###Setting up the environment

This tutorial is meant to reproduce the graphs and tables of chapter 3 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.

```
library(PKSFC)
```

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

```
## Warning: package 'expm' was built under R version 3.5.3
```

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

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

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

```
## Loading required package: igraph
```

```
## Warning: package 'igraph' was built under R version 3.5.3
```

```
##
## 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
```

```
## Warning: package 'networkD3' was built under R version 3.5.1
```

The, you need to download the attached 'SIM.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 comand line this looks like this but if you use Rstudio, you can use the graphical interface as well (Session>Set Working Dirctory>Choose Directory)

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

Now we are ready to start developing the model. The first thing to do is to have a look at the source code you just downloaded. The equations are exactly those of Appendix 3.1 in page 91 of G&L.

```
#MODM
C_s = C_d
G_s = G_d
T_s = T_d
N_s = N_d
Yd = W*N_s - T_s
T_d = theta*W*N_s
C_d = alpha1*Yd + alpha2*H_h(-1)
H_s = H_s(-1) + G_d - T_d
H_h = H_h(-1) + Yd - C_d
Y = C_s + G_s
N_d = Y/W
```

The source code has to follow a certain format.

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.

There should not be any empty lines.

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.

###Loading the model and completing it

The first thig to do is to load the model using the function `sfc.model`. The function takes one mandatory argument (the filename of the source code) and three optional arguments. We will not cover the usage of those execpt for the modelName which allows you to give a more detailed name for your model.

```
sim<-sfc.model("SIM.txt",modelName="SIMplest model")
```

```
## Warning in sfc.check(model, fill = fill): The following variables have lags
## but no initial values: - H_s - H_h
```

```
## Warning in sfc.check(model, fill = fill): Equations
##  G_d
##  W * 0 - 0
##  theta * W * 0
##  alpha1 * 0 + alpha2 * NA
##  NA + G_d - 0
##  0/W
## contain variables that are not defined, check the model
```

```
## Warning in sfc.check(model, fill = fill): No year defined
```

Note that you should have recieved some warning message, as indicated hereabove. The first block of warning indicate that there are undefined variables in the equations. Indeed, we have not defined `G_s`, `W`, `theta`, and so on. The second warning indicates that we have not set any timeline for the simulations. We will see how to complete the model. Note that at any time, you can check for the model status by using the `sfc.check` function.

```
sim<-sfc.check(sim,fill=FALSE)
```

```
## Warning in sfc.check(sim, fill = FALSE): The following variables have lags
## but no initial values: - H_s - H_h
```

```
## Warning in sfc.check(sim, fill = FALSE): Equations
##  G_d
##  W * 0 - 0
##  theta * W * 0
##  alpha1 * 0 + alpha2 * NA
##  NA + G_d - 0
##  0/W
## contain variables that are not defined, check the model
```

```
## Warning in sfc.check(sim, fill = FALSE): No year defined
```

If you set the `fill` parameter to `TRUE`, then R will ask you on the comand line if you want to enter manualy the undefined variables. You will then have to enter the name, value and description of each variable, until the model is complete. You can also add the timeline via the same process.

However, we will use the functions `sfc.addVar` and `sfc.addVars`. You can add a variable by using the first one.

```
sim<-sfc.addVar(sim,"G_d",20,"Government expenditure")
```

Or if you have more than one variable, you can add them 'en masse'.

```
sim<-sfc.addVars(sim,list(list(var="theta",init=0.2,desc="tax rate"),
                          list(var="alpha1",init=0.6,desc="propensity to consume out of income"),
                          list(var="alpha2",init=0.4,desc="propensity to consume out of wealth"),
                          list(var="W",init=1,desc="wage rate")))
```

We can also fill the timeline using the function `sfc.setYear`. The package needs to know how many periods the model has to be simulated. Because we follow G&L, we set the timelines to be equal to 1946 to 2010, but it would not change to put 1 to 65.

```
sim<-sfc.setYear(sim,1945,2010)
```

The last step to do in order to close the model is to set the initial values for all endogenous variables which are used with a lag in the equations. In our case, we have to do this for `H_h` and `H_s`. We can do that using the `sfc.editEnds` function.

```
sim<-sfc.editEnds(sim,list(list(var="H_h",init=0),list(var="H_s",init=0)))
```

We are now ready to run the first simulations. But before, let's check wether the model is complete.

```
sim<-sfc.check(sim,fill=FALSE)
```

###First simulations

To run simulation, we will use the `simulate` function.

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

We can now try to replicate table 3.4 page 69 of G&L

```
round(t(datasim$baseline[c(1,2,3,66),c("G_s","Y","T_s","Yd","C_s","H_s","H_h")]),digits=1)
```

```
##     1945 1946 1947 2010
## G_s   NA 20.0 20.0   20
## Y     NA 38.5 47.9  100
## T_s   NA  7.7  9.6   20
## Yd    NA 30.8 38.3   80
## C_s   NA 18.5 27.9   80
## H_s    0 12.3 22.7   80
## H_h    0 12.3 22.7   80
```

Before going further, we should analyse a bit the datastructure generated by the the `simulate` function. The result of this function is a list whose name are composed of the result of the baseline scenario simulation and all eventual scenarios. You can always check what result exists by using the 'names' function:

```
names(datasim)
```

```
## [1] "baseline"
```

In the present case, we have only one dataset which is composed of the simulation results of the baseline scenario. In order to obtain the various plots of the chapter, we need to add alternative scenarios. A scenario can be defined as a simulation for which you can define a set of initial values for the endogenous variables

and a set of (eventualy time varying) exogenous parameters values. The following code allows you to add a scenario to the current sim model. Why do we need it? Because as it is set up, the model starts with zero wealth, but most of the simualtion done in the chapter actualy start from the steady state emerging out of the government spending 20.

The first line of code allows to obtain the initial values for the scenario from the last value of the baseline simulation. The second line defines the scenario as being a shock applied to the exogenous variable 'g' whose value become 25 from 1960 until 2010. The initial value of the scenario being given by the init vector obtained previously.

```
init = datasim$baseline[66,]
sim<-sfc.addScenario(model=sim,vars="G_d",values=25,inits=1960,ends=2010,starts=init)
```

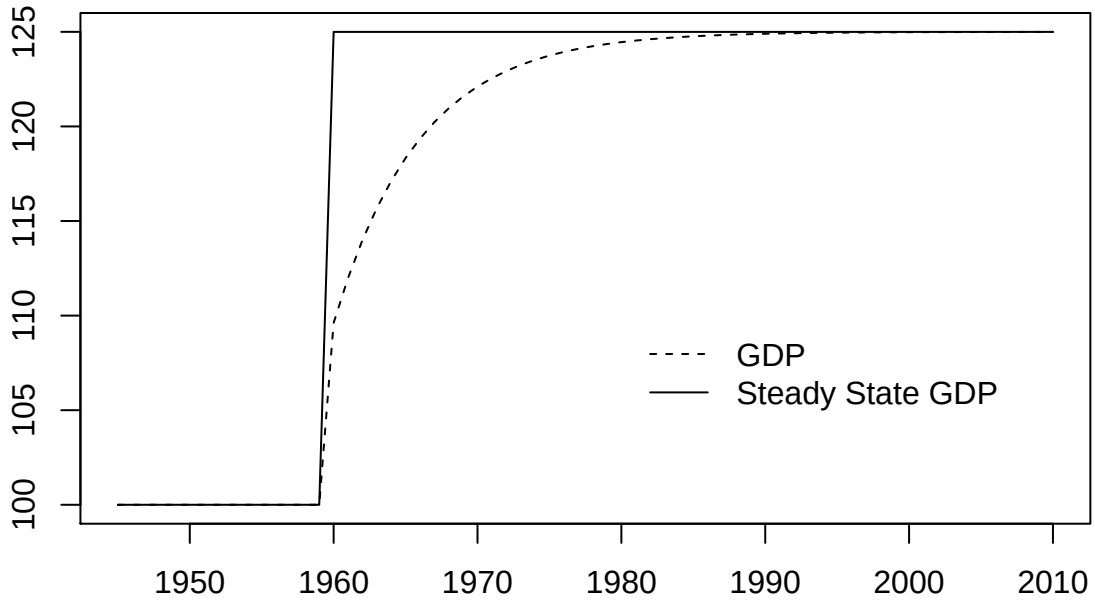
We can now run again the simulations. Note that now the datasim object is composed of two elements called baseline and scenario_1.

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

```
## [1] "baseline"   "scenario_1"
```

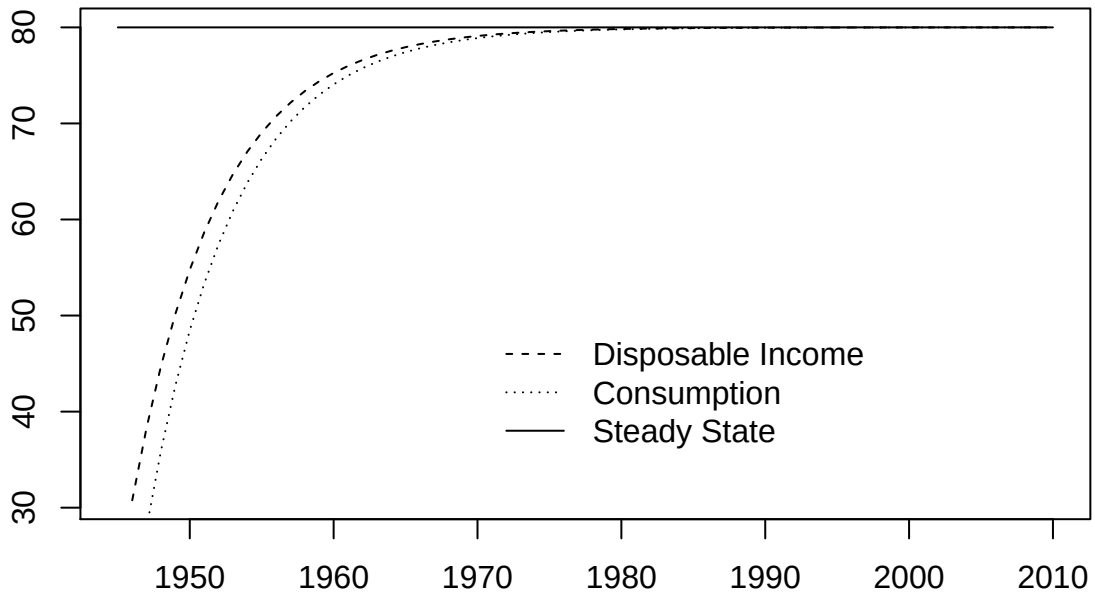
This replicates figure 3.1 page72

```
plot(sim$time,datasim$scenario_1[,"Y"],type="l",xlab="",ylab="",lty=2)
lines(sim$time,(sim$time>1959)*(datasim$scenario_1["2010","Y"]-datasim$scenario_1["1958","Y"])
      +datasim$scenario_1["1958","Y"])
legend(x=1980,y=110,legend=c("GDP","Steady State GDP"),lty=c(2,1),bty="n")
```



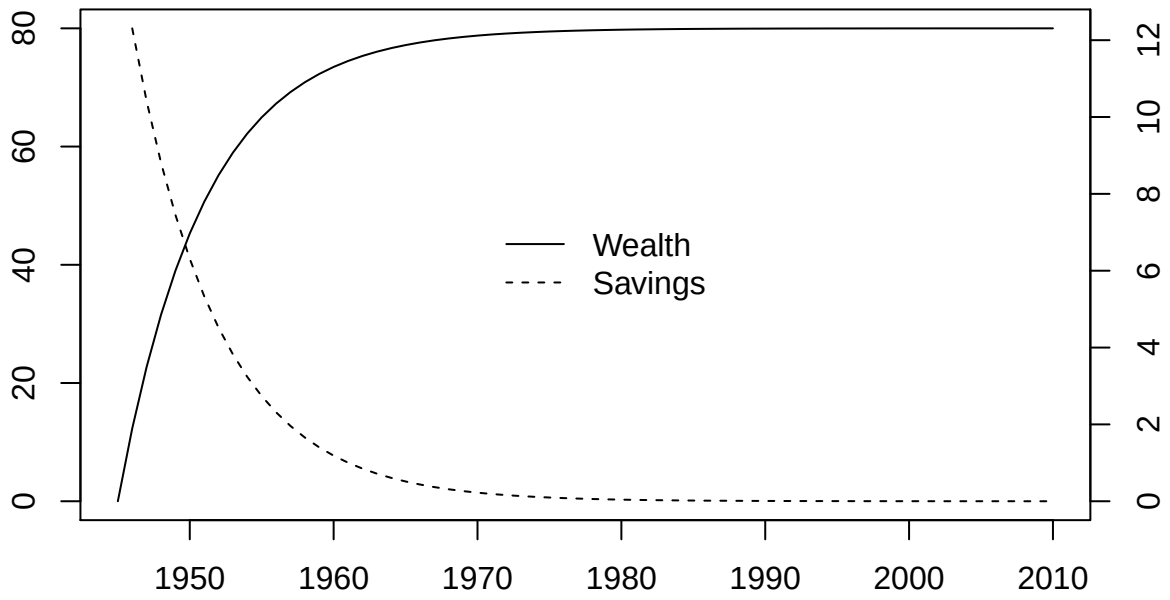
This replicates figure 3.2 page 73

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



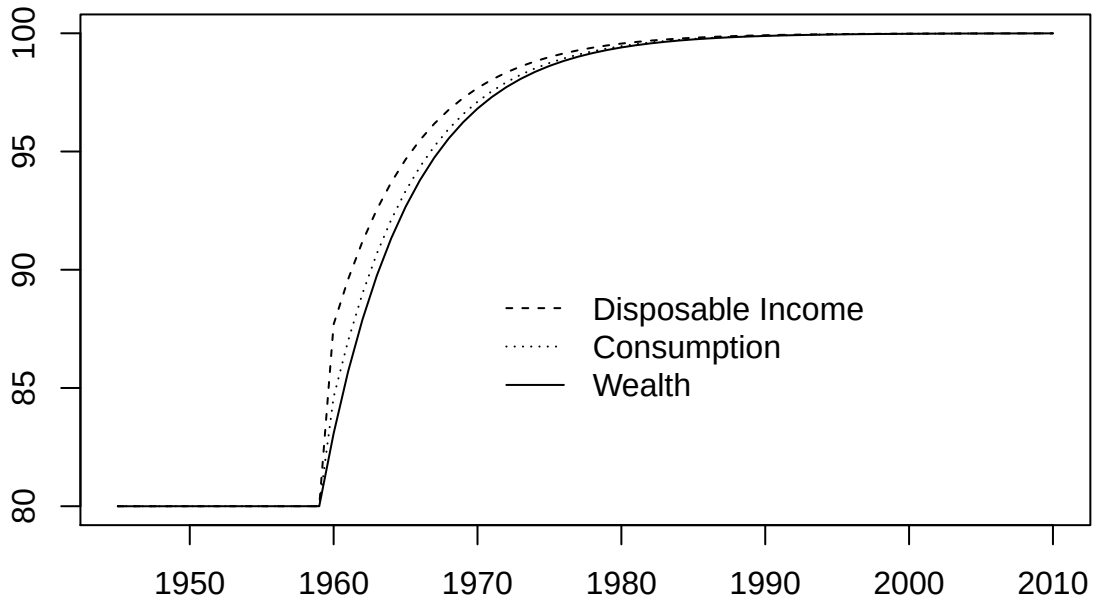
This replicates figure 3.3 page 75

```
plot(sim$time,datasim$baseline[,"H_s"],type="l",xlab="",ylab="")
legend(x=1970,y=50,legend=c("Wealth","Savings"),lty=c(1,2),bty="n")
par(new=T)
plot(sim$time,(datasim$baseline[,"Yd"]-datasim$baseline[,"C_s"]),
     lty=2,type="l",axes=F,ylab="",xlab="")
axis(4,pretty(c(0, 1.1*max(datasim$baseline[,"Yd"]-datasim$baseline[,"C_s"],na.rm=T))))
```



This replicates figure 3.4 page 76

```
plot(sim$time,datasim$scenario_1[,"Yd"],type="l",xlab="",ylab="",lty=2)
lines(sim$time,datasim$scenario_1[,"C_s"],lty=3)
lines(sim$time,datasim$scenario_1[,"H_s"])
legend(x=1970,y=90,legend=c("Disposable Income","Consumption","Wealth"),
       lty=c(2,3,1),bty="n")
```



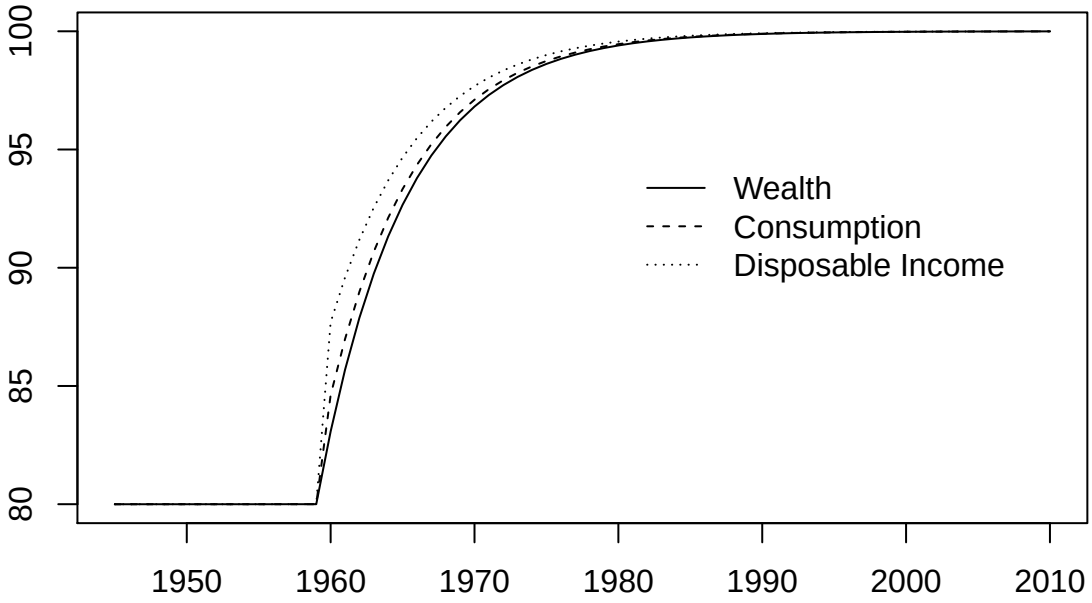
###Out of the steady state

Section 3.8 analyses the case of an economy switching to a new steady state due to an increase in the propopensity to consume for the model SIM.

```
init = datasim$baseline[66,]
sim<-sfc.addScenario(sim,"alpha1",0.7,1960,2010,init)
datasim<-simulate(sim)
```

This replicates figure 3.8 page 86

```
plot(sim$time,datasim$scenario_1[,"H_s"],type="l",xlab="",ylab="",xlim=range(sim$time),
     ylim=range(datasim$scenario_1[,"H_s"],datasim$scenario_1[,"C_d"],datasim$scenario_1[,"Yd"]))
lines(sim$time,datasim$scenario_1[,"C_d"],lty=2)
lines(sim$time,datasim$scenario_1[,"Yd"],lty=3)
legend(x=1980,y=95,legend=c("Wealth","Consumption","Disposable Income"),lty=c(1,2,3),bty="n")
```


100
95
90
85
80
1950
1960
1970
1980
1990
2000
2010
Wealth
Consumption
Disposable Income