МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информатики и прикладной математики

Кафедра прикладной математики и экономико-математических методов

ОТЧЁТ

по дисциплине:

«Математическое моделирование»

на тему:

«Статистические модели. Производственная функция Кобба-Дугласа.»

Направление (специальность)________________01.03.02_______________

(код, наименование)

Обучающийся ______________Бронников Егор Игоревич________________

(Ф.И.О. полностью)

Группа___ПМ-1901___

(номер группы)

Санкт-Петербург

2021

Часть 1. Параметрическая идентификация статических моделей.

Сначала нужно было найти коэффициенты (A, α, β), для этого воспользуемся историческими данными $(K_i,\ L_i,\ Y_i)_{i=1}^{M}$. В данном случае получается что $M = 15$. Рассмотрим 2 случая:

- 1 случай — $\alpha + \beta \neq 1$
- 2 случай — $\alpha + \beta = 1$

1 случай ($\boldsymbol{\alpha + \beta \neq 1}$)

Приведём функцию $Y = A\ K^{\alpha} L^{\beta}$ к линейному виду:

$$\ln(Y) = \ln(A) + \alpha\ln(K) + \beta\ln(L)$$

Проведём параметрическую идентификацию:

$$B \times \bar{C} = \bar{Z},\ где$$

$$B = \begin{pmatrix} M & \sum_{i=1}^{M}\ln(K_i) & \sum_{i=1}^{M}\ln(L_i) \\ \sum_{i=1}^{M}\ln(K_i) & \sum_{i=1}^{M}(\ln(K_i))^2 & \sum_{i=1}^{M}\ln(K_i)\ln(L_i) \\ \sum_{i=1}^{M}\ln(L_i) & \sum_{i=1}^{M}\ln(K_i)\ln(L_i) & \sum_{i=1}^{M}(\ln(L_i))^2 \end{pmatrix}$$

$$\bar{\mathrm{C}} = \begin{pmatrix} \ln(A) \\ \alpha \\ \beta \end{pmatrix}$$

$$\bar{Z} = \begin{pmatrix} \sum_{i=1}^{M}\ln(Y_i) \\ \sum_{i=1}^{M}\ln(Y_i)\ln(K_i) \\ \sum_{i=1}^{M}\ln(Y_i)\ln(L_i) \end{pmatrix}$$

тогда:

$$\begin{pmatrix} \ln(A) \\ \alpha \\ \beta \end{pmatrix} = B^{-1} \times \bar{Z}$$

Решив данное матричное уравнение получим желаемые коэффициенты. Только нужно не забыть:

$$A = e^{\ln(A)}$$

Функция, которая реализует данный алгоритм:

```
def coefficients1(GDP: List[int], K: List[int], L: List[int]) -> List[float]:
    """ Находит коэффициенты в производственной функции Кобба-Дугласа в случае когда у нас `alpha` + `beta` != 1

        :param GDP: Gross Domestic Product - ВВП
        :type GDP: List[int]
        :param K: Capital - капитал
        :type K: List[int]
        :param L: Labour - труд
        :type L: List[int]

        :return: список со значениями коэффициентов, где позиция: 0 -> `a`, 1 -> `alpha`, 2-> `beta`
        :rtype: List[float]
    """
    log_GDP = np.log(GDP)
    log_K = np.log(K)
    log_L = np.log(L)
    M = len(L)
    B = np.matrix([[M, np.sum(log_K), np.sum(log_L)],
                   [np.sum(log_K), np.sum(log_K**2), np.sum(log_K*log_L)],
                   [np.sum(log_L), np.sum(log_K*log_L), np.sum(log_L**2)]])
    Z = np.matrix([[np.sum(log_GDP)],
                   [np.sum(log_GDP*log_K)],
                   [np.sum(log_GDP*log_L)]])
    return np.array(np.dot(np.linalg.inv(B), Z)).flatten().tolist()
```

Таким образом, на выходе получили значения коэффициентов:

```
a1, alpha1, beta1 = coefficients1(GDP, K, L)
A1 = np.e**a1
print(A1, alpha1, beta1)
```

```
6.435886924194414e-05 0.3581662946526194 1.4818203831309802
```

$$A_1 = 0.000064358869;\ \alpha_1 = 0.358166295;\ \beta_1 = 1.48182$$

Функция, которая по формуле находит модельное ВВП:

```
def Y(A: float, alpha: float, beta: float, K: List[int], L: List[int]) -> float:
    """ Находит значение производственной функции Кобба-Дугласа

        :param A: коэффициент `A`
        :type A: float
        :param alpha: коэффициент `alpha`
        :type alpha: float
        :param beta: коэффициент `beta`
        :type beta: float
        :param K: Capital - капитал
        :type K: List[int]
        :param L: Labour - труд
        :type L: List[int]

        :return: значение функции Кобба-Дугласа
        :rtype: float
    """
    return A*K**alpha*L**beta
```

Таким образом, по исходным данным получили следующие модельные данные:

```
modelGDP1 = [Y(A1, alpha1, beta1, K[i], L[i]) for i in range(len(K))]
modelGDP1
```

```
[87529.18256192551,
 97861.77500398034,
 80197.7935222183,
 88797.7571289089,
 97991.4245214286,
 119733.64793525869,
 138600.11912201048,
 148331.82271765577,
 145734.5349721207,
 133494.06601823383,
 135126.88656692626,
 144259.46024796166,
 151415.75869893582,
 140048.81022179654,
 153376.28900715473]
```

2 случай ($\boldsymbol{\alpha + \beta = 1}$)

Приведём функцию $Y = A\,K^{\alpha}L^{(1-\alpha)}$ к линейному виду:

$$\ln\left(\frac{Y}{L}\right) = \ln(A) + \alpha\ln\left(\frac{K}{L}\right)$$

Проведём параметрическую идентификацию:

$$B \times \bar{C} = \bar{Z}, где$$

$$B = \begin{pmatrix} M & \sum_{i=1}^{M}\ln(\frac{K_i}{L_i}) \\ \sum_{i=1}^{M}\ln(\frac{K_i}{L_i}) & \sum_{i=1}^{M}(\ln\left(\frac{K_i}{L_i}\right))^2 \end{pmatrix}$$

$$\bar{C} = \begin{pmatrix} \ln(A) \\ \alpha \end{pmatrix}$$

$$\bar{Z} = \begin{pmatrix} \sum_{i=1}^{M}\ln(\frac{Y_i}{L_i}) \\ \sum_{i=1}^{M}\ln(\frac{Y_i}{L_i})\ln(\frac{K_i}{L_i}) \end{pmatrix}$$

тогда:

$$\begin{pmatrix} \ln(A) \\ \alpha \end{pmatrix} = B^{-1} \times \bar{Z}$$

Решив данное матричное уравнение получим желаемые коэффициенты. Только нужно не забыть:

$$A = e^{\ln(A)}$$

$$\beta = 1 - \alpha$$

Функция, которая реализует данный алгоритм:

```
def coefficients2(GDP: List[int], K: List[int], L: List[int]) -> List[float]:
    """ Находит коэффициенты в производственной функции Кобба-Дугласа в случае когда у нас `alpha` + `beta` = 1

        :param GDP: Gross Domestic Product - ВВП
        :type GDP: List[int]
        :param K: Capital - капитал
        :type K: List[int]
        :param L: Labour - труд
        :type L: List[int]

        :return: список со значениями коэффициентов, где позиция: 0 -> `a`, 1 -> `alpha`
        :rtype: List[float]
    """
    log_KL = np.log(np.array(K)/np.array(L))
    log_YL = np.log(np.array(GDP)/np.array(L))
    B = np.matrix([[M, np.sum(log_KL)],
                   [np.sum(log_KL), np.sum(log_KL**2)]])
    Z = np.matrix([[np.sum(log_YL)],
                   [np.sum(log_YL*log_KL)]])
    return np.array(np.dot(np.linalg.inv(B), Z)).flatten().tolist()
```

Таким образом, на выходе получили желаемые коэффициенты:

```
a2, alpha2 = coefficients2(GDP, K, L)
A2 = np.e**a2
beta2 = 1 - alpha2
print(A2, alpha2, beta2)
```

```
3.0516089478042185 -0.7729060269038968 1.7729060269038968
```

$$A_2 = 3.051608947;\ \alpha_2 = -0.772906;\ \beta_2 = 1.772906$$

Таким образом, по исходным данным получили следующие модельные данные:

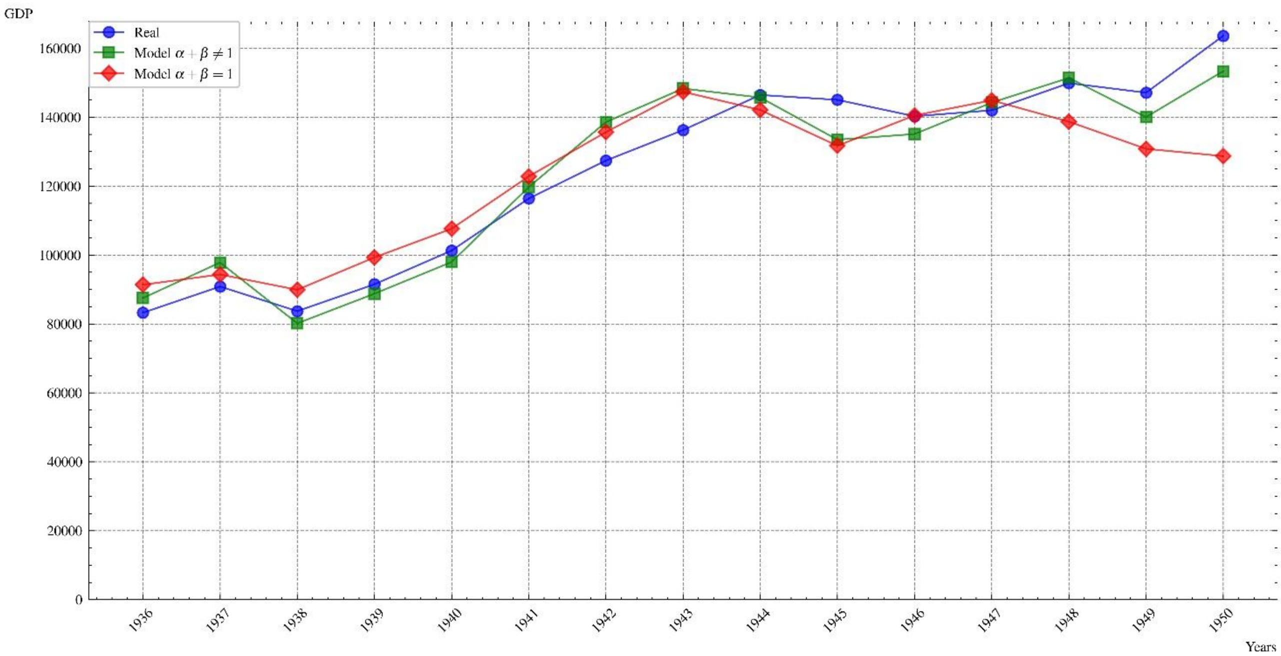
```
modelGDP2 = [Y(A2, alpha2, beta2, K[i], L[i]) for i in range(len(K))]
modelGDP2
```

```
[91408.33050108397,
 94378.69311505293,
 89940.98101806568,
 99323.9330728209,
 107647.30043949602,
 122827.76185949342,
 135686.26895065862,
 147371.3174195459,
 142127.0249122544,
 131727.02331960798,
 140520.7154262029,
 144907.02967622847,
 138716.6299519837,
 130837.6374054579,
 128732.87085961776]
```

График

Теперь построим график, чтобы посмотреть как модельные данные отличаются от реальных данных:

```
with plt.style.context(["science", "ieee", "grid"]):
    plt.rcParams["figure.figsize"] = [12, 6]
    plt.rcParams["axes.spines.top"] = False
    plt.rcParams["axes.spines.right"] = False
    plt.plot(GDP, "-bo", alpha=0.7)
    plt.plot(modelGDP1, "-gs", alpha=0.7)
    plt.plot(modelGDP2, "-rD", alpha=0.7)
    plt.xticks(np.arange(0, len(L)), years, rotation=45)
    plt.ylim(ymin=0)
    plt.legend([r"Real", r"Model $\alpha + \beta \neq 1$", r"Model $\alpha + \beta = 1$"], borderaxespad=0)
    plt.xlabel("Years", loc="right")
    plt.ylabel("GDP", loc="top", rotation=0)
    plt.savefig("./pics/plot.jpg")
    plt.show()
```



Выводы

Можно заметить, что модельные данные достаточно точно отражают реальную ситуацию.

Часть 2. Проведение экономического анализа производственного процесса на основе производственных функций.

Пусть нам дано какое-то начальное ВВП и капитал:

$$Y_1 = 50000; Y_2 = 100000;\ Y_3 = 150000$$

```
Y1 = 50000
Y2 = 100000
Y3 = 150000
```

Нужно по этим данным построить изокванты.

Находим труд из производственной функции Кобба-Дугласа:

$$L = \sqrt[\beta]{\frac{Y}{A\,K^{\alpha}}}$$

Функция, которая рассчитывает труд по заданным аргументам:

```
def cobbDuglas(A: float, alpha: float, beta: float, Y: List[int], K: List[int]) -> float:
    """ Находит Labour (L) труд по входным параметрам

        :param A: коэффициент `A`
        :type A: float
        :param alpha: коэффициент `alpha`
        :type alpha: float
        :param beta: коэффициент `beta`
        :type beta: float
        :param Y: GDP - ВВП
        :type Y: List[int]
        :param K: Capital - капитал
        :type K: List[int]

        :return: значение функции Кобба-Дугласа
        :rtype: float
    """
    return (Y/(A*K**alpha))**(1/beta)
```

Теперь рассчитываем труд по заданным данным:

```
ys1 = [cobbDuglas(A1, alpha1, beta1, Y1, k) for k in range(100, 200000, 200)]
ys2 = [cobbDuglas(A1, alpha1, beta1, Y2, k) for k in range(100, 200000, 200)]
ys3 = [cobbDuglas(A1, alpha1, beta1, Y3, k) for k in range(100, 200000, 200)]
```

Изокванты

```
xs = np.arange(100, 200000, 200)
with plt.style.context(["science", "ieee", "grid"]):
    plt.rcParams["figure.figsize"] = [12, 6]
    plt.rcParams["axes.spines.top"] = False
    plt.rcParams["axes.spines.right"] = False
    plt.plot(xs, ys1, "-b", alpha=0.7)
    plt.plot(xs, ys2, "-g", alpha=0.7)
    plt.plot(xs, ys3, "-r", alpha=0.7)
    plt.xlim(xmin=0)
    plt.legend([r"GDP 50k", r"GDP 100k", r"GDP 150k"], borderaxespad=0)
    plt.xlabel("L", loc="right")
    plt.ylabel("K", loc="top", rotation=0)
    plt.savefig("./pics/isoquant.jpg")
    plt.show()
```