上海对外经贸大学

2020 -2021 第二学期

《应用时间序列分析》

课 程 论 文

论文名称：	基于时间序列对航班乘客数量季节趋势的研究
学　　院：	统计与信息学院
专　　业：	数据科学与大数据技术
学　　号：	18076014
学生姓名：	陈郁欣
课程教师：	刘永辉,李睿
课程编号：	354.004.2[01]

2021　年　6 月

目录

基于时间序列对航班乘客数量季节趋势的研究

摘要

根据不同时段的航班乘客数量，合理安排航班班次，可以有效节省资源、避免时间上的浪费，从而促进世界经济、政治等领域的有效运作。本文根据某国际航班从 1949 年 1 月到 1960 年 12 月每月的航班乘客量进行研究分析，，绘制数据多方面趋势图，分析季节性的变动情况及规律，研究不同阶数的季节性自回归移动平均模型的建立与应用，结果表明：$SARIMA(0,1,1)\times(1,1,1)_{12}$模型可以很好的描述一年内航班乘客的变化趋势，误差在人数范围内的约 4%波动。在排除其他因素影响的情况下，使用该模型可以很好地预测未来乘客变动情况。

关键词： 航班乘客 时间序列 SARIMA 模型

一、 引言

航班联络着不同国家与地区，在促进世界间的交流上有着显著的意义，通过对乘客在短时间内进行长距离的运输，有利于节约时间资源，发挥着交通运输的重要作用，是时代发展的重要组成部分。由于飞机运作及维护在资源上的消耗比较大，所以对每个时间段航班乘客数量进行合理预测，有助于提前对飞机资源进行合理配置、对航班班次进行有效安排，避免财力上的浪费，也可以避免对旅客和物资在运输上的延误。

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，通常是在相等间隔的时间段内依照给定的采样率对某种潜在过程进行观测的结果，可以根据已有的历史数据来对未来进行预测。

本文将从时间序列出发，使用真实航班乘客数据进行季节趋势的研究。

二、 问题陈述

航班可能会由于安排冗余、缺失等安排不合理的问题出现同一时间空闲航班过多供大于求或者过少供不应求的问题，这不仅会给航班运作者带来财力上的损失，影响乘客的时间安排，同样也可能对世界的正常运作带来影响。同时特定时间段内乘客对航班的乘坐呈周期变化的特点，运作者可以通过分析不同时间内乘客数量变化的相关性，合理安排航班，达到资源利用最大化的效果。

现需根据航班乘客数量数据，设计一种研究数量季节性趋势并对未来进行预测的时间序列模型。

三、 问题分析

时间序列是某个统计指标按照时间进行先后排序的序列，随着科技的不断发展，数据逐渐海量化，故对于时间序列的分析与研究也越来越深入。基于时间序列，观测值之间的独立性是不成立的，不同时间之间具有一定的相关性，且大多数时间序列具有某些形式的季节性趋势，即特定时间的范围有着相似的变化，比如观察羽绒服销售数据会发现每个冬季的销售额更高等。考虑到一年之中可能会

出现旅游旺季和淡季，这就在某种程度上决定了一部分客流量突然增加或者减少的原因，所以可以首先对数据的趋势进行观察，并使用差分及季节差分等方法，对数据进行平稳化，从而对平稳化后的数据建立ACF和PACF图识别模型类型和阶数，在诊断模型后即可进行相关的预测，得出数据的具体趋势。

四、 模型建立与求解

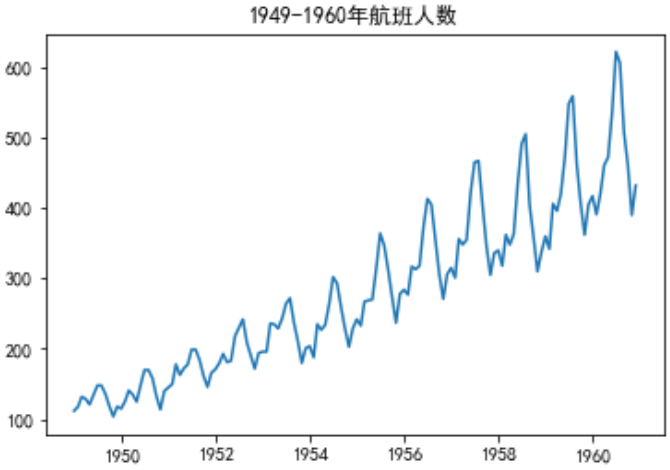
4.1 数据准备

为避免数据不真实导致建立的模型难以应用到实际生活中的情况，本文使用Analytics Vidhya平台的航空乘客数据集，此数据集包含了国际航班从1949年1月1日至1960年12月1日的月乘客人数（单位：千人），下表给出了部分数据表示形式。

表1：股票数据

时间	乘客数
1956/1/1	284
1956/2/1	277
1956/3/1	317
1956/4/1	313
1956/5/1	318
1956/6/1	374

图1：1946年-1949年月航班人数



由图1可知，从1949年开始，人数总体呈上升的情况，在1950年左右达到最低点，且随着时间的增长，可以观察到波动幅度越来越大，可能是由于科技的增长以及接触航空飞行的人越来越普遍，所以趋近于呈现在特定时间内大量乘坐或

大量不乘坐的局面。由于航班乘客数量的书简序列数据波动很大，缺乏平稳性，可能会减少时间序列模型预测的准确性，因此需要首先对数据进行平稳化。此外，时间序列图像在每一年内波动比较规律，说明存在季节性的因素。

4.2 数据分解

因为时间序列可以分解为：长期趋势因素、季节变动因素、循环变动因素和不规则变动因素，即

$$Y_t = f(T_t, S_t, C_t, I_t)$$

本文中，使用乘法模型对序列进行分解。

$$Y_t = T_t \times S_t \times C_t \times I_t$$

具体分解方法如下：

(1) 首先运用移动平均法，对每年十二个月的数据相加求平均值，得到没有季节性的年平均数。

$$X_i = (x_{i1} + x_{i2} + x_{i3} + \cdots + x_{i12})/12$$

并且由于随机性一般围绕中间值波动，求和后正负波动互相抵消，所以大大降低了数据中的随机性因素，这时就可以得到数据中的长期趋势和循环变动两个部分，也可以称为移动平均数序列MA

$$MA = T \times C$$

(2) 通过平均法求出季节指数S，对观察值剔除移动平均数，则只留下季节性和随机性。

$$\frac{Y_t}{MA} = \frac{T \times C \times S \times I}{T \times C} = S \times I$$

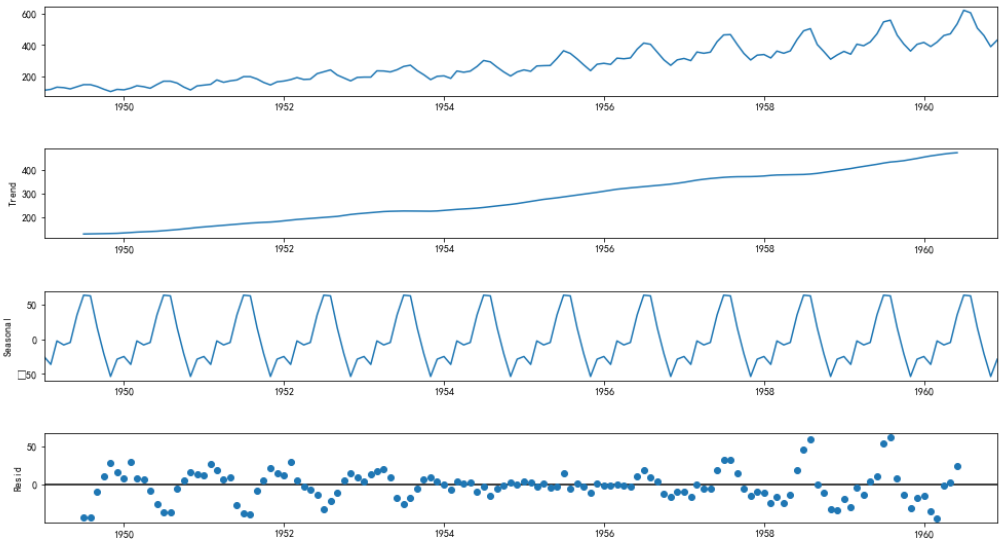
使用平均去除随机性的影响，即可获得数据的季节性。

$$\overline{S \times I} = S$$

(3) 最后使用趋势外推法分析长期趋势并使用移动平均数计算循环变动。

这里本文只需要使用分解后的图像来对数据进行趋势观察，使用Python相关函数对图像进行绘制，图2中的第一张图为原序列的趋势图，第二张为呈现向上增上的长期趋势图，第三章的季节趋势图反映出本数据的季节性非常明显，且具有年周期性。

图2：数据分解趋势图



4.3 数据平稳性判断

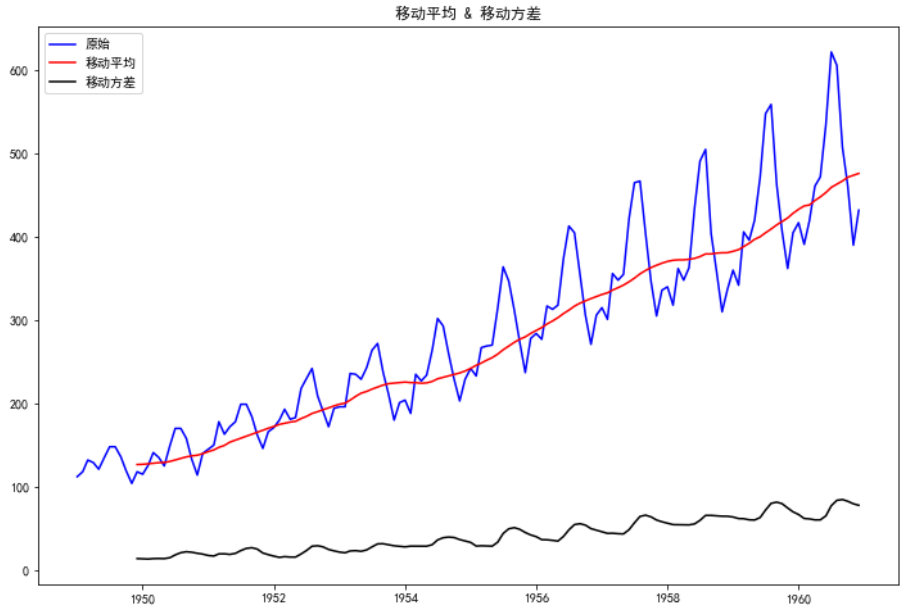
时间序列的平稳性是指其统计性质关于时间平移的不变性，因为对时间序列的预测需要从历史数据得到对未来规律的判断，所以需要保证不变性，从而才能对平稳的时间序列进行回归。

本文主要使用扩展的迪克富勒检验，假设此时间序列是非平稳的，得出测试统计量和不同置信水平的临界值，得到结果如下表所示，同时，由于观察到数据的年周期性，按照年为窗口，在任何时刻，采用每年去年的平均值/方差，绘制移动平均和移动方差，通过观察是否随时间变化也可以从主观上判断平稳性。

表2：迪克富勒检验结果

Test Statistic	0.815369
$\boldsymbol{p-value}$	0.991880
***Critical Value* (1%)**	-3.481682
***Critical Value* (5%)**	-2.884042
***Critical Value* (10%)**	-2.578770

图3：移动平均-移动方差图



从表3可以得到，$p-value$大于0.05，且$Test\ Statistic$大于三个置信水平的值。同时移动平均值也呈现向上的趋势，移动方差存在较大波动。

结合主观以及客观的判断，可以得出该时间序列不平稳的结论，因此需要对序列平稳化。

4.4 数据平稳化

首先对原始数列进行对数变换，将指数增长转为线性增长，且平稳序列的方差后，使用一阶差分对数据进行处理。

一阶差分为利用某个时期和上一时期的差值进行平稳化。

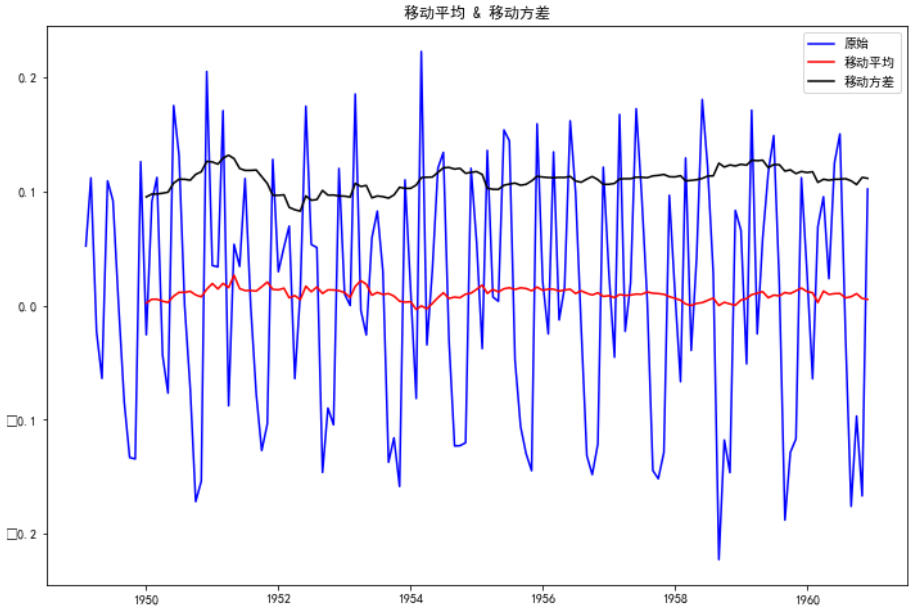
$$\Delta X_t = X_t - X_{t-1} = (1-B)X_t$$

从数据和图像中可以看出，改善了序列平稳，但是还没有达到标准，所以使用季节性差分继续进行处理，消除序列的季节性。

表3：一阶差分后迪克富勒检验结果

Test Statistic	-2.717131
$\boldsymbol{p-value}$	0.071121
***Critical Value* (1%)**	-3.482501
***Critical Value* (5%)**	-2.884398
***Critical Value* (10%)**	-2.578960

图4：一阶差分后的移动平均-移动方差图



一般地，对于一个周期为s的季节性时间序列$\{X_t\}$, 其季节差分为

$$\Delta_s X_t = X_t - X_{t-s} = (1 - B^s)X_t$$

其中$\Delta_s = (1 - B^s)$为季节差分。

由于观察到数据的季节性为以年为周期，对1一阶差分后的序列ΔX_t做季节差分，选取$s = 12$作为周期。

$$\Delta_{12}(\Delta X_t) = (1 - B^{12})\Delta X_t = \Delta X_t - \Delta X_{t-12} = X_t - X_{t-1} - X_{t-12} + X_{t-13}$$

差分后的结果如下，序列平稳。

表4：一阶差分和季节差分后迪克富勒检验结果

Test Statistic	−4. 443325
$\boldsymbol{p - value}$	0. 000249
***Critical Value* (1%)**	−3. 487022
***Critical Value* (5%)**	−2. 886363
***Critical Value* (10%)**	−2. 580009

4. 4 白噪声检验

白噪声检验为判断序列是否呈现白噪声序列的特征，可以用来判断序列是否有进一步分析的必要，同时可以判断模型是否充分。

建立假设检验问题，原假设为序列之间不存在相关关系，由此构造检验统计量并进行判断，若拒绝原假设证明此序列之间存在相关关系，可以继续进行下一

步建立模型的工作。

在这里本文使用Box-Pierce统计量，结果如下表。

表5：平稳后序列白噪声检验结果

滞后阶数	基于χ^2的$p-value$	基于χ^2下 Box-Pierce 检验的$p-value$
6 阶	7.10630704e-04	9.29608852e-04
12 阶	7.68546569e-07	3.12707671e-06

由于$p-value$皆非常小，所以拒绝原假设，认为该数据存在相关关系。

4.5 模型识别与定阶

模型识别是根据序列自相关函数和偏自相关函数的截尾限制，对$\{X_t\}$属于哪一类模型进行初步判断。

自相关函数（ACF）度量同一个序列在不同时间点的相关程度，也就是该序列过去的状态对现在的影响，表达式如下

$$\hat{\rho}_k=\frac{\sum_{t=1}^{T-k}(X_t-\bar{X})(X_{t+k}-\bar{X})}{\sum_{t=1}^{T}(X_t-\bar{X})^2}$$

其中$X_1,X_2,\cdots,X_T$来自平稳序列$\{X_t\}$，$\bar{X}$为样本均值。

偏自相关函数为剔除了自相关函数中的$k-1$个变量的影响后序列的相关程度，表达式如下

$$\widehat{\phi_{ss}}=\frac{\widehat{D_s}}{\widehat{D}}$$

其中

$$\widehat{D}=\begin{vmatrix}1 & \hat{\rho}_1 & \cdots & \hat{\rho}_{s-1}\\ \hat{\rho}_1 & 1 & \cdots & \hat{\rho}_{s-2}\\ \vdots & \vdots & \ddots & \vdots\\ \hat{\rho}_{s-1} & \hat{\rho}_{s-1} & \cdots & 1\end{vmatrix},\widehat{D}_s=\begin{vmatrix}1 & \hat{\rho}_1 & \cdots & \hat{\rho}_{s-1}\\ \hat{\rho}_1 & 1 & \cdots & \hat{\rho}_{s-2}\\ \vdots & \vdots & \ddots & \vdots\\ \hat{\rho}_{s-1} & \hat{\rho}_{s-1} & \cdots & \hat{\rho}_s\end{vmatrix}$$

将ACF和PACF计算出后根据如下标准进行模型选择

表6：平稳时间序列自相关和偏自相关函数特征

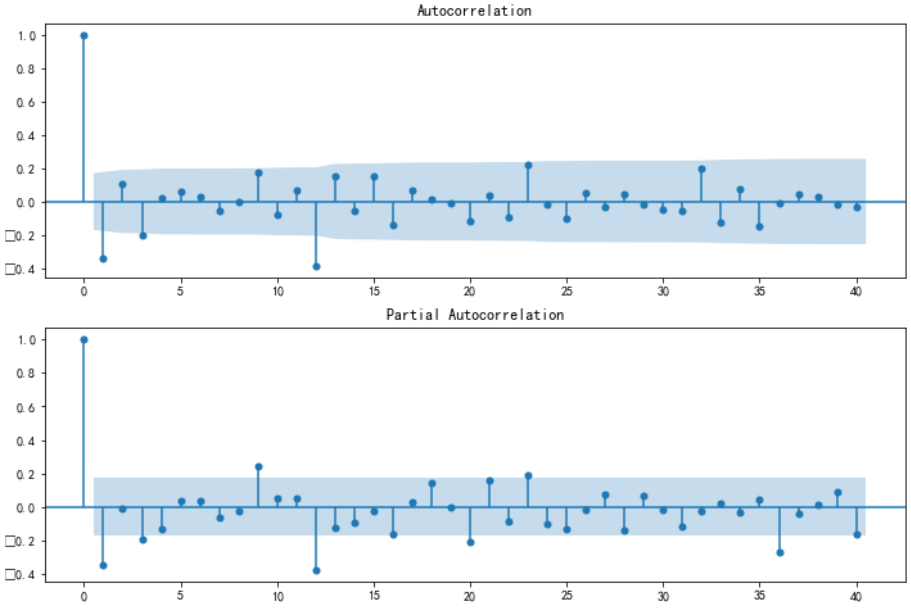
模型	$AR(p)$	$MA(p)$	$ARMA(p,q)$
ACF	拖尾	截尾	拖尾
PACF	截尾	拖尾	拖尾

其中拖尾是指序列以指数率单调递减或震荡衰减，而截尾指序列从某个时点变得非常小。同时可以根据截尾和拖尾判断模型的阶数：

- （偏）自相关系数d阶截尾：最初d阶明显大于2倍标准差范围，之后几乎95%的（偏）自相关系数都落在2倍标准差范围内，且由非零自相关系数衰减为在零附近小值波动的过程非常突然。
- （偏）自相关系数拖尾：超过5%的样本（偏）自相关系数都落入2倍标准差范围之外，或是由显著非0的（偏）自相关系数衰减为小值波动的过程比较缓慢或非常连续。

对一阶差分及季节差分后的序列绘制ACF和PACF图。

图5：序列的自相关与偏自相关函数图



从图上可以观察出皆存在波动性，并且由于原始数据经历过差分，所以初步判断对此序列使用ARIMA模型。同时可以观察到函数在每12个滞后期都存在规律性的波动，所以本文使用季节性ARIMA以提高对预测的准确性。

$\{X_t\}$是周期为s的季节性ARIMA过程的形式是$ARIMA(p,d,q)\times(P,D,Q)_s$，其表达式如下

$$\phi(B)\Phi(B^s)Y_t=\theta(B)\Theta(B^s)a_t,\{a_t\}\sim WN(0,\sigma^2)$$

其中 $\phi(z)=1-\phi_1(z)z-\phi_2(z)z^2-\cdots-\phi_p(z)z^p,\Phi(z)=1-\Phi_1(z)z-\Phi_2(z)z^2-\cdots-\Phi_P(z)z^P,\theta(z)=1+\theta_1(z)z+\theta_2(z)z^2+\cdots+\theta_q(z)z^q,\Theta(z)=1+\Theta_1(z)z+\Theta_2(z)z^2+\cdots+\theta_Q(z)z^Q$，且差分序列$Y_t=(1-B)^d(1-B^s)^DX_t$。

得出建立何种模型后，需要对模型阶数进行判断，但是从图4并不能很好地

定阶，并且主观定阶可能会影响预测的结果，所以选择建立不同阶数的模型，通过模型的AIC值结果来选择合适的模型。

AIC（赤池信息准则）的表达式如下：

$$AIC = \frac{-2}{T} lnL + \frac{2}{T} n$$

其中L是似然函数，n是样本的个数，T是样本容量，对模型阶数进行选择时，需要选择能使AIC达到最小的阶数，从而能够在综合考量模型的拟合优度和参数的个数的情况下确定阶数。

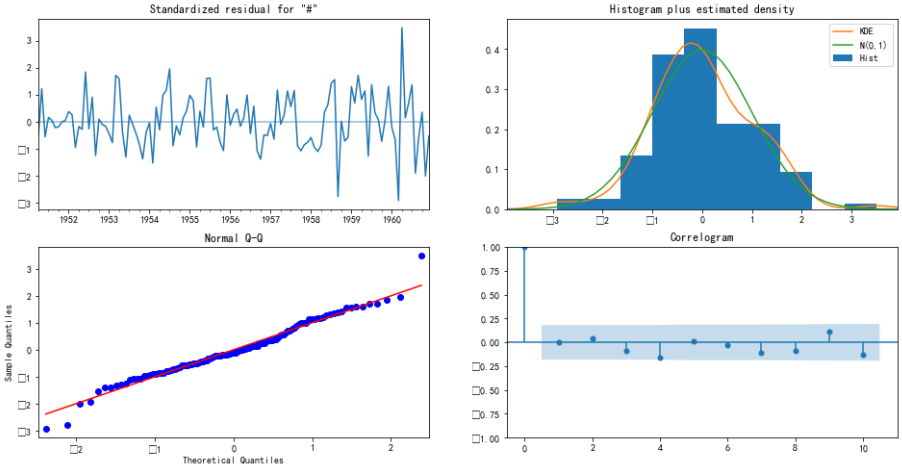
生成所有参数p, d, q及P, D, Q范围为0~2，$s = 12$的模型组合的情况，并进行AIC的计算，计算部分结果如下。

表7：部分测试模型AIC值

模型	***AIC***
$SARIMA(0,0,0) \times (0,0,0)_{12}$	2044.43294864
$SARIMA(0,0,0) \times (0,0,1)_{12}$	1887.35562466
$SARIMA(0,0,0) \times (0,1,0)_{12}$	1315.92119295
$SARIMA(0,0,0) \times (0,1,1)_{12}$	1156.29149015
$SARIMA(0,0,0) \times (1,0,0)_{12}$	1118.62591907

最后对AIC最小值进行筛选，模型定为$SARIMA(0,1,1) \times (1,1,1)_{12}$，$AIC$值为920.31929750，此时绘制模型诊断图。

图6：模型诊断图



在右上图中，可以观察到红色的KDE线与正态分布线接近，表示残留物正常分布，且左下角的QQ图显示蓝色残差有序分布遵循采用标准分布采样的线性趋势，同时随着时间的推移，左上图的残差不会显示任何明显的季节性，右下角的自相关图也证实了，时间序列残差与其本身的滞后版本具有低相关性，模型有效，可以进行下一步预测。

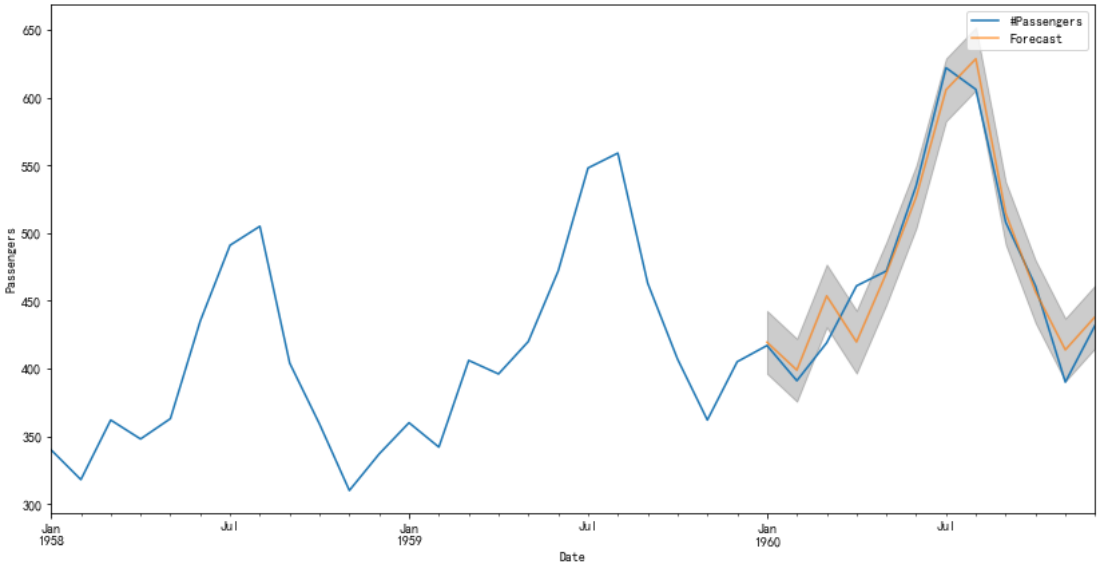
4.6 序列预测

本文采用使预测误差的均方值最小来作为评价模型的标准，计算公式如下

$$E[\hat{e}_t^2(l)] = E\{[X_{t+l} - \hat{X}_t(l)]^2\}$$

使用1960年开始的数据作为测试集，对模型进行参数估计后，预测1960年以后的数据，预测结果如下图，可以观察到橙色预测线位于95%的置信区间阴影内，预测值与真实值较相似

图7：有真实值的预测图

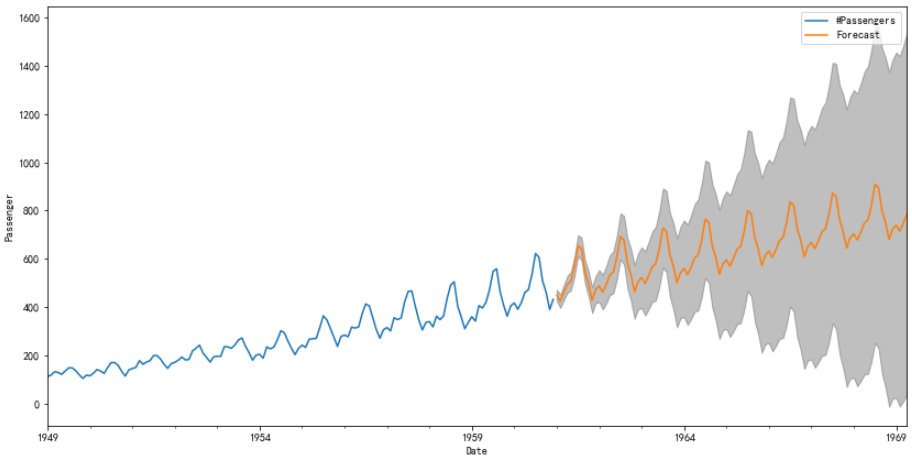


同时对预测结果进行计算，可以得到均方误差为374.73，对均方误差开方得到均方根误差19.36，这表示着本文建立的模型能够在实际乘坐人数的19.36千人次以内预测测试集中的乘客人数，乘客人数从100至650千人不等，所以模型的预测结果还是较好的。

同时对未来乘客人数进行预测，可以观察到随着时间的推移，置信区间会变大，这也是合理的，因为间隔时间过长会使预测变得不确定，这时需要更多近期

的值对模型进行训练与优化。

图8：对未来数据进行预测



五、 结论

本文对航班序列进行季节性趋势的研究并进行预测，从本文得出的结果可以看出，在进行平稳化研究后得出的季节性模型可以较好地对此序列进行预测，预测结果在乘客数量整体范围内约4%的范围进行波动，该方法预测比较准确，在航班运作者对未来航班的数量、时间等方面进行配置时，可以起到较好的借鉴效果。

六、 参考文献

[1] 邵鹏郡.(2020).基于 ARIMA 时间序列模型的美国失业率预测研究. 国际公关 (12),395-396. doi:10.16645/j.cnki.cn11-5281/c.2020.12.192.

[2] Susan, L. (2018). An End-to-End Project on Time Series Analysis and Forecasting with Python. [2021-06-18], from https://towardsdatascience.com/an-end-to-end-project-on-time-series-analysis-and-forecasting-with-python-4835e6bf050b?gi=ff4959b1984a

[3] JAIN, A. (2016). Time Series Forecasting In Python | R. [2021-06-18], from https://www.analyticsvidhya.com/blog/2016/02/time-series-forecasting-codes-python/

[4] SRIVASTAVA, T. (2015). Time Series Analysis | Time Series Modeling In R. [2021-06-18], from https://www.analyticsvidhya.com/blog/2015/12/complete-tutorial-time-series-modeling/

[5] 利用 python 进行时间序列分析——季节性 ARIMA. (2018). [2021-06-18], from https://zhuanlan.zhihu.com/p/35282988

七、 附录

```
#导入包
import warnings
import itertools
import numpy as np
import matplotlib.pyplot as plt
warnings.filterwarnings("ignore")
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['figure.figsize'] = 15, 6
import pandas as pd
import statsmodels.api as sm
import matplotlib
%matplotlib inline
import datetime

#导入数据
data = pd.read_csv('AirPassengers.csv')
dates=list(data['Month'])
dates = [datetime.datetime.strptime(date, '%Y-%m-%d') for date in dates]
data['Month']=dates
data = data.set_index('Month')
data.index

#绘制
plt.plot(data)
plt.title('1949-1960年航班人数')
plt.savefig('1949-1960年航班人数')
decomposition = sm.tsa.seasonal_decompose(data,model='additive')
fig = plt.figure()
fig = decomposition.plot()
fig.set_size_inches(15, 8)
plt.savefig('分解趋势图')

#平稳化
```

```
from statsmodels.tsa.stattools import adfuller   #Dickey-Fuller test
def test_stationarity(timeseries):

    #Determing rolling statistics
    rolmean = timeseries.rolling(12).mean()
    rolstd = timeseries.rolling(12).std()

    #Plot rolling statistics:
    fig = plt.figure(figsize=(12, 8))
    orig = plt.plot(timeseries, color='blue',label='原始')
    mean = plt.plot(rolmean, color='red', label='移动平均')
    std = plt.plot(rolstd, color='black', label = '移动方差')
    plt.legend(loc='best')
    plt.title('移动平均 & 移动方差')
    plt.savefig('移动平均 & 移动方差')
    plt.show()


    #Perform Dickey-Fuller test:
    print('迪克富勒检验结果:')
    dftest = adfuller(timeseries, autolag='AIC')   #autolag : {‘AIC’, ‘BIC’, ‘t-stat’, None}
    dfoutput = pd.Series(dftest[0:4], index=['Test Statistic','p-value','#Lags Used','Number of Observations Used'])
    for key,value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key] = value
    print(dfoutput)
test_stationarity(data)

#对数一阶差分
data_log=np.log(data)
log_diff1 = data_log.diff(1)
test_stationarity(log_diff1.dropna(inplace=False))

#季节差分
```

```
    sea_diff = log_diff1 - log_diff1.shift(12)
    test_stationarity(sea_diff.dropna(inplace=False))

    #白噪声检验
    from statsmodels.stats.diagnostic import acorr_ljungbox
    acorr_ljungbox(sea_diff.dropna(), lags = [6,12],boxpierce=True)

    #ACF PACF确定模型
    fig = plt.figure(figsize=(12,8))
    ax1 = fig.add_subplot(211)
    fig = sm.graphics.tsa.plot_acf(sea_diff.iloc[13:], lags=40, ax=ax1) #从13开始是因为做季节性差分时window是12
    ax2 = fig.add_subplot(212)
    fig = sm.graphics.tsa.plot_pacf(sea_diff.iloc[13:], lags=40, ax=ax2)
    plt.savefig('ACF & PACF')

    #确定阶数
    p = d = q = range(0, 2)
    pdq = list(itertools.product(p, d, q))
    seasonal_pdq = [(x[0], x[1], x[2], 12) for x in list(itertools.product(p, d, q))]
    warnings.filterwarnings("ignore")
    for param in pdq:
        for param_seasonal in seasonal_pdq:
            try:
                mod = sm.tsa.statespace.SARIMAX(data,
                                                order=param,
                                                seasonal_order=param_seasonal,
                                                enforce_stationarity=False,
                                                enforce_invertibility=False)
                results = mod.fit()
                print('ARIMA{}x{}12  -  AIC:{}'.format(param,  param_seasonal, results.aic))
            except:
                continue
```

```
#模型建立与诊断
mod = sm.tsa.statespace.SARIMAX(data, trend='n', order=(0,1,1), seasonal_order=(1,1,1,12),enforce_stationarity=False,enforce_invertibility=False)
results = mod.fit()
print(results.summary())
results.plot_diagnostics(figsize=(16, 8))
plt.savefig('模型诊断图')
plt.show()

#预测
pred = results.get_prediction(start=pd.to_datetime('1960-01-01'), dynamic=False)
pred_ci = pred.conf_int()
ax = data['1958':].plot(label='observed')
pred.predicted_mean.plot(ax=ax, label='Forecast', alpha=.7, figsize=(14, 7))
ax.fill_between(pred_ci.index,
                pred_ci.iloc[:, 0],
                pred_ci.iloc[:, 1], color='k', alpha=.2)
ax.set_xlabel('Date')
ax.set_ylabel('Passengers')
plt.savefig('有训练集的预测')
plt.legend()
y_forecasted = pred.predicted_mean
y_truth = data['1960-01-01':]['#Passengers']
mse = ((y_forecasted - y_truth) ** 2).mean()
print('The Mean Squared Error of our forecasts is {}'.format(round(mse, 2)))
print('The Root Mean Squared Error of our forecasts is {}'.format(round(np.sqrt(mse), 2)))

pred_uc = results.get_forecast(steps=100)
pred_ci = pred_uc.conf_int()
ax = data.plot(label='observed', figsize=(14, 7))
pred_uc.predicted_mean.plot(ax=ax, label='Forecast')
```

```
ax.fill_between(pred_ci.index,
                pred_ci.iloc[:, 0],
                pred_ci.iloc[:, 1], color='k', alpha=.25)
ax.set_xlabel('Date')
ax.set_ylabel('Passenger')
plt.legend()
plt.savefig('对未来数据进行预测')
plt.show()
```