時系列　（後半）

国際文化学部　1236573c　竹内　信吾

12.6 ARMA/ARIMAモデル

時系列データに適用されるモデルの一つ

自己回帰モデル（AR)に誤差の移動平均を加えたモデルを

自己回帰移動平均（ARMA : AutoRegressive Moving average)モデルと呼ぶ

$$y_t = \sum_{t=1}^{p} a_i y_{t-i} + e_t + \sum_{j=1}^{q} b_j e_{t-j}$$

y_tの d階の差分演算子 $\Delta^d y_t$ のARMAモデルを、自己回帰和分移動平均（ARIMA ： AutoRegressive Integrated Moving Average) モデルと呼び

ARIMA(p,d,q)で表す。

12.6.2 関数arimaとモデルの推定

パッケージstatsには、
単変量時系列データを当てはめるARIMAモデル関数arimaがある。

```
> arima
function (x, order = c(0L, 0L, 0L), seasonal = list(order = c(0L,
    0L, 0L), period = NA), xreg = NULL, include.mean = TRUE,
    transform.pars = TRUE, fixed = NULL, init = NULL, method = c("CSS-ML",
        "ML", "CSS"), n.cond, SSinit = c("Gardner1980", "Rossignol2011"),
    optim.method = "BFGS", optim.control = list(), kappa = 1e+06)
```

引数 x は、時系列データオブジェクトであり,orderは自己回帰の次数*p*
差分の階数*d* 過去の残差の移動平均の次数 q を指定する引数である。

☆　データlhを用いた、arima(2,0,1)の例

```
> (lh.ari<-arima(lh,order = c(2,0,1)))

Call:
arima(x = lh, order = c(2, 0, 1))

Coefficients:
         ar1      ar2      ma1  intercept
      1.1765  -0.5044  -0.5080     2.3946
s.e.  0.3990   0.2190   0.4517     0.0944

sigma^2 estimated as 0.1827:  log likelihood = -27.6,  aic = 65.2
```

返された係数（coefficients)を,ARIMA(2,0,1)モデルに用いると、

$$y_t = 2.3946 + 1.1765 y_{t-1} - 0.5044 y_{t-2} - 0.508 e_{t-1} + e_t$$

12.6.3 モデルの選択

関数arima(p,d,q)における引数orderのp,d,qの値を決める一つの方法は、ある範囲内のp,d,qのすべての組み合わせの中から、情報量基準（AICやBICなど）値が最も小さい組み合わせを用いる方法である。

```
> data<-lh; T<-0
> for(p in 1:4)
+  for(d in 0:1)
+   for(q in 0:4){
+    fit<-arima(data,order=c(p,d,q))
+    T<-T+1
+    if(T==1){
+      minaic<-fit$aic
+      orderP<-p; orderD<-d; orderQ<-q;
+    }else{
+     if (fit$aic<minaic){
+     minaic< -fit$aic;
+     orderP<-p; orderD<-d; orderQ<-q;
+   }
+  }
+ }
> cat("RESULT: p=",orderP,"d=",orderD,"q=",orderQ,"AIC=",minaic,"\n")
RESULT: p= 3 d= 0 q= 1 AIC= 64.75832
```

求めたp,d,qはそれぞれ3,0,1であるので

```
> (lh.ari<-arima(lh,order = c(3,0,1)))

Call:
arima(x = lh, order = c(3, 0, 1))

Coefficients:
          ar1     ar2      ar3     ma1  intercept
      -0.0608  0.4170  -0.3596  0.7758     2.3923
s.e.   0.1979  0.1415   0.1398  0.1795     0.1056

sigma^2 estimated as 0.1713:  log likelihood = -26.24,  aic = 64.47
```

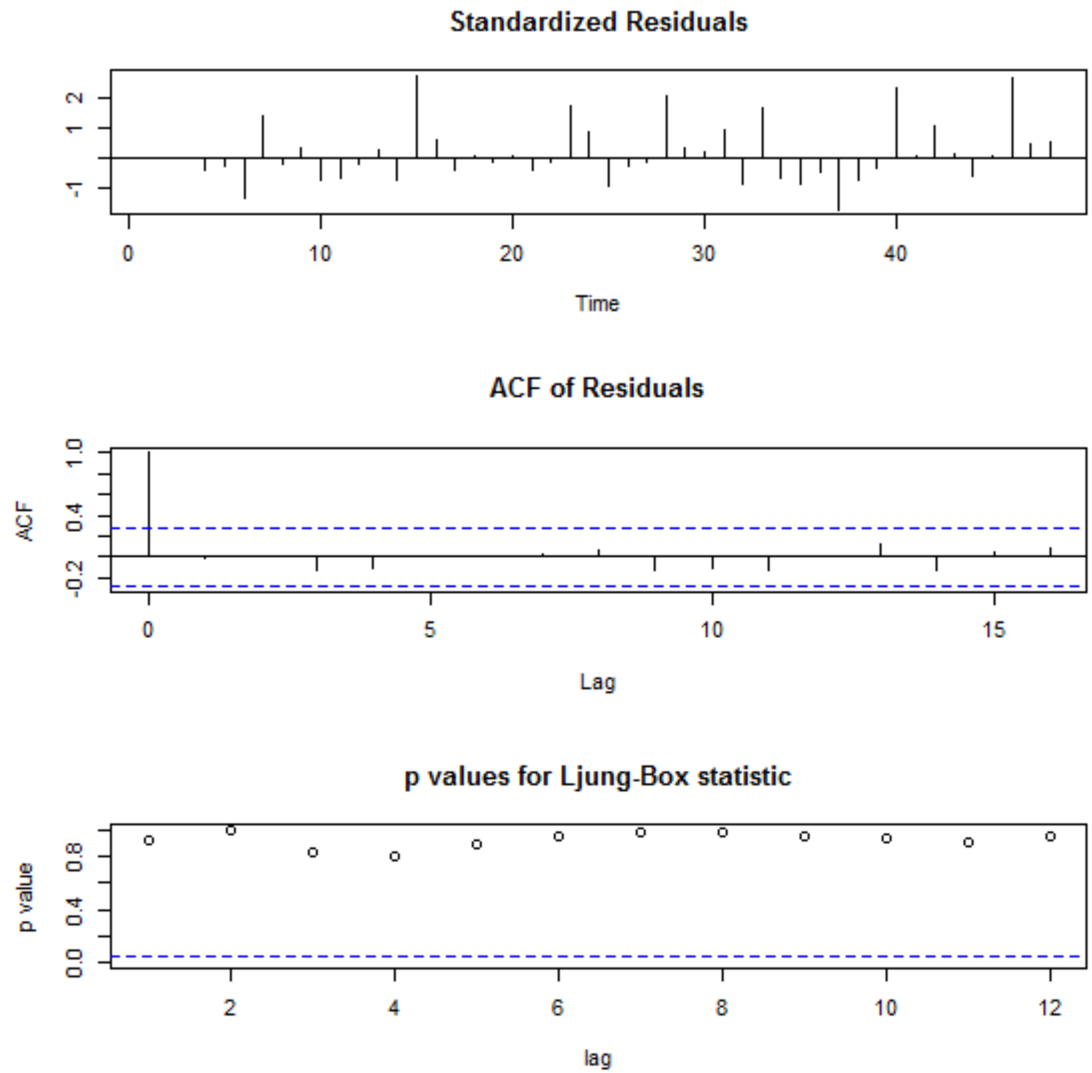
返される結果を小数点以下3桁まで丸めたARIMA(3,0,1)を示すと以下のように示される

$$y_t = 2.392 - 0.061 y_{t-1} + 0.417 y_{t-2} - 0.360 y_{t-3} + 0.776 e_t$$

☆　関数*tsdiag*

ARIMAモデルを診断(残差分析)するツールとして関数*tsdiag*がある。データlhのARIMA(3,0,1)モデルの診断図を次に示す。診断図の上部は、残差のプロット、中部は残差の自己相関のプロット、下部は引数gof.lagに対応するLjung-Box検定のp値のプロットである。

```
> tsdiag(fit,gof.lag=12)
```



12.7.1 ARIFIMAモデル

　ARIMAモデルでは過剰の差分が起こり得ると指摘されている。その短所を克服するため、差分の階数dが整数に限らず、任意の実数に一般化した方法を自己回帰実数和分移動平均（*ARFIMA: AutoRegressive Fractionally integrated moving Average*)と呼ぶ

　ARFIMA分析の専用パッケージとして*fracdiff*がある

☆航空会社の1949年から1960年までの国際旅客数に関する時系列データ

```
> (AP.d<-fdGPH(AirPassengers)$d)
[1] 0.6513916
> AP.fra<-fracdiff(AirPassengers,nar=3, dtol= AP.d,nma=1)
> summary(AP.fra)

Call:
  fracdiff(x = AirPassengers, nar = 3, nma = 1, dtol = AP.d)

Coefficients:
      Estimate Std. Error   z value Pr(>|z|)
d    4.604e-01  1.442e-01     3.192  0.00141 **
ar1 -1.159e-02  5.996e-05  -193.222  < 2e-16 ***
ar2  3.295e-01  3.103e-01     1.062  0.28826
ar3 -1.266e-01  7.667e-05 -1650.804  < 2e-16 ***
ma  -9.009e-01  1.197e-01    -7.527  5.2e-14 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
sigma[eps] = 29.69193
[d.tol = 0.1, M = 100, h = 7.322e-06]
Log likelihood: -693.9 ==> AIC = 1399.788 [6 deg.freedom]
```

12.7.2 GARCHモデル

　時系列データが、条件付き平均gt、条件付き分散htの正規分布N(gt,htに従うとき、htの変動を

$$h_t = \varpi + \sum_{i=1}^{q} \alpha_i e_{t-i}^2$$

で表現するモデルを自己回帰条件付き分散不均一(ARCH: AutoRegressive Conditional Heteroscedastic) モデルと呼び、通常ARCHモデルと呼ぶ。式の中のωは定数である。

　さらに、ARCHモデルを次のように拡張したものをGARCH(Generalized ARCH)モデルと呼ぶ。

$$h_t = \varpi + \sum_{i=1}^{q} \alpha_i e_{t-i}^2 + \sum_{j}^{r} \beta h_{t-j}$$

☆　Garchfit関数を用いた分析　パッケージfGarchを用いる

```
> UKg.d<-diff(UKgas)
> UKg.m<-garchFit(fomula=~arma(1,1)+garch(1,1),data=UKg.d,trace = F)
> summary(UKg.m)

Title:
 GARCH Modelling

Call:
 garchFit(data = UKg.d, trace = F, fomula = ~arma(1, 1) + garch(1,
    1))

Mean and Variance Equation:
 data ~ garch(1, 1)
<environment: 0x0000000009794398>
 [data = UKg.d]

Conditional Distribution:
 norm

Coefficient(s):
       mu      omega     alpha1      beta1
  8.29002  275.31909    0.44795    0.60021

Std. Errors:
 based on Hessian
```

```
Error Analysis:
        Estimate  Std. Error  t value Pr(>|t|)
mu        8.2900      6.8918    1.203  0.22902
omega   275.3191    446.4387    0.617  0.53743
alpha1    0.4479      0.1571    2.852  0.00435 **
beta1     0.6002      0.1298    4.623 3.78e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Log Likelihood:
 -684.3691    normalized:  -6.395973

Description:
 Fri Aug 08 09:43:40 2014 by user: 1236573c


Standardised Residuals Tests:
                                Statistic p-Value
 Jarque-Bera Test   R    Chi^2  7.677948  0.02151566
 Shapiro-Wilk Test  R    W      0.9282948 2.196842e-05
 Ljung-Box Test     R    Q(10)  433.5541  0
 Ljung-Box Test     R    Q(15)  586.1374  0
 Ljung-Box Test     R    Q(20)  794.2856  0
 Ljung-Box Test     R^2  Q(10)  170.0033  0
 Ljung-Box Test     R^2  Q(15)  197.6151  0
 Ljung-Box Test     R^2  Q(20)  209.9766  0
 LM Arch Test       R    TR^2   48.77365  2.291151e-06

Information Criterion Statistics:
     AIC      BIC      SIC     HQIC
12.86671 12.96663 12.86405 12.90722
```

情報量基準は@fit$ics,残差は@residuals,当てはめ値は@fitted.valuesで返すことができる

例）情報量基準*(@fit$ics)*を求める

```
> UKg.m@fit$ics
     AIC      BIC      SIC     HQIC
12.86671 12.96663 12.86405 12.90722
```

関数garchFitの結果を関数plotに代入すると、13種類のグラフを作成することができる。関数plotを実行すると、作成可能なグラフのリストを返す。「選択(Selection)：」の右にグラフの種類の番号を入力し、Enterキーを押す。

```
> plot(UKg.m)

Make a plot selection (or 0 to exit):

 1:   Time Series                        2:   Conditional SD                          3:   Series with 2 Conditional SD Superimposed
 4:   ACF of Observations                5:   ACF of Squared Observations             6:   Cross Correlation
 7:   Residuals                          8:   Conditional SDs                         9:   Standardized Residuals
10:   ACF of Standardized Residuals     11:   ACF of Squared Standardized Residuals   12:   Cross Correlation between r^2 and r
13:   QQ-Plot of Standardized Residuals

Selection: 1
```

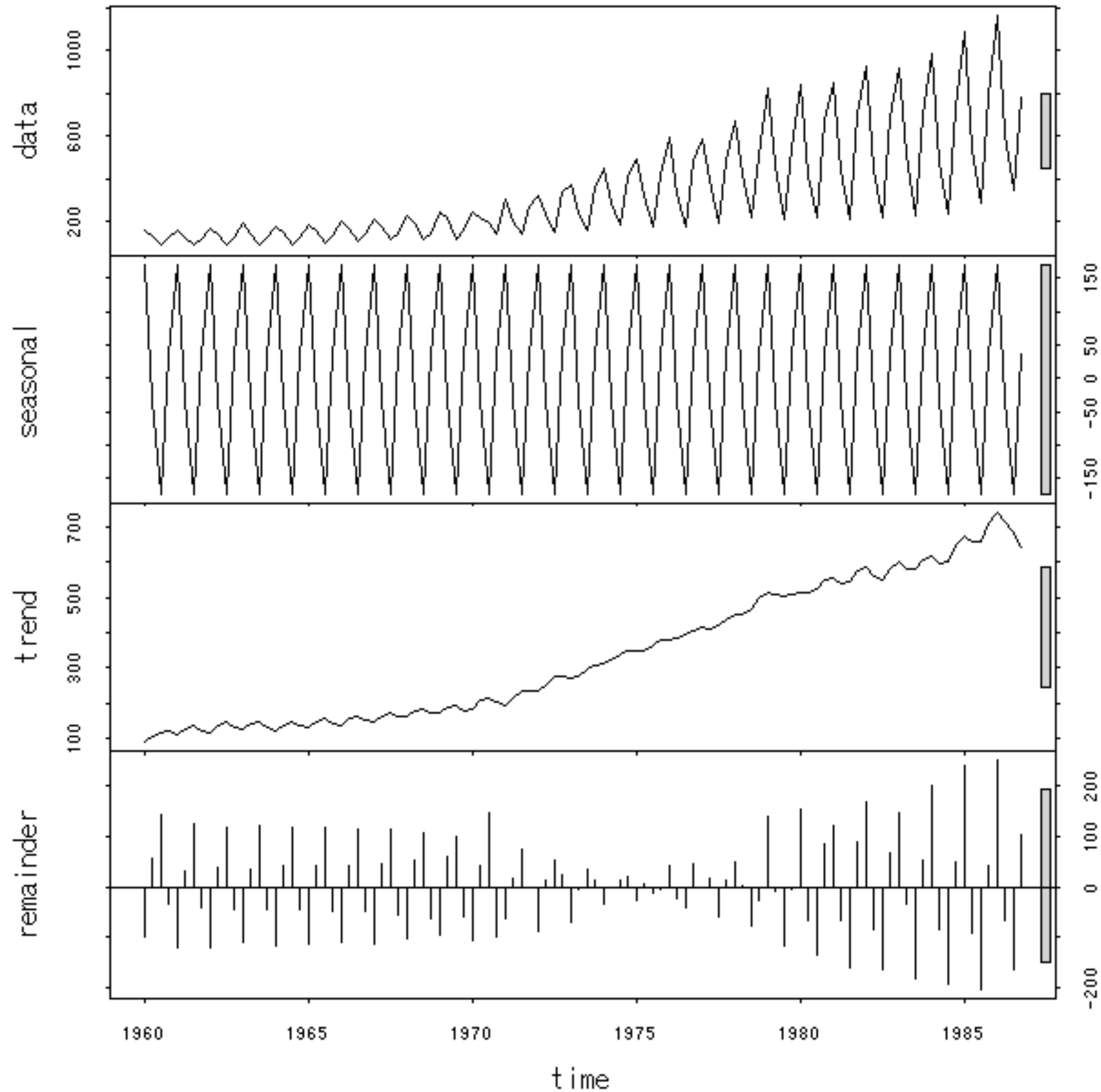
12.8　成分の分解

時系列データには、周期的に変動する周期性質を持つものも少なくない。一定の期間で周期的に変動する時系列データについて、それを幾つかの成分に分解することによって、より詳細に分析を行うことができる。

典型的な例として次のモデルがある。

観測値＝トレンド＋周期変動＋残差

```
> plot(stl(UKgas, s.window ="per"))
>
```


data
seasonal
trend
remainder
200
600
1000
150
50
0
-50
-150
100
300
500
700
200
100
0
-200
1960
1965
1970
1975
1980
1985
time

成分ごとのプロット

12.9 多変量時系列

　多変量時系列解析に関しては、最も多く知られているのはベクトル自己回帰（VAR:Vector AutoRegressive)モデルである。

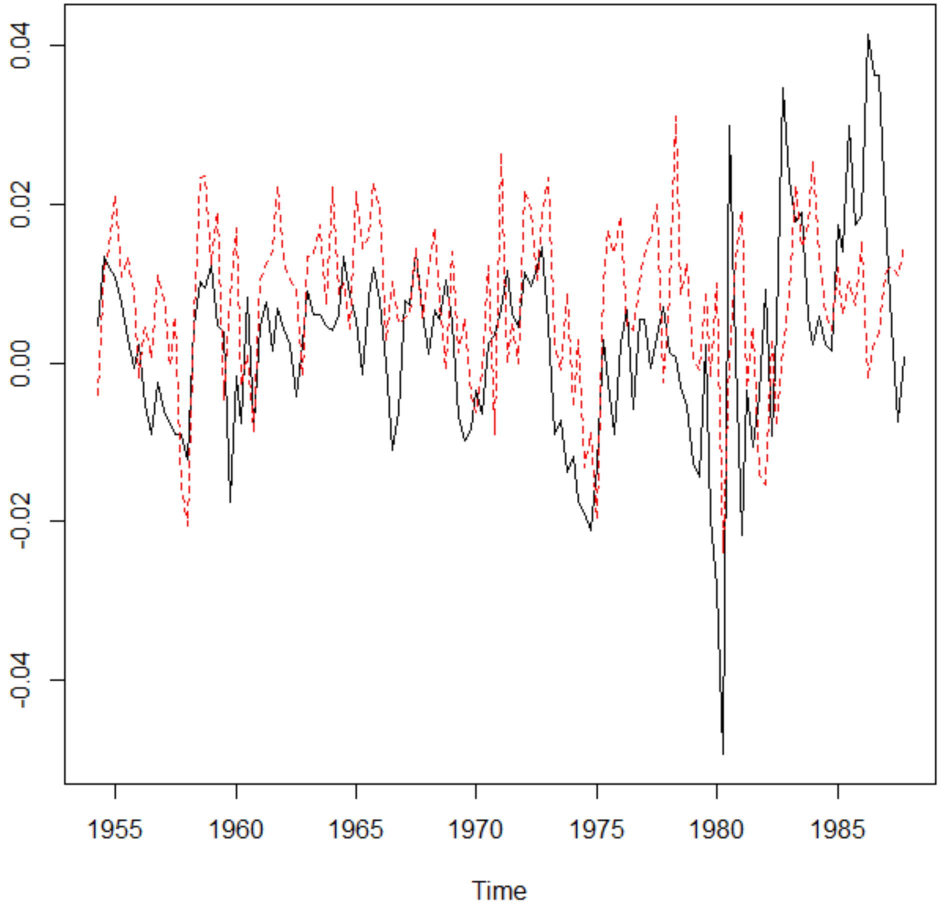
$$Y_1,\ldots,Y_{t-p},\ldots Y_t,\ldots,Y_{t+p},\ldots,Y_n$$

とすると、VARモデルは次のように定義される

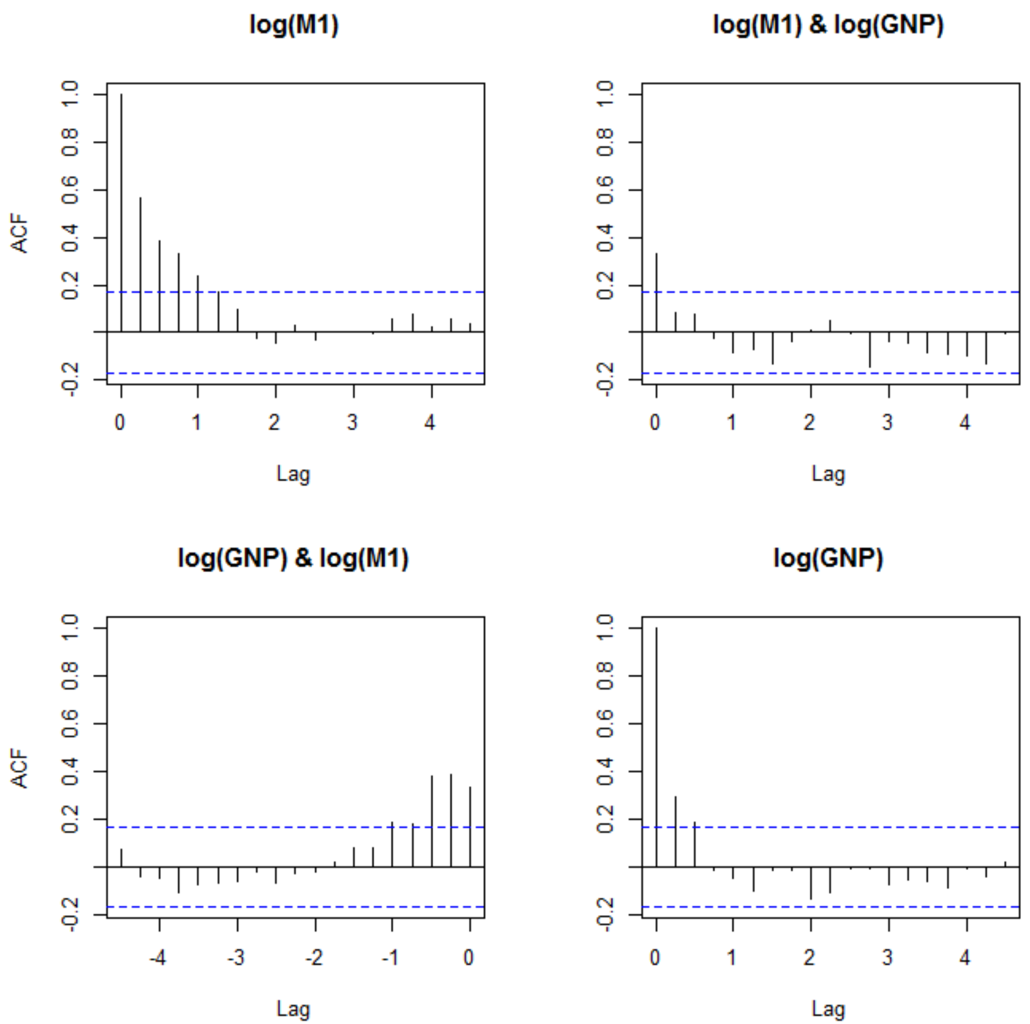
$$Y_t = A_1 Y_{t-1} + \cdot\ \cdot\ \cdot + A_p Y_{t-p} + E_t$$

☆経済時系列データUSeconomicを用いた解析

```
> library(tseries);data(USeconomic)
> USe.d<-diff(USeconomic[,1:2])
> ts.plot(USe.d,lty=c(1,2),col=c(1,2))
> legend(locator(1),c(colnaames(USe.d)[1],colnames(USe.d)[2],
```

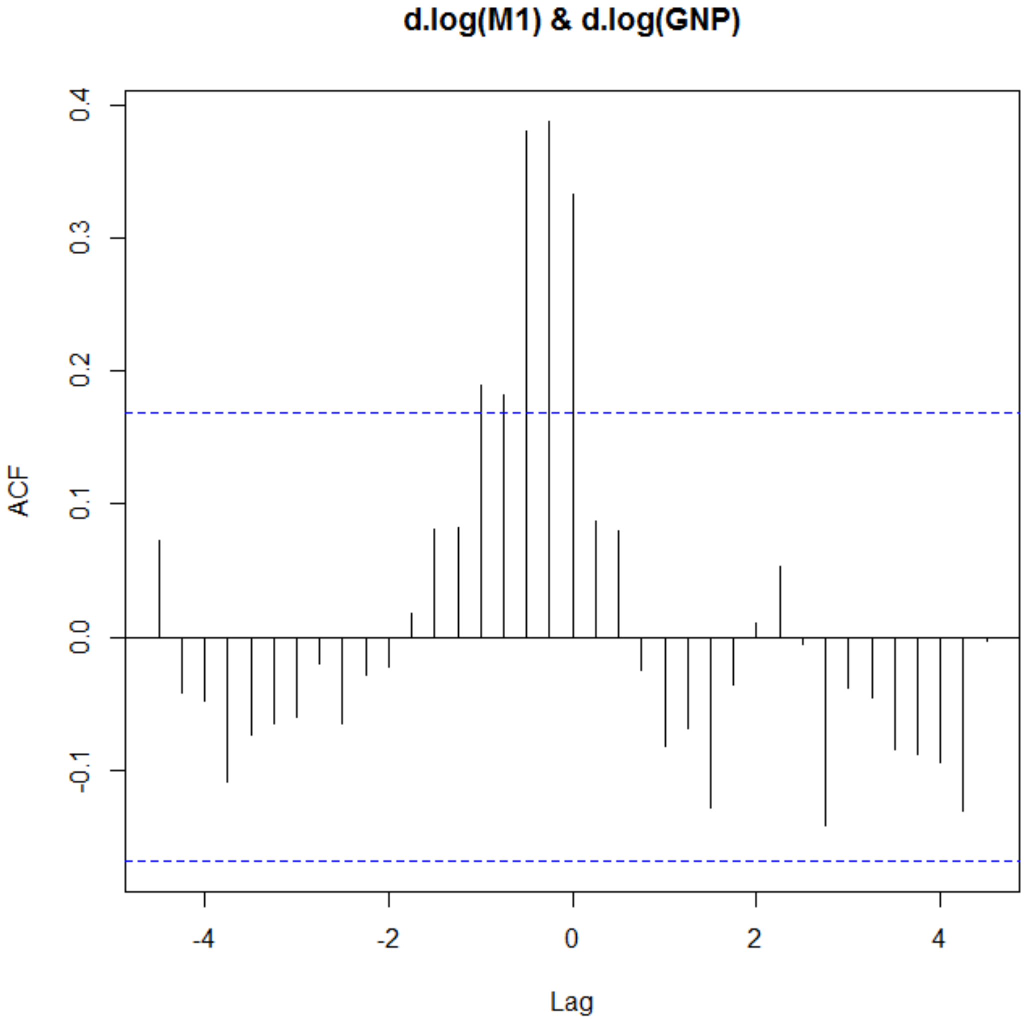


```
> acf(USe.d,na.action=na.pass)
>
```



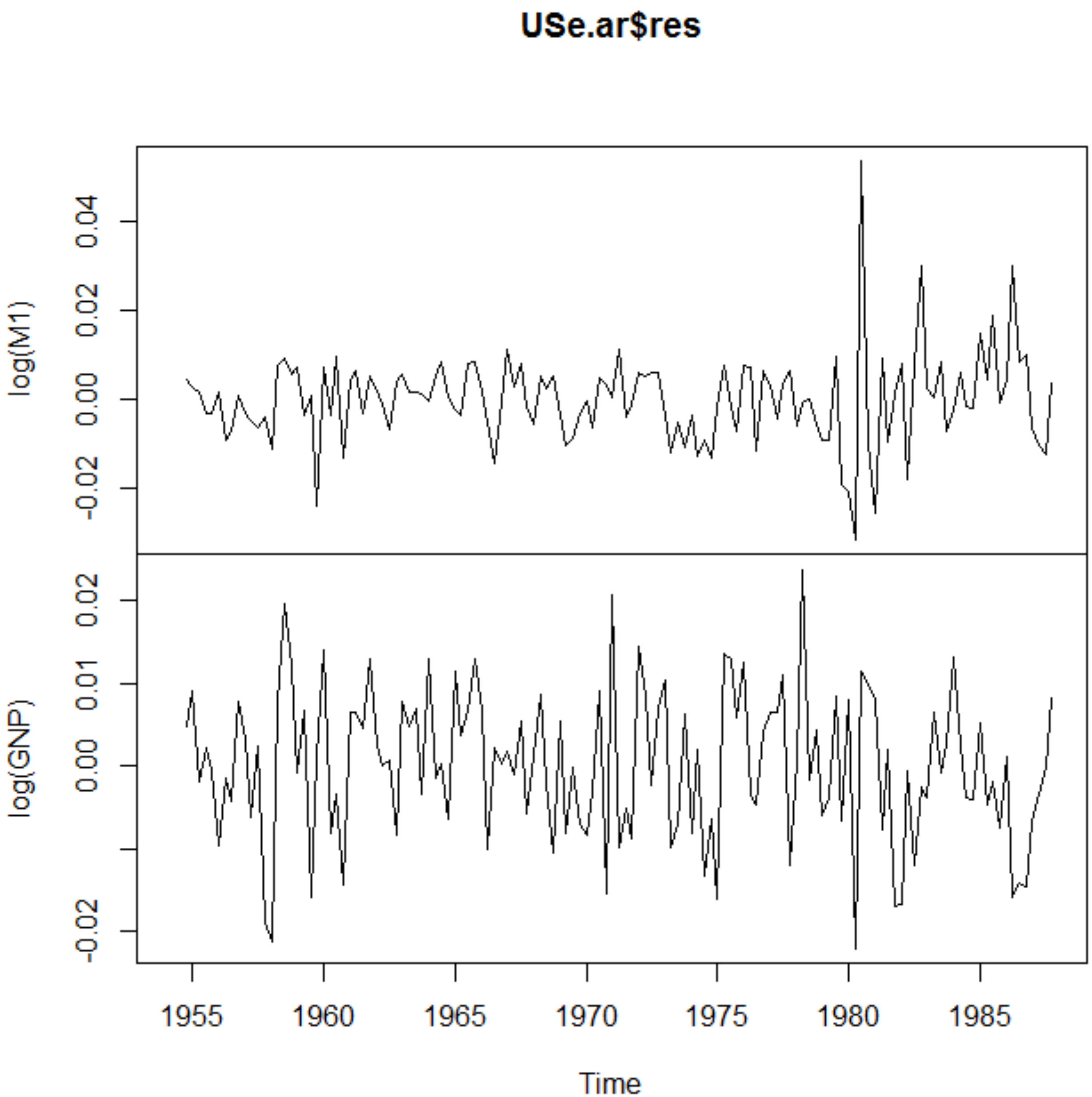
自己相関プロット

```
> ccf(USe.d[,1],USe.d[,2],main="d.log(M1) & d.log(GNP)")
> (USe.ar<-ar(USe.d,order.max=2,aic=F))
```



相互相関プロット

```
> (USe.ar<-ar(USe.d,order.max=2,aic=F))

Call:
ar(x = USe.d, aic = F, order.max = 2)

$ar
, , 1

          log(M1) log(GNP)
log(M1)    0.5446  -0.1888
log(GNP)   0.1981   0.1295

, , 2

          log(M1) log(GNP)
log(M1)    0.1231  0.04513
log(GNP)   0.1371  0.07110


$var.pred
            log(M1)   log(GNP)
log(M1)  1.063e-04 1.506e-05
log(GNP) 1.506e-05 8.494e-05
```



VAR(2)モデルの残差のプロット

12.10 カオス時系列

　不規則に変動する時系列データを非線形に解析する手法として、カオス理論に基づいた方法がある。

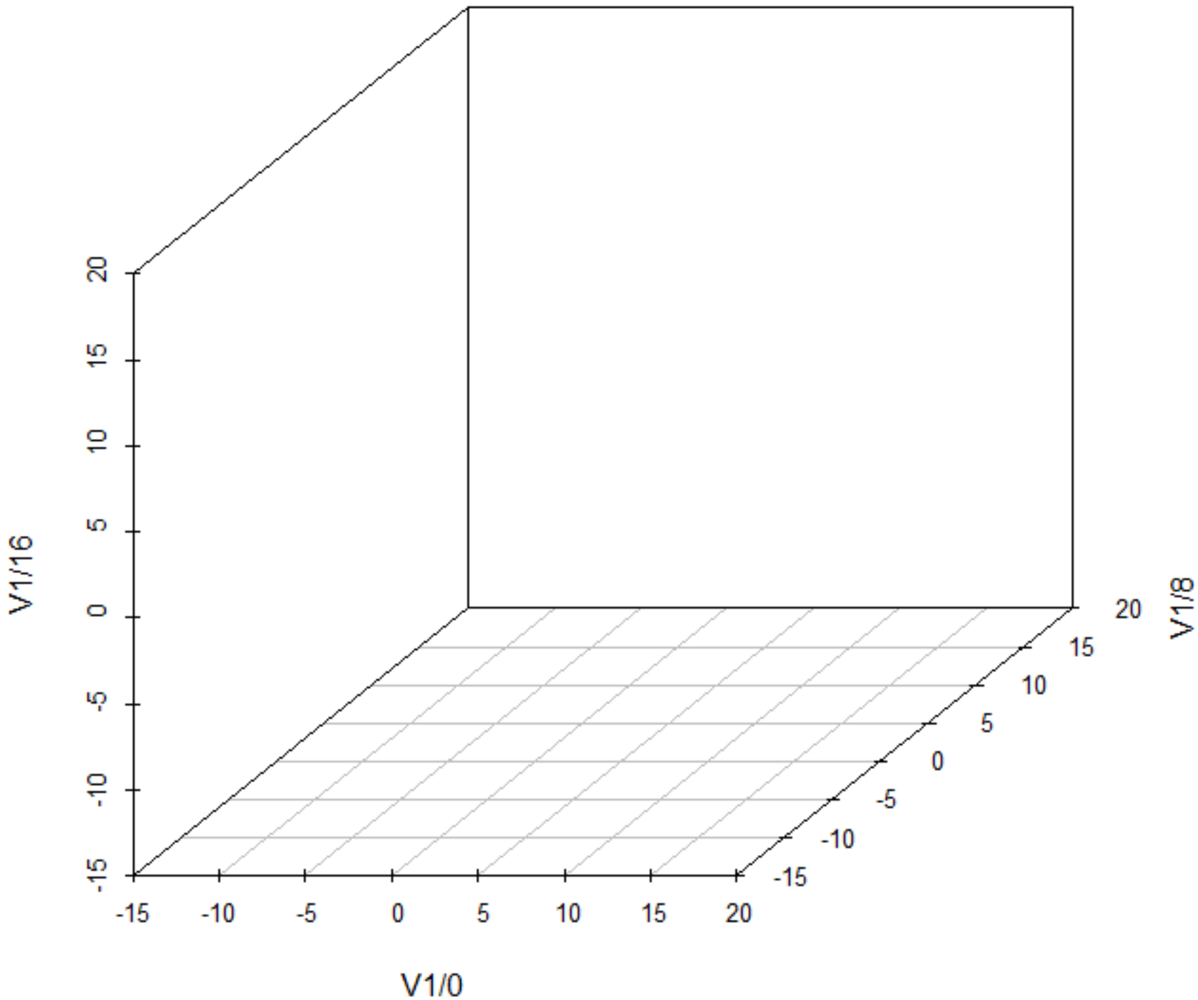
　Rには、カオス時系列解析のためのパッケージ*tseriesChaos*がある

　時系列データrossler.tsを関数emdedを用いて3次元に埋め込み(embedding),3次元空間を図示する。

　初めにパッケージ*tseriesChaos*と*Scatterplot3d*をダウンロードする

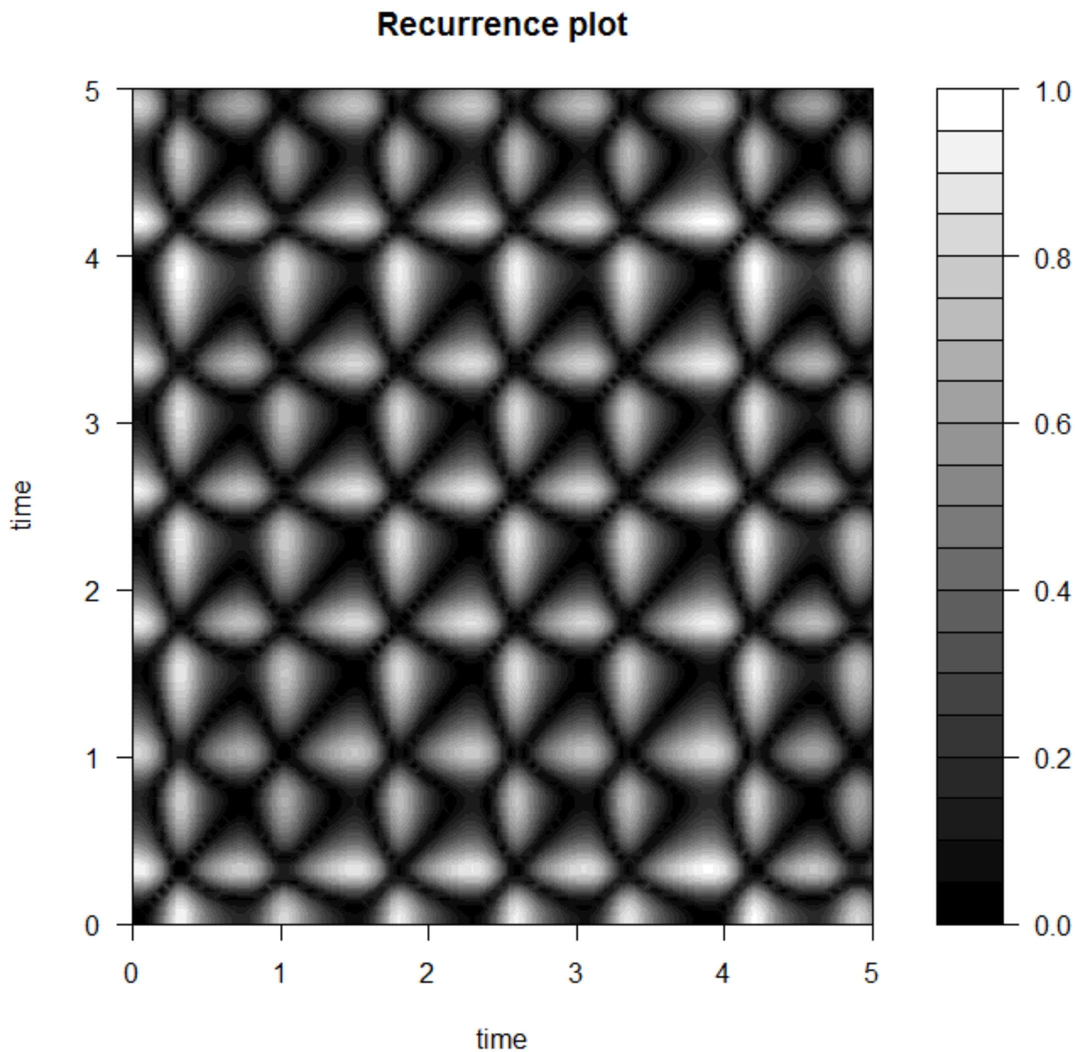
```
> install.packages("tseriesChaos"); library(tseriesChaos);
> install.packages("scatterplot3d"); library(scatterplot3d)
```

```
> library(scatterplot3d)
> recurr(lorenz.ts, m=3, d=0,start.time=10,end.time=15)
> 
```



埋め込み3次元プロット

```
> recurr(lorenz.ts, m=3, d=0,start.time=10,end.time=15)
>
```



リカレンスプロットの例