifically tweaked for parameter values
  N=4
      ptxt := seq( plots:-textplot([xrho(mu)-0.9,yrho(mu),'typeset'
  ('mu', " = ", evalf(mu))], 'align'={'above','right'}), mu=mulist
  [2..8]):
  else
```

```
    ptxt := seq( plots:-textplot([xrho(mu),yrho(mu),'typeset'
('mu', " = ", evalf(mu))], 'align'={'above','right'}), mu=mulist
[2..8]):
end if:

sTargetUrateFixedCRRAVaries := plots:-display([plot_s_rho,ptxt]):
%;

interface(displayprecision=mydisplayprecision):
```



```
> ### Table of target values s as rho and mu run through lists

  interface(displayprecision=6):
  svalues := Matrix([seq( [seq(sf(rho,mu), rho=rholist[1..8])],mu=
  mulist[2..8])]):
  svalues := ArrayTools:-Concatenate(2,Vector[column](evalf[2]
  (mulist[2..8])),svalues):
  svalues := ArrayTools:-Concatenate(1,Vector[row]([0,op(rholist[1.
  .8])]),svalues):
      'svalues' = evalf(%);
  interface(displayprecision=mydisplayprecision):
```

svalues **(20)**

```
= [[0., 1., 2., 3., 4., 5., 6., 7., 8.],
[0.010, 0.943925, 0.950993, 0.954865, 0.957350, 0.959098, 0.960403, 0.961420, 0.962238
],
[0.020, 0.957346, 0.959639, 0.961174, 0.962289, 0.963141, 0.963817, 0.964370, 0.964832
],
[0.040, 0.964200, 0.964870, 0.965395, 0.965820, 0.966173, 0.966472, 0.966729, 0.966954
],
[0.080, 0.967665, 0.967843, 0.967998, 0.968136, 0.968258, 0.968368, 0.968467, 0.968558
],
[0.16, 0.969406, 0.969449, 0.969489, 0.969527, 0.969562, 0.969595, 0.969627, 0.969656],
[0.32, 0.970279, 0.970288, 0.970297, 0.970305, 0.970313, 0.970321, 0.970329, 0.970337],
[0.64, 0.970716, 0.970718, 0.970719, 0.970720, 0.970721, 0.970722, 0.970723, 0.970724]]
```

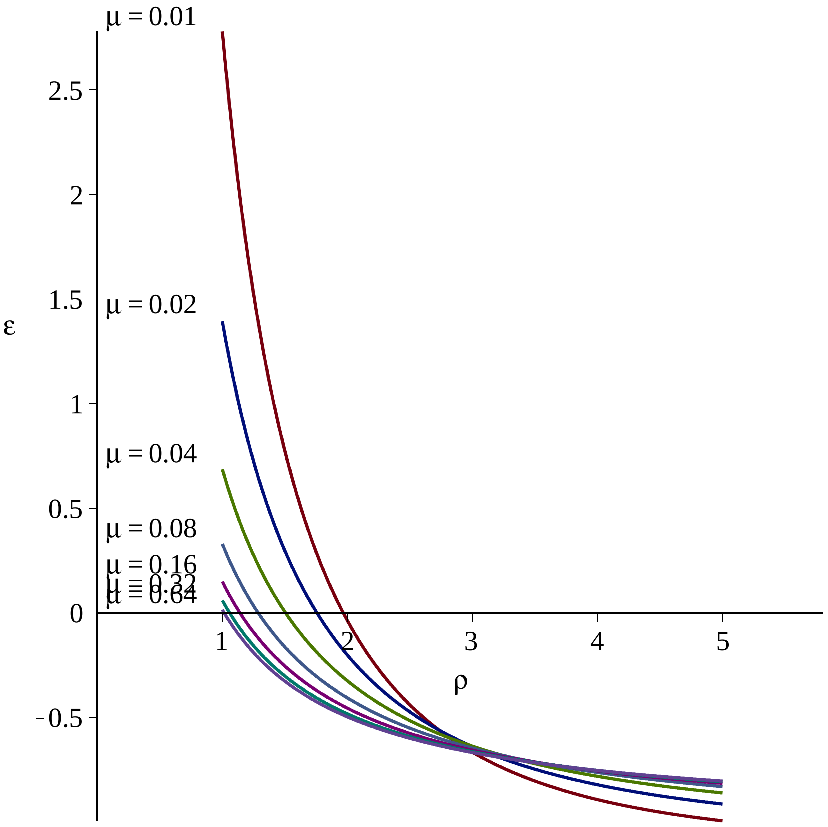
```
> ### Elasticity of s with respect to R

  ds := (R,beta,Gamma,rho,mu) -> diff(s(R,beta,Gamma,rho,mu),R):
  es :=  (R,beta,Gamma,rho,mu) -> R*ds(R,beta,Gamma,rho,mu)/s(R,
  beta,Gamma,rho,mu):
  eval(es(R,beta,Gamma,rho,mu),params):
  esf := unapply(%,(rho,mu)):
  interface(displayprecision=4):
      'es' = evalf(esf(rho,mu));
  interface(displayprecision=mydisplayprecision):
```

$$es = -\frac{1}{1+0.02970\left(\frac{0.9901^{-\rho}-1+\mu}{\mu}\right)^{\frac{1}{\rho}}}\left(1.030\left(1.020\left(-\frac{0.9426}{\rho}+0.9426\right)\left(\frac{0.9901^{-\rho}-1+\mu}{\mu}\right)^{\frac{1}{\rho}}+0.02884\left(\frac{0.9901^{-\rho}-1+\mu}{\mu}\right)^{\frac{1}{\rho}}-\frac{0.02884\left(\frac{0.9901^{-\rho}-1+\mu}{\mu}\right)^{\frac{1}{\rho}}0.9901^{-\rho}}{\rho\left(0.9901^{-\rho}-1+\mu\right)}\right)\right) \qquad \textbf{(21)}$$

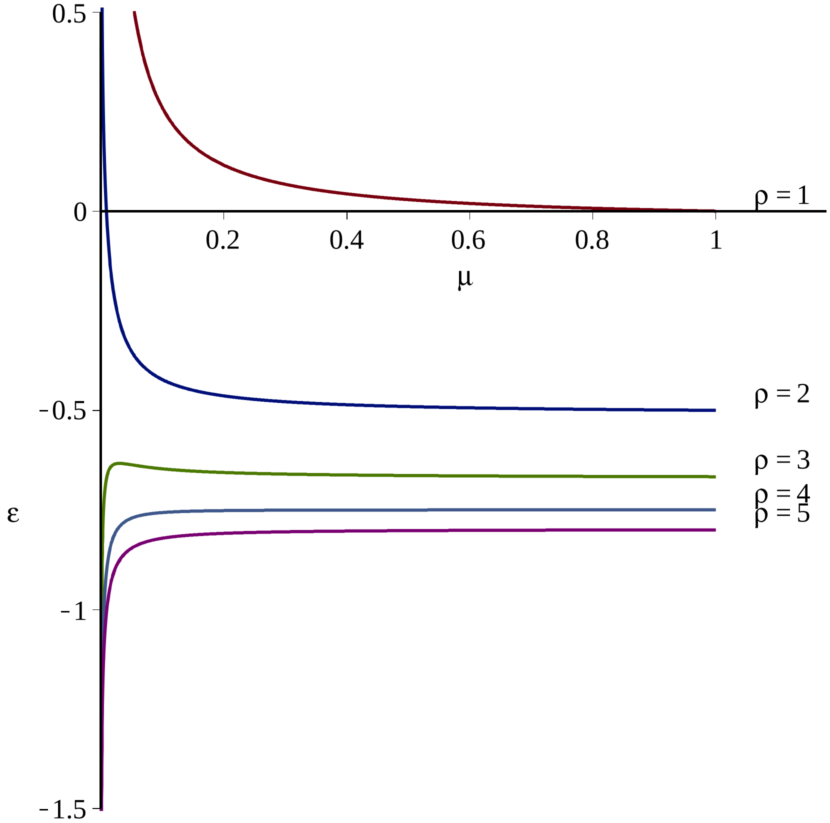
```
> ### Set position of the plot labels, tweaked for stated parameter
  values

  mumin := 1.0:
  mumax := 1.0:
  rhomin := 1:
  rhomax := 5:
```

```
  xmu:=rho->1.05*mumax:   ymu:=rho->esf(rho,xmu(rho)): # fix x-
  value, vary y-value
  xrho:=mu->mumin:        yrho:=mu->esf(xrho(mu),mu):  # fix x-
  value, vary y-value
```

```
> ### Plot of the elasticity of s with respect to R, for fixed
  values of mu

  interface(displayprecision=2):

  plot_es_rho := plot( [ seq( esf(rho,mu) , mu=mulist[2..8] ) ]
      , rho = 1 .. 5
      , 'numpoints' = 1000
      , 'tickmarks' = [ 6, 6 ]
      , 'labels' = [ rho, epsilon ]
      , 'view' = [ 0 .. 5.8, default ]
    ) :

  #### plot labels

  ptxt := seq( plots:-textplot([xrho(mu)-1,yrho(mu),'typeset'('mu',
  " = ", evalf(mu))], 'align'={'above','right'}), mu=mulist[2..8]):

  sElasticityUrateFixedCRRAVaries := plots:-display([plot_es_rho,
  ptxt]): %;

  interface(displayprecision=mydisplayprecision):
```



```
> ### Plot of the elasticity of s with respect to R, for fixed
  values of rho

  plot_es_mu := plot( [ seq( esf(rho,mu) , rho=rholist[1..5] ) ]
      , mu = 0 .. 1
      , 'numpoints' = 1000
      , 'tickmarks' = [ 6, 6 ]
      , 'labels' = [ mu, epsilon ]
      , 'view' = [ 0 .. 1.18, default ]
    ) :

  #### plot labels

  ptxt := seq( plots:-textplot([xmu(rho),ymu(rho),'typeset'('rho',
  " = ", rho)], 'align'={'above','right'}), rho=rholist[1..5]):

  sElasticityCRRAFixedUrateVaries := plots:-display([plot_es_mu,
  ptxt], 'view' = [ default, -3/2 .. 1/2 ]): %;
```



```
> ### Table of elasticity of target saving rate s after 1% Change
  in After-Tax Interest Rate
  ### Mid-Point Formula

  interface(displayprecision=6):
  schanges := Matrix([seq( [seq( 100*(s(Rf,betaf,Gammaf,rho,mu)-s
  (Rf-1/100,betaf,Gammaf,rho,mu))/((s(Rf,betaf,Gammaf,rho,mu)+s
  (Rf-1/100,betaf,Gammaf,rho,mu))/2) ,rho=rholist[1..8] )],mu=
  mulist[2..8])]):
  schanges := ArrayTools:-Concatenate(2,Vector[column](evalf[2]
  (mulist[2..8])),schanges):
  schanges := ArrayTools:-Concatenate(1,Vector[row]([0,op(rholist
  [1..8])]),schanges):
      'schanges' = evalf(%);
  interface(displayprecision=mydisplayprecision):
```

```
schanges = [[0., 1., 2., 3., 4., 5., 6., 7., 8.],                                  (22)
    [0.010, 2.65903, −0.212353, −0.748104, −0.929771, −1.00780, −1.04552, −1.06479,
    −1.07469],
    [0.020, 1.34369, −0.297149, −0.687112, −0.842496, −0.919266, −0.961986,
    −0.987586, −1.00368],
[0.040, 0.665112, −0.368890, −0.658419, −0.787563, −0.857761, −0.900421,
−0.928282, −0.947416],
[0.080, 0.320397, −0.421197, −0.648981, −0.757383, −0.819726, −0.859647,
−0.887053, −0.906809],
[0.16, 0.146659, −0.454475, −0.647795, −0.742627, −0.798653, −0.835466,
−0.861384, −0.880539],
[0.32, 0.0594414, −0.473619, −0.648823, −0.735849, −0.787812, −0.822306,
−0.846842, −0.865166],
[0.64, 0.0157450, −0.483951, −0.649908, −0.732756, −0.782414, −0.815492,
−0.839101, −0.856794]]
```

```
> #########################
> ### Export Plots
  ### The best quality 2d plots are postscript, the best 3d plots
  are png
  ### figures are converted to pdf or png with epstopdf and
  imagemagick with batch file
> interface(displayprecision=2): # necessary to strip some trailing
  zeros
> MakePlot(mTargetUrateVariesCRRAVaries,'extension'=png); # 3d
  postscript plots buggy in Maple 16 and ugly in earlier versions
> MakePlot(mTargetUrateVariesCRRAVariesAnimation,'extension'=gif);
> MakePlot(mTargetCRRAFixedUrateVaries,'extension'=ps);
> MakePlot(mTargetUrateFixedCRRAVaries,'extension'=ps);
> MakePlot(mTargetCRRAFixedUrateVariesApproximations,'extension'=
  ps);
> MakePlot(mSlopeCRRAFixedUrateVaries,'extension'=ps);
> MakePlot(mSlopeUrateFixedCRRAVaries,'extension'=ps);
> MakePlot(sTargetUrateVariesCRRAVaries,'extension'=png); # 3d
  postscript plots buggy in Maple 16 and ugly in earlier versions
> MakePlot(sTargetUrateVariesCRRAVariesAnimation,'extension'=gif);
> MakePlot(sTargetCRRAFixedUrateVaries,'extension'=ps);
> MakePlot(sTargetUrateFixedCRRAVaries,'extension'=ps);
> MakePlot(sElasticityCRRAFixedUrateVaries,'extension'=ps);
> MakePlot(sElasticityUrateFixedCRRAVaries,'extension'=ps);
> #########################
> ### Export Data to File
  theplace := cat(currentdir(),kernelopts(dirsep),convert(N,
  string),kernelopts(dirsep)):
  thedata := [ 'm'=m(R,beta,Gamma,rho,mu), 's'=s(R,beta,Gamma,rho,
  mu), 'parameters'=params ]:
> fd := fopen(cat(theplace,"ParametersAndFormulas_",convert(N,
  string),".txt"), WRITE):
  fprintf(fd, "%{c\n}a\n", <thedata>): fclose(fd):
> ExportMatrix(cat(theplace,"mvalues_mu_rho_",convert(N,string),".
  m")
      , evalf(mvalues), delimiter="&", format=rectangular, mode=
  ascii):
> ExportMatrix(cat(theplace,"mchanges_mu_rho_",convert(N,string),".
  m")
```

```
          , evalf(mchanges), delimiter="&", format=rectangular, mode=
  ascii):
> ExportMatrix(cat(theplace,"svalues_mu_rho_",convert(N,string),".
  m")
          , evalf(svalues), delimiter="&", format=rectangular, mode=
  ascii):
> ExportMatrix(cat(theplace,"schanges_mu_rho_",convert(N,string),".
  m")
          , evalf(schanges), delimiter="&", format=rectangular, mode=
  ascii):
> interface(displayprecision=mydisplayprecision): # restore
  preferences
```