**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение высшего образования**

**«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ УНИВЕРСИТЕТ имени академика С.П.КОРОЛЕВА (национальный исследовательский университет)»**

*ФАКУЛЬТЕТ ИНФОРМАТИКИ*

*КАФЕДРА ТЕХНИЧЕСКОЙ КИБЕРНЕТИКИ*

СТАТИСТИЧЕСКИЙ АНАЛИЗ И МОДЕЛИРОВАНИЕ

ПРОЦЕССОВ АВТОРЕГРЕССИИ И СКОЛЬЗЯЩЕГО СРЕДНЕГО

курсовая работа по дисциплине «**Теория случайных процессов**»

Вариант № 8

|  |  |
| --- | --- |
| Выполнил: | Лян А.И., гр.6307 |
| Проверил: | Храмов А.Г. |
| Оценка: |  |
| № зачётки: | 126100 |

Самара, 2015

**АННОТАЦИЯ**

В данной курсовой работе проводится исследование выборки из отсчётов некоторого неизвестного стационарного в широком смысле эргодического случайного процесса и моделирование нового процесса, подобного исходному процессу, с использованием моделей авторегрессии и скользящего среднего различных порядков. Модели АРСС исследуются на качество, проводится построение графиков нормированной корреляционной функции для исходного и смоделированного процессов.

Оглавление

[Оглавление 3](#_Toc418522076)

[1 Задание и исходные данные 4](#_Toc418522077)

[1.1 Исходные данные 4](#_Toc418522078)

[1.2 Содержание задания 4](#_Toc418522079)

[2 Ход выполнения работы 5](#_Toc418522080)

[2.1 Оценка моментных функций 5](#_Toc418522081)

[2.2 Построение и анализ моделей 7](#_Toc418522082)

[2.2.1 Построение и анализ моделей авторегрессии 8](#_Toc418522083)

[2.2.2 Построение и анализ моделей скользящего среднего 10](#_Toc418522084)

[2.2.3 Построение модели авторегрессии-скользящего среднего 11](#_Toc418522085)

# 1 Задание и исходные данные

## 1.1 Исходные данные

Дана реализация стационарного в широком смысле эргодического случайного процесса с дискретным временем (стационарная случайная последовательность, временной ряд) – выборка из 5000 последовательных значений (отсчётов) процесса.

Исходная выборка:

154.264 139.550 133.831 158.976 144.304 169.576 142.164 158.611 134.400 162.616

139.566 161.016 138.871 163.636 139.895 158.591 152.235 170.297 138.750 146.930

…

131.111 146.492 132.057 163.075 149.406 168.107 138.318 135.770 124.960 167.773

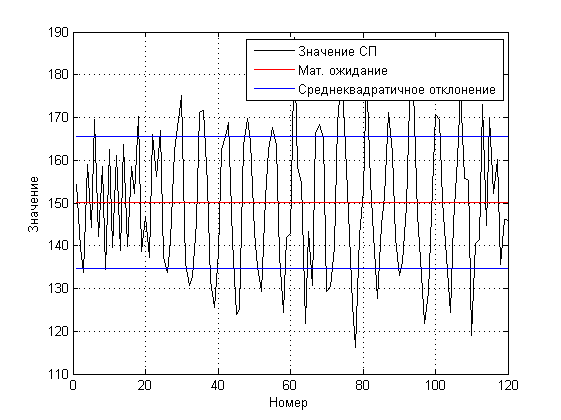
## 1.2 Содержание задания

1. Оценить моментные функции случайного процесса, рассчитав выборочное среднее, выборочную дисперсию и выборочную нормированную корреляционную функцию. Оценить радиус корреляции случайного процесса. Изобразить графически оценку нормированной корреляционной функции.
2. Построить модели авторегрессии АР(М)=АРСС(М,0) порядков М = 1,2,3, на основе решения системы уравнений Юла-Уокера. Для каждой модели рассчитать теоретические нормированные корреляционные функции выходной последовательности. На основе сравнения выборочной и теоретических нормированных корреляционных функций выбрать наилучшую (наиболее адекватную) модель случайного процесса.
3. Построить модели авторегрессии СС(N)=АРСС(0,N) порядков N = 1,2,3 на основе решения системы нелинейных уравнений. Для каждой модели рассчитать теоретические нормированные корреляционные функции выходной последовательности. На основе сравнения выборочной и теоретических нормированных корреляционных функций выбрать наилучшую модель случайного процесса.
4. Построить смешанные модели авторегрессии – скользящего среднего АРСС(M,N) до третьего порядка включительно (М =1,2,3; N = 1,2,3), на основе решения системы нелинейных уравнений. Рассчитать теоретические нормированные корреляционные функции выходной последовательности. На основе сравнения выборочной и теоретических нормированных корреляционных функций выбрать наилучшую модель случайного процесса. Сравнить параметры смешанных моделей АРСС, рассчитанные по двум методам. Для второго метода рассчитать и изобразить нормированную корреляционную функцию промежуточной случайной последовательности.
5. Для наилучшей модели: записать системы уравнений для расчёта параметров; записать системы уравнений для расчёта теоретической корреляционной функции; смоделировать случайный процесс; сравнить графически фрагменты реализаций исходного и смоделированного процессов; сравнить графически выборочную нормированную корреляционную функцию смоделированного процесса с выборочной нормированной корреляционной функцией исходного процесса и с теоретической нормированной корреляционной функцией.
6. Построить оценки моментных функций смоделированных процессов, сравнить их с оценками моментных функций исходного процесса и с теоретическими моментными функциями, соответствующими выбранной модели АРСС.

# 2 Ход выполнения работы

## 2.1 Оценка моментных функций

Пусть дана выборка  из  отсчётов стационарного в широком смысле эргодического дискретного случайного процесса .



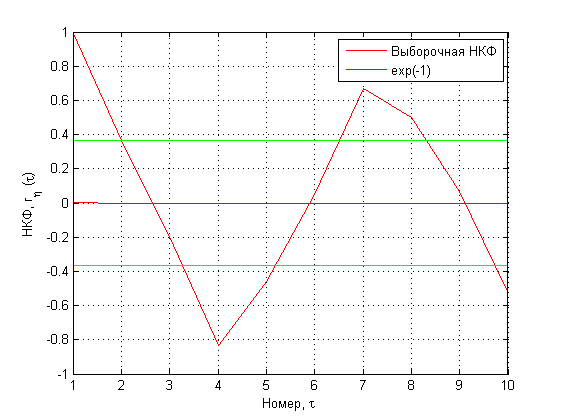
*Рисунок 1 – Фрагмент выборки*

Оценим моментные функции этого процесса.

1. Находим выборочное среднее (оценка математического ожидания ) по формуле , где  – соответствующие компоненты вектора. Результат расчета: .
2. Формула для расчёта выборочной дисперсии имеет вид: . Воспользуемся исправленной дисперсией , так как она является лучшей (несмещённой) оценкой дисперсии процесса . Результат расчета:.
3. В соответствии с оценкой дисперсии выберем формулу для оценки корреляционной функции : . Это исправленная выборочная корреляционная функция. В таблицу 1 запишем 11 первых неотрицательных значений этой функции, помня что: . Заметим, что 
4. Оценим также и нормированную корреляционную функцию, которая поможет количественно оценить корреляцию сечений: . Её первые значения также занесём в таблицу 1.

*Таблица 1 – Первые значения выборочной корреляционной и нормированной корреляционной функций*

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 0 | 237.5174 | 1.0000 |
| 1 | 86.4783 | 0.3641 |
| 2 | -47.0829 | -0.1982 |
| 3 | -197.7249 | -0.8325 |
| 4 | -109.0848 | -0.4593 |
| 5 | 11.9928 | 0.0505 |
| 6 | 159.2884 | 0.6706 |
| 7 | 118.7856 | 0.5001 |
| 8 | 16.2701 | 0.0685 |
| 9 | -122.7267 | -0.5167 |
| 10 | -119.6584 | -0.5038 |

1. Оценим радиус корреляции случайного процесса  по формуле:, только вместо , используем , и  будем полагать не слишком большим по сравнению с размером выборки во избежание чрезмерной ошибки: . В результате вычислений, полагая , получаем .
2. Изобразим графически на рисунке 2 оценку корреляционной функции.

*Рисунок 2 – Оценка нормированной корреляционной функции*

## 2.2 Построение и анализ моделей

Теперь построим все модели АРСС для порядков  и , не превышающих 3



где  – белый шум (случайная последовательность независимых и одинаково распределённых случайных величин с нулевым средним), - выходная случайная последовательность.

Затем полученные модели проверим на устойчивость исходя из условия, что все M корней  характеристического уравнения лежат внутри окружности единичного радиуса z-комплексной плоскости, т.е. . В частности:

* модель APCC (*0,N*) = СС (*N*) устойчива всегда,
* модель APCC (*1,N*) устойчива тогда и только тогда, когда ,
* модель APCC (*2,N*) устойчива тогда и только тогда, когда ,,

модель APCC (*3,N*) устойчива тогда и только тогда, когда , , . На устойчивость модели не влияют значения коэффициентов 

### 2.2.1 Построение и анализ моделей авторегрессии

Для построения моделей авторегрессии решим системы уравнений Юла-Уокера соответствующих порядков.



Для каждой модели АР(M) составим систему уравнений, для нахождения коэффициентов . Для модели АР(1): ;

Для модели АР(2): ;

Для модели АР(3):

Результаты вычислений занесём в таблицу 2.

*Таблица 2 – Результат построения моделей АР(M)*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Порядок модели | Параметры модели | | | |
| *M* |  |  |  |  |
| *1* | 0,3641 | - | - | 14,3524 |
| *2* | 0,5029 | -0,3813 | - | 13.2678 |
| *3* | 0,1974 | 0,0216 | -0,8012 | 7.9394 |

Неизвестные коэффициенты и находятся с помощью математического пакета Матлаб. Для начала решается последние M-1 уравнений (матричным способом), откуда находятся коэффициенты СС (бета). Далее, находим коэффициент альфа нулевое при всех известных значениях остальных переменных.

Проверив на устойчивость, выяснили, что условия устойчивости выполняются для всех моделей авторегрессии. Выбор наилучшей модели проведем на основе анализа нормированных корреляционных функций с использованием критерия среднего квадратичного отклонения по первым десяти отсчётам:, где  – выборочная нормированная корреляционная функция исходного процесса,  – рассчитанная теоретическая нормированная корреляционная функция, (M,N) – порядок модели АРСС. Будем считать, что первые N+M+1 отсчетов теоретической корреляции совпадают со значениями корреляционной функции исходного процесса, а остальные значения отыщем из системы: . Результаты вычислений  занесём в таблицу.

Рассмотрим подробнее расчет НКФ на примере модели АР(3). Для нее имеем N=0 и М=3. То есть первые 4 значения теоретической НКФ совпадут. Для остальных 10-4=6 значений будем приведенную систему, которая примет вид:



*Таблица 3 – Теоретические значения НКФ модели АР(3)*

|  |  |
| --- | --- |
| Номер | Значение |
| 0 | 1.0 |
| 1 | 0.3641 |
| 2 | -0.1982 |
| 3 | -0.8325 |
| 4 | -0.4603 |
| 5 | 0.0500 |
| 6 | 0.6669 |
| 7 | 0.5015 |
| 8 | 0.0734 |
| 9 | -0.5090 |
| 10 | -0.5007 |

*Таблица 4* – *Теоретические погрешности моделей АР(M)*

|  |  |  |  |
| --- | --- | --- | --- |
| ***M*** | 1 | 2 | 3 |
|  | 2,335346 | 1,679564 | 0,000106 |

Таким образом, получаем, что наилучшей моделью для АР является модель порядка 3 или АР(3).

### 2.2.2 Построение и анализ моделей скользящего среднего

Система уравнений для нахождения параметров модели скользящего среднего СС(N):



Для каждой модели CC(N) составим систему уравнений, для нахождения коэффициентов .

* Для модели СС(0): .
* Для модели СС(1): .
* Для модели СС(2): .
* Для модели СС(3):.

Результаты вычислений занесем в таблицу 5.

*Таблица 5 – Результат построения моделей СС(N)*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Порядок модели*** | ***Параметры модели*** | | | |
| *N* | 1 | 2 | 3 | 4 |
| *0* | 1.4301 |  |  |  |
| *1* | 5.4122 | 15.9784 |  |  |
| *2* | Модель не существует | | | |
| *3* | Модель не существует | | | |

Неизвестные коэффициенты находились с помощью математического пакета Scilab. Для этого применим функцию alphas(x,M,N) передавая ей в качестве параметров выборку процесса и порядок модели. В ней мы с помощью функции fsolve (стандартная функция SciLab для решения нелинейных систем уравнений) решаем систему уравнений для модели порядка N.

Также, проверяем модель на существование. Для этого смотрим все коэффициенты СС порядка (N), и если хотя бы один из них имеет ненулевую мнимую часть, то считаем, что данная модель не существует.

Далее найдем теоретические нормированные корреляционные функции выходной последовательности для каждой модели скользящего среднего, подставляя найденные оценки неизвестных параметров модели СС в систему:

Модель CC(N) устойчива всегда. Исследуем полученные модели на качество. Для этого аналогично случаю АР(M) построим теоретические нормированные корреляционные функции  и сравним с исходными корреляционными функциями  с помощью критерия среднего квадратичного по первым десяти отсчётам. Полученные значения  занесём в таблицу 5.

*Таблица 6* – *Теоретические погрешности моделей СС(N)*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***N*** | 0 | 1 | 2 | 3 |
|  | 2.3359 | 2.2868 | - | - |

Таким образом, получаем, что наилучшей моделью для CC является модель порядка 1 или СС(1).

### 2.2.3 Построение модели авторегрессии-скользящего среднего

Для нахождения параметров модели авторегрессии – скользящего среднего необходимо решить системы уравнений, соответствующих порядков.

, где 

Для этого:

1. Из системы M линейных уравнений

находятся оценки M параметров .

1. Последовательно выражаются оценки взаимной корреляционной функции

и подставляются в N уравнений системы для нахождения параметров АРСС (см. выше).

1. Из системы (N+1) нелинейных уравнений 

находятся оценки параметров .

.

*Таблица 7 – Параметры модели АРМА*

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Порядок модели*** | | ***Параметры модели*** | | | | | | | | |
| *M* | *N* |  |  |  |  |  | |  | |  |
| *1* | *1* | Модель не существует | | | | | | | | |
| *1* | *2* | Модель не существует | | | | | | | | |
| *1* | *3* | Модель не существует | | | | | | | | |
| *2* | *1* | Модель нестабильна | | | | | | | | |
| *2* | *2* | Модель не существует | | | | | | | | |
| *2* | *3* | Модель не существует | | | | | | | | |
| *3* | *1* | 0.1956 | 0.0225 | -0.8019 | 0.0394 | | 7.9401 | |  |  |
| *3* | *2* | 0.1957 | 0.0223 | -0.8018 | 0.0043 | | 0.0387 | | 7.9401 |  |
| *3* | *3* | 0.1921 | 0.0258 | -0.8082 | 0.1320 | | -0.0185 | | 0.0677 | 7.9394 |

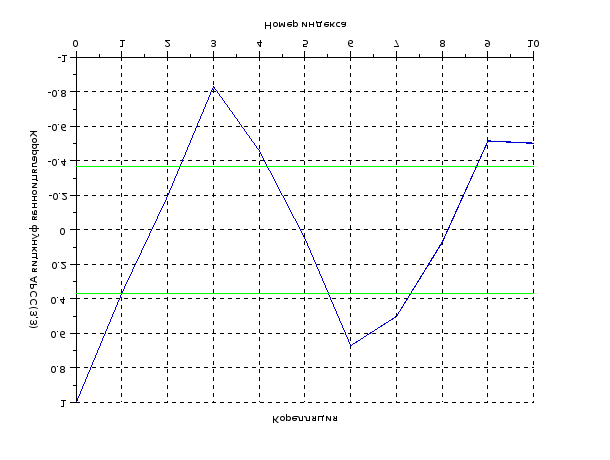
Неизвестные коэффициенты находим с помощью математического пакета Scilab. Для нахождения параметров по первому методу применим функцию alphas(x,M,N) и betas(x, M,N) передавая ей в качестве параметров нормированную корреляционную функцию исходного процесса и порядок модели. В ней мы с помощью функции fsolve (стандартная функция SciLab для решения нелинейных систем уравнений) решаем систему уравнений для модели порядка M, N.

Исследуем полученные модели на качество. Для этого аналогично случаю АР(M) и СС(N) строим теоретические нормированные корреляционные функции  и сравниваем с исходными корреляционными функциями  с помощью критерия среднего квадратичного по первым десяти отсчётам. Результат  приведен в таблице 8.

*Таблица 8* – *Теоретические погрешности моделей АРСС(M,N)*

|  |  |  |  |
| --- | --- | --- | --- |
| ***M*** | ***N*** | | |
| *1* | *2* | *3* |
| *1* | - | - | - |
| *2* |  | - | - |
| *3* | 0.000104 | 0.000100 | 0.000025 |

Таким образом, получаем, что наилучшей моделью авторегрессии-скользящего среднего является модель АРСС(3,3). Ее НКФ изображена на рисунке 3.



*Рисунок 3 – Оценка нормированной корреляционной функции промежуточного случайного процесса для модели АРСС(3,3)*

Код программы (среда Матлаб)

**Main.m**

output = 'output.xlsx';

tsp\_data = load('data.txt'); % загрузка данных

% print\_data(tsp\_data); % вывод последовательности

n= length(tsp\_data);

mx = sum(tsp\_data)/n; % матожидание

dx=0;

for i=1:n

dx=dx+(tsp\_data(i)-mx)^2;

end;

dx=dx/(n-1);

[cf, ncf] = cf\_ncf(tsp\_data,20); % корелляционная функция и норм. кф

rad=corel\_rad(ncf); % радиус корреляции

%plot\_ncf(ncf); % график нкф

beta\_ar1=beta\_ar(cf, 1);

beta\_ar2=beta\_ar(cf, 2);

beta\_ar3=beta\_ar(cf, 3); % нашли коэффициенты СС

%

alpha\_0\_ar1=alpha\_0\_ar(beta\_ar1,cf,1);

alpha\_0\_ar2=alpha\_0\_ar(beta\_ar2,cf,2);

alpha\_0\_ar3=alpha\_0\_ar(beta\_ar3,cf,3);% нашли оставшийся коэффициент альфа

% theor\_ncf(1,:)=theor\_ncf\_ar(beta\_ar1,alpha\_0\_ar1,1);

% theor\_ncf(2,:)=theor\_ncf\_ar(beta\_ar2,alpha\_0\_ar2,2);

% theor\_ncf(3,:)=theor\_ncf\_ar(beta\_ar3,alpha\_0\_ar3,3);

% графическое отображение моделей СП

osx=0:10;

plot(osx,ncf(1:11),'k');

hold on;

grid on;

plot(osx,theor\_ncf(1,1:11),'r');

plot(osx,theor\_ncf(2,1:11),'g');

plot(osx,theor\_ncf(3,1:11),'b');

xlabel('Номер НКФ, \tau');

ylabel('Значение НКФ, r\_\eta (\tau)');

legend('Выборочные НКФ','НКФ модели АР(1)','НКФ модели АР(2)','НКФ модели АР(3)');

hold off;

% оценка погрешностей

ar\_error=zeros(3,20);

for i=1:3

ar\_error(i)=ncf\_error(theor\_ncf(i,:),ncf(:));

end;

**plot\_ncf.m**

function [] = plot\_ncf( ncf )

% вывод значений, для 1 задания

n=length(ncf);

plot(0:n-1,ncf,'r');

hold on;

grid on;

plot([0 n],[exp(-1) exp(-1)],'g')

plot([0 n],[-exp(-1) -exp(-1)],'g')

ylabel('НКФ, r\_\eta (\tau)');

xlabel('Номер, \tau');

legend('Выборочная НКФ','exp(-1)');

hold off;

end

**corel\_rad.m**

function [ correlationRadius ] = corel\_rad(NCF)

%функция для расчета радиуса корреляции по известной нкф

i=length(NCF);

if (abs(NCF(i))>exp(-1))

disp('Необходимо больше значений НКФ')

end;

while abs(NCF(i))<exp(-1)

i=i-1;

end;

correlationRadius=i;

end

**print\_data.m**

function [] = print\_data(data)

% вывод исходных данных

n=length(data);

ox=1:120;

plot(ox,data(1:120),'k') hold on;

grid on;

data'

mx=sum(data)/n;

dx=0;

for i=1:n

dx=dx+(data(i)-mx)^2;

end;

dx=dx/(n-1);

dx=sqrt(dx);

plot([1 120],[mx mx],'r');

plot([1 120],[mx+dx mx+dx],'b'

plot([1 120],[mx-dx mx-dx],'b');

ylabel('Значение');

xlabel('Номер');

legend('Значение СП','Мат. ожидание','Среднеквадратичное отклонение');

hold off;

end

**cf\_ncf.m**

function [ cf, ncf ] = cf\_ncf(input,number\_cf)

%возвращает значения НКФ и КФ от выборки input, в том числе - нестационарной

n=length(input);

mx=sum(input)/n;

dx=0;

for i=1:n

dx=dx+(input(i)-mx)^2;

end;

dx=dx/(n-1);

dx=sqrt(dx);

cf(1:number\_cf)=0.0; ncf(1:number\_cf)=0.0;

cf(1)=dx; ncf(1)=1.0;

x1=0.0; x2=0.0;

y1=0.0; y2=0.0;

for k=2:number\_cf

x1=mx\*n; x2=x1; y1=(dx+mx^2)\*n; y2=y1;

for j=n-k+2:n x1=x1-input(j); end; x1=x1/(n-k+1); %MX1

for j=1:k-1 x2=x2-input(j); end; x2=x2/(n-k+1); %MX2

for j=n-k+2:n y1=y1-input(j)^2; end; y1=y1/(n-k+1)-x1^2; %DX1

for j=1:k-1 y2=y2-input(j)^2; end; y2=y2/(n-k+1)-x2^2; %DX2

%кф

for j=1:n-k+1

cf(k)=cf(k)+(input(j)-x1)\*(input(j+k-1)-x2);

end;

cf(k)=cf(k)/(n-k+1);

% нкф

ncf(k)=cf(k)/sqrt(y1\*y2);

end;

end

**beta\_ar.m**

function [ BettaAR ] = beta\_ar(cf,M)

% находим коэффициенты авторегрессии

% путем решения системы линейных неоднороных уравнений

% сама система есть в отчете

B=cf(2:M+1)';

matrixAR=zeros(M);

for i=1:M

for j=1:M

matrixAR(i,j)=cf(abs(i-j)+1);

end;

end;

BettaAR=matrixAR\B;

end

alpha\_0\_ar**.m**

function [ alpha0AR ] = alpha\_0\_ar( BettaAR,cf,M)

% нахождение коэффициента альфа 0 для модели авторегрессии

% параметры - BettaAR - коэффициенты АР, должны быть ранее найдены

% CF оценка корреляционной функции

% M порядок модели АР

alpha0AR=cf(1);

for i=1:M

alpha0AR=alpha0AR-BettaAR(i)\*cf(i+1);

end;

alpha0AR=sqrt(alpha0AR);

end

(Далее приведен код для среды Scilab 5.5.2)

*// Задание 2*

function **betas**=ar(**x**, **arLevel**, **maLevel**)

if **arLevel**=0 then **arLevel**=1

end

R = zeros(2 \* **arLevel**, 1);

for i = (**maLevel** - **arLevel** + 1) : (**maLevel** + **arLevel**),

R(i - **maLevel** + **arLevel**) = correlation(i, **x**);

end;

Rmm = zeros(**arLevel**, **arLevel**);

\_n = **maLevel**;

for i = 1:**arLevel**, *// ошибка при arLevel=0*

\_m = \_n;

for j = 1:**arLevel**,

Rmm(i, j) = R(\_m - **maLevel** + **arLevel**);

\_m = \_m - 1;

end;

\_n = \_n + 1;

end;

Rm = -R(**arLevel** + 1 : 2 \* **arLevel**);

**betas** = linsolve(Rmm, Rm);

endfunction;

*// Корреляционная функция*

function **rrr**=mcorrelation(**k**, **alph**, **betas**)

**rrr** = **alph**(**k**+1);

len = min(**k**, length(**betas**));

for j = 1 : len,

**rrr** = **rrr** + **betas**(j) \* mcorrelation(**k** - j, **alph**);

end;

endfunction;

*// Поиск коэффициентов скользящего среднего*

function **alphas**=ma(**x**, **arLevel**, **maLevel**, **betas**)

for i = 0 : max([**arLevel**, **maLevel**]),

R(i+1) = correlation(i, **x**);

end;

function **zr**=syst(**alph**)

for k = 0 : **maLevel**,

**zr**(k+1) = -R(k+1);

for i = k : **maLevel**,

**zr**(k+1) = **zr**(k+1) + **alph**(i+1) \* mcorrelation(i - k, **alph**, **betas**);

end;

for j = 1 : **arLevel**,

**zr**(k+1) = **zr**(k+1) + **betas**(j) \* R(abs(k - j) + 1);

end;

end;

endfunction;

[alphas, values, info] = fsolve([1 : (maLevel+1)], syst);

for i = 1:length(values),

if (abs(values(i)) > EPSILON | info == 4) then

alphas(1) = %i;

break;

end;

end;

endfunction;

*// Вектор*

function **s**=image(**v**)

**s** = %F;

for i = 1:length(**v**),

if (imag(**v**(i)) <> 0) then

**s** = %T;

break;

end;

end;

endfunction;

*// стабильность модели*

function **s**=stable(**betas**)

p = poly([pertrans(-**betas**) 1], "z", "coeff");

z = roots(p);

**s** = %T;

for i = 1:length(z),

if (abs(z(i)) >= 1) then

**s** = %F;

break;

end;

end;

endfunction;

*// Main*

alphas\_list = list();

betas\_list = list();

cf=list();

*// для одного прохода и нахождения только лишь ар*

MAX\_AR\_LEVEL=0; *// приравнять MAX\_AR\_LVEL 0 для нахождения коэффициентов сс*

*// при MAX\_MA\_LEVEL=3 И MAX\_AR\_LEVEL=3 ищутся все модели АРСС(1-3,1-3)*

for i = 0 : MAX\_AR\_LEVEL,

for j = 0 : MAX\_MA\_LEVEL,

betas = ar(x, i, j); *// коэффиценты АР*

alphas = ma(x, i, j, betas); *// СС*

alphas\_list($+1) = alphas;

betas\_list($+1) = betas;

printf("ARMA(" + INT\_FORMAT + "," + INT\_FORMAT + ")\n", i, j);

if (image(alphas)) then *// проверка существования*

printf("Model does not exist.\n");

continue;

end;

if (~stable(betas)) then *// проверка устойчивости*

printf("Model exists, but not stable.\n");

continue;

end;

printf("alpha:\n")

printMat(alphas, FLOAT\_FORMAT);

printf("beta:\n");

printMat(betas, FLOAT\_FORMAT);

end;

end;

*//Задание3*

*// Теорет. корреляция для АРМА*

function **R**=theoretical\_corr(**betas**, **startR**, **k**)

nm = length(**startR**) - 1;

**k** = abs(**k**);

if (**k** > nm) then

**R** = 0;

M = length(**betas**);

for j = 1 : M,

**R** = **R** + **betas**(j) \* theoretical\_corr(**betas**, **startR**, **k** - j);

end;

else

**R** = **startR**(**k** + 1);

end;

endfunction;

*// Норм. теор. корр. для АРМА*

function **r**=norm\_theoretical\_corr(**betas**, **startR**, **k**)

**r** = theoretical\_corr(**betas**, **startR**, **k**) / theoretical\_corr(**betas**, **startR**, 0);

endfunction;

*// Квадратичное отклонение*

function **epsilon**=quadratic\_error(**x**, **y**)

**epsilon** = 0;

m = min(length(**x**), length(**y**));

for j = 1 : m,

**epsilon** = **epsilon** + (**x**(j) - **y**(j))^2;

end;

endfunction;

function corrplot2(**betas**, **R**, **N**, **M**, **m\_**)

p.thickness = 6;

m = **m\_**;

t = [-m:m];

y = zeros(length(t), 1);

for i = 1:length(t),

y(i) = norm\_theoretical\_corr(**betas**, **R**(1 : **N** + **M** + 1), t(i));

end;

plot2d3(t, y, axesflag=5, style=2);

a = gca();

p = a.children.children;

p.thickness = 3;

p.mark\_mode = "on";

p.mark\_size\_unit = "point";

p.mark\_style = 11;

p.mark\_size = 3;

endfunction;

function **eta**=imitate(**alphas**, **betas**, **meanx**, **count**)

defect = 1000;

**eta** = zeros(**count** + defect + 1, 1);

ksi = grand(**count** + defect + 1, 1, 'nor', 0, 1);

N = length(**alphas**) - 1;

M = length(**betas**);

for k = 1 : **count** + defect + 1,

**eta**(k) = 0;

for i = 0 : N,

if (k - i > 0) then

**eta**(k) = **eta**(k) + **alphas**(i+1) \* ksi(k - i);

end;

end;

for j = 1 : M,

if (k - j > 0) then

**eta**(k) = **eta**(k) + **betas**(j) \* **eta**(k - j);

end;

end;

end;

**eta** = **eta**(defect + 2 : **count** + defect + 1) + **meanx**;

endfunction;

m = 10; *// Analysis Depth*

epsilon = zeros(MAX\_AR\_LEVEL, MAX\_MA\_LEVEL);

best\_ar\_eps = %inf;

best\_ma\_eps = %inf;

best\_arma\_eps = %inf;

best\_ar\_alpha = [];

best\_ma\_alpha = [];

best\_arma\_alpha = [];

best\_ar\_beta = [];

best\_arma\_beta = [];

R = zeros(m+1, 1);

r = zeros(m+1, 1);

for k = 0 : m,

R(k+1) = correlation(k, x);

r(k+1) = R(k+1) / R(1);

end;

r\_model = zeros(m+1, 1);

for i = 0 : MAX\_AR\_LEVEL,

for j = 0 : MAX\_MA\_LEVEL,

alphas = alphas\_list(i \* (MAX\_MA\_LEVEL + 1) + j + 1);

betas = betas\_list(i \* (MAX\_MA\_LEVEL + 1) + j + 1);

for k = 0 : m,

r\_model(k+1) = norm\_theoretical\_corr(betas, R(1 : (i + j + 1)), k);

end;

epsilon(i+1, j+1) = quadratic\_error(r, r\_model);

if (i == 0) & (j~=3)&(j~=2)&(j~=1)&(epsilon(1, j+1) < best\_ma\_eps) then

best\_ma\_eps = epsilon(1, j+1);

best\_ma\_alpha = alphas;

elseif (j == 0) & (epsilon(i+1, 1) < best\_ar\_eps) then

best\_ar\_eps = epsilon(i+1, 1);

best\_ar\_alpha = alphas;

best\_ar\_beta = betas;

elseif (epsilon(i+1, j+1) < best\_arma\_eps) then

best\_arma\_eps = epsilon(i+1, j+1);

best\_arma\_alpha = alphas;

best\_arma\_beta = betas;

end;

end;

end;

printf("Epsilon:\n");

printMat(epsilon, '%16.7f');

printf("Best models:\nAR(" + INT\_FORMAT + "), MA(" + INT\_FORMAT + "), ARMA(" + INT\_FORMAT + "," + INT\_FORMAT + ").\n", length(best\_ar\_beta), length(best\_ma\_alpha) - 1, length(best\_arma\_beta), length(best\_arma\_alpha) - 1);