时序分析(3) -- 自回归模型(AR)

如无特殊说明，本系列文章中的数据将使用2012~2017年，分别代表国内股票、香港股票、国内债卷和国内货币的四个指数数据。

> 上一篇文章我们探讨了时序数据的平稳性和单根检验，我们看到单根过程和自回归时序是有关系的。这一节我们主要讨论时序数据自回归模型。

首先我们介绍自回归模型的基本概念:

Autoregressive Models - AR(p)

自回归模型是时序分析中的一项基本技术，理解和掌握AR模型是学习更高级和复杂时序分析模型的基础。
AR模型定义如下:

如果一个单变量时序数据$\{y_t; t = 0, 1, 2...\}$，y_t可以以此时序数据本身的多个t时刻之前的点的值来回归，这种情况称为自回归，公式如下

$$\begin{aligned} y_t &= \alpha_1 y_{t-1} + \alpha_2 y_{t-2} + \ldots + \alpha_p y_{t-p} + \omega_t \\ &= \sum_{i=1}^{p} \alpha_i x_{t-i} + \omega_t \end{aligned}$$

这里p称为自回归模型的阶数，记作AR(p)。α是系数项，ω_t是白噪声。

首先我们需要介绍一下自回归分析中的两个重要的分析指标：ACF 和 PACF

- ACF (Auto Correlation Function)

一个随机过程的自相关函数就是在该过程时序数据不同时间点的Pearson相关系数。假如时序数据有两个时间点s, t，其自相关系数定义为:

$$R(s, t) = \frac{E[(y_t - \mu_t)(y_s - \mu_s)]}{\sigma_t \sigma_s}$$

其中μ, s分别为均值和方差。

求出y_t与$\{y_{t-1}, y_{t-2}, \ldots\}$的自相关系数，就可以得到一个以时间跨度$k$为参数的函数，称为自相关函数。

- PACF (Partial Auto Correlation Function)

PACF偏自相关函数就是y_t与y_{t-k}的相关系数，但是移除了y_t对y_{t-k+1}到y_{t-1}的线性依赖。

1. 导入必要的python包

In [44]:

```
import warnings
warnings.simplefilter('ignore')
```

In [45]:

```
import pandas as pd
import numpy as np
%matplotlib inline

from fintechtools.backtest import *
from fintechtools.datasource import *
from fintechtools.SimuMultiTest import *

import matplotlib
import matplotlib as mpl
from matplotlib.ticker import FuncFormatter
mpl.style.use('classic')

plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['font.serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
import seaborn as sns
sns.set_style("whitegrid",{"font.sans-serif":['simhei', 'Arial']})
sns.set_context("talk")

%load_ext autoreload
%autoreload 2
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

2. 读入数据

In [46]:

```
start = '2012-01-01'
end = '2017-02-05'
```

In [47]:

```
indexs = pd.read_excel('./data/华夏指数.xlsx')
indexs_pv = indexs.pivot_table(index='日期', columns= '简称', values= '收盘价(元)')
indexs_pv.index = pd.to_datetime(indexs_pv.index, unit='d')
```

In [48]:

```
indexs_pv.columns = ['国内债券', '国内股票', '香港股票', '国内货币']
indexs_pv = indexs_pv[['国内债券', '国内股票', '国内货币', '香港股票']]
indexs_pv.fillna(axis=0,method='bfill',inplace=True)
indexs_sub = indexs_pv.loc[start:end,]
```

国内债卷：中债综合财富(总值)指数
国内股票：中证全指
香港股票：恒生指数

国内货币：货币基金

In [49]:

```
indexs_sub.head()
```

Out[49]:

	国内债券	国内股票	国内货币	香港股票
日期				
2012-01-04	141.5160	2571.951	1166.7726	18727.31
2012-01-05	141.5501	2513.699	1166.9696	18813.41
2012-01-06	141.7277	2527.247	1167.1185	18593.06
2012-01-09	141.8669	2619.638	1167.5058	18865.72
2012-01-10	142.0118	2713.529	1167.6330	19004.28

In [50]:

```
indexs_logret = indexs_sub.apply(log_return).dropna()
```

In [51]:

```
indexs_logret.head()
```

Out[51]:

	国内债券	国内股票	国内货币	香港股票
日期				
2012-01-05	0.000241	-0.022909	0.000169	0.004587
2012-01-06	0.001254	0.005375	0.000128	-0.011782
2012-01-09	0.000982	0.035906	0.000332	0.014558
2012-01-10	0.001021	0.035214	0.000109	0.007318
2012-01-11	0.000188	-0.002115	0.000113	0.007740

In [52]:

```
def tsplot(y, lags=None, figsize=(16, 10), style='bmh'):
    if not isinstance(y, pd.Series):
        y = pd.Series(y)
    with plt.style.context(style):
        fig = plt.figure(figsize=figsize)
        #mpl.rcParams['font.family'] = 'Ubuntu Mono'
        layout = (3, 2)
        ts_ax = plt.subplot2grid(layout, (0, 0), colspan=2)
        acf_ax = plt.subplot2grid(layout, (1, 0))
        pacf_ax = plt.subplot2grid(layout, (1, 1))
        qq_ax = plt.subplot2grid(layout, (2, 0))
        pp_ax = plt.subplot2grid(layout, (2, 1))

        y.plot(ax=ts_ax)
        ts_ax.set_title('Time Series Analysis Plots')
        smt.graphics.plot_acf(y, lags=lags, ax=acf_ax, alpha=0.5)
        smt.graphics.plot_pacf(y, lags=lags, ax=pacf_ax, alpha=0.5)
        sm.qqplot(y, line='s', ax=qq_ax)
        qq_ax.set_title('QQ Plot')
        scs.probplot(y, sparams=(y.mean(), y.std()), plot=pp_ax)

        plt.tight_layout()
    return
```

再详细介绍AR模型之前，我们先介绍三个常见的更为简单的模型：Linear Model，log-Linear Model 和 Random Walk。

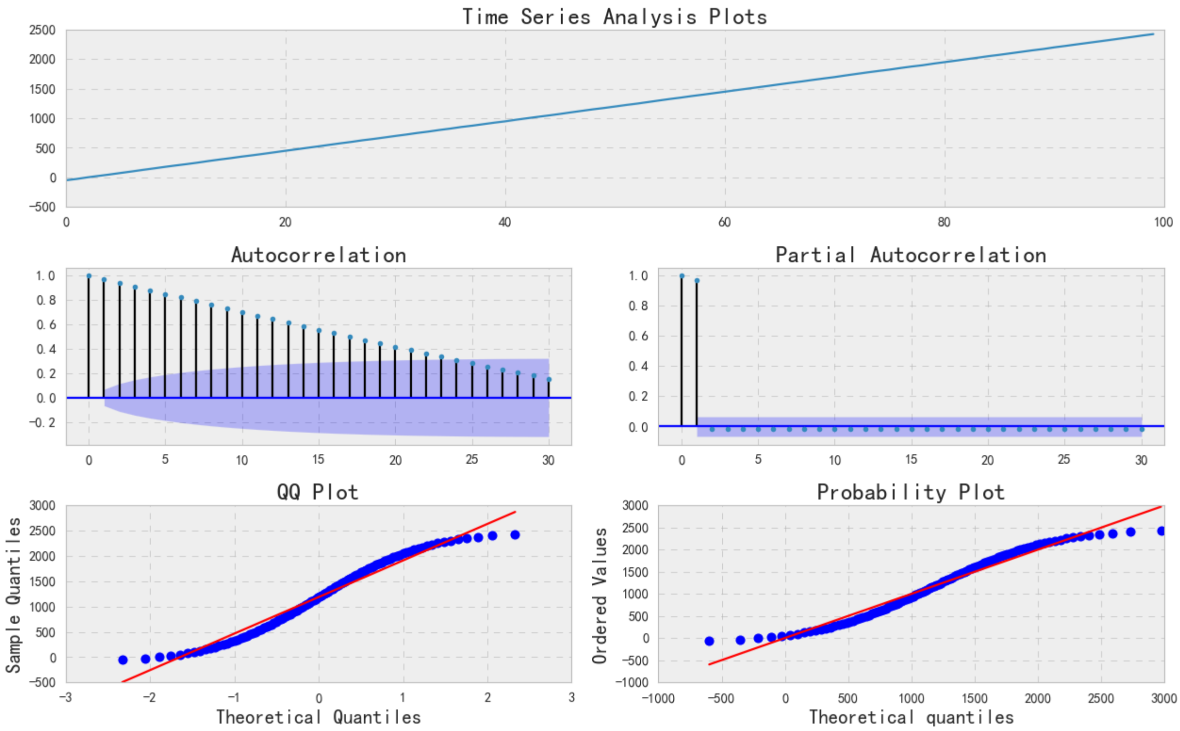
- 线性模型(Linear Model)

$$y_t = b_0 + b_1 t + \omega_t$$

时序变量y_t是时间的线性函数

In [53]:

```
w = np.random.randn(100)
y = np.empty_like(w)
lags = 30
b0 = -50.
b1 = 25.
for t in range(len(w)):
    y[t] = b0 + b1*t + w[t]

_ = tsplot(y, lags=lags)
```



从上图可观测到，线性模型的自相关函数呈现阶梯型下降，而偏相关函数只在时间跨度为1时显示较强相关性，大于1的时间跨度时相关性为0。

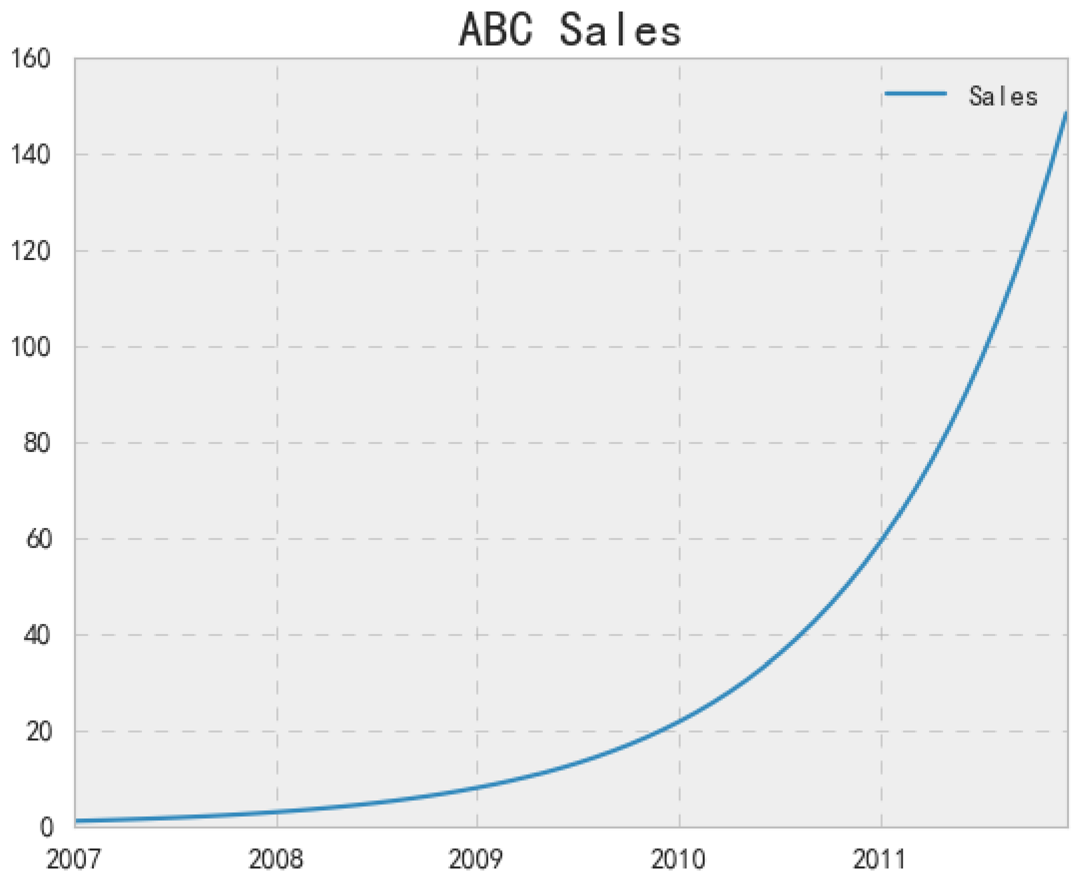
- 对数线性模型(Log-linear Model)

$$y_t = exp(\lambda t)$$

通常表示固定的连续增长，如果对两边取对数，可以得到线性模型。

In [54]:

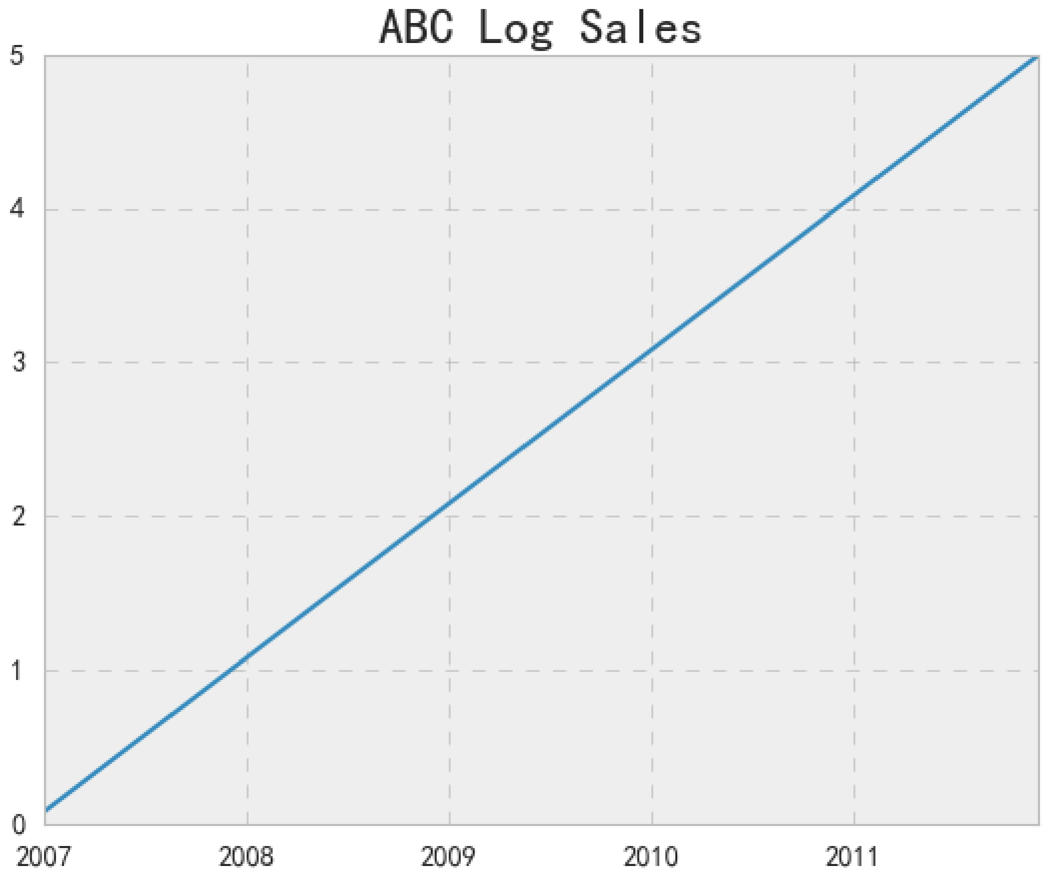
```
idx = pd.date_range('2007-01-01', '2012-01-01', freq='M')

# fake sales increasing at exponential rate
sales = [np.exp( x/12 ) for x in range(1, len(idx)+1)]

# create dataframe and plot
df = pd.DataFrame(sales, columns=['Sales'], index=idx)

with plt.style.context('bmh'):
    df.plot()
    plt.title('ABC Sales')
```



当我们取对数之后

In [55]:

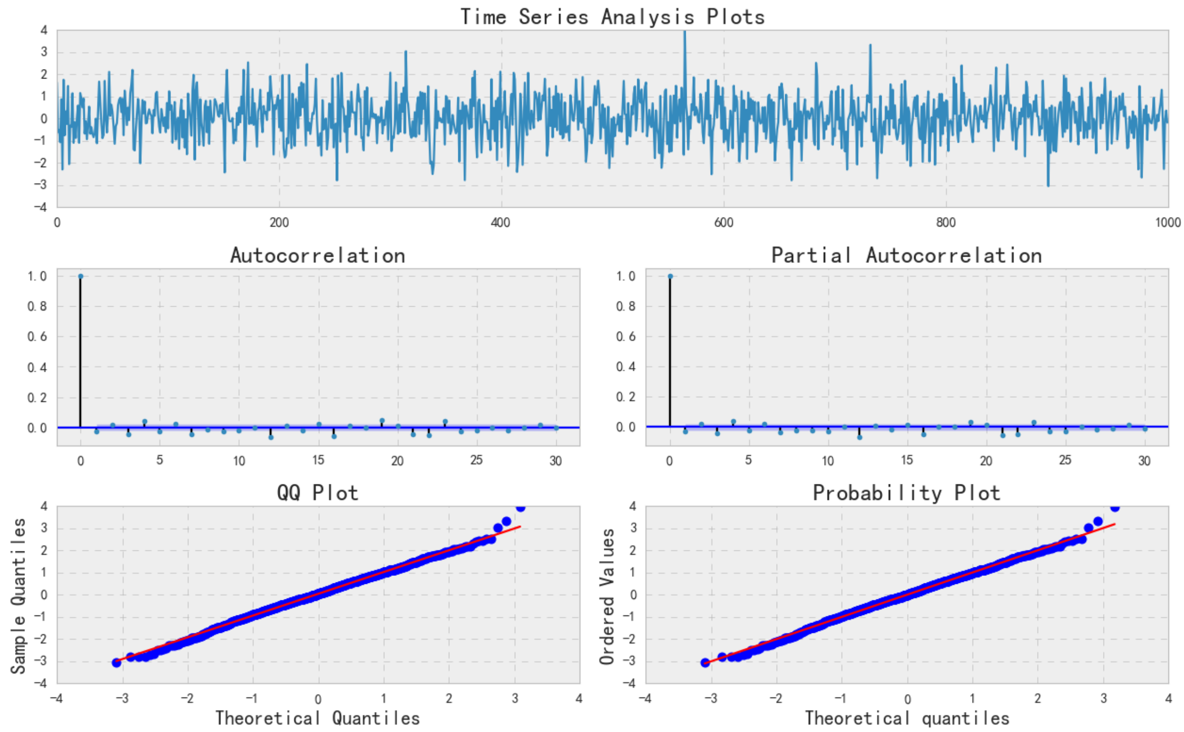
```
with plt.style.context('bmh'):
    pd.Series(np.log(sales), index=idx).plot()
    plt.title('ABC Log Sales')
```



- 白噪声

In [56]:

```
np.random.seed(1)

# plot of discrete white noise
randser = np.random.normal(size=1000)
tsplot(randser, lags=30)
```



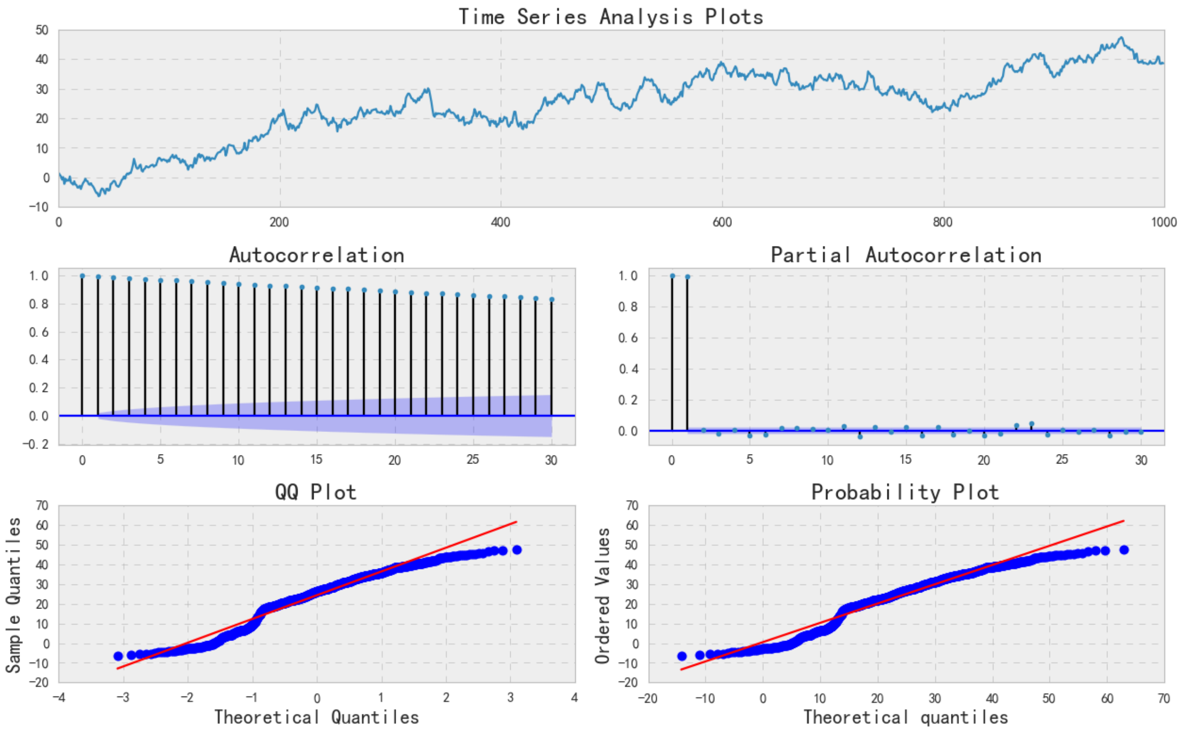
白噪声的自相关性几乎全为0.

- 随机步行(Random Walk)

$$y_t = y_{t-1} + \omega_t$$

In [57]:

```
np.random.seed(1)
n_samples = 1000

x = w = np.random.normal(size=n_samples)
for t in range(n_samples):
    x[t] = x[t-1] + w[t]

_ = tsplot(x, lags=30)
```

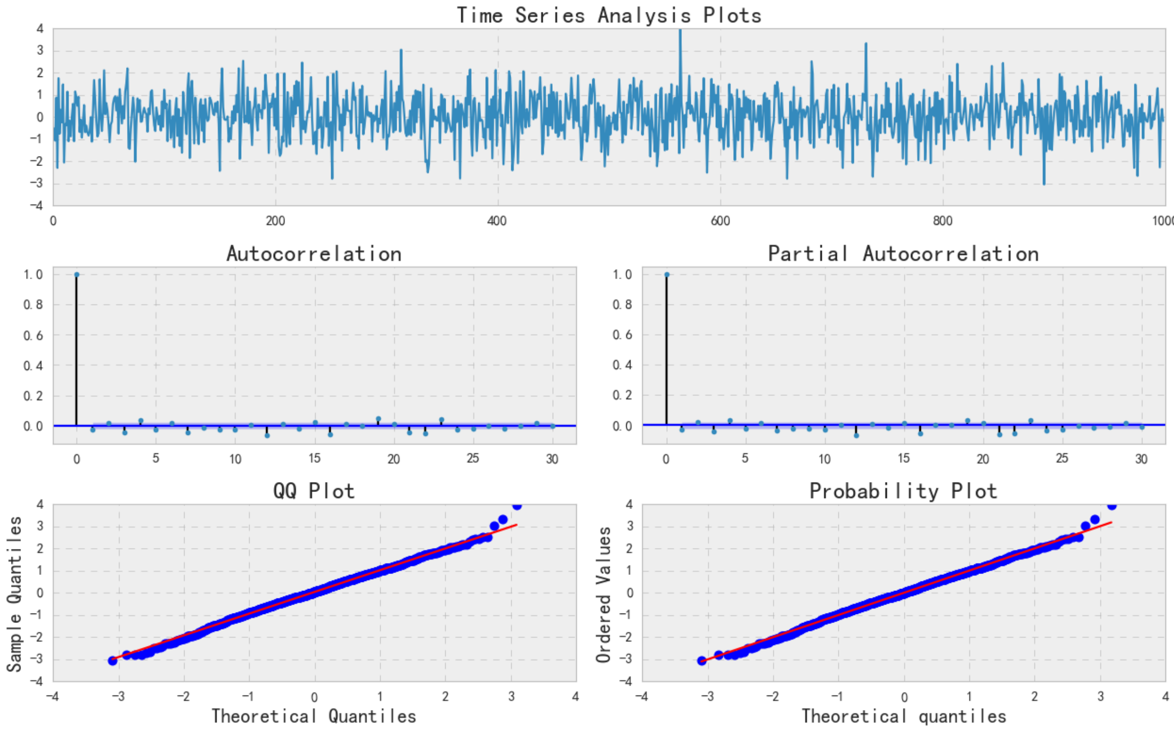


随机步行的自相关函数显示较慢下降的强相关性，而偏相关函数只在k=1时显示非常强的相关性。

对随机步行进行一阶差分后

In [58]:

```
_ = tsplot(np.diff(x), lags=30)
```



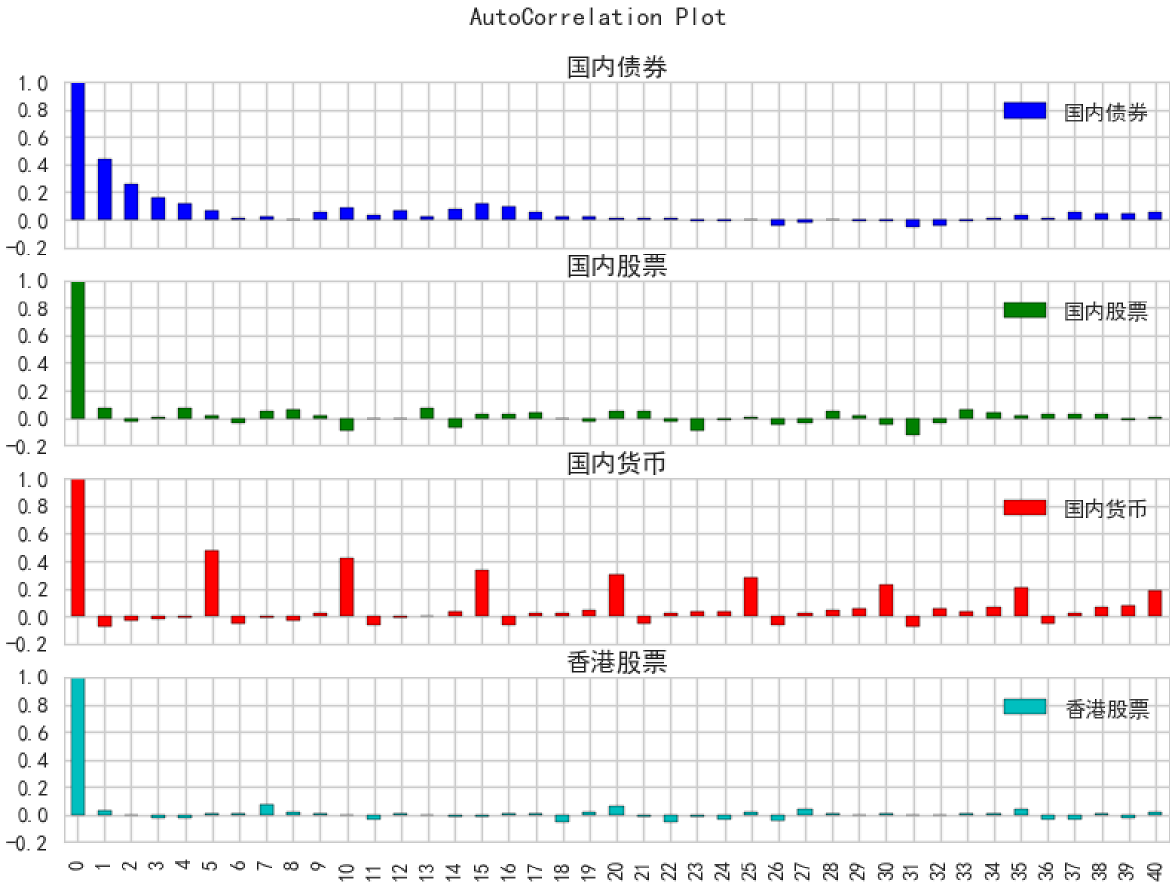
和我们预期的一样，随机步行的一阶差分类似于白噪声。

3. 指数收益率数据 自相关分析

ACF

In [59]:

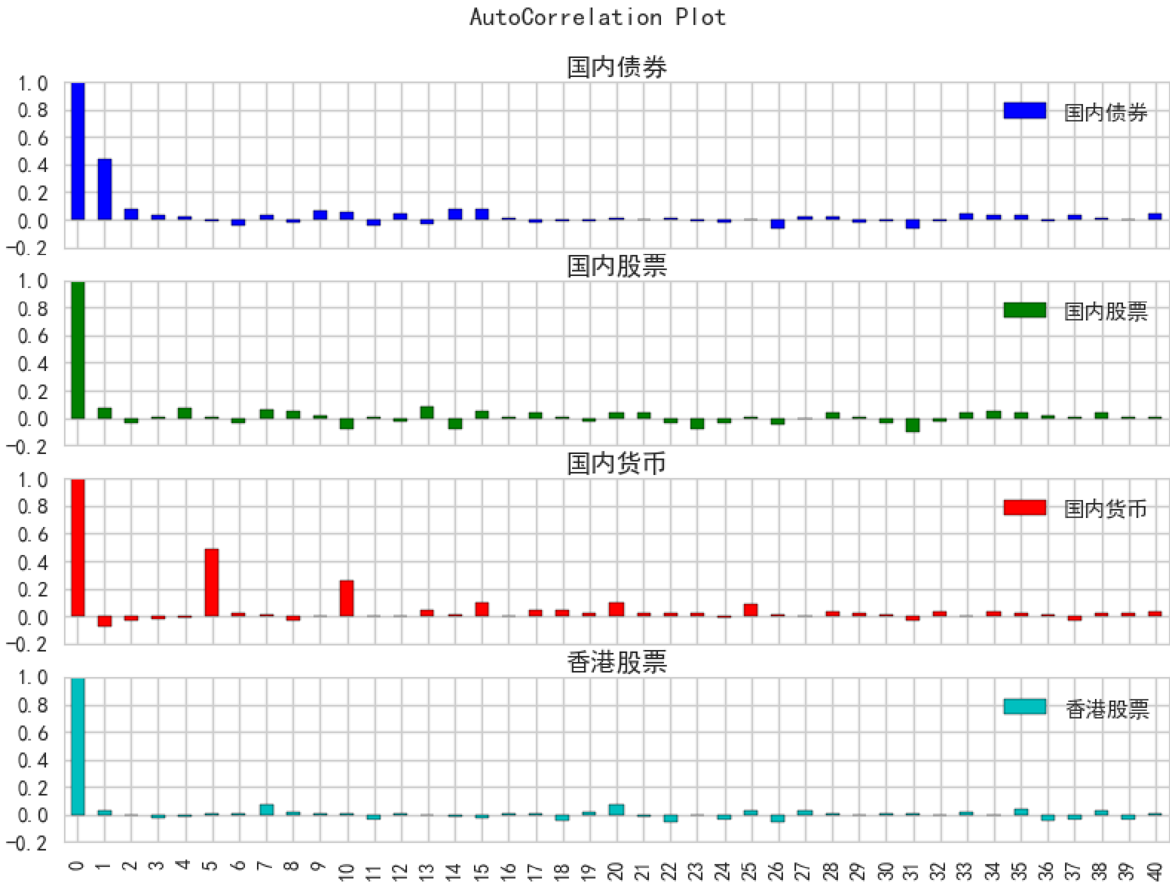
```
acf = pd.DataFrame()
for index_name in indexs_logret.columns:
    acf[index_name] = sm.tsa.stattools.acf(indexs_logret[index_name])
_ = acf.plot(kind='bar',subplots=True,title='AutoCorrelation Plot',figsize=(12,8))
```



PACF

In [60]:

```
pacf = pd.DataFrame()
for index_name in indexs_logret.columns:
    pacf[index_name] = sm.tsa.stattools.pacf(indexs_logret[index_name])
_ = pacf.plot(kind='bar',subplots=True,title='AutoCorrelation Plot',figsize=(12,8))
```

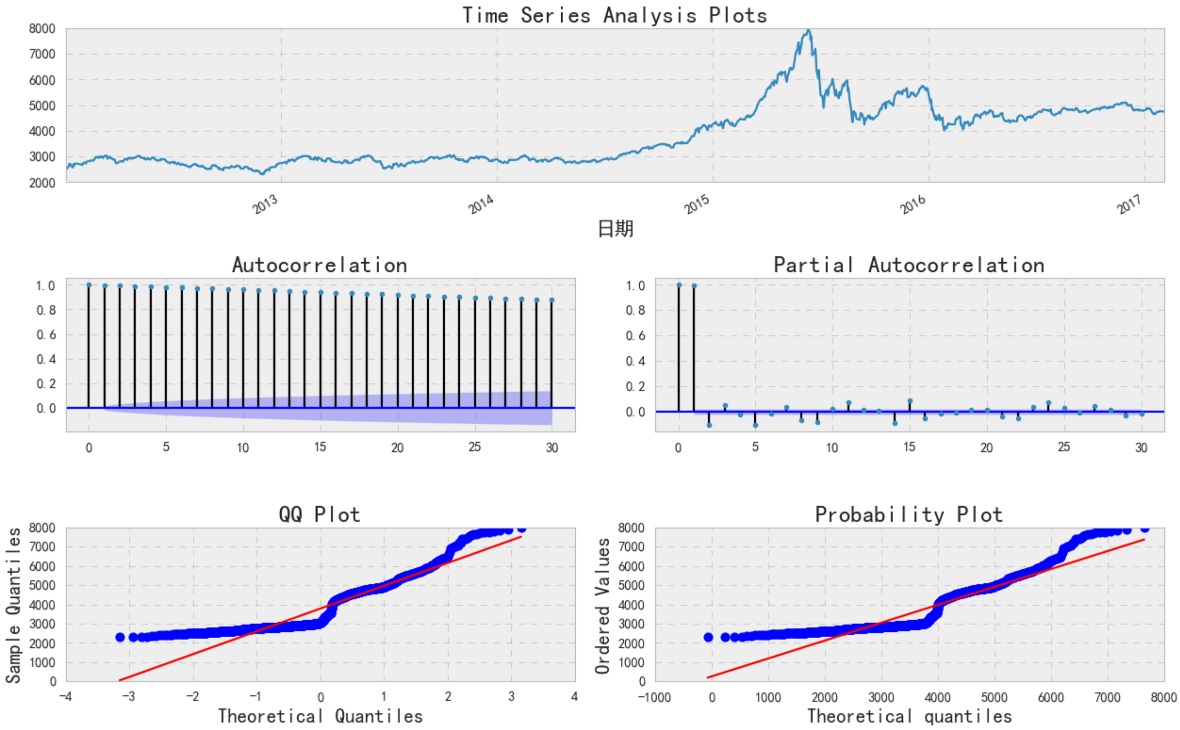


可以看出,
国内货币显示出周期性的自相关性。
国内债卷在时间跨度较小时显示相关性。
国内股票和香港股票则无明显自相关性。

- 国内股票

In [61]:

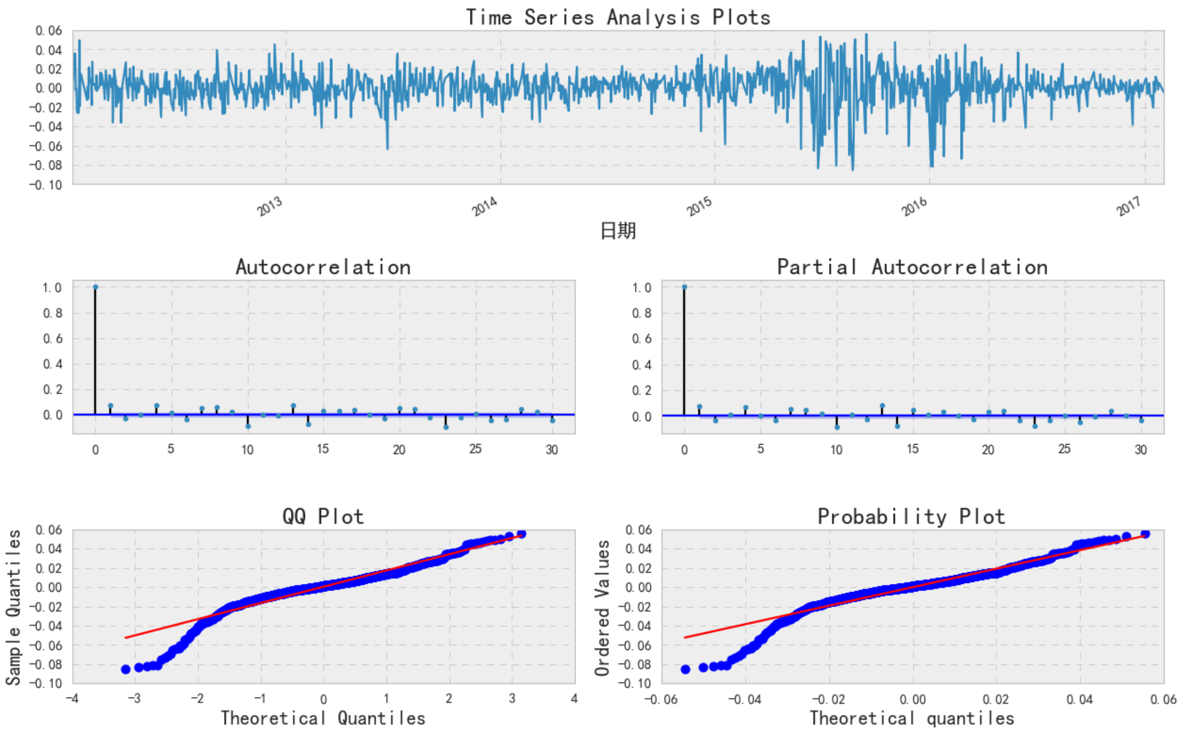
```
tsplot(indexs_sub['国内股票'],lags=30)
```



国内股票收益率

In [62]:

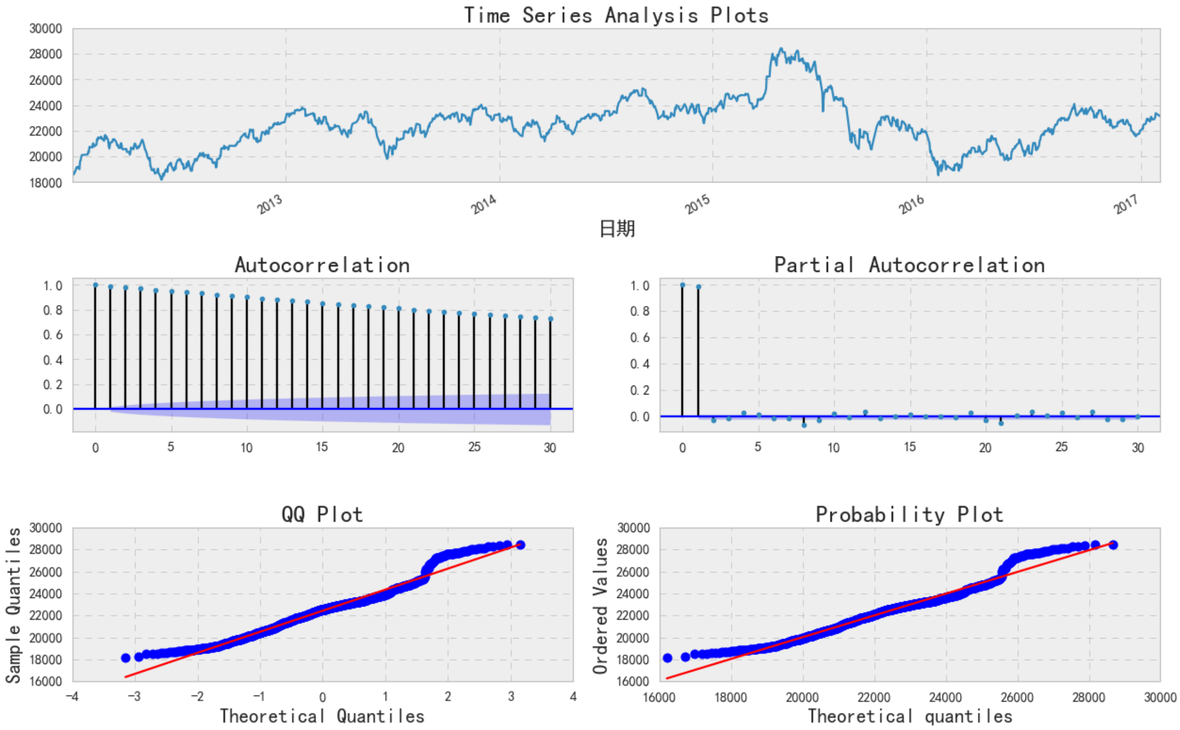
```
tsplot(indexs_logret['国内股票'],lags=30)
```



香港股票

In [63]:

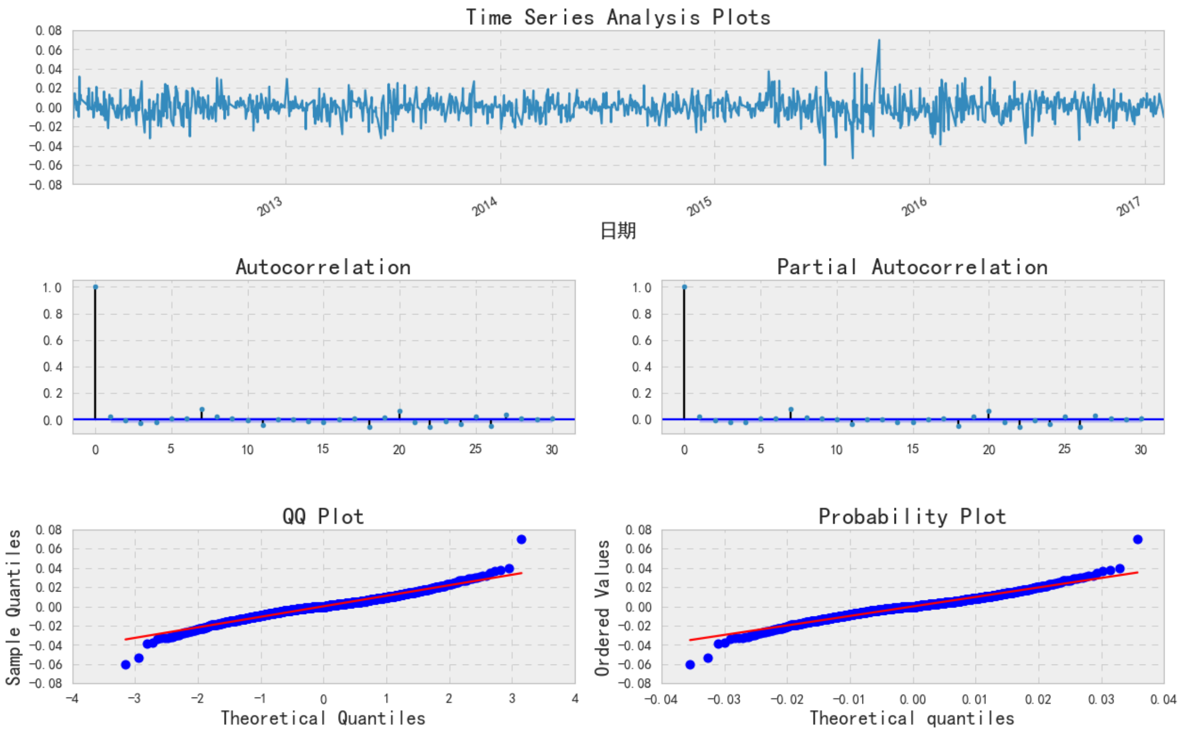
```
tsplot(indexs_sub['香港股票'],lags=30)
```



香港股票收益率

In [64]:

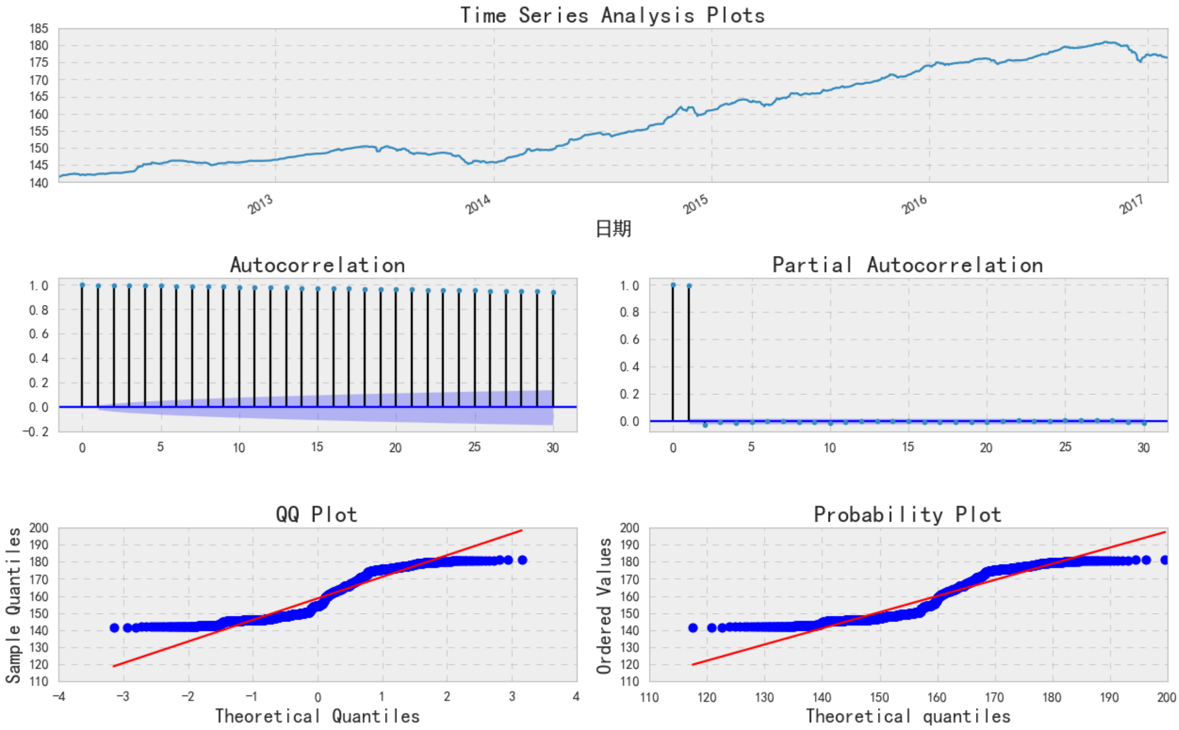
```
tsplot(indexs_logret['香港股票'],lags=30)
```



国内债卷

In [65]:

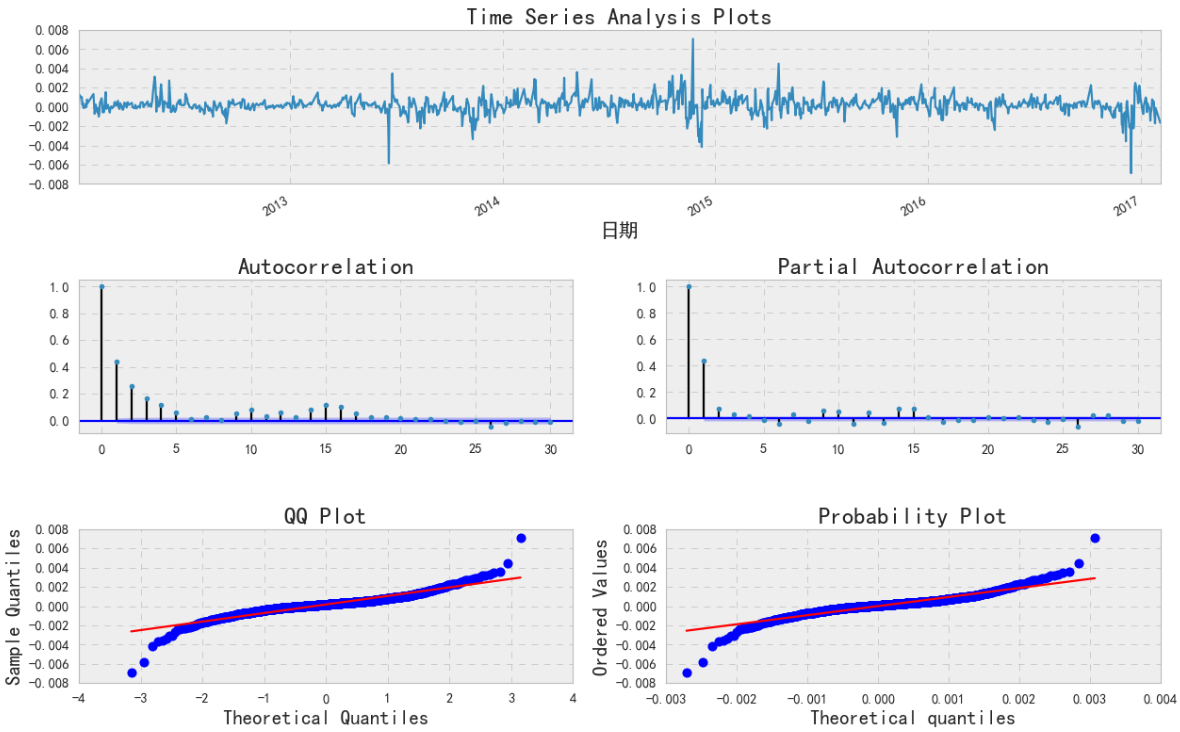
```
tsplot(indexs_sub['国内债券'],lags=30)
```



国内债卷收益率

In [66]:

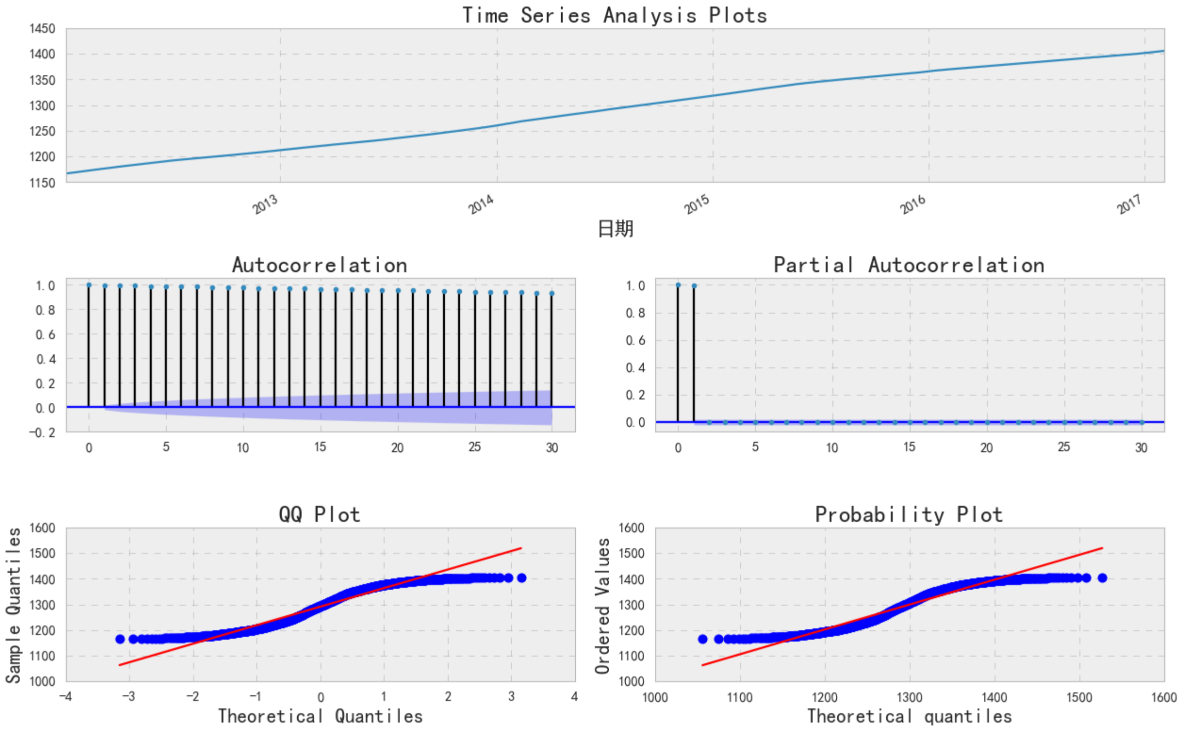
```
tsplot(indexs_logret['国内债券'],lags=30)
```



国内货币

In [67]:

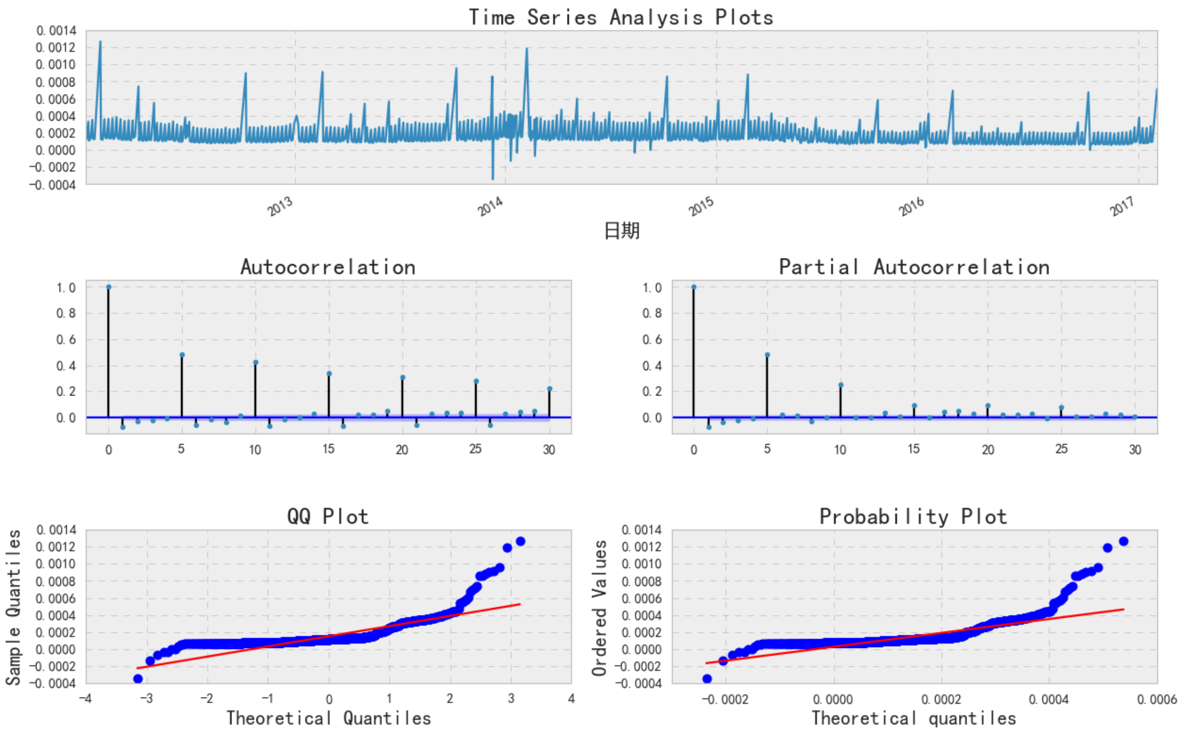
```
tsplot(indexs_sub['国内货币'],lags=30)
```



国内货币收益率

In [68]:

```
tsplot(indexs_logret['国内货币'],lags=30)
```



4. AR Modeling

对国内股票对数收益率进行建模

```
以AIC为优劣准则，所估算的最佳阶数为15.
fpe ( Final Predict Error ) = 0.000276
```

In [69]:

```
import math
result_df = pd.DataFrame(columns=['data','AIC','BIC','LLF','RMSE'])
max_lag = 30
```

In [70]:

```
mdl_ar_gg = smt.AR(indexs_logret['国内股票']).fit(maxlag=max_lag, ic='aic', trend='nc')
est_order_ar_gg = smt.AR(indexs_logret['国内股票']).select_order(
    maxlag=max_lag, ic='aic', trend='nc')
print('{}: best estimated lag order = {}'.format('国内股票',est_order_ar_gg))
print('fpe:',mdl_ar_gg.fpe)
```

```
国内股票: best estimated lag order = 15
fpe: 0.0002758011032636643
```

In [73]:

```
from sklearn.metrics import mean_squared_error
pstart='2017-01-03'
pend = '2017-02-03'
predicted_value = mdl_ar_gg.predict(pstart,pend)
real_value = indexs_logret['国内股票']
ar_gg_rmse = math.sqrt(mean_squared_error(real_value[pstart:pend].values, predicted_value))
```

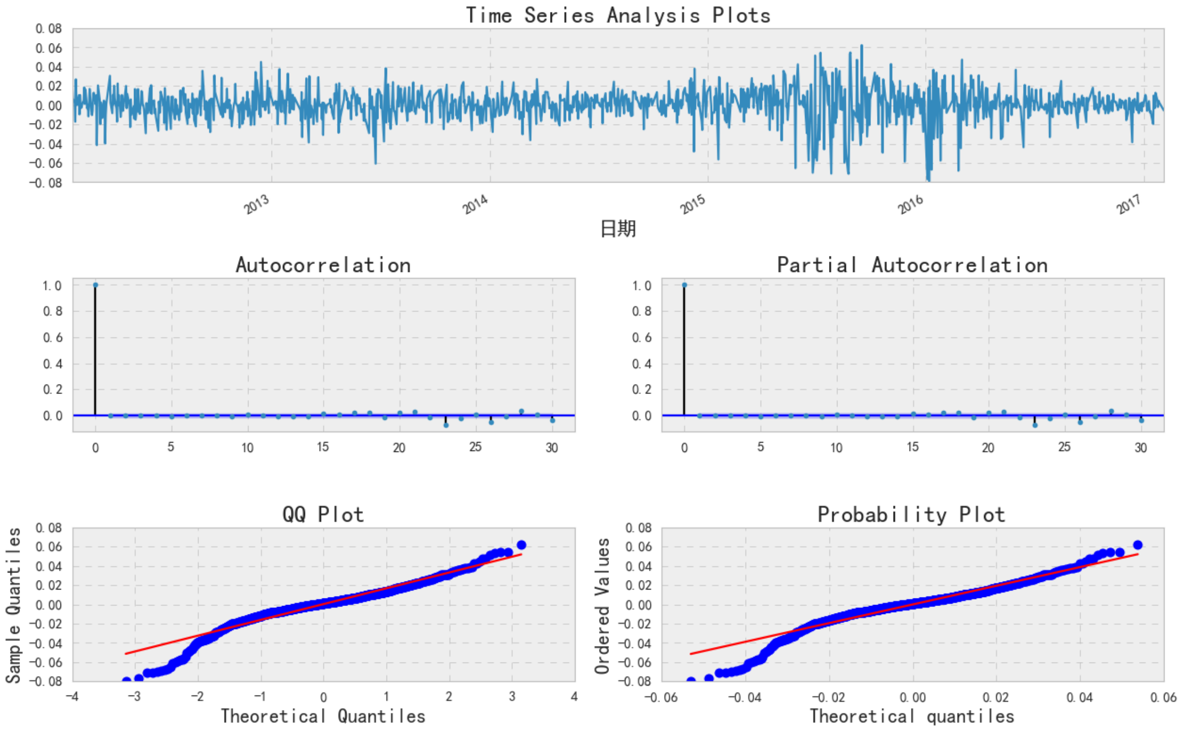
In [ ]:

```
result_df.loc[0] = ['国内股票',mdl_ar_gg.aic,mdl_ar_gg.bic,mdl_ar_gg.llf,ar_gg_rmse]
```

- 残差plot 残差图已无自相关特征
  但残差并非正态分布

In [74]:

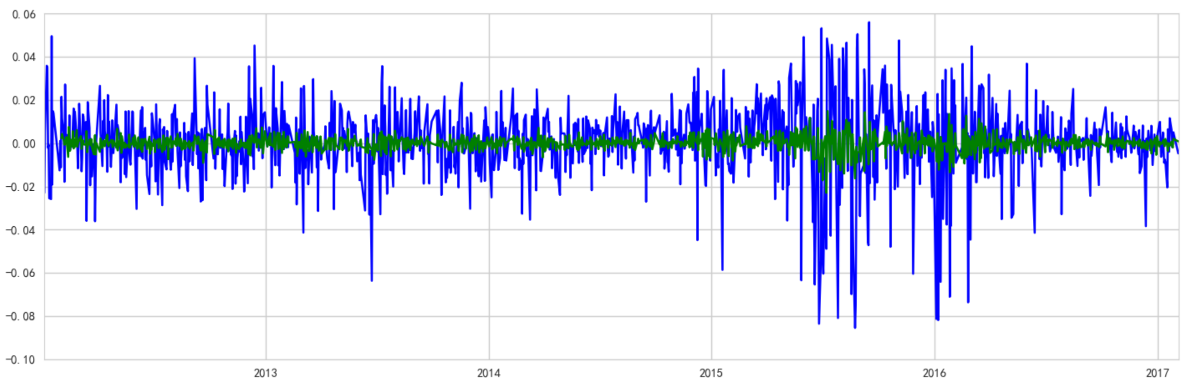
```
tsplot(mdl_ar_gg.resid,lags=30)
```



- AR与真实值比较
  蓝色为真实值

In [77]:

```
fig = plt.figure(figsize=(18,6))
plt.plot(indexs_logret['国内股票'])
plt.plot(mdl_ar_gg.fittedvalues)
plt.tight_layout()
```



对香港股票对数收益率AR建模

In [78]:

```
mdl_ar_xg = smt.AR(indexs_logret['香港股票']).fit(maxlag=max_lag, ic='aic', trend='nc')
est_order_ar_xg = smt.AR(indexs_logret['香港股票']).select_order(
    maxlag=max_lag, ic='aic', trend='nc')
print('{}: best estimated lag order = {}'.format('香港股票',est_order_ar_xg))
print(mdl_ar_xg.fpe)
```

```
香港股票: best estimated lag order = 1
0.00011970079461654166
```

In [79]:

```
predicted_value = mdl_ar_xg.predict(pstart,pend)
real_value = indexs_logret['香港股票']
ar_gg_rmse = math.sqrt(mean_squared_error(real_value[pstart:pend].values, predicted_value))
result_df.loc[1] = ['香港股票',mdl_ar_xg.aic,mdl_ar_xg.bic,mdl_ar_xg.llf,ar_gg_rmse]
```

国内债卷AR建模

In [83]:

```
max_lag = 30
mdl = smt.AR(indexs_logret['国内债券']).fit(maxlag=max_lag, ic='aic', trend='nc')
est_order = smt.AR(indexs_logret['国内债券']).select_order(
    maxlag=max_lag, ic='aic', trend='nc')
print('{}: best estimated lag order = {}'.format('国内债券',est_order))
print(mdl.fpe)
```

```
国内债券: best estimated lag order = 15
6.355019484090043e-07
```

国内货币AR建模

In [84]:

```
max_lag = 30
mdl = smt.AR(indexs_logret['国内货币']).fit(maxlag=max_lag, ic='aic', trend='nc')
est_order = smt.AR(indexs_logret['国内货币']).select_order(
    maxlag=max_lag, ic='aic', trend='nc')
print('{}: best estimated lag order = {}'.format('国内货币',est_order))
print(mdl.fpe)
```

```
国内货币: best estimated lag order = 17
9.379100981226173e-09
```

总结

本文展示了采用Python语言为四个指数时序数据进行自回归分析建模，介绍了AR模型、ACF和PACF等相关概念，并对四个指数时序收益率进行了AR分析和建模。