R 藤 copula 教学与分析

作者：bryanlzs 联系 qq：451843071

1.演示数据包含五个市场 2018-1-2 至 2019-12-31 日共 5 组每组 487 个时间序列

2.本文使用 Ri386 4.0.0，安装好 R 后，再安装 Rstudio 即可看到代码

4.本文部分参考资料已放在相关资料文件夹中

5.本文的相关 R 语言代码已标黄，读者可以根据自己需要微调代码，若有问题可以联系作者

最后修订日期：2020/6

目录

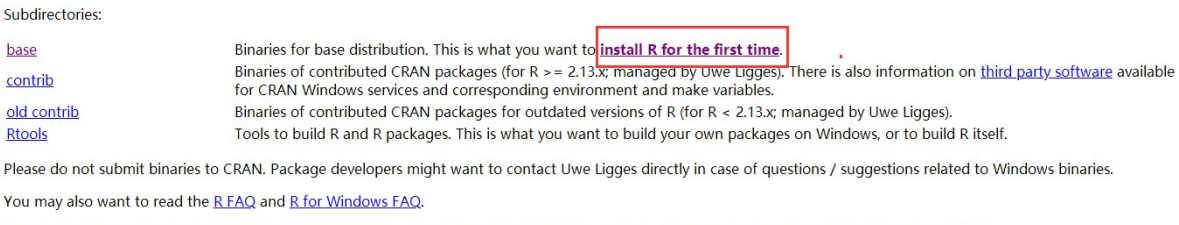
一、R studio 界面说明及准备工作

1.1 R i386 4.0.0 安装和 Rstudio 安装

• R 安装

https://mirrors.tuna.tsinghua.edu.cn/CRAN/

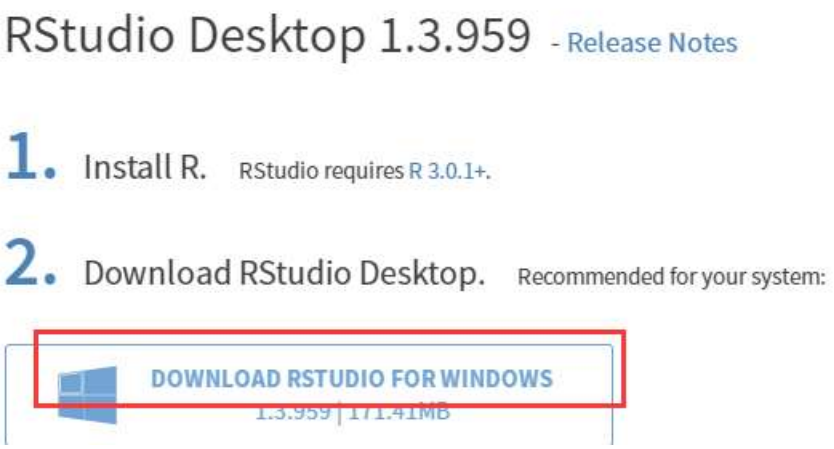
1. 

2. 

3. 

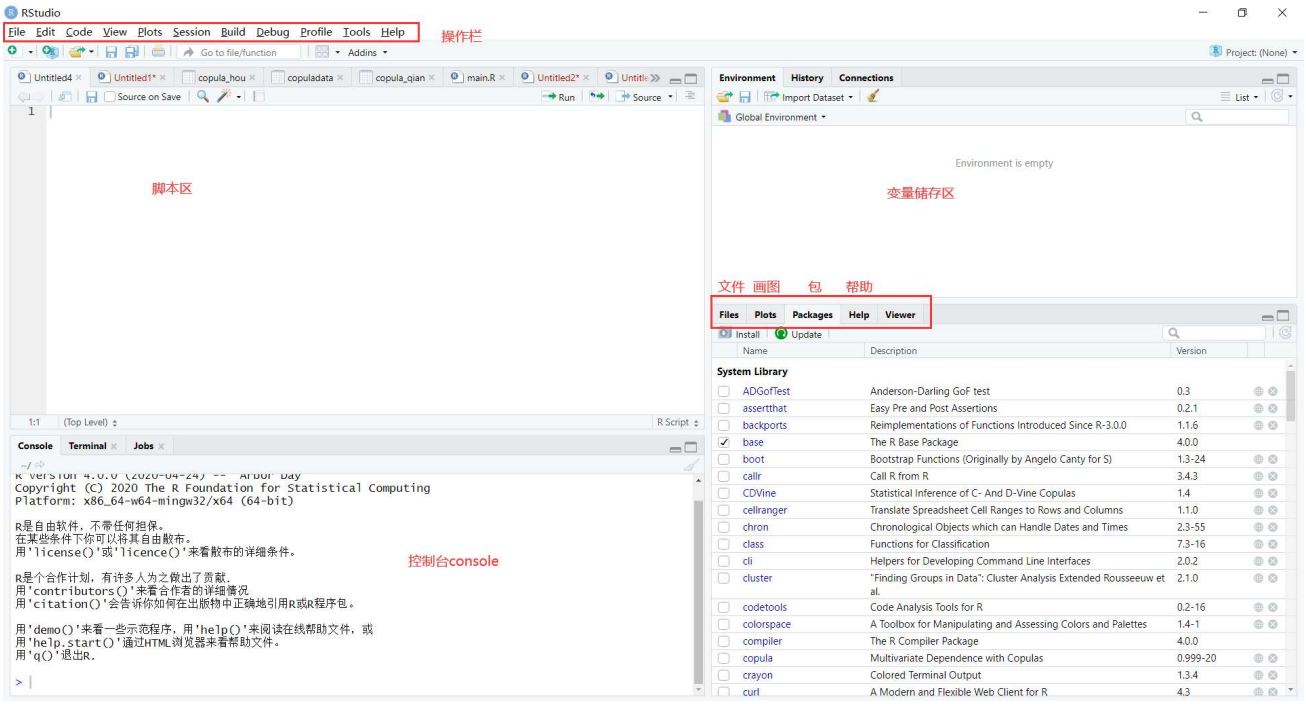
• R studio 安装

https://rstudio.com/products/rstudio/download/



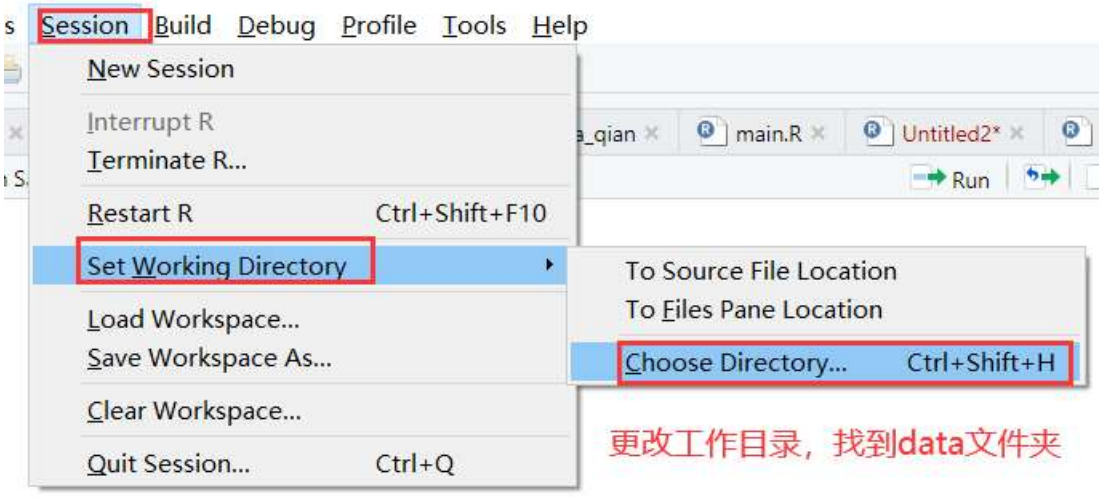
1.2 更改工作路径

R 自带的界面难以进行方便快捷的操作，因此安装 RStudio。整个界面切成多个模块进行同步操作显示，脚本区、控制台区、文件区非常清晰易用。



1.3 更改工作路径

- 这一部分非常重要，因为保证了调用文件、储存文件时不会出现错误

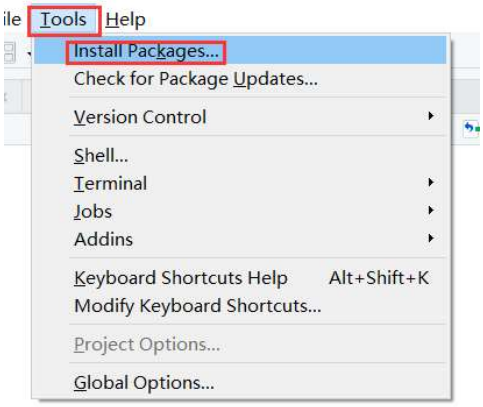


1.4 安装 R 藤 copula 所需包

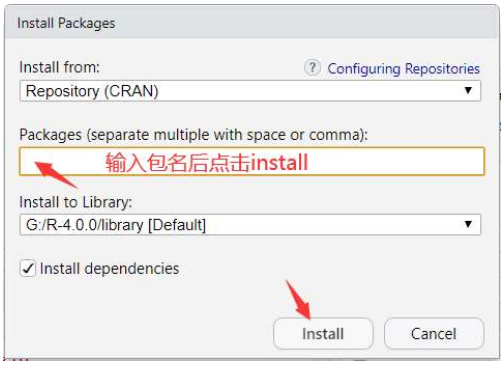
需要安装的包：readxl、tseries、zoo、forecast、rugarch、VineCopula、copula、CDVine

方式一：

- 点击 Tools 后点击 install packages



- 输入包名字后点击 install 安装



方式二：

直接在控制台输入

```
install.packages("readxl")
install.packages("tseries")
install.packages("zoo")
install.packages("forecast")
install.packages("rugarch")
install.packages("VineCopula")
install.packages("copula")
install.packages("CDVine")
```

安装好以上包后输入 library(库名)来导入已安装好的包

```
library(readxl)
library(tseries)
library(zoo)
library(forecast)
library(rugarch)
library(VineCopula)
library(copula)
```

```
library(CDVine)
```

二、数据读取及处理

数据说明：本演示数据包含六个金融市场的价格数据，首先对价格数据求收益率序列

2.1 数据读取

• 操作完 1.3 的更改工作路径以后，从 excel 导入数据，文件名为 data.xlsx

```
library(readxl)
library(tseries)
data = read_xlsx("data.xlsx")
data
```

```
> data
# A tibble: 545 x 7
   日期                  `1`   `2`   `3`   `4`   `5`   `6`
   <dttm>              <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
 1 2018-01-02 00:00:00 12871. 5289. 7648. 21845. 10079.  66.6
 2 2018-01-03 00:00:00 12978. 5331. 7671. 21905. 10116   68.0
 3 2018-01-04 00:00:00 13168. 5414. 7696. 22512. 10314.  68.1
 4 2018-01-05 00:00:00 13320. 5471. 7724. 22762. 10411.  67.8
 5 2018-01-08 00:00:00 13368. 5487. 7697. 22846. 10398.  67.9
 6 2018-01-09 00:00:00 13386. 5524. 7731. 23005. 10426.  69.2
 7 2018-01-10 00:00:00 13281. 5505. 7749. 23157. 10428.  69.1
 8 2018-01-11 00:00:00 13203. 5489. 7763. 23305. 10435.  69.1
 9 2018-01-12 00:00:00 13245. 5517. 7779. 23430. 10462.  69.8
10 2018-01-15 00:00:00 13201. 5510. 7769. 23544. 10467.  70.2
# ... with 535 more rows
```

2.2 数据处理

• 由于第一列是日期，不是数值，所以需要取后面六列数据

```
data_series = ts(data[,2:6])
```

• 生成收益率序列：

所用对数收益率公式：$r_{i,t} = \ln(p_{i,t}/p_{i,t-1})$

```
ret_series = log(data_series/lag(data_series,1L))
ret_series
```

```
> ret_series
Time Series:
Start = 1
End = 544
Frequency = 1
    data_series.1 data_series.2 data_series.3 data_series.4 data_series.5 data_series.6
  1 -8.264778e-03 -8.037799e-03 -3.004074e-03 -2.715448e-03 -3.654356e-03 -0.0212599994
  2 -1.450949e-02 -1.533957e-02 -3.223796e-03 -2.735944e-02 -1.942265e-02 -0.0011761248
  3 -1.145835e-02 -1.048479e-02 -3.675726e-03 -1.105094e-02 -9.360383e-03  0.0039749776
  4 -3.607696e-03 -3.042481e-03  3.593867e-03 -3.657259e-03  1.249411e-03 -0.0019158506
  5 -1.331421e-03 -6.633174e-03 -4.473828e-03 -6.948236e-03 -2.698694e-03 -0.0180921188
  6  7.818713e-03  3.492734e-03 -2.259759e-03 -6.604533e-03 -1.726221e-04  0.0008679300
  7  5.923540e-03  2.934535e-03 -1.860562e-03 -6.363839e-03 -6.614423e-04 -0.0001447073
  8 -3.185885e-03 -5.181006e-03 -2.020387e-03 -5.330911e-03 -2.603171e-03 -0.0100778279
  9  3.366923e-03  1.336749e-03  1.222040e-03 -4.841901e-03 -4.586805e-04 -0.0058559048
 10 -3.465068e-03 -7.493077e-04  1.701764e-03  2.052348e-03 -5.069671e-03  0.0128997059
 11  4.719594e-03  3.602900e-03  3.940227e-03 -8.249318e-04  4.362950e-03 -0.0018736044
 12 -7.365880e-03 -1.528827e-04  3.172488e-03 -4.926887e-03  4.008175e-03  0.0098394659
 13 -1.145548e-02 -5.748863e-03 -3.866060e-03 -4.998319e-03 -4.475864e-03  0.0072971719
 14 -2.174129e-03 -2.797129e-03  1.987541e-03 -5.917808e-03 -9.922459e-03 -0.0086050265
 15 -7.098351e-03  1.215103e-03 -2.122059e-03  2.245281e-03 -2.406399e-03 -0.0080995525
 16  1.074068e-02  7.270834e-03  1.149912e-02  9.017487e-03  4.392497e-03 -0.0120276364
 17  8.713383e-03  2.541825e-03  3.616166e-03 -4.105836e-03 -3.053178e-03  0.0095817680
 18 -3.139065e-03 -8.708217e-03 -6.504671e-03 -5.766692e-03 -9.438103e-06 -0.0050168528
 19  1.176839e-03  1.368234e-03 -7.811140e-04  2.326552e-03  3.763420e-03  0.0112150708
 20  9.559615e-03  8.696442e-03  1.095066e-02  1.356253e-02  1.214285e-02  0.0152986156
 21  6.237875e-04 -1.487809e-03  7.199038e-03 -1.112625e-03 -2.231834e-03 -0.0112418149
 22  1.417023e-02  5.007106e-03  5.745513e-03 -1.462320e-03  5.035861e-03 -0.0129815756
```

2.3 数据提取

• 将六组收益率序列分别提取，命名为 m1,m2,m3,m4,m5,m6：

```
m1 = as.ts(ret_series[,1])
m2 = as.ts(ret_series[,2])
m3 = as.ts(ret_series[,3])
m4 = as.ts(ret_series[,4])
m5 = as.ts(ret_series[,5])
```

三、描述性统计和数据检验

3.1 单位根检验

```
adf.test(m1)
adf.test(m2)
adf.test(m3)
adf.test(m4)
adf.test(m5)
```

```
> adf.test(m1)

        Augmented Dickey-Fuller Test

data:  m1
Dickey-Fuller = -8.4519, Lag order = 7, p-value = 0.01
alternative hypothesis: stationary
```

单位根检验显著，说明序列是平稳序列

（省略剩余五个市场的结果）

单位根检验是指检验序列中是否存在单位根，因为存在单位根就是非平稳时间序列了。当 p 值显著，说明序列是平稳序列。若序列中存在单位根过程就不平稳，会使回归分析中存在伪回归。

3.2 Jarque-Bera 检验

JB 统计量全称叫 Jarque-Bera 统计量，用于检验样本数据是否服从正态分布，若变量越服从正态分布，则 JB 统计量越接近 0，若变量不是正态，则 JB 统计量值较大。如果 JB 统计量值较大，比如为 11，则可以计算出卡方值大于 11 的概率为 0.004，这个概率过小，因此不能认为样本来自正态分布。反之，成立。

```
jarque.bera.test(m1)
jarque.bera.test(m2)
jarque.bera.test(m3)
jarque.bera.test(m4)
jarque.bera.test(m5)
```

```
> jarque.bera.test(m1)

        Jarque Bera Test

data:  m1
X-squared = 108.27, df = 2, p-value < 2.2e-16
```

JB统计量为108.27，p值非常显著，说明序列不服从正态分布

（省略剩余五个市场的结果）

3.3 Ljung-Box test 检验

Ljung-Box test 是对序列纯随机性进行检验，或者可以说是对时间序列是否存在滞后相关的检验。

LBQ检验的原假设和备择假设分别为：
H0: 原本的数据都是独立的，即总体的相关系数为0，能观察到的某些相关仅仅产生于随机抽样的误差。即 $\hat{\rho}_1 = \hat{\rho}_2 = \cdots = \hat{\rho}_m = 0$，其中m是人为给定的，有时我们在软件中仅仅给定一个上界，而不是具体的m。
H1: 原本的数据不是独立的，即至少存在某个 $\hat{\rho}_k \neq 0$，其中 $k \leqslant m$。
构造的统计量是

$$Q(m) = T(T+2)\sum_{i=1}^{m}\frac{\hat{\rho}_i}{T-i}$$

其中T是样本容量，m是人为选定的一个数，$\hat{\rho}_i$是i阶滞后的自相关系数。
在原假设成立的条件下，Q(m)服从自由度为m的卡方分布。给定显著性水平α，则拒绝域是$Q > \chi^2_{1-\alpha,m}$。接受原假设意味着认为原序列是白噪声序列，否则认为序列存在相关性。

LBQ 还用于在**拟合时间序列模型（例如 ARIMA）后评估假设**以确保残差彼此独立。
对ARMA(p,q)模型的平方残差应用Ljung-Box统计量检查模型的不足，这个统计量是

$$Q(m) = T(T+2)\sum_{i=1}^{m}\frac{\hat{\rho}_i(a_t^2)}{T-i}$$

其中T是样本容量，m是人为选定的一个数，a_t是残差，而$\hat{\rho}_i(a_t^2)$是a_t^2的i阶自相关函数（ACF）。
这个统计量的原假设和备择假设为：
H0：$\beta_1 = \beta_2 = \cdots = \beta_m = 0$，其中$\beta_i$是下面的线性回归的$a_{t-i}^2$的系数：

$$a_t^2 = \beta_0 + \beta_1 a_{t-1}^2 + \cdots + \beta_m a_{t-m}^2 + e_t$$

其中$t = m+1, \cdots, T$. 因为这个统计量由残差计算得到（而不是直接观察得到），所以自由度是m-p-q.
H1：至少存在某个$\beta_k \neq 0$，其中$k \leqslant m$.

以上总结为两点：

• 对序列的随机性检验，若 p 值显著（小于 0.05），说明序列不是随机序列，即时间序列存在滞后相关。（本小节的检验）

• 检验一段序列是否是独立观测值，如果不是独立时间序列，则存在自相关，会导致 ARIMA 模型的不准确（后面构建边缘分布时，要用 LB 检验来判断是否合理）

```
Box.test (m1, lag = 20, type = "Ljung")
Box.test (m2, lag = 20, type = "Ljung")
Box.test (m3, lag = 20, type = "Ljung")
Box.test (m4, lag = 20, type = "Ljung")
Box.test (m5, lag = 20, type = "Ljung")
```

调参：lags（滞后阶数，此处意为检测直至 20 阶是否存在滞后相关）

```
> Box.test (m1, lag = 20, type = "Ljung")

        Box-Ljung test

data:  m1
X-squared = 32.946, df = 20, p-value = 0.03421
```

LB统计量为32.946，p值小于0.05，说明序列不是随机序列，存在滞后相关

（省略剩余五个市场的结果）

3.4 ARCH-LM 检验（自回归条件异方差）

ARCH-LM 在检验前后使用的目的是不同的，使用 ARCH 模型之前检验，是为了判断是否存在 ARCH 效应，使用 ARCH 模型之后再用该检验是为了判断 ARCH 模型是否消除了自回归条件异方差的影响。

本部分在使用 ARCH 模型之前对六组市场时间序列进行检验，为了判断原序列是否存在 ARCH 效应，统一展示在统计性描述中。

H0:残差序列中直到 P 阶不存在 ARCH 效应

```
library(FinTS)
ArchTest (m1, lags=20, demean = FALSE)
ArchTest (m2, lags=20, demean = FALSE)
ArchTest (m3, lags=20, demean = FALSE)
ArchTest (m4, lags=20, demean = FALSE)
ArchTest (m5, lags=20, demean = FALSE)
```

调参：lags（滞后阶数，此处意为检测直至 20 阶是否存在 ARCH 效应）

```
> ArchTest (m1, lags=20, demean = FALSE)

        ARCH LM-test; Null hypothesis: no ARCH effects

data:  m1
Chi-squared = 41.405, df = 20, p-value = 0.003304
```

ARCH-LM检验统计量为41.405，p值显著，说明序列在滞后20阶内存在ARCH效应

（省略剩余五个市场的结果）

3.5 相关系数

此处使用 Pearson 相关系数对收益率序列求相关系数矩阵

```
cor(ret_series)
```

```
> cor(ret_series)
              data_series.1 data_series.2 data_series.3 data_series.4 data_series.5
data_series.1     1.0000000     0.7063635     0.6764992     0.8530357     0.9958280
data_series.2     0.7063635     1.0000000     0.9181372     0.6885465     0.6394151
data_series.3     0.6764992     0.9181372     1.0000000     0.6378401     0.6174259
data_series.4     0.8530357     0.6885465     0.6378401     1.0000000     0.8401338
data_series.5     0.9958280     0.6394151     0.6174259     0.8401338     1.0000000
```

3.6 统计性描述

```
library(pastecs)
stat.desc(ret_series,norm = TRUE)
```

```
> stat.desc(ret_series,norm = TRUE)
             data_series.1 data_series.2 data_series.3 data_series.4 data_series.5
nbr.val       4.860000e+02  4.860000e+02  4.860000e+02  4.860000e+02  4.860000e+02
nbr.null      0.000000e+00  0.000000e+00  4.000000e+00  2.000000e+00  0.000000e+00
nbr.na        0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
min          -2.615090e-02 -2.499553e-02 -2.963641e-02 -2.591631e-02 -2.901993e-02
max           2.976246e-02  4.807534e-02  4.106662e-02  5.512779e-02  3.131994e-02
range         5.591336e-02  7.307086e-02  7.070304e-02  8.104411e-02  6.033987e-02
sum          -8.828842e-02  2.736943e-02 -3.003799e-02 -7.650407e-02 -1.013815e-01
median       -6.563291e-04 -5.822825e-04 -7.464651e-04 -8.661101e-04 -6.951038e-04
mean         -1.816634e-04  5.631569e-05 -6.180656e-05 -1.574158e-04 -2.086039e-04
SE.mean       3.273341e-04  3.917407e-04  4.841113e-04  4.429293e-04  3.343075e-04
CI.mean.0.95  6.431682e-04  7.697185e-04  9.512144e-04  8.702973e-04  6.568699e-04
var           5.207375e-05  7.458194e-05  1.139008e-04  9.534659e-05  5.431610e-05
std.dev       7.216215e-03  8.636083e-03  1.067243e-02  9.764558e-03  7.369946e-03
coef.var     -3.972299e+01  1.533513e+02 -1.726747e+02 -6.203036e+01 -3.532985e+01
skewness      6.543439e-01  6.056808e-01  1.594434e-01  7.079712e-01  6.765880e-01
skew.2SE      2.953609e+00  2.733951e+00  7.197032e-01  3.195675e+00  3.054016e+00
kurtosis      1.883390e+00  2.210648e+00  4.373129e-01  2.259093e+00  2.236574e+00
kurt.2SE      4.259280e+00  4.999373e+00  9.889817e-01  5.108930e+00  5.058005e+00
normtest.W    9.597786e-01  9.759515e-01  9.939049e-01  9.707778e-01  9.533708e-01
normtest.p    2.962688e-10  3.567975e-07  4.821861e-02  2.890813e-08  2.929309e-11
```

• 此处将上述所有的结果汇总至三线表中

	m1	m2	m3	m4	m5
mean	-0.0002	0.0001	-0.0001	-0.0002	-0.0002
Minimum	-0.0262	-0.0250	-0.0296	-0.0259	-0.0290
Maximum	0.0298	0.0481	0.0411	0.0551	0.0313
Std. Dev.	0.0072	0.0086	0.0107	0.0098	0.0074
Skewness	0.6543	0.6057	0.1594	0.7080	0.6766
kurtosis	1.8834	2.2106	0.4373	2.2591	2.2366
J-B	108.27	130.8	6.2	146.19	140.57
	[0.000]	[0.000]	[0.045]	[0.000]	[0.000]
Q(20)	32.946	29.818	25.17	28.403	28.621
	[0.034]	[0.073]	[0.195]	[0.100]	[0.095]
ARCH-LM	41.405	32.611	22.301	26.717	41.106
	[0.003]	[0.037]	[0.324]	[0.143]	[0.004]

注： ARCH-LM 计算 20 阶内的 ARCH 效应。Q(20) 是 Ljung-Box 计算滞后 20 阶序列自相关性统计量。方括号内是 p 值

3.6 绘制自相关和偏自相关图

```
m1_acf = acf(m1,plot = T)
m1_pacf = pacf(m1,plot = T)
m2_acf = acf(m2,plot = T)
m2_pacf = pacf(m2,plot = T)
m3_acf = acf(m3,plot = T)
m3_pacf = pacf(m3,plot = T)
m4_acf = acf(m4,plot = T)
m4_pacf = pacf(m4,plot = T)
m5_acf = acf(m5,plot = T)
m5_pacf = pacf(m5,plot = T)
```

四、Copula 理论知识讲解

Copula 理论：Sklar 指出，可以将联合分布分解为 k 个边缘分布和一个 Copula 函数，这个 Copula 函数描述了变量之间的非线性相关性。因此，Copula 函数实际上是将一类联合分布函数与他们各自的边缘分布函数连接在一起的函数。

这个函数有很多种类型。椭圆 Copula 函数簇（t copula、Gaussian copula）、阿基米德 Copula 函数簇（Gumbel Copula、Clayton Copula、FrankCopula）等等。

4.1 基于 copula 函数的相关性测度

在描述 copula 函数之前，首先要了解描述 copula 函数的相关系数。

4.1.1Kendall'秩相关系数τ

考查两个变量之间的相关性时，最简单的方式是考查他们变化的趋势是否趋于一致，如果一致，则存在正相关，如果反则存在负相关，由此建立了相关性测度的联系。

Hollander 和 Wolfe(1973)、Lehmann(1975)给出了 Kendall'秩相关系数的定义：由两组随机变量(X,Y)构成 N 组观测值，将所有观测值的两两组合遍历(x_i,y_i)和(x_j,y_j)，分为两组，c 表示变化一致组合的数量，d 表示变化不一致组合的数量。

$$\tau=\frac{c-d}{c+d}=\frac{c-d}{C_N^2}$$

4.1.2 尾部相关系数

在测量不同金融市场的风险的关系中，尾部相关性更有意义，并且这一特性用 Copula 函数来处理非常便捷，对于尾部相关系数的定义如下：

定义 1-12 令 X、Y 为两个连续的随机变量，具有边缘分布 $F(\cdot)$、$G(\cdot)$和 Copula 函数 $C(\cdot,\cdot)$，分别定义

$$\lambda^{up}\equiv\lim_{u^*\to 1}P[Y>G^{-1}(u^*)\mid X>F^{-1}(u^*)]=\lim_{u^*\to 1}\frac{\hat{C}(1-u^*,1-u^*)}{1-u^*}\quad(1\text{-}32)$$

$$\lambda^{lo}\equiv\lim_{u^*\to 0}P[Y<G^{-1}(u^*)\mid X<F^{-1}(u^*)]=\lim_{u^*\to 0}\frac{C(u^*,u^*)}{u^*}\quad(1\text{-}33)$$

为上尾相关系数和下尾相关系数[12]。

若 λ^{up}（或 λ^{lo}）存在且在区间(0,1]内，则随机变量 X、Y 上尾（或下尾）相关；若 λ^{up}（或 λ^{lo}）等于零，则随机变量 X、Y 独立。

可见，用基于 Copula 函数的尾部相关系数来分析金融市场或金融资产之间的尾部相关性非常方便。例如它可以直观的反映一支股票价格的暴跌是否会引起另一支股票价格的暴跌，或一个股票市场的大波动是否会引起其他股票市场的大波动，这对于金融市场的波动溢出分析是极为有用的。

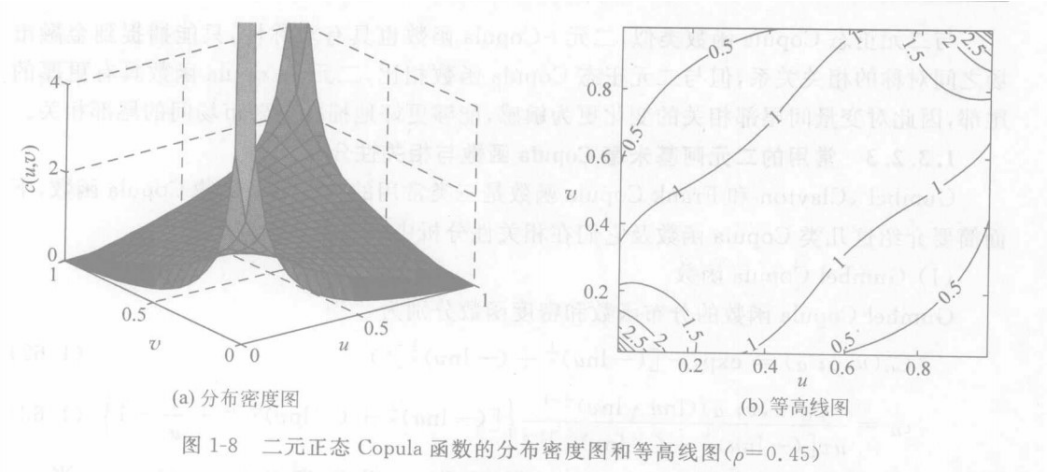
可以理解为，上尾相关性可以理解为一个金融市场暴涨会引起另一个金融市场的暴涨的相关性；下尾相关性理解为一个金融市场暴跌会引起另一个金融市场暴跌的相关性。记λ_L为下尾相关系数，λ_U为上尾相关系数。

4.2 copula 函数

4.2.1 正态 copula（Gaussian copula）

分布函数：$\mathrm{C}_N(u,v|\rho)=\Phi(\Phi^{-1}(u),\Phi^{-1}(v))$

上下尾相关性：没有上下尾相关性 $\lambda_U=\lambda_L=0$



(a) 分布密度图　　(b) 等高线图

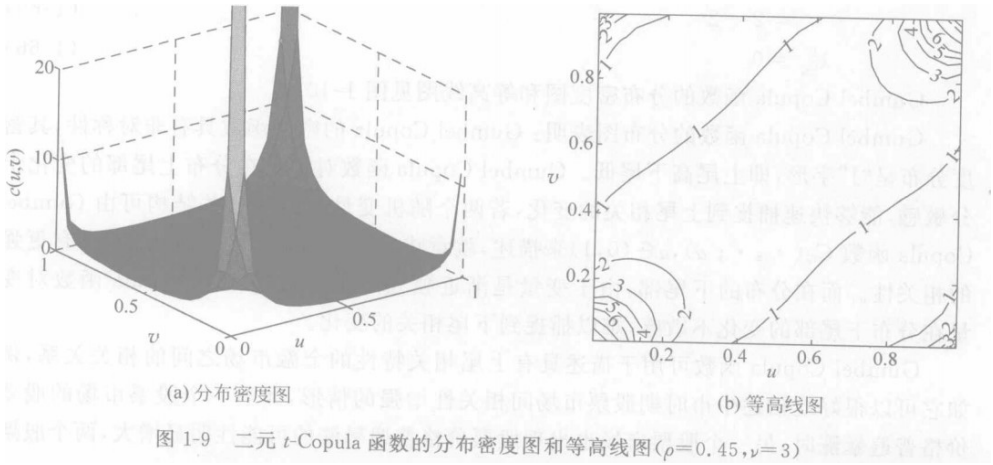
图 1-8　二元正态 Copula 函数的分布密度图和等高线图(ρ=0.45)

二元正态 copula 函数可以很好拟合样本数据，很好描述变量的相关性，但因为二元正态 copula 函数具有对称性质，所以无法捕捉市场的非对称相关关系。

4.2.2 t-copula

分布函数：$\mathrm{C}_{ST}(u,v|\rho,v)=\mathrm{T}\left(\mathrm{t}_v{}^{-1}(u),\mathrm{t}_v{}^{-1}(v)\right)$

上下尾相关性：对称上下尾相关性 $\lambda_U=\lambda_L=2\mathrm{t}_{v+1}(-\sqrt{v+1}\sqrt{1-\rho}/\sqrt{1+\rho})$



(a) 分布密度图　　(b) 等高线图

图 1-9　二元 t-Copula 函数的分布密度图和等高线图(ρ=0.45,ν=3)

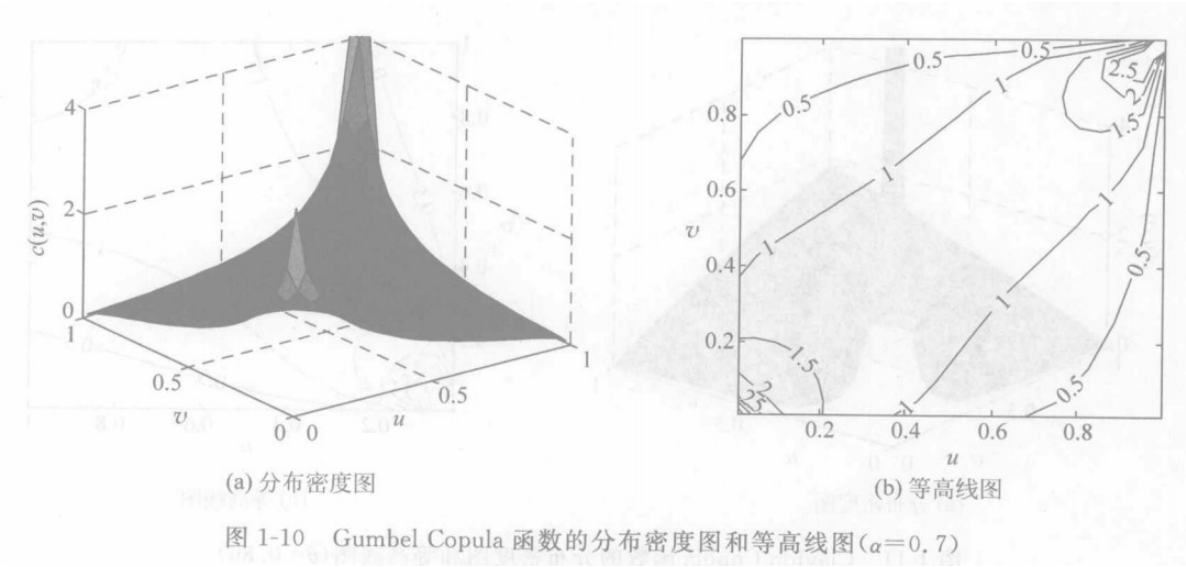
二元 t-copula 与正态 copula 类似，也具有对称性，只能捕捉到金融市场的对称的相关关系，但 t-copula 尾部变化特别灵敏，尾部更厚，因此可以捕捉金融市场的尾部相关性。

4.2.3 Gumbel copula

分布函数：$\mathrm{C}_G(u,v|\delta)=\exp(-((-logu)^{\delta}+(-logv)^{\delta})^{\frac{1}{\delta}})$

Kendall's 秩相关系数：$\tau=1-\delta$

上下尾相关性：$\lambda_L=0,\lambda_U=2-2^{1/\delta}$ 只有上尾相关性



(a) 分布密度图　　(b) 等高线图

图 1-10　Gumbel Copula 函数的分布密度图和等高线图(α=0.7)

Gumbel copula 具有明显的非对称结构，密度分布呈现“J”形，上尾高下尾低，由上图可以看出，Gumbel copula 函数对上尾部非常敏感，可以快速捕捉到上尾的关系，但难以捕捉到下尾的关系。

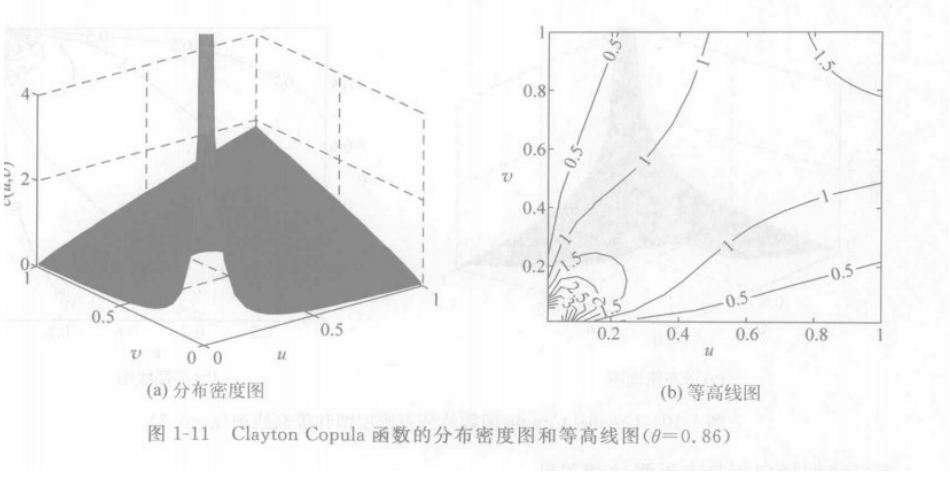
可以理解为，Gumbel copula 函数可以描述具有上尾特性的金融市场的相关关系，比如当一个金融市场暴涨时，另一个金融市场暴涨的可能性也很大，用具有上尾相关性的 Gumbel copula 函数来连接就非常合适。

4.2.4 Clayton copula

分布函数：$C_c(u,v|\theta)=\left(u^{-\theta}+v^{-\theta}-1\right)^{-\frac{1}{\theta}}$

Kendall’s 秩相关系数：$\tau=\theta/(\theta+2)$

上下尾相关性：$\lambda_L=2^{1/\theta}$，$\lambda_U=0$ 只有下尾相关性



(a) 分布密度图　　(b) 等高线图

图 1-11　Clayton Copula 函数的分布密度图和等高线图(θ=0.86)

Clayton copula 也具有明显的非对称结构，密度分布呈现“L”形，下尾高上尾低，由上图可以看出，Clayton copula 函数对下尾部非常敏感，可以快速捕捉到下尾的关系，但难以捕捉到上尾的关系。

可以理解为，Clayton copula 函数可以描述具有下尾特性的金融市场的相关关系，比如当一个金融市场暴跌时，另一个金融市场暴跌的可能性也很大，用具有下尾相关性的 Clayton copula 函数来连接就非常合适。

4.2.5 Frank copula

分布函数：$\mathrm{C}_F(u,v|\lambda)=-\frac{1}{\lambda}\ln(1-(\frac{(1-e^{-\lambda u})(1-e^{-\lambda v})}{(1-e^{-\lambda})})$

Kendall's 秩相关系数：$\tau=1+\frac{4}{\lambda}[D(\lambda)-1]$，其中$D(\cdot)$为 Debye 函数。

上下尾相关性：没有上下尾相关性 $\lambda_U=\lambda_L=0$

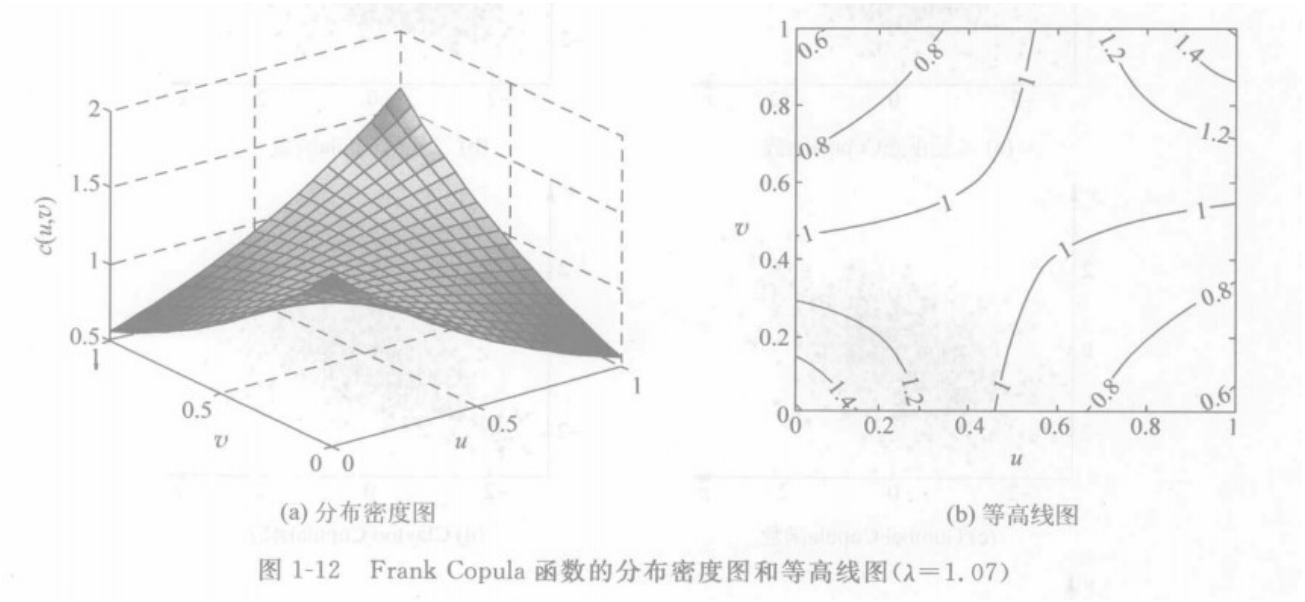


图 1-12 Frank Copula 函数的分布密度图和等高线图(λ=1.07)

上述的 Gumbel copula 和 Clayton copula 只能描述变量间的非负相关关系，而 Frank copula 还可以描述变量之间的负相关关系，分布函数中的λ是 Frank copula 函数的参数，$\lambda>0$表示随机变量u,v正相关，$\lambda<0$表示随机变量u,v负相关。

Frank copula 密度函数为“U”形分布，有对称性，无法捕捉非对称市场的关系，且尾部变化不敏感，没有尾部相关性。

4.2.6 Symmetrized Joe-Clayton copula

C_{JC} 是 Joe Clayton copula, 他的密度函数如下

$C_{JC}(u_1,u_2|\tau^U,\tau^L)=1-(1-\{[1-(1-u_1)^{\kappa}]^{-\gamma}+[1-(1-u_2)^{\kappa}]^{-\gamma}-1\}^{-\frac{1}{\gamma}})^{-1/\kappa}$,

其中， $\kappa=1/log_2(2-\tau^U)$, $\gamma=-1/log_2(2-\tau^L)$ and $\tau^U,\tau^L\in(0,1)$

SJC 函数拥有上尾和下尾相关性，其密度函数为

$$C_{SJC}(u,v|\tau^U,\tau^L)=0.5(C_{JC}(u,v|\tau^U,\tau^L)+C_{JC}(1-u,1-v|\tau^U,\tau^L)+u+v-1)$$

五、藤 copula(vine-copula)

在藤 copula 模型中，可以衡量直接和间接的金融市场价格传导渠道，衡量了不同金融市场之间的依赖性。模型使用由一组捕捉两个变量之间依赖的双变量 copula 组成的层次结构来描述高维联合分布。这种经验方法提供了建模的灵活性，因为边际模型和多元依赖结构是独立建模的。

5.1 藤 copula 理论

具体的 R 藤结构是一系列树的组合，每棵树的边对应一个 copula 函数或是条件 copula 函数，一个由 n 个变量的 R 藤由 n-1 棵树构成，分别记为$\mathrm{T}_1, \mathrm{T}_2, \cdots \mathrm{T}_{n-1}$,第 i 棵树的节点集记为$N_i$，边集记为$E_i(i = 1,2, \cdots, n-1)$，他们需要满足以下几个条件：

①树T_1的节点集$N_1 = \{1,2, \cdots, n\}$，边集为E_1

②第i棵树的节点集记为$N_i = E_{i-1}\ (i = 2,3, \cdots, n-1)$，即第$i$棵树的节点集是第$i-1$棵树的边集

③如果树T_i中的两条边在树T_{i+1}中用边连接，那么这两条边在树T_i中必须有一个共同的节点

建立一个 n 元 R 藤统计模型，有 n-1 棵树，记节点集为$N = \{N_1, N_2, \cdots, N_{n-1}\}$，边集$E = (E_1, E_2, \cdots, E_{n-1})$，$E_i$中的边$e = j(e), k(e)|D(e)$，其中$j(e), k(e)$是边 e 相连接的两个节点，$D(e)$是条件，我们将边 e 对应的 copula 密度函数表示成$\mathrm{c}_{j(e),k(e)|D(e)}$。设 n 个随机变量为$X_1, X_2, \cdots X_n$，用$X_{D(e)}$表示由条件集$D(e)$决定的子向量，设第$i$个随机变量边缘密度函数为$f_i$，那么 X 的联合密度函数可以表示为

$$f(x_1, x_2, \cdots x_n) = \prod_{k-1}^{n} f_k(x_k) \prod_{i=1}^{n-1} \prod_{e \in E_i} c_{j(e),k(e)|D(e)} \left(F\left(x_{j(e)} | x_{D(e)}\right), F\left(x_{k(e)} | x_{D(e)}\right) \right) \tag{5}$$

从定义可以看到，每一个藤都是许多棵树(Tree)组成，每棵树上有许多结点(Node)，连接两个结点的线称为边缘(edge)。这些树，结点和边缘合起来构成了集合，根据不同组合方式就组合成了藤结构，有众多常见的藤结构，如 C 藤(C-vine)、D 藤(D-vine)、R 藤(R-vine)。其中，Morales-Napoles 等指出 R 藤 copula 比 C 藤 D 藤 copula 更具有多样和灵活的相依结构，在金融和其他领域中高维变量相依关系建模时获得了更多的应用，因此本教程着重对 **R 藤 copula** 建模方式进行指导。

我们选择最大生成树来选择 R 藤 copula 的结构，用以下公式解决每棵树的优化问题：

$$\max \sum_{edges\ e=\{i,j\}\ in\ spanning\ tree} |\hat{\tau}_{ij}| \tag{6}$$

$\hat{\tau}_{ij}$是两两之间的 Kendall 相关系数

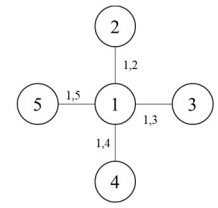
接下来我们来介绍三种藤结构。

5.2 藤结构理论

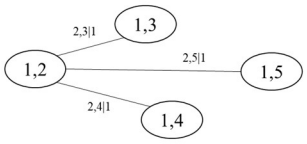
如果某个建立在 n 维变量上的藤的第 j 棵树上的两条边缘由 j+1 棵树上的一条边缘连接，且这些边缘分享同一个结点，则称这个藤为规则藤。规则藤有很多结构，如果每个结点连接其他结点的个数不多于两个，那么这种规则藤成为 D 藤，如果规则藤中的第 i 棵树只有唯一的结点 i 连接了其他 n-i 个结点，且第一棵树称为根结点，这种藤为 C 藤。

5.2.1 C 藤结构

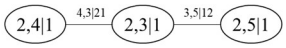
按照上述条件构造一个简单的五元 C 藤如图所示（画图可以用文档里的 PPT 进行）



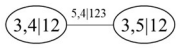
T1



T2



T3

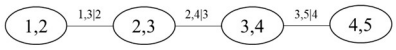


T4

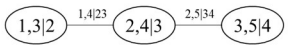
5.2.2 D 藤结构

按照上述条件构造一个简单的五元 D 藤如图所示（画图可以用文档里的 PPT 进行）

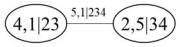
T1



T2



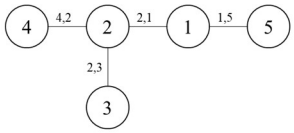
T3



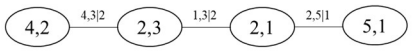
T4

5.2.3 R 藤结构

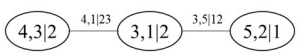
按照上述条件构造一个简单的五元 R 藤如图所示（画图可以用文档里的 PPT 进行）



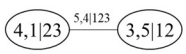
T1



T2



T3



T4

R 藤 copula 比 C 藤 D 藤 copula 更具有多样和灵活的相依结构，在金融和其他领域中高维变量相依关系建模时获得了更多的应用，不仅 copula 函数可以有多种选择而且藤结构也可以根据需要自由调整，通过调整藤结构可以获取条件相关性的更多信息。

六、Copula 边缘分布建模

Copula 建模的前提是时间序列不存在异方差和自相关，因此构建 Copula 模型之前首先要选择一个金融序列时间模型的边缘分布。目前常用的单变量时间序列模型有两类，一类是时间序列模型，另一类是波动模型。由于金融时间序列具有偏斜，高峰，后尾等特征，近年来加入波动模型得到广泛应用和关注。

6.1 时间序列模型

时间序列模型主要包括自回归模型(AR)、移动平均模型(MA)、自回归移动平均模型(ARMA)和自回归积分移动平均模型(ARIMA)

6.1.1 自回归模型

自回归模型描述序列$\{x_t\}$在某一时刻 t 与前 p 个时刻序列值之间的线性关系；p 阶自回归模型表示为

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + \varepsilon_t \qquad (2\text{-}1)$$

记为 AR(p)

6.1.2 移动平均模型

另一类时间序列模型是移动平均模型，移动平均模型中将序列$\{x_t\}$表示为若干个白噪声的线性加权和，q 阶移动平均模型表示为

$$x_t = \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} \qquad (2\text{-}7)$$

记为 MA(q)

6.1.3 自回归移动平均模型

自回归模型和移动平均模型的结合即构成自回归移动平均模型。该模型兼顾上述两种模型的特点，以尽可能少的参数描述平稳时间序列数据的变化过程。自回归移动平均模型表示为

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} \qquad (2\text{-}10)$$

6.1.4 自回归移动平均模型

前面三个模型使用的前提是平稳的时间序列（可以用 3.1 中的单位根检验进行检验），若时间序列并不平稳，具有明显的上升或下降趋势，则要考虑他们的差分，若序列经过 d 次差分以后为平稳序列，则**记为 ARIMA(p,d,q)**

6.2 波动模型

金融市场的时间序列存在条件异方差性质，所以使用自回归条件异方差(ARCH)模型来刻画时间序列的这一性质。

6.2.1 线性 ARCH 模型(LARCH)

Engle(1982)引入条件方差来分析方差的变化，规定条件方差h_t是q期扰动平方$\{\varepsilon_{t-1}^2, \varepsilon_{t-2}^2, \cdots, \varepsilon_{t-q}^2$的线性函数，则有

$$y_t = \mu_t + \varepsilon_t$$

其中，$\varepsilon_t = \sqrt{h_t}\xi_t,\ \xi_t \sim i.i.N(0,1),\ h_t = \alpha_0 + \sum_{i=1}^{q} \varepsilon_{t-i}^2$

ARCH(q)模型实际应用中需要很大的滞后阶数才拟合效果较好，会带来多重共线性等问题，因此Bollerslev(1986)扩展了广义自回归条件异方差模型(GARCH)。

6.2.2 GARCH 模型

GARCH 是 ARCH 模型的扩展，需要滞后阶数较小，与 ARMA 有相似的结构，GARCH(p,q)定义如下：

$$y_t = \mu_t + \varepsilon_t$$
$$\varepsilon_t | I_{t-1} \sim N(0, h_t)$$
$$h_t = \alpha_0 + \sum_{i=1}^{q} \alpha_i \varepsilon_{t-i}^2 + \sum_{i=1}^{p} \beta_i h_{t-i}^2$$

GARCH 模型可以用于描述具有条件异方差特性的时间序列条件边缘分布。

6.2.3 几类特殊的 GARCH 过程

(1) GARCH(1,1)过程

GARCH(1,1)是最简单的 GARCH 过程，它的条件方差函数为

$$h_t = \alpha_0 + \alpha\varepsilon_{t-1}^2 + \beta h_{t-1}$$

其中 $\alpha_0 > 0, \alpha \geqslant 0, \beta \geqslant 0$。

GARCH(1,1)是平稳的充要条件是 $\alpha+\beta<1$。如果取 $\nu_t = \varepsilon_t^2 - h_t$ 有

$$\varepsilon_t^2 = \alpha_0 + (\alpha+\beta)\varepsilon_{t-1}^2 + \nu_t - \beta\nu_{t-1}$$

说明$\{\varepsilon_t^2\}$服从 ARMA(1,1)过程。

(2) GARCH(1,1)-t 过程

GARCH(1,1)-t 过程为

$$y_t = \mu + \varepsilon_t$$
$$\sqrt{\frac{\nu}{h_t(\nu-2)}} \cdot \varepsilon_t \mid I_{t-1} \sim t(\nu)$$
$$h_t = \alpha_0 + \alpha\varepsilon_{t-1}^2 + \beta h_{t-1}$$

其中 μ 表示收益的均值，h_t 表示 ε_t 的条件方差，$\alpha \geqslant 0, \beta \geqslant 0$，$t(\nu)$表示自由度为 ν 的标准 t 分布。

大量实证研究表明，在大多数情况下，简洁的 GARCH(1,1)、GARCH(1,2)、GARCH(2,1)模型已能充分反映长时期的金融数据的波动特征[33-37]。其中 GARCH-t、GARCH-GED 模型能更好地刻画一些金融时间序列的波动特性和分布的高峰厚尾现象，对金融时间序列条件分布特性的刻画能力优于正态分布假设下的 GARCH 模型。

6.3 边缘分布建模步骤

第一步：对边缘分布建模，本文选择 ARMA(p,q)-GARCH(1,1)-偏 t 分布模型来构建边缘分布，模型假设如下：

$$\begin{cases} r_{i,t} = \phi_0 + \phi_1 r_{i,1} \cdot r_{i,t-1} + \cdots + \phi_p r_{i,p} \cdot r_{i,t-p} + \varphi_1 a_{i,1} \cdot \varepsilon_{i,t-1} + \cdots + \varphi_q a_{i,q} \cdot \varepsilon_{i,t-q} + \varepsilon_{i,t} \\ \varepsilon_{i,t} = \sigma_{i,t} \cdot \mu_{i,t} \\ \mu_{i,t} \sim skewed\ \ t(skew_i, shape_i) \\ {\sigma_{i,t}}^2 = \omega_i + \alpha_{i,1} \cdot {\varepsilon_{i,t-1}}^2 + \beta_{i,1} \cdot {\sigma_{i,t-1}}^2 \end{cases} \tag{10}$$

u_i为截距项$\phi r_{i,1}, \ldots, \phi r_{i,p}$为滞后 p 期的对数收益率参数；$\varphi a_{i,1}, \ldots, \varphi a_{i,t-q}$为滞后 p 期的误差项参数；$skew_i, shape_i$为偏 t 分布的两个参数，$\omega_i, \alpha_{i,1}, \beta_{i,1}$是待估参数。

第二步：首先对残差进行标准化处理，再进行概率积分变换得到进行 copula 建模的残差序列$\{u_t\}, \{v_t\}$

七、边缘分布建模 R 语言实操分析

7.1 使用 auto.arma 自动定阶

```
auto.arima(m1)

Series: m1
ARIMA(1,0,0) with zero mean

Coefficients:
         ar1
      0.1779
s.e.  0.0446

auto.arima(m2)

Series: m2
ARIMA(1,0,0) with zero mean

Coefficients:
         ar1
      0.1785
s.e.  0.0446

auto.arima(m3)

Series: m3
ARIMA(0,0,1) with zero mean

Coefficients:
         ma1
      0.1588
s.e.  0.0439

auto.arima(m4)

Series: m4
ARIMA(2,0,2) with zero mean

Coefficients:
          ar1     ar2     ma1      ma2
      -0.3284  0.5107  0.5027  -0.4268
s.e.   0.2088  0.1722  0.2201   0.2015

auto.arima(m5)
```

```
Series: m5
ARIMA(0,0,1) with zero mean

Coefficients:
          ma1
      0.1511
s.e.  0.0445
```

红框内为根据最小 AIC 原则得到的各个金融市场时间序列的最优 ARMA 滞后阶数，分别记录后对边缘分布进行 ARMA(p,q)-GARCH(1,1)-偏 t 建模。

7.2 构建 arma-garch 模型

• 导入 garch 库

```
library(rugarch)
```

• 设定边缘分布模型命名为 spec_m1

```
spec_m1 <- ugarchspec(variance.model = list(model = "sGARCH",
                                            garchOrder = c(1, 1),
                                            submodel = NULL,
                                            external.regressors = NULL,
                                            variance.targeting = FALSE),

                      mean.model     = list(armaOrder = c(3, 3),
                                            external.regressors = NULL),
                      distribution.model="sstd",
                      start.pars = list(),
                          fixed.pars = list())
```

参数说明:

variance.model 是波动模型的参数

model = “sGARCH”选用的是 GARCH 模型

Garchorder = c(1,1)是 GARCH(1,1)模型

mean.model 是时间序列模型参数

armaorder = c(3,3)是 ARMA(3,3)模型

distribution.model="sstd" 是偏 t 分布，可以选的参数还有 norm 是正态分布、std 是 t 分布等

• 同理分别对剩余四个市场建立边缘分布

```
spec_m2 <- ugarchspec(variance.model = list(model = "sGARCH",
                                            garchOrder = c(1, 1),
                                            submodel = NULL,
                                            external.regressors = NULL,
                                            variance.targeting = FALSE),

                      mean.model     = list(armaOrder = c(4, 4),
                                            external.regressors = NULL),
                      distribution.model="sstd",
                      start.pars = list(),
                      fixed.pars = list())
```

```
garch_m2 <- ugarchfit(spec = spec_m2, data = m2, solver.control = list(trace=0))

spec_m3 <- ugarchspec(variance.model = list(model = "sGARCH",
                                            garchOrder = c(1, 1),
                                            submodel = NULL,
                                            external.regressors = NULL,
                                            variance.targeting = FALSE),
                      mean.model     = list(armaOrder = c(0, 0),
                                            external.regressors = NULL),
                      distribution.model="sstd",
                      start.pars = list(),
                      fixed.pars = list())
garch_m3 <- ugarchfit(spec = spec_m3, data = m3, solver.control = list(trace=0))

spec_m4 <- ugarchspec(variance.model = list(model = "sGARCH",
                                            garchOrder = c(1, 1),
                                            submodel = NULL,
                                            external.regressors = NULL,
                                            variance.targeting = FALSE),
                      mean.model     = list(armaOrder = c(3, 2),
                                            external.regressors = NULL),
                      distribution.model="sstd",
                      start.pars = list(),
                      fixed.pars = list())
garch_m4 <- ugarchfit(spec = spec_m4, data = m4, solver.control = list(trace=0))

spec_m5 <- ugarchspec(variance.model = list(model = "sGARCH",
                                            garchOrder = c(1, 1),
                                            submodel = NULL,
                                            external.regressors = NULL,
                                            variance.targeting = FALSE),
                      mean.model     = list(armaOrder = c(3, 1),
                                            external.regressors = NULL),
                      distribution.model="sstd",
                      start.pars = list(),
                      fixed.pars = list())
garch_m5 <- ugarchfit(spec = spec_m5, data = m5, solver.control = list(trace=0))
```

7.3 模型结果输出及参数解释

分别输出五个金融市场边缘分布结果

garch_m1

garch_m2

garch_m3

garch_m4

garch_m5

• 以 garch_m4 为例

```
*----------------------------------*
*          GARCH Model Fit         *
*----------------------------------*

Conditional Variance Dynamics
-----------------------------------
GARCH Model      : sGARCH(1,1)
Mean Model       : ARFIMA(2,0,2)
Distribution     : sstd

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu     -0.000411    0.000483 -0.85214 0.394135
ar1    -0.400183    0.920886 -0.43456 0.663880
ar2     0.456105    0.809737  0.56328 0.573247
ma1     0.527457    0.956470  0.55146 0.581317
ma2    -0.399330    0.901597 -0.44291 0.657828
omega   0.000006    0.000001  5.54960 0.000000
alpha1  0.079259    0.015725  5.04030 0.000000
beta1   0.858591    0.025852 33.21126 0.000000
skew    1.189433    0.084018 14.15688 0.000000
shape   7.529175    2.351234  3.20222 0.001364
```

ARIMA(2,0,2)-GARCH(1,1)-偏t分布

系数 标准误 统计量 p值

• 对数似然值

```
LogLikelihood : 1594.233
```

• 对照公式：

$$r_{i,t} = \phi_0 + \phi_1 r_{i,1} \cdot r_{i,t-1} + \cdots + \phi_p r_{i,p} \cdot r_{i,t-p} + \varphi_1 a_{i,1} \cdot \varepsilon_{i,t-1} + \cdots + \varphi_q a_{i,q} \cdot \varepsilon_{i,t-q} + \varepsilon_{i,t}$$

$$\varepsilon_{i,t} = \sigma_{i,t} \cdot \mu_{i,t}$$

$$\mu_{i,t} \sim skewed\ \ t(skew_i, shape_i)$$

$$\sigma_{i,t}{}^2 = \omega_i + \alpha_{i,1} \cdot \varepsilon_{i,t-1}{}^2 + \beta_{i,1} \cdot \sigma_{i,t-1}{}^2$$

• 参数解释

mu：ϕ_0

ar1-ar2 自回归系数：ϕ_1、ϕ_2

ma1-ma2 移动平均系数：φ_1、φ_2

omega：garch 模型常数项ω_i

alpha1：ARCH 项系数α

beta1：GARCH 项系数β

skew：偏 t 分布$skew_i$参数

shape：偏 t 分布$shape_i$参数

```
Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.8164  0.3662
Lag[2*(p+q)+(p+q)-1][11]   3.9429  0.9999
Lag[4*(p+q)+(p+q)-1][19]   6.6378  0.9366
d.o.f=4
H0 : No serial correlation
```

LB对标准化残差序列检验

```
Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.4684  0.4937
Lag[2*(p+q)+(p+q)-1][5]    0.9262  0.8762
Lag[4*(p+q)+(p+q)-1][9]    1.6491  0.9425
d.o.f=2
```

LB2对标准化残差序列平方项检验

```
Weighted ARCH LM Tests
------------------------------------
            Statistic Shape Scale P-Value
ARCH Lag[3]    0.0332 0.500 2.000  0.8554
ARCH Lag[5]    1.0595 1.440 1.667  0.7153
ARCH Lag[7]    1.3543 2.315 1.543  0.8501
```

ARCH-LM检验是否存在ARCH效应

7.4 模型结果三线表格式及解释

对 garch_m4 结果汇总到三线表中，结果如下：

	M4
mean	
ϕ_0	0.000(-0.852)
ϕ_1	-0.400(-0.435)
ϕ_2	0.456(0.563)
φ_1	0.527(0.551)
φ_2	-0.399(-0.443)
variance	
ω	0.000*** (5.550)
α	0.079***(5.040)
β	0.858***(33.211)
skew	1.189***(14.157)
shape	7.529***(3.202)
LogLik	1594.233
LB	6.638[0.936]
LB2	1.649[0.942]
ARCH-LM	1.354[0.850]

边缘模型的参数估计数和统计量(括号内)。*表示 10%的水平上显著性，**表示 5%的水平上显著性，***表示 1%的水平上显著性。LB, LB2 分别表示残差模型和平方残差模型中序列相关的对数似然值和 Ljung-Box 统计量。ARCH-LM 是指残差中 ARCH 效应的 Engle's LM 检验。P 值[方括号中]低于 0.05 表示正确拒绝规范的原假设。

对于 market4 的边缘分布模型中表中显示，自回归(AR)和移动平均(MA)系数不显著，说明 market4 的平均收益不显示序列依赖性。根据波动性模型中的两个参数，ARCH 和 GARCH 的系数都显著，这说明了波动性在 m4 时间序列中是可靠的。

LB（Ljung-Box Test）检验 p 值为 0.936，非常不显著，无法拒绝序列中没有自相关性的原假设，LB2 同理，p 值为 0.942，非常不显著，说明残差平方序列中没有序列相关性，并且 ARCH-LM 统计数据表明，p 值非常不显著，无法拒绝模型残差中不存在 GARCH 效应的原假设。

Copula 建模的前提是时间序列不存在异方差和自相关，所以 LB、LB2、ARCH-LM 检验中的 p 值此处不显著可以证实此处的 copula 边缘分布建模是合理的。

7.5 用经验分布构建

• 提取 sigma 序列和残差序列

```
sigma_matrix =
matrix(data =
c(garch_m1@fit$sigma,garch_m2@fit$sigma,garch_m3@fit$sigma,garch_m4@fit$sigma,garch_m5@fit$sigma),nrow = length(garch_m1@fit$sigma),ncol = 5,byrow=FALSE)

residual_matrix =
matrix(data=c(garch_m1@fit$residuals,garch_m2@fit$residuals,garch_m3@fit$residuals,garch_m4@fit$residuals,garch_m5@fit$residuals),nrow =
length(garch_m1@fit$residuals),ncol = 5,byrow=FALSE)
```

• 用 ecdf 经验分布构建 copula 分布数据，并储存在 copuladata 里

```
std_sigma_matrix = matrix(nrow = 486,ncol = 5)
copuladata = matrix(nrow = 486,ncol = 5)
for (i in c(1:5)) {
  std_sigma_matrix[,i] = residual_matrix[,i]/sigma_matrix[,i]
  f = ecdf(as.numeric(std_sigma_matrix[,i]))
  copuladata[,i] = f(std_sigma_matrix[,i])
}
copuladata
```

```
> copuladata
             [,1]        [,2]        [,3]        [,4]        [,5]
  [1,] 0.267489712 0.255144033 0.316872428 0.380658436 0.269547325
  [2,] 0.111111111 0.148148148 0.133744856 0.183127572 0.123456790
  [3,] 0.211934156 0.364197531 0.333333333 0.201646091 0.187242798
  [4,] 0.514403292 0.333333333 0.360082305 0.061728395 0.537037037
  [5,] 0.442386831 0.606995885 0.497942387 0.843621399 0.417695473
  [6,] 0.590534979 0.720164609 0.454732510 0.695473251 0.565843621
  [7,] 0.251028807 0.467078189 0.423868313 0.209876543 0.230452675
  [8,] 0.098765432 0.106995885 0.102880658 0.203703704 0.121399177
  [9,] 0.296296296 0.506172840 0.763374486 0.411522634 0.275720165
 [10,] 0.697530864 0.251028807 0.020576132 0.477366255 0.744855967
 [11,] 0.152263374 0.234567901 0.388888889 0.333333333 0.160493827
 [12,] 0.691358025 0.329218107 0.209876543 0.611111111 0.728395062
 [13,] 0.158436214 0.306584362 0.323045267 0.485596708 0.156378601
 [14,] 0.127572016 0.150205761 0.236625514 0.623456790 0.146090535
 [15,] 0.205761317 0.119341564 0.094650206 0.483539095 0.232510288
 [16,] 0.446502058 0.261316872 0.183127572 0.775720165 0.479423868
 [17,] 0.450617284 0.553497942 0.827160494 0.751028807 0.436213992
 [18,] 0.059670782 0.131687243 0.032921811 0.240740741 0.080246914
 [19,] 0.923868313 0.870370370 0.810699588 0.703703704 0.917695473
 [20,] 0.936213992 0.967078189 0.855967078 0.816872428 0.938271605
 [21,] 0.423868313 0.156378601 0.179012346 0.308641975 0.510288066
 [22,] 0.598765432 0.718106996 0.701646091 0.516460905 0.590534979
 [23,] 0.993827160 0.925925926 0.798353909 0.981481481 0.993827160
 [24,] 0.991769547 0.962962963 0.882716049 0.989711934 0.991769547
 [25,] 0.617283951 0.997942387 1.000000000 0.600823045 0.547325103
 [26,] 0.467078189 0.236625514 0.716049383 0.129629630 0.487654321
 [27,] 0.967078189 0.860082305 0.767489712 0.983539095 0.967078189
 [28,] 0.415637860 0.936213992 0.965020576 0.637860082 0.390946502
 [29,] 0.170781893 0.121399177 0.242798354 0.055555556 0.170781893
```

八、构建 R 藤模型

利用表中所报告的每个边缘模型的标准化残差的概率积分变换，适当的藤结构是根据对变量之间的 Kendall's tau 的估计来选择的。

8.1 寻找最优 Rcopula 结构

• 导入构建藤模型所需库

```
library(VineCopula)
library(copula)
library(CDVine)
```

• 寻找最优的 R 藤结构

```
Rst = RVineStructureSelect(copuladata,family = c(1:6),progress = TRUE,se = TRUE,method = 'itau',rotations = TRUE)
```

参数说明：

• RVineStructureSelect 函数是用于寻找最优的 R 藤结构

• Copuladata 是 7.5 中构建的 copula 数据

• family = c(1:6)是选用家族为 1-6 所有的 copula 函数

可以选的 copula 家族参数有以下若干种

family　Lower (or upper) triangular d x d matrix with zero diagonal entries that assigns the pair-copula families to each (conditional) pair defined by Matrix (default: family = array(0,dim=dim(Matrix))). The bivariate copula families are defined as follows:

0 = independence copula
1 = Gaussian copula
2 = Student t copula (t-copula)
3 = Clayton copula
4 = Gumbel copula
5 = Frank copula
6 = Joe copula

基本的copula函数

7 = BB1 copula
8 = BB6 copula
9 = BB7 copula
10 = BB8 copula
13 = rotated Clayton copula (180 degrees; survival Clayton'') \cr `14` = rotated Gumbel copula (180 degrees; survival Gumbel")
16 = rotated Joe copula (180 degrees; survival Joe'') \cr `17` = rotated BB1 copula (180 degrees; survival BB1")
18 = rotated BB6 copula (180 degrees; survival BB6'')\cr `19` = rotated BB7 copula (180 degrees; survival BB7")
20 = rotated BB8 copula (180 degrees; "survival BB8")
23 = rotated Clayton copula (90 degrees)
'24' = rotated Gumbel copula (90 degrees)
'26' = rotated Joe copula (90 degrees)
'27' = rotated BB1 copula (90 degrees)
'28' = rotated BB6 copula (90 degrees)
'29' = rotated BB7 copula (90 degrees)
'30' = rotated BB8 copula (90 degrees)

旋转copula函数

'33' = rotated Clayton copula (270 degrees)
'34' = rotated Gumbel copula (270 degrees)
'36' = rotated Joe copula (270 degrees)
'37' = rotated BB1 copula (270 degrees)
'38' = rotated BB6 copula (270 degrees)
'39' = rotated BB7 copula (270 degrees)
'40' = rotated BB8 copula (270 degrees)

• progress = TRUE 意思是输出构建 copula 的过程

• se = TRUE 是输出参数的标准误

• method = 'itau'构建 copula 藤结构所用的参数是 Kendall's tau

• rotations = TRUE 是包含旋转 copula 函数，若为 FALSE，则只包含基本的 copula 函数

8.2 输出 Rcopula 结构结果

```
summary(RVM)
```

```
> summary(RVM)
tree       edge | family   cop    par (0.01)    par2 (1.94) |   tau   utd   ltd
-------------------------------------------------------------------------------
   1        2,3 |      2     t    0.91 (0.01)    7.96 (1.94) |  0.73  0.53  0.53
            1,4 |      2     t    0.83 (0.02)    5.21 (1.09) |  0.62  0.48  0.48
            1,2 |      5     F    5.95 (0.40)              - |  0.51     -     -
            5,1 |      6     J   31.35 (0.85)              - |  0.94  0.98     -
   2      1,3;2 |      2     t    0.16 (0.05)   10.00 (3.41) |  0.10  0.02  0.02
          2,4;1 |      4     G    1.15 (0.04)              - |  0.13  0.17     -
          5,2;1 |      5     F   -7.81 (0.53)              - | -0.59     -     -
   3    5,3;1,2 |     23   C90   -0.03 (0.07)              - | -0.02     -     -
          5,4;2,1 |     14    SG    1.04 (0.03)              - |  0.04     -  0.05
   4  4,3;5,1,2 |     36  J270   -1.01 (0.04)              - | -0.01     -     -
```

copula函数 参数1 参数2

kendall tau 相关系数、上尾相关系数、下尾相关系数

• 可以看到 R 藤 copula 结构，五个市场总共有四棵树，第一棵树的结构为 2,3 1,4 1,2 5,1 可以看到后三个有公共的结点 1,

•copula 函数分别为 t-copula、t-copula、Frank copula、Joe copula、t-copula、Gumbel copula、Frank copula、Rot. 90 Clayton copula、survival Gumbel copula(是旋转了 180 度的 Gumbel copula)

• par 是 copula 函数的参数，par2 是 copula 函数第二个参数(t-copula 会有两个参数)

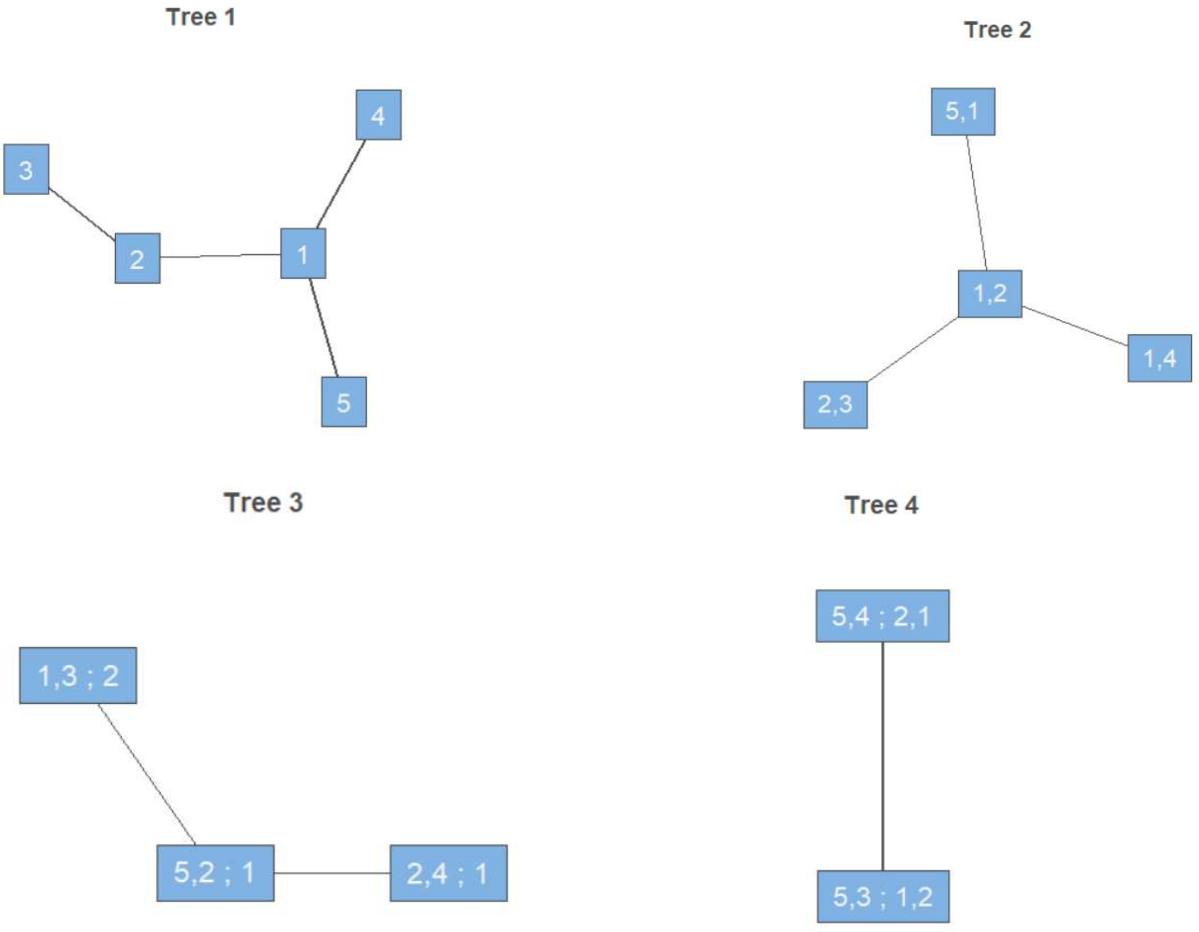
• tau 是 kendall 秩相关系数

• utd 是上尾相关系数

• ltd 是下尾相关系数

输出各棵树的结构

```
RVineTreePlot(RVM)
```

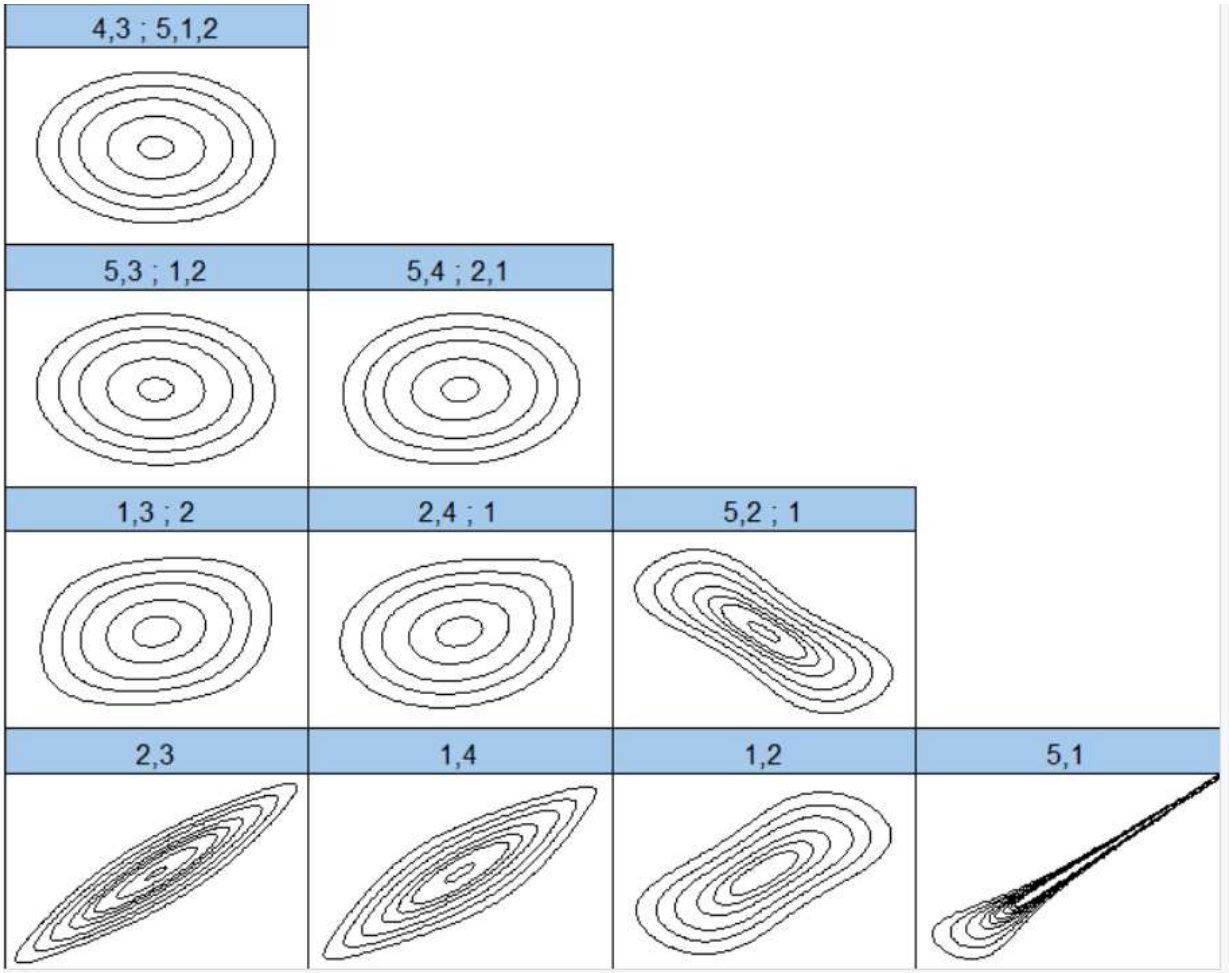


• 输出藤结构各个参数的矩阵形式

```
RVM$pair.AIC
RVM$pair.logLik
RVM$par
RVM$se
RVM$tau
```

• 输出各个结点的密度分布结构

```
contour(RVM)
```



8.3 R 藤 copula 结构三线表格式

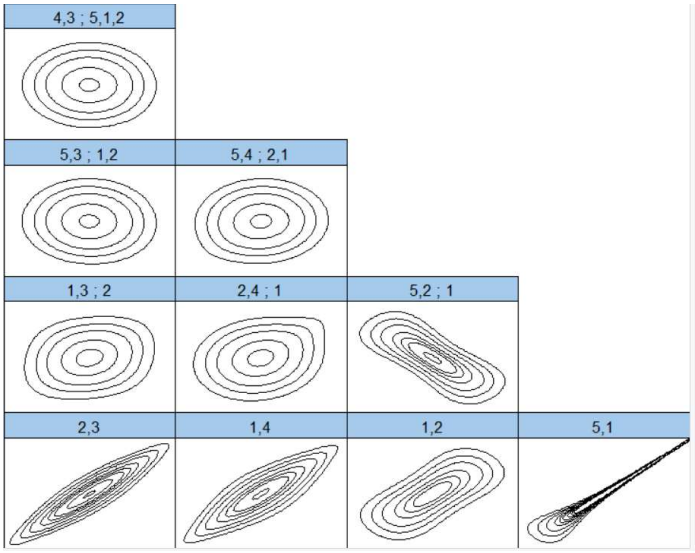
tree	edge	copula	par	par2	tau	Loglik	AIC
1	2,3	t-copula	0.91* (0.01)	7.96* (-1.94)	0.73	414.21	-824.42
	1,4	t-copula	0.83* (0.02)	5.21* (-1.09)	0.62	276.43	-548.87
	1,2	Frank copula	5.95* (0.4)		0.51	157.23	-312.46
	5,1	Joe copula	31.35* (0.85)		0.94	1631.53	-3261.07
2	1,3;2	t-copula	0.16* (0.05)	10.00* (-3.41)	0.10	7.37	-10.75
	2,4;1	Gumbel copula	1.15* (0.04)		0.13	25.20	-48.40
	5,2;1	Frank copula	-7.81* (0.53)		-0.59	219.59	-437.19
3	5,3;1,2	Rot.90 Clayton copula	-0.03 (0.07)		-0.02	0.91	0.19
	5,4;2,1	Survival Gumbel copula	1.04* (0.03)		0.04	2.96	-3.92
4	4,3;5,1,2	Rot.270 Joe copula	-1.01* (0.04)		0.071	2.100	-2.201

注：藤结构所述 copula 模型的参数估计数和标准误(括号内)和 Kendall 相关系数，(*)表示在 5%水平下显著

8.4 R 藤 copula 结果详细解释

```
> summary(RVM)
tree       edge | family   cop   par          par2 |   tau   utd   ltd
-------------------------------------------------------------------------
   1        2,3 |      2     t   0.91 (0.01)   7.96 (1.94) |  0.73  0.53  0.53
            1,4 |      2     t   0.83 (0.02)   5.21 (1.09) |  0.62  0.48  0.48
            1,2 |      5     F   5.95 (0.40)            - |  0.51     -     -
            5,1 |      6     J  31.35 (0.85)            - |  0.94  0.98     -
   2      1,3;2 |      2     t   0.16 (0.05)  10.00 (3.41) |  0.10  0.02  0.02
          2,4;1 |      4     G   1.15 (0.04)            - |  0.13  0.17     -
          5,2;1 |      5     F  -7.81 (0.53)            - | -0.59     -     -
   3    5,3;1,2 |     23   C90  -0.03 (0.07)            - | -0.02     -     -
        5,4;2,1 |     14    SG   1.04 (0.03)            - |  0.04     -  0.05
   4  4,3;5,1,2 |     36  J270  -1.01 (0.04)            - | -0.01     -     -
```

copula函数 参数1 参数2 kendall tau相关系数、上尾相关系数、下尾相关系数



首先考虑非条件藤 copula 结构，由表和图中可以看出，1,5 市场表现的相关性非常强烈，他们的 kendall 秩相关系数为 0.95，说明他们收益率变动同向的相关性为 0.95，并且拥有很强的上尾相关性，说明当一个市场暴涨时，另一个市场暴涨的相关性为 0.95，两个市场表现了很强的传染性和依赖性结构。2,3 和 1,4 有对称的 copula 结构，他们的 kendall 秩相关系数为 0.73，说明他们收益率同向变动的相关性为 0.73 和 0.62，且拥有相同的上下尾关系，说明同涨同跌的相关性为 0.53 和 0.48。1,2 市场没有上下尾相关性，收益率同方向变动相关性为 0.51。

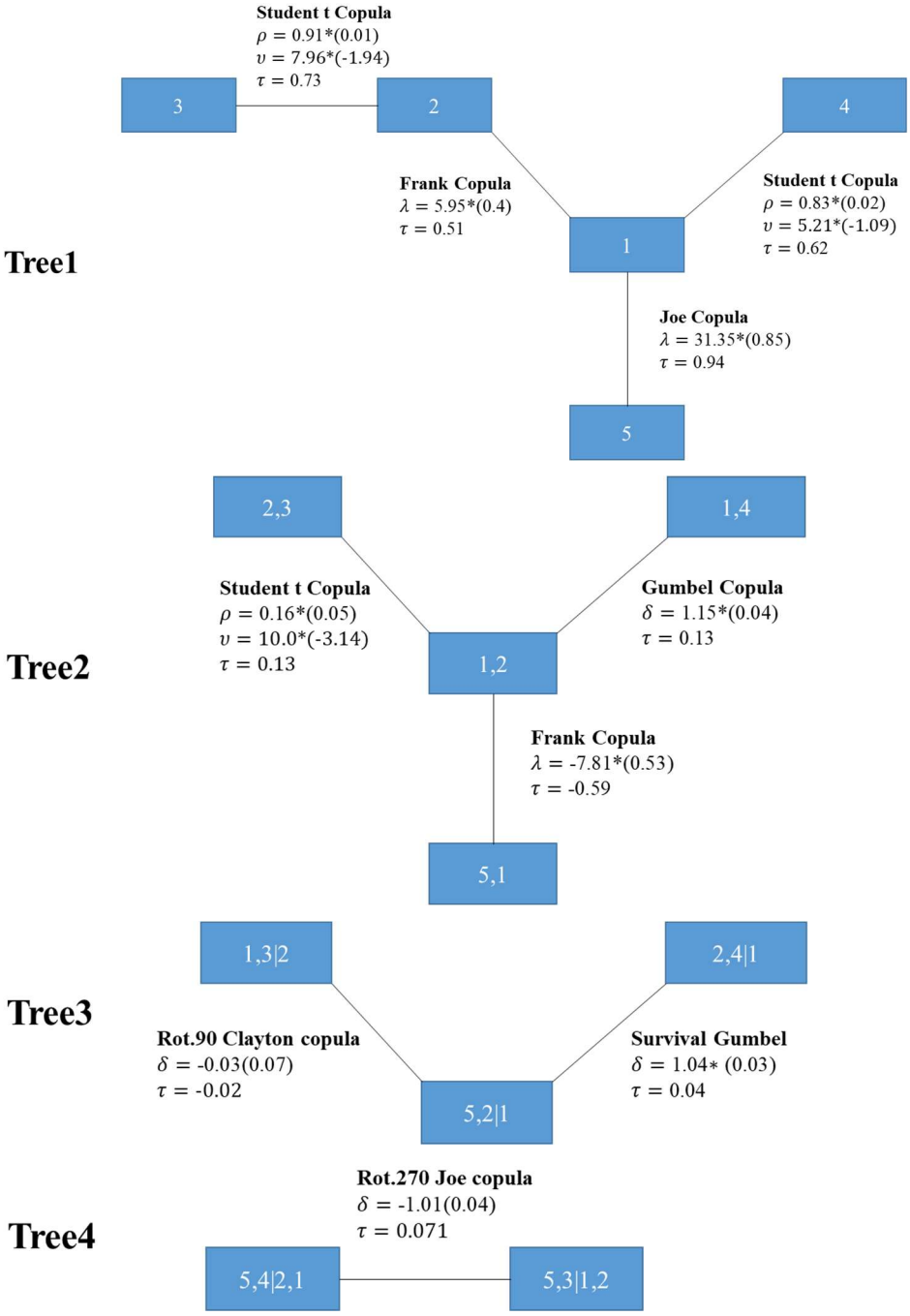
其次考虑条件藤 copula 结构，这里考虑条件相关系数，也就是考虑两个市场的间接传染结构，取定一个第 1 个金融市场为条件后，2,4 市场有 0.13 的 kendall 秩相关性，有 0.17 的上尾相关性，说明了 2,4 个市场的间接传染结构为正，但相关性不明显。5,2 市场没有上下尾相关性，他们的间接 kendall 秩相关系数为-0.59，说明 5,2 市场间接收益率变动相反。

再考虑第三层高维 copula 结构，其实越高维的 copula 结构相当于更间接的市场传染性结构，理论上其相关性会逐渐递减，给定 1,2 市场条件下，5,3 和 5,4 市场的相关性均小于 0.1，相关性不强。

最后一层 copula 结构在给定 5,1,2 条件下，3,4 市场相关性为-0.01，可以看到高维的 copula 结构相关性不强，说明间接传染结构在高维 copula 结构不明显。

存在明显的疫情传染效应，经过高维相依结构的分析，可以看到间接影响程度。

8.4 R 藤 copula 结构画图



（可以用文件里的 PPT 画藤结构图）