Setting up the environment

This tutorial is meant to reproduce the graphs and tables of chapter 4 of Godley and Lavoie (2007, G&L for now on) and at the same time discover some of the tools provided by the package PK-SFC.

Before doing any modelling, we need to load the package in the R environment.

```r
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

## Loading required package: networkD3
```

The, you need to download the two attached 'SIM.txt' and 'SIMEX.txt' file and save it in the folder of your choice. Make sure to set the working directory where you saved the downloaded file. In command line this looks like this but if you use Rstudio, you can use the graphical interface as well (Session>Set Working Directory>Choose Directory)

```r
setwd("pathToYourDirectory")
```

Loading the model

The first thing to do is to load the model and check for completeness.

```r
pc<-sfc.model("PC.txt",modelName="Portfolio Choice Model")
pc<-sfc.check(pc,fill=FALSE)
```

Let's have a look at the graphical representation of model PC. In order to do so, we need to load the `Rgraphviz` library:

```r
library("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'
```

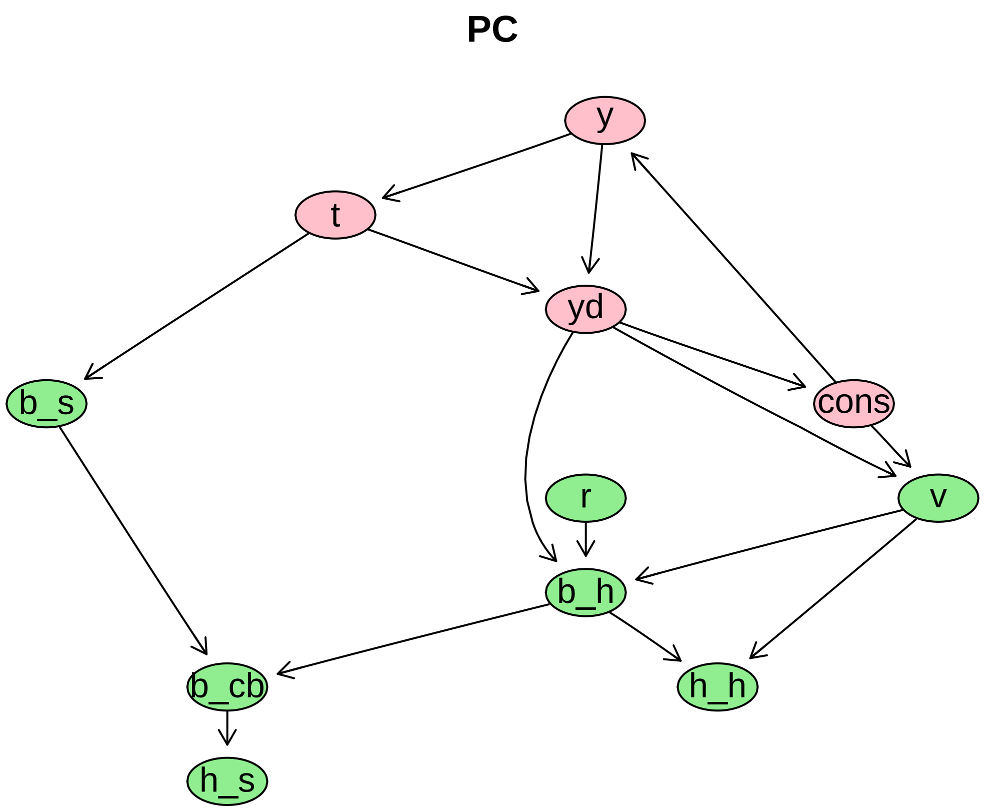
```
## The following objects are masked from 'package:igraph':
##
##     degree, edges, intersection
```

```
## Loading required package: grid
```

We can now look at the graph of model PC:

```r
plot.dag(pc,main="PC" )
```

```
## [1] "We now have a DAG to work with!"
```



You can see that there is a cycle in the graph, implying that GDP, taxes, disposable income and consumption are determined all together (and that they fully adapt to any shock applied to the economy). While this is a mathematical property of the system of equations representing the economy we wish to model, it has an economic meaning and you want to be sure that this is what you believe to be the best representation of what you have in mind.

We are now ready to simulate the model

```
datapc<-simulate(pc)
```

Expectations and random shocks

The first of experiment is meant to observe the buffer role of money, in case of random shocks applied to expected disposable income. To do these, we need to modify slightly model pc and change the equations determining consumption, demand for bonds and money, and the expectations on income and wealth. We will call the new model pcRand.

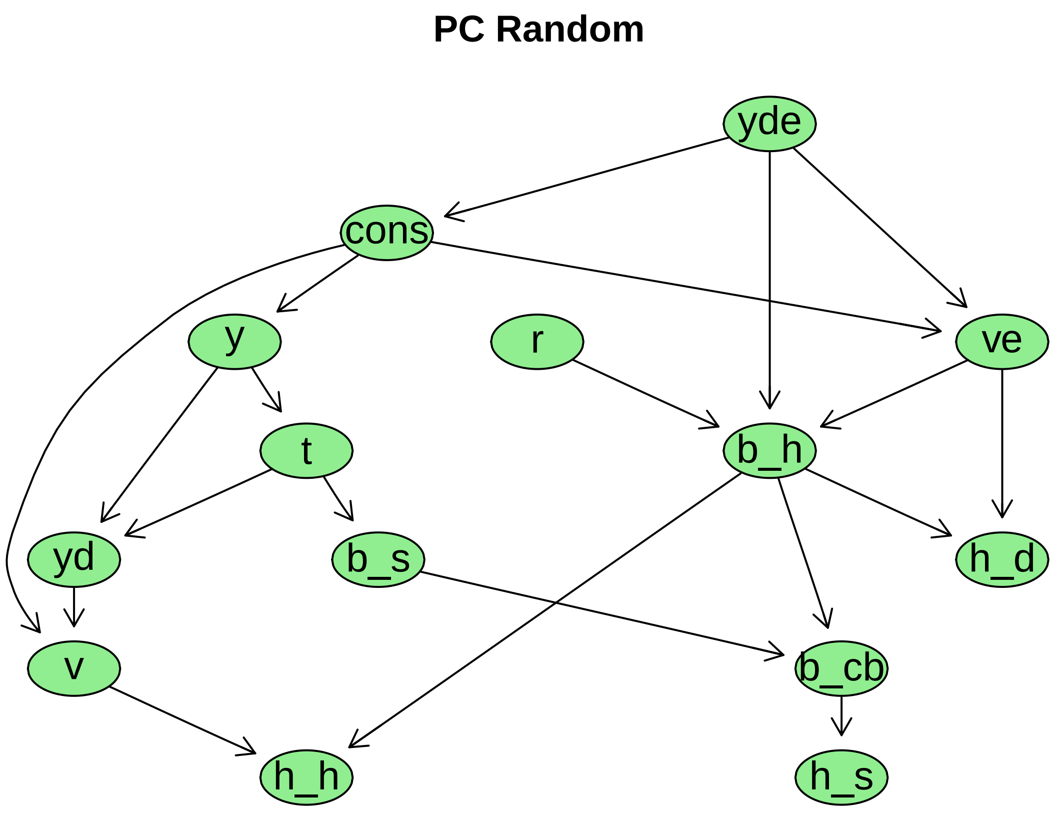
```
pcRand<-sfc.editEqus(pc,list(list(var="cons",eq="alpha1*yde+alpha2*v(-1)"),
                             list(var="b_h",eq="ve*(lambda0 + lambda1*r - lambda2*(yde/ve))")))
pcRand<-sfc.addEqus(pcRand,list(
  list(var="yde",eq="yd(-1)*(1+rnorm(1,sd=0.1))",desc="Expected disposable income depending on random s
  list(var="h_d",eq="ve-b_h",desc="Money demand"),
  list(var="ve",eq="v(-1)+yde-cons",desc="Expected disposable income")))
pcRand<-sfc.editVar(pcRand,var="yd",init=86.48648)
pcRand<-sfc.check(pcRand)
```

You probably have noticed that the `yde` equation contains the R function `rnorm` which allows you to extract a random number from a normal distribution centered around 0, with standard variation of 0.1. The package indeed allows you to use any R function such as `min`, `max` or `rnorm`. However, because of the way the Gauss-Seidel algorithm works, we are facing a problem. Indeed, as the `rnorm` function extract a number in each iteration, this will mean that the algorithm cannot converge since the difference between each iteration of the algorithm depends of the difference between each extraction. This is why we need to restrict the number of iteration of the Gauss-Seidel.

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

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

```
## [1] "We now have a DAG to work with!"
```

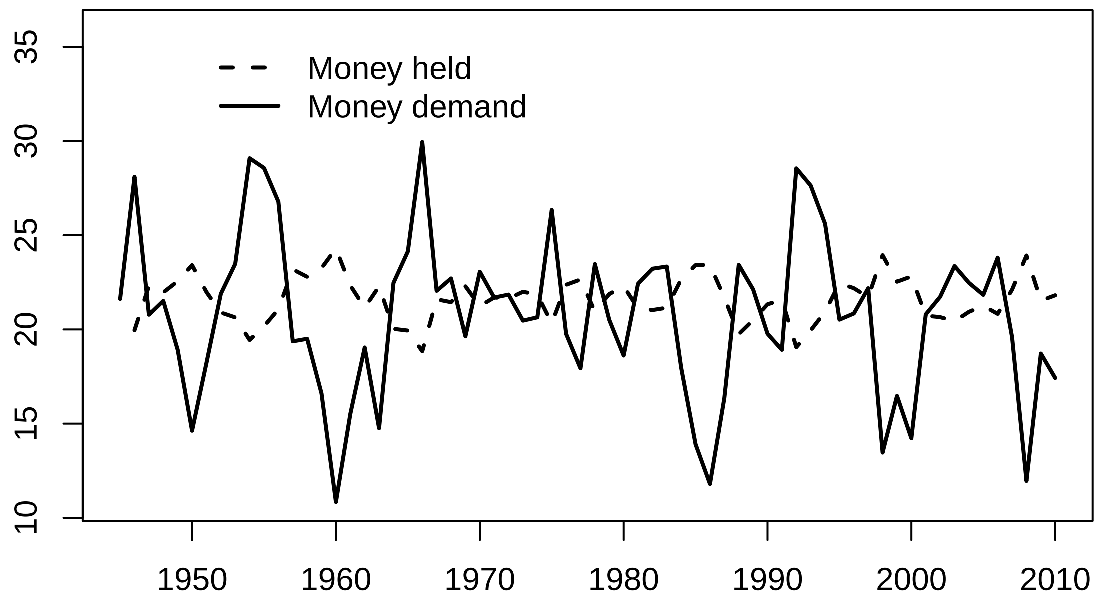


You can now see that the cycle observed in the original model has disapeared. Indeed, consumption doesn't depend on disposable income anymore but on expected disposable income. Let's now simulate the model.

```
datapcRand<-simulate(pcRand,maxIter=2)
```

This replicates figure 4.1, page 110

```
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. Indeed, because expectations are incorrect or because demand might not be equal to supply in any market, at least 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.

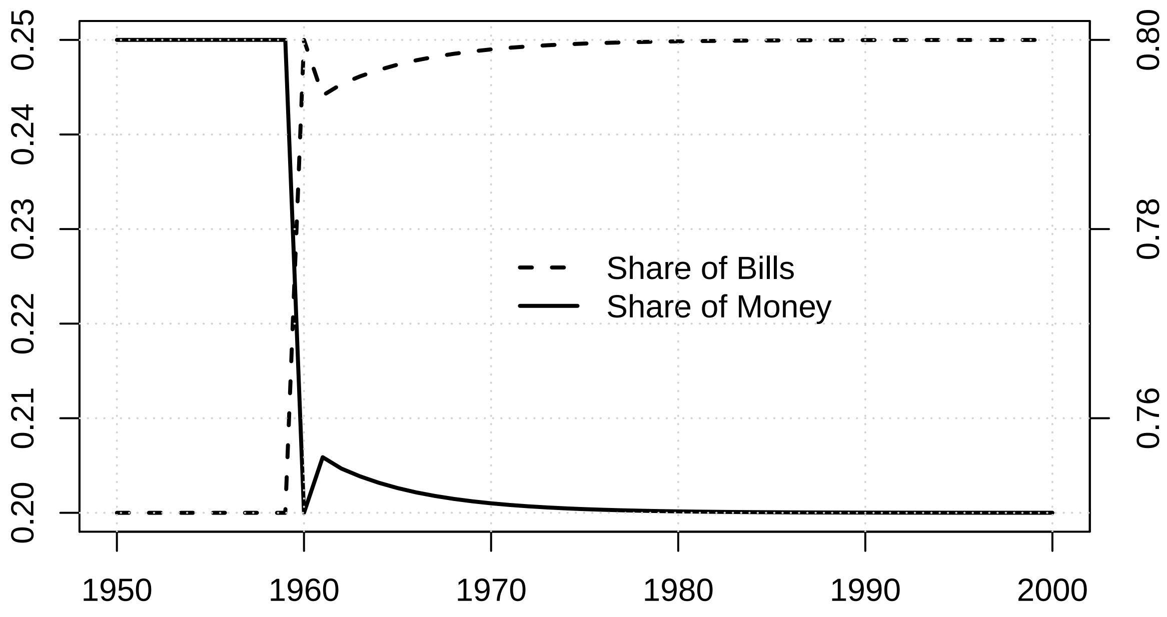
Interest rates impacts

The graphs presented on the paper are based on the model PCEX, which includes expectation on disposable income and wealth. We will first create the model before simulating it.

```
pcex<-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))")))
pcex<-sfc.addEqus(pcex,list(
  list(var="yde",eq="yd(-1)",desc="Epected disposable income depending on random shocks"),
  list(var="h_d",eq="ve-b_h",desc="Money demand"),
  list(var="ve",eq="v(-1)+yde-cons",desc="Expected disposable income")))
pcex<-sfc.editVar(pcex,var="yd",init=86.48648)
pcex<-sfc.check(pcex)
init = datapc$baseline[56,]
pcex<-sfc.addScenario(model=pcex,vars="r_bar",values=0.035,inits=1960,ends=2010,starts=init)
datapcex<-simulate(pcex)
```

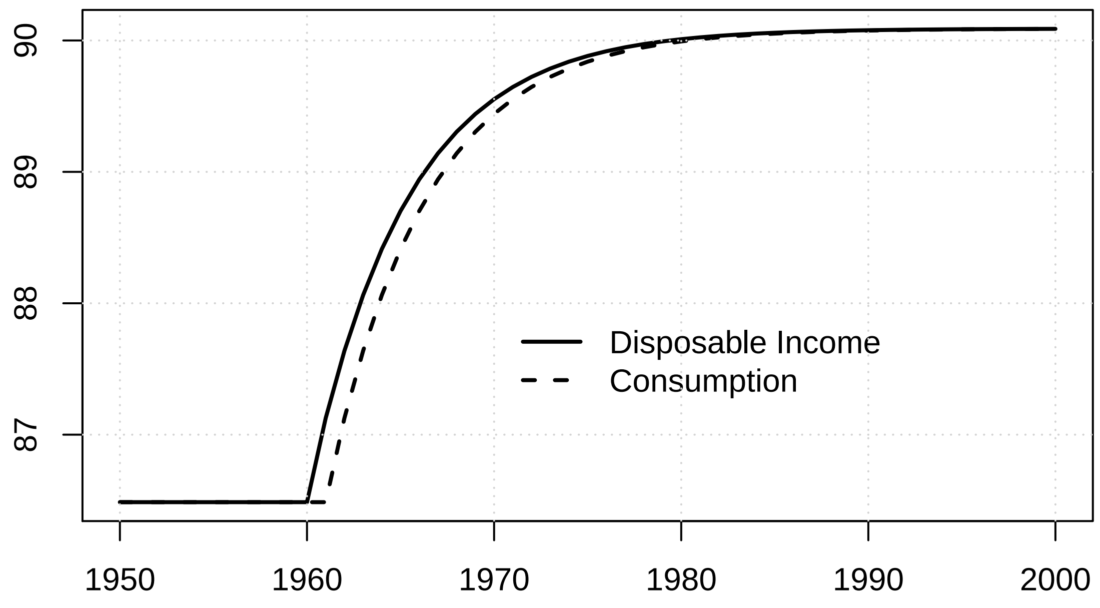
This replicates plot figure 4.3.. p-112

```
time2=c(1950:2000)
plot(time2,
     datapcex$scenario[as.character(time2),"h_h"]/datapcex$scenario[as.character(time2),"v"],
     type="l",xlab="",ylab="",lty=1,lwd=2)
par(new=T)
plot(time2,
     (datapcex$scenario[as.character(time2),"b_h"]/datapcex$scenario[as.character(time2),"v"]),
     lty=2, lwd=2,type="l",axes=F,ylab="",xlab="")
axis(4,pretty(c(0.750, 1.1*max(datapcex$scenario[,"b_h"]/datapcex$scenario[,"v"],na.rm=T))))
legend(x=1970,y=0.78,legend=c("Share of Bills","Share of Money"),lty=c(2,1),lwd=2,bty="n")
grid()
```



This replicates fig 4.4 . p 113

```
time2=c(1950:2000)
plot(time2,datapcex$scenario[as.character(time2),"yd"],type="l",xlab="",ylab="",lty=1,lwd=2)
lines(time2,datapcex$scenario[as.character(time2),"cons"],type="l",xlab="",ylab="",lty=2,lwd=2)
legend(x=1970,y=88,legend=c("Disposable Income","Consumption"),lwd=c(2,2),lty=c(1,2),bty="n")
grid()
```



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.