```
> 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]));
```

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

$$R\,\beta = 0.936 \qquad (1)$$

```
> ### 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]));
```

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

$$R\,\beta = 0.936 \qquad (2)$$

```
> ### 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]));
```

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

$$R\,\beta = 1.01 \qquad (3)$$

```
> ### 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]));
```

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

$$R\,\beta = 0.936 \qquad (4)$$

```
> ### 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 := 4:  # Parameter lists are numbered: N = 1,2,3...
      params := parameters[N]:
      'params' = evalf(params);
```

$$params = [R = 1.03, \beta = 0.909, \Gamma = 1.01] \tag{6}$$

```
> ### 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)

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

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

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

$$\rho = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] \quad (10)$$

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

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

$$\mu = [0., 0.0100, 0.0200, 0.0400, 0.0800, 0.160, 0.320, 0.640, 1.28, 2.56] \quad (11)$$

```
> ### 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(%);
```

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

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

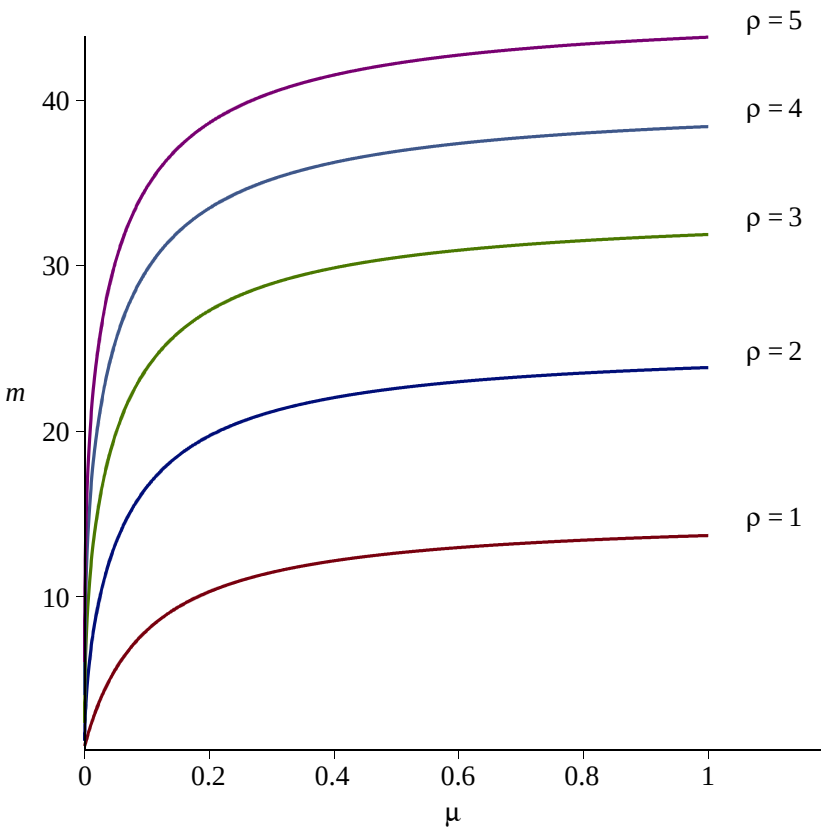
$$\begin{bmatrix} true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \\ true & true & true & true & true & true & true & true & true & true \end{bmatrix} \quad \textbf{(12)}$$

```
> ### 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):
```

$$m = 1 + \frac{1}{-0.0194 + \left(1 - 0.971\ 0.936^{\frac{1}{\rho}}\right)\left(1 + \frac{\left(0.990\ 0.936^{\frac{1}{\rho}}\right)^{-\rho} - 1}{\mu}\right)^{\frac{1}{\rho}}} \quad \textbf{(13)}$$

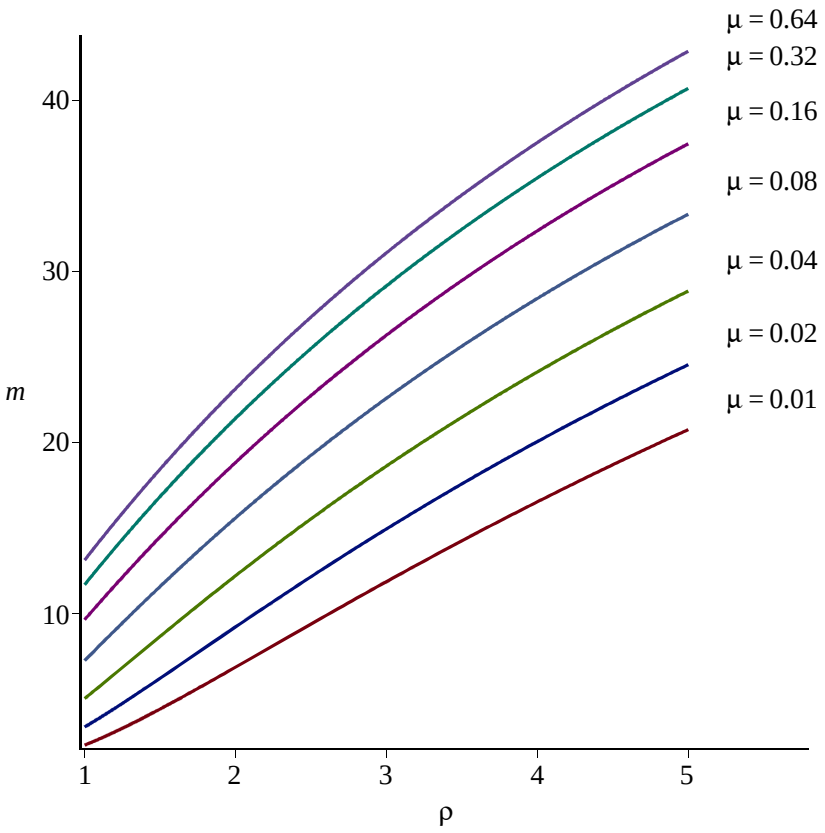
```
> ### Plot of m as rho and mu vary

  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):
```

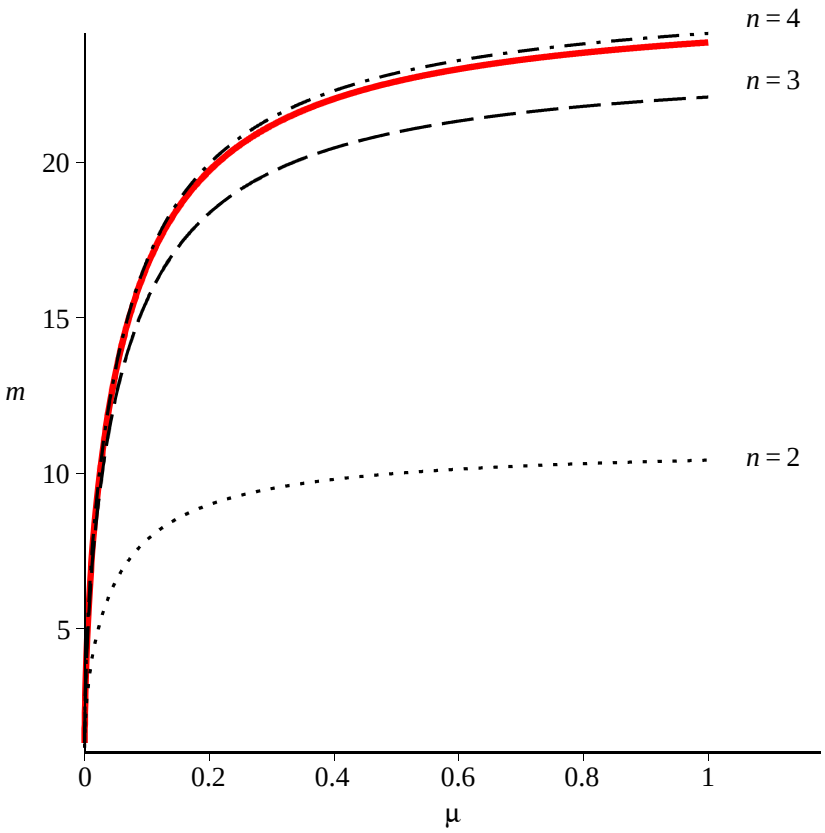
mvalues **(14)**

$$= \begin{bmatrix} 0. & 1. & 2. & 3. & 4. & 5. & 6. & 7. & 8. \\ 0.010 & 2.27160 & 6.83323 & 11.8371 & 16.5080 & 20.7330 & 24.5334 & 27.9578 & 31.0559 \\ 0.020 & 3.33128 & 9.18618 & 14.9208 & 20.0313 & 24.5306 & 28.5024 & 32.0305 & 35.1857 \\ 0.040 & 4.99647 & 12.1787 & 18.5998 & 24.1062 & 28.8411 & 32.9502 & 36.5512 & 39.7361 \\ 0.080 & 7.21674 & 15.5637 & 22.5668 & 28.4070 & 33.3381 & 37.5580 & 41.2137 & 44.4147 \\ 0.16 & 9.60779 & 18.8086 & 26.2563 & 32.3650 & 37.4631 & 41.7843 & 45.4964 & 48.7222 \\ 0.32 & 11.6573 & 21.3895 & 29.1449 & 35.4579 & 40.6964 & 45.1143 & 48.8916 & 52.1593 \\ 0.64 & 13.0974 & 23.1247 & 31.0739 & 37.5279 & 42.8723 & 47.3708 & 51.2098 & 54.5246 \end{bmatrix}$$

```
> ### 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 ]
        ) : %;
```

$$mfn = \left[1 + \frac{1}{-0.02 + 0.1\sqrt{1 + \frac{0.09}{\mu}}}, 1 + \frac{1}{-0.019 + 0.12\sqrt{1 + \frac{0.089}{\mu}}}, 1\right.$$

$$+\frac{1}{-0.0194+0.064\sqrt{1+\frac{0.0894}{\mu}}},1+\frac{1}{-0.0194+0.0600\sqrt{1+\frac{0.0894}{\mu}}},1$$
$$+\frac{1}{-0.0194+0.0606\sqrt{1+\frac{0.0894}{\mu}}}\Bigg]$$




```
> ##########################
> ### 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)**

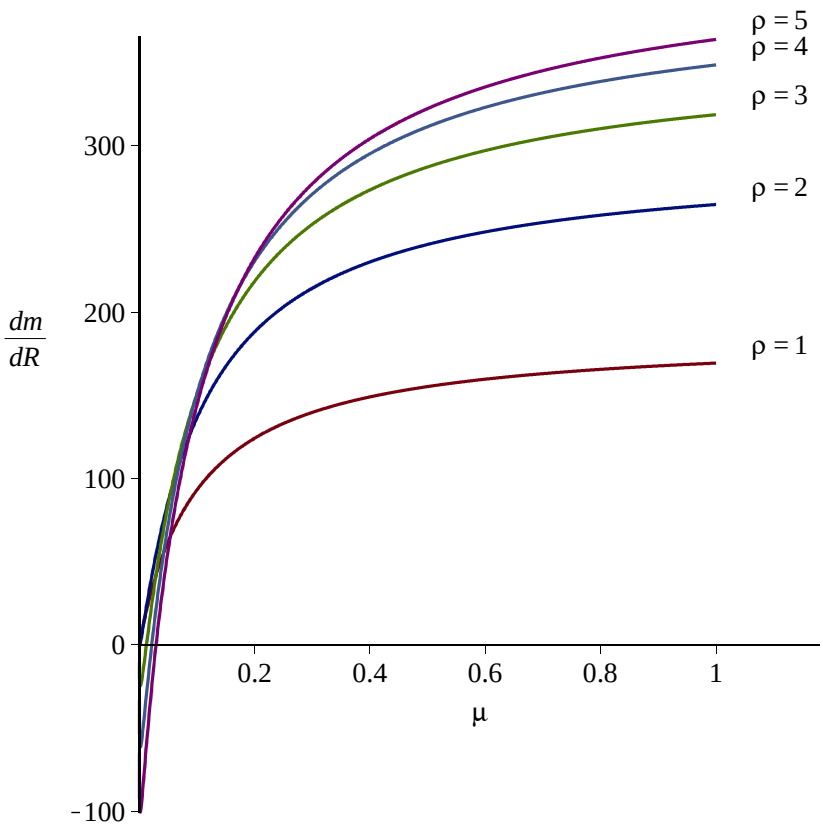
```
> ### Derivative of m with respect to R

  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):
```

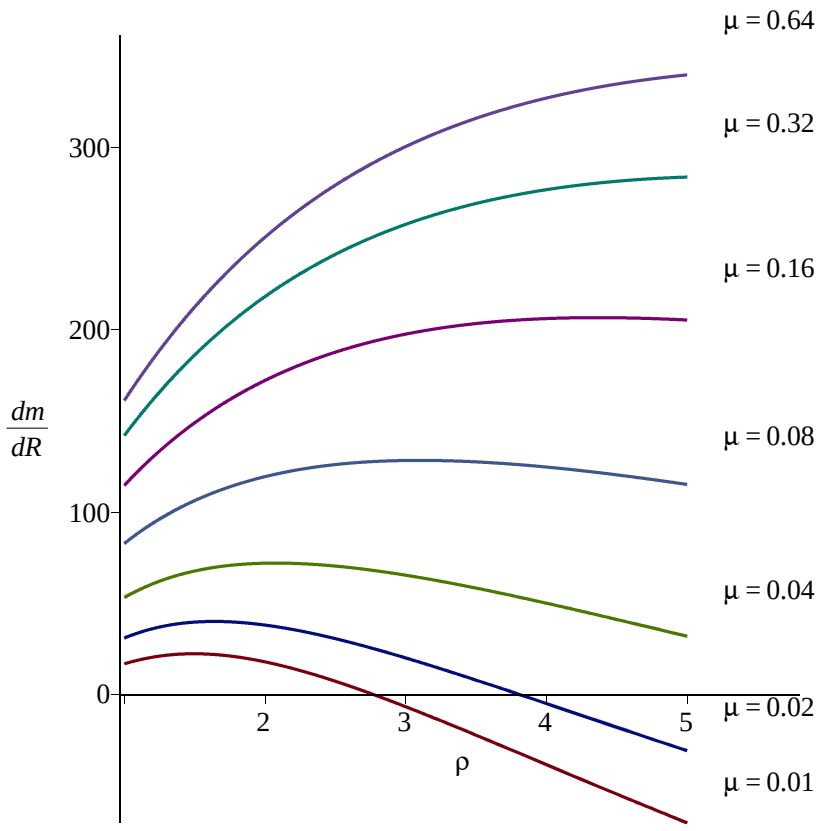
$$dm = -\left(-0.9520 + \left(-\frac{0.9426\ 0.9364^{\frac{1}{\rho}}}{\rho} + 0.9426\ 0.9364^{\frac{1}{\rho}} \right) \left(1 + \frac{\left(0.9901\ 0.9364^{\frac{1}{\rho}} \right)^{-\rho} - 1}{\mu} \right)^{\frac{1}{\rho}} - \frac{1}{\rho\,\mu \left(1 + \frac{\left(0.9901\ 0.9364^{\frac{1}{\rho}} \right)^{-\rho} - 1}{\mu} \right)} \left(0.9709 \left(1 - 0.9709\ 0.9364^{\frac{1}{\rho}} \right) \left(1 + \frac{\left(0.9901\ 0.9364^{\frac{1}{\rho}} \right)^{-\rho} - 1}{\mu} \right)^{\frac{1}{\rho}} \left(0.9901\ 0.9364^{\frac{1}{\rho}} \right)^{-\rho} \right) \Bigg/ \left(-0.01942 + \left(1 - 0.9709\ 0.9364^{\frac{1}{\rho}} \right) \left(1 + \frac{\left(0.9901\ 0.9364^{\frac{1}{\rho}} \right)^{-\rho} - 1}{\mu} \right)^{\frac{1}{\rho}} \right)^2 \right) \tag{16}$$

```
> ### 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):
```

$$mchanges = \begin{bmatrix} [0., 1., 2., 3., 4., 5., 6., 7., 8.], \\ [0.010, 6.86290, 2.13925, -1.09020, -2.86861, -3.92958, -4.59963, -5.03678, \\ -5.32577], \\ [0.020, 8.65397, 3.63897, 0.857244, -0.710922, -1.69244, -2.34905, -2.80700, \\ -3.13488], \end{bmatrix} \quad \textbf{(17)}$$

```
[0.040, 9.95268, 5.38369, 3.05514, 1.66701, 0.738645, 0.0745576, −0.421061, −0.801690
],
[0.080, 10.7601, 7.13167, 5.23513, 4.00837, 3.12590, 2.45335, 1.92211, 1.49210],
[0.16, 11.2150, 8.60094, 7.07694, 5.99708, 5.16760, 4.50280, 3.95562, 3.49672],
[0.32, 11.4572, 9.64001, 8.39554, 7.43732, 6.66367, 6.02199, 5.47983, 5.01527],
[0.64, 11.5822, 10.2798, 9.21735, 8.34526, 7.61763, 7.00158, 6.47342, 6.01567]]
> ##########################
> ### 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):
```

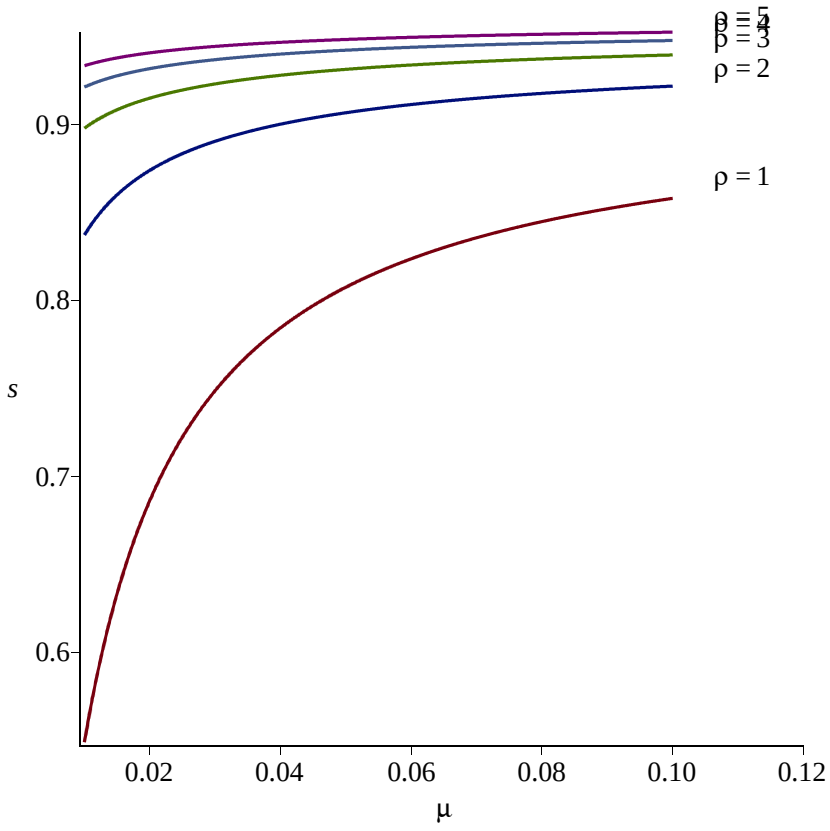
$$s = \frac{1}{1 + 1.020\left(1 - 0.9709\,0.9364^{\frac{1}{\rho}}\right)\left(\frac{\left(0.9901\,0.9364^{\frac{1}{\rho}}\right)^{-\rho} - 1 + \mu}{\mu}\right)^{\frac{1}{\rho}}} \qquad \textbf{(18)}$$

```
> ### 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 **(19)**

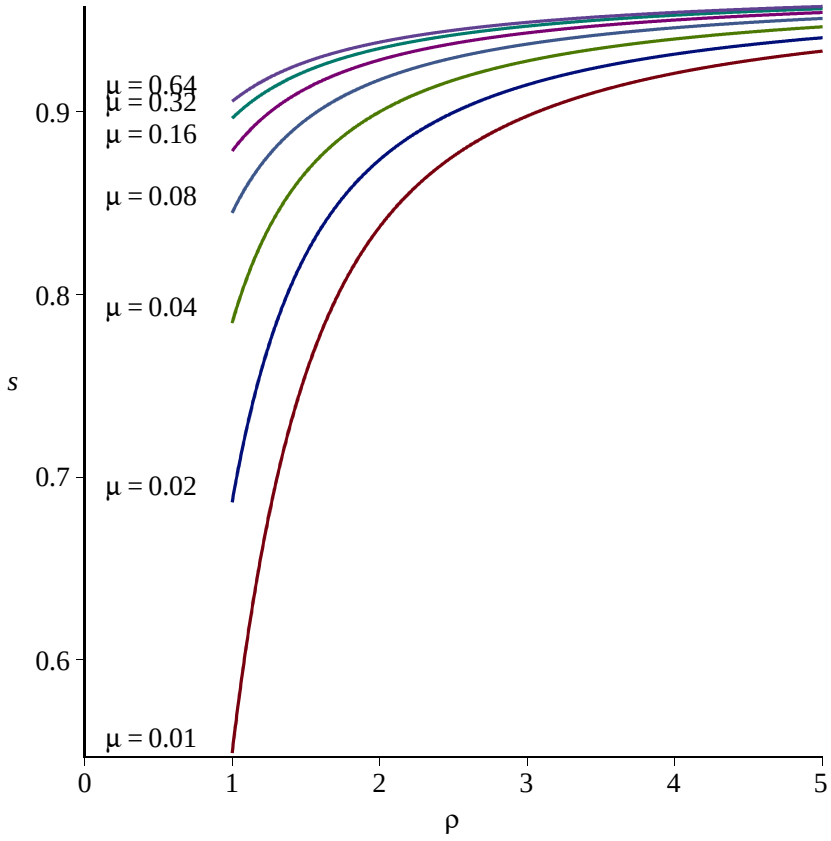
```
> ### Animated plot of m as rho and mu vary

  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, 0.548913, 0.837080, 0.897742, 0.921182, 0.933287, 0.940613, 0.945509, 0.949008
],
[0.020, 0.686226, 0.873837, 0.914863, 0.931630, 0.940609, 0.946179, 0.949968, 0.952714
],
[0.040, 0.784328, 0.900067, 0.927862, 0.939905, 0.946583, 0.950823, 0.953755, 0.955905
],
[0.080, 0.844706, 0.917578, 0.937130, 0.946063, 0.951169, 0.954474, 0.956790, 0.958505
],
[0.16, 0.878521, 0.928448, 0.943236, 0.950285, 0.954408, 0.957115, 0.959030, 0.960457 ],
[0.32, 0.896465, 0.934739, 0.946937, 0.952928, 0.956487, 0.958847, 0.960526, 0.961783 ],
[0.64, 0.905714, 0.938178, 0.949026, 0.954453, 0.957710, 0.959882, 0.961434, 0.962598 ] ]
```

```
> ### 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):
```

$$es = -\left(1.030\left(1.020\left(-\frac{0.9426\,0.9364^{\frac{1}{\rho}}}{\rho}\right.\right.\right. \tag{21}$$

$$\left.\left.\left.+0.9426\,0.9364^{\frac{1}{\rho}}\right)\left(\frac{\left(0.9901\,0.9364^{\frac{1}{\rho}}\right)^{-\rho}-1+\mu}{\mu}\right)^{\frac{1}{\rho}}+0.9901\left(1\right.\right.\right.$$

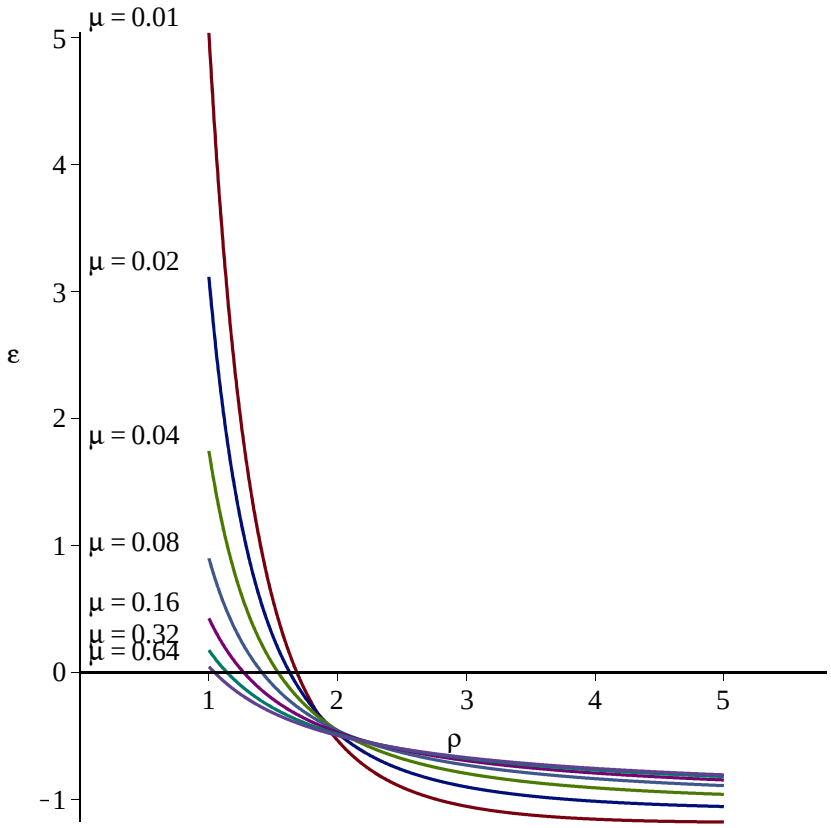
$$\left.\left.\left.-0.9709\,0.9364^{\frac{1}{\rho}}\right)\left(\frac{\left(0.9901\,0.9364^{\frac{1}{\rho}}\right)^{-\rho}-1+\mu}{\mu}\right)^{\frac{1}{\rho}}\right.\right.\right.$$

$$-\frac{1}{\rho\left(\left(0.9901\ 0.9364^{\frac{1}{\rho}}\right)^{-\rho}-1+\mu\right)}\left(0.9901\left(1-0.9709\ 0.9364^{\frac{1}{\rho}}\right)\left(\frac{\left(0.9901\ 0.9364^{\frac{1}{\rho}}\right)^{-\rho}-1+\mu}{\mu}\right)^{\frac{1}{\rho}}\left(0.9901\ 0.9364^{\frac{1}{\rho}}\right)^{-\rho}\right)\right)$$

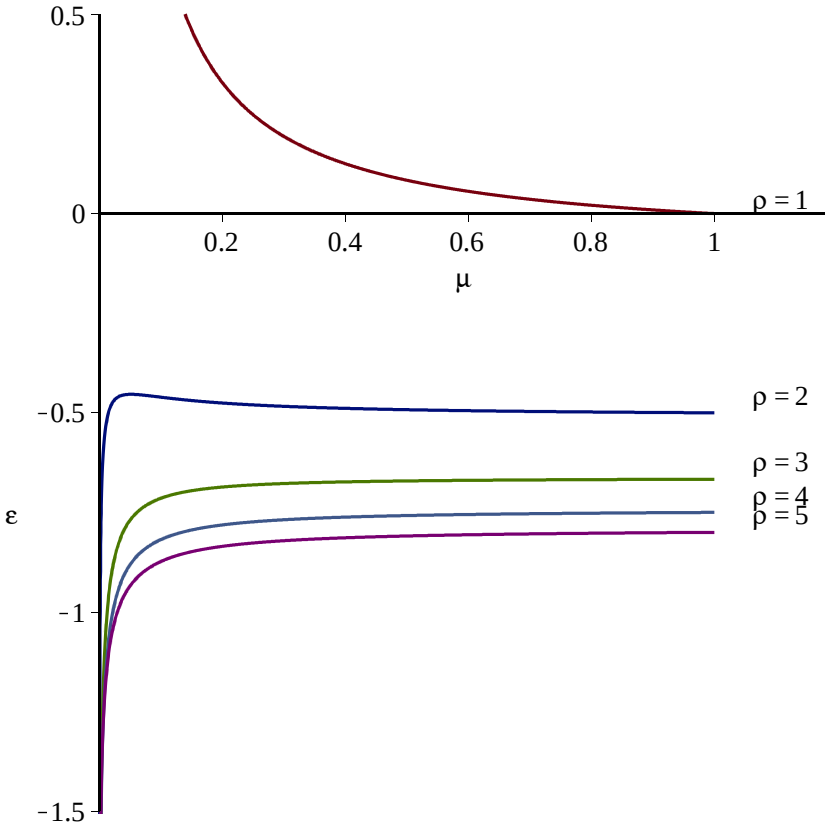
$$\Big/\left(1+1.020\left(1-0.9709\ 0.9364^{\frac{1}{\rho}}\right)\left(\frac{\left(0.9901\ 0.9364^{\frac{1}{\rho}}\right)^{-\rho}-1+\mu}{\mu}\right)^{\frac{1}{\rho}}\right)$$

```
> ### 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):
```

$$schanges = \begin{bmatrix} [0., 1., 2., 3., 4., 5., 6., 7., 8.], \\ [0.010, 4.77454, -0.604226, -1.07561, -1.15780, -1.17071, -1.16648, -1.15769, \\ -1.14806], \\ [0.020, 2.98144, -0.521825, -0.913747, -1.01282, -1.04690, -1.05999, -1.06478, \\ -1.06585], \end{bmatrix} \quad \textbf{(22)}$$

```
    [0.040, 1.68008, −0.479464, −0.799175, −0.902833, −0.948978, −0.973275,
    −0.987428, −0.996221],
    [0.080, 0.870553, −0.466527, −0.726039, −0.826257, −0.877366, −0.907646,
    −0.927338, −0.940978],
    [0.16, 0.414283, −0.469671, −0.684708, −0.778305, −0.830027, −0.862600,
    −0.884879, −0.901007],
    [0.32, 0.171319, −0.477644, −0.663761, −0.751188, −0.801810, −0.834767,
    −0.857898, −0.875007],
    [0.64, 0.0458459, −0.484609, −0.653800, −0.736919, −0.786306, −0.819022,
    −0.842284, −0.859669]]
> ##########################
> ### 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 as Matrix in Matlab data format
> ExportMatrix(cat(currentdir(),kernelopts(dirsep),convert(N,
  string),kernelopts(dirsep),"mvalues_mu_rho_",convert(N,string),".
  m")
      , evalf(mvalues), delimiter="&", format=rectangular, mode=
  ascii):
> ExportMatrix(cat(currentdir(),kernelopts(dirsep),convert(N,
  string),kernelopts(dirsep),"mchanges_mu_rho_",convert(N,string),
  ".m")
      , evalf(mchanges), delimiter="&", format=rectangular, mode=
  ascii):
> ExportMatrix(cat(currentdir(),kernelopts(dirsep),convert(N,
  string),kernelopts(dirsep),"svalues_mu_rho_",convert(N,string),".
  m")
```

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