Lab No. 4 — The Dynamic Factor Model for SGPE

Advanced Time Series Econometrics Labs

Ping Wu (ping.wu@strath.ac.uk (mailto:ping.wu@strath.ac.uk))

Outline of today's lab

- The Dynamic Factor Model (DFM)

Packages we will use in this lab

- dfms (https://cran.r-project.org/web/packages/dfms/dfms.pdf): Efficient estimation of Dynamic Factor Models using the Expectation Maximization (EM) algorithm or Two-Step (2S) estimation, supporting datasets with missing data.
- xts (https://cran.r-project.org/web/packages/xts/xts.pdf): eXtensible Time Series. Provide for uniform handling of R's different time-based data classes by extending zoo, maximizing native format information preservation and allowing for user level customization and extension, while simplifying cross-class interoperability.

```
# Important packages described above
install.packages("dfms", repos = "https://cran.rstudio.com/",
    dependencies = TRUE)
```

```
## Warning: cannot remove prior installation of package 'dfms'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\qdb18176\AppData\Local\Programs\R\R-4.3.3\library\00LOCK\dfms\libs\x64\
dfms.dll
## to
## C:\Users\qdb18176\AppData\Local\Programs\R\R-4.3.3\library\dfms\libs\x64\dfms.dl
l:
## Permission denied
```

```
## Warning: restored 'dfms'
```

```
install.packages("xts", repos = "https://cran.rstudio.com/",
    dependencies = TRUE)
```

```
## Warning: cannot remove prior installation of package 'xts'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\qdb18176\AppData\Local\Programs\R\R-4.3.3\library\00LOCK\xts\libs\x64\x
ts.dll
## to
## C:\Users\qdb18176\AppData\Local\Programs\R\R-4.3.3\library\xts\libs\x64\xts.dll:
## Permission denied
```

```
## Warning: restored 'xts'
```

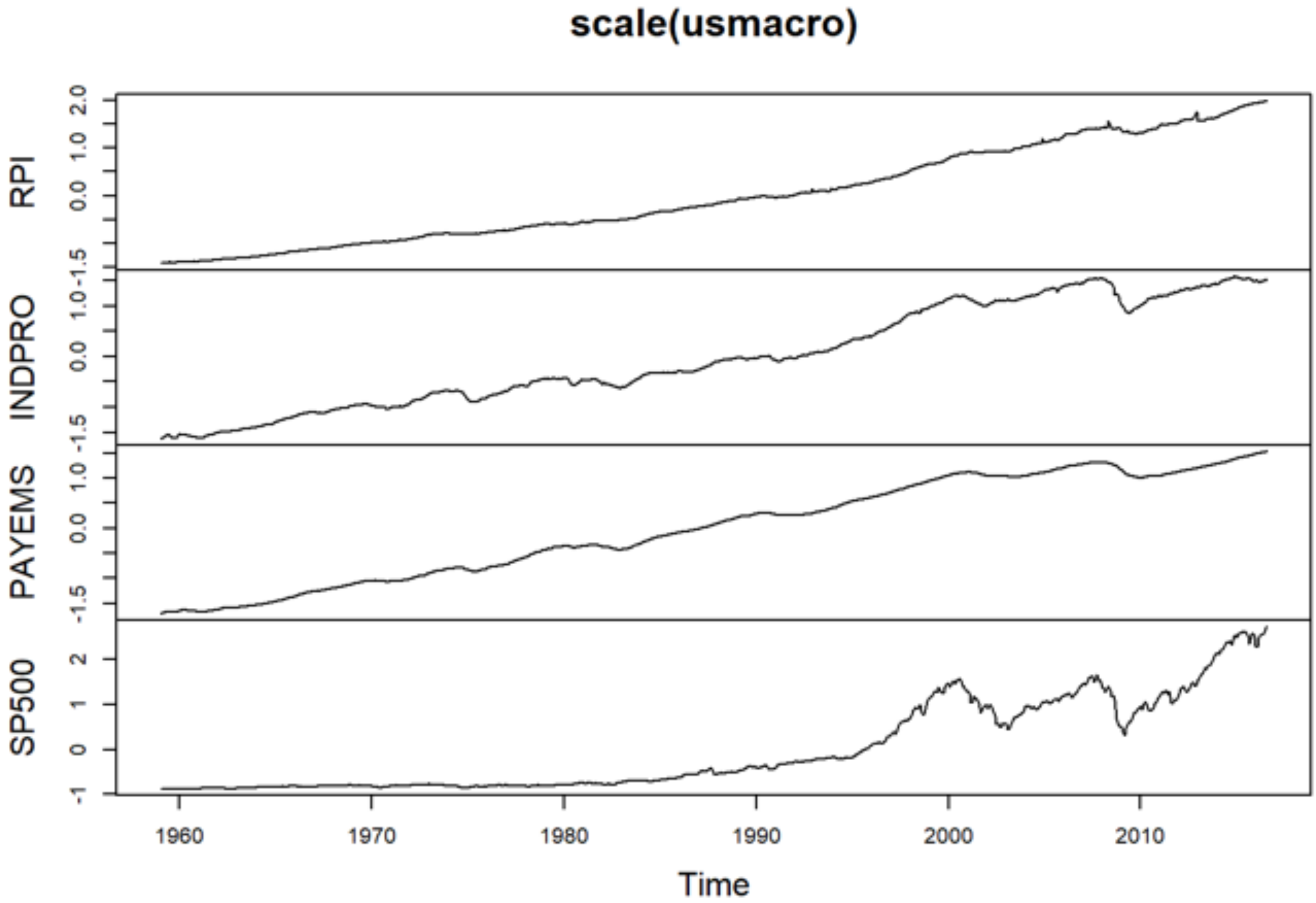
The next step is to make sure that you can access the routines in this package by making use of the *library* command, which would need to be run regardless of the machine that you are using.

```
# Important packages described above
library(dfms)
library(xts)
library(magrittr)
```

Part 0: Import data into R

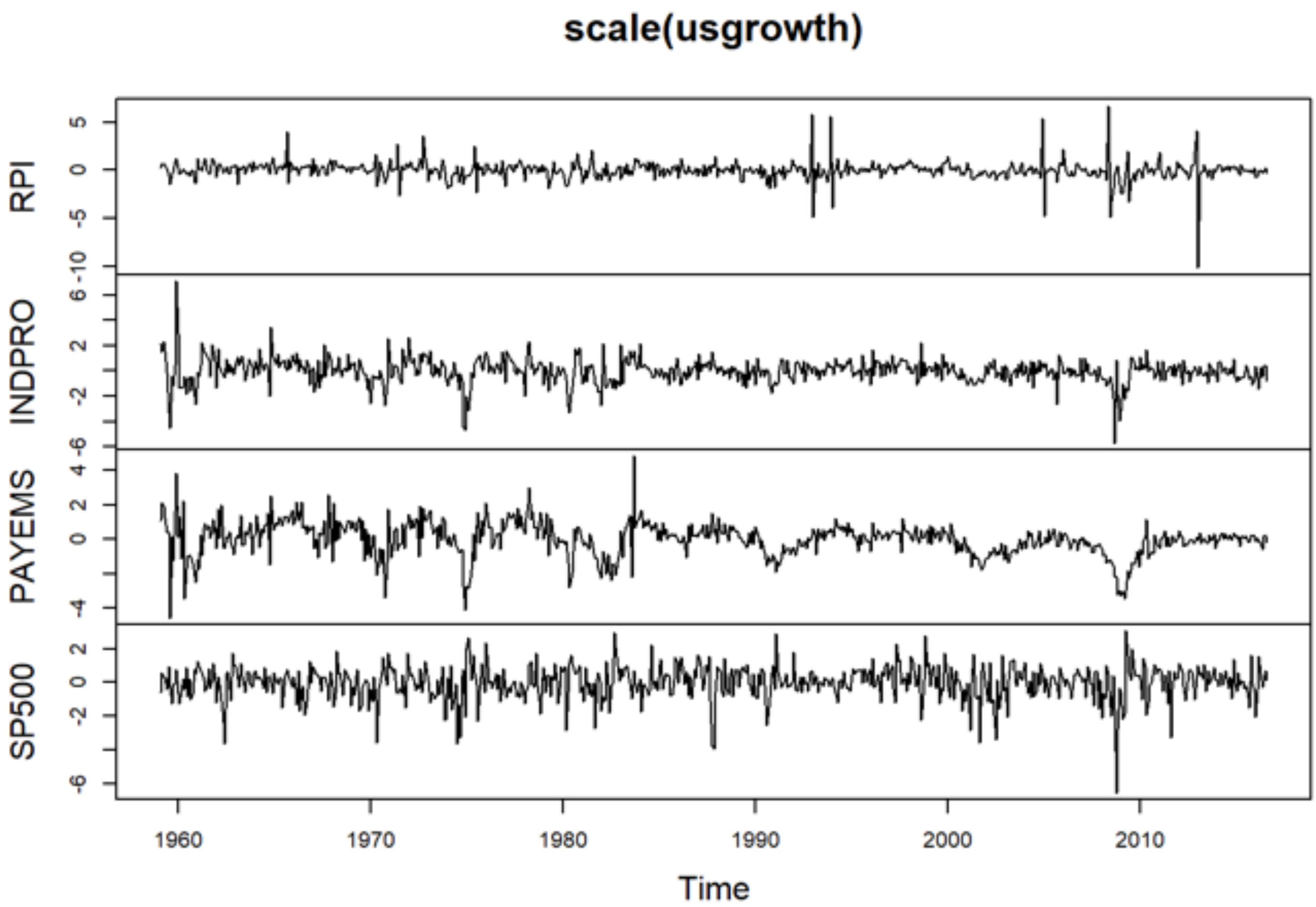
The data provided in the package in xts format is taken from Banbura and Modugno (2014), henceforth BM14, and covers the Euro Area from January 1980 through September 2009.

```
usmacro <- read.table("macro_data_DFM.csv", sep=",",header=T)
# Select the following macroeconomic variables
var.slct <- c("RPI", "INDPRO", "PAYEMS", "SP500")
usmacro <- ts(usmacro[,var.slct], end = c(2016, 8), frequency = 12)
plot(scale(usmacro), lwd = 1)
```



You will see the data is not stationary, so first take the log difference

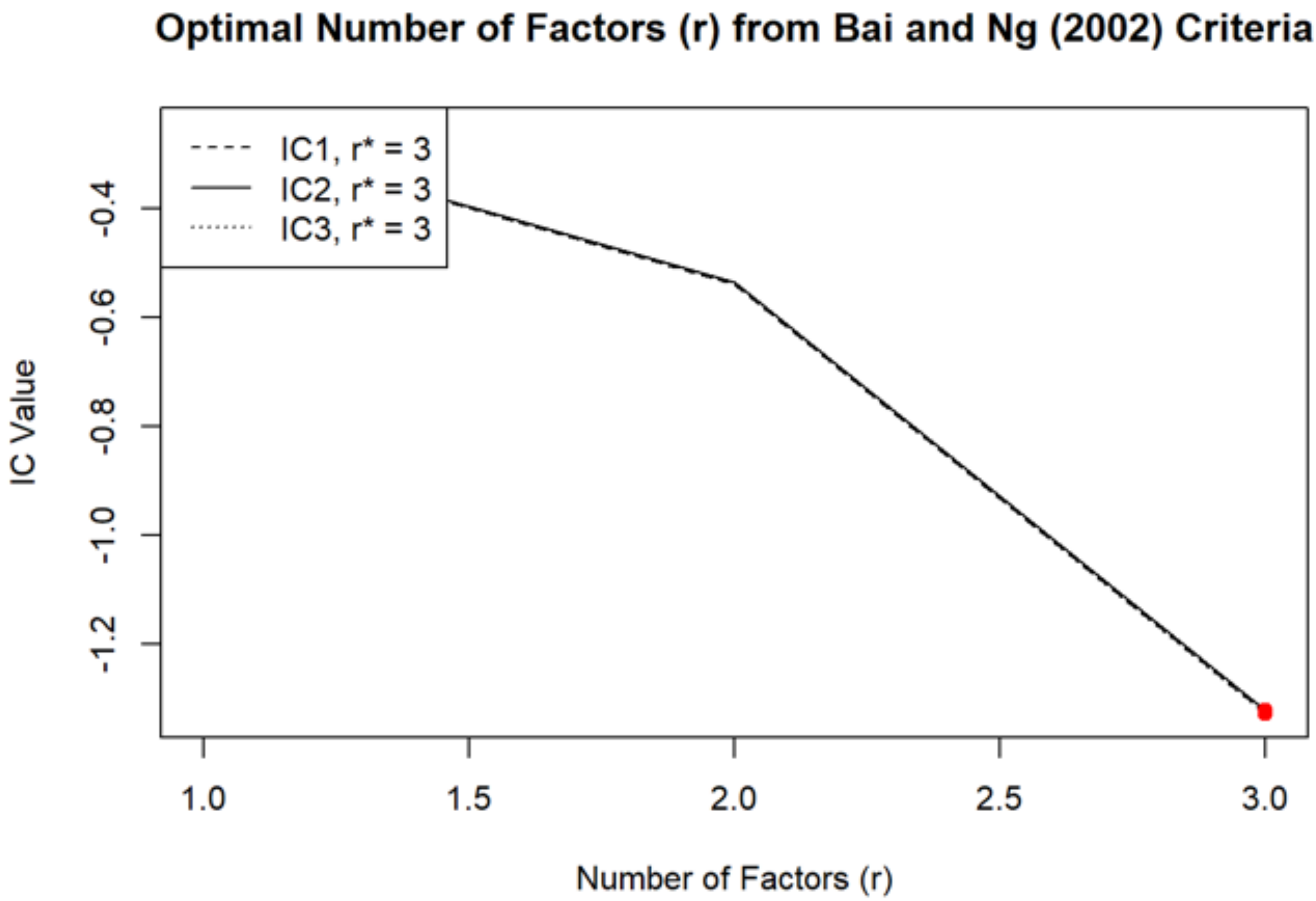
```
usgrowth = usmacro %>%
  log() %>%   # take log
  diff()      # take the difference
plot(scale(usgrowth), lwd = 1)
```



Part I: Brief idea about the Structure of the Model

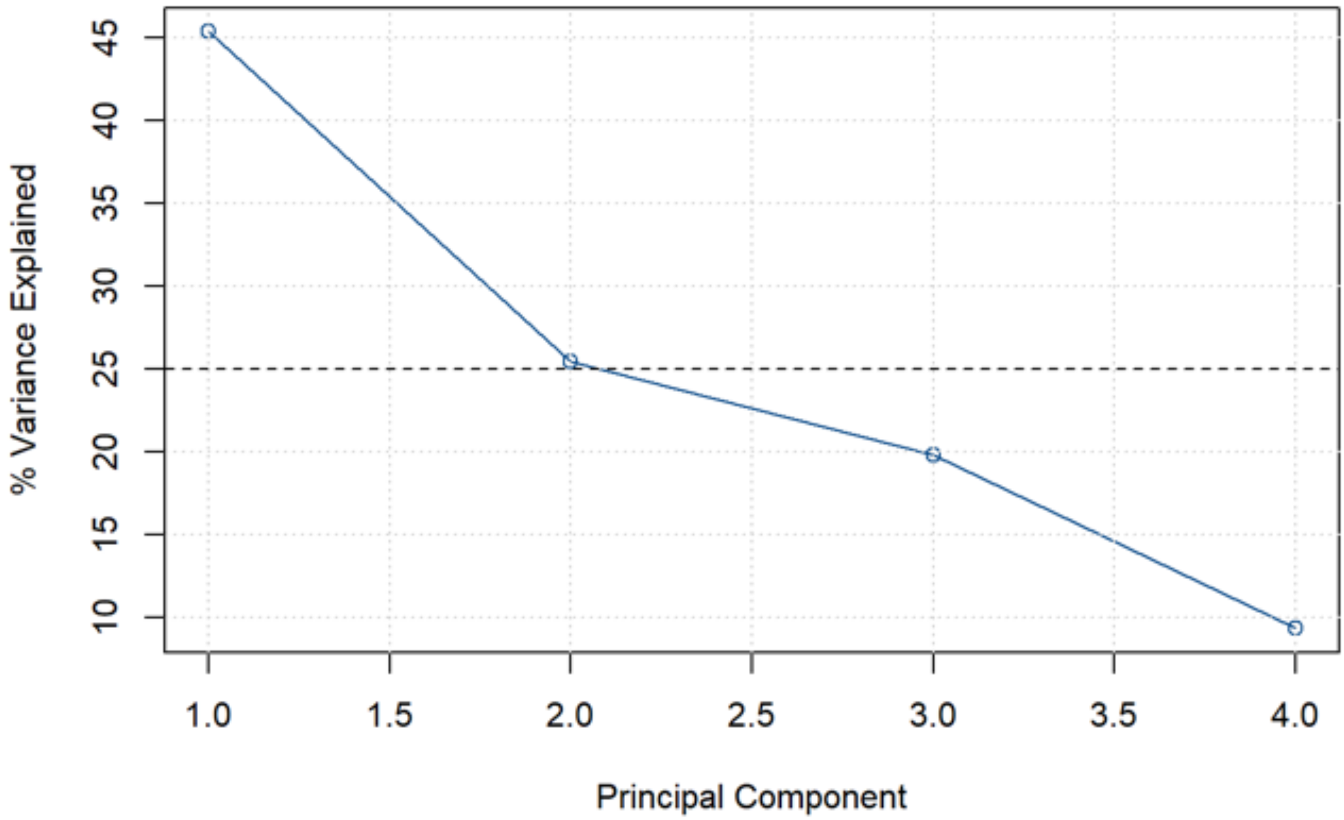
Before estimating a model, the *ICr()* function can be applied to determine the number of factors. It computes 3 information criteria proposed in Bai and NG (2002), whereby the second criteria generally suggests the most parsimonious model

```
ic = ICr(usgrowth)
print(ic)
plot(ic)
```



Another option is to use a Screeplot to gauge the number of factors by looking for a kink in the plot. A mathematical procedure for finding the kink was suggested by Onatski (2010), but this is currently not implemented in *ICr()*

```
screeplot(ic)
```



Based on both the information criteria and the Screeplot, I gauge that a model with 2 or 3 factors should be estimated. Next to the number of factors, the lag order of the factor-VAR of the transition equation should be estimated (the default is 1 lag). This can be done using the *VARselect()* function from the vars package, with PCA factor estimates reported by *ICr()*

```
vars::VARselect(ic$F_pca[, 1:3]) # Using 3 PCs to estimate the VAR lag order
```

```
## $selection
## AIC(n)  HQ(n)  SC(n) FPE(n)
##      3      3      2      3
##
## $criteria
##                   1           2            3           4           5
## AIC(n) -0.078635092 -0.16586566 -0.205639155 -0.19541039 -0.18718124
## HQ(n)  -0.047783265 -0.11187497 -0.128509587 -0.09514195 -0.06377394
## SC(n)   0.001075257 -0.02637255 -0.006363282  0.06364824  0.13166015
## FPE(n)  0.924377550  0.84716187  0.814131953  0.82250849  0.82931494
##                  6           7           8           9          10
## AIC(n) -0.19316353 -0.18148087 -0.17120004 -0.16114373 -0.15916801
## HQ(n)  -0.04661735 -0.01179582  0.02162388  0.05481905  0.07993365
## SC(n)   0.18546063  0.25692605  0.32698964  0.39682871  0.45858720
## FPE(n)  0.82438298  0.83409047  0.84273644  0.85128817  0.85301417
```

The selection thus suggests we should estimate a factor model with *r = 3* factors and *p = 3* lags. (In Gary's slides, the number of factors is represented by *m*.)

Part II: DFM estimation

The DFM can be written as

$$
\begin{aligned}
y_t &= C f_t + \varepsilon_t, \\
f_t &= A_1 f_{t-1} + A_2 f_{t-2} + \cdots + A_p f_{t-p} + u_t, \quad \varepsilon_t \sim \mathcal{N}(0, W), \\
\varepsilon_t &= \rho_1 \varepsilon_{t-1} + \cdots + \rho_q \varepsilon_{t-q} + e_t, \quad e_t \sim \mathcal{N}(0, V).
\end{aligned}
$$

Estimation can be done using the *DFM()* function with parameters *r* and *p*

```
model1 = DFM(usgrowth,
             r = 3,          # Number of factors
             p = 3,          # Number of lags
             idio.ar1 = TRUE  # Model observation errors as AR(1) processes?
             )
```

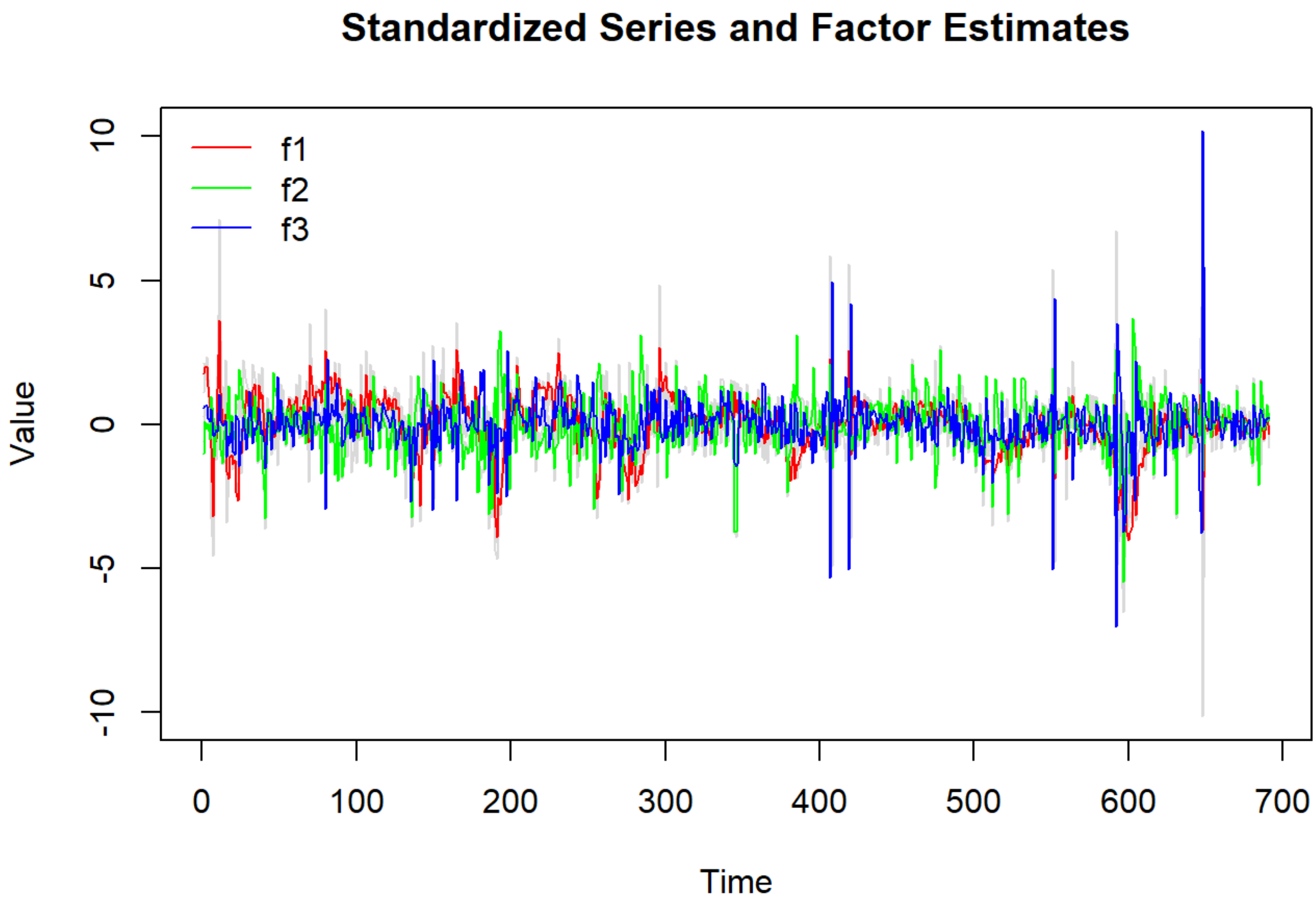
```
## Converged after 26 iterations.
```

```
summary(model1)
```

```
## Dynamic Factor Model: n = 4, T = 691, r = 3, p = 3, %NA = 0
##    with AR(1) errors: mean(abs(rho)) = 0.225
##
## Call:  DFM(X = usgrowth, r = 3, p = 3, idio.ar1 = TRUE)
##
## Summary Statistics of Factors [F]
##        N     Mean  Median      SD      Min     Max
## f1   691   0.0008  0.1111   1.294  -5.2052  4.6495
## f2   691  -0.0002  0.0486  1.0081  -5.5235  3.6923
## f3   691   0.0003  0.0426  0.8707  -6.1426  8.8664
##
## Factor Transition Matrix [A]
##        L1.f1    L1.f2    L1.f3     L2.f1    L2.f2     L2.f3    L3.f1   L3.f2
## f1  0.45171 -0.07127  0.32123  0.144893  0.05173  0.135272  0.11836 0.05136
## f2 -0.02297  0.24484  0.07677 -0.071177 -0.10631  0.007592 -0.07891 0.06957
## f3  0.11758  0.07664 -0.14349 -0.004614  0.05430 -0.102023  0.12552 0.02013
##        L3.f3
## f1  0.08071
## f2 -0.04880
## f3 -0.06292
##
## Factor Covariance Matrix [cov(F)]
##           f1        f2        f3
## f1   1.6745    0.0212   -0.0705
## f2   0.0212    1.0163    0.0120
## f3  -0.0705    0.0120    0.7581
##
## Factor Transition Error Covariance Matrix [Q]
##          u1     u2      u3
## u1  0.8368 0.1816 -0.1864
## u2  0.1816 0.9011  0.0344
```

```
## u3 -0.1864 0.0344  0.6941
##
## Observation Matrix [C]
##            f1      f2      f3
## RPI    0.4325  0.2671 -0.8612
## INDPRO 0.6314 -0.1888  0.2582
## PAYEMS 0.6351 -0.1468  0.2738
## SP500  0.1039  0.9335  0.3417
##
## Observation Error Covariance Matrix [diag(R) - Restricted]
##    RPI INDPRO PAYEMS  SP500
## 0.0000 0.5144 0.1550 0.0004
##
## Observation Residual Covariance Matrix [cov(resid(DFM))]
##              RPI    INDPRO    PAYEMS     SP500
## RPI      0.0000    0.0000    0.0000*   0.0000*
## INDPRO   0.0000    0.4915   -0.0711*   0.0000
## PAYEMS   0.0000*  -0.0711*   0.1169    0.0000
## SP500    0.0000*   0.0000    0.0000    0.0000
##
## Residual AR(1) Serial Correlation
##      RPI   INDPRO   PAYEMS    SP500
##  0.09498  0.28922 -0.41985  0.09488
##
## Summary of Residual AR(1) Serial Correlations
##   N    Mean  Median      SD      Min     Max
##   4  0.0148  0.0949  0.3039  -0.4198  0.2892
##
## Goodness of Fit: R-Squared
##    RPI INDPRO PAYEMS  SP500
## 1.0000 0.5085 0.8831 1.0000
##
## Summary of Individual R-Squared's
##   N    Mean  Median      SD     Min  Max
##   4  0.8479  0.9415  0.2329  0.5085    1
```

```
plot(model1)
```



It also gives you the log likelihood until convergence

```
ll <- model1$loglik
cat("LL", ll[length(ll)])
```

```
## LL -3459.03
```

If I give the output table, can you write down the equation and the estimates of parameters?

Can you experiment with different values for *p*, *q*, and *r* and compare their information criteria?