# Mathématiques actuarielles du risque – Exemples avec R

Etienne Marceau

8 novembre 2017

Mots-clés : Codes R ; Lois discrètes bivariées ; Lois composées bivariées ; Algorithme de Panjer ; FFT.

# 1 Notions de base

## 1.1 Loi exponentielle et mesures de risque

### 1.1.1 Énoncé

Le code R contient des calculs en lien avec la loi exponentielle univariée : VaR, TVaR, etc.

### 1.1.2 Code R

```
\texttt{# Lyon
}
\texttt{# A2017
}
\texttt{# Cours Vendredi 2017-11-03
}
\texttt{#
}
\texttt{# Loi exponentielle
}
\texttt{bet<-1/10
}
\texttt{vu<-(1:999)/1000
}
\texttt{VaRX<-qexp(vu,bet)
}
\texttt{EX<-1/bet
}
\texttt{vEX<-rep(EX,999)
}
\texttt{TVaRX<-VaRX+EX
}
\texttt{
matplot(vu,cbind(VaRX,vEX,TVaRX),type="l",main="VaR TVaR Loi exponentielle")
}
```

## 1.2 Loi gamma et mesures de risque

### 1.2.1 Énoncé

Le code R contient des calculs en lien avec la loi gamma univariée : VaR, TVaR, etc.

### 1.2.2 Code R

```
# Loi gamma
#
alp<-5
bet<-1/2
vu<-(1:999)/1000
VaRX<-qgamma(vu,alp,bet)
EX<-alp/bet
EX<-rep(EX,999)
TVaRX<-EX*(1-pgamma(VaRX,alp+1,bet))/(1-vu)
matplot(vu,cbind(VaRX,vEX,TVaRX),type="l",main="Loi gamma",xlab="u",ylab="VaR et TVaR")
\newpage
```

## 1.3 Loi binomiale et mesures de risque

### 1.3.1 Énoncé

Le code R contient des calculs en lien avec la loi binomiale univariée : VaR, TVaR, etc.

### 1.3.2 Code R

```
#
# Loi binomiale
#
qq<-0.0017
bb<-100000
EX<-bb*qq
EX
kap<-0.995
VaRX<-bb*qbinom(kap,1,qq)
VaRX
TVaRX<-EX/(1-kap)
TVaRX
```

```
vn<-c(1,10,100,1000,10000,100000,1000000)
vVaRN<-qbinom(kap,vn,qq)
vVaRN
vVaRS<-bb*vVaRN
vVaRS
nono<-length(vn)
vTVaRN<-rep(0,nono)
for (i in 1:nono)
{
 vk<-0:vn[i]
 partie1<-sum(vk*dbinom(vk,vn[i],qq)*1*(vk>vVaRN[i]))
 partie2<-vVaRN[i]*(pbinom(vVaRN[i],vn[i],qq)-kap)
 vTVaRN[i]<-(partie1+partie2)/(1-kap)
}
cbind(kap,vVaRN,vTVaRN)
vTVaRS<-bb*vTVaRN
round(cbind(vn,vVaRS/vn,vTVaRS/vn,TVaRX-vTVaRS/vn),2)
round(cbind(vn,vVaRS/vn,vTVaRS/vn,TVaRX-vTVaRS/vn),2)
```

## 1.4 Mutualisation des risques

### 1.4.1 Énoncé

Le code R contient des calculs en lien avec la mutualisation des risques.

### 1.4.2 Code R

```
# Aggr\U{e9}gation des co\U{fb}ts par contrat
# Loi de Xi : exponentielle(bet)
# nb de contrats : n
# Sn = co\U{fb}ts totaux pour n contrats
# Wn = co\U{fb}ts par contrat pour un ptf de n contrats
# Loi de Sn : gamma(n,bet)
# Loi de Wn : gamma(n,bet*n)
bet<-1/10
vn<-10^(0:4)
vx<-(0:100)/5
matfWn<-matrix(0,101,5)
for (i in (1:5))
{
 matfWn[,i]<-dgamma(vx,vn[i],bet*vn[i])
}
matplot(vx,matfWn,type="l",xlab="x",ylab="fonction de densit\U{e9} de Wn")
#
```

## 1.5 Générateur congruentiel linéaire

### 1.5.1 Énoncé

Le code R contient une illustration du générateur congruentiel linéaire.

### 1.5.2 Code R

```
# g\U{e9}n\U{e9}rateur de r\U{e9}alisations U(j) de la v.a. U \sim Unif(0,1)
#
aa<-41358
mm<-2^31-1
x0<-2017
nn<-1000000
vx<-rep(0,nn)
vx[1]<-(aa*x0)%%mm
for (i in 2:nn)
{
 vx[i]<-(aa*vx[i-1])%%mm
}

#cbind(1:nn,vx,vx/mm)
vU<-vx/mm
v1<-vU[1:(nn-1)]
v2<-vU[2:nn]
#plot(v1,v2)
mean(vU)
mean(qexp(vU))
mean(qgamma(vU,2,1/5))
#
#
```

## 1.6 Simulation Monte-Carlo

### 1.6.1 Énoncé

Le code R contient des calculs en lien avec la simulation Monte-Carlo.

### 1.6.2 Code R

```
# somme de 2 v.a. ind\U{e9}pendantes
#
# loi de X1: gamma(a1,bet)
# loi de X2: gamma(a2,bet)
a1<-2.5
a2<-1.5
bet<-1/10
nsim<-10^6
set.seed(2017)
matU<-matrix(runif(nsim*2),nsim,2,byrow=T)
#matU
X1<-qgamma(matU[,1],a1,bet)
X2<-qgamma(matU[,2],a2,bet)
matX<-cbind(X1,X2)
S<-X1+X2
#cbind(1:nsim,X1,X2,S)
mean(S)
mean(1*(S>50))
quantile(S,c(0.5,0.9),type=1)
EX1<-a1/bet
EX2<-a2/bet
ES<-EX1+EX2
ES
xx<-50
mean(1*(S>xx))
1-pgamma(xx,a1+a2,bet)
xx<-100
mean(1*(S>xx))
1-pgamma(xx,a1+a2,bet)
kap<-c(0.5,0.9,0.99,0.999)
quantile(S,kap,type=1)
qgamma(kap,a1+a2,bet)
kap1<-0.99999
VaRSapp<-quantile(S,kap1,type=1)
TVaRSapp<-sum(S*1*(S>VaRSapp))/nsim/(1-kap1)
VaRS<-qgamma(kap1,a1+a2,bet)
```

```
TVaRS<-ES*(1-pgamma(VaRS,a1+a2+1,bet))/(1-kap1)
c(kap1,VaRSapp,VaRS,TVaRSapp,TVaRS)
```

# 2 Modèles de bases en actuariat non-vie

## 2.1 Loi Poisson composée avec sinistre individuel de loi gamma

### 2.1.1 Énoncé

Le code R permet d'effectuer des calculs en lien avec la loi Poisson composée (avec sinistres indivuels de loi gamma) et les mesures de risque.

### 2.1.2 Code R

```
# Lyon
# A2017
# Cours Lundi 2017-11-06
# Loi Poisson compos\U{e9}e avec sinistre individuel de loi gamma
# Loi de M : Poisson
lambda=0.5
EM<-lambda
VarM<-lambda
# Loi de B : Gamma
alp<-5
bet<-1/200
EB<-alp/bet
VarB<-EB/bet
#
EX<-EM*EB
VarX<-EM*VarB+VarM*(EB^2)
EX
VarX
#
# Fonction de r\U{e9}partition de X
#
Fpoisgamma<-function(x,la,aa,bb,kmax=1000)
{
 p0<-dpois(0,la)
 vk<-1:kmax
 pk<-dpois(vk,la)
 vprob<-pgamma(x,aa*vk,bb)
 FX<-p0+sum(pk*vprob)
 return(FX)
}
Fpoisgamma(x=3200,la=lambda,aa=alp,bb=bet,kmax=1000)
vx<-(0:20)*500
```

```
long<-length(vx)
vFx<-rep(0,long)
for(i in 1:long)
{
 vFx[i]<-Fpoisgamma(x=vx[i],la=lambda,aa=alp,bb=bet,kmax=1000)
}
plot(c(0,vx),c(0,vFx),type="l",xlab="x",ylab="F(x)",main="Loi Pois Comp (B de loi gamma)")
#
# VaR et TVaR
#
# on utilise cette approche pour kappa > F_X(0)
Fpoisgamma(0,la=lambda,aa=alp,bb=bet,kmax=1000)
kappa<-0.9999
f<-function(x) abs(Fpoisgamma(x,la=lambda,aa=alp,bb=bet,kmax=1000)-kappa)
res<-optimize(f, c(0,10000),tol=0.000000001)
res
VaRX<-res$minimum
Fpoisgamma(VaRX,la=lambda,aa=alp,bb=bet,kmax=1000)
TVaRpoisgamma<-function(u,la,aa,bb,kmax=1000,bornes=c(0,10000))
{
 kappa<-u
 f<-function(x) abs(Fpoisgamma(x,la=lambda,aa=alp,bb=bet,kmax=1000)-kappa)
 res<-optimize(f, bornes,tol=0.000000001)
 VaR<-res$minimum
 vk<-1:kmax
 pk<-dpois(vk,la)
 vEtronc<-(1-pgamma(VaR,aa*vk+1,bb))*aa/bb*(vk)
 TVaR<-sum(pk*vEtronc)/(1-u)
 return(c(VaR,TVaR))
}
TVaRpoisgamma(0.9999,la=lambda,aa=alp,bb=bet,kmax=1000,bornes=c(0,10000))
```

## 2.2 Loi Binomiale négative composée (sinistres de loi lognormale)

### 2.2.1 Énoncé

Le code R permet d'effectuer des calculs en lien avec la loi binomiale négative composée. la méthode de simulation Monte Carlo et les mesures de risque.

### 2.2.2 Code R

```
#
# Loi Binomiale n\U{e9}gative compos\U{e9}e (sinistres de loi lognormale)
# loi de M: binomiale n\U{e9}gative
rr<-0.5
qq<-0.5
EM<-rr*(1-qq)/qq
VarM<-EM/qq
# Loi de B : LNormale
mu<-log(1000)-0.32
sig<-0.8
EB<-exp(mu+(sig^2)/2)
EB2<-exp(2*mu+2*(sig^2))
VarB<-EB2-(EB^2)
# X:
EX<-EM*EB
VarX<-EM*VarB+VarM*(EB^2)
EX
VarX
# Simulons :
set.seed(2017)
nsim<-100000
vM<-rep(0,nsim)
vX<-rep(0,nsim)
for(i in 1:nsim)
{
 U<-runif(1)
 vM[i]<-qnbinom(U,rr,qq)
 if (vM[i]>0)
 {
 vU<-runif(vM[i])
 vX[i]<-sum(qlnorm(vU,mu,sig))
 }
}
#cbind(vM,vX)
plot.ecdf(vX)
```

```
vkap<-(1:9999)/10000
VaRX<-quantile(vX,prob=vkap,type=1)
plot(vkap,VaRX,type="l",xlab="kappa",ylab="VaRX")
```

# 3 Méthodes d'agrégation

## 3.1 Algorithme de Panjer

### 3.1.1 Énoncé

Le code R comporte 1 exercice en lien avec l'algorithme de Panjer.

### 3.1.2 Code R

```
# Lyon
# A2017
# Cours Mardi 2017-11-06
#
# Algorithme de Panjer
#
# Fonction
#
panjer.poisson<-function(lam,ff,smax)
{
 aa<-0
 bb<-lam
 ll<-length(ff)
 ffs<-exp(lam*(ff[1]-1))
 ff<-c(ff,rep(0,smax-ll+1))
 for (i in 1 :smax)
 {
 j<-i+1
 ffs<-c(ffs,(1/(1-aa*ff[1]))*sum(ff[2 :j]*ffs[i :1]*(bb*(1 :i)/i+aa)))
 }
 return(ffs)
}
#
# Pareto
ppareto<-function(x,aa,la)
{
 FF<-1-((la/(la+x))^aa)
```

```
 return(FF)
}
ppareto(0:10,aa=2,la=5)
# Utilisation de l'algo de Panjer
#
# loi de X : Poisson compos\U{e9}e
# loi de M : Poisson
# param\U{e8}tre de la loi de Poisson : lambda
# repr\U{e9}sentation de X : X = \sum_{k=1}^M B_k
# fmp de B : fB
# fmp de X : fX
#
#
alphaP<-3
lambdaP<-20
vk<-1:10000
fB<-c(0,ppareto(vk,aa=alphaP,la=lambdaP)-ppareto(vk-1,aa=alphaP,la=lambdaP))
sum(fB)
lambda<-2
EM<-lambda
EB<-sum(fB*c(0,vk))
EB2<-sum(fB*c(0,vk^2))
EX<-EM*EB
VarX<-lambda*EB2
EM
EB
EX
VarX
#
fX<-panjer.poisson(lam=lambda,ff=fB,smax=18000)
#
# V\U{e9}rifications
sum(fX)
#
EXv<-sum((0:18000)*fX)
EX2v<-sum(((0:18000)^2)*fX)
VarXv<-EX2v-(EXv^2)
c(EX,EXv)
c(VarX,VarXv)
# FX
FX<-cumsum(fX)
plot(0:1000,FX[1:1001],type="l")
# prime stop-loss
long<-length(FX)-1
vk<-0:long
k<-100
```

```
SL<-sum(pmax(vk-k,0)*fX)
SL
#
# VaR
#
FX[1:10]
kap<-0.00001
kap
VaRX.1<-sum((FX<kap)*1)
VaRX.2<-min(vk[(FX>=kap)])
VaRX.1
VaRX.2
VaRX<-VaRX.2
EXtron<-sum(vk*fX*(vk>VaRX))
TVaRX<-(EXtron+VaRX*(FX[1+VaRX]-kap))/(1-kap)
c(kap,VaRX,TVaRX)
c(EX,VarX)
```

## 3.2 FFT

### 3.2.1 Énoncé

Le code R comporte 3 exercices simples utilisant la méthode FFT.

### 3.2.2 Code R

```
# Lyon
# A2017
# Cours Mardi 2017-11-06
#
# FFT = Transform\U{e9}e de Fourier rapide
#
# ------ Exercice de r\U{e9}chauffement - Approche na\U{ef}ve -------
f1<-c(0.3,0.4,0.2,0.1)
nbim<-1i
vk<-0:3
f1
sum(f1)
#
f1t<-rep(0,4)
# construction
for (j in 0:3)
{
```

```
 f1t[j+1]<-sum(exp(nbim*2*pi*vk*j/4)*f1)
}
f1t
f1v<-rep(0,4)
# inversion
for (k in 0:3)
{
 f1v[k+1]<-(1/4)*sum(exp(-nbim*2*pi*vk*k/4)*f1t)
}
Re(f1v)
# ----- Exercice pour s'amuser un peu -------
f1<-c(0.3,0.4,0.2,0.1)
f2<-c(0.2,0.5,0.25,0.05)
nn<-8
f1c<-c(f1,rep(0,4))
f2c<-c(f2,rep(0,4))
nbim<-1i
vk<-0:(nn-1)
f1c
sum(f1c)
f2c
sum(f2c)
#
f1t<-rep(0,nn)
f2t<-rep(0,nn)
# construction
for (j in 0:(nn-1))
{
 f1t[j+1]<-sum(exp(nbim*2*pi*vk*j/nn)*f1c)
 f2t[j+1]<-sum(exp(nbim*2*pi*vk*j/nn)*f2c)
}
fst<-f1t*f2t
cbind(f1t,f2t,fst)
fsv<-rep(0,nn)
# inversion
for (k in 0:(nn-1))
{
 fsv[k+1]<-(1/nn)*sum(exp(-nbim*2*pi*vk*k/nn)*fst)
}
fs<-rep(0,nn)
for (k in 1:nn)
{
 fs[k]<-sum(f1c[1:k]*f2c[k:1])
}
cbind(0:(nn-1),round(Re(fsv),6),fs)
2^15
```

```
# ----------- Exercice Poisson compos\U{e9}e -----------
#
# Pareto continue
ppareto<-function(x,aa,la)
{
 FF<-1-((la/(la+x))^aa)
 return(FF)
}
ppareto(0:10,aa=2,la=5)
#
# loi de X : Poisson compos\U{e9}e
# loi de M : Poisson
# param\U{e8}tre de la loi de Poisson : lambda
# repr\U{e9}sentation de X : X = \sum_{k=1}^M B_k
# fmp de B : fB
# fmp de X : fX
#
#
alphaP<-3
lambdaP<-20
vk<-1:10000
# d\U{e9}finition du vecteur fB (fonction de masses de prob de la v.a. B)
fB<-c(0,ppareto(vk,aa=alphaP,la=lambdaP)-ppareto(vk-1,aa=alphaP,la=lambdaP))
sum(fB)
# param\U{e8}tre de la loi Poisson
lambda<-2
# calculs
EM<-lambda
EB<-sum(fB*c(0,vk))
EB2<-sum(fB*c(0,vk^2))
EX<-EM*EB
VarX<-lambda*EB2
EM
EB
EX
VarX
#
# - On utilise FFT
nn<-2^15
nn
long<-length(fB)
# On ajoute des "0"
fBc<-c(fB,rep(0,nn-long))
# On utilise fft pour calculer les valeurs de la fn caract\U{e9}ristique de B
fBt<-fft(fbt)
# on calculer les valeurs de la fn caract\U{e9}ristique de X
```

```
fXt<-exp(lambda*(fBt-1))
# on inverse avec fft pour calculer les valeurs de fX
fX<-Re(fft(fXt,inverse=TRUE)/nn)
#
# V\U{e9}rifications
sum(fX)
#
EXv<-sum((0:18000)*fX)
EX2v<-sum(((0:18000)^2)*fX)
VarXv<-EX2v-(EXv^2)
c(EX,EXv)
c(VarX,VarXv)
```

# 4 Lois multivariées discrètes et composées

## 4.1 Loi Poisson bivariée de Teicher

### 4.1.1 Énoncé

Soit une paire de v.a. $(M_1, M_2)$ avec

$$P_{M_1,M_2}(t_1,t_2) = \mathrm{e}^{(\lambda_1-\alpha_0)(t_1-1)}\mathrm{e}^{(\lambda_2-\alpha_0)(t_2-1)}\mathrm{e}^{\alpha_0(t_1t_2-1)},\ |t_i| \leq 1,\ i=1,2.$$

On définit $N = M_1 + M_2$.

On déduit

$$\mathcal{P}_N(t) = \mathcal{P}_{M_1,M_2}(t,t),\ |t| \leq 1.$$

et

$$\phi_N(t) = \mathrm{e}^{(\lambda_1-\alpha_0)\left(\mathrm{e}^{it}-1\right)}\mathrm{e}^{(\lambda_2-\alpha_0)\left(\mathrm{e}^{it}-1\right)}\mathrm{e}^{\alpha_0\left(\mathrm{e}^{it\times 2}-1\right)} = \mathcal{P}_{M_1,M_2}\left(\phi_B(t), \phi_B(t)\right),$$

où $\phi_B(t) = \mathrm{e}^{it}$.

Objectif : Calculer $\Pr(N=k)$, $k \in \mathbb{N}$, avec Panjer et FFT.

### 4.1.2 Code R

```
# Lyon
# A2017
# Cours Mercredi 2017-11-08
#
# But : calculer Pr(N=k) o\U{f9} N=M1+M2
# (M1,M2) ob\U{e9}it \U{e0} une loi Poisson bivari\U{e9}e Teicher
# 2 options : FFt ou Panjer
# important : les valeurs calcul\U{e9}es sont exactes
#
# Algorithme de Panjer
#
# Fonction
#
panjer.poisson<-function(lam,ff,smax)
{
 aa<-0
 bb<-lam
 ll<-length(ff)
 ffs<-exp(lam*(ff[1]-1))
```

```
 ff<-c(ff,rep(0,smax-ll+1))
 for (i in 1 :smax)
 {
 j<-i+1
 ffs<-c(ffs,(1/(1-aa*ff[1]))*sum(ff[2 :j]*ffs[i :1]*(bb*(1 :i)/i+aa)))
 }
 return(ffs)
}
#
# Loi Poisson Bivari\U{e9}e Teicher
#
la1<-2
la2<-3
al0<-1
mm<-2^10
EN<-la1+la2
CovM1M2<-al0
VarN<-la1+la2+2*CovM1M2
# option #1 : FFT
fB<-rep(0,mm)
fB[2]<-1
fBt<-fft(fB)
fNt<-exp((la1-al0)*(fBt-1))*exp((la2-al0)*(fBt-1))*exp(al0*(fBt^2-1))
fN<-Re(fft(fNt,inverse=TRUE)/mm)
sum(fN)
vk<-0:(mm-1)
ENv<-sum(vk*fN)
EN2v<-sum((vk^2)*fN)
VarNv<-EN2v-(ENv^2)
c(EN,ENv)
c(VarN,VarNv)
# option #2 : Panjer
laN<-la1+la2-al0
fC1<-(la1+la2-2*al0)/laN
fC2<-al0/laN
fC<-c(0,fC1,fC2)
fNpanjer<-panjer.poisson(lam=laN,ff=fC,smax=1000)
sum(fNpanjer)
vk<-0:(mm-1)
ENw<-sum(vk*fN)
EN2w<-sum((vk^2)*fN)
VarNw<-EN2w-(ENw^2)
c(EN,ENv,ENw)
c(VarN,VarNv,VarNw)
round(cbind(0:30,fN[1:31],fNpanjer[1:31]),6)
```

## 4.2 Loi composée bivariée

### 4.2.1 Énoncé

Soit une paire de v.a. $(M_1, M_2)$ avec

$$\mathcal{P}_{M_1,M_2}(t_1,t_2) = (p_{00} + p_{10}t_1 + p_{01}t_2 + p_{11}t_1t_2)^n,\ |t_i| \leq 1,\ i = 1,2.$$

On définit une paire de v.a. v.a. $(X_1, X_2)$ avec

$$X_1 = \sum_{k_1=1}^{M_1} B_{1,k_1} \text{ et } X_2 = \sum_{k_2=1}^{M_2} B_{2,k_2}$$

où

- $\underline{B}_1 = \{B_{1,k_1}, k_1 \in \mathbb{N}^+\}$ forme une suite de v.a. i.i.d. avec $B_{1,k_1} \sim B_1$, $k_1 \in \mathbb{N}^+$ ;
- $\underline{B}_2 = \{B_{2,k_2}, k_2 \in \mathbb{N}^+\}$ forme une suite de v.a. i.i.d. avec $B_{2,k_1} \sim B_2$, $k_2 \in \mathbb{N}^+$ ;
- $\underline{B}_1$, $\underline{B}_2$ et $(M_1, M_2)$ ;
- $B_1 \sim Pois(\lambda)$ et $B_2 \sim BNég(r,q)$.

Expression de la f.g.p. de $(X_1, X_2)$ :

$$\mathcal{P}_{X_1,X_2}(t_1,t_2) = \mathcal{P}_{M_1,M_2}(\mathcal{P}_{B_1}(t_1), \mathcal{P}_{B_2}(t_2)),\ |t_i| \leq 1,\ i = 1,2.$$

Expression de la f.c. de $(X_1, X_2)$ :

$$\phi_{X_1,X_2}(t_1,t_2) = \mathcal{P}_{M_1,M_2}\left(\phi_{B_1}(t_1), \phi_{B_2}(t_2)\right).$$

On définit $S = X_1 + X_2$.

On déduit

$$\mathcal{P}_S(t) = \mathcal{P}_{X_1,X_2}(t,t) = \mathcal{P}_{M_1,M_2}(\mathcal{P}_{B_1}(t), \mathcal{P}_{B_2}(t)),\ |t| \leq 1.$$

et

$$\phi_S(t) = \phi_{X_1,X_2}(t,t) = \mathcal{P}_{M_1,M_2}\left(\phi_{B_1}(t), \phi_{B_2}(t)\right).$$

Objectif : Calculer $\Pr(S = k)$, $k \in \mathbb{N}$, avec FFT.

### 4.2.2 Code R

```
# Lyon
# A2017
# Cours Mercredi 2017-11-08
```

```
#
# Loi de (X_1,X_2) : binomialecompos\U{e9}e bivari\U{e9}e
# Loi de B1 : discr\U{e8}tes
# Loi de B2 : discr\U{e8}tes
# S = X_1 + X_2
# But : calculer Pr(S=k) o\U{f9} N=M1+M2
# 1 option pr\U{e9}sent\U{e9}e : FFt
# important : les valeurs calcul\U{e9}es sont exactes
#
p00<-0.7
p10<-0.15
p01<-0.05
p11<-0.1
p00+p01+p10+p11
nn<-10
q1<-p10+p11
q2<-p01+p11
q1
q2
EM1<-nn*q1
EM2<-nn*q2
mm<-2^10
vk<-0:(mm-1)
fB1<-dpois(vk,2)
fB2<-dnbinom(vk,1.5,1/3)
EB1<-2
EB2<-1.5*(1-1/3)/(1/3)
EX1<-EM1*EB1
EX2<-EM2*EB2
ES<-EX1+EX2
ES
fB1t<-fft(fB1)
fB2t<-fft(fB2)
fSt<-(p00+p10*fB1t+p01*fB2t+p11*fB1t*fB2t)^nn
fS<-Re(fft(fSt,inverse=TRUE)/mm)
sum(fS)
ESv<-sum(vk*fS)
ES
ESv
plot(0:40,fS[1:41],type="h")
```