基于 Python 的时间序列分析实验报告

莫力炬-210810419

摘要

时间序列分析是一种用于研究时间相关数据的统计方法。通过时间序列分析，我们可以预测未来的趋势和模式，从而做出更准确的决策。在本次实验中，通过对时间序列传统理论的学习，如 ARMA、ARIMA、ARCH 等经典模型的公式推导和理解，建立对时间序列数据的基本建模和处理方法。并且基于 Python 应用这些模型，完成了对合成数据以及 sarima 数据进行性质分析和模型选择、训练与评估，在部分模型中得到了较好的预测效果。

关键词：时间序列分析；ARIMA 模型；ARCH 模型；趋势分析

目录

一、 时间序列理论

1.1 ARMA 模型

ARMA（自回归滑动平均）模型是一种常用的时间序列分析模型，其由自回归（AR）和滑动平均（MA）两部分组成。AR 部分表示时间序列的当前观测值与过去观测值的线性组合，而 MA 部分则表示当前观测值与过去观测误差的线性组合。这两个部分的组合允许模型对时间序列数据的自相关性和随机性进行建模。ARMA 模型的数学表达式为：

$$X_t = c + \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p X_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + ... + \theta_q \varepsilon_{t-q} \quad (1)$$

其中 X_t 为时间序列的观测值，c 为常数，$\phi_1, \phi_2, \ldots, \phi_p$ 为 AR 模型的系数，$\theta_1, \theta_2, ..., \theta_p$ 为 MA 模型的系数，ε_t 为白噪声误差项。AR 部分的阶数 p 表示模型使用的过去观测值的数量，即当前观测值与之前 p 个观测值相关。MA 部分的阶数 q 表示模型使用的过去误差项的数量，即当前观测误差与之前 q 个误差相关。

ARMA 模型能够捕捉时间序列数据中的趋势性、周期性和季节性变化，提供对未来观测值的预测和分析。但是当时间序列数据存在非线性关系或者季节性特征时，ARMA 模型可能无法很好地拟合数据 [1]。

1.2 ARIMA 模型

ARIMA（自回归滑动平均差分整合）模型是在 ARMA 模型的基础上引入差分操作的一种时间序列分析模型。ARIMA 模型常用于对非平稳时间序列数据进行建模和预测。

ARIMA 模型的核心思想是通过差分操作将非平稳时间序列转化为平稳时间序列，然后应用 ARMA 模型对平稳时间序列进行建模。差分操作能够消除时间序列数据中的趋势性和季节性变化，使得模型更加准确和稳健。ARIMA 模型的数学表达式为：

$$(1 - \phi_1 B - \phi_2 B^2 - \ldots - \phi_p B^p)(1 - B)^d X_t = (1 + \theta_1 B + \theta_2 B^2 + \ldots + \theta_q B^q)\varepsilon_t \quad (2)$$

其中，X_t 是原始时间序列数据，$\phi_1, \phi_2, ..., \phi_p$ 是自回归（AR）模型的系数，B 是滞后操作符，d 是差分阶数，$\theta_1, \theta_2, ..., \theta_q$ 是滑动平均（MA）模型的系数，ε_t 是白噪声误差项。

与 ARMA 模型相比，ARIMA 模型在处理非平稳时间序列数据时具有明显优势，通过差分操作能够减少非平稳性带来的挑战。然而，对于长期依赖或者非线性关系较强的时间序列数据，ARIMA 模型可能仍然不够适用。

1.3 季节性 ARIMA 模型

季节性 ARIMA（Seasonal ARIMA）模型是 ARIMA 模型的扩展，专门用于处理具有季节性变化的时间序列数据。季节性 ARIMA 模型能够捕捉时间序列数据中的季节性趋势，并对其进行建模和预测。季节性 ARIMA 模型的数学表达式为：

$$\begin{aligned}(1-\phi_1 B-\phi_2 B^2-\ldots-\phi_p B^p)(1-\Phi_1 B^s-\Phi_2 B^{2s}-\ldots-\Phi_P B^{Ps})(1-B)^d(1-B^s)^D X_t \\ =(1+\theta_1 B+\theta_2 B^2+\ldots+\theta_q B^q)(1+\Theta_1 B^s+\Theta_2 B^{2s}+\ldots+\Theta_Q B^{Qs})\varepsilon_t\end{aligned} \tag{3}$$

其中，X_t 是原始时间序列数据，$\phi_1, \phi_2, ..., \phi_p$ 和 $\Phi_1, \Phi_2, ..., \Phi_P$ 分别是非季节性自回归（AR）和季节性 AR 模型的系数，B 是滞后操作符，s 是季节周期，d 是非季节性差分阶数，D 是季节性差分阶数，$\theta_1, \theta_2, ..., \theta_q$ 和 $\Theta_1, \Theta_2, ..., \Theta_Q$ 分别是非季节性滑动平均（MA）和季节性滑动平均模型的系数，ε_t 是白噪声误差项。

季节性 ARIMA 模型在处理具有明显季节性变化的时间序列数据时非常有效。它能够捕捉时间序列数据中存在的季节性趋势，比基础的 ARIMA 模型进一步提高预测的准确性和可靠性。

1.4 ARCH 模型

ARCH（Autoregressive Conditional Heteroskedasticity）模型是一种用于处理时间序列数据中异方差性（heteroskedasticity）的经典模型。它在条件异方差性的背景下对时间序列数据的波动性进行建模和预测。

ARCH 模型的基本思想是，当前时刻的波动性是过去一段时间内的波动性的自回归函数，即波动性具有自相关性。该模型通过引入一个或多个滞后平方波动项（即自回归模型）来描述波动性的变化。ARCH 模型的数学表达式 [2] 为：

$$\sigma_t^2=\alpha_0+\alpha_1\varepsilon_{t-1}^2+\alpha_2\varepsilon_{t-2}^2+\ldots+\alpha_p\varepsilon_{t-p}2+u_t \tag{4}$$

其中，σ_t^2 为当前时刻的方差，ε_t 为当前时刻的误差项，$\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_p$ 为模型的参数。参数 α_0 表示方差的长期平均水平，参数 $\alpha_1, \alpha_2, \ldots, \alpha_p$ 表示方差受到过去 p 个误差项平方的影响程度。

ARCH 模型的优势在于它能够捕捉时间序列数据中存在的方差的异动性和非恒定性。通过引入自回归结构，模型能够较好地描述和预测数据的方差的变化。ARCH 模型可以用于金融市场波动性的建模，例如股票价格波动、利率变动等。

1.5 GARCH 模型

GARCH（Generalized Autoregressive Conditional Heteroskedasticity）模型是在 ARCH 模型的基础上发展而来的一种更为灵活和强大的异方差建模工具。GARCH 模型目的是捕捉时间序列数据中存在的异方差特征，并通过引入多个滞后期的方差项，更准确地描述和预测方差的变化 [2]。GARCH 模型可以表示为如下形式：

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{p} \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2 \tag{5}$$

其中，σ_t^2 为当前时刻的方差，ε_t 为当前时刻的误差项，$\alpha_0, \alpha_1, \ldots, \alpha_p$ 为 ARCH 项的参数，$\beta_1, \beta_2, \ldots, \beta_q$ 为 GARCH 项的参数。模型中的 ARCH 项 $\alpha_i \varepsilon_{t-i}^2$ 表示方差受到过去 p 个误差项平方的影响，GARCH 项 $\beta_j \sigma_{t-j}^2$ 表示方差受到过去 q 个方差项的影响。

与 ARCH 模型相比，GARCH 模型具有较好的建模灵活性和能力，能够更准确地拟合数据中的方差动态性。GARCH 模型能够处理多种方差特征，例如波动聚集（volatility clustering）、波动性偏离（volatility persistence）和杠杆效应（leverage effect）等。

二、 时间序列分析的方法

2.1 相干性、自相干与偏自相干

2.1.1 相干性

相干性是时间序列分析中的重要概念，用于衡量两个或多个时间序列之间的线性关系程度。相干性可以通过计算相关系数来度量，常见的方法有皮尔逊相关系数（Pearson correlation coefficient）和斯皮尔曼相关系数（Spearman correlation coefficient）。皮尔逊相关系数适用于线性关系的度量，而斯皮尔曼相关系数适用于非线性关系的度量。相干性的研究可以帮助我们确定变量之间的关联性，并为后续的预测和建模提供基础。

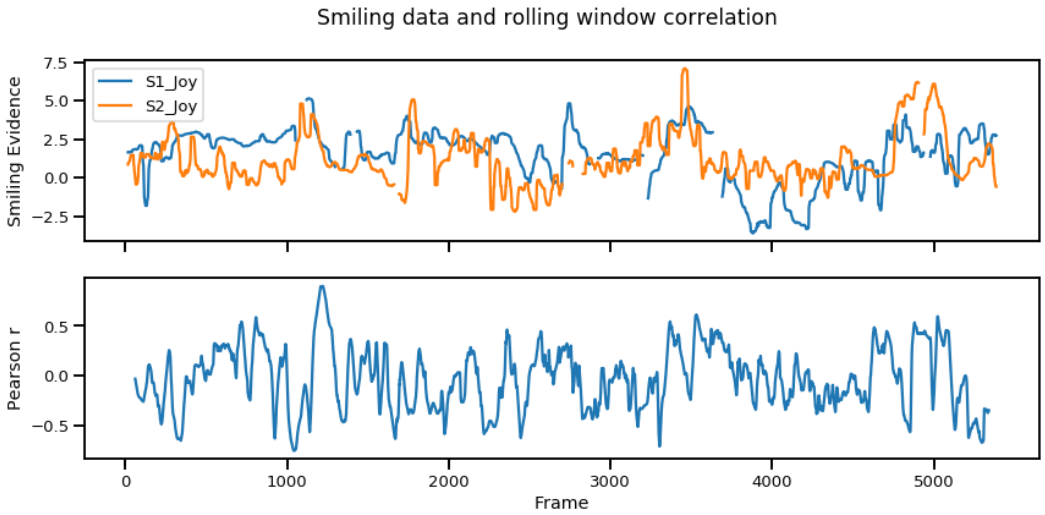

图 **1**　相干性分析示例

2.1.2 自相干

自相干是时间序列中自身的相关性。我们可以通过自相关函数（ACF）来衡量自相干性，它反映了时间序列与其自身在不同滞后阶数上的相关性。自相关函数可以帮助我们发现时间序列中的周期性和趋势性。通过观察自相关函数的图形，我们可以确定时间序列是否存在显著的滞后相关性，并根据相关性的模式来选择合适的时间序列模型。

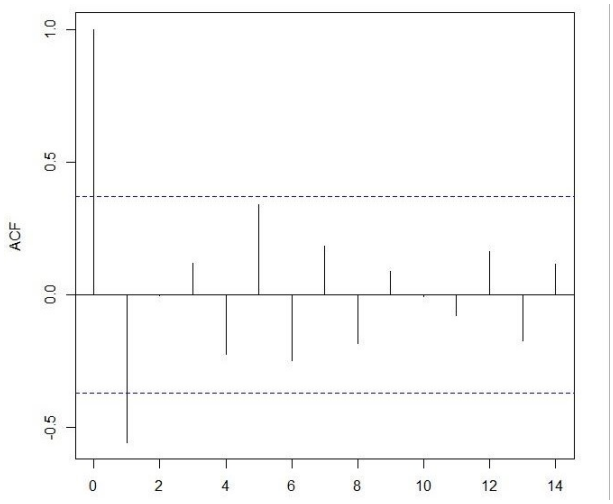

图 **2**　**ACF** 图线示例

2.1.3 偏自相干

偏自相干是指在考虑其他相关变量的情况下，两个时间序列之间的相关性。我们可以使用偏自相关函数（PACF）来衡量偏自相干性，它消除了其他滞后阶数的影响，使我们能够更准确地评估两个时间序列之间的独立关系。偏自相关函数的分析有助于确定

时间序列中的独立结构，并为模型的选择和参数估计提供指导。

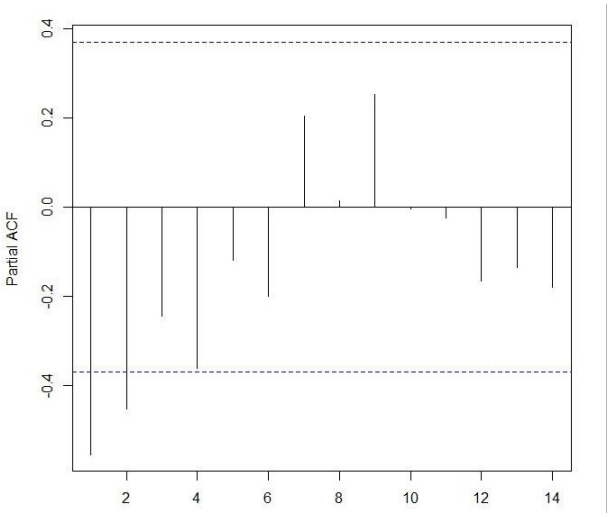

图 **3** **PACF** 图线示例

2.2 协方差矩阵

协方差矩阵是时间序列分析中常用的工具，用于衡量多个变量之间的关联程度和方向。协方差矩阵的计算公式如下：

$$C_{ij} = \frac{1}{m}\sum_{k=1}^{m}(X_i - \mu_i)(X_j - \mu_j) \tag{6}$$

其中，m 表示样本数量，X_i 和 X_j 分别表示第 i 个和第 j 个变量的样本值，μ_i 和 μ_i 分别表示第 i 个和第 j 个变量的样本均值。

协方差矩阵 C 是一个 $n \times n$ 对称矩阵，其中每个元素表示两个变量之间的协方差。如果 C_{ij} 为正值，则表示变量 X_i 和 X_j 呈正相关关系；如果 C_{ij} 为负值，则表示它们呈负相关关系；如果 C_{ij} 接近于 0，则表示它们之间没有线性关系。协方差的绝对值越大，表示两个变量之间的关联程度越强。协方差矩阵的对角线上的元素是各个变量的方差，即 C_{ii} 表示变量 X_i 的方差。方差衡量了一个变量自身的变化程度，方差越大，表示该变量的波动性越高。

2.3 残差与噪音

在时间序列分析中，残差和噪音是评估模型拟合程度和预测准确性的重要指标。

残差是指模型对观测值的预测与实际观测值之间的差异。它可以通过计算实际观测值与模型预测值之间的差异来得到。残差表示了模型无法解释的部分，即模型未能捕捉到的数据中的变异性。通过分析残差，我们可以评估模型的拟合程度。如果残差接近于零且呈随机分布，则表示模型能够很好地拟合数据。

噪音是指时间序列中无法解释的随机波动。噪音是由于系统中的随机性或未知因素引起的，它无法被模型所捕捉。噪音是时间序列中的随机成分，不能被预测或建模。在时间序列分析中，我们通常假设噪音项是独立同分布的，并且没有任何模式或结构。

评估残差和噪音的方法包括观察残差图、计算残差的统计指标和检验残差的随机性。残差图可以帮助我们检查残差是否呈现出任何模式或趋势，以及是否存在异方差性。常见的统计指标包括残差的均值、方差和自相关系数等。此外，还可以使用统计检验来检验残差序列的随机性，例如 Ljung-Box 检验或 Durbin-Watson 检验。

三、 时间序列的探索性分析和预处理

3.1 合成数据一

在生成合成数据一后，我们首先对原始数据进行可视化展示，以便更好地了解数据的特征和趋势如图4。不难看出，数据没有明显的周期性和趋势，已经基本平稳，下面进行更深入的检验。

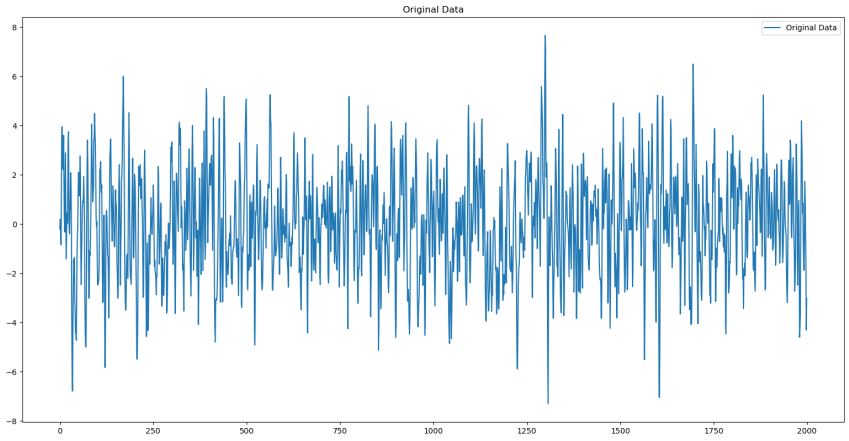

图 **4**　合成数据一

调用 statsmodels.tsa.stattools 中的 adfuller 方法进行 ADF（Augmented Dickey-Fuller）检验，结果显示 p 值远小于 0.05，可以认为时间序列数据已经平稳，不需要进行差分和其他预处理。

```
ADF Statistic: -15.541351123846113
p-value: 2.1573095295722298e-28
```

图 **5**　**adf** 检验结果

3.2 SARIMA 数据

同样先画出原始数据图线进行观察，可以发现呈明显的上升趋势和周期性性质，于是对其进行季节性分解（seasonal-trend decomposition），进一步确认其趋势（trend）、季节性（seasonality）和误差（residual）。

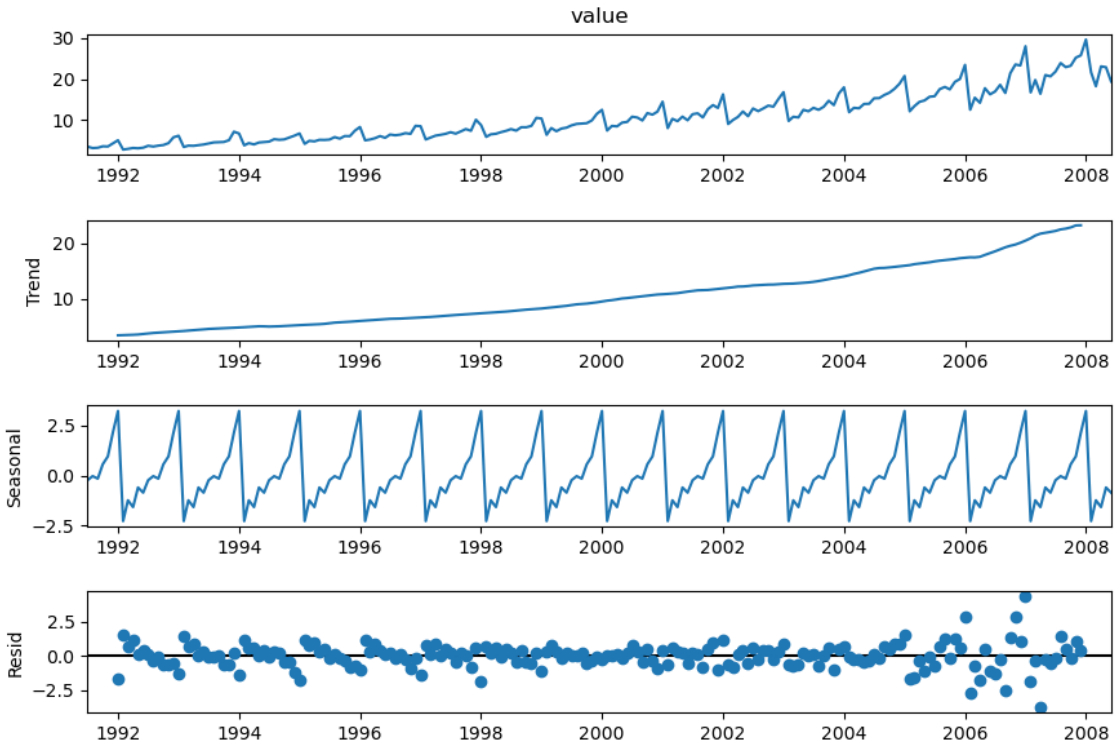

图 **6**　季节分解结果

从季节分解的结果可以看出，数据确实有明显的正相关趋势和周期性性质，而且周期大致为一年（即 12 个单位时间），下面尝试多种差分方法：分别有一次差分、两次差分、一次季节差分、一次季节差分和一次差分，结果展示如下图。

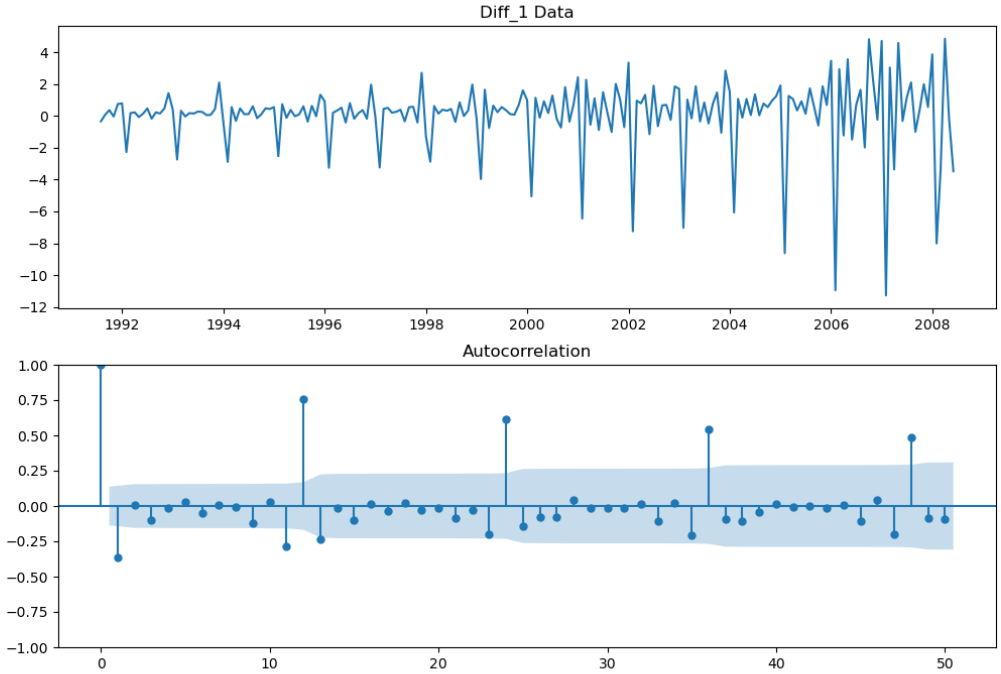

图 **7**　一次差分

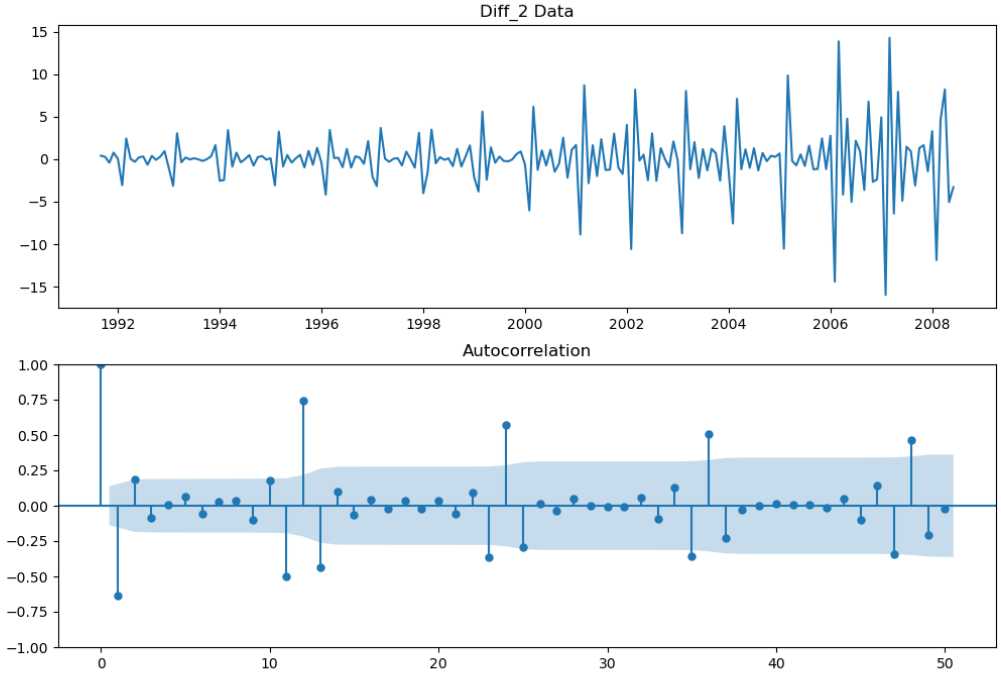

图 **8**　两次差分

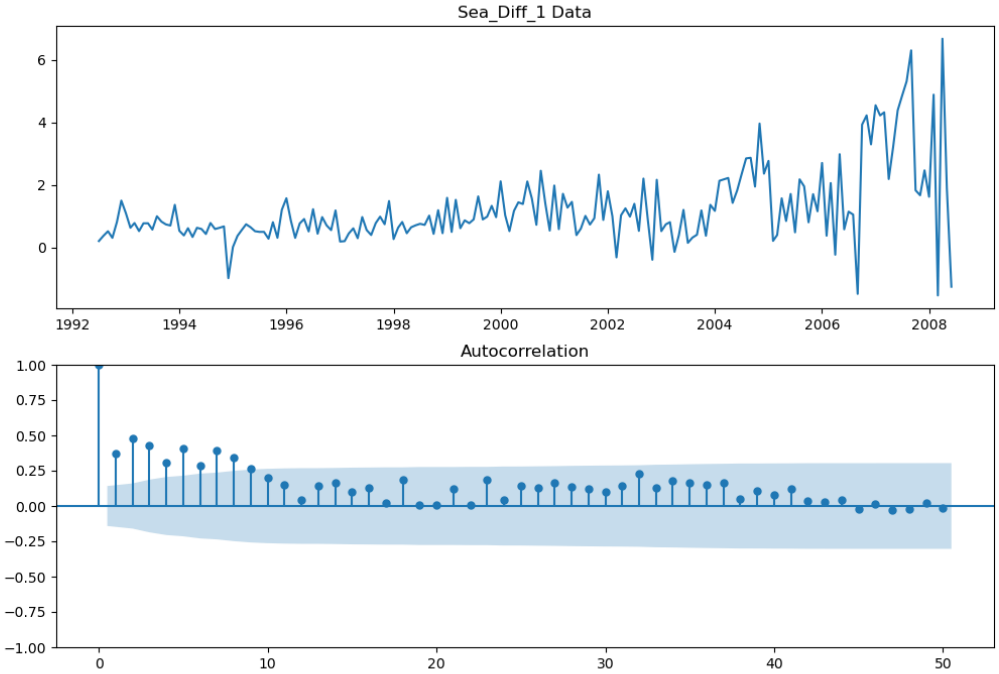

图 **9**　一次季节差分

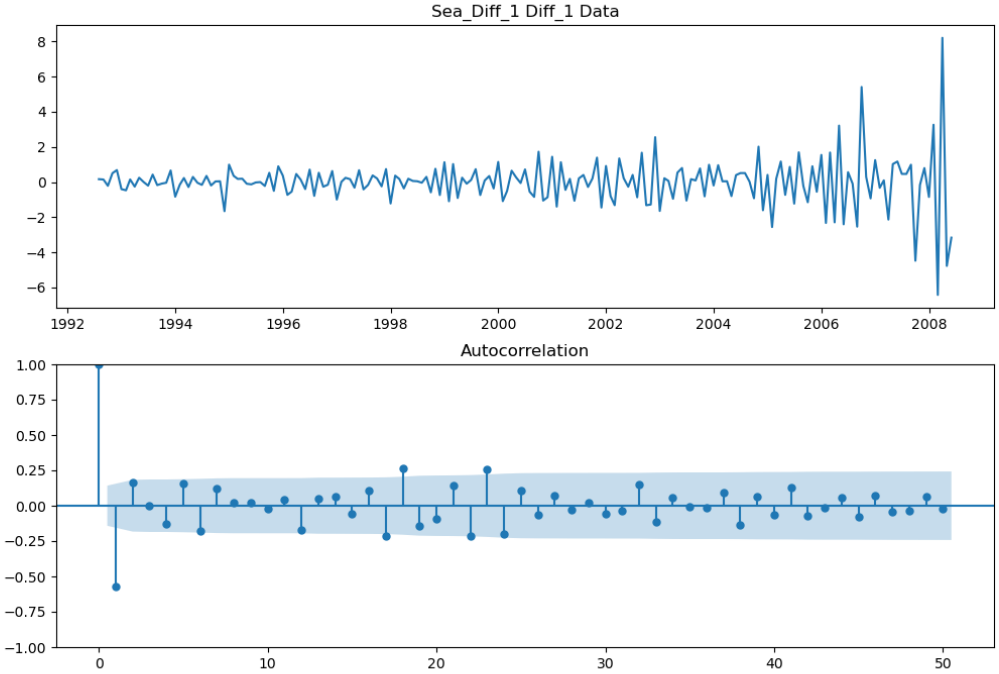

图 **10**　一次季节差分和一次差分

接着对几种组合同样进行 ADF 检验见表1，可以看出，进行两次差分后数据列变得平稳，能够通过 ADF 检验，其中相比进行两次普通差分，进行一次季节差分和一次普通差分得到的数据季节性大部分被抹除，数据更为平稳。因此我们选择对原数据进行一次季节差分和一次差分以得到平稳数据列。

表 **1**　**ADF** 检验统计表

	一次差分	两次差分	一次季节差分	一次季节差分和一次差分
t 统计量	-2.495	-10.292	-2.093	-4.783
p 值	0.117	3.543e-18	0.247	5.849e-5

3.3 合成数据二

先对合成数据二进行可视化，发现其平稳性相当明显，但是规律性不明显，结合生成函数可以得知，合成数据是一组符合高斯分布的随机噪声序列，均值和方差都十分稳定，不需要进行差分。

ARIMA 模型（自回归移动平均模型）通常用于分析具有明显的趋势和季节性的时间序列数据。ARIMA 模型的主要假设是时间序列数据具有一定的自相关性和平稳性。其中自相关性意味着当前观测值与过去观测值之间存在相关性，而平稳性意味着时间序列的统计特性在时间上是恒定的。由于合成数据二并没有明显的相关性和趋势，因此 ARIMA 模型并不适合对这组数据进行分析。

四、 时间序列建模与评估

4.1 合成数据一

采用 ARIMA 模型进行建模，由图3.1中可知参数 $d = 0$ 接着作出数据序列的 ACF 图和 PACF 图，并从中确定 ARIMA 模型的参数 p，q。

作出图线（图11、12）可以观察到，ACF 滞后 1、2 非常重要，因为它远高于显着性线。滞后 3 也稍微超过了显著性区间（蓝色区域），但是我们先将 q 定为 2。显然 PACF 图中几个滞后阶的数值都远高于显著性界限，观察得知 p=11。

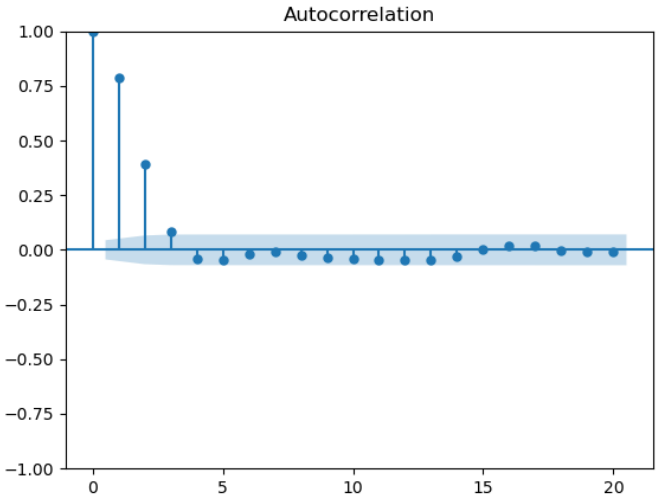

图 **11**　**acf** 图

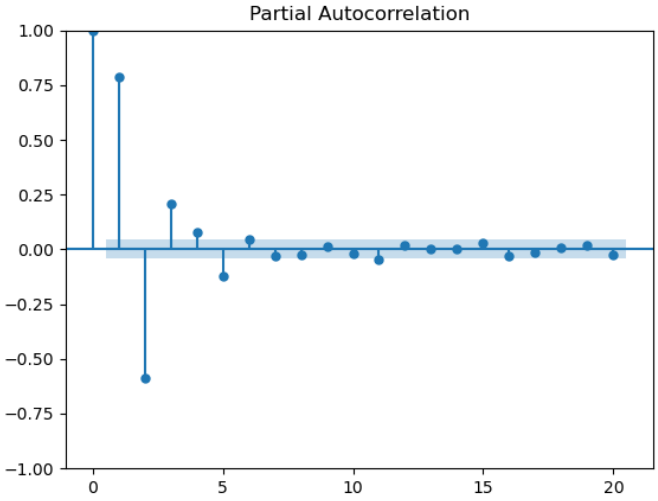

图 **12**　**pacf** 图

下面调用 Python 的 statsmodels 库中的 ARIMA(yy_arma, order=(11, 0, 2)) 方法对合成数据一进行拟合。观察图13、14可知，残差方差稳定，均值大致为 0，通过 Ljung-Box 检验可知，残差序列为白噪声，模型拟合效果不错。

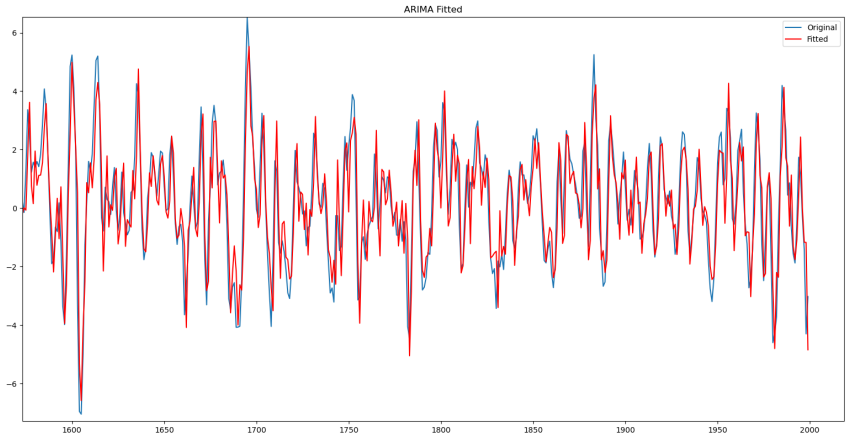

图 **13**　**ARIMA** 结果图

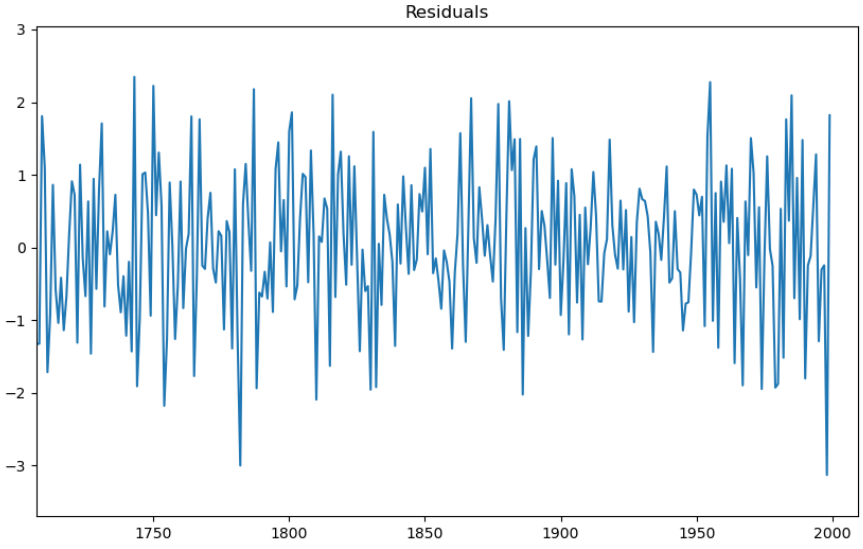

图 **14**　**ARIMA** 残差图

下面使用训练好的模型对未来 50 个时间节点的数据进行预测，预测结果如图15。预测结果并不好，这是因为 ARIMA 模型对于长期趋势的预测效果较差，对于非线性关系的建模能力有限等局限性，合成数据一数据集较大，而且具有非线性趋势，ARIMA 模型可能无法很好地捕捉到这些特征 [3]。

我们决定在更小的数据集上再做尝试，截取数据集的最后 100 条作为训练集进行模型训练，参数保持不变，收到了更好的预测效果，如图16。

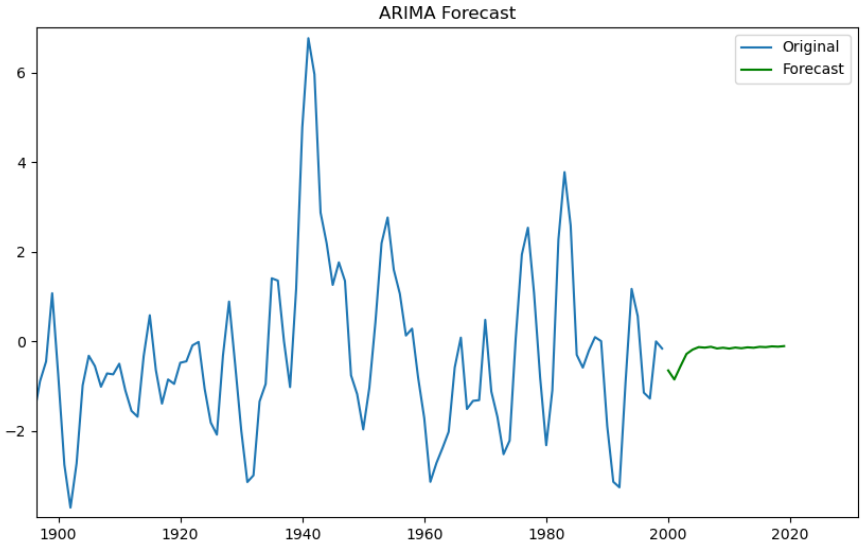

图 **15**　**ARIMA** 预测图 **1**

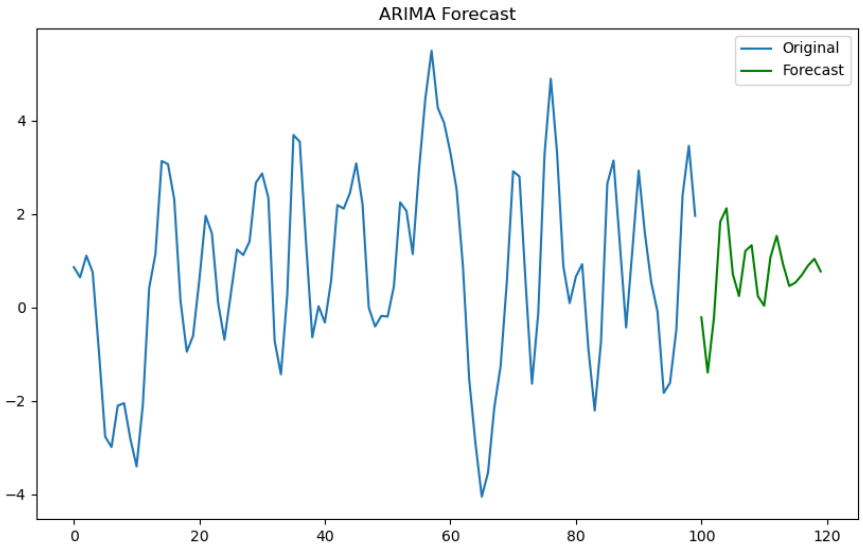

图 **16**　**ARIMA** 预测图 **2**

4.2 SARIMA 数据

我们先尝试使用了 ARIMA 模型对数据进行分析，同样画出 ACF 图和 PACF 图（图17、18）可以观察到，acf 图线在超过一阶后便不再显著，可确定 q=1；pacf 图线波动较大，高阶数仍有许多数据显著，因此需要更进一步分析。

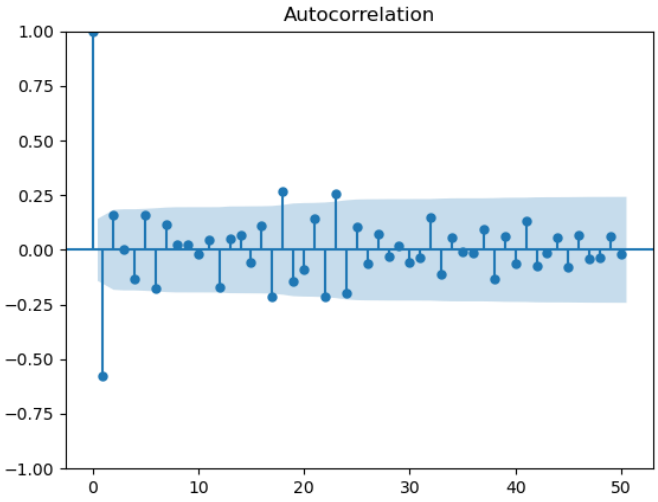

图 **17**　**acf** 图

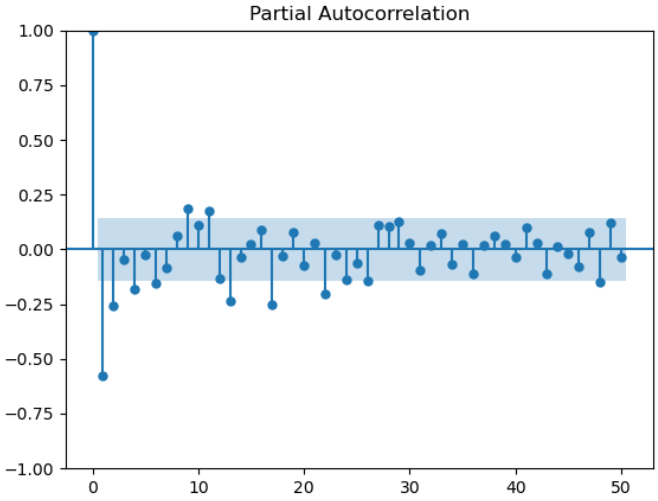

图 **18**　**pacf** 图

下面采用网格搜索方法尝试选择参数。我们设定参数 p，q 的取值范围是 $[0,5]$，决策标准设为 AIC（Akaike Information Criterion），作出热力图如下。从图中可以看出，如果只从 AIC 角度进行评判，参数组合 p=0，q=5 是最优参数，但是结合18可知，滞后的阶数非常高，远不止设定范围的上限 5，又已知数据的季节性比较强，因此转而选择 Seasonal ARIMA 模型是更好的选择。

图 **19**　热力图

下面确定 $SARIMA(p,d,q)\times(P,D,Q,m)$ 的参数，从3.2已经确定模型的参数 d=1，D=1，另外从6不难看出 m=12。接下来同样采用网格搜索方法确定剩余的参数，评判标准同上，由于参数较多，将每个参数的范围改为 $[0,3]$，最终得到最优参数为 $(3,1,2)\times(2,1,0,12)$。

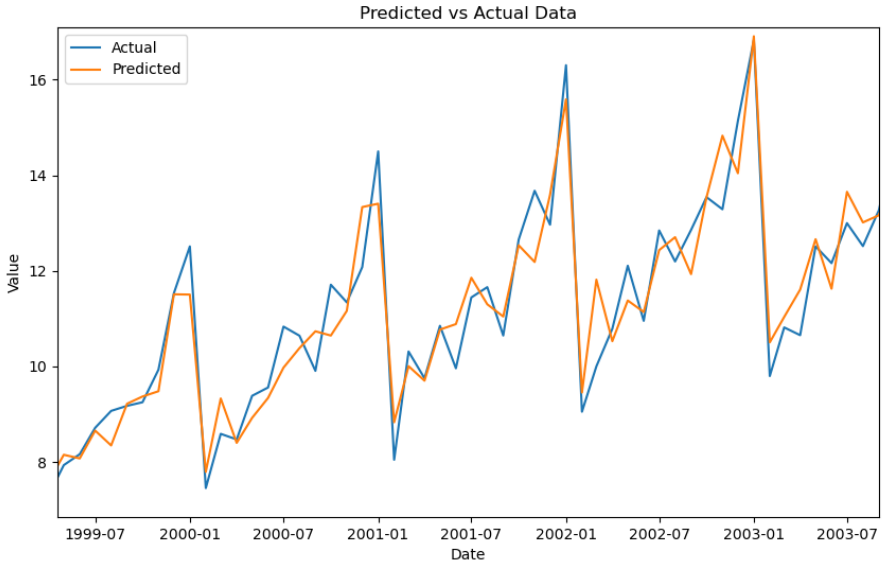

图 **20**　**SARIMA** 结果图

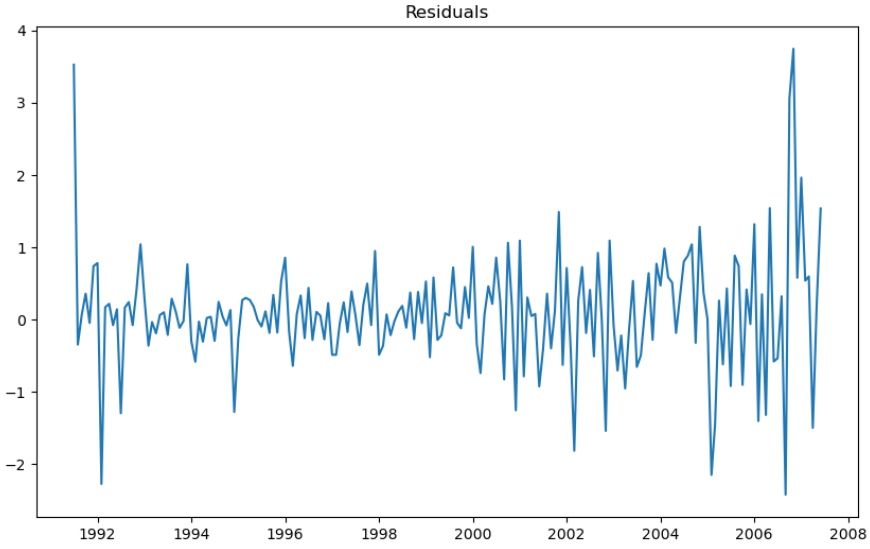

图 **21**　**SARIMA** 残差图

用得到的参数进行模型训练，事先将数据集划分成训练集和测试集，其中最后一个周期（即最后 12 个时间点）作为测试集，前面的数据作为训练集，拟合结果如下。残差的波动较大，但是通过 Ljung-Box 检验可知，滞后 10 阶以内均为白噪声，且均值大致仍为 0，模型基本可用。

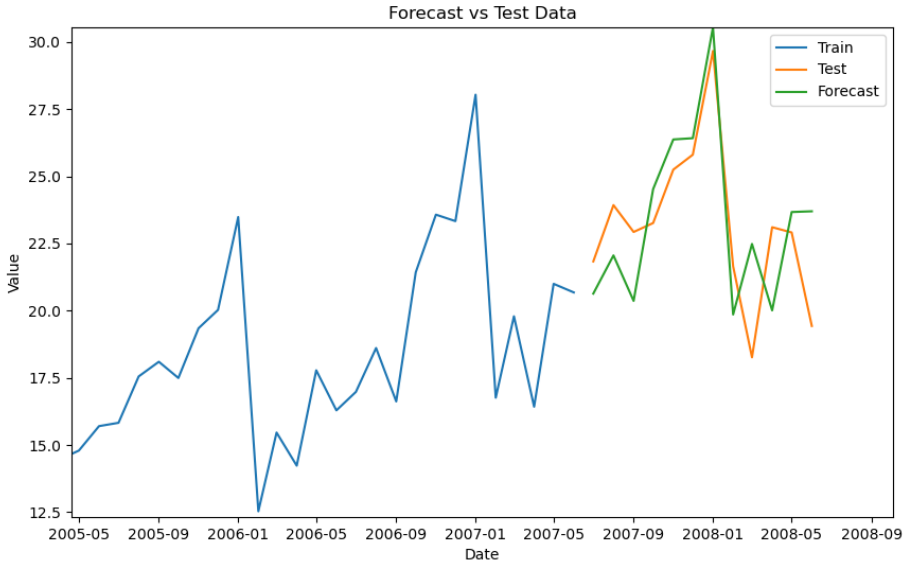

图 **22** **SARIMA** 预测图

然后将训练好的模型在测试集上进行预测，预测的结果对比如图，曲线大致走向符合真实数据，而且预测值接近实际情况，模型效果很合适。

4.3 合成数据二

同样画出 ACF 图和 PACF 图（23、24）可以观察到，二者都迅速下降到 0 附近，随后在 0 附近上下波动，因此可以初步使用 p=1，q=1 作为 arch 和 garch 模型的参数。

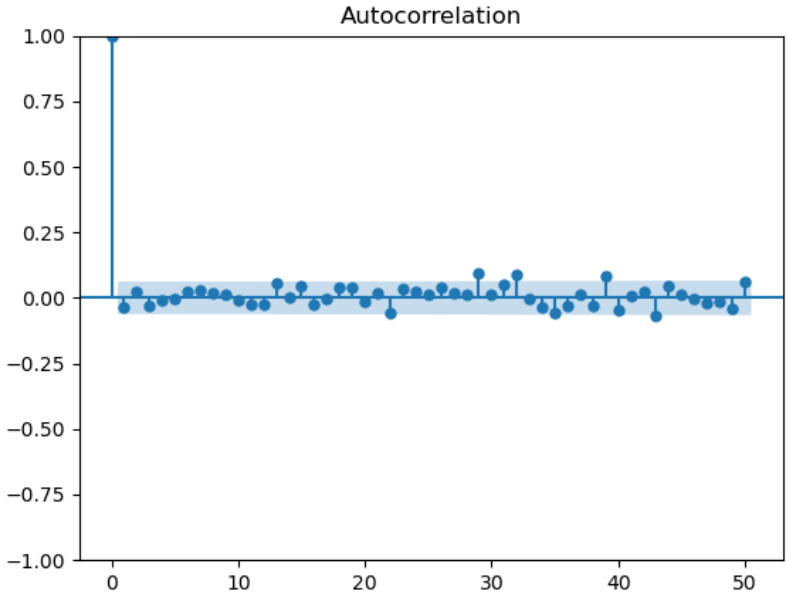

图 **23**　**acf** 图

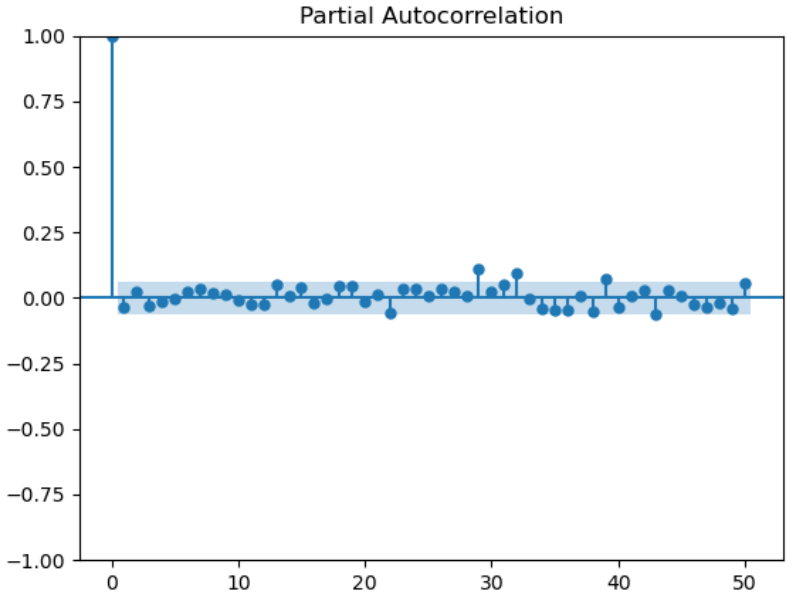

图 **24**　**pacf** 图

经过手动的参数微调发现，取参数 p=3 模型效果更好，因此下面使用 ARCH(3) 和 GARCH (3,1) 进行建模预测，结果如下图。以 ARCH 模型为例，其拟合参数如图25，模型参数均显著，AIC 为 1300.74。ARCH 和 GARCH 模型的拟合结果如图26，可以看出，虽然具体值差距挺大，但是均值和方差的变化相似，模型成功地预测拟合了数据的波动情况，但由于 R-square 较小，拟合精度较差。

Zero Mean - ARCH Model Results

Dep. Variable:	y	R-squared:	0.000
Mean Model:	Zero Mean	Adj. R-squared:	0.001
Vol Model:	ARCH	Log-Likelihood:	-646.370
Distribution:	Normal	AIC:	1300.74
Method:	Maximum Likelihood	BIC:	1320.37
		No. Observations:	1000
Date:	Tue, Oct 17 2023	Df Residuals:	1000
Time:	11:07:54	Df Model:	0

Volatility Model

	coef	std err	t	P>|t|	95.0% Conf. Int.
omega	0.0394	1.437e-02	2.741	6.116e-03	[1.123e-02,6.756e-02]
alpha[1]	0.3972	7.925e-02	5.011	5.405e-07	[0.242, 0.553]
alpha[2]	0.2900	5.699e-02	5.089	3.605e-07	[0.178, 0.402]
alpha[3]	0.3128	7.056e-02	4.433	9.281e-06	[0.175, 0.451]

图 **25**　**ARCH** 模型参数

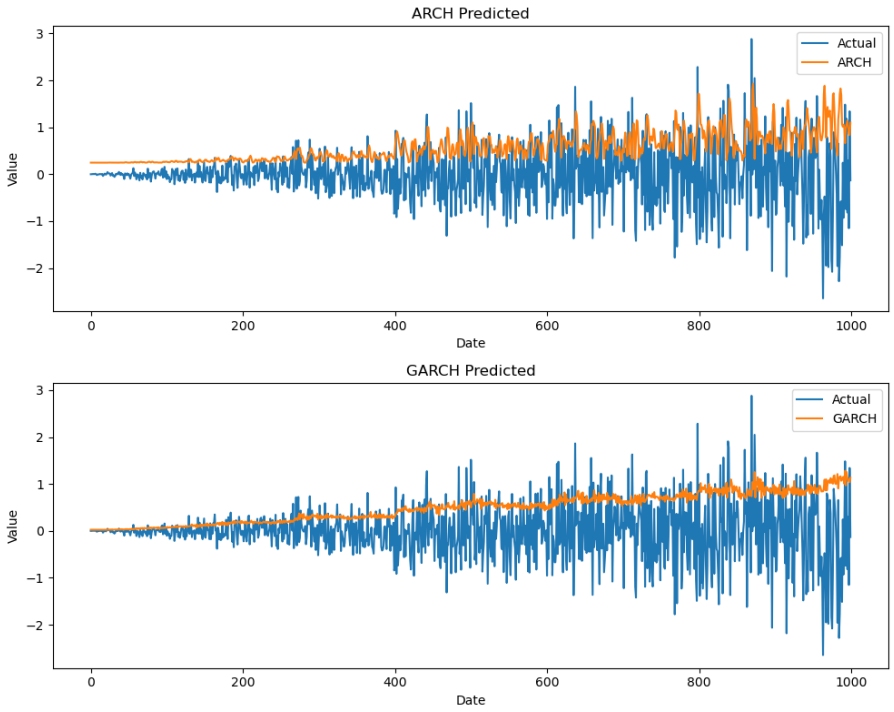

图 **26**　**ARCH**、**GARCH** 结果图

作出模型的残差图，同样采用 Ljung-Box 检验对其进行检验，二者的 p 值都较大，无法拒绝原假设，即残差序列并不是白噪声，模型的拟合效果一般，但由于 ARCH 和 GARCH 模型更多用于拟合、预测数据的波动率，具体数值的准确率并不那么重要。下面使用模型进行预测。

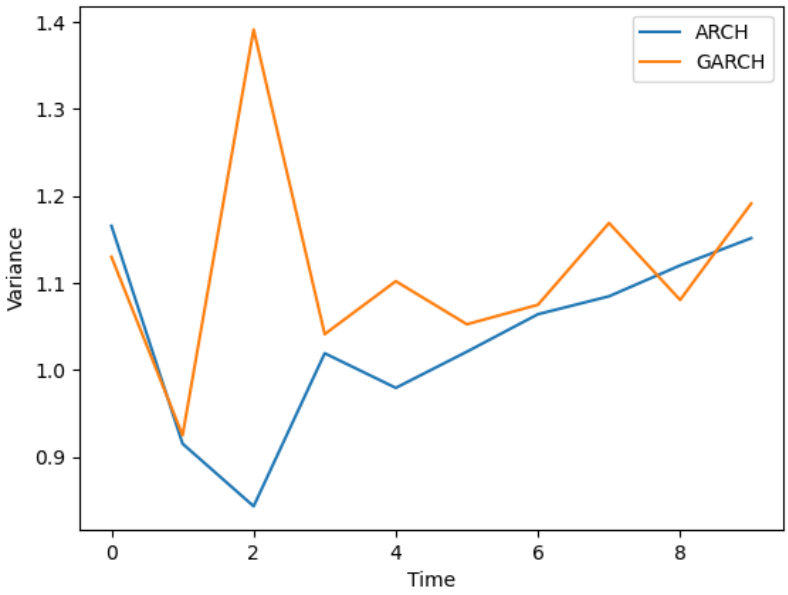

图 **27** **ARCH、GARCH** 预测图

预测结果显示，虽然两个模型的对于具体数值的预测以及波动的时间相距甚远，但是它们预测的波动方向和大致趋势是相似的，但从26可以得知，ARCH 对波动率的拟合效果更好，但是从理论上来说应该是 GARCH 效果更优，这可能是因为 GARCH 的模型复杂度更高，对参数更加敏感，在实验中我并没有找到很好的 GARCH 参数选择方法，只能使用网格搜索的方法确定了一组较优的参数 $(18, 18, 3)$，但是效果和起先的参数相差不多。总之 GARCH 模型效果欠佳就是本次实验最大的缺憾。

五、 结论

本实验旨在通过应用经典的时间序列分析模型来拟合数据序列，包括 ARIMA、SARIMA、ARCH 和 GARCH 模型。我们在实践中探索、理解了这些模型的特点和优劣势，并针对数据序列的特性进行了评估。通过对模型的比较，我们可以得出以下结论。

1. ARIMA 模型的主要优点在于能够很好地捕捉时间序列中的趋势和周期性，同时也能够考虑到序列的季节性变化。ARIMA 模型的主要局限性在于对于非线性和非平稳的时间序列拟合效果较差。
2. SARIMA 模型是 ARIMA 模型的一种扩展形式，它引入了季节性成分的考虑。因此相对于 ARIMA 模型在处理具有季节性变化的时间序列方面更加有效。由于引入了更多的参数，SARIMA 模型的计算复杂度较高，并且需要更多的样本数据来进行准确的参数估计 [4]。
3. ARCH 模型是一种用于建模时序数据波动方差的模型。其主要优点是能够很好地捕

捉到数据序列中的波动性和异方差性。然而，ARCH 模型需要明确地指定滞后期数，且对于样本较小或非线性序列的拟合效果并不理想。

4. GARCH 模型是 ARCH 模型的一种扩展形式，它在考虑波动方差的同时，还引入了滞后期的波动方差。GARCH 模型通过引入滞后期的波动方差成分，可以更好地捕捉数据序列中的长期依赖性。其计算复杂度同样较高，而且参数选择比较困难 [5]。

在这次实验中，我也遇到了一些困难。一是对时间序列数据不熟悉，不懂得常用的分析思路和检验方法，建立模型时遇到的问题很多，比如差分次数、差分方式的选择。最终是通过阅读大量资料、文献和书籍，大致了解了一些分析的方法，然后建模过程才变得相对顺利。二是对这些模型的形式、参数不熟悉，对参数调整毫无头绪，不清楚每个参数的意义和对应的作用。经过对一些案例的学习和迁移，我了解了各个模型的参数选择方法，虽然不一定是最优的，实验中得到的结果也不尽如人意，但这也是我两周来的努力和尝试的结晶了。

参考文献

[1] Time Series Analysis in Python. (n.d.). Retrieved from https://www.machinelearningplus.com/time-series/

[2] ARCH Documentation. (n.d.). Retrieved from https://arch.readthedocs.io/en/latest/index.html

[3] Kaggle 时间序列分析教程. (n.d.). Retrieved from https://www.kaggle.com/learn/time-series

[4] Shumway, R. H., & Stoffer, D. S. (2017). Time Series Analysis and Its Applications: With R Examples. Springer.

[5] Montgomery, D. C., Jennings, C. L., & Kulahci, M. (2015). Introduction to Time Series Analysis and Forecasting. John Wiley & Sons.

附录 A　网格搜索法寻找最优参数

```
def grid_param(train_data):
    # 定义要搜索的参数范围
    p = range(0, 4) # SARIMA模型的自回归阶数
    d = range(0, 2) # SARIMA模型的差分阶数
    q = range(0, 3) # SARIMA模型的移动平均阶数
    P = range(0, 3) # 季节性SARIMA模型的季节性自回归阶数
    D = [1] # 季节性SARIMA模型的季节性差分阶数
    Q = range(0, 3) # 季节性SARIMA模型的季节性移动平均阶数
    s = [12] # 季节性周期

    # 创建参数网格
    param_grid = {'order': list(itertools.product(p, d, q)), 'seasonal_order':
        list(itertools.product(P, D, Q, s))}

    # 创建空列表存储每个参数组合的AIC值
    aics = []

    # 遍历所有参数组合并拟合模型
    for params in ParameterGrid(param_grid):
    try:
        model = SARIMAX(train_data, order=params['order'],
            seasonal_order=params['seasonal_order'], trace=False)
        model_fit = model.fit()
        aics.append((params, model_fit.aic))
    except:
        continue


    # 找到AIC值最小的参数组合
    best_params, best_aic = min(aics, key=lambda x: x[1])
    print("Best Parameters:", best_params)
    print("Best AIC:", best_aic)

    return best_params
```

这个函数的主要功能是根据给定的参数范围，遍历所有的参数组合，寻找其中 AIC 指标最低的一组，并将之返回，用于寻找模型的最优参数组合。没有多余传入参数，使用时直接调用并传入原始数据即可。

附录 B　绘制 ARCH，GARCH 模型拟合结果

```
def plot_result(results_arch, results_garch):
```

```
    fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(10, 8))

    ax2.plot(data_arch, label='Actual')
    ax2.plot(results_garch.conditional_volatility, label='GARCH')
    ax2.set_title('GARCH Predicted')
    ax2.set_xlabel('Date')
    ax2.set_ylabel('Value')
    ax2.legend()

    ax1.plot(data_arch, label='Actual')
    ax1.plot(results_arch.conditional_volatility, label='ARCH')
    ax1.set_xlabel('Date')
    ax1.set_ylabel('Value')
    ax1.set_title('ARCH Predicted')
    ax1.legend()

    plt.tight_layout()
    plt.show()
```

这是一个用于可视化的函数，它针对拟合好的 ARCH，GARCH 模型，在训练集上进行预测，并同时画出原始数据和模型预测的数据列用于对比、凸显拟合效果。

使用示例如下：

```
# 拟合ARCH模型
model_arch = arch_model(data_arch, vol='ARCH', mean='Zero', p=3)
results_arch = model_arch.fit()

# 拟合GARCH模型
model_garch = arch_model(data_arch, vol='GARCH', mean='Zero', p=3, q=1)
results_garch = model_garch.fit()

# 绘制拟合结果
plot_result(results_arch, results_garch)
```

附录 C　时间序列的基本性质刻画

```
class Analysis:
    def __init__(self, data):
        self.ts_data = data

    def plot_ori(self):
        # 数据可视化 - 时间序列图
        plt.plot(self.ts_data['value'])
        plt.xlabel('Date')
```

```
        plt.ylabel('Value')
        plt.title('Time Series Plot')
        plt.show()

    def data_summary(self):
        # 描述性统计
        statistics = self.ts_data['value'].describe()
        print(statistics)

    def acf(self):
        # 自相关性分析
        plot_acf(self.ts_data['value'], lags=20)
        plt.title('Autocorrelation Plot')
        plt.show()

        plot_pacf(self.ts_data['value'], lags=20)
        plt.title('Partial Autocorrelation Plot')
        plt.show()

    def adf(self):
        # 平稳性检验
        diff_data = self.ts_data['value'].diff().dropna()
        result = adfuller(diff_data)
        print('ADF Statistic:', result[0])
        print('p-value:', result[1])

    def cycle(self):
        # 周期性检验 - 自相关函数（ACF）
        acf_vals = acf(self.ts_data['value'])
        plt.stem(acf_vals)
        plt.xlabel('Lag')
        plt.ylabel('ACF')
        plt.title('Autocorrelation Function')
        plt.show()

    def cor(self):
        # 相关性检验 - 计算相关系数矩阵
        corr_matrix = self.ts_data.corr()
        print(corr_matrix)
```

这是对时间序列进行初步分析的类，这个类名为 Analysis，它包含了一些用于时间序列分析的方法。下面我们逐个解释每个方法的功能：

- __init__(self, data)：这是类的构造函数，用于初始化类的实例。它接收一个参数 data，表示时间序列数据。

- plot_ori(self)：这个方法用于绘制原始数据的时间序列图。

- data_summary(self) : 这个方法用于计算数据的描述性统计，包括均值、标准差、最小值、最大值等。

- acf(self) : 这个方法用于绘制数据的自相关性图，包括自相关函数（ACF）和偏自相关函数（PACF）。

- adf(self) : 这个方法用于进行平稳性检验，具体使用了差分法（diff）和 ADF 检验（Augmented Dickey-Fuller test）。

- cycle(self) : 这个方法用于进行周期性检验，通过绘制自相关函数（ACF）来识别数据中的周期性。

- cor(self) : 这个方法用于计算数据的相关系数矩阵，以评估不同变量之间的相关性。

使用示例如下：

```
ana = Analysis(data)
sum = ana.data_summary()
```