

Рис. 3 – Амплітудно-фазові характеристики за каналом $X_{Nimp} = 4 \rightarrow Y_{Nout} = 3$ у середовищах:
а – TURBO PASCAL; б – MatLab

Надійшла до редакції 05.12.2015

Kubrak A. I., Kovalyuk D. O., Zadvornyi B. V.

THE ALGORITHM OF FREQUENCY RESPONSES' CALCULATION WITH THE ORDINARY DIFFERENTIAL EQUATIONS' SYSTEM

The article offers a program realization by means of TURBO PASCAL and Matlab of frequency responses' calculation with the ordinary differential equations' system. The structure of programs and its components is described in detail.

Keywords: frequency response, control systems, TURBO PASCAL, MatLab.

References

1. Dorf, R. and Bishop, R. (2002), *Sovremennyye sistemy upravleniya* [Modern control systems], Laboratorija Bazovyh Znanij, Moscow, Russia.
2. Kvasko, M.Z., Kubrak, A.I. and Zhuchenko, A.I. (2003), *Chyslovi metody komp'yuternoho modelyuvannya avtomatychnykh system. Alhorytmy i prohramy* [Numerical methods for computer modeling of automated systems. Algorithms and software], Politekhnik, Kyiv, Ukraine.

УДК 681.5.03

КУБРАК А. І., к.т.н., проф.; КОВАЛЮК Д. О., к.т.н., доц.; СПОЛОВИЧ Р. Ю., магістрант
Національний технічний університет України «Київський політехнічний інститут»

ВИЗНАЧЕННЯ ПЕРЕДАТНИХ ФУНКЦІЙ ЗА СИСТЕМОЮ ЗВИЧАЙНИХ ДИФЕРЕНЦІАЛЬНИХ РІВНЯНЬ У КАНОНІЧНІЙ ФОРМІ

Вирішено задачу визначення передатних функцій елементів системи керування, що описується системою диференціальних рівнянь у канонічній формі. Наведено алгоритм розв'язку, спосіб опису вихідних даних у пам'яті ЕОМ і реалізація засобами MatLab. Здійснено тестування підходу на контрольному прикладі.

Ключові слова: передатна функція, диференціальні рівняння, MatLab.

© Кубрак А. І., Ковалюк Д. О., Сполович Р. Ю., 2016.

Постановка проблеми. Опис об'єктів відіграє важливу роль під час синтезу систем керування. У теорії автоматичного керування найбільш зручним способом опису об'єктів є їхнє подання у формі передатних функцій за заданими каналами [1]. Це дозволяє одержати їхні динамічні й частотні характеристики. Водно-

час, об'єкти керування описують диференціальними рівняннями. Тому постає задача одержання передатних функцій за системою диференціальних рівнянь.

Метою статті є розроблення алгоритму визначення передатних функцій елементів керування, що описується системою диференціальних рівнянь у канонічній формі.

Алгоритм одержання передатних функцій. Маємо систему диференціальних рівнянь у канонічній формі $dy_z/dt = \sum_{s=1}^n A_{zs}y_s + \sum_{s=1}^m B_{zs}x_s$, де y_s – вихідні величини (параметри стану); $1 \leq s \leq n$; x_s – вхідні сигнали; $1 \leq s \leq m$; n – порядок системи; m – кількість входів; A_{zs}, B_{zs} – коефіцієнти.

Виконаємо над цим рівнянням перетворення Лапласа за нульових початкових умов:

$$\sum_{s=1}^n C_{zs}(p)\bar{y}_s = \sum_{s=1}^m B_{zs}\bar{x}_s, \quad 1 \leq z \leq n, \quad \text{де } C_{zs} = \begin{cases} p - A_{zs}, & \text{якщо } s = z; \\ -A_{zs}, & \text{якщо } s \neq z; \end{cases} \quad \bar{x}_s = \int_0^{\infty} x_s e^{-pt} dt, \quad \bar{y}_s = \int_0^{\infty} y_s e^{-pt} dt - \text{зображення}$$

за Лапласом від x_s і y_s .

Якщо з усіх можливих входів x_s ($1 \leq s \leq m$) надати ненульовий приріст лише входу x_{Nimp} , на який поділити систему, то $\sum_{s=1}^n C_{z,s}(p)W_{Nimp,s}(p) = B_{z,Nimp}$, де $W_{Nimp,s}(p)$ – передатна функція каналу $x_{Nimp} \rightarrow y_s$.

Розширена матриця Rm коефіцієнтів цієї системи матиме таку структуру:

	1	2	3	...	n	$n + 1$
1	$-A_{11} + p$	$-A_{12}$	$-A_{13}$...	A_{1n}	B_{1Nimp}
2	$-A_{21}$	$-A_{22} + p$	$-A_{23}$...	A_{2n}	B_{2Nimp}
3	$-A_{31}$	$-A_{32}$	$-A_{33} + p$...	A_{3n}	B_{3Nimp}
...
n	$-A_{n1}$	$-A_{n2}$	$-A_{n3}$...	$-A_{nn} + p$	B_{nNimp}
	$W_{Nimp1}(p)$	$W_{Nimp2}(p)$	$W_{Nimp3}(p)$...	$W_{Nimpn}(p)$	

Під кожним із перших n стовпців матриці показано передатну функцію (невідому), коефіцієнтами при якій є елементи відповідного стовпця. Елементи матриці – це масив коефіцієнтів системи, що розв'язується.

Зведемо матрицю Rm до трикутного вигляду (прямий хід схеми Гаусса). Обнулення елемента $Rm[z, i]$ виконуємо, віднімаючи від елементів рядка z , помножених на $Rm[i, i]$, елементи рядка i , помножені на $Rm[z, i]$. Результати поелементного віднімання заносимо в рядок z .

Після завершення прямого ходу останній рядок n матриці Rm відповідатиме рівнянню

$$Rm[n, n]W_{Nimp,n}(p) = Rm[n, n + 1], \quad \text{звідки } W_{Nimp,n}(p) = \frac{Rm[n, n + 1]}{Rm[n, n]}.$$

Оскільки всі передатні функції об'єкта із зосередженими параметрами є дробово-раціональними, їх можна подати зі спільним знаменником $Rm[n, n]$. Отже, під час виконання зворотного ходу схеми Гаусса достатньо визначити лише чисельники передатних функцій $W_{Nimp,z}$, де z змінюється від $(n - 1)$ до 1 (чисельник для $W_{Nimp,n}$ уже фігурує в комірці $Rm[n, n + 1]$). Решту чисельників будемо заносити в комірки Rm за схемою «чисельник $W_{Nimp,z} \rightarrow$ комірка $Rm[z, n + 1]$ ».

Незначним модифікуванням матриці Rm , зокрема внесенням у стовпчик $(n + s)$ значень B_{zs} ($1 \leq s \leq m$) і обслуговуванням стовпчиків до $(n + m)$ включно, можна забезпечити одержання чисельників передатних функцій $W_{s,z}(p)$, $1 \leq s \leq m$, $1 \leq z \leq n$, тобто визначення всіх можливих у заданому об'єкті передатних функцій. Їх можна записати в комірки масиву Rm за схемою «чисельник $W_{s,z}(p) \rightarrow$ комірка $Rm[z, n + s]$ ».

Програмна реалізація. Робота програми складатиметься з таких кроків: формування матриці; прямий і зворотній хід, що визначають чисельники й знаменники передатних функцій; виведення результатів для потрібного каналу. Описаний алгоритм реалізовано в середовищі *MatLab* у такий спосіб:

```
function W=WpSU(A,B,x,y)
n = size(A, 2);
m = size(B, 2);
dimention_cell = 100;
for z = 1 : n
```

```
for s = 1 : n
    if s==z
        Rm(z,s,dimension_cell) = 2;
        Rm(z,s,1) = -1;
        Rm(z,s,2) = A(z,z);
    else
        Rm(z,s,dimension_cell) = 2;
        Rm(z,s,1) = 0;
        Rm(z,s,2) = A(z,s);
    end;
end;
for s = 1 : m
    Rm(z,n+s,dimension_cell) = 2;
    Rm(z,n+s,1) = 0;
    Rm(z,n+s,2) = -B(z,s);
end;
end;
for i = 1 : n - 1
    C = getCoef(Rm, i, i);
    for z = i + 1 : n
        if (zeroCoef(Rm, z, i) == 0)
            D = getCoef(Rm, z, i);
            for s = i+1 : n+m
                row_polinoms_1 = getCoef(Rm, z, s);
                row_polinoms_2 = getCoef(Rm, i, s);
                E = conv(row_polinoms_1, C);
                F = conv(row_polinoms_2, D);
                Coef_subtraction = E - F;
                Rm = setCoef(Rm, Coef_subtraction, z, s);
            end;
        end;
    end;
end;
for z = n-1 : -1 : 1
    for s = 1 : m
        SUM = [];
        SUM(1) = 0;
        for i = z + 1 : n
            P1 = getCoef(Rm, z, i);
            P2 = getCoef(Rm, i, n+s);
            D = conv(P1, P2);
            SUM = polinomSum(SUM, D);
        end;
        P1 = getCoef(Rm, z, n+s);
        P2 = getCoef(Rm, n, n);
        D = conv(P1, P2);
        C = polinomSubtract(D, SUM);
        P1 = getCoef(Rm, z, z);
        [W, r] = deconv(C, P1);
        Rm = setCoef(Rm, W, z, n+s);
    end;
end;
```

```
num=formpol(Rm(y,n+x,:));  
den=formpol(Rm(n,n,:));  
W=tf(num,den);  
end
```

Обчислювана функція має такі аргументи: A – масив коефіцієнтів при змінних y ; B – масив коефіцієнтів змінних x ; x – номер входу; y – номер виходу. Функція повертає передатну функцію, попередньо сформованою вбудованою функцією $tf(\dots)$.

Розроблена програма використовує такі функції:

– *formpol(A)*, що формує одновимірний масив коефіцієнтів полінома з масиву матриці Rm . Вона отримує комірку масиву, де розміщено поліном, і повертає його:

```
function [A]=formpol(A)  
A=reshape(A,1,[]);  
A=A(1:A(100));  
end
```

– *getCoef(A, m, n)*, що повертає вектор коефіцієнтів полінома:

```
function [Coef] = getCoef(A, m, n)  
dimention_cell = 100;  
for i = 1 : A(m, n, dimention_cell)  
Coef(i) = A(m, n, i);  
end;  
end
```

– *polinomSubtract(p1, p2)*, що віднімає поліноми:

```
function [subtract] = polinomSubtract(p1, p2)  
subtract = [zeros(1, size(p2,2) - size(p1,2)) p1] –  
[zeros(1, size(p1,2) - size(p2,2)) p2];  
end
```

– *polinomSum(p1, p2)*, що додає поліноми:

```
function [sum] = polinomSum(p1, p2)  
sum = [zeros(1, size(p1,2) - size(p2,2)) p2] +  
[zeros(1, size(p2,2) - size(p1,2)) p1];  
end
```

– *setCoef(A, Coef, m, n)*, що зберігає вектор коефіцієнтів полінома:

```
function [A] = setCoef(A, Coef, m, n)  
dimention_cell = 100;  
A(m, n, dimention_cell) = length(Coef);  
for i = 1 : length(Coef)  
A(m, n, i) = Coef(i);  
end;  
end
```

– *zeroCoef(A, m, n)*, що перевіряє вектор коефіцієнтів полінома:

```
function [is_zero] = zeroCoef(A, m, n)  
is_zero = 1;  
dimention_cell = 100;  
for i = 1 : A(m, n, dimention_cell)  
if A(m, n, i) ~= 0  
is_zero = 0;  
end  
end;  
end
```

Контрольний приклад. Маємо замкнуту систему (рис. 1), що описується системою рівнянь

$$\begin{cases} T_1 y_1' + y_1 = x_1 + y_4 \\ T_2 y_2' + y_2 = x_2 + y_1 \\ T_3 y_3' + y_3 = x_3 + y_2 \\ y_4' = k(x_4 - y_3) \end{cases}$$

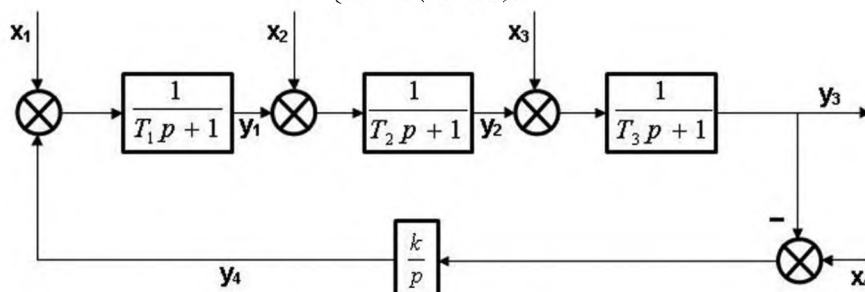


Рис. 1 – Структурна схема

Подамо її у формі

$$\begin{cases} y_1' = \frac{-1}{T_1} y_1 + 0 \cdot y_2 + 0 \cdot y_3 + \frac{1}{T_1} y_1 + \frac{1}{T_1} x_1 + 0 \cdot x_2 + 0 \cdot x_3 + 0 \cdot x_4, \\ y_2' = \frac{1}{T_2} y_1 - \frac{1}{T_2} y_2 + 0 \cdot y_3 + 0 \cdot y_4 + 0 \cdot x_1 + \frac{1}{T_2} x_2 + 0 \cdot x_3 + 0 \cdot x_3 + 0 \cdot x_4, \\ y_3' = 0 \cdot y_1 + \frac{1}{T_3} y_2 - \frac{1}{T_3} y_3 + 0 \cdot y_4 + 0 \cdot x_1 + 0 \cdot x_2 + \frac{1}{T_3} x_3 + 0 \cdot x_4, \\ y_4' = 0 \cdot y_1 + 0 \cdot y_2 - k \cdot y_3 + 0 \cdot y_4 + 0 \cdot x_1 + 0 \cdot x_2 + 0 \cdot x_3 + 0 \cdot x_4. \end{cases}$$

>> A=[-1 0 0 1; 0.5 -0.5 0 0; 0 0.25 -0.25 0; 0 0 -0.2 0]

A =

```
-1.0000    0    0    1.0000
 0.5000  -0.5000    0    0
 0    0.2500  -0.2500    0
 0    0    -0.2000    0
```

>> B=[1 0 0 0; 0 0.5 0 0; 0 0 0.25 0; 0 0 0 0.2]

B =

```
1.0000    0    0    0
 0    0.5000    0    0
 0    0    0.2500    0
 0    0    0    0.2000
```

>> WpSU(A,B,1,2)

ans =

```
0.5 s^2 + 0.125 s
-----
s^4 + 1.75 s^3 + 0.875 s^2 + 0.125 s + 0.025
```

Continuous-time transfer function.

Рис. 2 – Результат виконання алгоритму в середовищі Matlab

Передатна функція, розрахована в середовищі MatLab, відповідає розрахованій аналітично.

Із цієї системи сформуємо масиви коефіцієнтів A і B із змінними y_i й x_i :

$$A = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 0,5 & -0,5 & 0 & 0 \\ 0 & 0,2 & -0,2 & 0 \\ 0 & 0 & -0,2 & 0 \end{bmatrix};$$

$$B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0,5 & 0 & 0 \\ 0 & 0 & 0,2 & 0 \\ 0 & 0 & 0 & 0,2 \end{bmatrix}.$$

Ініціалізуємо функцію WpSU (рис. 2).

Передатна функція, розрахована аналітичним шляхом:

$$\begin{aligned} W_{12}(p) &= \frac{1}{T_1 p + 1} \cdot \frac{1}{T_2 p + 1} = \\ &= \frac{1}{T_1 p + 1} \cdot \frac{1}{T_2 p + 1} \cdot \frac{1}{T_3 p + 1} \cdot \frac{k}{p} + 1 = \\ &= \frac{4p^2 + 1}{8p^4 + 14p^3 + 2p^2 + 1p + 0,2} = \\ &= \frac{0,5p^2 + 0,125p}{p^4 + \frac{14}{8}p^3 + \frac{7}{8}p^2 + \frac{1}{8}p + \frac{0,2}{8}} \end{aligned}$$

Висновки. Написаний програмний код алгоритму в поєднанні з вбудованими можливостями пакету MatLab, дозволяє отримати передатні функції з системи диференціальних рівнянь, що описують систему за будь-яким каналом.

У подальшому планується розробити метод одержання передатних функцій для об'єктів із розподіленими параметрами, а також таких, що описуються диференціальними рівняннями вищих порядків.

Список використаної літератури

1. Дорф Р. Современные системы управления / Р. Дорф, Р. Бишоп. – М. : Лабор. базовых знаний, 2002. – 832 с.

Надійшла до редакції 10.11.2015

Kurbak, A. I., Kovalyuk D. O., Spalevic G. Yu.

DETERMINATION OF THE TRANSFER FUNCTIONS FOR THE SYSTEM OF ORDINARY DIFFERENTIAL EQUATIONS IN CANONICAL FORM

The article considers the problem of determining transfer functions of control system components, described by a system of differential equations in canonical form. The algorithm of solutions, method of presentation of output data in computer memory and software implementation by Matlab software package. Testing this approach on a test sample, which confirms the correctness and accuracy of calculations.

Keywords: transfer function, differential equation, MatLab.

References

1. Dorf, R. and Bishop, R. (2002), *Sovremennyye sistemy upravleniya* [Modern control systems], Laboratorija Bazovyh Znanij, Moscow, Russia.

УДК 004.94

ОНИЩЕНКО В. О., асп.; МИЛЕНЬКИЙ В. В., к.т.н., доц.

Національний технічний університет України «Київський політехнічний інститут»

МОДЕЛЮВАННЯ АЛГОРИТМУ ПЕЙДЖА Й КУМУЛЯТИВНИХ СУМ ДЛЯ ВИЯВЛЕННЯ ДЕФЕКТІВ У КАНАЛАХ ЗВ'ЯЗКУ З ОБ'ЄКТОМ

За результатами досліджень послідовних алгоритмів виявлено зміни математичного очікування й дисперсії сформованого алгоритму, що дозволяє виявити тип дефекту. Визначено середню тривалість знаходження відхилень й відновлення дефектів. Установлено шляхи визначення ймовірності помилкового виявлення дефекту.

Ключові слова: дефект, відновлення дефектів, алгоритм Пейджса.

© Оніщенко В. О., Миленький В. В., 2016.

Постановка проблеми. Множина задач, в яких виникає потреба виявити зміну властивостей випадкових процесів, є достатньо великою. Найважливішими з них є завдання контролю й діагностики в автоматизованих системах управління технологічними процесами. Окрім виявлення розладнання випадкового процесу, існує також завдання локалізації дефекту, вирішення якої має вказати місце (тип) дефекту [1].

Метою цієї статті є побудова алгоритму імітаційного моделювання для виявлення дефектів, визначення середньої тривалості виявлення відхилень, пошуку відновлювальних дефектів і тривалості відновлення, визначення ймовірності помилкового виявлення дефекту.

Викладення основного матеріалу. Алгоритм Пейджса має вигляд:

$$G(n) = \begin{cases} S_n - \min_{1 < k < n} S_k, \mu_1 < \mu_2, S_n = \sum_{i=1}^n (z(i) - \mu_1 - v/2) \\ \max_{1 < k < n} S_k - S_n, \mu_1 > \mu_2, S_n = \sum_{i=1}^n (z(i) - \mu_1 + v/2) \end{cases},$$

де $\mu_2 = \bar{z}$, μ_1 – математичні сподівання вихідного сигналу z до появи дефекту; $v > 0$ – поріг чутливості.