In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
In D:\anaconda3\lib\site-packages\matplotlib\mpl-data\stylelib\_classic_test.mplstyle:
The text.latex.preview rcparam was deprecated in Matplotlib 3.3 and will be removed two minor releases later.
In D:\anaconda3\lib\site-packages\matplotlib\mpl-data\stylelib\_classic_test.mplstyle:
The mathtext.fallback_to_cm rcparam was deprecated in Matplotlib 3.3 and will be removed two minor releases later.
In D:\anaconda3\lib\site-packages\matplotlib\mpl-data\stylelib\_classic_test.mplstyle: Support for setting the 'mathtext.fallback
_to_cm' rcParam is deprecated since 3.3 and will be removed two minor releases later; use 'mathtext.fallback : 'cm' instead.
In D:\anaconda3\lib\site-packages\matplotlib\mpl-data\stylelib\_classic_test.mplstyle:
The validate_bool_maybe_none function was deprecated in Matplotlib 3.3 and will be removed two minor releases later.
In D:\anaconda3\lib\site-packages\matplotlib\mpl-data\stylelib\_classic_test.mplstyle:
The savefig.jpeg_quality rcparam was deprecated in Matplotlib 3.3 and will be removed two minor releases later.
In D:\anaconda3\lib\site-packages\matplotlib\mpl-data\stylelib\_classic_test.mplstyle:
The keymap.all_axes rcparam was deprecated in Matplotlib 3.3 and will be removed two minor releases later.
In D:\anaconda3\lib\site-packages\matplotlib\mpl-data\stylelib\_classic_test.mplstyle:
The animation.avconv_path rcparam was deprecated in Matplotlib 3.3 and will be removed two minor releases later.
In D:\anaconda3\lib\site-packages\matplotlib\mpl-data\stylelib\_classic_test.mplstyle:
The animation.avconv_args rcparam was deprecated in Matplotlib 3.3 and will be removed two minor releases later.
```

In [2]:

```
data_raw1=pd.read_excel("上海银行间同业拆放利率(SHIBOR)(日).xls",index_col="指标名称").iloc[1:,:]
data_raw1.head()
```

Out[2]:

	SHIBOR:3个月
指标名称	
2013-12-31 00:00:00	5.5565
2014-01-02 00:00:00	5.5657
2014-01-03 00:00:00	5.5661
2014-01-06 00:00:00	5.5732
2014-01-07 00:00:00	5.576

In [3]:

```
data_raw2=pd.read_excel("社会融资规模存量(月).xls",index_col="指标名称").iloc[1:,:]
data_raw2.head()
```

Out[3]:

	社会融资规模存量:同比
指标名称	
2013-12-31 00:00:00	17.5
2014-12-31 00:00:00	14.3
2015-03-31 00:00:00	13
2015-06-30 00:00:00	11.9
2015-09-30 00:00:00	12.5

In [4]:

```
data_raw=pd.merge(data_raw1,data_raw2,on=["指标名称"]).dropna()
data_raw.head()
```

Out[4]:

	SHIBOR:3个月	社会融资规模存量:同比
指标名称		
2013-12-31 00:00:00	5.5565	17.5
2014-12-31 00:00:00	5.1351	14.3
2015-03-31 00:00:00	4.8975	13
2015-06-30 00:00:00	3.233	11.9
2015-09-30 00:00:00	3.153	12.5

In [5]:

```
#这里引入一个月未来函数。如果不用把30改为0
import datetime
data_raw.index=data_raw.index+datetime.timedelta(days=30)
data_raw.index=data_raw.index.strftime("%Y-%m-%d %H:%M:%S")
data_raw.head()
```

Out[5]:

	SHIBOR:3个月	社会融资规模存量:同比
2014-01-30 00:00:00	5.5565	17.5
2015-01-30 00:00:00	5.1351	14.3
2015-04-30 00:00:00	4.8975	13
2015-07-30 00:00:00	3.233	11.9
2015-10-30 00:00:00	3.153	12.5

In [6]:

```
#求二次差（加速度）作为好坏的判断标准
data_raw_accelerate=data_raw.diff(1).diff(1).dropna().apply(lambda x:np.where(x>=0,1,0))
#生成四个象限
data_raw_accelerate["state"]=0
for i in range(len(data_raw_accelerate)):
    info=data_raw_accelerate.iloc[i,:]
    if info[0]==1 and info[1]==1:data_raw_accelerate.iloc[i,2]=1
    if info[0]==1 and info[1]==0:data_raw_accelerate.iloc[i,2]=2
    if info[0]==0 and info[1]==1:data_raw_accelerate.iloc[i,2]=3
    if info[0]==0 and info[1]==0:data_raw_accelerate.iloc[i,2]=4
data_raw_accelerate.index.name="日期"
data_raw_accelerate.head()
```

Out[6]:

	SHIBOR:3个月	社会融资规模存量:同比	state
日期			
2015-04-30 00:00:00	1	1	1
2015-07-30 00:00:00	0	1	3
2015-10-30 00:00:00	1	1	1
2016-01-30 00:00:00	1	0	2
2016-03-30 00:00:00	0	1	3

In [7]:

```
#获取中国四类资产的数据
#股票：上证指数，债券：中证全债，商品：南华商品指数，现金：货币基金指数
from datetime import datetime
industry=pd.read_excel("四品种数据.xlsx",index_col="日期")
industry.head()
```

Out[7]:

	上证指数000001.SH	中证全债H11001.CSI	南华商品指数NH0100.NHF	货基指数CN6112.CNI
日期				
2013-01-04	2276.992	144.358	1381.29	1272.14
2013-01-07	2285.364	144.448	1382.60	1272.51
2013-01-08	2276.070	144.54	1384.39	1272.68
2013-01-09	2275.340	144.605	1379.80	1272.81
2013-01-10	2283.658	144.628	1383.42	1272.94

In [8]:

```
#直接merge得到空dataframe，现在先全部转成datatime格式
industry["a"]=industry.index
industry["a"]=industry["a"].apply(pd.to_datetime,format='%Y-%m-%d')
data_raw_accelerate["a"]=data_raw_accelerate.index
data_raw_accelerate["a"]=data_raw_accelerate["a"].apply(pd.to_datetime,format='%Y-%m-%d')
```

In [9]:

```
#合并表格
data=pd.merge(data_raw_accelerate,industry,on=["a"],how='inner').iloc[:-1,2:]
data.index=data["a"]
del data["a"]
data.index.name="日期"
#计算两期间收益率
data.iloc[:,1:]=data.iloc[:,1:]/data.iloc[:,1:].shift(1)-1
data=data.dropna()
data.head()
```

Out[9]:

	state	上证指数000001.SH	中证全债H11001.CSI	南华商品指数NH0100.NHF	货基指数CN6112.CNI
日期					
2015-07-30	3	-0.165679	0.0208095	-0.106822	0.00892936
2015-10-30	1	-0.087217	0.0249122	-0.043777	0.00665966
2016-03-30	3	-0.112907	0.0285247	0.088810	0.011931
2016-06-30	2	-0.023675	0.00372065	0.145371	0.00629373
2016-09-30	3	0.025634	0.0219902	0.026770	0.00636142

In [10]:

```
#计算各个指标的年化收益率
for i in range(1,len(data)):
    #计算两个周期相差多少天
    day=(data.index[i]-data.index[i-1]).days
    data.iloc[i,1:]=data.iloc[i,1:].apply(lambda x:float(x)/(day/252))
data=data.dropna()
data.head()
```

Out[10]:

	state	上证指数000001.SH	中证全债H11001.CSI	南华商品指数NH0100.NHF	货基指数CN6112.CNI
日期					
2015-07-30	3	-0.165679	0.0208095	-0.106822	0.00892936
2015-10-30	1	-0.238898	0.0682378	-0.119910	0.0182417
2016-03-30	3	-0.187189	0.047291	0.147238	0.0197803
2016-06-30	2	-0.064848	0.0101913	0.398190	0.0172394
2016-09-30	3	0.070214	0.0602341	0.073326	0.0174248

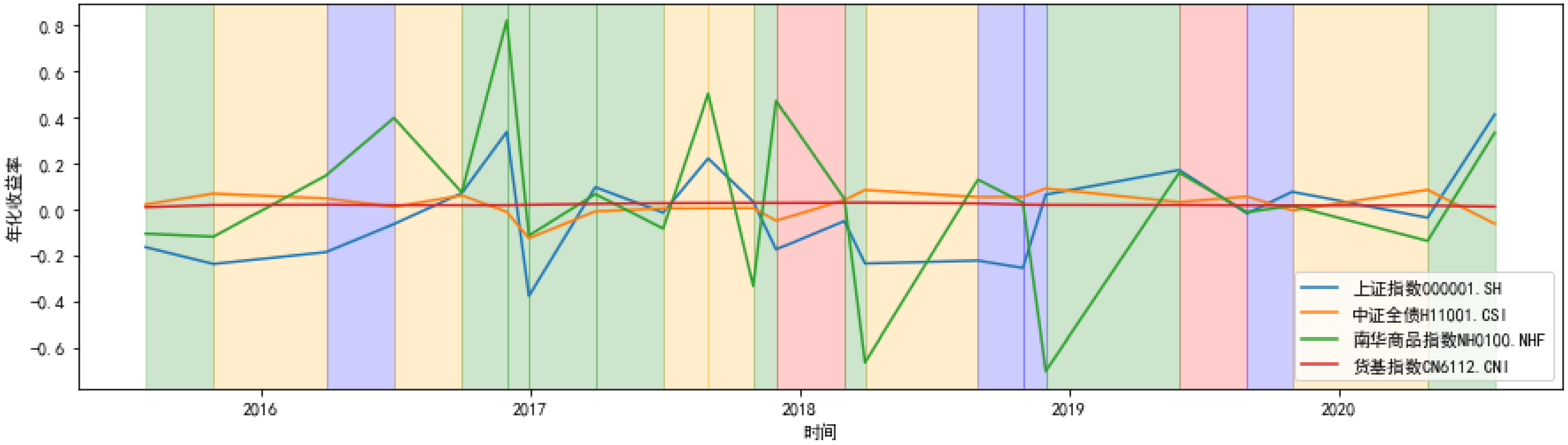
In [11]:

```
#画状态图
plt.figure(figsize=(15, 4))
x=data.index
for i in range(1,len(data.index)):
    if data.iloc[i,0]==1:
        plt.axvspan(x[i-1], x[i], ymin=1.5, ymax=-1.5, alpha=0.2, color='green')
    if data.iloc[i,0]==2:
        plt.axvspan(x[i-1], x[i], ymin=1.5, ymax=-1.5, alpha=0.2, color='blue')
    if data.iloc[i,0]==3:
        plt.axvspan(x[i-1], x[i], ymin=1.5, ymax=-1.5, alpha=0.2, color='orange')
    if data.iloc[i,0]==4:
        plt.axvspan(x[i-1], x[i], ymin=1.5, ymax=-1.5, alpha=0.2, color='red')
plt.plot(data.iloc[:,1:],)
plt.legend(data.columns[1:])
plt.xlabel("时间")
plt.ylabel("年化收益率")
print("绿色：货币+信用+，蓝色：货币+信用-，橙色：货币-信用+，红色：货币-信用-")
```

```
D:\anaconda3\lib\site-packages\pandas\plotting\_matplotlib\converter.py:103: FutureWarning: Using an implicitly registered dateti
me converter for a matplotlib plotting method. The converter was registered by pandas on import. Future versions of pandas will r
equire you to explicitly register matplotlib converters.

To register the converters:
        >>> from pandas.plotting import register_matplotlib_converters
        >>> register_matplotlib_converters()
  warnings.warn(msg, FutureWarning)
```

绿色：货币+信用+，蓝色：货币+信用-，橙色：货币-信用+，红色：货币-信用-



In [12]:
```
#统计各区间累计收益
result=data.groupby("state").apply(np.mean).iloc[:,1:]
result
```

Out[12]:

state	上证指数000001.SH	中证全债H11001.CSI	南华商品指数NH0100.NHF	货基指数CN6112.CNI
1	-0.003016	-0.008238	0.096114	0.021041
2	-0.045293	0.037254	-0.065703	0.017981
3	-0.041361	0.039358	0.038909	0.020271
4	-0.035094	0.047727	0.016396	0.022514

In [13]:

```
result.sum()
```

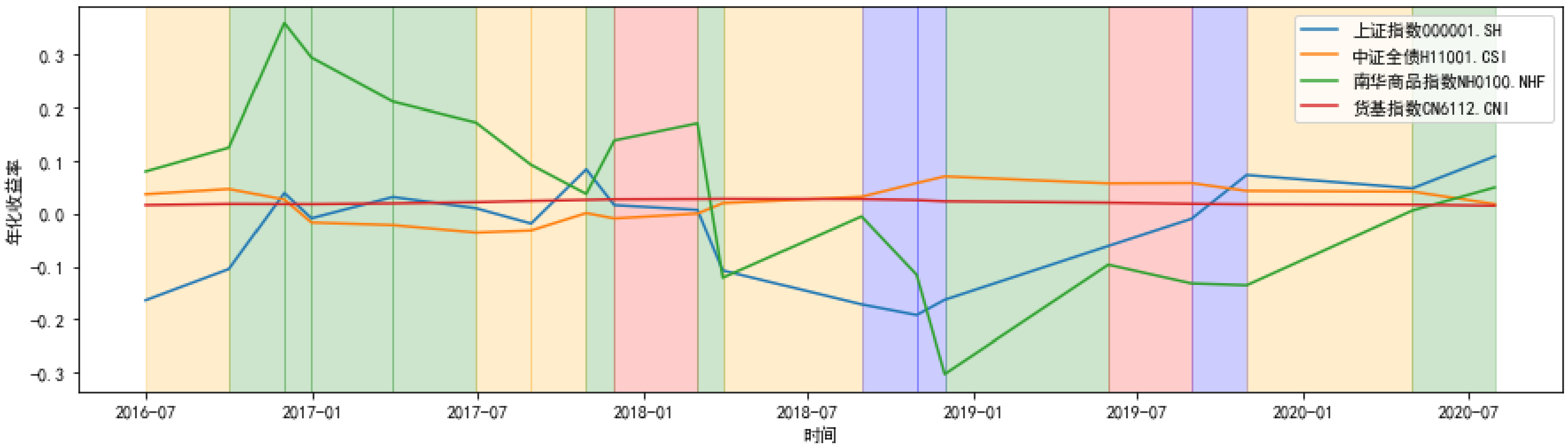
Out[13]:

```
上证指数000001.SH      -0.124764
中证全债H11001.CSI      0.116101
南华商品指数NH0100.NHF    0.085716
货基指数CN6112.CNI      0.081807
dtype: float64
```

In [14]:

```
#四期rolling mean再画图
data_rolling=data
data_rolling.iloc[:,1:]=data_rolling.iloc[:,1:].rolling(4).mean()
data_rolling=data_rolling.dropna()
plt.figure(figsize=(15, 4))
x=data_rolling.index
for i in range(1,len(data_rolling.index)):
    if data_rolling.iloc[i,0]==1:
        plt.axvspan(x[i-1], x[i], ymin=1.5, ymax=-1.5, alpha=0.2, color='green')
    if data_rolling.iloc[i,0]==2:
        plt.axvspan(x[i-1], x[i], ymin=1.5, ymax=-1.5, alpha=0.2, color='blue')
    if data_rolling.iloc[i,0]==3:
        plt.axvspan(x[i-1], x[i], ymin=1.5, ymax=-1.5, alpha=0.2, color='orange')
    if data_rolling.iloc[i,0]==4:
        plt.axvspan(x[i-1], x[i], ymin=1.5, ymax=-1.5, alpha=0.2, color='red')
plt.plot(data_rolling.iloc[:,1:])
plt.legend(data_rolling.columns[1:])
plt.xlabel("时间")
plt.ylabel("年化收益率")
print("绿色：货币+信用+，蓝色：货币+信用-，橙色：货币-信用+，红色：货币-信用-")
```

```
绿色：货币+信用+，蓝色：货币+信用-，橙色：货币-信用+，红色：货币-信用-
```



In [ ]:

In [ ]: