# Interest Rate Models in `StocVal`

Nathan Esau

June 21, 2015

To perform the valuation, the 1-factor hull white interest rate model was chosen. To calibrate this model, the following term structure for USD rates on treasury bonds was used.

```
> library(StocVal)
> print(termStruct, row.names=F)
```

```
 term      RFR
    1 0.003039
    2 0.006425
    3 0.010487
    4 0.015358
    5 0.020330
    7 0.027362
    8 0.029422
    9 0.031710
   10 0.034151
   15 0.040815
   20 0.042166
   30 0.043495
```

Notice that we do not have information on bonds for all the terms between 1 and 30. However, we can use interpolation to approximate the yields for such bonds.

```
> termStruct.out <- cubic_spline(termStruct)
> print(termStruct.out, row.names=F)
```

```
 Term        RFR
    1 0.00303900
    2 0.00642500
    3 0.01048700
    4 0.01535800
    5 0.02033000
    6 0.02441579
    7 0.02736200
    8 0.02942200
```

```
    9 0.03171000
   10 0.03415100
   11 0.03622338
   12 0.03786271
   13 0.03913225
   14 0.04009527
   15 0.04081500
   16 0.04134821
   17 0.04172566
   18 0.04197161
   19 0.04211030
   20 0.04216600
   21 0.04216297
   22 0.04212545
   23 0.04207772
   24 0.04204402
   25 0.04204862
   26 0.04211577
   27 0.04226972
   28 0.04253474
   29 0.04293508
   30 0.04349500
```

Next, we would like to have information on longer term bonds. To do so, we will need to "extend" the yield curve. There are several ways to do this, but the Neilson and Siegel method is used here.

```
> NS_paramsDF <- data.frame(variable=c("m", "B0", "B1", "B2",
+   "tau1", "tau2"), value=c(NS_params[1], NS_params[2],
+   NS_params[3], NS_params[4], NS_params[5], NS_params[6]))
> print(NS_paramsDF, row.names=F)

 variable       value
        m  3.00000000
       B0  0.05140000
       B1 -0.14848102
       B2 -0.04140831
     tau1  0.31641700
     tau2  5.39576700

> termStruct.extension <- curve_extensionNS(termStruct.out)
> print(termStruct.extension, row.names=F)

  [1] 0.003034392 0.006404448 0.010432393 0.015241260 0.020126104 0.024122492
  [7] 0.026994352 0.028997480 0.031217620 0.033580800 0.035582736 0.037163513
 [13] 0.038385994 0.039312311 0.040004060 0.040516233 0.040878629 0.041114693
```

```
 [19] 0.041247791 0.041301240 0.041298328 0.041262333 0.041216528 0.041184191
 [25] 0.041188601 0.041253037 0.041400758 0.041654997 0.042038930 0.042575656
 [31] 0.042832520 0.043078212 0.043312066 0.043534718 0.043746782 0.043948850
 [37] 0.044141487 0.044325233 0.044500596 0.044668061 0.044828082 0.044981088
 [43] 0.045127484 0.045267646 0.045401930 0.045530669 0.045654173 0.045772736
 [49] 0.045886629 0.045996107 0.046101410 0.046202762 0.046300370 0.046394431
 [55] 0.046485129 0.046572635 0.046657110 0.046738704 0.046817560 0.046893811
 [61] 0.046967580 0.047038985 0.047108136 0.047175137 0.047240086 0.047303075
 [67] 0.047364189 0.047423511 0.047481118 0.047537082 0.047591473 0.047644356
 [73] 0.047695792 0.047745840 0.047794554 0.047841988 0.047888191 0.047933209
 [79] 0.047977089 0.048019872 0.048061600 0.048102310 0.048142039 0.048180823
 [85] 0.048218695 0.048255686 0.048291826 0.048327146 0.048361672 0.048395430
 [91] 0.048428447 0.048460746 0.048492351 0.048523283 0.048553564 0.048583214
 [97] 0.048612253 0.048640700 0.048668571 0.000000000
```

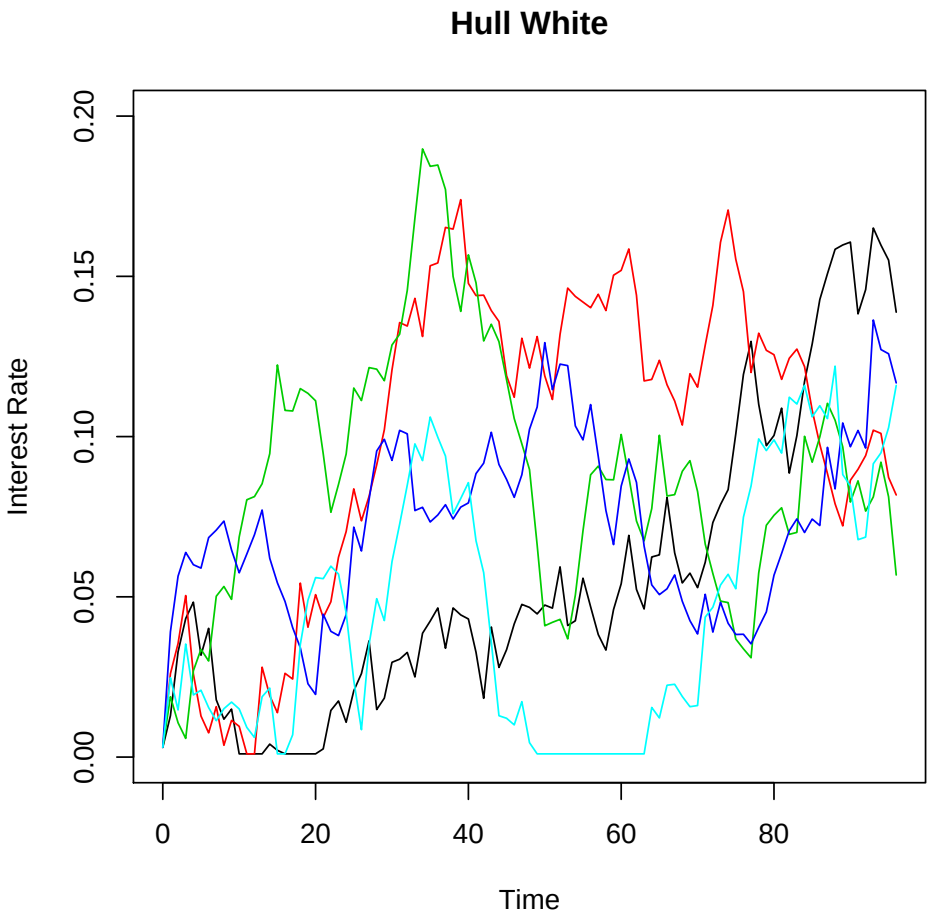
The following parameters were used for the hull white model[1]

```
> srinputDF <- data.frame(variable=c("b", "tau", "# scenarios",
+   "projection years",  "Random no. seed"), value=c(srinput[1],
+   srinput[2],srinput[3], srinput[4], srinput[5]))
> print(srinputDF, row.names=F)

         variable        value
                b 9.516603e-01
              tau 1.158078e-02
      # scenarios 1.000000e+03
 projection years 1.000000e+02
  Random no. seed 1.375310e+06
```

1000 scenarios were then generated using the hull white model. The first five scenarios are plotted below.

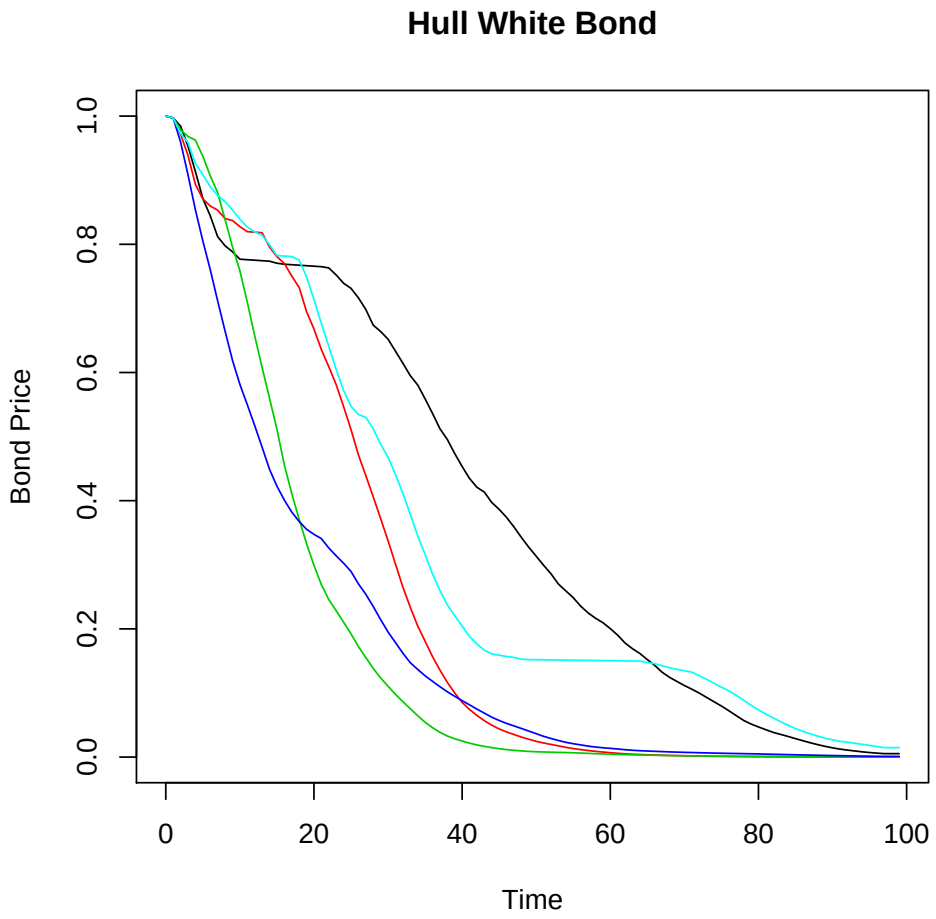
```
> hw.out <- hull_white(srinput, termStruct.extension, termofyield=1)
> plot(x=seq(0,100-4,1),y=hw.out[1,1:(100-3)],xlab="Time", ylab="Interest Rate",
+   type='l', col=1, ylim=c(0,0.20), main="Hull White")
> for(i in 2:5)
+   lines(x=seq(0,100-4,1),y=hw.out[i,1:(100-3)],xlab="Time",
+     ylab="Interest Rate", type='l', col=i)
```

[1]The `srinput` dataset as well as many other datasets used were defined in the package code file `data.R`



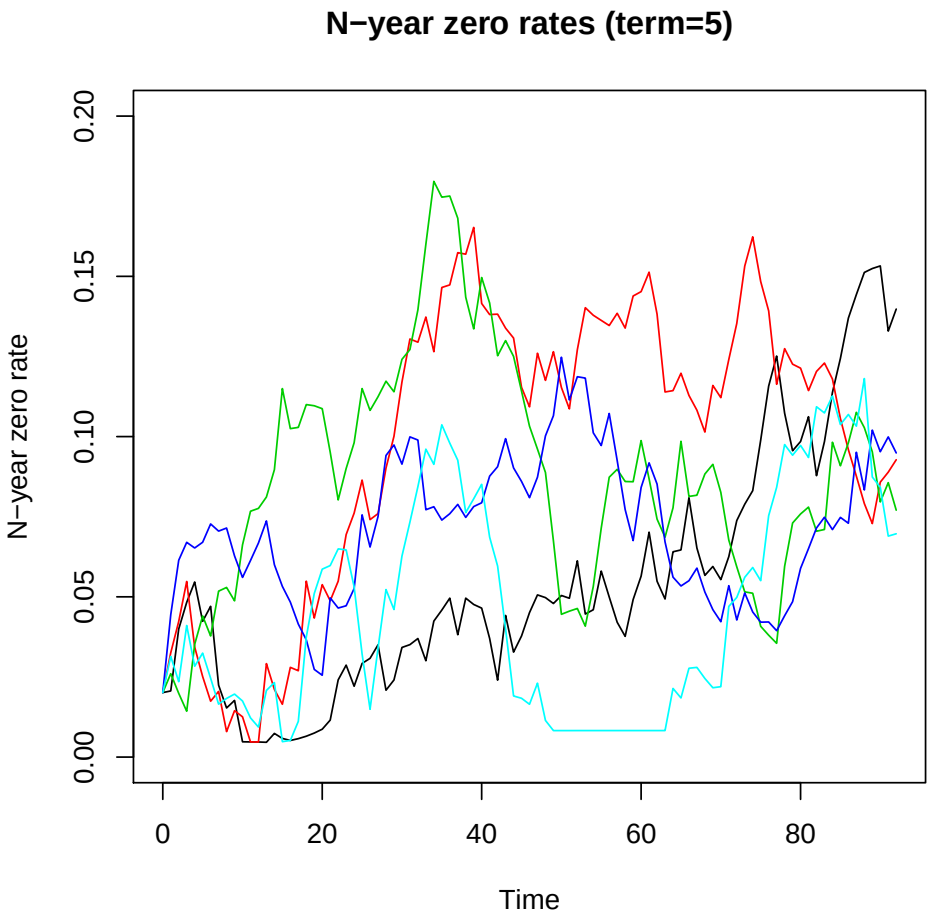

For each scenario, we need to derive bond prices. The first five scenarios are plotted below.

```
> hw.bond <- hull_whiteBond(hw.out)
> plot(x=seq(0,99,1),y=hw.bond[1,],xlab="Time", ylab="Bond Price", type='l',
+   col=1, ylim=c(0,1), main="Hull White Bond")
> for(i in 2:5)
+   lines(x=seq(0,99,1),y=hw.bond[i,],xlab="Time", ylab="Bond Price", type='l',
+     col=i)
```




The interest rates shown above represent the one year interest rate prevailing at a particular time (plotted on the $x$-axis). Using these rates, we can estimate the $n$-year rates prevailing at a particular time. The first five scenarios for 5-year rates are plotted below.

```
> nyearZero.out <- nyearZero(hw.out,termStruct.extension,srinput,term = 5)
> plot(x=seq(0,(100-5-3),1),y=nyearZero.out[1,1:(100-5-2)],xlab="Time",
+   ylab="N-year zero rate", type='l', col=1, ylim=c(0,0.20),
+   main="N-year zero rates (term=5)")
> for(i in 2:5)
+   lines(x=seq(0,(100-5-3),1),y=nyearZero.out[i,1:(100-5-2)],xlab="Time",
+     ylab="N-year zero rate", type='l', col=i)
```

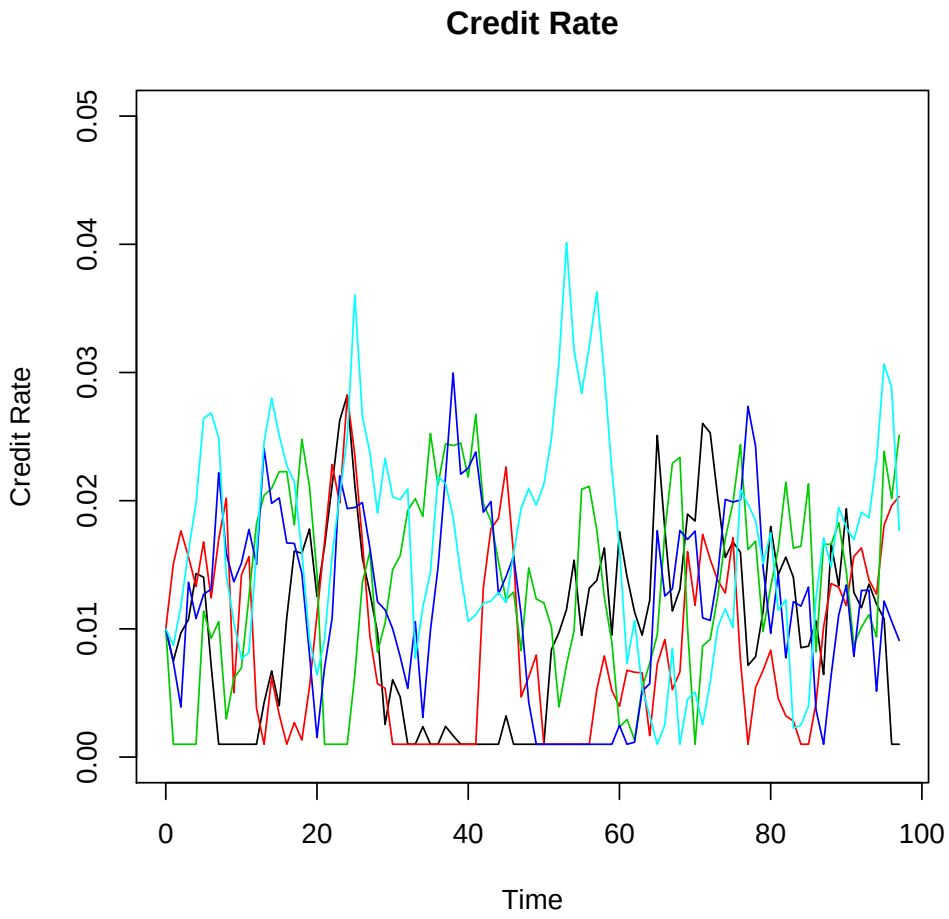



In addition to the interest rates, we can also like to generate scenarios for credit rates, equity returns, and inflation rates. The credit rates were also produced using a 1-factor hull white model. Equity returns and inflation rates were produced using a lognormal model.

```
> credit_spread <- rep(0.01, 100) # expected value
> crspinputDF <- data.frame(variable=c("b", "tau", "# scenarios",
+   "projection years",  "Random no. seed"), value=c(crspinput[1],
+   crspinput[2], crspinput[3], crspinput[4], crspinput[5]))
> print(crspinputDF, row.names=F)

         variable      value
                b      0.900
              tau      0.005
      # scenarios   1000.000
 projection years    100.000
  Random no. seed 137510.000

> hw.out2 <- hull_white(crspinput, credit_spread, termofyield=1)
> plot(x=seq(0,100-3,1),y=hw.out2[1,1:(100-2)],xlab="Time",
```

```
+      ylab="Credit Rate", type='l', col=1, ylim=c(0,0.05), main="Credit Rate")
> for(i in 2:5)
+   lines(x=seq(0,100-3,1),y=hw.out2[i,1:(100-2)],xlab="Time",
+     ylab="Credit Rate", type='l', col=i)
> print(volterm.equity) # eq. volatility assumption for different term lengths

  [1] 0.2344808 0.2439279 0.2550815 0.2552156 0.2552156 0.2552156 0.2552156
  [8] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [15] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [22] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [29] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [36] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [43] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [50] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [57] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [64] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [71] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [78] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [85] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [92] 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156 0.2552156
 [99] 0.2552156 0.2552156

> ers.out <- ers(rpi, volterm.equity, termStruct.extension) # risk premium = 0
> plot(x=seq(1,100,1),y=ers.out[1,],xlab="Time", ylab="Excess Equity Return",
+   type='l', col=1, ylim=c(-0.5,1.2), main="Excess Equity Return")
> for(i in 2:5)
+   lines(x=seq(1,100,1),y=ers.out[i,],xlab="Time", ylab="Excess Equity Return",
+     type='l', col=i)
> ersTotal.out <- ersTotal(hw.out, ers.out)
> plot(x=seq(1,100,1),y=ersTotal.out[1,],xlab="Time", ylab="Equity Total Return",
+      type='l', col=1, ylim=c(-0.5,1.2), main="Equity Total Return")
> for(i in 2:5)
+   lines(x=seq(1,100,1),y=ersTotal.out[i,],xlab="Time",
+     ylab="Equity Total Return", type='l', col=i)
> print(inflation_mean) # expected inflation

  [1] 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023
 [13] 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023
 [25] 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023
 [37] 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023
 [49] 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023
 [61] 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023
 [73] 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023
 [85] 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023 0.023
 [97] 0.023 0.023 0.023 0.023
```

```
> print(volterm.inflation) # inflation volatility assumption

  [1] 0.01931348 0.03179106 0.04437992 0.05694939 0.06860786 0.08182150
  [7] 0.09503515 0.10824880 0.12146245 0.13467610 0.13467610 0.13467610
 [13] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [19] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [25] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [31] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [37] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [43] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [49] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [55] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [61] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [67] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [73] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [79] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [85] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [91] 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610 0.13467610
 [97] 0.13467610 0.13467610 0.13467610 0.13467610

> inflationrs.out <- inflationrs(inflation_in, inflation_mean, volterm.inflation)
> plot(x=seq(1,100,1),y=inflationrs.out[1,],xlab="Time", ylab="Inflation Rate",
+   type='l', col=1, ylim=c(-0.35,0.5), main="Inflation")
> for(i in 2:5)
+   lines(x=seq(1,100,1),y=inflationrs.out[i,],xlab="Time", ylab="Inflation Rate",
+     type='l', col=i)
```




When generating the stochastic scenarios used in the valuation, we need to consider the correlation between the interest rate, the credit spread, the excess equity return and the inflation rate. In order to generate the scenarios, we need to find the cholesky decomposition of the correlation matrix between these variables.

```
> cholinputDF <- data.frame(cholinput)
> names(cholinputDF) <- c("short_rate", "credit_spread", "excess_return",
+   "inflation_rate")
> row.names(cholinputDF) <- c("short_rate", "credit_spread", "excess_return",
+   "inflation_rate")
> print(cholinputDF)
```

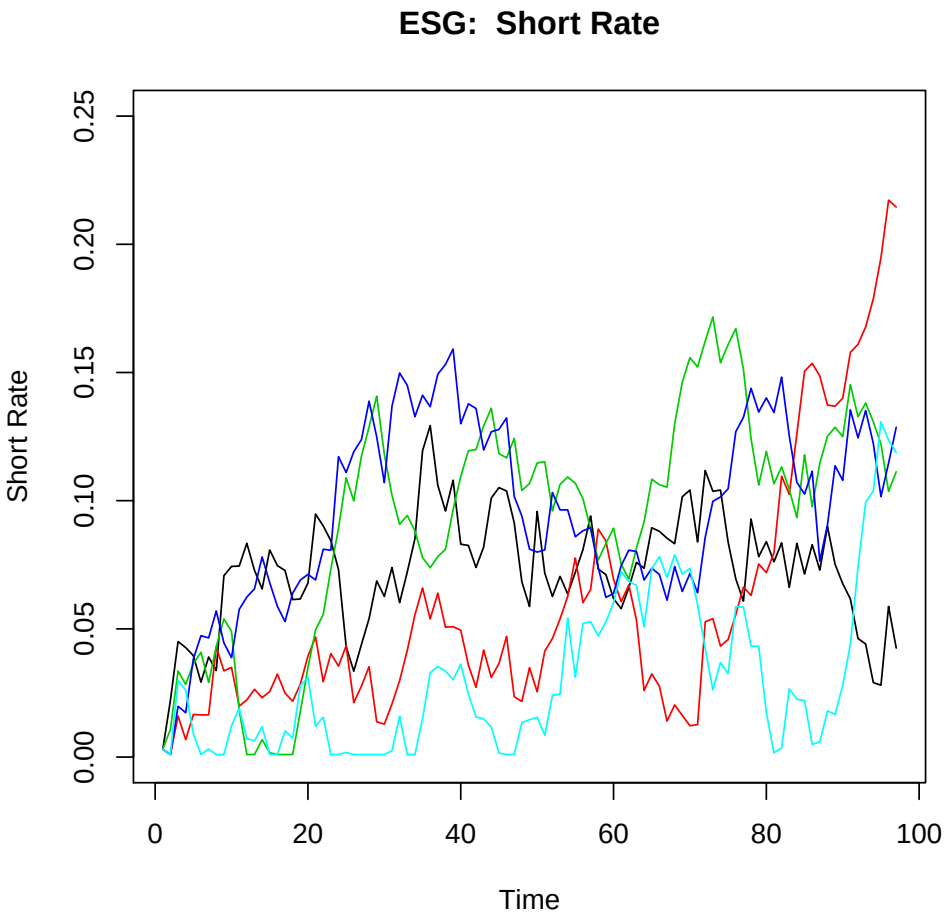
```
               short_rate credit_spread excess_return inflation_rate
short_rate      1.0000000  -0.226739420     0.2040000    0.312225859
credit_spread  -0.2267394   1.000000000    -0.1677961   -0.006969051
excess_return   0.2040000  -0.167796135     1.0000000   -0.210583579
inflation_rate  0.3122259  -0.006969051    -0.2105836    1.000000000
```

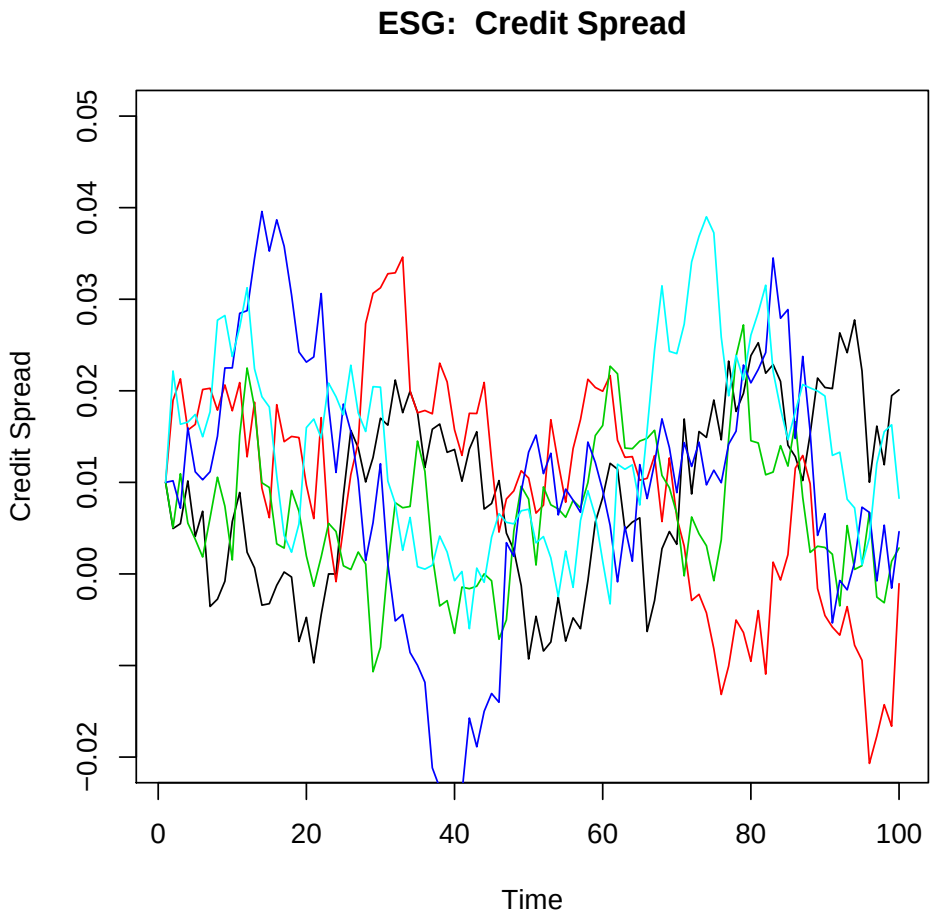
```
> cholesky.out <- cholesky(cholinput)
> print(cholesky.out)
```

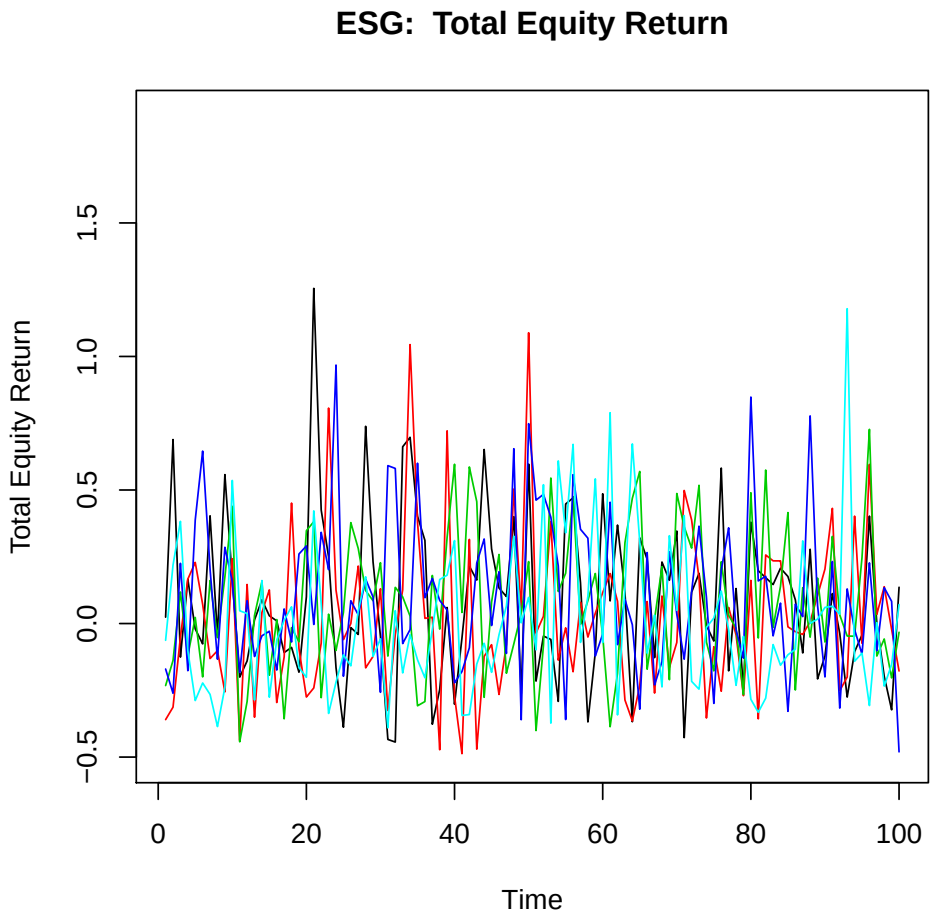
```
           [,1]       [,2]       [,3]         [,4]
[1,]  1.0000000 -0.2267394  0.2040000  0.312225859
[2,] -0.2267394  1.0000000 -0.1677961 -0.006969051
[3,]  0.2040000 -0.1247914  1.0000000 -0.210583579
[4,]  0.3122259  0.0655316 -0.2740516  1.000000000
```

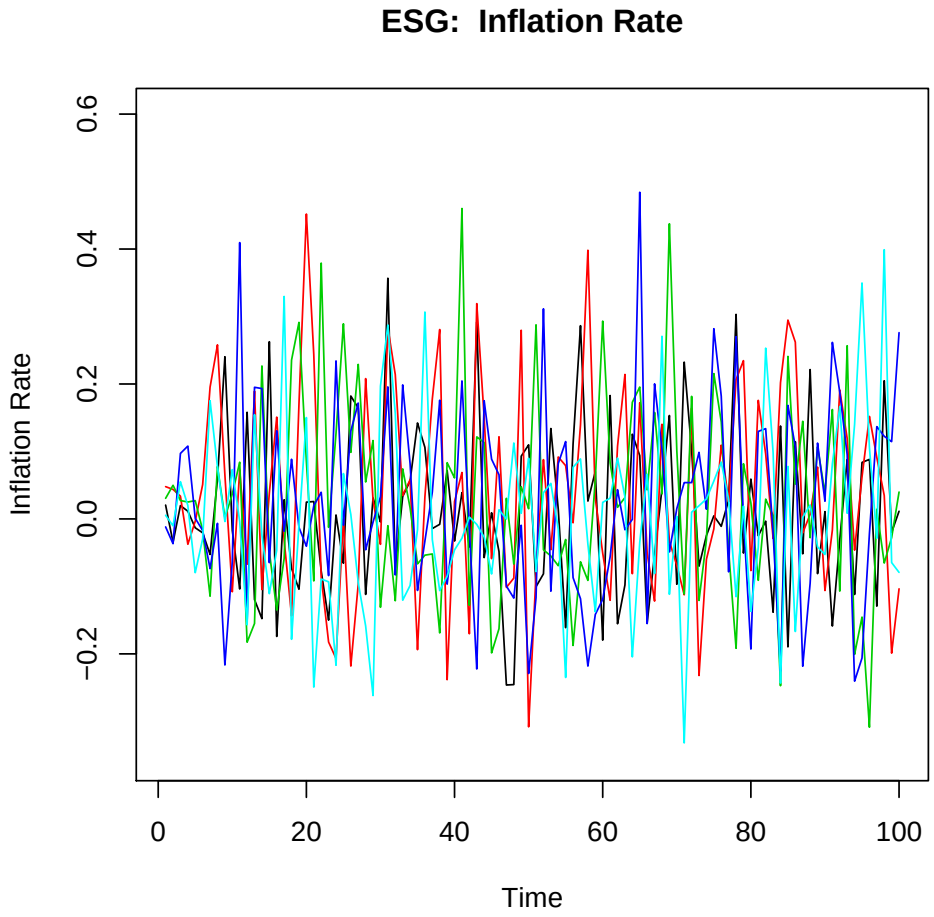
We can then calculate stochastic scenarios for each of these variables. The plot for the first five scenarios for each variable is shown below.

```
> esga.out <- esga(esgin, credit_spread, volterm.equity, inflation_mean,
+   volterm.inflation , cholesky.out, termStruct.extension)
> indexes <- c("Short Rate", "Credit Spread", "Total Equity Return",
+   "Inflation Rate")
> xremove <- c(3,0,0,0)
> ylim <- list(sr=c(0,0.25), cs=c(-0.02,0.05), ter=c(-0.5,1.9), ir=c(-0.35,0.6))
> for(j in 1:4) {
+   file=paste("esgafile", j, ".pdf", sep="")
+   pdf(file=file, width=6, height=6)
+   plot(x=seq(1,100-xremove[j],1),y=esga.out[[j]][1,1:(100-xremove[j])],
+     xlab="Time", ylab=indexes[j], type='l', col=1, ylim=ylim[[j]],
+     main=paste("ESG: ", indexes[j]))
+   for(i in 2:5)
+     lines(x=seq(1,100-xremove[j],1),
+       y=esga.out[[j]][i,1:(100-xremove[j])],xlab="Time", ylab=indexes[j],
+       type='l', col=i)
+   dev.off()
+   cat("\\includegraphics{", file, "}\n\n", sep="")
+ }
```


ESG: Short Rate
Short Rate
0.00
0.05
0.10
0.15
0.20
0.25
0
20
40
60
80
100
Time


ESG: Credit Spread
Credit Spread
0.05
0.04
0.03
0.02
0.01
0.00
−0.02
0
20
40
60
80
100
Time

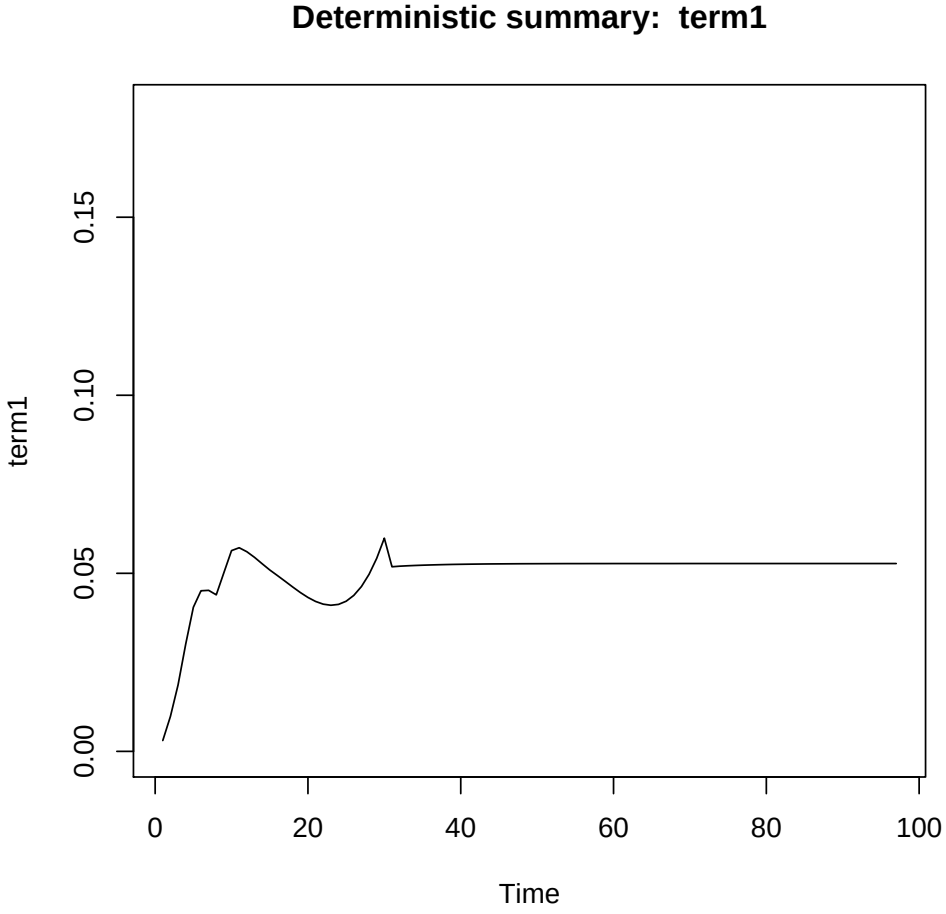

ESG: Total Equity Return
Total Equity Return
1.5
1.0
0.5
0.0
−0.5
0
20
40
60
80
100
Time

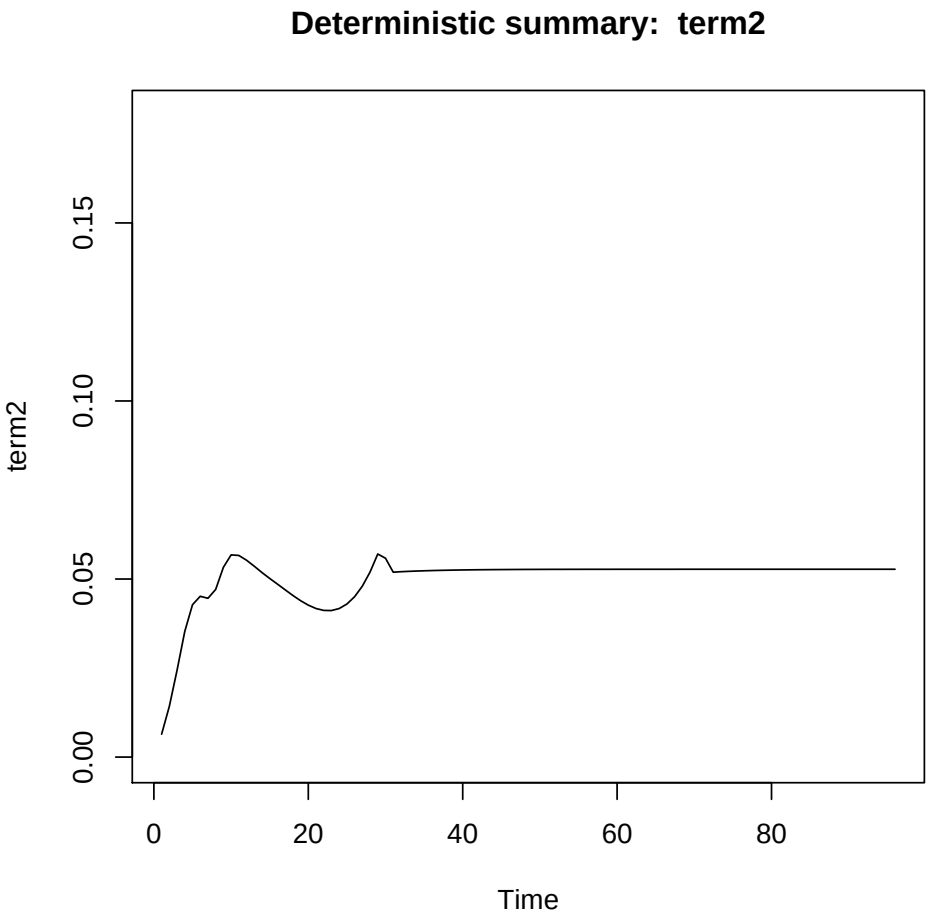


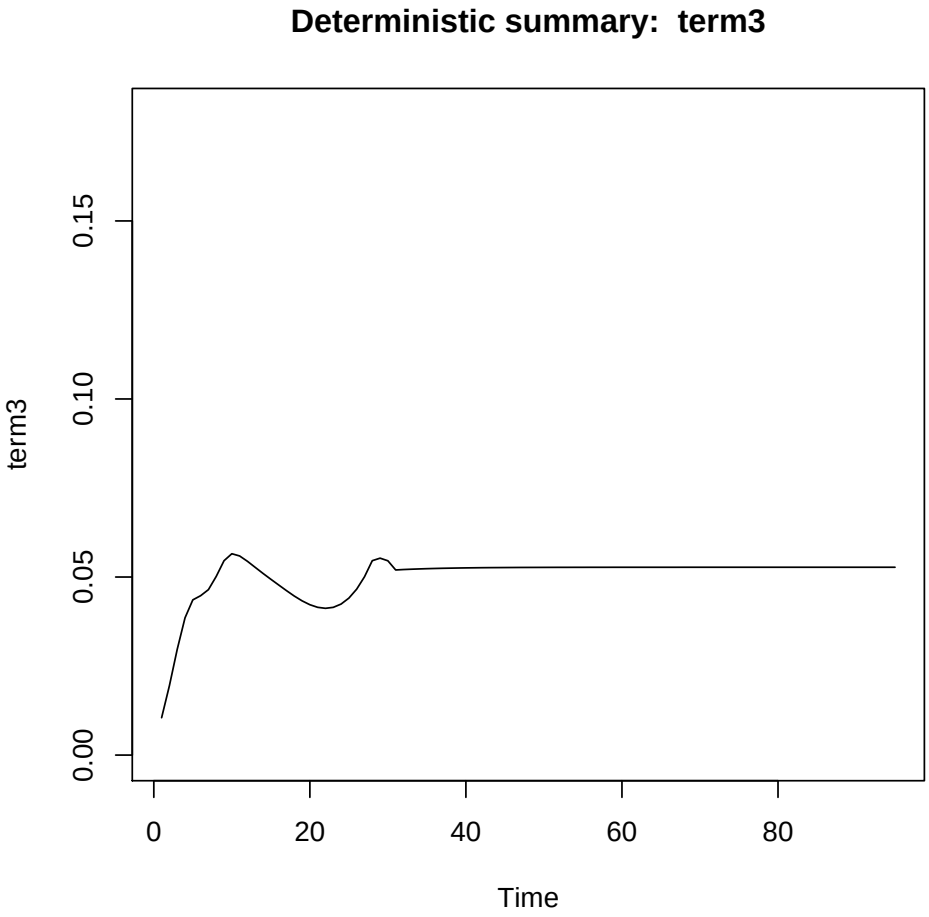

Before summarizing the stochastic scenarios, the deterministic scenario for interest rates with terms 1, 2, 3, 5, 7, 10, 15, 20, 30, inflation rates, credit rates, an equity index and a bond index are shown in plots below. The deterministic scenario (unlike the stochastic scenarios) does not involve generating random numbers.

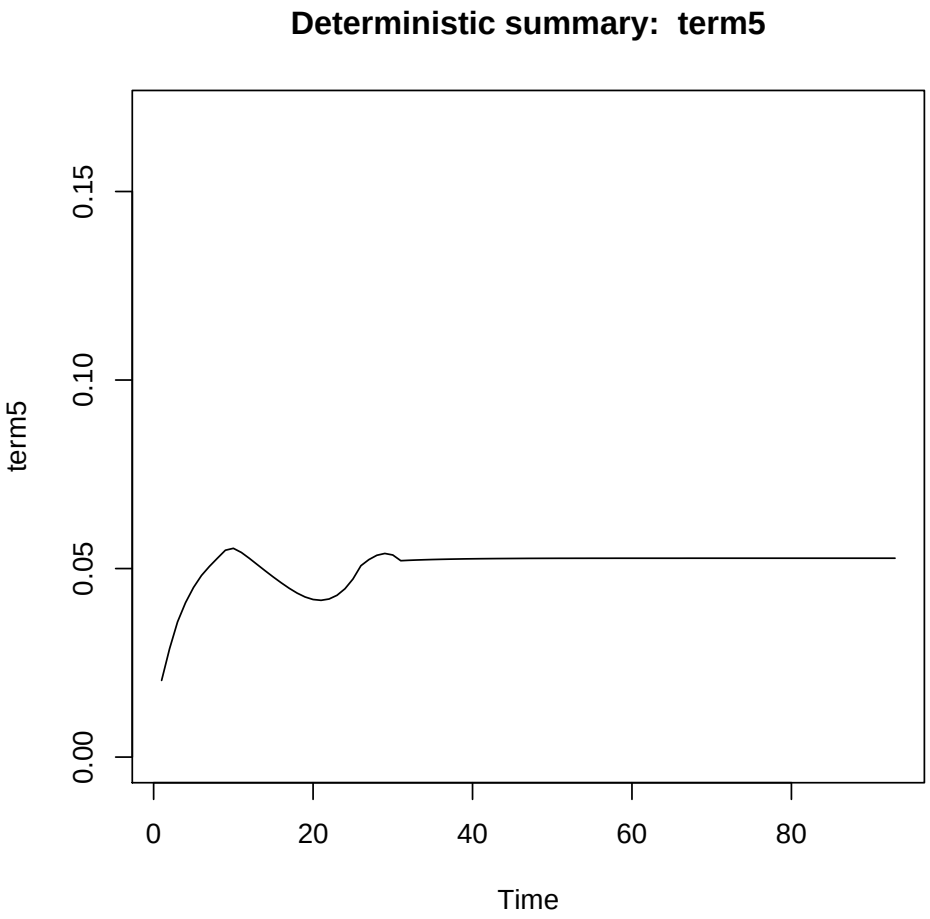
```
> determScenario.out <- determScenario(termStruct.out)
> plotNames <- names(determScenario.out)
> ylim <- list(term1=c(0,0.18), term2=c(0,0.18), term3=c(0,0.18),
+   term5=c(0,0.17), term7=c(0.01,0.16), term10=c(0.01,0.15),
+   term15=c(0.01,0.15), term20=c(0.01,0.15), term30=c(0.02,0.15),
+   inflation=c(-0.35,0.6), credit=c(-0.04,0.05), total=c(0,0.20),
+   equity=c(-0.5,150), bond=c(0,150))
> xremove <- c(3,3+(2-1),3+(3-1),3+(5-1),3+(7-1),3+(10-1),3+(15-1),3+(20-1),
+   3+(30-1),0,0,0,0,0)
> for(j in 1:14) {
+   file=paste("deterfile", j, ".pdf", sep="")
+   pdf(file=file, width=6, height=6)
+   plot(x=seq(1,100-xremove[j],1),y=determScenario.out[[j]][1:(100-xremove[j])],
+     xlab="Time", ylab=plotNames[j], type='l', col=1, ylim=ylim[[j]],
```

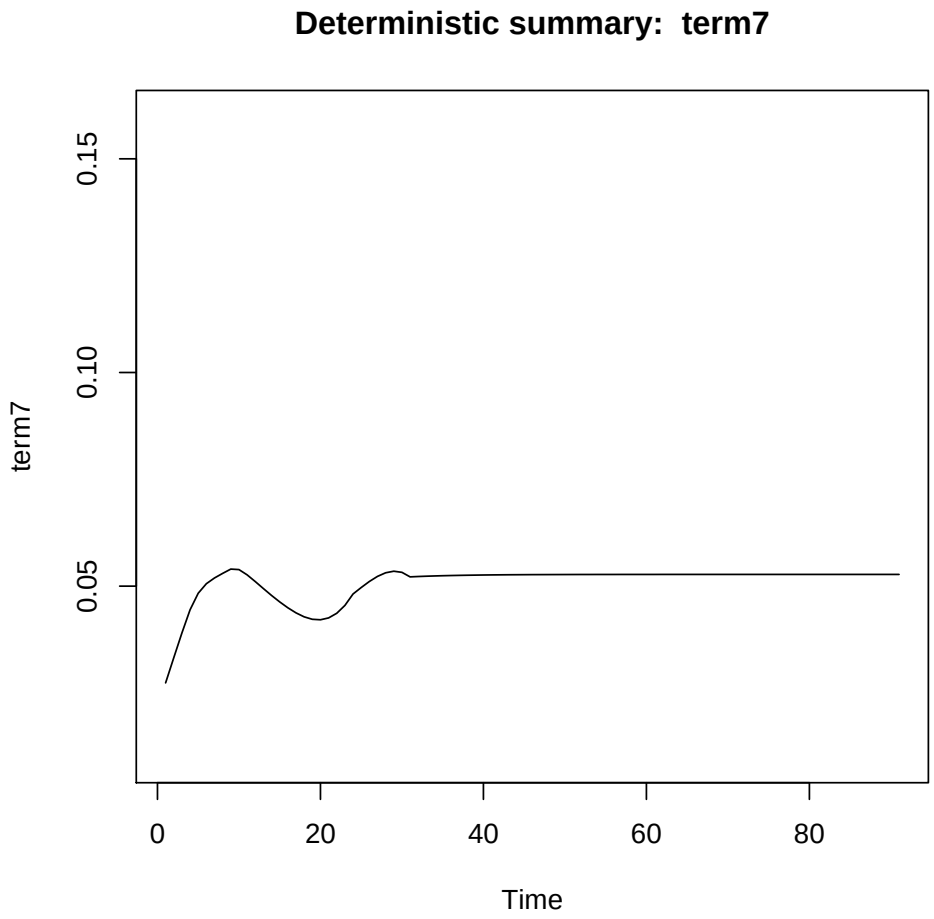
```
+     main=paste("Deterministic summary: ", plotNames[j]))
+   dev.off()
+   cat("\\includegraphics{", file, "}\n\n", sep="")
+ }
```

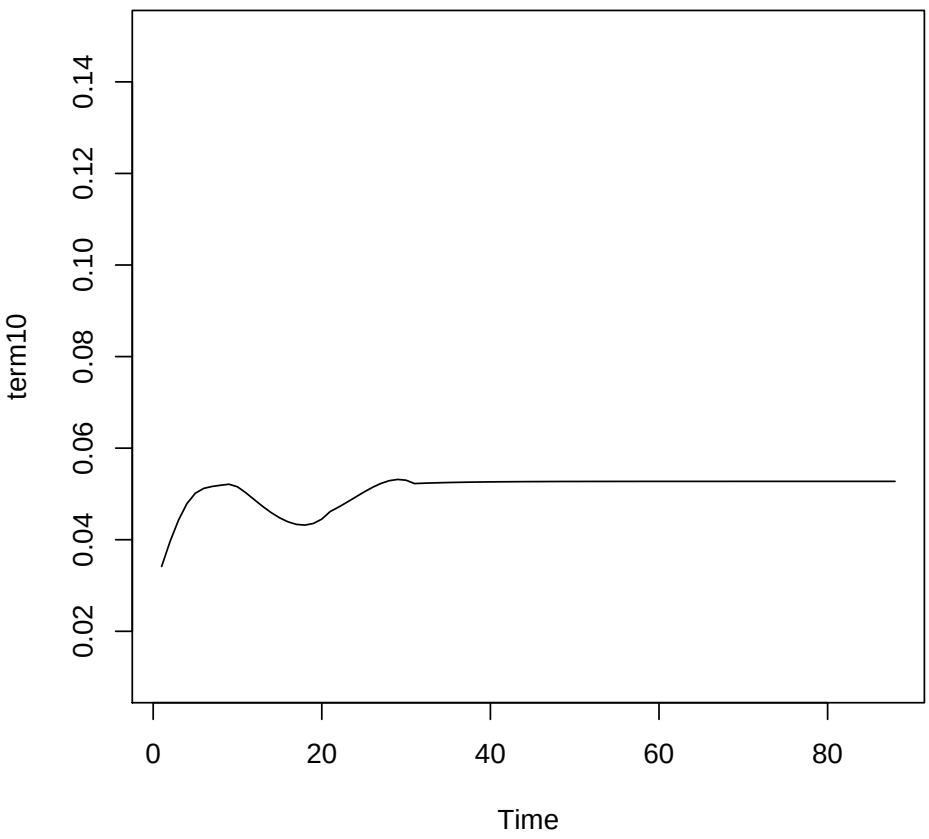


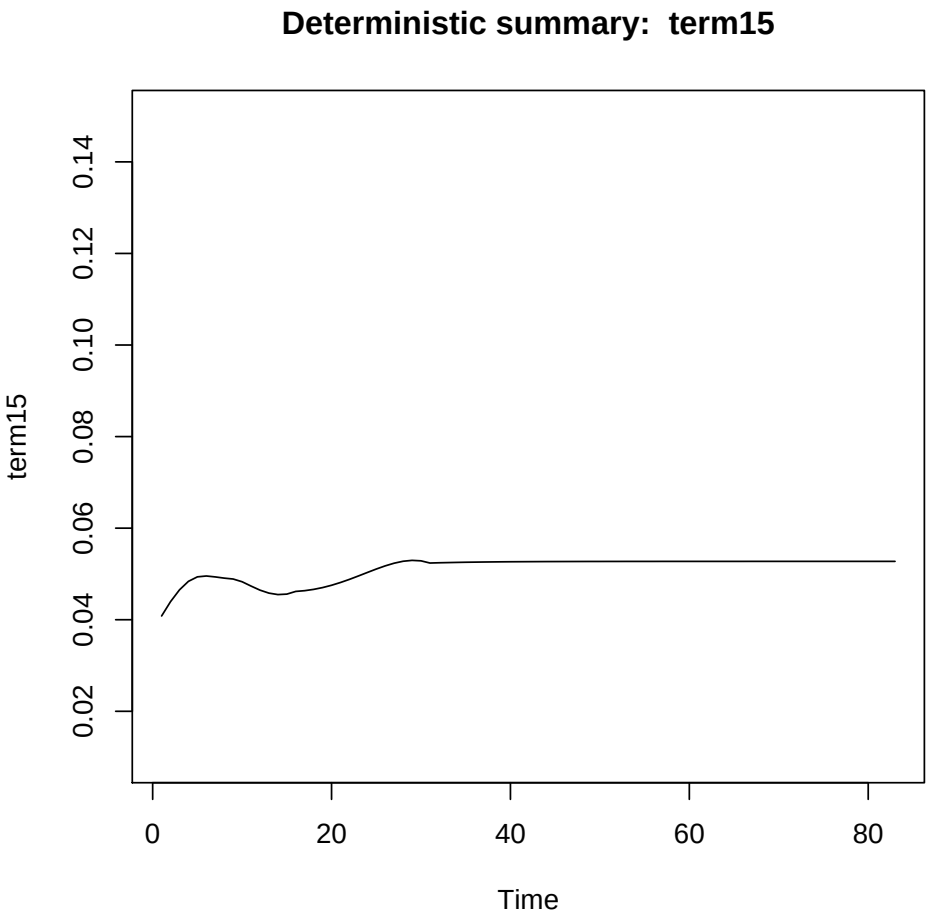

Deterministic summary: term2
term2
0.00
0.05
0.10
0.15
0
20
40
60
80
Time

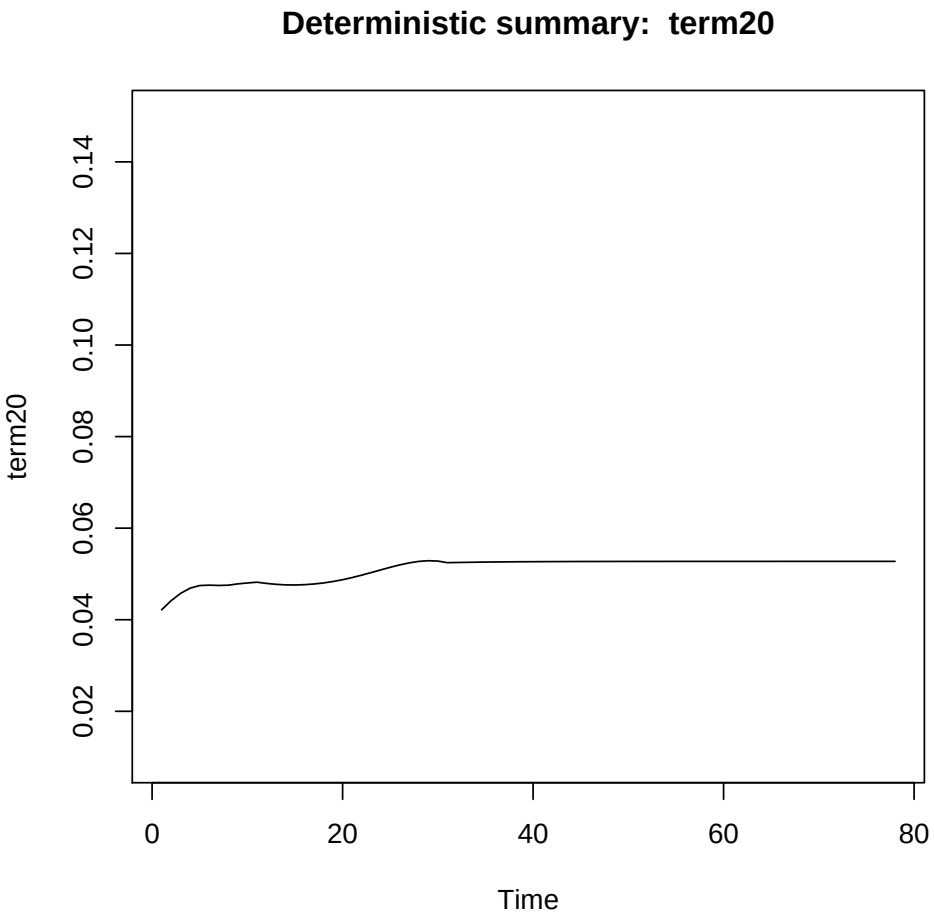

Deterministic summary: term3
term3
0.00
0.05
0.10
0.15
0
20
40
60
80
Time

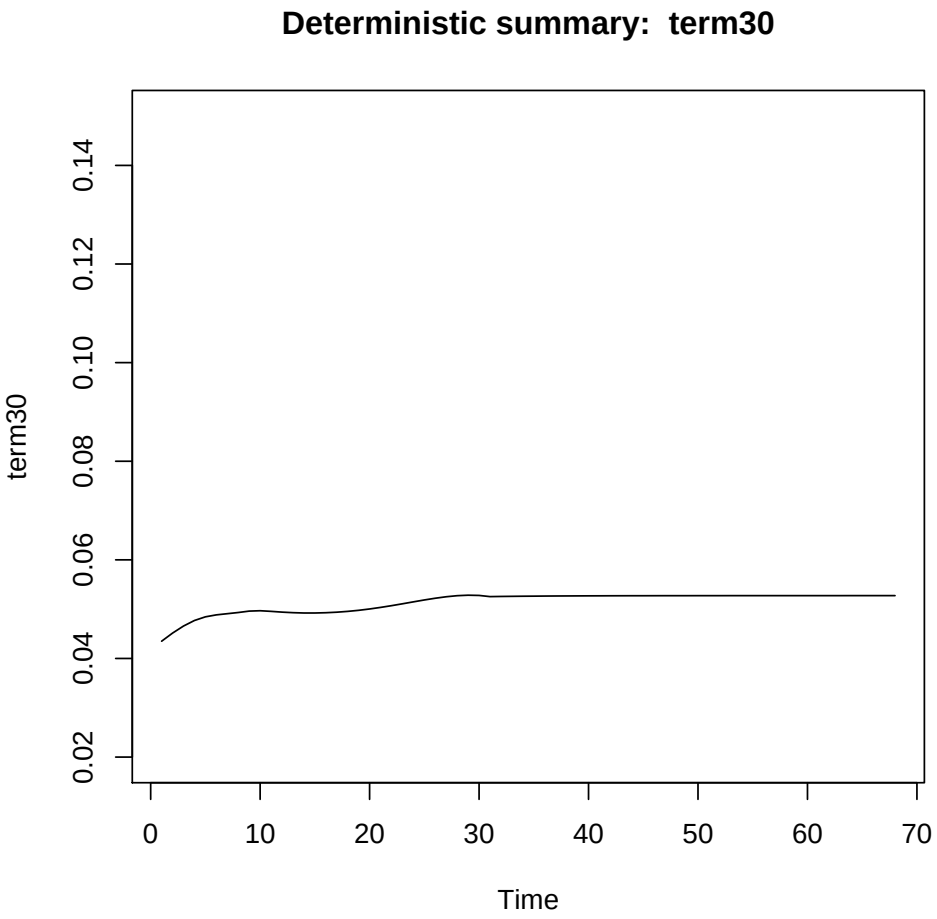

Deterministic summary: term5
term5
0.00
0.05
0.10
0.15
0
20
40
60
80
Time

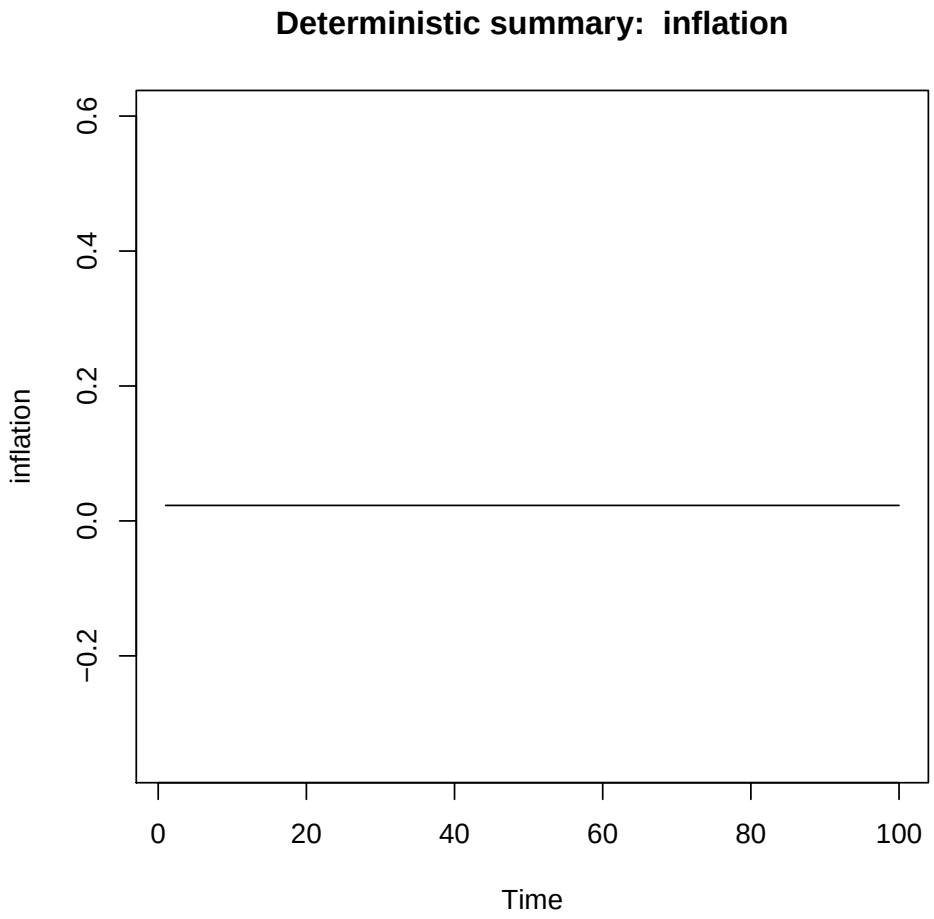

Deterministic summary: term7
term7
0.05
0.10
0.15
0
20
40
60
80
Time

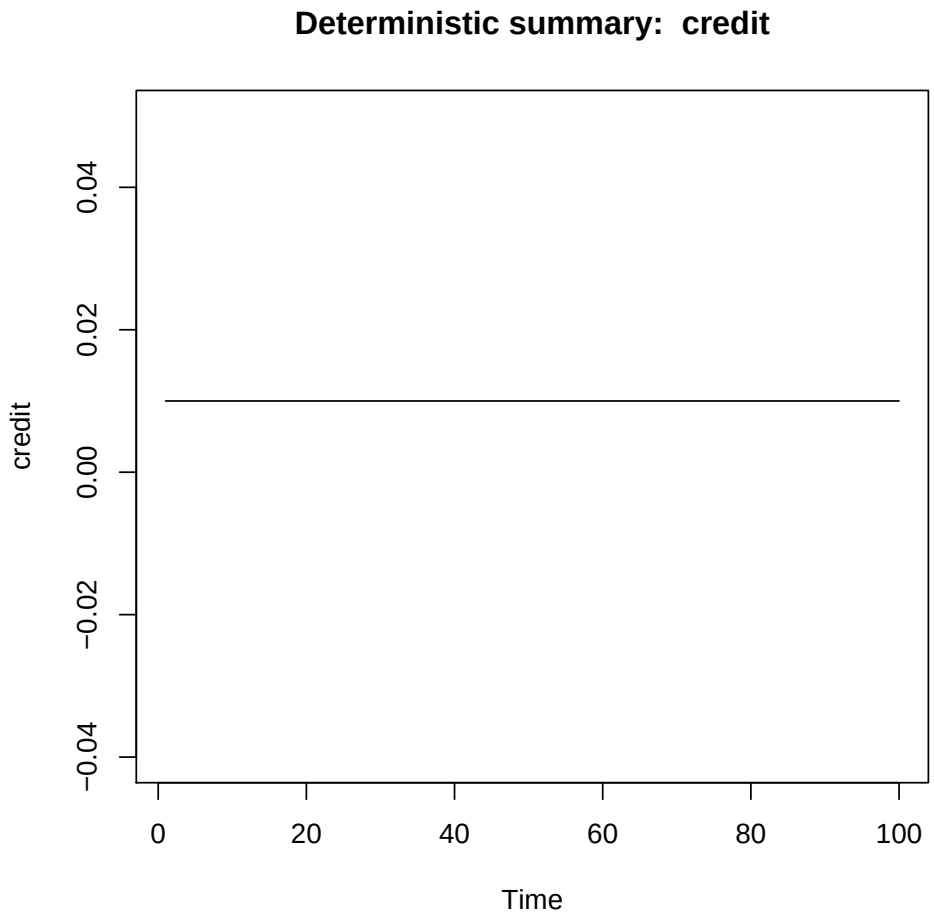

Deterministic summary: term10
term10
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0
20
40
60
80
Time

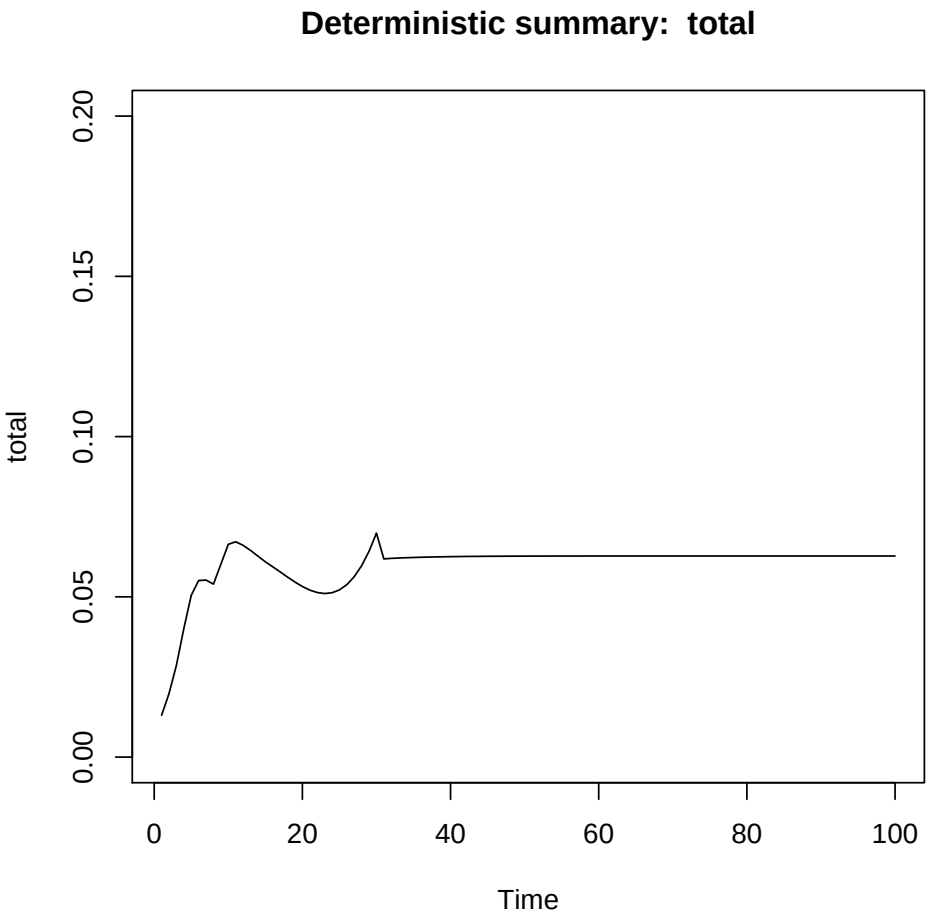

Deterministic summary: term15
term15
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0
20
40
60
80
Time

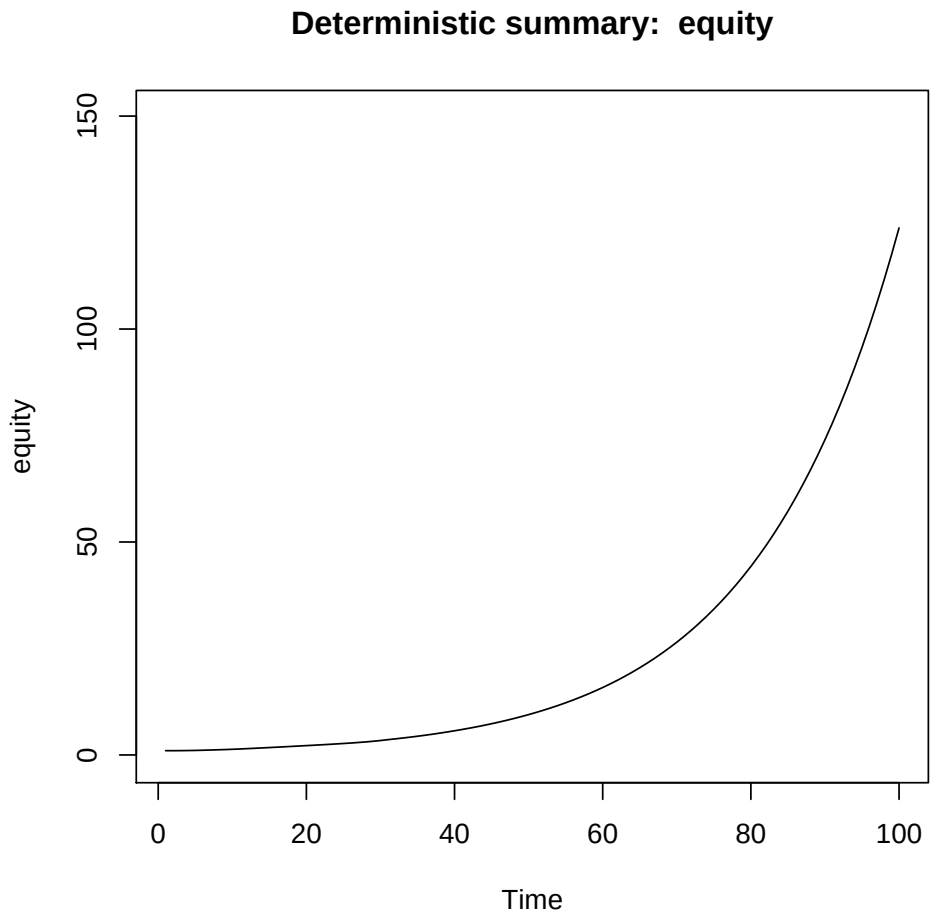

Deterministic summary: term20
term20
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0
20
40
60
80
Time

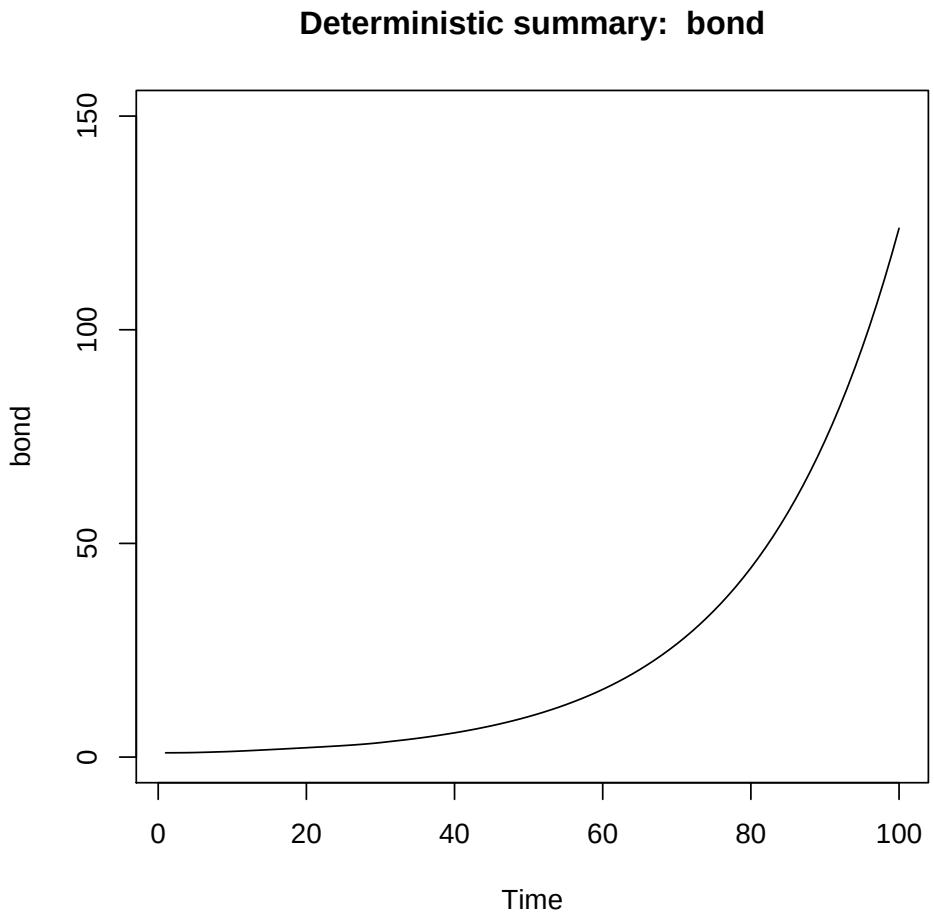

Deterministic summary: term30
term30
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0
10
20
30
40
50
60
70
Time


Deterministic summary: inflation
inflation
0.6
0.4
0.2
0.0
−0.2
0
20
40
60
80
100
Time


Deterministic summary: credit
credit
0.04
0.02
0.00
−0.02
−0.04
0
20
40
60
80
100
Time


Deterministic summary: total
total
0.00
0.05
0.10
0.15
0.20
0
20
40
60
80
100
Time

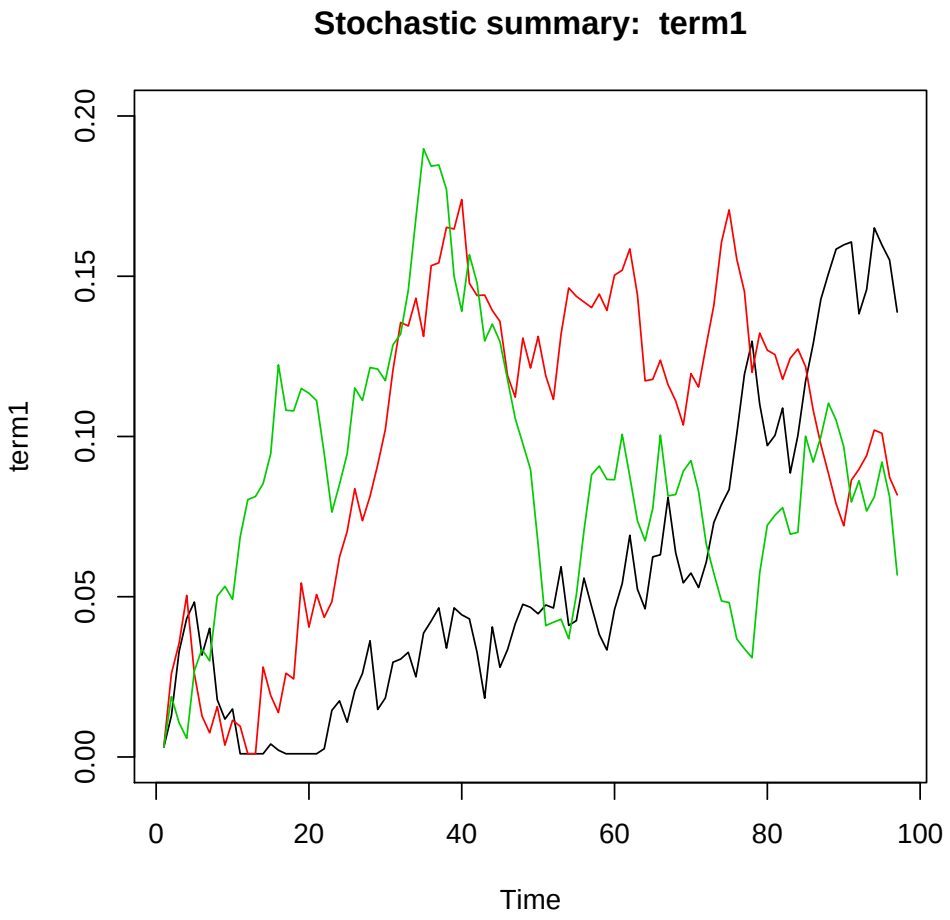

Deterministic summary: equity
equity
0
50
100
150
0
20
40
60
80
100
Time

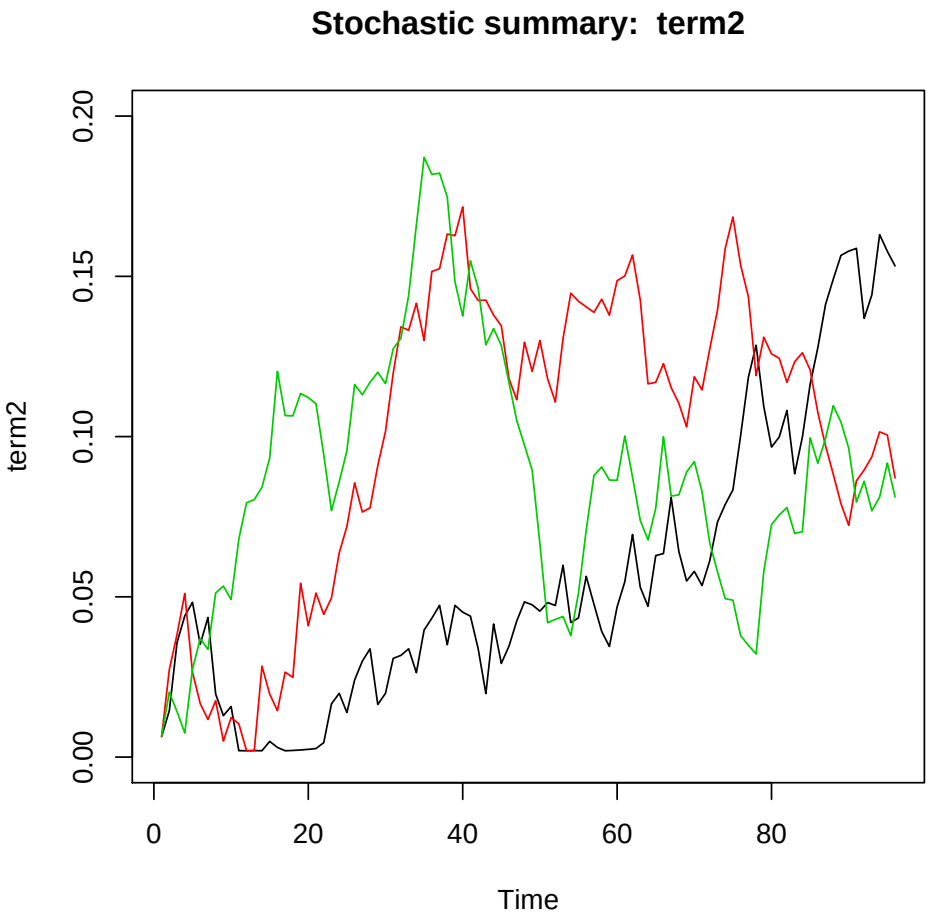


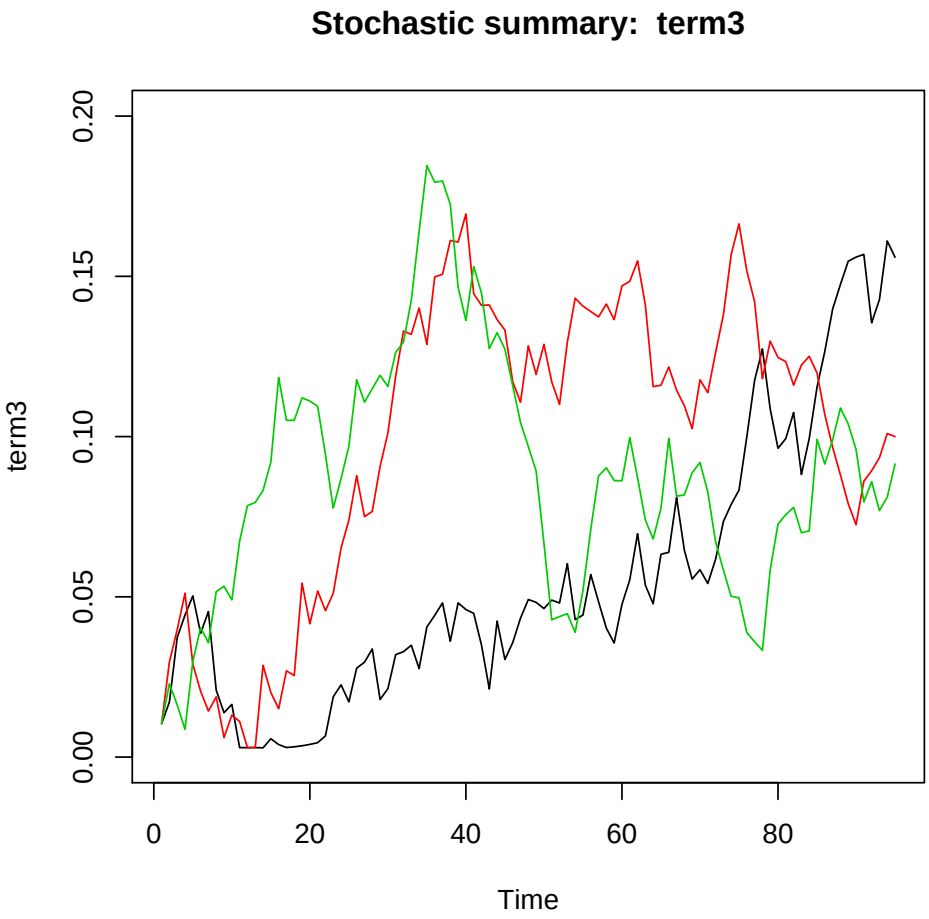

Finally, the first *three* scenarios for interest rates with terms 1, 2, 3, 5, 7, 10, 15, 20, 30, inflation rates, credit rates, total equity return, an equity index and a bond index are shown in plots below.

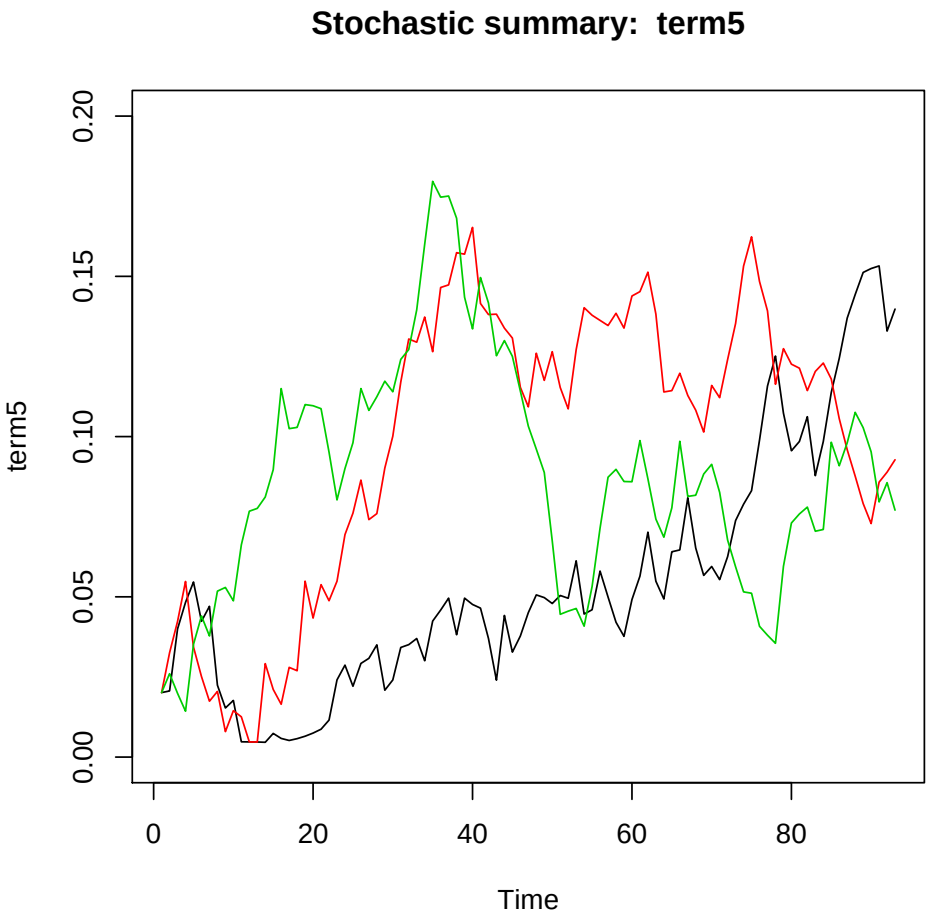
```
> stochasticScenarios.out <- stochasticScenarios(hw.out,termStruct.extension,
+   srinput, esga.out)
> plotNames <- names(stochasticScenarios.out)
> ylim <- list(term1=c(0,0.20), term2=c(0,0.20), term3=c(0,0.20),
+   term5=c(0,0.20), term7=c(0.01,0.20), term10=c(0.01,0.20),
+   term15=c(0.01,0.20), term20=c(0.01,0.20), term30=c(0.02,0.20),
+   inflation=c(-0.35,0.6), credit=c(-0.04,0.05), total=c(0,0.20),
+   equity=c(0,300), bond=c(0,10000))
> xremove <- c(3,3+(2-1),3+(3-1),3+(5-1),3+(7-1),3+(10-1),3+(15-1),
+   3+(20-1),3+(30-1),0,0,3,0,0)
> for(j in 1:14) {
+   file=paste("stocfile", j, ".pdf", sep="")
+   pdf(file=file, width=6, height=6)
+   plot(x=seq(1,100-xremove[j],1),
+     y=stochasticScenarios.out[[j]][1,1:(100-xremove[j])], xlab="Time",
```

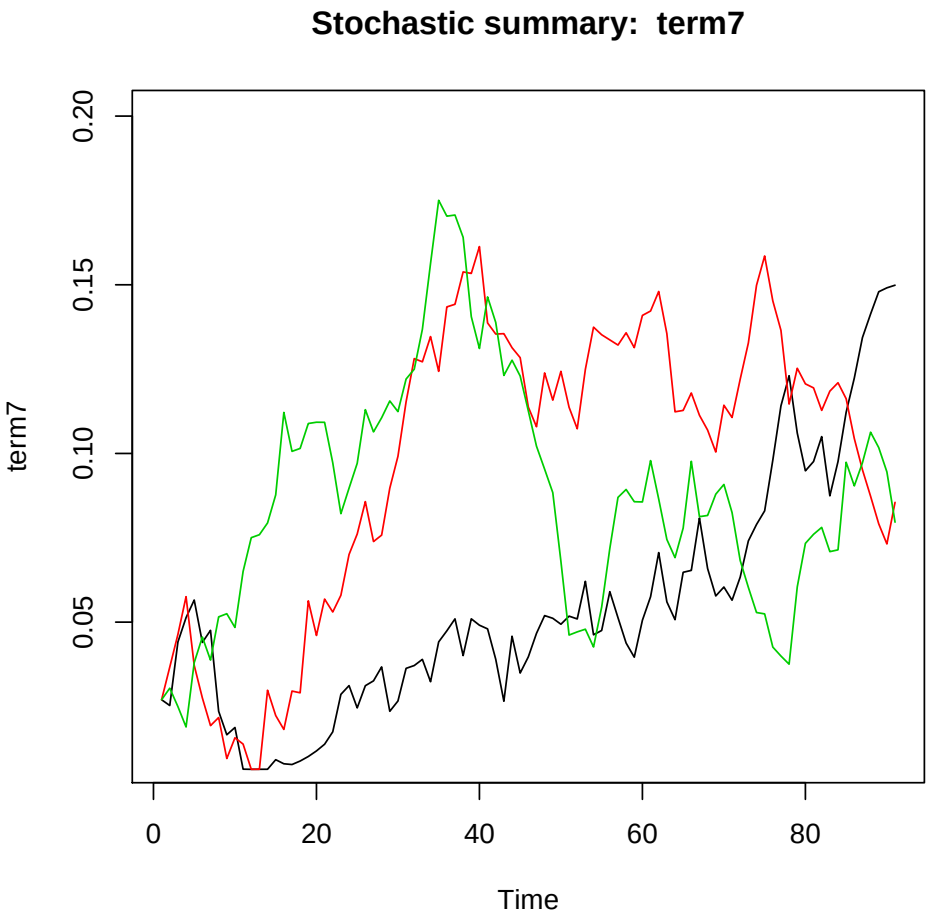
```
+     ylab=plotNames[j], type='l', col=1, ylim=ylim[[j]],
+     main=paste("Stochastic summary: ", plotNames[j]))
+   for(i in 2:3)
+     lines(x=seq(1,100-xremove[j],1),
+       y=stochasticScenarios.out[[j]][i,1:(100-xremove[j])], xlab="Time",
+       ylab=plotNames[j], type='l', col=i)
+   dev.off()
+   cat("\\includegraphics{", file, "}\n\n", sep="")
+ }
```

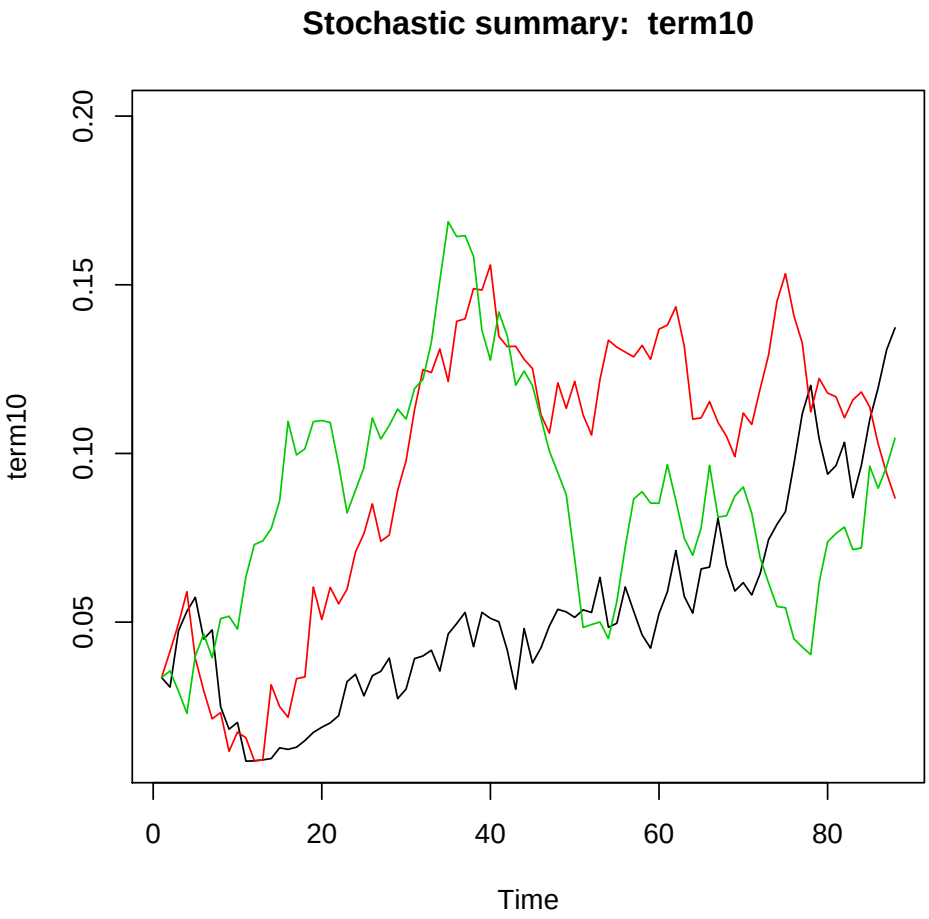


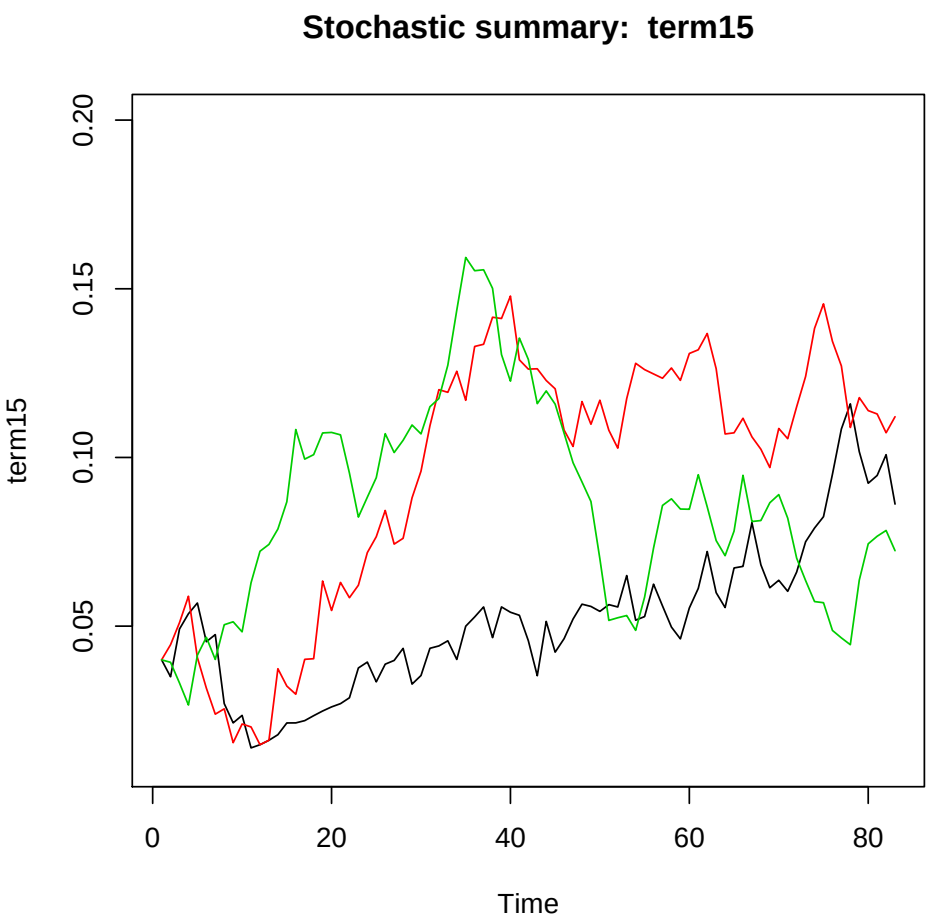

Stochastic summary: term2
term2
0.00
0.05
0.10
0.15
0.20
0
20
40
60
80
Time

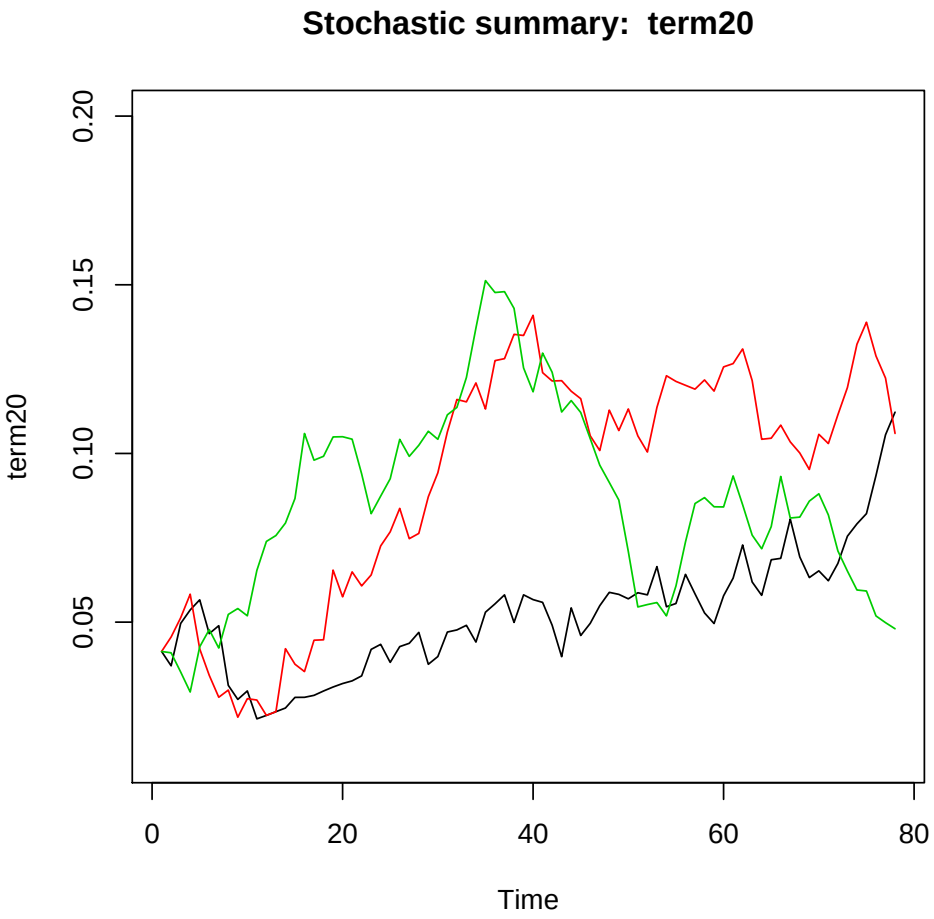

Stochastic summary: term3
term3
0.00
0.05
0.10
0.15
0.20
0
20
40
60
80
Time

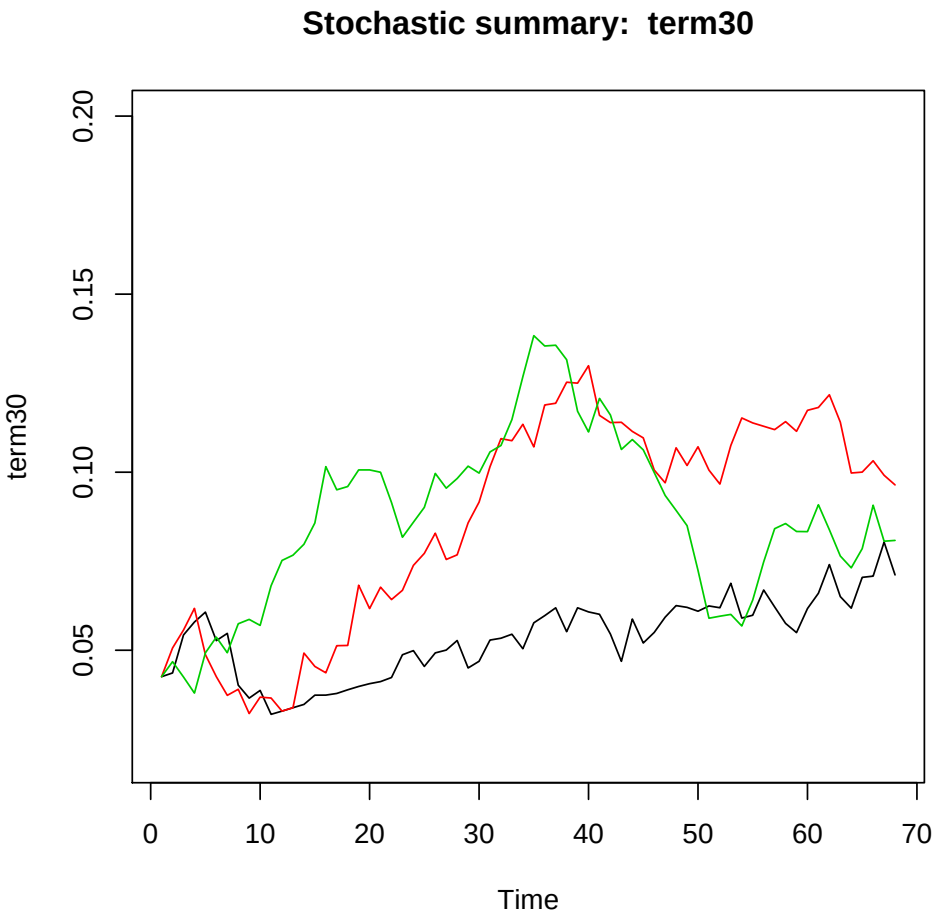

Stochastic summary: term5
term5
0.00
0.05
0.10
0.15
0.20
0
20
40
60
80
Time

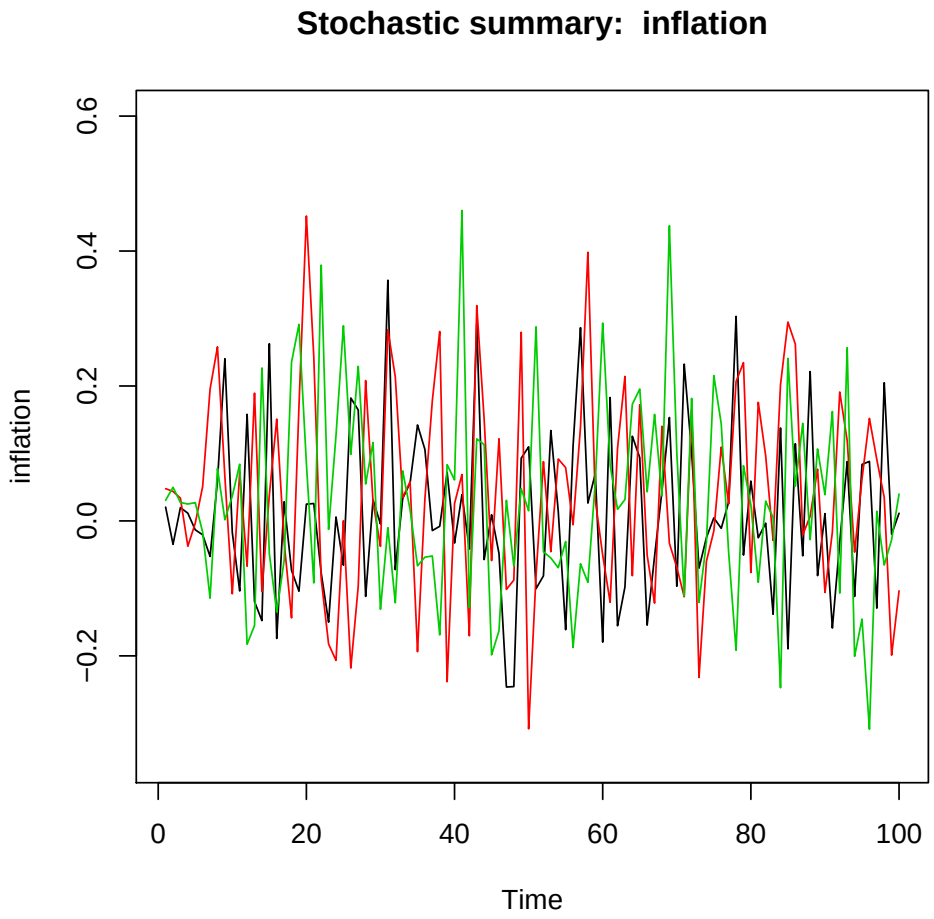

Stochastic summary: term7
term7
0.05
0.10
0.15
0.20
0
20
40
60
80
Time

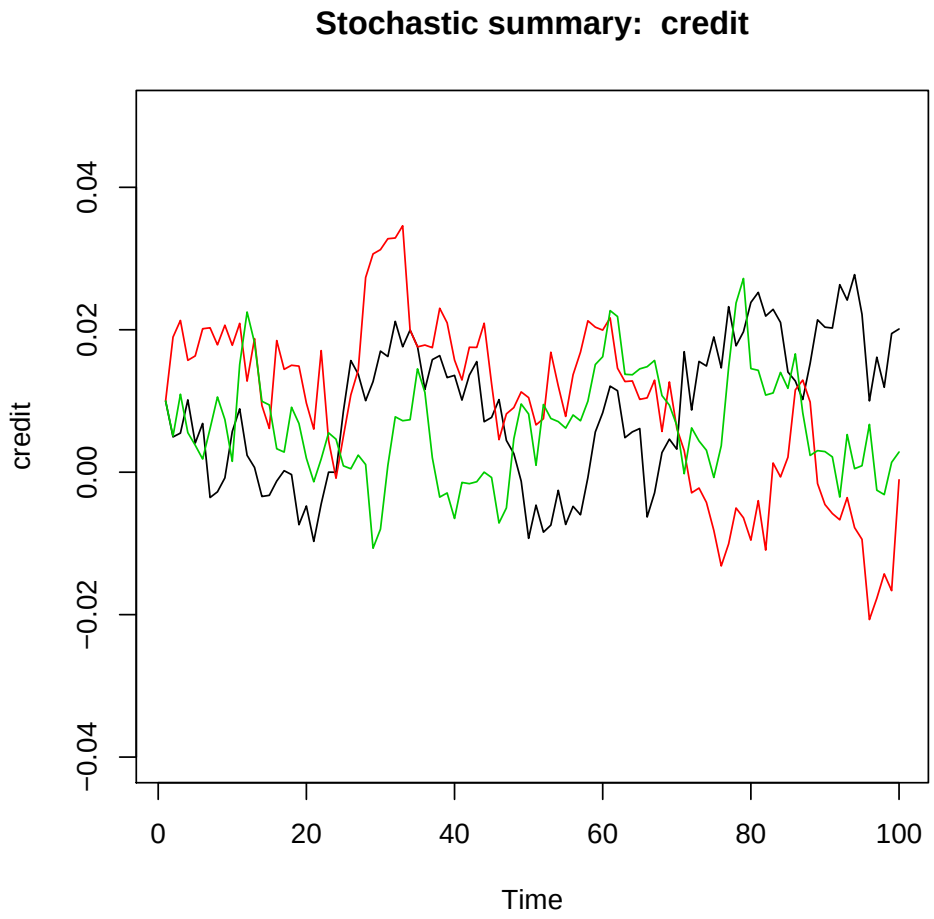

Stochastic summary: term10
term10
0.05
0.10
0.15
0.20
0
20
40
60
80
Time

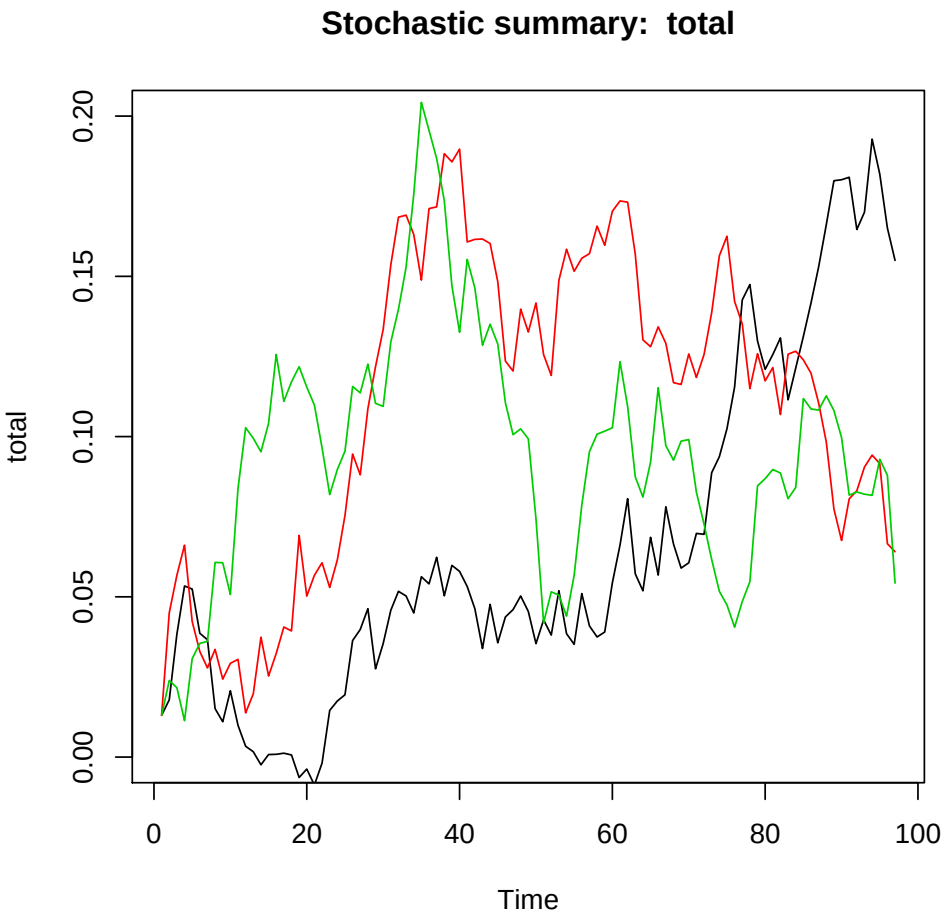

Stochastic summary: term15
term15
0.05
0.10
0.15
0.20
0
20
40
60
80
Time

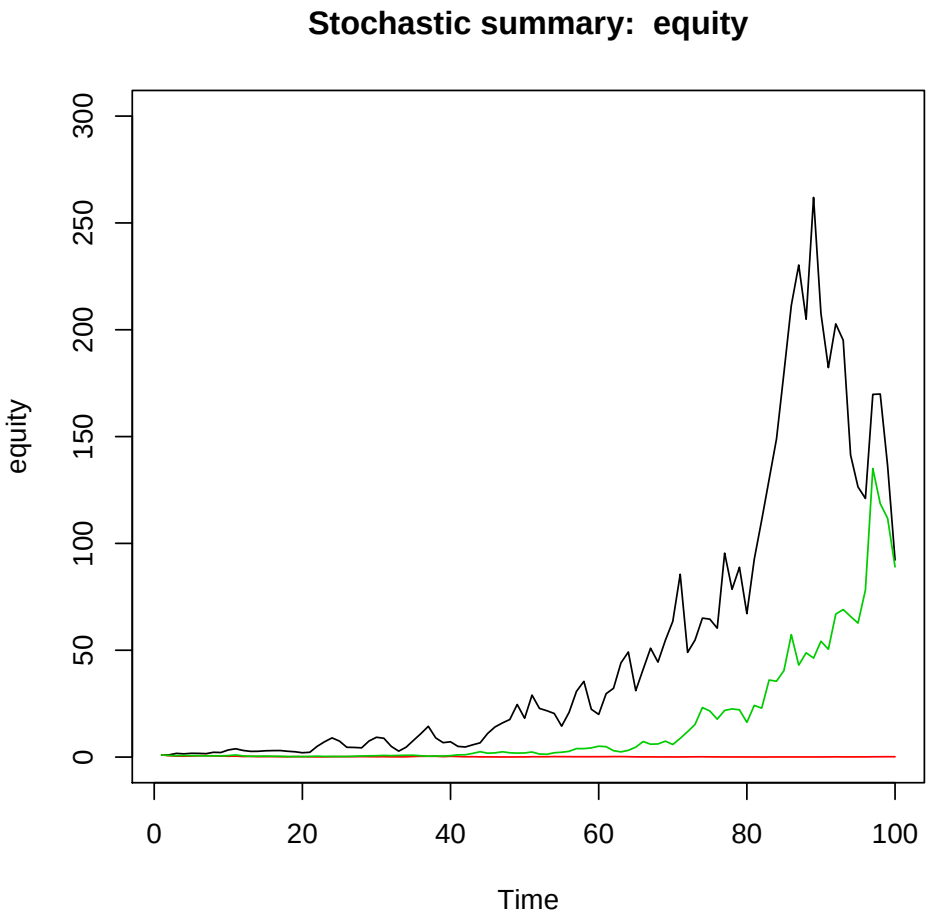

Stochastic summary: term20
term20
0.05
0.10
0.15
0.20
0
20
40
60
80
Time

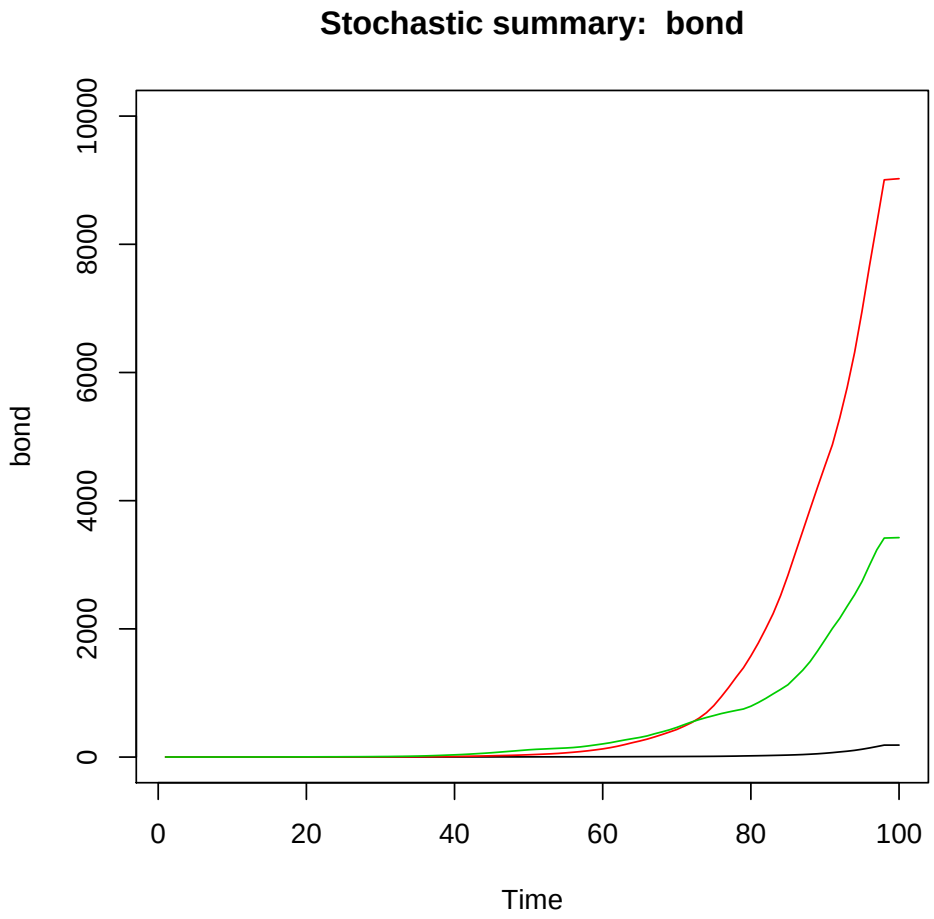

Stochastic summary: term30
term30
0.05
0.10
0.15
0.20
0
10
20
30
40
50
60
70
Time


Stochastic summary: inflation
inflation
0.6
0.4
0.2
0.0
−0.2
0
20
40
60
80
100
Time


Stochastic summary: credit
credit
0.04
0.02
0.00
−0.02
−0.04
0
20
40
60
80
100
Time


Stochastic summary: total
total
0.00
0.05
0.10
0.15
0.20
0
20
40
60
80
100
Time


Stochastic summary: equity
equity
0
50
100
150
200
250
300
0
20
40
60
80
100
Time


Stochastic summary: bond
bond
0
2000
4000
6000
8000
10000
0
20
40
60
80
100
Time