HW4 数学建模报告

PB19151769　　马宇骁

摘要：利用国家统计局网站 http://www.stats.gov.cn/ 上的月度数据，建立数学模型，分析“粮食、经济作物、畜产品、水产品、蔬菜、水果”这六大类农产品集贸市场价格之间的联系，以及他们对于居民消费价格指数 CPI 的影响.
关键词：农产品，居民消费价格指数，CPI

1　背景

从 2020 年，全国工作重心转向两个主要的方面：保障医疗物资、助力医疗救助；保障人民生活物资、维护安定社会环境。生活物资的保障，是关系到每位老百姓一日三餐、衣食住行的国计民生的大事。[1]

生活物资需求主要来源于一日三餐的食物，主要依靠农产品提供保障。粮食，是老百姓生存最基本的能量和食物的来源，更是一个国家经济发展的基础。而经济作物作为为轻工业提供原料的作物，不但经济价值很高，也是人民重要的生活保障物资。

除去粮食作物和经济作物，畜产品和水产品也在老百姓餐桌上也扮演了非常重要的角色。畜产品主要包括肉类（如猪、牛、羊、鸡、鸭、鹅、兔等）、蛋类（如鸡蛋、鸭蛋、鹅蛋、鸽子蛋、鹌鹑蛋等）、奶类（如牛奶、羊奶、马奶等）、蜂产品（如蜂蜜、蜂花粉、蜂王浆等）及其他副产品。水产品主要包括淡水产品和海水产品两大类，按照生物学特征可分为鱼类（如草鱼、鲤鱼等）、甲壳动物类（如明虾、海蟹等）、软体动物类（如鲍鱼、扇贝等）、棘皮动物类（如海胆、海参等）、腔肠动物类（如海蜇等）及其他水产品。

粮食作物主要为生命提供葡萄糖以转换可利用的能量，畜产品和水产品为人们提供生命活动必不可少的蛋白质。蔬菜和水果作为人体食物主要的补充来源，主要补充人体所需要的维生素、纤维素和矿物质。蔬菜和水果所含的多种天然化学物质是富含生物活性的化合物，保护植物免受自然界里面细菌、病毒和真菌的侵害。这些天然化学物质的生活活性机理还不能被人们所完全认识。但其对人体预防和对抗各种病原体入侵、吸收毒素、器官衰老和组织癌变所起到的作用，已经为大家所认知和利用。

2　分析

从国家统计局的数据库中下载从 2013 年 1 月至今（2022 年）的六大类农产品集贸市场价格月度数据 [2]。由于每类数据中所包含的农产品种类很多，因此，只取前三个价格当期数据作图并展示如下：

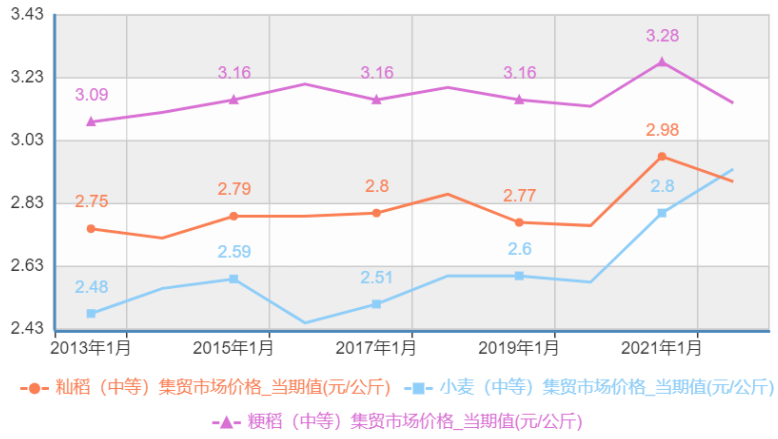

图 1: 粮食集贸市场价格

看出粮食的价格基本在稳步上升，但在最近两年有波动。

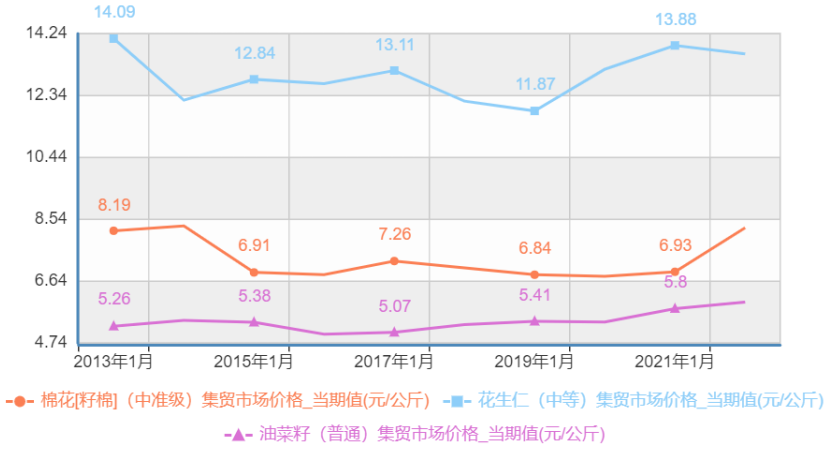

图 2: 经济作物集贸市场价格

经济作物的价格在近十年价格变动不明显。

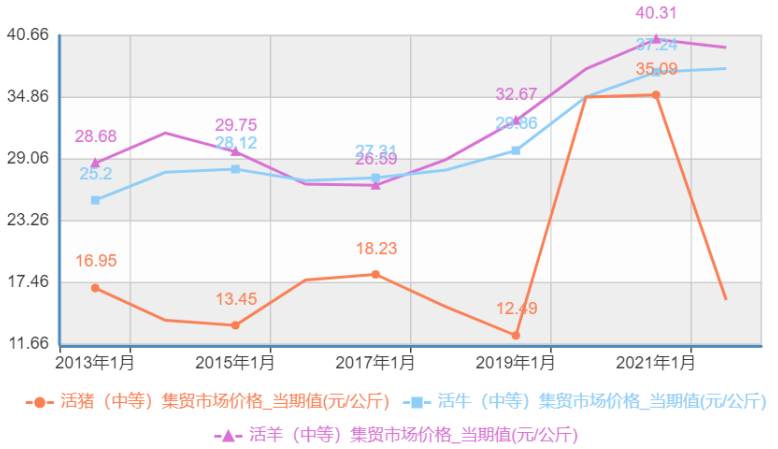

图 3: 畜产品集贸市场价格

畜产品价格具有周期性，详细分析见马宇骁的猪周期的分析 [3]，该篇文章详细分析了畜

产品价格的周期性的原因原理和预测方式。

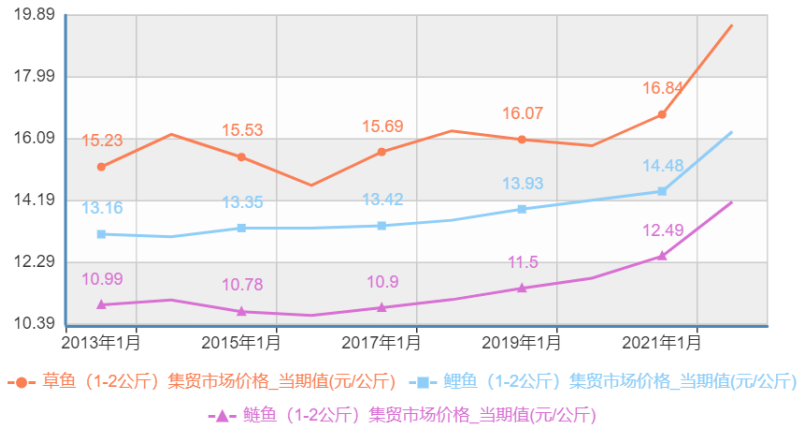

图 4: 水产品集贸市场价格

水产品的整体价格栽在提升，且趋势愈发明显。

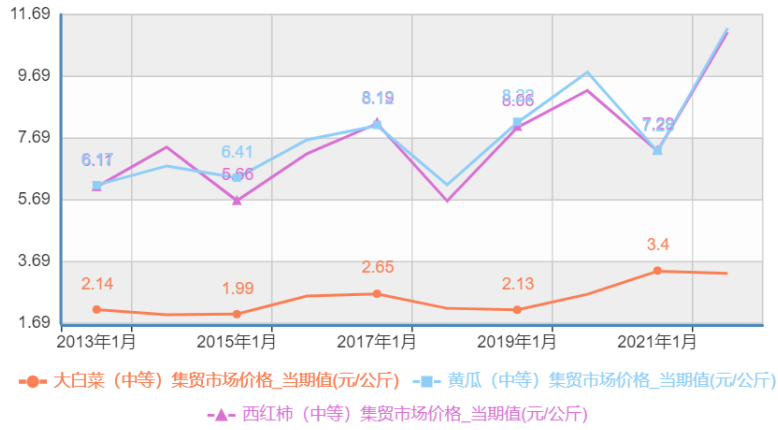

图 5: 蔬菜集贸市场价格

从图5可以看出蔬菜中，大白菜的价格稍有提升，黄瓜和西红柿的价格有明显周期性。

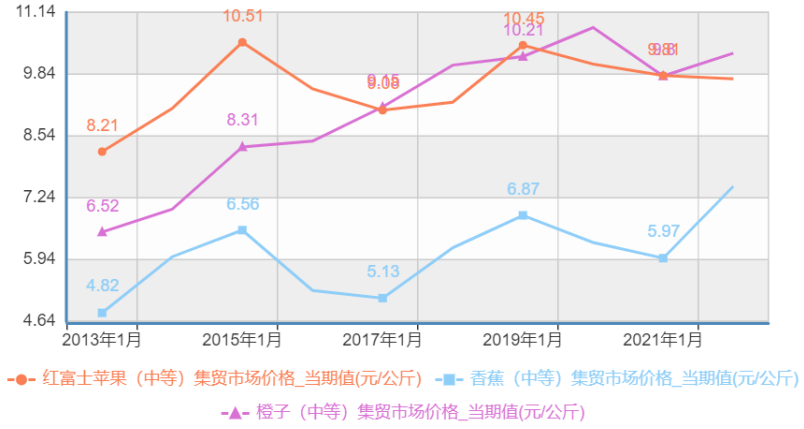

图 6: 水果集贸市场价格

图6显示出水果的价格在近年的变动，也有周期趋势。

CPI 数据是采用的以每一个上月为 100 计算的数据 (由于数据库 CPI 的数据只有从 2016 年 1 月份开始的，因此只展示 2016.1 之后的数据，第二个建模也是如此)，原始数据绘图如下：

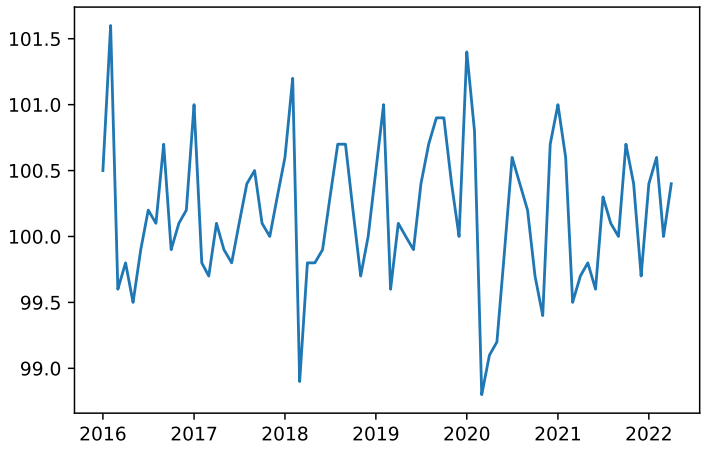

图 7: CPI

假设以 2015.12 为 CPI 为 100，做累计 CPI 转化，这样得到的调整 CPI 数据进行绘制如图8.

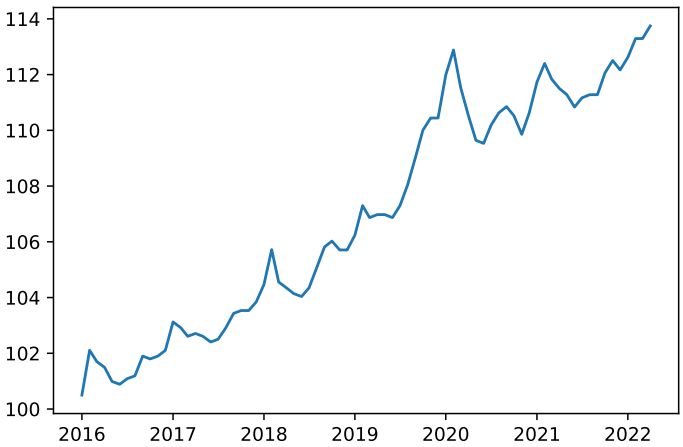

图 8: 调整 CPI

至此，数据初分析步骤结束，引入两个建模模型。

3 建模

由于要分析的 6 个类里面的项目过多，在建模时考虑简化将每个类中抽出一组出来作为特征数据进行建模: 小麦，花生仁，活猪，鲤鱼，黄瓜，香蕉。

3.1 农产品集贸市场价格之间的联系

使用向量自回归（vector autoregressive, VAR）模型。[4]

VAR 模型是用模型中所有当期变量对所有变量的若干滞后变量进行回归。即向量自回归模型把系统中每一个内生变量作为系统中所有内生变量的滞后值的函数来构造模型，从而实现了将单变量自回归模型推广到由多元时间序列变量组成的“向量”自回归模型。

VAR 模型常用于预测相互联系的时间序列系统以及分析随机扰动对变量系统的动态影响，主要应用于宏观经济学。是处理多个相关经济指标的分析与预测中最容易操作的模型之一。

由于向量自回归模型把每个内生变量作为系统中所有内生变量滞后值的函数来构造模型，从而避开了结构建模方法中需要对系统每个内生变量关于所有内生变量滞后值的建模问题。

3.1.1 VAR 模型

模型的基本形式是弱平稳过程的自回归表达式，描述的是在同一样本期间内的若干变量可以作为它们过去值的线性函数。

$$Y_t = \Phi_0 + \Phi_1 Y_{t-1} + \cdots + \Phi_p Y_{t-p} + BX_t + \varepsilon_t, \quad t = 1, 2, \cdots, T \tag{1}$$

其中，

$$Y_t = \begin{pmatrix} y_{1t} \\ y_{2t} \\ \vdots \\ y_{kt} \end{pmatrix}, \varepsilon_t = \begin{pmatrix} \varepsilon_{1t} \\ \varepsilon_{2t} \\ \vdots \\ \varepsilon_{kt} \end{pmatrix}, \quad \Phi_0 = \begin{pmatrix} \phi_{10} \\ \phi_{20} \\ \vdots \\ \phi_{k0} \end{pmatrix}$$

$$\Phi_i = \begin{pmatrix} \phi_{11}(i) & \phi_{12}(i) & \cdots & \phi_{1k}(i) \\ \phi_{21}(i) & \phi_{22}(i) & \cdots & \phi_{2k}(i) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{k1}(i) & \phi_{k2}(i) & \cdots & \phi_{kk}(i) \end{pmatrix}, \quad i = 1, 2, \cdots, p$$

- Y_t 表示 k 维内生变量列向量
- Y_{t-i}，i=1,2, ,p 为滞后的内生变量
- X_t 表示 d 维外生变量列向量，它可以是常数变量、线性趋势项或者其他非随机变量
- p 是滞后阶数
- T 为样本数目
- Φ_i 即 $\Phi_1, \Phi_2 \cdots, \Phi_p$ 为 k × k 维的待估矩阵
- B 为 k × d 维的待估矩阵
- $\varepsilon_t \sim N(0, \Sigma)$ 为 k 维白噪声向量，它们相互之间可以同期相关，但不与自己的滞后项相关（ε_t 独立同分布，而 ε_t 中的分量不要求相互独立），也不与上式中右边的变量相关。Σ 是 ε_t 的协方差矩阵，是一个 k × k 的正定矩阵。

VAR 模型的特点:

1. 不以严格的经济理论为依据。在建模过程中只需明确两件事： 共有哪些变量是相互有关系的，把有关系的变量包括在 VAR 模型中； 确定滞后期 p。使模型能反映出变量间相互影响的绝大部分。

2. VAR 模型对参数不施加零约束。（对无显着性的参数估计值并不从模型中剔除，不分析回归参数的经济意义。）

3. VAR 模型的解释变量中不包括任何当期变量，所有与联立方程模型有关的问题在 VAR 模型中都不存在（主要是参数估计量的非一致性问题）。

4. VAR 模型的另一个特点是有相当多的参数需要估计。比如一个 VAR 模型含有三个变量，最大滞后期 p=3，则有 27 个参数需要估计。当样本容量较小时，多数参数的估计量误差较大。

5. 无约束 VAR 模型的应用之一是预测。由于在 VAR 模型中每个方程的右侧都不含有当期变量，这种模型用于样本外一期预测的优点是不必对解释变量在预测期内的取值做任何预测。

6. 用 VAR 模型做样本外近期预测非常准确。做样本外长期预测时，则只能预测出变动的趋势，而对短期波动预测不理想。

3.1.2 数据建模

先计算数据之间的相关系数，结果如下：

```
[[ 1.          0.33175984  0.16392529  0.76916386  0.49366228  0.46088096]
 [ 0.33175984  1.          0.5702487   0.40129528  0.15913218 -0.21637976]
 [ 0.16392529  0.5702487   1.          0.11164018  0.20574528 -0.05915983]
 [ 0.76916386  0.40129528  0.11164018  1.          0.25729392  0.24322081]
 [ 0.49366228  0.15913218  0.20574528  0.25729392  1.          0.13887486]
 [ 0.46088096 -0.21637976 -0.05915983  0.24322081  0.13887486  1.        ]]
```

存在一定程度的线性相关。接下来对 6 个因子的原始数据进行平稳性检验，也就是 ADF 检验。VAR 模型要求所有因子数据同阶协整，也就是 6 个因子里面如果有一个因子数据不平稳，就要全体做差分，一直到平稳为止。

利用 statsmodels.api 中的 tsa.stattools.adfuller 进行检验，结果小麦的 p-Value 为 0.9988888206197162 远大于 0.05 即不平稳，因此先对全体做一阶差分。此时，6 个因子的 p-Value 为：

4.4230116313574625e-12,
2.501592689892194e-07,
1.7504784916922888e-11,
2.081139096614198e-07,
1.0106031873420553e-14,
1.3650732557033655e-11

这时全部通过 ADF 检验，即一阶差分数据平稳。

继续做协整检验 python 里面的协整检验通过 coint() 这个函数进行的，返回 p-Value 值，越小，说明协整关系越强。对 6 个因子差分后数据两两进行协整检验，返回 p-Value 如下：

3.794476144670872e-11,
5.6650734297401955e-11,
4.197438490154251e-11,
1.571125632531838e-10,
2.3952266770030205e-11,
1.4050118223059445e-06,
3.323609186664604e-06,
2.4405406556407465e-06,
1.6947364044637295e-06,
2.134621409174666e-11,
9.49864513220687e-11,
2.0062078357777857e-10,
1.626602232542773e-06,
2.4495993845569412e-06,
8.080175996078994e-13

发现这所有关系均存在长期均衡关系，VAR 模型应采用一阶差分后的数据构建。

使用 2013 年 1 月至 2021 年 12 月的数据进行 VAR 模型构建，将 2022 年数据留作预测检验。

先确定滞后项阶数：

通过 statsmodels.tsa.vector_ar.var_model.VAR 中的 select_order() 函数实现，综合考虑，采用 1 阶滞后系数作为模型参数。（输出结果如下表，* 代表最优参数）

VAR Order Selection (* highlights the minimums)

	AIC	BIC	FPE	HQIC
0	-14.16	-14.00	7.088e-07	-14.09
1	-15.19	-14.06*	2.544e-07	-14.73*
2	-15.12	-13.03	2.730e-07	-14.28
3	-14.96	-11.90	3.274e-07	-13.73
4	-15.04	-11.01	3.169e-07	-13.41
5	-14.91	-9.912	3.873e-07	-12.89
6	-15.14	-9.168	3.464e-07	-12.72
7	-15.17	-8.235	3.943e-07	-12.37
8	-15.74	-7.833	2.825e-07	-12.54
9	-16.26	-7.384	2.330e-07*	-12.67
10	-16.32	-6.482	3.431e-07	-12.35
11	-17.06	-6.255	3.081e-07	-12.70
12	-18.19*	-6.415	2.471e-07	-13.43

经过拟合 VAR（1），输出结果为：

```
  Summary of Regression Results
==================================
Model:                         VAR
Method:                        OLS
Date:           Tue, 17, May, 2022
Time:                     22:25:41
--------------------------------------------------------------------
No. of Equations:         6.00000    BIC:                   -14.4118
Nobs:                     106.000    HQIC:                  -15.0394
Log likelihood:          -40.6847    FPE:                1.91952e-07
AIC:                     -15.4672    Det(Omega_mle):     1.30785e-07
--------------------------------------------------------------------
Results for equation 1
==============================================================================
            coefficient       std. error           t-stat            prob
------------------------------------------------------------------------------
const         0.003006         0.001968            1.527           0.127
L1.1          0.163221         0.090445            1.805           0.071
L1.2         -0.029237         0.011709           -2.497           0.013
L1.3          0.001539         0.001056            1.457           0.145
L1.4          0.000521         0.006005            0.087           0.931
L1.5          0.007118         0.001951            3.648           0.000
L1.6         -0.011380         0.006025           -1.889           0.059
==============================================================================

Results for equation 2
==============================================================================
            coefficient       std. error           t-stat            prob
------------------------------------------------------------------------------
const         0.000407         0.013593            0.030           0.976
L1.1          0.204820         0.624667            0.328           0.743
L1.2          0.505980         0.080867            6.257           0.000
L1.3          0.016170         0.007295            2.216           0.027
L1.4         -0.036150         0.041477           -0.872           0.383
L1.5         -0.034109         0.013477           -2.531           0.011
L1.6         -0.093127         0.041610           -2.238           0.025
==============================================================================

Results for equation 3
==============================================================================
            coefficient       std. error           t-stat            prob
------------------------------------------------------------------------------
const         0.127205         0.175612            0.724           0.469
L1.1        -20.065400         8.070445           -2.486           0.013
L1.2          0.142134         1.044773            0.136           0.892
L1.3          0.270497         0.094251            2.870           0.004
L1.4         -1.066723         0.535861           -1.991           0.047
L1.5          0.136125         0.174114            0.782           0.434
L1.6         -0.715172         0.537579           -1.330           0.183
==============================================================================

Results for equation 4
==============================================================================
            coefficient       std. error           t-stat            prob
------------------------------------------------------------------------------
const         0.011866         0.028300            0.419           0.675
L1.1          0.987921         1.300570            0.760           0.447
L1.2          0.126892         0.168367            0.754           0.451
L1.3          0.003195         0.015189            0.210           0.833
L1.4          0.525357         0.086355            6.084           0.000
L1.5         -0.074237         0.028059           -2.646           0.008
L1.6         -0.091176         0.086632           -1.052           0.293
==============================================================================

Results for equation 5
==============================================================================
            coefficient       std. error           t-stat            prob
------------------------------------------------------------------------------
const         0.038110         0.094791            0.402           0.688
L1.1         -4.706545         4.356249           -1.080           0.280
L1.2         -0.742301         0.563945           -1.316           0.188
L1.3          0.022692         0.050875            0.446           0.656
L1.4         -0.329610         0.289246           -1.140           0.254
L1.5          0.318474         0.093983            3.389           0.001
L1.6         -0.818694         0.290173           -2.821           0.005
==============================================================================

Results for equation 6
==============================================================================
            coefficient       std. error           t-stat            prob
------------------------------------------------------------------------------
const         0.002724         0.031055            0.088           0.930
L1.1          2.281493         1.427146            1.599           0.110
L1.2          0.134317         0.184754            0.727           0.467
L1.3          0.000739         0.016667            0.044           0.965
L1.4         -0.108945         0.094760           -1.150           0.250
L1.5          0.017197         0.030790            0.559           0.576
L1.6          0.334841         0.095063            3.522           0.000
==============================================================================

Correlation matrix of residuals
          1         2         3         4         5         6
1    1.000000  0.163600  0.084212  0.135256  0.054834  0.203966
2    0.163600  1.000000 -0.041363  0.121994  0.070230  0.014092
3    0.084212 -0.041363  1.000000 -0.065499  0.112752  0.122225
4    0.135256  0.121994 -0.065499  1.000000  0.168433  0.309363
5    0.054834  0.070230  0.112752  0.168433  1.000000  0.127551
6    0.203966  0.014092  0.122225  0.309363  0.127551  1.000000
```

即，写成矩阵形式，模型最终为：

$$\begin{pmatrix} y_{1t} \\ y_{2t} \\ y_{3t} \\ y_{4t} \\ y_{5t} \\ y_{6t} \end{pmatrix} = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \end{pmatrix} + \begin{pmatrix} 0.003006 \\ 0.000407 \\ 0.127205 \\ 0.011866 \\ 0.038110 \\ 0.002724 \end{pmatrix}$$

$$+ \begin{pmatrix} 0.163221 & -0.029237 & 0.001539 & 0.000521 & 0.007118 & -0.011380 \\ 0.204820 & 0.505980 & 0.016170 & -0.036150 & -0.034109 & -0.093127 \\ -20.065400 & 0.142134 & 0.270497 & -1.066723 & 0.136125 & -0.715172 \\ 0.987921 & 0.126892 & 0.003195 & 0.525357 & -0.074237 & -0.091176 \\ -4.706545 & -0.742301 & 0.022692 & -0.329610 & 0.318474 & -0.818694 \\ 2.281493 & 0.134317 & 0.000739 & -0.108945 & 0.017197 & 0.334841 \end{pmatrix} \begin{pmatrix} y_{1,t-1} \\ y_{2,t-1} \\ y_{3,t-1} \\ y_{4,t-1} \\ y_{5,t-1} \\ y_{6,t-1} \end{pmatrix}$$

其中，$y_1, y_2, y_3, y_4, y_5, y_6$ 为六大类农产品集贸市场价格（小麦，花生仁，活猪，鲤鱼，黄瓜，香蕉）的一阶差分。

接下来检验模型：

首先，先通过绘制残差项自相关图来观察自相关性。可以看到，6 个变量基本都在边界范围内，无明显自相关性。

图 9: acf

残差项自相关性检验通过 Q 检验（Ljung-Box 方法）实现，Q 检验的原假设为 H_0: 检验最大滞后项 m 的自相关系数为 0，即 $\hat{\rho_1} = \hat{\rho_2} = \ldots\ldots = \hat{\rho_m} = 0$。在原假设成立的条件下，Q(m) 服从自由度为 m 的卡方分布。通过 statsmodels.tsa.stattools 中的 acf() 函数实现 Q 检验，p-Value 返回如下：

[array([0.52740219, 0.4384854 , 0.49037819, 0.64767928, 0.77306447, 0.79696484, 0.87616611, 0.87238704, 0.45788608, 0.55245569]),
array([0.49107943, 0.74735034, 0.51245593, 0.61830119, 0.16763974, 0.18792779, 0.23932733, 0.31682872, 0.36286832, 0.44770081]),
array([0.47051762, 0.76221575, 0.52713904, 0.63919973, 0.73425818, 0.80729527, 0.75780236, 0.81106089, 0.8668216 , 0.61634637]),
array([0.25724948, 0.42584887, 0.5648922 , 0.26907737, 0.14589302, 0.10697081, 0.15511247, 0.20354774, 0.22310506, 0.28581413]),
array([0.79108719, 0.54843777, 0.5303361 , 0.35351412, 0.48682279, 0.33834312, 0.43503051, 0.18594445, 0.2542709 , 0.30858343]),
array([0.23637255, 0.02322106, 0.02940898, 0.06009484, 0.08360409, 0.12794423, 0.18925959, 0.22232667, 0.1699084 , 0.0566266])]

都大于 0.05，证明无法拒绝原假设，可以证明变量的残差为白噪声。模型拟合十分良好。

利用 statsmodels.tsa.vector_ar.var_model.VARResults 的 forecast 做 4 步预测:

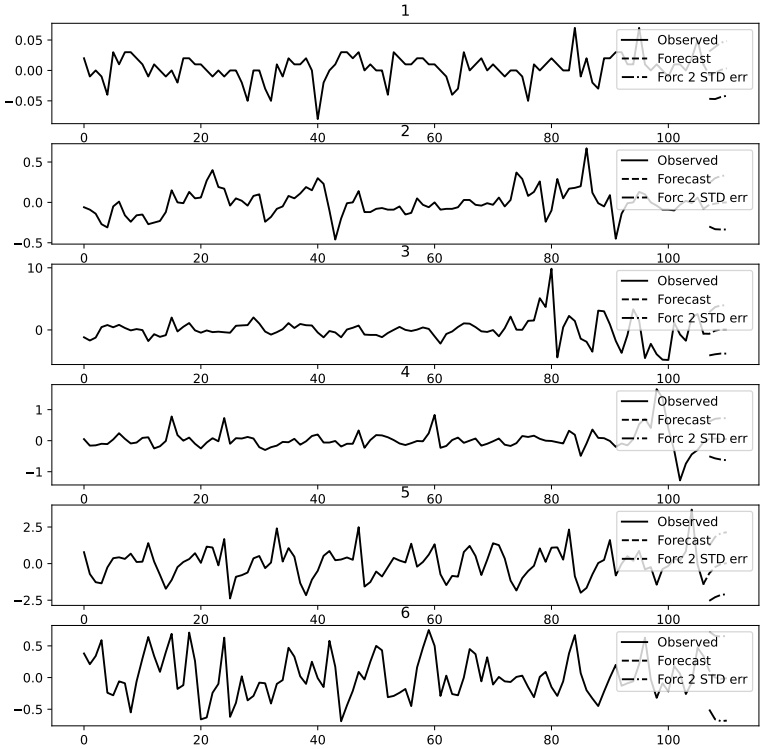

图 10: 四步预测

将预测结果的差分值与最后一期相加每一步在进行转换（转换代码见代码附录），将预测与拟合结果和原始的真实值一并作图展示如下：

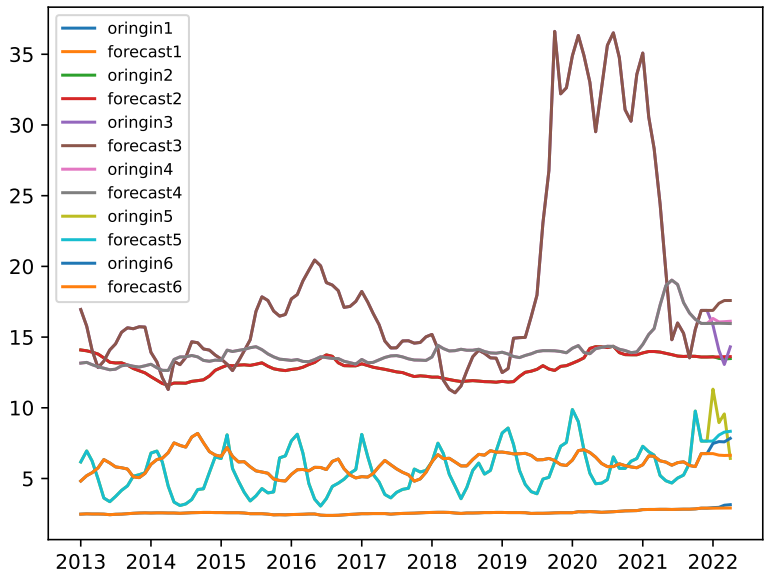

图 11: 'Forecast vs Actuals'

基本重合，模型拟合良好。

3.1.3 模型分析

通过一阶差分的 VAR(1) 模型，从模型结果来看：

首先，每个农产品的集贸市场价格都与自身的上个月的价格十分相关，即，上个月的价格很可能影响到下个月的价格。

其次，通过一阶差分使得每个序列平稳后，呈现出来的结果表明：

- 小麦价格的一阶差分与上个月的小麦、活猪、鲤鱼、黄瓜的一阶差分成正向的线性关系，与上个月的花生仁、香蕉的价格一阶差分有负向的线性关系；
- 花生仁价格的一阶差分与上个月的小麦、花生仁、活猪的一阶差分成正向的线性关系，与上个月的香蕉、鲤鱼、黄瓜的价格一阶差分有负向的线性关系；
- 活猪价格的一阶差分与上个月的花生仁、活猪、黄瓜的一阶差分成正向的线性关系，与上个月的小麦、鲤鱼、香蕉的价格一阶差分有负向的线性关系；
- 鲤鱼价格的一阶差分与上个月的小麦、花生仁、活猪、鲤鱼的一阶差分成正向的线性关系，与上个月的香蕉、黄瓜的价格一阶差分有负向的线性关系；
- 黄瓜价格的一阶差分与上个月的活猪、黄瓜的一阶差分成正向的线性关系，与上个月的小麦、花生仁、鲤鱼、香蕉的价格一阶差分有负向的线性关系；
- 香蕉价格的一阶差分与上个月的小麦、活猪、花生仁、香蕉、黄瓜的一阶差分成正向的线性关系，与上个月的鲤鱼的价格一阶差分有负向的线性关系；

- 所有自己的一阶差分都与自己的上个月一阶差分成正向线性关系。

这在经济学上可能是由于这些商品之间的互补品和替代品的关系在市场选择中呈现出来的规律。

3.2 对于居民消费价格指数 CPI 的影响

使用多元线性回归模型（multivariable linear regression model ）。

多元线性回归模型通常用来研究一个应变量依赖多个自变量的变化关系，如果二者的以来关系可以用线性形式来刻画，则可以建立多元线性模型来进行分析。多元线性回归模型通常用来描述变量 y 和 x 之间的随机线性关系，即：

$$y_i = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k + \xi_i \tag{2}$$

式中，$x_1, \cdots, x_k$ 是非随机的变量；y 是随机的因变量；$\beta_0, \cdots, \beta_k$ 是回归系数; ξ 是随机误差项。

用矩阵表示为：

$$y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad x = \begin{pmatrix} 1 & x_{11} & \ldots & x_{k1} \\ 1 & x_{12} & \ldots & x_{k2} \\ \vdots & \vdots & & \vdots \\ 1 & x_{1n} & \ldots & x_{kn} \end{pmatrix} \quad , \quad \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{pmatrix} \quad , \quad \xi = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix} \tag{3}$$

此时模型可以写作:

$$y_i = X_i \beta + \xi_i \tag{4}$$

建模步骤:

1. 根据数据建立回归模型;
2. 对模型进行显著性检验;
3. 对模型进行回归诊断

3.2.1 数据建模

由所选的特征列，加入常数 1 列，对 2022 年以前的 CPI 的调整值进行回归，结果如下方展示。回归方程为：

$$\hat{y_i} = 51.1172 + 16.8529 x_{1i} - 0.9055 x_{2i} + 0.2714 x_{3i} + 0.1127 x4i + 0.1127 x_{5i} + 1.2372 x_{6i}$$

且模型的每个参数的 p-Value 都很小，参数显著，初步说明模型拟合不错。

对残差进行正态性检验：

图12显示，残差的质量分布大致与零均值的正态分布相似，下面用 QQ 图[1]再检验。

基本在一条直线上，可以认为残差符合零均值的正态分布。

以上步骤证明模型拟合很好，模型通过了所有检验，因此考虑将模型与真实之间做比较来证明农产品集贸市场价格对于居民消费价格指数 CPI 的影响模型的准确性。

[1]QQ plot 的全称是 Quantile-Quantile Plot，即分位数-分位数图。如果两个分布相似，则该 Q-Q 图趋近于落在 y=x 线上。如果两分布线性相关，则点在 Q-Q 图上趋近于落在一条直线上，但不一定在 y=x 线上。Q-Q 图可以用来可在分布的位置-尺度范畴上可视化的评估参数。

OLS Regression Results

Dep. Variable:	cpi_adj	R-squared:	0.944
Model:	OLS	Adj. R-squared:	0.938
Method:	Least Squares	F-statistic:	181.2
Date:	Wed, 18 May 2022	Prob (F-statistic):	1.36e-38
Time:	10:26:56	Log-Likelihood:	-96.493
No. Observations:	72	AIC:	207.0
Df Residuals:	65	BIC:	222.9
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	51.1172	3.861	13.240	0.000	43.407	58.828
1	16.8529	1.821	9.253	0.000	13.215	20.490
2	-0.9055	0.277	-3.264	0.002	-1.460	-0.352
3	0.2714	0.024	11.125	0.000	0.223	0.320
4	0.6793	0.173	3.932	0.000	0.334	1.024
5	0.1127	0.087	1.291	0.201	-0.062	0.287
6	1.2372	0.244	5.063	0.000	0.749	1.725

Omnibus:	0.269	Durbin-Watson:	0.563
Prob(Omnibus):	0.874	Jarque-Bera (JB):	0.267
Skew:	0.135	Prob(JB):	0.875
Kurtosis:	2.875	Cond. No.	1.05e+03

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.05e+03. This might indicate that there are
strong multicollinearity or other numerical problems.

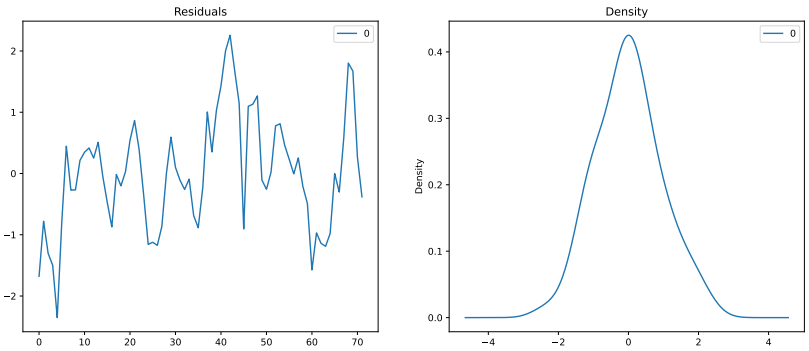

图 12: 残差质量分布

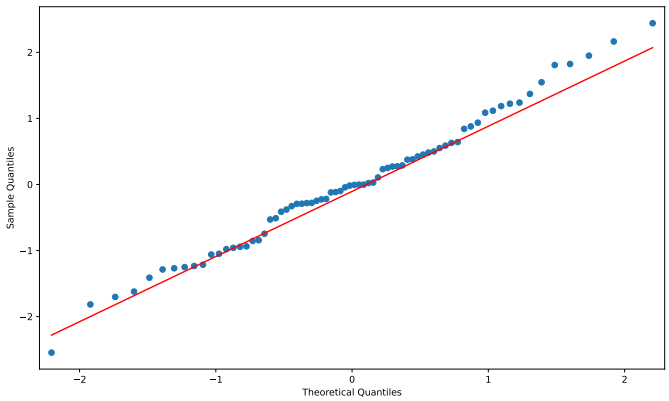

图 13: 残差 QQ 图

3.2.2 模型分析

此时，对于模型做 4 步预测结果如下：

113.203838
112.681031
114.567686
115.334756

真实情况为：

112.6169365064331,
113.2926381254717,
113.2926381254717,
113.74580867797359

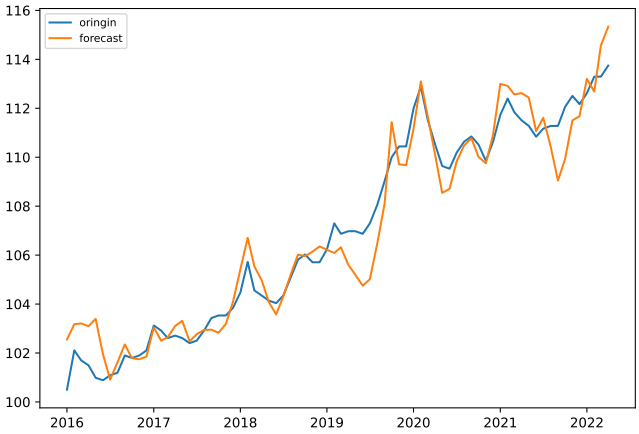

图 14: 'Forecast vs Actuals'

考虑到置信区间看到，基本上与原始数据契合。

最终，可以从模型中分析得到，CPI 调整值与六大类农产品集贸市场价格存在关系，且除了经济作物花生价格是负相关，与其他产品之间都存在正向的线性关系。

4 结论

通过两个模型对“粮食、经济作物、畜产品、水产品、蔬菜、水果”这六大类农产品集贸市场价格之间的联系，以及他们对于居民消费价格指数 CPI 的影响的结果可以看出，两个模型的拟合都非常好，都能够准确的对于现在的情况做出预测，从而为相关从业者的分析和操作提供非常有效的工具来参考。

如果想要对未来再分析，需要更多的近期数据进行迭代，使得两个模型再次发挥作用。

随着农业信息化的发展，价格数据的采集逐渐实现了短剑个采集，因此今后的研究中可利用更加海量的数据进行模拟预测，例如可利用每日价格数据进行预测，从而获得更加精确到超短期预测结果，或者利用周期更长的月度价格进行预测，研究长期变化规律。[5]

代码附录

部分 Python 代码展示如下：

```
import pandas as pd
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
import statsmodels.api as sm
from scipy import stats
from scipy.stats import kstest
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
data = pd.read_excel('月度数据 .xls')
name = data.columns
data = data.reindex(index=data.index[::-1])
data.index = [i for i in range(len(data))]
cpid = (data[[name[0],name[1]]].copy(deep=True)).dropna()
cpid.index = [i for i in range(len(cpid))]

from datetime import datetime
cpid[name[0]] = pd.to_datetime(cpid[name[0]])

date = cpid[name[0]]
cpi = cpid[name[1]]
plt.plot(date,cpi)
plt.savefig('cpi.pdf')
plt.show()

cpi_adj = []
cpi_adj.append(cpi[0])
for i in range(len(cpi)-1):
    c = cpi_adj[i]*cpi[i+1]/100
    cpi_adj.append(c)
plt.plot(date,cpi_adj)
plt.savefig('cpi_adj.pdf')
plt.show()

m1 = data[[name[3],name[6],name[8],name[12],name[15],name[18]]].copy(deep=True)
name1 = m1.columns

plt.rc('figure', figsize=(12, 7))
plt.text(0.01, 0.05, str(m1.corr().values), {'fontsize': 10}, fontproperties = '
                                                monospace') # approach improved by OP
                                                 -> monospace!
plt.axis('off')
plt.tight_layout()
# plt.savefig('corr.pdf')

adfResult1 = sm.tsa.stattools.adfuller(m1[name1[0]])
adfResult2 = sm.tsa.stattools.adfuller(m1[name1[1]])
adfResult3 = sm.tsa.stattools.adfuller(m1[name1[2]])
adfResult4 = sm.tsa.stattools.adfuller(m1[name1[3]])
adfResult5 = sm.tsa.stattools.adfuller(m1[name1[4]])
adfResult6 = sm.tsa.stattools.adfuller(m1[name1[5]])

m11 = m1.diff().dropna()
adfResult1 = sm.tsa.stattools.adfuller(m11[name1[0]])
adfResult2 = sm.tsa.stattools.adfuller(m11[name1[1]])
adfResult3 = sm.tsa.stattools.adfuller(m11[name1[2]])
adfResult4 = sm.tsa.stattools.adfuller(m11[name1[3]])
```

```
adfResult5 = sm.tsa.stattools.adfuller(m11[name1[4]])
adfResult6 = sm.tsa.stattools.adfuller(m11[name1[5]])

from statsmodels.tsa.stattools import coint
c=[]
for i in range(5):
    for j in range(i+1,6):
        c.append(coint(m11[name1[i]],m11[name1[j]])[1])

m10 = m11.head(len(m11)-5)
from statsmodels.tsa.vector_ar.var_model import VAR
mod = VAR(m10)
lag_order = mod.select_order()
print(lag_order.summary())
plt.rc('figure', figsize=(12, 7))
plt.text(0.01, 0.05, str(lag_order.summary()), {'fontsize': 10}, fontproperties = '
                                                monospace') # approach improved by OP
                                                 -> monospace!
plt.axis('off')
plt.tight_layout()
# plt.savefig('lag.pdf')

res = mod.fit(1)
res.summary()
# # 设置中文编码和符号的正常显示
# plt.rcParams["font.sans-serif"] = "SimHei"
# plt.rcParams["axes.unicode_minus"] = False
plt.rc('figure', figsize=(12, 20))
plt.text(0.01, 0.05, str(res.summary()), {'fontsize': 10}, fontproperties = '
                                                monospace') # approach improved by OP
                                                 -> monospace!
plt.axis('off')
plt.tight_layout()
# plt.savefig('var1.pdf')

# 绘制残差项自相关图，最大滞后系数=10
res.plot_acorr(nlags=10, resid=True, linewidth=6)
# plt.savefig('acf.pdf')
plt.show()

from statsmodels.tsa.stattools import acf
# 以USDCNY变量为例，调用acf函数获得Q检验结果
pvalue = [i for i in range(6)]
for i in range(6):
    (resid_acf, qstat, pvalue[i]) = acf(res.resid[i+1], nlags=10, qstat=True)

res.plot_forecast(4)
plt.savefig('figfore.pdf')
plt.show()

pred1 = res.forecast(m10.values,4)
pred =  []
p = [m1[name1[j]][len(m1)-5] for j in range(6)]
for i in range(4):
    pp = []
    for j in range(6):
        pp.append(p[j]+pred1[i][j])
    pred.append(pp)

```

```
m01 = m1.head(len(m1)-4)
for i in range(4):
    m01.loc[i+len(m01)] = pred[i]

    # import matplotlib as mpl
plt.style.use('default')
date1 = pd.to_datetime(data[name[0]])
# Plot
# plt.figure(figsize=(12,5), dpi=100)
for i in range(6):
    plt.plot(date1,m1.loc[:,name1[i]], label='oringin'+str(i+1))
    # plt.plot(test, label='actual')
    plt.plot(date1,m01.loc[:,name1[i]], label='forecast'+str(i+1))
# plt.fill_between(lower_series.index, lower_series, upper_series,
#                  color='k', alpha=.15)
# plt.title('Forecast vs Actuals')
plt.legend(loc='upper left', fontsize=8)
# plt.savefig('predict.pdf')
plt.show()

m2 = data[[name[3],name[6],name[8],name[12],name[15],name[18]]].copy(deep=True)
m2 = m2.dropna().tail(len(cpi_adj))
m2['cpi_adj'] = cpi_adj
m2.index = [str(i) for i in range(len(m2))]
name2 = m2.columns
# m2.columns = [i for i in range(7)]
# m2.rename(columns={'6':'cpi_adj'})
y = m2[name2[6]].head(len(y)-4)
x = m2[[name2[i] for i in range(6)]].head(len(x)-4)
x.columns = [i for i in range(1,7)]
# from statsmodels.formula.api import ols
import statsmodels.api as sm
import statsmodels.formula.api as smf
# 小写的 ols 函数才会自带截距项, OLS 则不会
# 固定格式: 因变量 ~ 自变量(+ 号连接)

x = sm.add_constant(x)
regression = sm.OLS(y, x) #用最小二乘法建模
model = regression.fit() #数据拟合
model.summary()
plt.rc('figure', figsize=(8, 5))
# plt.style.use('default')
plt.text(0.01, 0.05, str(model.summary()), {'fontsize': 10}, fontproperties = '
                                                monospace') # approach improved by OP
                                                 -> monospace!
plt.axis('off')
plt.tight_layout()
# plt.savefig('ols.pdf')

plt.rcParams.update({'figure.figsize':(15,6), 'figure.dpi':200})
residuals = pd.DataFrame(model.resid)
fig, ax = plt.subplots(1,2)
residuals.plot(title="Residuals", ax=ax[0])
residuals.plot(kind='kde', title='Density', ax=ax[1])
plt.savefig('res.pdf')
plt.show()

resid = model.resid
# plt.rcParams.update({'figure.figsize':(12,7), 'figure.dpi':100})
```

```
from statsmodels.graphics.api import qqplot

qqplot(resid, line='q', fit=True)
plt.savefig('qq.pdf')
plt.show()

test = m2
xtest = test[[name2[i] for i in range(6)]]
xtest = sm.add_constant(xtest)
pred = model.predict(exog=xtest)

import matplotlib.pyplot as plt
m22 = data[[name[0],name[1]]].dropna()
date2 = pd.to_datetime(m22[name[0]])

plt.plot(date2,cpi_adj, label='oringin')

plt.plot(date2,pred, label='forecast')

plt.legend(loc='upper left', fontsize=8)
plt.savefig('predict2.pdf')
plt.show()
```

参考文献

[1] 何叶, “2020 年第一季度农产品集贸市场价格分析,” *广东蚕业*, 2020.

[2] 国家统计局, “National data 国家数据.” `https://data.stats.gov.cn/easyquery.htm?cn=A01`, 2022. Accessed 2022.

[3] 马宇骁, “数学建模 hw2 实验报告（猪周期）,” *数学建模*, 2022.

[4] R. S. Tsay, *Analysis of financial time series*. John wiley & sons, 2005.

[5] 屠星月, 薛佳妮, 郭承坤, 封文杰, and 陈英义, “基于时间序列与 rbf 的农产品市场价格短期预测模型,” *广东农业科学*, vol. 41, no. 23, pp. 168–173, 2014.