时间序列分析（1） 基本概念与实战

随风
大数据、人工智能

关注他

188 人赞同了该文章

1 前言

时间序列是指将同一统计指标的数值按其发生的时间先后顺序排列而成的数列。正如人们常说，人生的出场顺序很重要，时间序列中隐藏着一些过去与未来的关系。时间序列分析试图通过研究过去来预测未来。

时间序列分析在工程、金融、科技等众多领域有着广泛的应用。在大数据时代，时间序列分析已经成为 AI 技术的一个分支，通过将时间序列分析与分类模型相结合，更好的应用于数据检测、预测等场景。

作为一篇入门介绍，本文将结合具体数据，讲解时间序列分析中的基本概念。

语言：python3

数据集：余额宝在2014-02-01~2014-07-31期间每日申购的总金额（数据来自天池大赛）

数据下载地址：tianchi.aliyun.com/comp

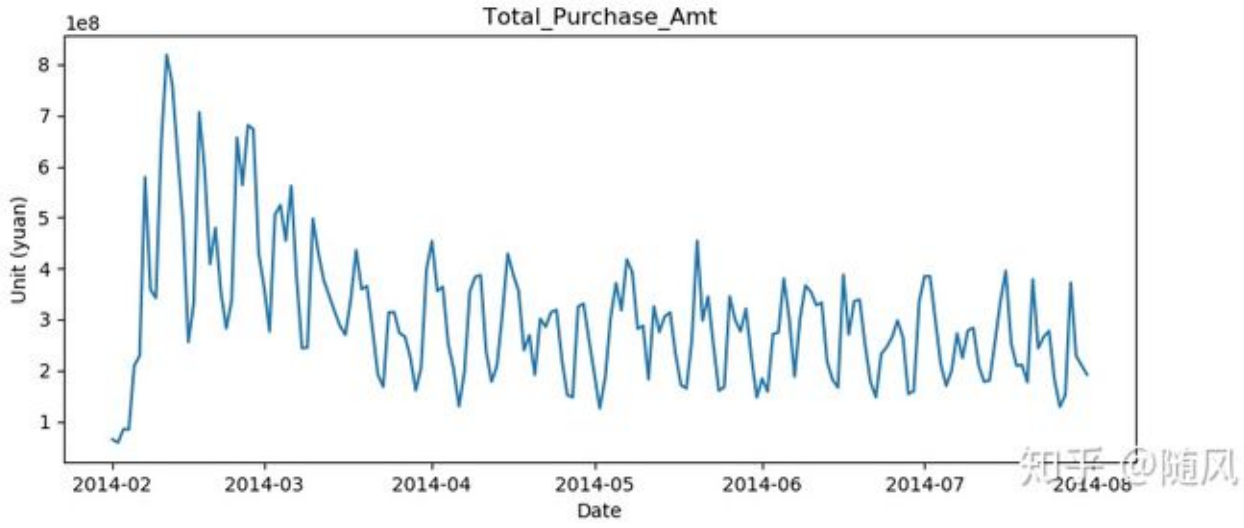
首先我们对user_balance_table.csv 文件进行处理，取出从2014-02-01到2014-07-31每日申购的总金额。

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

def generate_purchase_seq():
    dateparse = lambda dates: pd.datetime.strptime(dates, '%Y%m%d')
    user_balance = pd.read_csv('./user_balance_table.csv', parse_dates=['report_date']
                               index_col='report_date', date_parser=dateparse)

    df = user_balance.groupby(['report_date'])['total_purchase_amt'].sum()

    purchase_seq = pd.Series(df, name='value')
    purchase_seq_201402_201407 = purchase_seq['2014-02-01':'2014-07-31']
    purchase_seq_201402_201407.to_csv(path='./purchase_seq_201402_201407.csv', header=

generate_purchase_seq()
```

2 趋势、季节变化、相关性、随机噪声

我们将生成的 purchase_seq_201402_201407.csv 文件绘制成曲线图。

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

def purchase_seq_display(timeseries):
    graph = plt.figure(figsize=(10, 4))
    ax = graph.add_subplot(111)
    ax.set(title='Total_Purchase_Amt',
           ylabel='Unit (yuan)', xlabel='Date')
    plt.plot(timeseries)
    plt.show()

dateparse = lambda dates: pd.datetime.strptime(dates, '%Y-%m-%d')
purchase_seq_201402_201407 = pd.read_csv('./purchase_seq_201402_201407.csv', parse_dat
                                 index_col='report_date', date_parser=dateparse)

purchase_seq_display(purchase_seq_201402_201407)
```



趋势：趋势是时间序列在某一方向上持续运动，现象是在较长时期内受某种根本性因素作用而形成的总的变动趋势。上图中可以看到从2014年2月到2014年4月，余额宝的申购资金一路下降，这是一个明显的趋势。

季节变化：许多时间序列中包含季节变化，现象是在一年内随着季节的变化而发生的有规律的周期性变动。上图中肉眼很难看出时间序列随季节变化，第 3 小结将通过时间序列分解（STL）来展示序列中的季节变化。

序列相关性：时间序列的一个最重要特征是序列相关性，又称为自相关性。上图中可以看到，数据之间存在一定的正相关与负相关。例如某天的数据上升，它的前一天或者后一天也上升或者下降。**自相关性是时间序列可以预测未来的前提（序列中存在的规律），如果没有自相关性，就变成了白噪声（无规律）。**

随机噪声：它是时间序列中除去趋势、季节变化和自相关性之后的剩余随机扰动。由于时间序列存在不确定性，随机噪声总是夹杂在时间序列中，致使时间序列表现出某种震荡式的无规律运动。

时间序列分析的核心就是挖掘该时间序列中的自相关性，第 5 小结将详细介绍时间序列的自相关性。

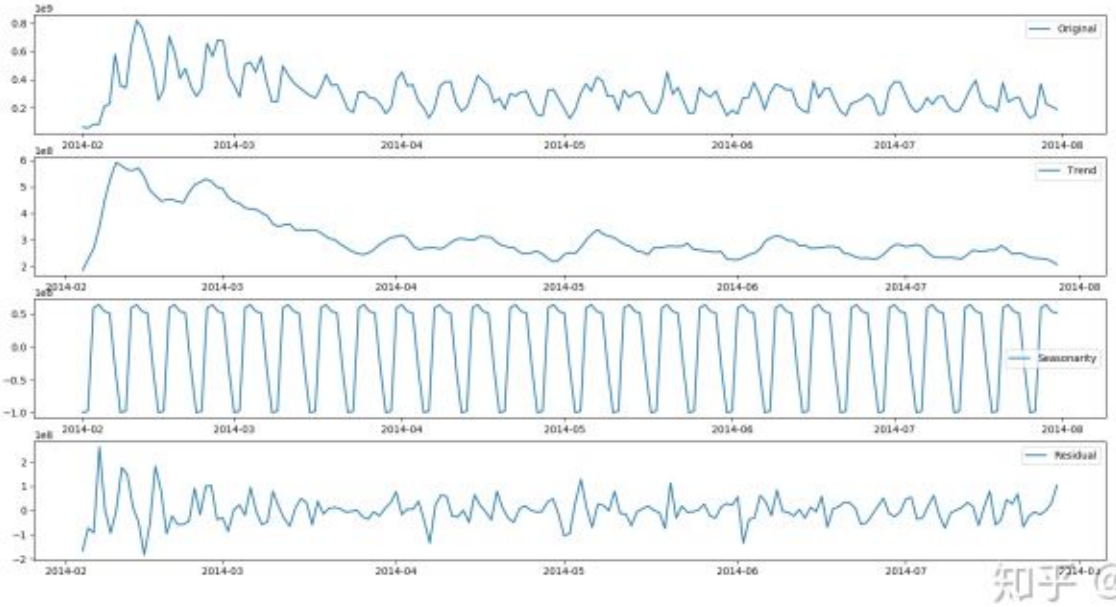
3 时间序列分解

大多数情况下，业务分析人员很难从高高低低的曲线中找到数据的规律。**通过时间序列分解，可以帮助大家从时间序列的波动中挖掘信息。**

我们将生成的 purchase_seq_201402_201407.csv 文件进行时间序列分解，这里采用 STL 算法进行时间序列分解。（看不清楚可以将图片放大）

```
import matplotlib.pyplot as plt
import pandas as pd
from statsmodels.tsa.seasonal import seasonal_decompose

def decomposing(timeseries):
    decomposition = seasonal_decompose(timeseries)
    trend = decomposition.trend
    seasonal = decomposition.seasonal
    residual = decomposition.resid

    plt.figure(figsize=(16, 12))
    plt.subplot(411)
    plt.plot(timeseries, label='Original')
    plt.legend(loc='best')
    plt.subplot(412)
    plt.plot(trend, label='Trend')
    plt.legend(loc='best')
    plt.subplot(413)
    plt.plot(seasonal, label='Seasonarity')
    plt.legend(loc='best')
    plt.subplot(414)
    plt.plot(residual, label='Residual')
    plt.legend(loc='best')
    plt.show()
```

```
dateparse = lambda dates: pd.datetime.strptime(dates, '%Y-%m-%d')
purchase_seq_201402_201407 = pd.read_csv('./purchase_seq_201402_201407.csv', parse_dat
                                index_col='report_date', date_parser=dateparse)

decomposing(purchase_seq_201402_201407)
```



四个序列从上到下依次表示：原始序列、趋势序列、季节序列、残差序列。

上图中可以看出，从2014年2月开始，余额宝的申购总金额是呈现出一个逐步下降的趋势（如果大家结合余额宝和银行的利率曲线可以了解到，其实在这段时间内，相比于2013年末到2014年初，余额宝和银行的利息都降低了不少）。原始序列具有强烈的季节变化（或者称为周期性），几乎是每个月四个周期，也就是以星期为一个周期波动（这一点也很好理解，人们对于资金的存取习惯，跟工作日与节假日有密切的关系）。最后一项是残差序列，也就是原始序列中去除趋势和季节变化后的序列，这一部分是序列中的不稳定因素，具有一定的随机性，需要做进一步分析。

时间序列分解的成功与否，取决于两个因素：一是数据序列本身是隐藏着规律的，不可预测的部分只是其中的一小部分；二是分解的方法要合适，尤其是周期的判断要准确。

4 时间序列的平稳性

经典回归分析的一个重要假设是：数据是平稳的。非平稳数据往往导致“虚假回归”，表现为两个没有任何因果关系的变量，却有很高的相关性。比如在时间序列中，本来没有自相关性的两个时间点，产生了相关性。**因此平稳性是时间序列分析的基础。**

平稳性定义（弱平稳）：

1、均值 u 是与时间 t 无关的常数

2、对于任意时刻 t 和任意时间间隔 k，时间序列 z_t 与 z_{t-k} 的自协方差 r_k 只与 k 有关，与 t 无关。

从统计学的角度讲，平稳性的要求就是对于一个时间序列（分布未知），**这个时间序列的取值一定满足一个确定的分布。**比如我们知道任意一个时间点的取值只能为集合（1, 2, 3）中的某一个数字，取值概率分别为（0.3，0.3，0.4），那么我们认为这个时间序列是平稳的。但是如果在某一个时间点出现了数字4，则从该时刻开始，我们认为时间序列就不是平稳的了。

平稳性对于我们分析时间序列至关重要。如果一个时间序列不是平稳的，通常需要通过差分的方式将其转化为平稳时间序列。

> 举一个例子，假设我们想知道2017 年 5 月 16 日这天上证指数收益率的均值是多少，而且我们假设它是来自一个未知的分布。也许你会马上说“查一下 Wind 不就知道了？上证指数那天的收益率是 0.74%”。注意，**0.74% 这个数值仅仅是那天上证指数未知收益率分布的一个实例**！它不是均值，因此从时间序列分析的角度来说仅仅知道 0.74% 远远不够。
> **对于一般的未知概率分布，只要通过进行大量重复性实验，就可以有足够多的独立观测点来进行统计推断（计算均值和方差这些统计量）。**按照这个思路，我们必须把 2017 年 5 月 16 日这一天经历许多遍，得到许多个那天的收益率观测值，然后用这些观测值计算出收益率的均值。
> **不幸的是，历史只发生一次，时间也一去不复返，我们只能实实在在的经历一遍 2017 年 5 月 16 日，只能得到一个收益率的观测点，即 0.74%。**
> **然而，如果我们假设上证指数的收益率序列满足弱平稳，就柳暗花明了。根据弱平稳假设，上证指数的日收益率序列 $\{r_t\}$ 的均值是一个与时间无关的常数，即 $\mathrm{E}[r_t]=\mu$ 。这样便可以利用一段时间的历史数据来计算出日收益率的均值。**比如我们可以对上证指数在 2017 年交易日的日收益率序列取平均，把它作为对总体均值 μ 的一个估计。根据弱平稳性，该平均值也正是 2017 年 5 月 16 日的收益率均值。
> 同样的道理，在弱平稳的假设下，可以根据历史数据方便的对时间序列的诸多统计量进行推断。

对于一个时间序列，如何确定它是否满足平稳性要求？通常采用 ADF 检验。

先来看两个特殊的时间序列：

对于时间序列 $\{w_t:t=1,\cdots,n\}$。如果该序列的成分 w_t 满足均值为0，方差 σ^2 ，且对于任意的 $k\geq 1$ ，自相关系数 ρ_k 均为0，则称该时间序列为一个离散的白噪声。

$$X_t=w_t,w_t(0,\sigma^2)$$

对于时间序列 $\{x_t\}$ ，如果它满足 $x_t=x_{t-1}+w_t$ ，其中 w_t 是一个均值为0、方差为 σ^2 的白噪声，则序列 $\{x_t\}$ 为一个随机游走。

$$X_t = X_{t-1} + w_t, X_t(0, t\sigma^2)$$

显然对于白噪声序列，它满足正态分布，均值与方差都是与时间t无关的函数，它满足平稳性要求。对于随机游走，它的均值为0，方差与时间t有关，他不满足平稳性要求。

ADF 大致的思想就是基于随机游走的，对 X_t 回归，如果发现p=1，说明序列满足随机游走，就是非平稳的。

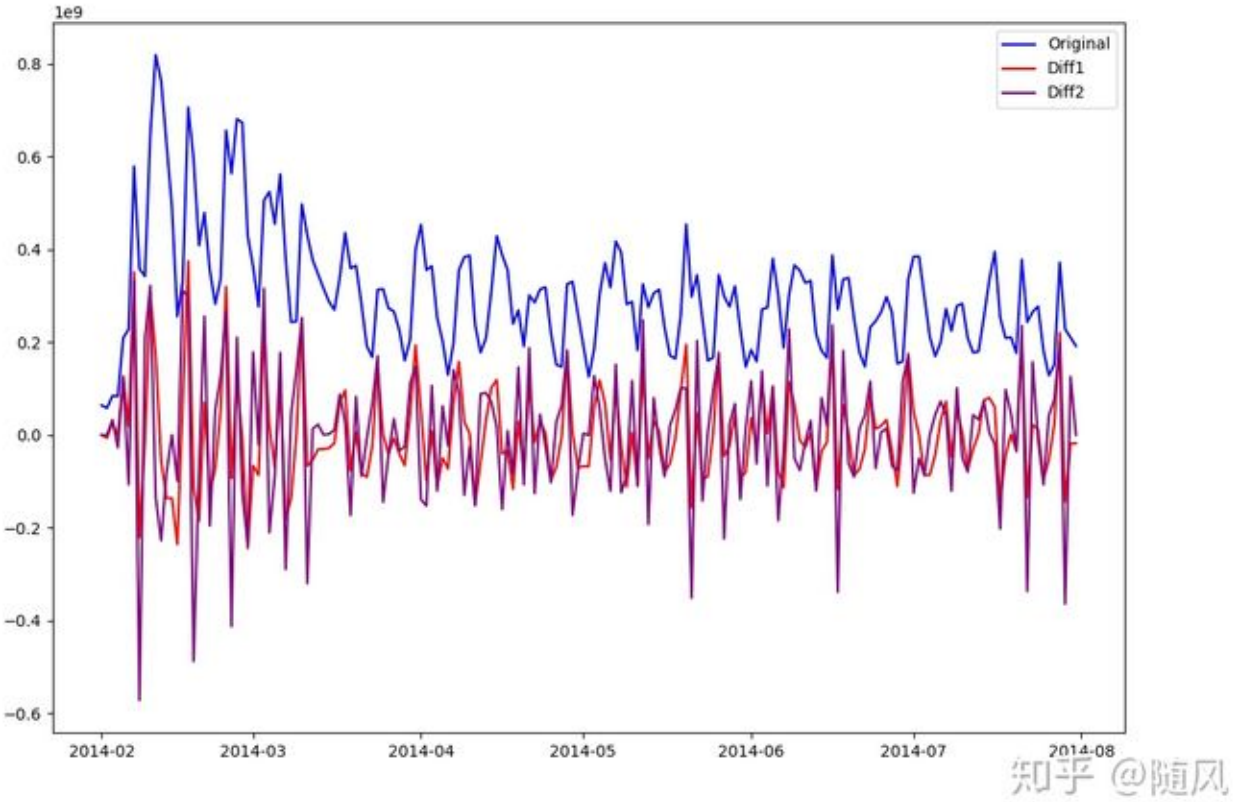
$X_t = X_{t-1} + u_t$ 随机游走，非平稳
$X_t = pX_{t-1} + u_t$ 对待估计的时间序列回归，如果发现p=1，则称 X_t 有一个单位根

通过上式判断 X_t 是否具有单位根，来确定是否平稳。

我们将生成的 purchase_seq_201402_201407.csv 文件进行一阶差分与二阶差分，并分别计算它们的ADF值。

```
import matplotlib.pyplot as plt
import pandas as pd
import statsmodels.api as sm
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.stattools import adfuller as ADF


def diff(timeseries):
    timeseries_diff1 = timeseries.diff(1)
    timeseries_diff2 = timeseries_diff1.diff(1)

    timeseries_diff1 = timeseries_diff1.fillna(0)
    timeseries_diff2 = timeseries_diff2.fillna(0)



    timeseries_adf = ADF(timeseries['value'].tolist())
    timeseries_diff1_adf = ADF(timeseries_diff1['value'].tolist())
    timeseries_diff2_adf = ADF(timeseries_diff2['value'].tolist())

    print('timeseries_adf : ', timeseries_adf)
    print('timeseries_diff1_adf : ', timeseries_diff1_adf)
    print('timeseries_diff2_adf : ', timeseries_diff2_adf)

    plt.figure(figsize=(16, 12))
    plt.plot(timeseries, label='Original', color='blue')
    plt.plot(timeseries_diff1, label='Diff1', color='red')
    plt.plot(timeseries_diff2, label='Diff2', color='purple')
```

```
    plt.legend(loc='best')
    plt.show()

dateparse = lambda dates: pd.datetime.strptime(dates, '%Y-%m-%d')
purchase_seq_201402_201407 = pd.read_csv('./purchase_seq_201402_201407.csv', parse_dat
                                 index_col='report_date', date_parser=dateparse)


diff(purchase_seq_201402_201407)
```



从上图中可以看出，经过一阶差分，原序列的趋势（**有趋势一定是非平稳的**）被消除了，整个序列基本围绕确定的均值震荡。经过二阶差分，与一阶差分相比，只是在震荡幅度上扩大了，因此对于该序列，采用一阶差分比较合适。一**般情况下，采用一阶、二阶差分就可以使序列变得平稳。**

再来看一下ADF计算的结果，首先解释一下主要参数的含义，以 timeseries_adf 为例：

第一个参数 -1.742812211577193： T检验，假设检验值。

第二个参数 0.4092001716091834： P-value：假设检验结果。

第五个参数 {'10%': -2.5760826967621644, '1%': -3.470126426071447, '5%': -2.8790075987120027} : 不同程度拒绝原假设的统计值。

```
timeseries_adf :  (-1.742812211577193, 0.4092001716091834, 13, 167, {'10%': -2.5760826
timeseries_diff1_adf :  (-10.06109433503844, 1.332927648174115e-17, 12, 168, {'10%': -
timeseries_diff2_adf :  (-8.369488957177882, 2.686267936266921e-13, 14, 166, {'10%': -
```

如何确定该序列是否平稳呢?

1%、%5、%10不同程度拒绝原假设的统计值和 ADF 假设检验值比较，ADF 假设检验值同时小于1%、5%、10%即说明非常好地拒绝该假设。
P-value是否非常接近0。

本数据中，原序列的 ADF 假设检验值为-1.742812211577193， 大于三个level的统计值，所以是非平稳的。而一阶差分序列的 ADF 假设检验值为-10.06109433503844，小于三个level的统计值，再来看P-value的值为1.332927648174115e-17，接近0，所以是平稳的。

ADF检验的原假设是存在单位根，只要这个统计值是小于1%水平下的数字就可以极显著的拒绝原假设，认为数据平稳。注意，ADF值一般是负的，也有正的，但是它只有小于1%水平下才能认为是及其显著的拒绝原假设。
对于ADF结果在1% 以上 5%以下的结果，也不能说不平稳，关键看检验要求是什么样子的。

5 自相关系数（ACF）、偏自相关系数（PACF）

首先介绍一下协方差，假设两个随机变量 **X** 和 **Y** 满足未知的概率分。**X** 和 **Y** 的**协方差**为：

$$\operatorname{Cov}(X,Y) = \operatorname{E}[(X - \mu_X)(Y - \mu_Y)]$$

其中， μ_X 和 μ_Y 分别为 **X** 和 **Y** 的**均值**。

在实际中，由于总体的概率分布未知，我们只能通过 **X** 和 **Y** 的观测值来计算**样本均值**。假设我们各有 **X** 和 **Y** 的观测值 n 个，则它们的**样本协方差**为：

$$\frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})$$

其中， $\bar{X}$ 和 $\bar{Y}$ 为 **X** 和 **Y** 的样本均值。上面公式中右侧之所以除以 n-1 而非 n 的原因是，这么做可以保证样本协方差是总体协方差的一个**无偏估计**。

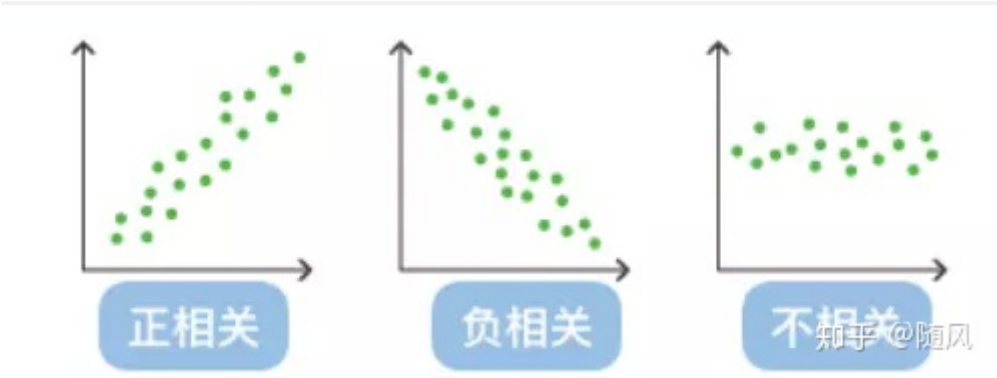
对于**无偏估计**，大致的意思就是我们需要计算总体的均值和方差，然而我们拿到的数据是对总体的采样，因此得到的均值与方差是对采样样本而言的，而不是总体的均值与方差。可以证明样本的方

差比总体方差小 $\frac{\sigma^2}{n}$ ，因此在计算方差的时候，分母取 $n-1$ 而不是 n 。具体细节可以参考这篇文章：matongxue.com/madocs/60

假设随机变量 **X** 和 **Y** 分别构成序列：X = [x1, x2, ..., xn] Y = [y1, y2, ..., yn]

在二维空间中，构成的点为{ (x1, y1), (x2, y2), ... , (xn, yn) }

将它们映射到二维空间中构成散点图如下：



对于图（1），当 X 增大或减小，则 Y 增大或减小，也就是 X 与 Y 的变化趋势相同， 称 X 与 Y 正相关。

对于图（2），当 X 增大或减小，则 Y 减小或增大，也就是 X 与 Y 的变化趋势相反， 称 X 与 Y 负相关。

对于图（3），当 X 增大或减小，Y 没有与之对应的变化趋势，称 X 与 Y 不相关。

然而用协方差来衡量变量之间的相关性是有问题的，比如计算变量 **X** 和 **Y** 的协方差为100。我们考虑变量 10***X** 与 10***Y** 之间的协方差，应该为 10000，相关性提升了100倍！显然我们知道 **X** 和 **Y** 之间的相关性与10***X** 与 10***Y** 之间的相关性是一样的，只是两个变量都放大了10倍。

上面的例子说明使用协方差衡量变量相关性的**致命缺点：协方差是有量纲的，因此它的大小受随机变量本身波动范围的影响。最简单的做法就是用变量自身的波动对协方差进行标准化。相关系数便由此得来。**

令 ρ 表示 **X** 和 **Y** 的**总体相关系数**，它的定义为：

$$\rho(X,Y)=\frac{\mathrm{E}[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X\sigma_Y}=\frac{\mathrm{Cov}(X,Y)}{\sigma_X\sigma_Y}$$

其中 σ_X 和 σ_Y 分别为 **X** 和 **Y** 的**总体标准差**。通过使用 **X** 和 **Y** 的标准差对它们的协方差归一化， ρ 的取值范围在 -1 到 +1 之间，即 [-1, +1]：

$\rho(X,Y)=1$ **表示X 和 Y之间存在确切的线性正相关；**
$\rho(X,Y)=0$ **表示X 和 Y 之间不存在任何线性相关性；**
$\rho(X,Y)=-1$ **表示X 和 Y之间存在确切的线性负相关。**

值得一提的是，**相关系数仅仅刻画 X 和 Y 之间的线性相关性；它不描述它们之间的（任何）非线性关系。**在实际中，由于总体的概率分布未知，我们只能通过 **X** 和 **Y** 的观测值来计算 **X** 和 **Y** 的**样本相关系数**：

$$\hat{\rho}(X,Y)=\frac{\sum_{i=1}^{n}(X_i-\bar{X})(Y_i-\bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i-\bar{X})^2\sum_{i=1}^{n}(Y_i-\bar{Y})^2}}$$

时间序列的特点是一维，因此如果借用上面的指标衡量，有些不太适宜。**根据时间序列的特点，形成了自相关函数、偏自相关函数。**看到前面都加了一个“自”，原因是时间序列没法在找到一个别的数据和自己来进行比较；**只能自己和自己来比较，自己和自己慢几拍（滞后期）的这些数据进行比较，所以加入了一个“自”。**

自相关系数（ACF）

自相关系数度量的是同一事件在两个不同时期之间的相关程度，形象的讲就是度量自己过去的行为对自己现在的影响。

$$ACF(k)=\Sigma_{t=k+1}^{n}\frac{(Z_t-\bar{Z})(Z_{t-k}-\bar{Z})}{\Sigma_{t=1}^{n}(Z_t-\bar{Z})^2}$$

自相关（autocorrelation），也叫序列相关，是一个信号于其自身在不同时间点的相关度。非正式地来说，它就是两次观察之间的相似度对它们之间的时间差的函数。它是找出重复模式（如被噪声掩盖的周期信号），或识别隐含在信号谐波频率中消失的基频的数学工具。它常用于信号处理中，用来分析函数或一系列值，如时域信号。

偏自相关系数（PACF）

根据ACF求出滞后k自相关系数 $ACF(k)$ **时，实际上得到并不是Z(t)与Z(t-k)之间单纯的相关关系。**

因为Z(t)同时还会受到中间k-1个随机变量Z(t-1)、Z(t-2)、……、Z(t-k+1)的影响，而这k-1个随机变量又都和z(t-k)具有相关关系，所以自相关系数里面实际掺杂了其他变量对Z(t)与Z(t-k)的影响。

为了能单纯测度Z(t-k)对Z(t)的影响，引进偏自相关系数（PACF）的概念。对于平稳时间序列{Z(t)}，所谓滞后k偏自相关系数指在给定中间k-1个随机变量Z(t-1)、Z(t-2)、……、Z(t-k+1)的条

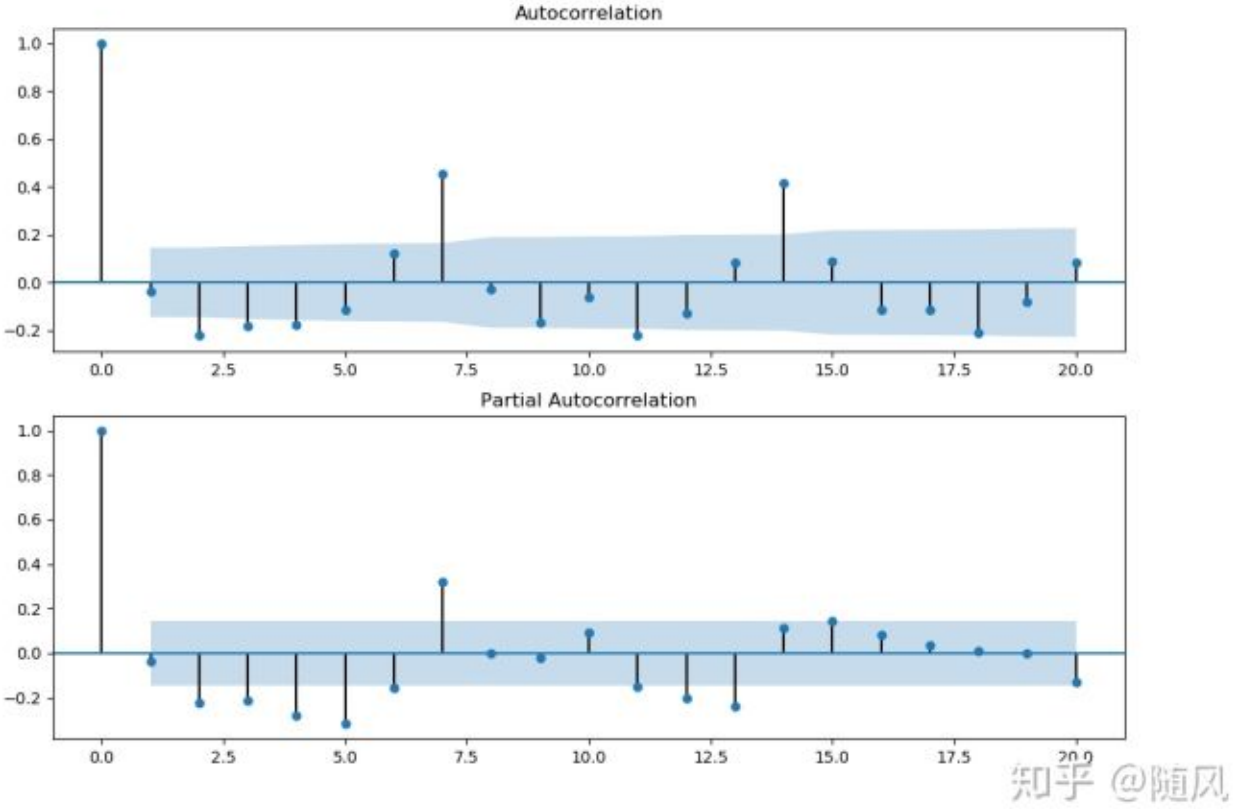
件下，或者说，在剔除了中间k-1个随机变量Z(t-1)、Z(t-2)、……、Z(t-k+1)的干扰之后，Z(t-k)对Z(t)影响的相关程度。

$$PACF(k) = \frac{E(Z_t - EZ_t)(Z_{t-k} - EZ_{t-k})}{\sqrt{E(Z_t - EZ_t)^2}\sqrt{E(Z_{t-k} - EZ_{t-k})^2}} = \frac{cov[(Z_t - \bar{Z}_t), (Z_{t-k} - \bar{Z_{t-k}})]}{\sqrt{var(Z_t - \bar{Z}_t)}\sqrt{var(Z_{t-k} - \bar{Z_{t-k}})}}$$

计算某一个要素对另一个要素的影响或相关程度时，把其他要素的影响视为常数，即暂不考虑其他要素的影响，而单独研究那两个要素之间的相互关系的密切程度时，称为偏相关。

根据第 4 小节关于平稳性的讨论，为了防止“虚假回归”，需要对 purchase_seq_201402_201407.csv 文件进行一阶差分后，再讨论序列的自相关性。

```
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.stattools import adfuller as ADF
import statsmodels.api as sm
from statsmodels.tsa.seasonal import seasonal_decompose


def autocorrelation(timeseries, lags):
    fig = plt.figure(figsize=(12, 8))
    ax1 = fig.add_subplot(211)
    sm.graphics.tsa.plot_acf(timeseries, lags=lags, ax=ax1)
    ax2 = fig.add_subplot(212)
    sm.graphics.tsa.plot_pacf(timeseries, lags=lags, ax=ax2)
    plt.show()

dateparse = lambda dates: pd.datetime.strptime(dates, '%Y-%m-%d')
purchase_seq_201402_201407 = pd.read_csv('./purchase_seq_201402_201407.csv', parse_dat
                                         index_col='report_date', date_parser=datepars

purchase_seq_201402_201407_diff1 = purchase_seq_201402_201407.diff(1)
purchase_seq_201402_201407_diff1 = purchase_seq_201402_201407_diff1.fillna(0)
autocorrelation(purchase_seq_201402_201407_diff1, 20)
```



对时间序列建模，最重要的就是挖掘出该序列中的不同间隔 k 的自相关性。相关图可以帮助我们判断模型是否合适。这是因为时间序列的特征中往往包括相关性和随机噪声。**如果模型很好的捕捉了自相关性，那么原始时间序列与模型拟合的时间序列之间的残差应该近似的等于随机噪声。**残差序列自然也是一个时间序列，因此可以对它画出相关图。**一个标准随机噪声的自相关满足 $\rho_0 = 1$ 以及 $\rho_k = 0$ ，$k = 1, 2, 3, \cdots$ ，即对于任意不为 0 的间隔，随机噪声的自相关均为 0。**