# Министерство образования и науки Российской Федерации

## Федеральное государственное бюджетное образовательное учреждение высшего образования

## «Новосибирский государственный технический университет»



## Кафедра теоретической и прикладной информатики

### Лабораторная работа № 3 по дисциплине «Методы активной идентификации динамических систем»

**ВЫЧИСЛЕНИЕ ПРОИЗВОДНЫХ ИНФОРМАЦИОННОЙ МАТРИЦЫ ФИШЕРА ПО КОМПОНЕНТАМ ВХОДНОГО СИГНАЛА**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Факультет: | ПМИ |  |
| Группа: | ПМИМ-01 |
| Студенты: | Ершов П.К., Дорош А.Э. |
| Вариант: | 3 |
| Уровень сложности: | 1 |
| Преподаватель: | Чубич В. М. |

Новосибирск

2021

1. **Цель работы**

Научиться вычислять производные ИМФ по компонентам входного сигнала.

1. **ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ**
2. В соответствии с выбранным уровнем сложности разработать программу вычисления производных ИМФ по компонентам входного сигнала. Проверить правильность работы программы на специально подготовленном тестовом примере.
3. Вычислить производные ИМФ по компонентам входного сигнала согласно варианту задания. Результаты привести в отчете.

1. **Необходимые теоретические данные**

**Алгоритм вычисления значения критерия идентификации:**

1. Определить F, Ψ , Γ, H , Q , R , , и

, где

F, Ψ , Γ, H , Q , R – матрицы модели, которые задаются в условиях задачи, – начальный вектор состояний, – ковариационная матрица ошибок прогнозирования.

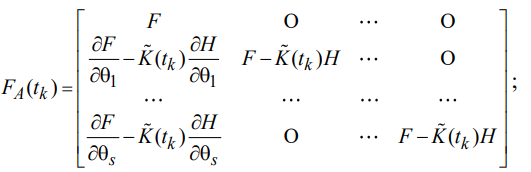
Сформировать матрицу по формуле (24) : .

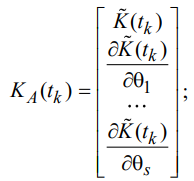
1. Положить

,

1. Определить .
2. Если k = 0, вычислить по формуле (21) где

(21)





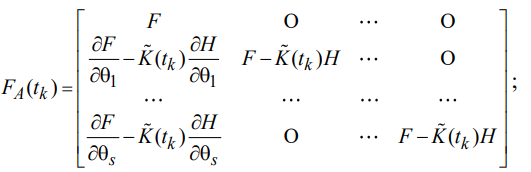
*;*

,

после перейти на шаг 8.

1. Найти по формуле (26)

(26)

1. Сформировать матрицу по формуле (23) 

(23)

1. Вычислить по формуле (21).
2. Найти , , , по формулам (10), (12), (13) и (15) соответственно.

(10)

(12)

(13)

(15).

1. Положить .
2. Если k = 0, вычислить при помощи равенств (32) и (33) и перейти на шаг 13.

(32)

(33)

1. Сформировать по формуле (33).
2. Вычислить по формуле (32).
3. Используя выражение (31), получить , отвечающее текущему значению и k .

(31)

где матрица

и имеют размерность n x n(s + 1), а вектор вычисляется по формуле (21).

1. Положить .
2. Увеличить на единицу. Если , перейти на шаг 10.
3. Увеличить k на единицу. Если , перейти на шаг 3, иначе закончить процесс.
4. **Полученные результаты**

Входные данные тестовой модели.

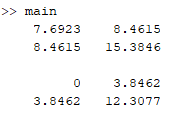
Исходя из указанной (для первого уровня сложности) модели:

*.*

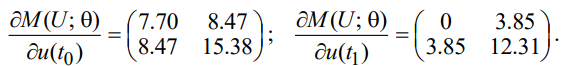
Можно получить следующие входные данные:

|  |  |  |  |
| --- | --- | --- | --- |
| Матрицы для моделей состояния и измерения | Ковариационные матрицы шумов и начальные условия |  |  |
| ,  ,  , | , R= 0.3,  , | *,* | , |

*Результаты работы алгоритма:*

**

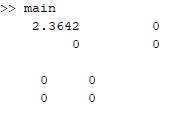
*Требуемое значение:*

**

Входные данные дня заданного варианта:

|  |  |  |  |
| --- | --- | --- | --- |
| Матрицы для моделей состояния и измерения | Ковариационные матрицы шумов и начальные условия |  |  |
| ,  ,  , | , R= 0.1,  , | *,*  *,* | , |

*Результаты работы алгоритма:*



1. **Текст программы**

Файл initi\_data.m

function [F, Psi, Gamma, H, Q, R, x0, P0, u] = init\_data(theta)

F = [-0.8, 1;

theta(1), 0];

Psi = [1; 1];

Gamma = [1; 1];

H = [1, 1];

Q = theta(2);

R = 0.1;

x0 = [0; 0];

P0 = [0.1, 0;

0, 0.1];

u = [3];

end

Файл init\_grad.m

function [F\_grad, Psi\_grad, Gamma\_grad, H\_grad, Q\_grad, R\_grad, x0\_grad, P0\_grad] = init\_grad()

F\_grad = cell(2,1);

F\_grad{1} = [0, 0; 1, 0];

F\_grad{2} = [0, 0; 0, 0];

Psi\_grad = cell(2,1);

Psi\_grad{1} = [0;0];

Psi\_grad{2} = [0;0];

Gamma\_grad = cell(2,1);

Gamma\_grad{1} = [0;0];

Gamma\_grad{2} = [0;0];

H\_grad = cell(2,1);

H\_grad{1} = [0,0];

H\_grad{2} = [0,0];

Q\_grad = cell(2,1);

Q\_grad{1} = [0];

Q\_grad{2} = [1];

R\_grad = cell(2,1);

R\_grad{1} = [0];

R\_grad{2} = [0];

x0\_grad = cell(2,1);

x0\_grad{1} = [0;0];

x0\_grad{2} = [0;0];

P0\_grad = cell(2,1);

P0\_grad{1} = [0, 0; 0, 0];

P0\_grad{2} = [0, 0; 0, 0];

end

Файл Get\_C.m

function [C] = Get\_C(n, alpha, i)

n2 = (alpha+1) \* n;

C = zeros(n, n2);

C(:, n\*i + 1 : n\*(i+1)) = eye(n);

end

Файл calculate\_Fisher\_Matrix.m

function [M\_grad] = calculate\_Fisher\_Matrix(theta, N)

[F, Psi, Gamma, H, Q, R, x0, P0, u] = init\_data(theta);

[F\_grad, Psi\_grad, Gamma\_grad, H\_grad, Q\_grad, R\_grad, x0\_grad] = init\_grad();

n\_1 = size(x0,1);

n\_2 = size(x0,2);

alpha = length(theta);

r = length(u);

% выделение памяти

M\_grad = cell(r, N); %zeros(alpha, alpha);

for i = 1: r

for j = 1: N

M\_grad{i, j} = zeros(alpha, alpha);

end

end

dM = cell(r, N); %zeros(alpha, alpha);

for i = 1: r

for j = 1: N

dM{i, j} = zeros(alpha, alpha);

end

end

dM\_buf = zeros(alpha, alpha);

x\_a = zeros(n\_1 \* (alpha + 1), r);

F\_a = zeros(n\_1 \* (alpha + 1), n\_1 \* (alpha + 1));

% сформировать матрицу пси(а) по формуле 24

Psi\_a = zeros(n\_1 \* (alpha + 1), n\_2);

Psi\_a(1:n\_1, :) = Psi;

for i = 1 : alpha

Psi\_a(n\_1\*i+1 : n\_1\*(i+1), :) = Psi\_grad{i};

end

Psi\_a\_grad = cell(r, 1); %zeros(n\_1 \* (alpha + 1), r);

for i = 1: r

Psi\_a\_grad{i} = zeros(n\_1 \* (alpha + 1), r);

end

x\_a\_all = cell(N, N);

x\_a\_grad = cell(r, 1);%zeros(n\_1 \* (alpha + 1), r);

for i = 1: r

x\_a\_grad{i} = zeros(n\_1 \* (alpha + 1), r);

end

for i = 1: N

for j = 1: N

x\_a\_all{i, j} = x\_a\_grad;

end

end

% задание начальных условий 2 шаг

P\_k = P0;

for k = 0 : N - 1

if (k == 0) % то 4 шаг, затем 8

x\_a(1 : n\_1\*1, :) = F\*x0 + Psi \* u; % 21 u0

for i = 1 : alpha

x\_a(n\_1\*i + 1 : n\_1\*(i+1), :) = F\_grad{i} \* x0 + F \* x0\_grad{i} + Psi\_grad{i} \* u; % u0

end

end

% 5-7 шаги

if (k ~= 0)

% 5 шаг

K\_ = F \* K;

% 6 шаг

F\_a(1:n\_1, 1:n\_1) = F;

for i = 1 : alpha

F\_a(n\_1\*i+1 : n\_1\*(i+1), 1:n\_1) = F\_grad{i} - K\_ \* H\_grad{i};

F\_a(n\_1\*i+1 : n\_1\*(i+1), n\_1\*i+1 : n\_1\*(i+1)) = F - K\_ \* H;

end

% 7 шаг

x\_a = F\_a \* x\_a + Psi\_a \* u(k);

end

% 8 шаг

P\_k1\_k = F \* P\_k \* F' + Gamma \* Q \* Gamma'; % 10 формула

B = H \* P\_k1\_k \* H' + R; % 12

K = P\_k1\_k \* H' / B; % 13

P\_k1 = (eye(n\_1) - K \* H) \* P\_k1\_k; % 15

for b = 0 : N - 1

if (k == 0)

if (b == k)

for i = 1 : r

u\_g = zeros(r, 1);

u\_g(i) = 1;

Psi\_a\_grad{i} = Psi\_a \* u\_g;

end

end

if (b ~= k)

for i = 1 : r

Psi\_a\_grad{i} = zeros(n\_1 \* (alpha + 1), r);

end

end

for i = 1 : r

x\_a\_grad{i} = Psi\_a\_grad{i};

end

end

if(k ~= 0)

if (b == k)

for i = 1 : r

u\_g = zeros(r, 1);

u\_g(i) = 1;

Psi\_a\_grad{i} = Psi\_a \* u\_g;

end

end

if (b ~= k)

for i = 1 : r

Psi\_a\_grad{i} = zeros(n\_1 \* (alpha + 1), r);

end

end

x\_a\_grad = x\_a\_all{k, b + 1};

for i = 1 : r

x\_a\_grad{i} = F\_a \* x\_a\_grad{i} + Psi\_a\_grad{i};

end

end

for t = 1: r

% шаг 9

dM\_buf = dM{t, b + 1};

for i = 1 : alpha

for j = 1 : alpha

dM\_buf(i,j) = trace(H\_grad{i} \* Get\_C(n\_1, alpha, 0) \* (x\_a\_grad{t} \* x\_a' + x\_a \* x\_a\_grad{t}') \* Get\_C(n\_1, alpha,0)' \* H\_grad{j}' / B)...

+ trace(H\_grad{i} \* Get\_C(n\_1, alpha, 0) \* (x\_a\_grad{t} \* x\_a' + x\_a \* x\_a\_grad{t}') \* Get\_C(n\_1, alpha,j)' \* H' / B) ...

+ trace(H \* Get\_C(n\_1, alpha, i) \* (x\_a\_grad{t} \* x\_a' + x\_a \* x\_a\_grad{t}') \* Get\_C(n\_1, alpha,0)' \* H\_grad{j}' / B) ...

+ trace(H \* Get\_C(n\_1, alpha, i) \* (x\_a\_grad{t} \* x\_a' + x\_a \* x\_a\_grad{t}') \* Get\_C(n\_1, alpha,j)' \* H' / B);

end

end

dM{t, b + 1} = dM\_buf;

M\_grad{t, b + 1} = M\_grad{t, b + 1} + dM{t, b + 1};

end

x\_a\_all{k + 1, b + 1} = x\_a\_grad;

end

P\_k = P\_k1;

end

end

Файл main.m

N = 2;

% истинная тета

theta\_ist = [-1.5; 0.5];

M = calculate\_Fisher\_Matrix(theta\_ist, N);

disp(M{1});

disp(M{2});