Applied SFC modelling with R

Antoine Godin

03 August 2016

Outline

- pdfetch
- PKSFC package
- SFC visualisation
- Direct Acyclical Graphs
- Sankey diagrams
- Shiny Applications
- Calibration procedures
- Theoretical steady/stationary states
- Empirical calibration
- Bank of England Model

pdfecth

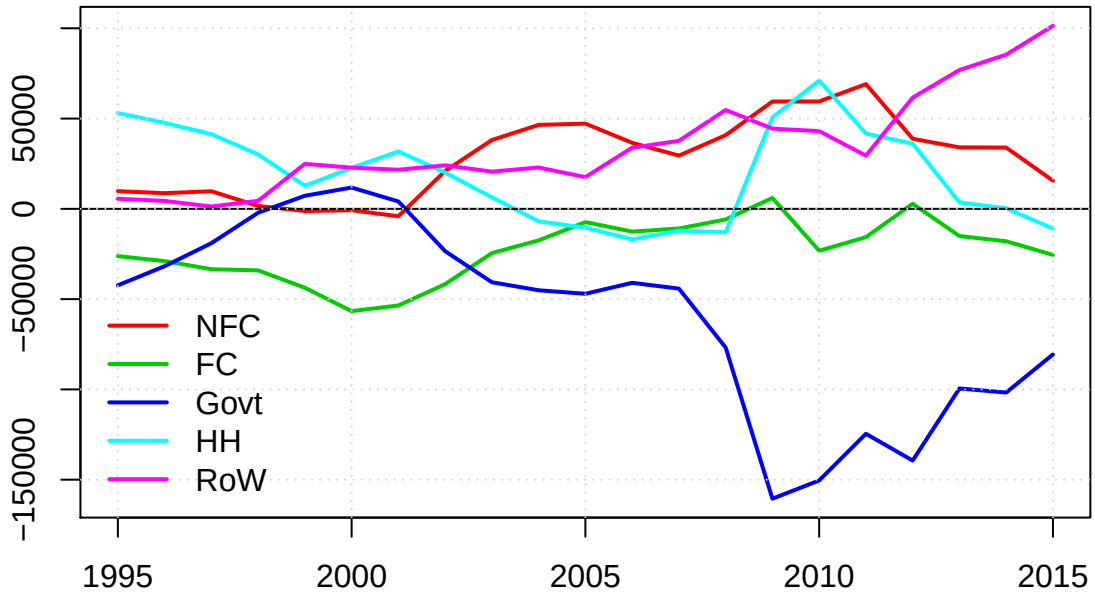
- Fetch Economic and Financial Time Series Data from Public Sources
- Package developped by Abiel Reinhart
- We will be using the `pdfetch_EUROSTAT` function

```
?pdfetch_EUROSTAT
```

Example 1: Net lending per sector, UK

```
# Specifying the name of the flows of interests
names<-c("B9")

# Downloading the data by specifying the various filters
UKdata_raw = pdfetch_EUROSTAT(flowRef = "nasa_10_nf_tr", UNIT="CP_MNAC",NA_ITEM=names, GEO="UK",
      DIRECT="PAID",SECTOR=c("S11","S12","S13","S14_S15","S2"))

# Transforming the obtained data into a data.frame
UKdata<-as.data.frame(UKdata_raw)

# Setting readable names
colnames(UKdata)<-c("NFC","FC","Govt","HH","RoW")

# Matricial plot
matplot(1995:2015,UKdata,lwd=2,type="l",lty=1,ylab="",xlab="",col=2:6)

# Adding the horizontal line
abline(h=0,col=1)

# Adding a grid
grid()
```

```
# Adding a legend
legend("bottomleft",col=c(2:6),lwd=2,lty=1,legend=c("NFC","FC","Govt","HH","RoW"),bty='n')
```



Example 2: Household income statement from 2014

```
# Selecting the flows
names<-c("B5G","D5","D61","D62","D7","D8","B6G","P3","B8G","P5G","D9","NP","B9")

# Obtaining the data
EZdata_raw = pdfetch_EUROSTAT("nasa_10_nf_tr", UNIT="CP_MNAC",NA_ITEM=names, GEO="EU28",
        SECTOR=c("S14_S15"), TIME="2014")

# Transforming the data into a data.frame
EZdata<-as.data.frame(EZdata_raw)

# Automatic procedure to remove the non-interesting bit of the colnames
coln<-colnames(EZdata)
newcoln<-c()
HHdata<-c()

for(i in 1:length(coln)){
  name<-coln[i]
  tname<-strsplit(name,"\\.")[[1]]
```

```
  newname<-paste(tname[3:4],collapse=".")
  # If the column contains only NA, remove it from the dataset
  if(!is.na(EZdata[16,i])){
   newcoln<-c(newcoln,newname)
   HHdata<-c(HHdata,EZdata[16,i])
  }
}

# Creating a new dataset with only values 2014
HHdata<-as.data.frame(t(HHdata))
colnames(HHdata)<-newcoln

# Creating the aggregates
HHdata_1<-as.data.frame(c(HHdata$PAID.B5G,-HHdata$PAID.D5,-HHdata$PAID.D61+HHdata$RECV.D61,
      +HHdata$RECV.D62-HHdata$PAID.D62,-HHdata$PAID.D7+HHdata$RECV.D7,HHdata$PAID.B6G))
colnames(HHdata_1)<-"2014"
rownames(HHdata_1)<-c("Total Income","Taxes","Social Contributions","Social Benefits",
      "Other transfers","Gross Disposable Income")
kable(HHdata_1)
```

	2014
Total Income	9851688
Taxes	-1442130
Social Contributions	-2425244
Social Benefits	2635931
Other transfers	109116
Gross Disposable Income	8729361

```
HHdata_2<-as.data.frame(c(HHdata$PAID.B6G,-HHdata$PAID.P3,-HHdata$PAID.D8+HHdata$RECV.D8,
      HHdata$PAID.B8G))
colnames(HHdata_2)<-"2014"
rownames(HHdata_2)<-c("Gross Disposable Income","Consumption","Adjustments in Pensions",
      "Gross Savings")
kable(HHdata_2)
```

	2014
Gross Disposable Income	8729361
Consumption	-8012603
Adjustments in Pensions	201553
Gross Savings	918310

```
HHdata_3<-as.data.frame(c(HHdata$PAID.B8G,-HHdata$PAID.P5G,-HHdata$PAID.D9+HHdata$RECV.D9,
      -HHdata$PAID.NP,HHdata$PAID.B9))

colnames(HHdata_3)<-"2014"
rownames(HHdata_3)<-c("Gross Savings","Gross Capital Formation","Capital Transfer",
      "Net Non-Produced NF Assets","Net Lending Position")
kable(HHdata_3)
```

	2014
Gross Savings	918310
Gross Capital Formation	-716849
Capital Transfer	14213
Net Non-Produced NF Assets	6360
Net Lending Position	222034

PKSFC package

- Allowing to simulate SFC models in an open source environment}
- Still preliminary
- Only one numerical solver: Gauss-Seidel algorithm (Kinsella and O'Shea 2010)
- github.com/s120/pksfc
- Technical aspect
- R package
- EViews translator
- Read equation files
- Build model from console
- Visualisation/Design tools and helpers

Installing the dependent libraries and the package

You need to install all the required libraries This is for traditional libraries

```
install.packages("expm")
install.packages("igraph")
```

For non-conventional libraries, such as the one need to visualize Direct Acyclical Graphs (DAG), you need to do the following

```
source("http://bioconductor.org/biocLite.R")
biocLite("Rgraphviz")
```

Finally you can then download the PKSFC package from github and install it locally

```
install.packages("path/PKSFC_1.5.tar.gz", repos = NULL, type="source")
```

Testing

Now we're ready to load the package:

```
library(PKSFC)
```

```
## Loading required package: expm
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'expm'
```

```
## The following object is masked from 'package:Matrix':
##
##     expm

## Loading required package: igraph

##
## Attaching package: 'igraph'

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union
```

1. Load SIM (download SIM.txt from Github)

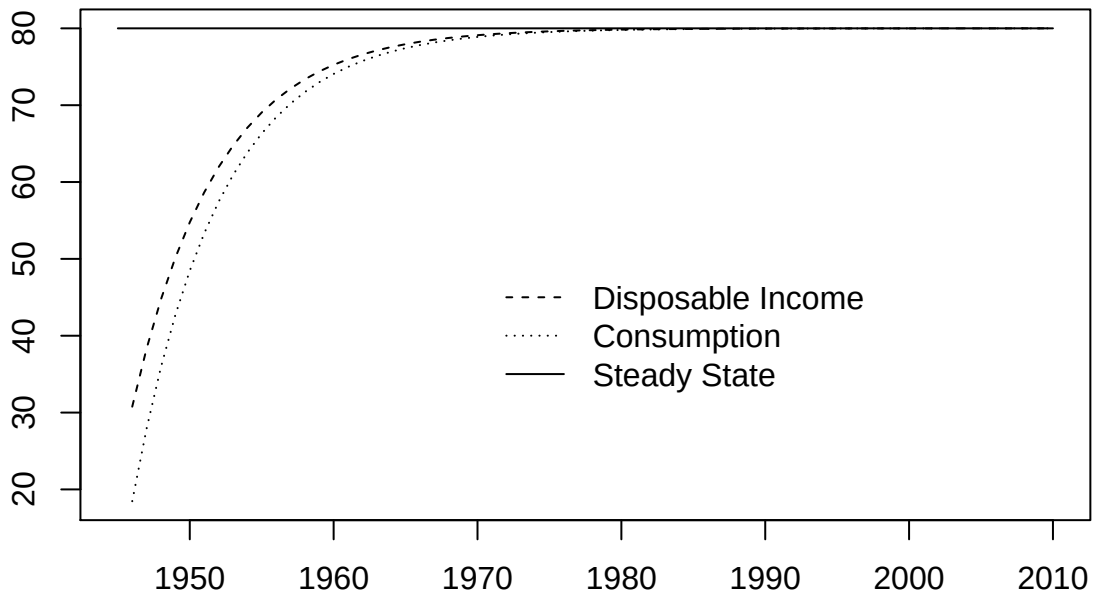
```
sim<-sfc.model("SIM.txt",modelName="SIMplest model from chapter 3 of Godley and Lavoie (2007)")
```

2. Simulate the model

```
datasim<-simulate(sim)
```

3. replicate figure 3.2 of page 73.

```
plot(sim$time,datasim$baseline[,"Yd"],type="l", xlab="", ylab="", lty=2,
     ylim=range(datasim$baseline[,c("Yd","C")],na.rm=T))
lines(sim$time,datasim$baseline[,"C"],lty=3)
lines(sim$time,vector(length=length(sim$time))+datasim$baseline["2010","C"])
legend(x=1970,y=50,legend=c("Disposable Income","Consumption","Steady State"),lty=c(2,3,1),bty="n")
```



How does it work?

- The package parses a text file containing the equations
- It generates an internal representation of the model
- It checks the internal consistency of the model, the calibration
- Allows to simulate the model using a linear solver, the Gauss-Seidel Algorithm

Source code of SIM

```
#1. EQUATIONS
C = C
G = G
T = T
N = N
Yd = W*N - T
T = theta*W*N
C = alpha1*Yd + alpha2*H_h(-1)
H = H(-1) + G - T
H_h = H_h(-1) + Yd - C
Y = C + G
N = Y/W
#2. PARAMETERS
alpha1=0.6
```

```
alpha2=0.4
theta=0.2
#EXOGENOUS
G=20
W=1
#INITIAL VALUES
H=0
H_h=0
#3. Timeline
timeline 1945 2010
```

A few important points regarding the model source code:

- The first line of the code should be a comment line (starting with `#`)
- The file should not contain any empty lines.
- You should avoid naming your variables with reserved names in R such as 'in' or 'max'.
- There should be only one equation per line.
- There should be only one variable on the left hand side of the equation.
- You can use R functions such as min, max, or logical operators such as > or <=. In the case the logical operatur returns true, the numeric value will be one. Thus `(100>10)` will return `1`.
- The lag operator is represented by `(-x)` where `x` is the lag.
- You can add as many comments, using the `#` character at the begining of the line. Each comment exactly above an equation will be considered as the description of the equation and will be stored in the internal representation of the sfc model object.

Internal representation

```
print(sim)
```

```
## $name
## [1] "SIMplest model from chapter 3 of Godley and Lavoie (2007)"
##
## $simulated
## [1] FALSE
##
## $variables
##       name     initial value description
##  [1,] "Yd"     NA            "1. EQUATIONS"
##  [2,] "T"      NA            ""
##  [3,] "C"      NA            ""
##  [4,] "H"      "0"           ""
##  [5,] "H_h"    "0"           ""
##  [6,] "Y"      NA            ""
##  [7,] "N"      NA            ""
##  [8,] "alpha1" "0.6"         "2. PARAMETERS"
##  [9,] "alpha2" "0.4"         ""
## [10,] "theta"  "0.2"         ""
## [11,] "G"      "20"          "EXOGENOUS"
## [12,] "W"      "1"           ""
##
```

```
## $endogenous
##      name  initial value lag description
## [1,] "Yd"  NA            "0" "1. EQUATIONS"
## [2,] "T"   NA            "0" ""
## [3,] "C"   NA            "0" ""
## [4,] "H"   "0"           "1" ""
## [5,] "H_h" "0"           "1" ""
## [6,] "Y"   NA            "0" ""
## [7,] "N"   NA            "0" ""
##
## $equations
##      endogenous value equation                 description
## [1,] "Yd"             "W*N-T"                  "1. EQUATIONS"
## [2,] "T"              "theta*W*N"              ""
## [3,] "C"              "alpha1*Yd+alpha2*H_h_1" ""
## [4,] "H"              "H_1+G-T"                ""
## [5,] "H_h"            "H_h_1+Yd-C"             ""
## [6,] "Y"              "C+G"                    ""
## [7,] "N"              "Y/W"                    ""
##
## $time
##  [1] 1945 1946 1947 1948 1949 1950 1951 1952 1953 1954 1955 1956 1957 1958
## [15] 1959 1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972
## [29] 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986
## [43] 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000
## [57] 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
##
## $matrix
##     Yd T C H H_h Y N
## Yd   0 1 0 0   0 0 1
## T    0 0 0 0   0 0 1
## C    1 0 0 0   0 0 0
## H    0 1 0 0   0 0 0
## H_h  1 0 1 0   0 0 0
## Y    0 0 1 0   0 0 0
## N    0 0 0 0   0 1 0
##
## $blocks
## $blocks[[1]]
## [1] 2 3 4 6 7 1 5
##
##
## attr(,"class")
## [1] "sfc"
```

Output data structure

- Output is a list of matrix where each element of the list are a scenario
- baseline
- scenario_i

```
kable(datasim$baseline)
```

	Yd	T	C	H	H_h	Y	N	alpha1	alpha2	theta	G	W
1945	NA	NA	NA	0.00000	0.00000	NA	NA	0.6	0.4	0.2	20	1
1946	30.76923	7.692308	18.46154	12.30769	12.30769	38.46154	38.46154	0.6	0.4	0.2	20	1
1947	38.34320	9.585799	27.92899	22.72189	22.72189	47.92899	47.92899	0.6	0.4	0.2	20	1
1948	44.75193	11.187984	35.93992	31.53391	31.53391	55.93992	55.93992	0.6	0.4	0.2	20	1
1949	50.17471	12.543678	42.71839	38.99023	38.99023	62.71839	62.71839	0.6	0.4	0.2	20	1
1950	54.76322	13.690805	48.45402	45.29943	45.29943	68.45402	68.45402	0.6	0.4	0.2	20	1
1951	58.64580	14.661450	53.30725	50.63798	50.63798	73.30725	73.30725	0.6	0.4	0.2	20	1
1952	61.93106	15.482766	57.41383	55.15521	55.15521	77.41383	77.41383	0.6	0.4	0.2	20	1
1953	64.71090	16.177725	60.88862	58.97749	58.97749	80.88862	80.88862	0.6	0.4	0.2	20	1
1954	67.06307	16.765767	63.82884	62.21172	62.21172	83.82884	83.82884	0.6	0.4	0.2	20	1
1955	69.05337	17.263341	66.31671	64.94838	64.94838	86.31671	86.31671	0.6	0.4	0.2	20	1
1956	70.73746	17.684366	68.42183	67.26401	67.26401	88.42183	88.42183	0.6	0.4	0.2	20	1
1957	72.16247	18.040617	70.20309	69.22339	69.22339	90.20309	90.20309	0.6	0.4	0.2	20	1
1958	73.36824	18.342061	71.71030	70.88133	70.88133	91.71030	91.71030	0.6	0.4	0.2	20	1
1959	74.38851	18.597128	72.98564	72.28421	72.28421	92.98564	92.98564	0.6	0.4	0.2	20	1
1960	75.25182	18.812955	74.06477	73.47125	73.47125	94.06477	94.06477	0.6	0.4	0.2	20	1
1961	75.98231	18.995577	74.97789	74.47567	74.47567	94.97789	94.97789	0.6	0.4	0.2	20	1
1962	76.60041	19.150104	75.75052	75.32557	75.32557	95.75052	95.75052	0.6	0.4	0.2	20	1
1963	77.12343	19.280857	76.40428	76.04471	76.04471	96.40428	96.40428	0.6	0.4	0.2	20	1
1964	77.56598	19.391494	76.95747	76.65322	76.65322	96.95747	96.95747	0.6	0.4	0.2	20	1
1965	77.94044	19.485111	77.42555	77.16811	77.16811	97.42555	97.42555	0.6	0.4	0.2	20	1
1966	78.25730	19.564324	77.82162	77.60378	77.60378	97.82162	97.82162	0.6	0.4	0.2	20	1
1967	78.52541	19.631351	78.15676	77.97243	77.97243	98.15676	98.15676	0.6	0.4	0.2	20	1
1968	78.75227	19.688067	78.44033	78.28437	78.28437	98.44033	98.44033	0.6	0.4	0.2	20	1
1969	78.94423	19.736056	78.68028	78.54831	78.54831	98.68028	98.68028	0.6	0.4	0.2	20	1
1970	79.10665	19.776663	78.88332	78.77165	78.77165	98.88332	98.88332	0.6	0.4	0.2	20	1
1971	79.24409	19.811023	79.05511	78.96062	78.96062	99.05511	99.05511	0.6	0.4	0.2	20	1
1972	79.36038	19.840096	79.20048	79.12053	79.12053	99.20048	99.20048	0.6	0.4	0.2	20	1
1973	79.45879	19.864697	79.32348	79.25583	79.25583	99.32348	99.32348	0.6	0.4	0.2	20	1
1974	79.54205	19.885513	79.42756	79.37032	79.37032	99.42756	99.42756	0.6	0.4	0.2	20	1
1975	79.61250	19.903126	79.51563	79.46719	79.46719	99.51563	99.51563	0.6	0.4	0.2	20	1
1976	79.67212	19.918030	79.59015	79.54916	79.54916	99.59015	99.59015	0.6	0.4	0.2	20	1
1977	79.72256	19.930640	79.65320	79.61852	79.61852	99.65320	99.65320	0.6	0.4	0.2	20	1
1978	79.76524	19.941311	79.70656	79.67721	79.67721	99.70656	99.70656	0.6	0.4	0.2	20	1
1979	79.80136	19.950340	79.75170	79.72687	79.72687	99.75170	99.75170	0.6	0.4	0.2	20	1
1980	79.83192	19.957980	79.78990	79.76889	79.76889	99.78990	99.78990	0.6	0.4	0.2	20	1
1981	79.85778	19.964445	79.82222	79.80445	79.80445	99.82222	99.82222	0.6	0.4	0.2	20	1
1982	79.87966	19.969915	79.84957	79.83453	79.83453	99.84957	99.84957	0.6	0.4	0.2	20	1
1983	79.89817	19.974543	79.87272	79.85999	79.85999	99.87272	99.87272	0.6	0.4	0.2	20	1
1984	79.91384	19.978460	79.89230	79.88153	79.88153	99.89230	99.89230	0.6	0.4	0.2	20	1
1985	79.92709	19.981774	79.90887	79.89975	79.89975	99.90887	99.90887	0.6	0.4	0.2	20	1
1986	79.93831	19.984578	79.92289	79.91518	79.91518	99.92289	99.92289	0.6	0.4	0.2	20	1
1987	79.94780	19.986950	79.93475	79.92823	79.92823	99.93475	99.93475	0.6	0.4	0.2	20	1
1988	79.95583	19.988958	79.94479	79.93927	79.93927	99.94479	99.94479	0.6	0.4	0.2	20	1
1989	79.96263	19.990657	79.95328	79.94861	79.94861	99.95328	99.95328	0.6	0.4	0.2	20	1
1990	79.96838	19.992094	79.96047	79.95652	79.95652	99.96047	99.96047	0.6	0.4	0.2	20	1
1991	79.97324	19.993310	79.96655	79.96321	79.96321	99.96655	99.96655	0.6	0.4	0.2	20	1
1992	79.97736	19.994340	79.97170	79.96887	79.96887	99.97170	99.97170	0.6	0.4	0.2	20	1
1993	79.98084	19.995210	79.97605	79.97366	79.97366	99.97605	99.97605	0.6	0.4	0.2	20	1
1994	79.98379	19.995947	79.97974	79.97771	79.97771	99.97974	99.97974	0.6	0.4	0.2	20	1
1995	79.98628	19.996571	79.98285	79.98114	79.98114	99.98285	99.98285	0.6	0.4	0.2	20	1
1996	79.98839	19.997098	79.98549	79.98404	79.98404	99.98549	99.98549	0.6	0.4	0.2	20	1

	Yd	T	C	H	H_h	Y	N	alpha1	alpha2	theta	G	W
1997	79.99018	19.997545	79.98772	79.98650	79.98650	99.98772	99.98772	0.6	0.4	0.2	20	1
1998	79.99169	19.997923	79.98961	79.98857	79.98857	99.98961	99.98961	0.6	0.4	0.2	20	1
1999	79.99297	19.998242	79.99121	79.99033	79.99033	99.99121	99.99121	0.6	0.4	0.2	20	1
2000	79.99405	19.998513	79.99256	79.99182	79.99182	99.99256	99.99256	0.6	0.4	0.2	20	1
2001	79.99497	19.998741	79.99371	79.99308	79.99308	99.99371	99.99371	0.6	0.4	0.2	20	1
2002	79.99574	19.998935	79.99468	79.99414	79.99414	99.99468	99.99468	0.6	0.4	0.2	20	1
2003	79.99640	19.999099	79.99549	79.99504	79.99504	99.99549	99.99549	0.6	0.4	0.2	20	1
2004	79.99695	19.999237	79.99619	79.99581	79.99581	99.99619	99.99619	0.6	0.4	0.2	20	1
2005	79.99742	19.999355	79.99677	79.99645	79.99645	99.99677	99.99677	0.6	0.4	0.2	20	1
2006	79.99782	19.999454	79.99727	79.99700	79.99700	99.99727	99.99727	0.6	0.4	0.2	20	1
2007	79.99815	19.999538	79.99769	79.99746	79.99746	99.99769	99.99769	0.6	0.4	0.2	20	1
2008	79.99844	19.999609	79.99805	79.99785	79.99785	99.99805	99.99805	0.6	0.4	0.2	20	1
2009	79.99868	19.999669	79.99835	79.99818	79.99818	99.99835	99.99835	0.6	0.4	0.2	20	1
2010	79.99888	19.999720	79.99860	79.99846	79.99846	99.99860	99.99860	0.6	0.4	0.2	20	1

The Gauss Seidel Algorithm

- Principle: Solving $Ax = b$, $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$ via an iterative algorithm, where each iteration can be represented by $Lx^{k+1} = b - Ux^k$, $A = L + U$. Where L is lower triangular and U is upper triangular.
- Pseudo-code:

1. Select initial values x^0
2. While $k < maxIter$ & $\delta < tolValue$
   a. For each $i = 1, ..., n$:
   $$x_i^{k+1} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{k+1} - \sum_{j=i+1}^{n} a_{ij} x_j^k \right)$$
   b. Compute δ:
   $$\delta = \frac{x^{k+1} - x^k}{x^k}$$

System of (in)dependent equations

See Fenell et. al (2016)

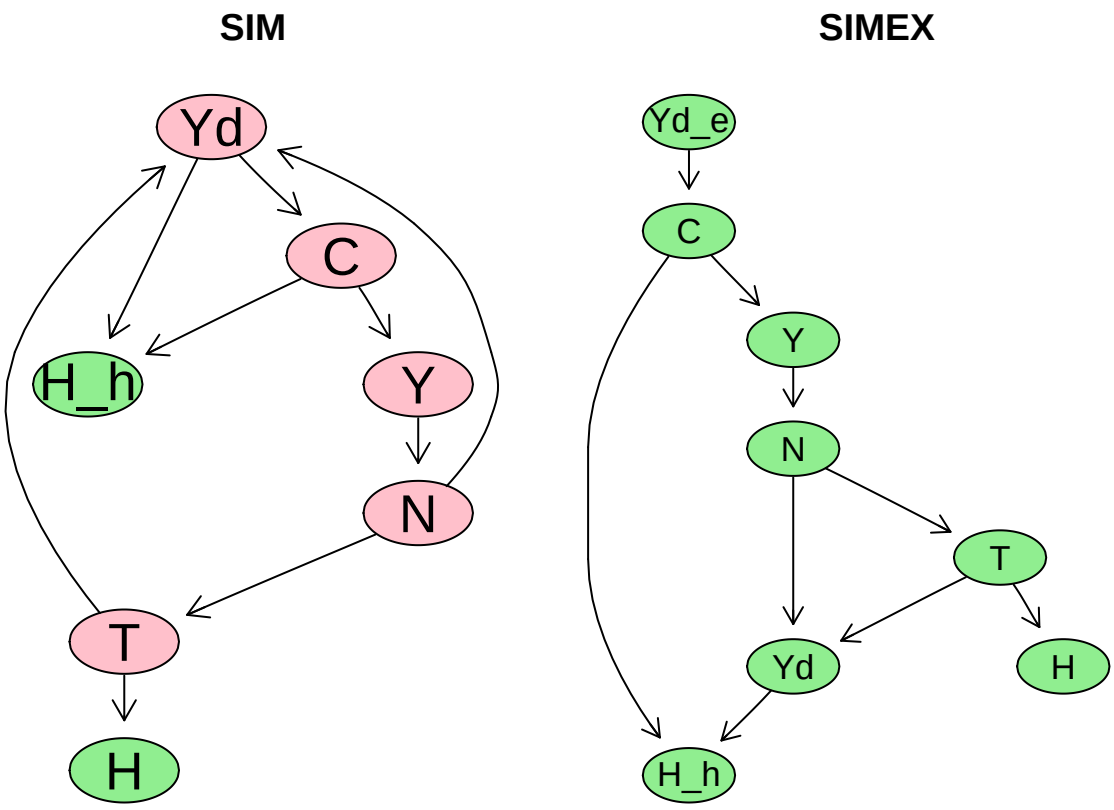
- The Gauss-Seidel has to be used only in the case of system of dependent equations
- In other case, we only need to find order in which each variable is computed and simply compute the new value in each period
- This order is the "logical causal order" of the model and can be visualised using Direct (A)Cyclical Graphs

Direct Acyclic graphs

```
simex<-sfc.model("SIMEX.txt",modelName="SIMplest model with expectation")
layout(matrix(c(1,2),1,2))
plot.dag(sim,main="SIM" )
```

```
## Loading required package: Rgraphviz

## Loading required package: graph

## Loading required package: BiocGenerics

## Loading required package: parallel

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:igraph':
##
##     normalize, union

## The following objects are masked from 'package:stats':
##
##     IQR, mad, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, cbind, colnames,
##     do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, lengths, Map, mapply,
##     match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##     Position, rank, rbind, Reduce, rownames, sapply, setdiff,
##     sort, table, tapply, union, unique, unsplit

##
## Attaching package: 'graph'

## The following objects are masked from 'package:igraph':
##
##     degree, edges, intersection

## Loading required package: grid
```

```
plot.dag(simex,main="SIMEX" )
```

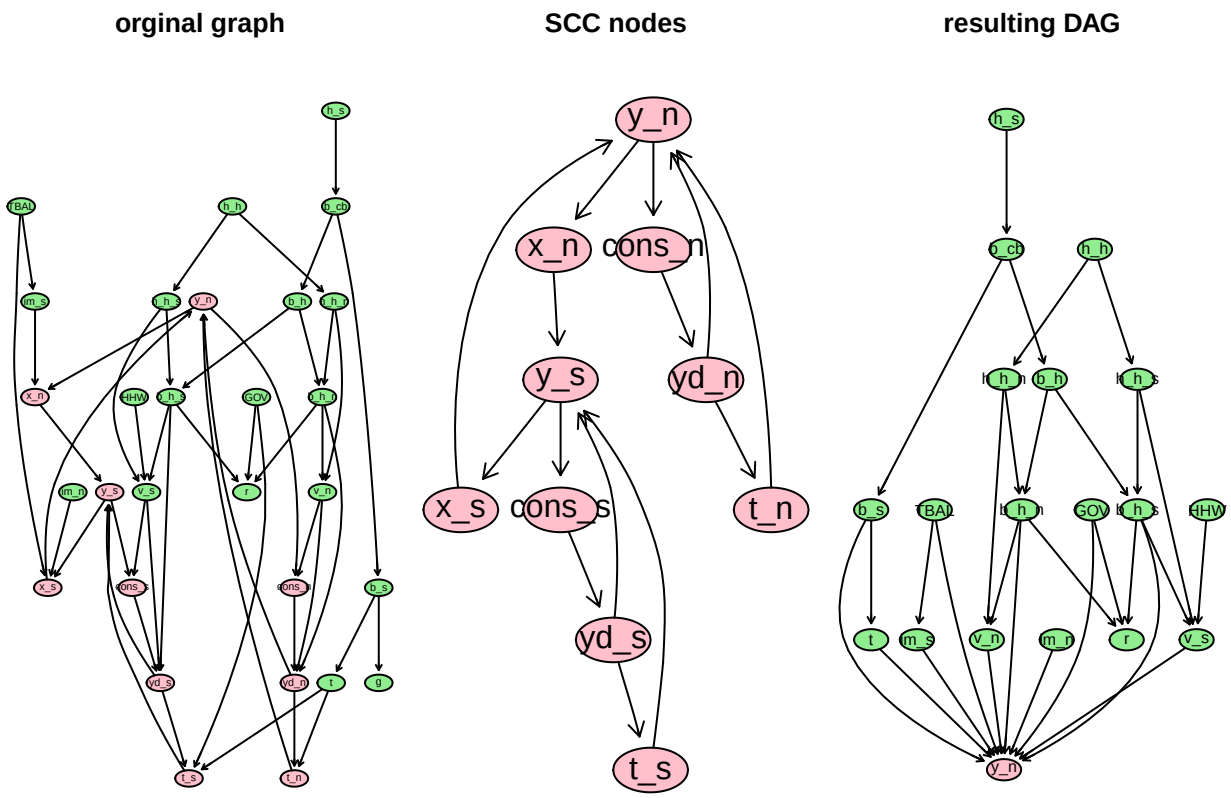


- Asside from the mathematical implication that a system of equation represent, it also has an economic meaning:
- the economy represented by SIM will adjust in one period to any shock applied to government spending.
- for SIMEX it is not the case because consumption depend on expected disposable income which is equal to previous period disposable income.
- in this case, the economy represented by SIMEX will adjust slowly to a shock applied to government spending, via the stocks (and particularly the buffer stock)

In the case of a more complex model - Chapter 6

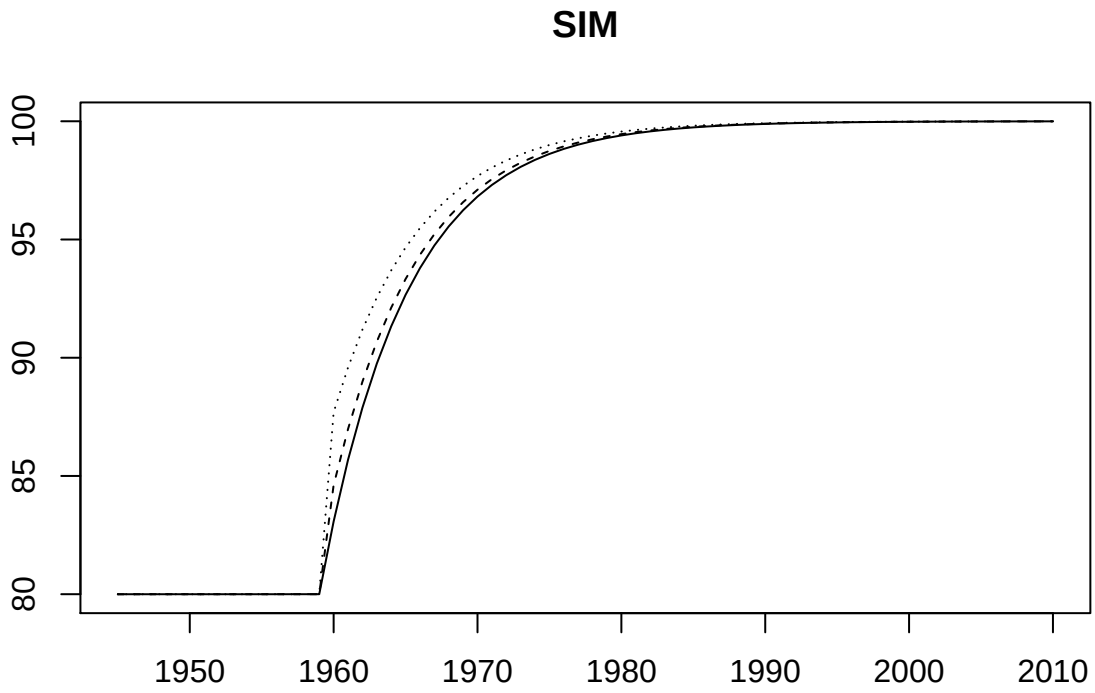
- We can generate the plots that allow us to delve into the actual structure of the system.
- Nodes that do not form a cycle are green while nodes that form a cycle in the system are pink.
- The Gauss-Seidel needs to be applied only for the cycles

```
ch6 <- sfc.model("ch6.txt",modelName="Chapter6_openmodel")
graphs = generate.DAG.collapse( adjacency = ch6$matrix )
par(mfrow = c(1,3))
# first plot the orgianl grpah
plot_graph_hierarchy( graphs$orginal_graph, main = "orginal graph" )
# plot hte nodes that form the strongly connected compoent
plot_graph_hierarchy( graphs$SCC_graph, main = "SCC nodes" )
# plot the result ing DAG when we take the condensation of the graph
plot_graph_hierarchy( graphs$DAG, main = "resulting DAG" )
```

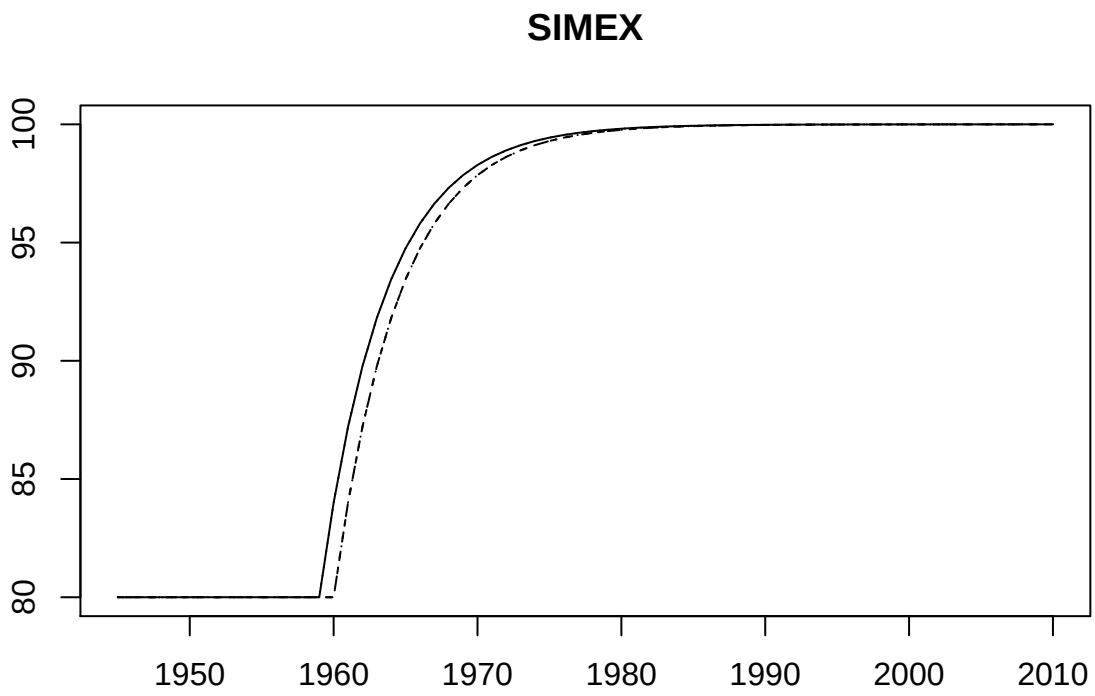


Systems of dependent vs independent equations

```
datasimex<-simulate(simex)
init = datasimex$baseline[66,]
simex<-sfc.addScenario(simex,"G",25,1960,2010,init)
datasimex<-simulate(simex)
datasim<-simulate(sim)
init = datasim$baseline[66,]
sim<-sfc.addScenario(sim,"G",25,1960,2010,init)
datasim<-simulate(sim)
plot(sim$time,datasim$scenario_1[,"H"],type="l",xlab="",ylab="",main="SIM")
lines(sim$time,datasim$scenario_1[,"C"],lty=2)
lines(sim$time,datasim$scenario_1[,"Yd"],lty=3)
legend(x=1944,y=130,legend=c("Wealth","Consumption","Disposable Income"),
       lty=c(1,2,3),bty="n")
```




```
plot(simex$time,datasimex$scenario_1[,"H"],type="l",xlab="",ylab="",main="SIMEX")
lines(simex$time,datasimex$scenario_1[,"C"],lty=2)
lines(simex$time,datasimex$scenario_1[,"Yd"],lty=3)
lines(simex$time,datasimex$scenario_1[,"Yd_e"],lty=4)
legend(x=1944,y=130,legend=c("Wealth","Consumption","Disposable Income",
                             "Expecetd Disposable Income"),lty=c(1,2,3,4),bty="n")
```



Computational implications

Let's see how much time it takes to run sim:

```
ptm <- proc.time()
data1<-simulate(sim)
print(paste("Elapsed time is ",proc.time()[3]-ptm[3],"seconds"))
```

```
## [1] "Elapsed time is  4.9 seconds"
```

Now lets play with some of the parameters of the simulate function:

1. tolValue

```
ptm <- proc.time()
data2<-simulate(sim,tolValue = 1e-3)
print(paste("Elapsed time is ",proc.time()[3]-ptm[3],"seconds"))
```

```
## [1] "Elapsed time is  0.27 seconds"
```

2. maxIter

```
ptm <- proc.time()
data3<-simulate(sim, maxIter=10)
print(paste("Elapsed time is ",proc.time()[3]-ptm[3],"seconds"))
```

```
## [1] "Elapsed time is  0.280000000000001 seconds"
```

Observing the results of the three simulations

```
kable(round(t(data1$baseline[c(1,2,20,40,66),c("G","Y","T","Yd","C","H","H_h")]),digits=3))
```

	1945	1946	1964	1984	2010
G	20	20.000	20.000	20.000	20.000
Y	NA	38.462	96.957	99.892	99.999
T	NA	7.692	19.391	19.978	20.000
Yd	NA	30.769	77.566	79.914	79.999
C	NA	18.462	76.957	79.892	79.999
H	0	12.308	76.653	79.882	79.998
H_h	0	12.308	76.653	79.882	79.998

```
kable(round(t(data2$baseline[c(1,2,20,40,66),c("G","Y","T","Yd","C","H","H_h")]),digits=3))
```

	1945	1946	1964	1984	2010
G	20	20.000	20.000	20.000	20.000
Y	NA	38.464	96.874	99.577	99.911
T	NA	7.693	19.369	19.909	19.980
Yd	NA	30.772	77.478	79.644	79.920
C	NA	18.460	76.870	79.566	79.917
H	0	12.308	76.745	81.506	82.795
H_h	0	12.303	76.569	79.535	79.922

```
kable(round(t(data3$baseline[c(1,2,20,40,66),c("G","Y","T","Yd","C","H","H_h")]),digits=3))
```

	1945	1946	1964	1984	2010
G	20	20.000	20.000	20.000	20.000
Y	NA	34.006	94.965	99.672	99.991
T	NA	5.978	18.970	19.933	19.998
Yd	NA	23.982	75.899	79.733	79.992
C	NA	14.136	74.878	79.666	79.990
H	0	14.094	91.289	97.979	98.432
H_h	0	9.554	74.332	79.631	79.989

Block Gauss-Seidel

The order of equations matters, if first compute variables that do not depend on current period, this speeds the process. Define blocks of equation independent from the others.

```
print(simex)
```

```
## $name
## [1] "SIMplest model with expectation"
##
## $simulated
## [1] FALSE
##
## $variables
##       name     initial value description
##  [1,] "Yd"     "0"           "1. EQUATIONS"
##  [2,] "T"      NA            ""
##  [3,] "C"      NA            ""
##  [4,] "H"      "0"           ""
##  [5,] "H_h"    "0"           ""
##  [6,] "Y"      NA            ""
##  [7,] "Yd_e"   NA            ""
##  [8,] "N"      NA            ""
##  [9,] "alpha1" "0.6"         "2. PARAMETERS"
## [10,] "alpha2" "0.4"         ""
## [11,] "theta"  "0.2"         ""
## [12,] "G"      "20"          "EXOGENOUS"
## [13,] "W"      "1"           ""
##
## $endogenous
##      name   initial value lag description
## [1,] "Yd"   "0"           "1" "1. EQUATIONS"
## [2,] "T"    NA            "0" ""
## [3,] "C"    NA            "0" ""
## [4,] "H"    "0"           "1" ""
## [5,] "H_h"  "0"           "1" ""
## [6,] "Y"    NA            "0" ""
## [7,] "Yd_e" NA            "0" ""
## [8,] "N"    NA            "0" ""
##
## $equations
##      endogenous value equation                   description
## [1,] "Yd"              "W*N-T"                    "1. EQUATIONS"
## [2,] "T"               "theta*W*N"                ""
## [3,] "C"               "alpha1*Yd_e+alpha2*H_h_1" ""
## [4,] "H"               "H_1+G-T"                  ""
## [5,] "H_h"             "H_h_1+Yd-C"               ""
## [6,] "Y"               "C+G"                      ""
## [7,] "Yd_e"            "Yd_1"                     ""
## [8,] "N"               "Y/W"                      ""
##
## $time
##  [1] 1945 1946 1947 1948 1949 1950 1951 1952 1953 1954 1955 1956 1957 1958
## [15] 1959 1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972
```

```
## [29] 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986
## [43] 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000
## [57] 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
##
## $matrix
##      Yd T C H H_h Y Yd_e N
## Yd    0 1 0 0   0 0    0 1
## T     0 0 0 0   0 0    0 1
## C     0 0 0 0   0 0    1 0
## H     0 1 0 0   0 0    0 0
## H_h   1 0 1 0   0 0    0 0
## Y     0 0 1 0   0 0    0 0
## Yd_e  0 0 0 0   0 0    0 0
## N     0 0 0 0   0 1    0 0
##
## $blocks
## $blocks[[1]]
## [1] 7
##
## $blocks[[2]]
## [1] 3
##
## $blocks[[3]]
## [1] 6
##
## $blocks[[4]]
## [1] 8
##
## $blocks[[5]]
## [1] 2
##
## $blocks[[6]]
## [1] 1 4
##
## $blocks[[7]]
## [1] 5
##
##
## $scenarios
## $scenarios[[1]]
## $scenarios[[1]]$var
## [1] "G"
##
## $scenarios[[1]]$value
## [1] 25
##
## $scenarios[[1]]$init
## [1] 1959
##
## $scenarios[[1]]$end
## [1] 2010
##
## $scenarios[[1]]$start
##           Yd            T            C            H          H_h
```

```
##     79.99996     19.99999     79.99995     79.99996     79.99996
##            Y         Yd_e            N       alpha1       alpha2
##     99.99995     79.99995     99.99995      0.60000      0.40000
##        theta            G            W iter block 1 iter block 2
##      0.20000     20.00000      1.00000      2.00000      2.00000
## iter block 3 iter block 4 iter block 5 iter block 6 iter block 7
##      2.00000      2.00000      2.00000      2.00000      2.00000
##
##
##
## attr(,"class")
## [1] "sfc"
```

Simulation of SIMEX

```
ptm <- proc.time()
dataex<-simulate(simex,tolValue = 1e-10)
print(paste("Elapsed time is ",proc.time()[3]-ptm[3],"seconds"))
```

```
## [1] "Elapsed time is  0.359999999999999 seconds"
```

Results for SIMEX

```
kable(round(t(dataex$baseline[c(1,2,20,40,66),c("G","Y","T","Yd","Yd_e","C","H","H_h")]),digits=3))
```

	1945	1946	1964	1984	2010
G	20	20	20.000	20.000	20
Y	NA	20	98.559	99.983	100
T	NA	4	19.712	19.997	20
Yd	0	16	78.847	79.987	80
Yd_e	NA	0	78.559	79.983	80
C	NA	0	78.559	79.983	80
H	0	16	78.847	79.987	80
H_h	0	16	78.847	79.987	80

Output data structure

- Output is a list of matrix where each element of the list are a scenario
- baseline
- scenario_i
- In the result matrix, there is a column indicating the number of iteration in the Gauss-Seidel algorithm per block of equations per period

```
kable(dataex$baseline)
```

	Yd	T	C	H	H_h	Y	Yd_e	N	alpha1	alpha2	theta
1945	0.00000	NA	NA	0.00000	0.00000	NA	NA	NA	0.6	0.4	0.2
1946	16.00000	4.00000	0.00000	16.00000	16.00000	20.00000	0.00000	20.00000	0.6	0.4	0.2
1947	28.80000	7.20000	16.00000	28.80000	28.80000	36.00000	16.00000	36.00000	0.6	0.4	0.2
1948	39.04000	9.76000	28.80000	39.04000	39.04000	48.80000	28.80000	48.80000	0.6	0.4	0.2
1949	47.23200	11.80800	39.04000	47.23200	47.23200	59.04000	39.04000	59.04000	0.6	0.4	0.2
1950	53.78560	13.44640	47.23200	53.78560	53.78560	67.23200	47.23200	67.23200	0.6	0.4	0.2
1951	59.02848	14.75712	53.78560	59.02848	59.02848	73.78560	53.78560	73.78560	0.6	0.4	0.2
1952	63.22278	15.80570	59.02848	63.22278	63.22278	79.02848	59.02848	79.02848	0.6	0.4	0.2
1953	66.57823	16.64456	63.22278	66.57823	66.57823	83.22278	63.22278	83.22278	0.6	0.4	0.2
1954	69.26258	17.31565	66.57823	69.26258	69.26258	86.57823	66.57823	86.57823	0.6	0.4	0.2
1955	71.41007	17.85252	69.26258	71.41007	71.41007	89.26258	69.26258	89.26258	0.6	0.4	0.2
1956	73.12805	18.28201	71.41007	73.12805	73.12805	91.41007	71.41007	91.41007	0.6	0.4	0.2
1957	74.50244	18.62561	73.12805	74.50244	74.50244	93.12805	73.12805	93.12805	0.6	0.4	0.2
1958	75.60195	18.90049	74.50244	75.60195	75.60195	94.50244	74.50244	94.50244	0.6	0.4	0.2
1959	76.48156	19.12039	75.60195	76.48156	76.48156	95.60195	75.60195	95.60195	0.6	0.4	0.2
1960	77.18525	19.29631	76.48156	77.18525	77.18525	96.48156	76.48156	96.48156	0.6	0.4	0.2
1961	77.74820	19.43705	77.18525	77.74820	77.74820	97.18525	77.18525	97.18525	0.6	0.4	0.2
1962	78.19856	19.54964	77.74820	78.19856	78.19856	97.74820	77.74820	97.74820	0.6	0.4	0.2
1963	78.55885	19.63971	78.19856	78.55885	78.55885	98.19856	78.19856	98.19856	0.6	0.4	0.2
1964	78.84708	19.71177	78.55885	78.84708	78.84708	98.55885	78.55885	98.55885	0.6	0.4	0.2
1965	79.07766	19.76942	78.84708	79.07766	79.07766	98.84708	78.84708	98.84708	0.6	0.4	0.2
1966	79.26213	19.81553	79.07766	79.26213	79.26213	99.07766	79.07766	99.07766	0.6	0.4	0.2
1967	79.40970	19.85243	79.26213	79.40970	79.40970	99.26213	79.26213	99.26213	0.6	0.4	0.2
1968	79.52776	19.88194	79.40970	79.52776	79.52776	99.40970	79.40970	99.40970	0.6	0.4	0.2
1969	79.62221	19.90555	79.52776	79.62221	79.62221	99.52776	79.52776	99.52776	0.6	0.4	0.2
1970	79.69777	19.92444	79.62221	79.69777	79.69777	99.62221	79.62221	99.62221	0.6	0.4	0.2
1971	79.75821	19.93955	79.69777	79.75821	79.75821	99.69777	79.69777	99.69777	0.6	0.4	0.2
1972	79.80657	19.95164	79.75821	79.80657	79.80657	99.75821	79.75821	99.75821	0.6	0.4	0.2
1973	79.84526	19.96131	79.80657	79.84526	79.84526	99.80657	79.80657	99.80657	0.6	0.4	0.2
1974	79.87621	19.96905	79.84526	79.87621	79.87621	99.84526	79.84526	99.84526	0.6	0.4	0.2
1975	79.90096	19.97524	79.87621	79.90096	79.90096	99.87621	79.87621	99.87621	0.6	0.4	0.2
1976	79.92077	19.98019	79.90096	79.92077	79.92077	99.90096	79.90096	99.90096	0.6	0.4	0.2
1977	79.93662	19.98415	79.92077	79.93662	79.93662	99.92077	79.92077	99.92077	0.6	0.4	0.2
1978	79.94929	19.98732	79.93662	79.94929	79.94929	99.93662	79.93662	99.93662	0.6	0.4	0.2
1979	79.95944	19.98986	79.94929	79.95944	79.95944	99.94929	79.94929	99.94929	0.6	0.4	0.2
1980	79.96755	19.99189	79.95944	79.96755	79.96755	99.95944	79.95944	99.95944	0.6	0.4	0.2
1981	79.97404	19.99351	79.96755	79.97404	79.97404	99.96755	79.96755	99.96755	0.6	0.4	0.2
1982	79.97923	19.99481	79.97404	79.97923	79.97923	99.97404	79.97404	99.97404	0.6	0.4	0.2
1983	79.98338	19.99585	79.97923	79.98338	79.98338	99.97923	79.97923	99.97923	0.6	0.4	0.2
1984	79.98671	19.99668	79.98338	79.98671	79.98671	99.98338	79.98338	99.98338	0.6	0.4	0.2
1985	79.98937	19.99734	79.98671	79.98937	79.98937	99.98671	79.98671	99.98671	0.6	0.4	0.2
1986	79.99149	19.99787	79.98937	79.99149	79.99149	99.98937	79.98937	99.98937	0.6	0.4	0.2
1987	79.99319	19.99830	79.99149	79.99319	79.99319	99.99149	79.99149	99.99149	0.6	0.4	0.2
1988	79.99456	19.99864	79.99319	79.99456	79.99456	99.99319	79.99319	99.99319	0.6	0.4	0.2
1989	79.99564	19.99891	79.99456	79.99564	79.99564	99.99456	79.99456	99.99456	0.6	0.4	0.2
1990	79.99652	19.99913	79.99564	79.99652	79.99652	99.99564	79.99564	99.99564	0.6	0.4	0.2
1991	79.99721	19.99930	79.99652	79.99721	79.99721	99.99652	79.99652	99.99652	0.6	0.4	0.2
1992	79.99777	19.99944	79.99721	79.99777	79.99777	99.99721	79.99721	99.99721	0.6	0.4	0.2
1993	79.99822	19.99955	79.99777	79.99822	79.99822	99.99777	79.99777	99.99777	0.6	0.4	0.2
1994	79.99857	19.99964	79.99822	79.99857	79.99857	99.99822	79.99822	99.99822	0.6	0.4	0.2
1995	79.99886	19.99971	79.99857	79.99886	79.99886	99.99857	79.99857	99.99857	0.6	0.4	0.2
1996	79.99909	19.99977	79.99886	79.99909	79.99909	99.99886	79.99886	99.99886	0.6	0.4	0.2

	Yd	T	C	H	H_h	Y	Yd_e	N	alpha1	alpha2	theta
1997	79.99927	19.99982	79.99909	79.99927	79.99927	99.99909	79.99909	99.99909	0.6	0.4	0.2
1998	79.99942	19.99985	79.99927	79.99942	79.99942	99.99927	79.99927	99.99927	0.6	0.4	0.2
1999	79.99953	19.99988	79.99942	79.99953	79.99953	99.99942	79.99942	99.99942	0.6	0.4	0.2
2000	79.99963	19.99991	79.99953	79.99963	79.99963	99.99953	79.99953	99.99953	0.6	0.4	0.2
2001	79.99970	19.99993	79.99963	79.99970	79.99970	99.99963	79.99963	99.99963	0.6	0.4	0.2
2002	79.99976	19.99994	79.99970	79.99976	79.99976	99.99970	79.99970	99.99970	0.6	0.4	0.2
2003	79.99981	19.99995	79.99976	79.99981	79.99981	99.99976	79.99976	99.99976	0.6	0.4	0.2
2004	79.99985	19.99996	79.99981	79.99985	79.99985	99.99981	79.99981	99.99981	0.6	0.4	0.2
2005	79.99988	19.99997	79.99985	79.99988	79.99988	99.99985	79.99985	99.99985	0.6	0.4	0.2
2006	79.99990	19.99998	79.99988	79.99990	79.99990	99.99988	79.99988	99.99988	0.6	0.4	0.2
2007	79.99992	19.99998	79.99990	79.99992	79.99992	99.99990	79.99990	99.99990	0.6	0.4	0.2
2008	79.99994	19.99998	79.99992	79.99994	79.99994	99.99992	79.99992	99.99992	0.6	0.4	0.2
2009	79.99995	19.99999	79.99994	79.99995	79.99995	99.99994	79.99994	99.99994	0.6	0.4	0.2
2010	79.99996	19.99999	79.99995	79.99996	79.99996	99.99995	79.99995	99.99995	0.6	0.4	0.2

Checking the number of iteractions

- For sim, no simulate parameters:

```
kable(round(t(data1$baseline[c(1,2,20,40,66),c("iter block 1")]),digits=3))
```

1945	1946	1964	1984	2010
0	293	218	175	119
- For s	im, tol	Value f	ixed:	

```
kable(round(t(data2$baseline[c(1,2,20,40,66),c("iter block 1")]),digits=3))
```

1945	1946	1964	1984	2010
0	89	15	2	1
- For s	im, max	Iter fi	xed:	

```
kable(round(t(data2$baseline[c(1,2,20,40,66),c("iter block 1")]),digits=3))
```

1945	1946	1964	1984	2010
0	89	15	2	1
- For s	imex, n	o simul	ate par	ameters

```
kable(round(t(dataex$baseline[c(1,2,20,40,66),c("iter block 1","iter block 2","iter block 3","iter bloc
```

	1945	1946	1964	1984	2010
iter block 1	0	2	2	2	2
iter block 2	0	2	2	2	2
iter block 3	0	2	2	2	2

	1945	1946	1964	1984	2010
iter block 4	0	2	2	2	2
iter block 5	0	2	2	2	2
iter block 6	0	2	2	2	2
iter block 7	0	2	2	2	2

Buffer stocks, using model PC

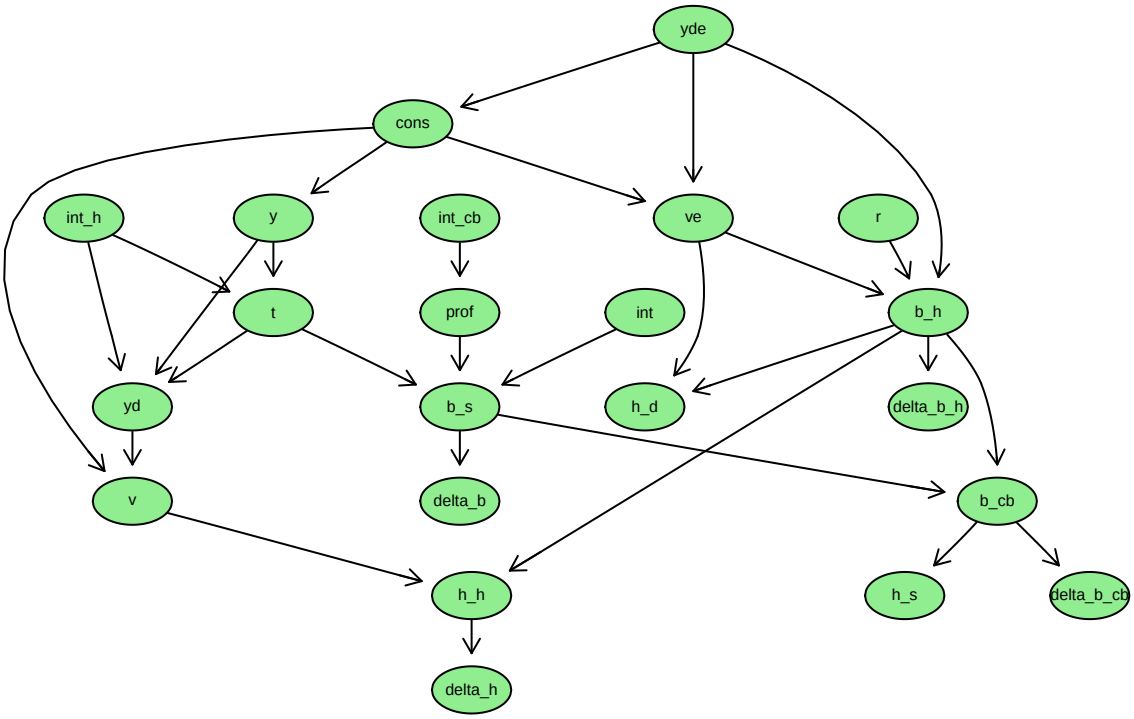
- Observe the buffer role of money, in case of random shocks applied to expected disposable income
- we need to modify slightly model pc and change the equations determining
- consumption,
- demand for bonds and money,
- expectations on income and wealth

```
pc<-sfc.model("PC.txt",modelName="Portfolio Choice Model")
# Changing Existing equations
pcRand<-sfc.editEqus(pc,list(
  list(var="cons",eq="alpha1*yde+alpha2*v(-1)"),
  list(var="b_h",eq="ve*(lambda0 + lambda1*r - lambda2*(yde/ve))")))

# Adding equations
pcRand<-sfc.addEqus(pcRand,list(
  list(var="yde",eq="yd(-1)*(1+rnorm(1,sd=0.1))",desc="Expected disposable income depending on random sh
  list(var="h_d",eq="ve-b_h",desc="Money demand"),
  list(var="ve",eq="v(-1)+yde-cons",desc="Expected disposable income")))

# Adding the initial value of yd (to be used in the expectation function)
pcRand<-sfc.editVar(pcRand,var="yd",init=86.48648)

# Checking if the model is complete
pcRand<-sfc.check(pcRand)
```

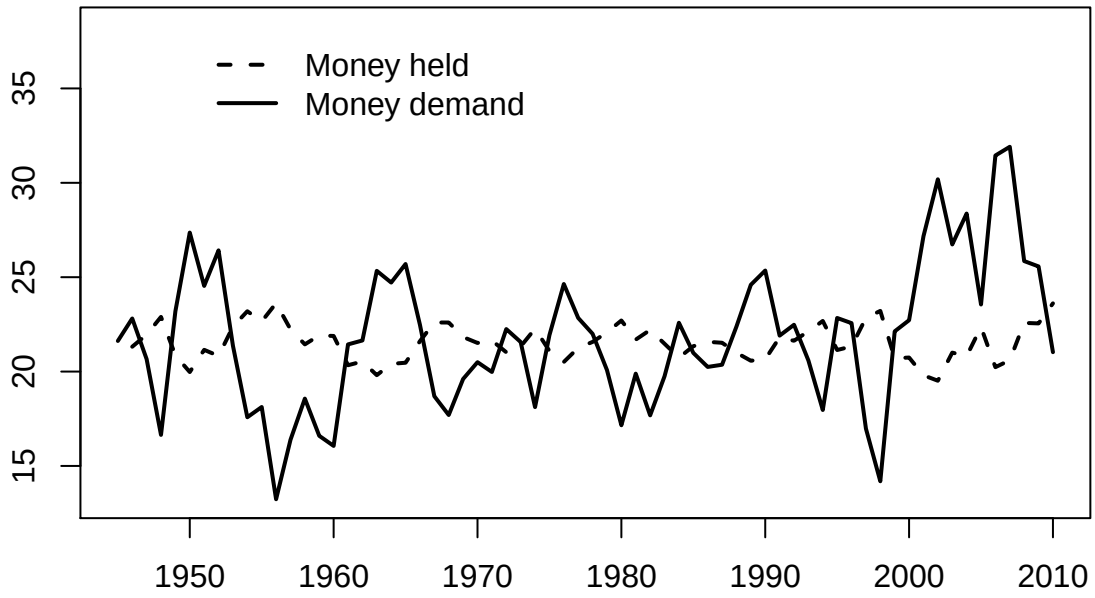
- Before simulatin the model, let's have a look at how the graph of the model has changed:

```
plot.dag(pcRand,main="PC Random" )
```

PC Random




```
datapcRand<-simulate(pcRand,maxIter=2)
plot(pcRand$time,datapcRand$baseline[,"h_h"],type="l",xlab="",ylab="",lty=1,lwd=2,
     ylim=c(1*min(datapcRand$baseline[,c("h_h","h_d")],na.rm=T),
            1.2*max(datapcRand$baseline[,c("h_h","h_d")],na.rm=T)))
lines(pcRand$time,datapcRand$baseline[,"h_d"],lty=2,lwd=2)
legend(x=1950,y=1.2*max(datapcRand$baseline[,c("h_h","h_d")],na.rm=T),
       legend=c("Money held","Money demand"),lty=c(2,1),lwd=2,bty="n")
```



- This graph highlights the buffer role of certain stocks in PK-SFC models.
- expectations are incorrect
- demand might not be equal to supply in any market
- one stock will not be equal to the targeted level, as highlighted by Foley (1975), in a model without perfect foresight you need a buffer stock in order to obtain equilibrium between demand and supply.
- The role of buffer stocks in PK-SFC model is thus fundamental and is at the hart of the approach used by Godley (1999) in his seven unsustainable processes. - It is by observing the dynamics of certain stock-flow norms that you are able to observe the unsustainable processes evolving in an economy, because stocks precisely absorb disequilibrium.

```
plot.sankey(as.data.frame(datapcRand$baseline),filename="TFM_PC.csv",period=2)
```

References

Foley, D. 1975. "On Two Specifications of Asset Equilibrium in Macroeconomic Models." *Journal of Political Economy* 83 (2).

Godley, Wynne. 1999. "Seven Unsustainable Processes: Medium-Term Prospects and Policies for the United States and the World." Strategic Analysis. The Levy Economic Institute of Bard College.

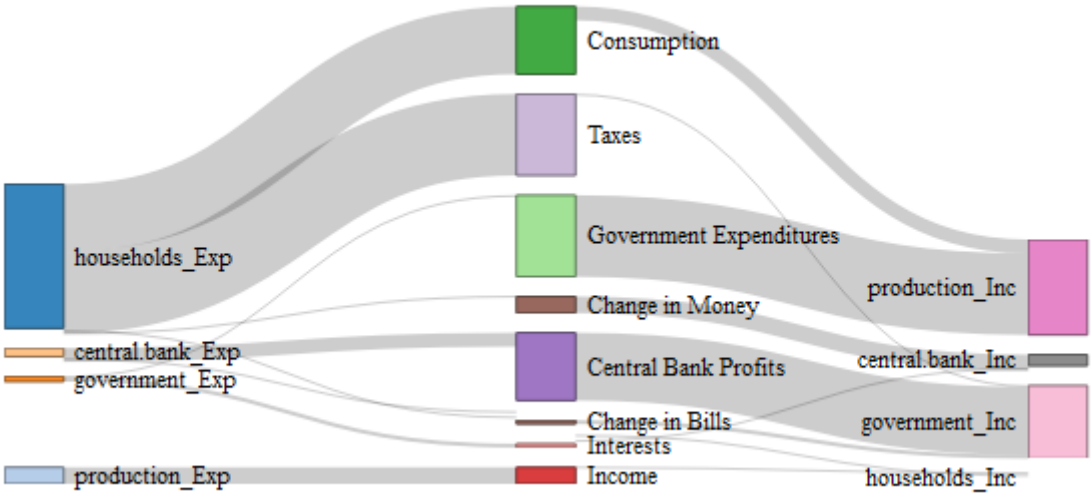

households_Exp
central.bank_Exp
government_Exp
production_Exp
Consumption
Taxes
Government Expenditures
Change in Money
Central Bank Profits
Change in Bills
Interests
Income
production_Inc
central.bank_Inc
government_Inc
households_Inc

Figure 1: