

МИНОБРНАУКИ РОССИИ

---

Санкт-Петербургский государственный электротехнический  
университет «ЛЭТИ» им. В. И. Ульянова (Ленина)

---

## **Моделирование электромеханических систем средствами MATLAB**

Методические указания  
к курсовому расчету по дисциплине:  
«Моделирование систем управления»

Санкт-Петербург  
Издательство СПбГЭТУ «ЛЭТИ»  
2017

Моделирование электромеханических систем средствами MATLAB. Методические указания к курсовому расчету по дисциплине «Моделирование систем управления» / Сост.: О. Ю. Лукомская, А. Л. Стариченков, А. Г. Шпекторов,. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2017. 40 с.

Содержат указания по базовым процедурам исследования и моделирования линейных и нелинейных динамических систем на примере электрических машин постоянного тока. Приведены теоретические основы анализа динамических систем, а также практические рекомендации по программированию и моделированию электрических машин в среде MATLAB v.6.x.

Предназначены для подготовки студентов по направлению 27.03.04 - «Управление в технических системах», обучающихся по дисциплине «Моделирование систем управления».

Утверждено  
редакционно-издательским советом университета  
в качестве методических указаний

## ВВЕДЕНИЕ

Рассмотрим математическую модель электрической машины на примере генератора постоянного тока смешанного возбуждения (эта модель будет использована в дальнейшем для иллюстрации излагаемого материала). Динамику объекта описывает следующая система уравнений:

$$\begin{aligned} F &= i_B w + i_T w_c; \\ \Phi &= \Lambda(F); \\ U_B &= i_B r_B + w \frac{d\Phi}{dt}; \\ c_e \Phi \omega &= i_T (r_{\text{я}} + R_0) + L_{\text{я}} \frac{di_T}{dt} + w_c \frac{d\Phi}{dt}; \\ J \frac{d\omega}{dt} &= M_B - c_M \Phi i_T, \end{aligned} \quad (1)$$

где  $w$  – число витков обмотки независимого возбуждения;  $w_c$  – число витков обмотки последовательного возбуждения;  $r_B$  – сопротивление обмотки независимого возбуждения;  $\Phi$  – магнитный поток;  $F$  – магнитодвижущая сила (МДС);  $r_{\text{я}}$  – сопротивление якоря, щеток и т.д.;  $L_{\text{я}}$  – индуктивность якоря;  $R_0$  – сопротивление нагрузки;  $M_B$  – внешний вращающий момент;  $c_e$  – конструктивный коэффициент;  $c_M$  – конструктивный коэффициент;  $J$  – момент инерции якоря;  $\omega$  – угловая скорость вращения якоря;  $\Lambda(F)$  – кривая намагничивания электрической машины;  $i_T$  – ток якоря. Размерность всех величин задана в системе СИ.

Численные значения постоянных величин следующие:  $w=1500$ ;  $w_c=5$ ;  $r_B=97,2$  Ом;  $r_{\text{я}}=0,054$  Ом;  $L_{\text{я}}=0,03$  Гн;  $R_0=1,246$  Ом;  $c_e=130$ ;  $c_M=120$ ;  $J=1,2$  кг·м<sup>2</sup>

Номинальные значения переменных, описывающих динамику объекта, следующие:  $\Phi_H=0,01$  Вб;  $M_{BH}=300$  нм;  $\omega_H=100$  с<sup>-1</sup>;  $i_{TH}=100$  А;  $u_{BH}=220$  В.

Кривая намагничивания  $\Phi(F)$  приведена в таблице 1:

Таблица 1

$F, (Aw) \cdot 10^3$	0,0	0,5	1,0	1,5	2,0	2,5	3,0	3,5	4,0	4,5	7,0
$\Phi, \text{Вб} \cdot 10^{-3}$	0,0	0,33	0,66	0,93	1,18	1,38	1,5	1,6	1,67	1,79	1,8

В исходных уравнениях объекта (1) можно выделить следующие сигналы:  $F, \Phi, i_B, i_T, u_B, \omega, M_B, R_0$ , представляющие собой некоторые функции

времени. Остальные величины – конструктивные постоянные. Переменные  $u_B, M_B, R_0$  могут быть изменены произвольно, независимо от остальных (но остальные переменные зависят от их значений); они определяют внешнее воздействие на объект и их называют входными переменными. Переменные  $\Phi, i_T, \omega$  содержат в уравнениях объекта свои производные по времени, они называются переменными состояния. Переменные  $F, i_B$  являются внутренними, они могут быть выражены через предыдущие шесть.

Математическая модель (1) включает в себя три дифференциальных и два алгебраических уравнения. В модели присутствуют нелинейности: кривая намагничивания, заданная таблично в промежуточных значениях, а также взаимные произведения переменных состояния между собой или с входными переменными. Таким образом, модель относится к классу систем нелинейных дифференциальных уравнений (СНДУ), а описываемый ею объект является нелинейной динамической системой.

В процессе моделирования электрической машины постоянного тока возникают следующие задачи:

- 1) аппроксимация обратной кривой намагничивания;
- 2) исследование переходных процессов в системе;
- 3) расчет статических характеристик электрической машины;
- 4) исследование линеаризованной математической модели;
- 5) расчет передаточных функций и частотных характеристик линейной системы.

Пути решения данных задач с применением среды MATLAB изложены далее в методическом описании проектирования курсовой работы.

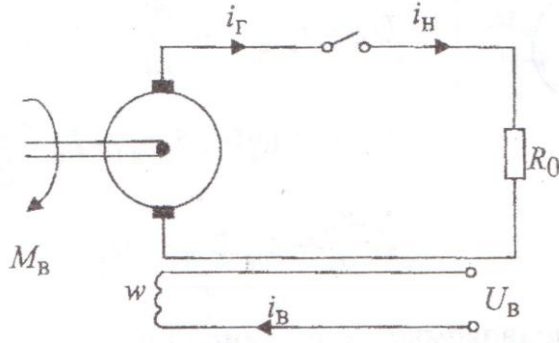
## ВАРИАНТЫ ЗАДАНИЙ

Ниже в таблицах и на рисунках приведены исходные данные по электрическим машинам (ЭМ) для 7 вариантов.

Таблица 2

№ варианта	Тип ЭМ
1	ГПТ НВ, работающий на активную нагрузку
2	ГПТ НВ, работающий на сеть большой мощности
3	ГПТ ПрВ, работающий на сеть большой мощности
4	ГПТ СВ, работающий на активную нагрузку
5	ДПТ НВ, управляемый со стороны коллектора
6	ДПТ НВ, управляемый со стороны ОВ
7	ДПТ ПВ

1



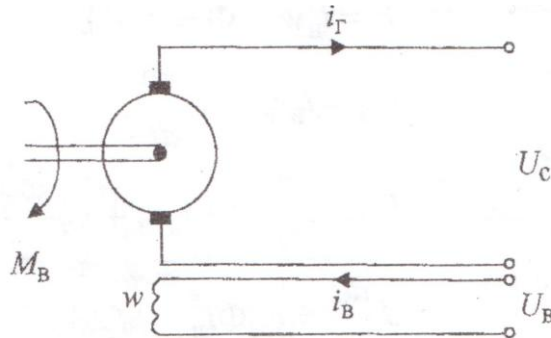
$$F = i_B w, \quad \Phi = \Lambda(F),$$

$$U_B = i_B r_B + w \frac{d\Phi}{dt},$$

$$c_e \omega \Phi = i_\Gamma r_\Sigma + L_\Sigma \frac{di_\Gamma}{dt} + i_\Gamma R_0,$$

$$J \frac{d\omega}{dt} = M_B - c_M \Phi i_\Gamma.$$

2



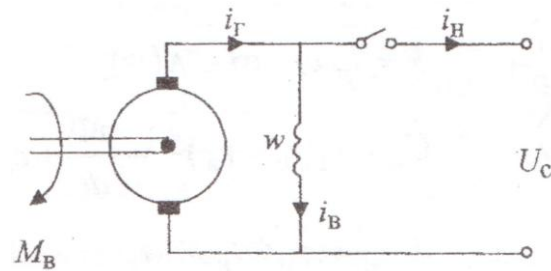
$$F = i_B w, \quad \Phi = \Lambda(F),$$

$$U_B = i_B r_B + w \frac{d\Phi}{dt},$$

$$c_e \omega \Phi = i_\Gamma r_\Sigma + L_\Sigma \frac{di_\Gamma}{dt} + U_C,$$

$$J \frac{d\omega}{dt} = M_B - c_M \Phi i_\Gamma.$$

3



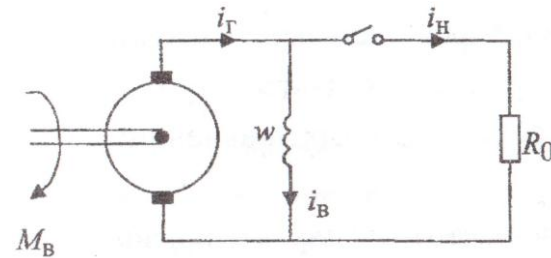
$$F = i_B w, \quad \Phi = \Lambda(F),$$

$$U_C = i_B r_B + w \frac{d\Phi}{dt},$$

$$c_e \omega \Phi = i_\Gamma r_\Sigma + L_\Sigma \frac{di_\Gamma}{dt} + U_C,$$

$$J \frac{d\omega}{dt} = M_B - c_M \Phi i_\Gamma.$$

4



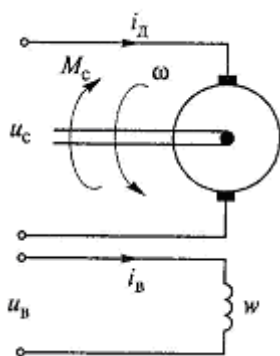
$$F = i_B w, \quad \Phi = \Phi_0 + \Lambda(F),$$

$$i_H R_0 = i_B r_B + w \frac{d\Phi}{dt},$$

$$c_e \omega \Phi = i_\Gamma r_\Sigma + L_\Sigma \frac{di_\Gamma}{dt} + i_H R_0,$$

$$J \frac{d\omega}{dt} = M_B - c_M \Phi i_\Gamma.$$

5



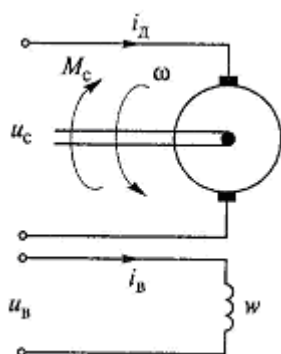
$$F = i_B w, \quad \Phi = \Lambda(F),$$

$$U_B = i_B r_B + w \frac{d\Phi}{dt},$$

$$U_c = c_e \omega \Phi + i_d r_{\text{я}} + L_{\text{я}} \frac{di_d}{dt},$$

$$J \frac{d\omega}{dt} = c_M \Phi i_d - M_c.$$

6



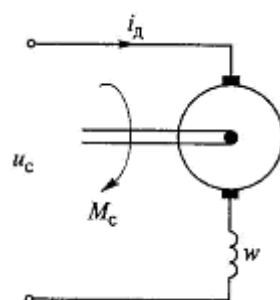
$$F = i_B w, \quad \Phi = \Lambda(F),$$

$$U_B = i_B r_B + w \frac{d\Phi}{dt},$$

$$U_c = c_e \omega \Phi + i_d r_{\text{я}} + L_{\text{я}} \frac{di_d}{dt},$$

$$J \frac{d\omega}{dt} = c_M \Phi i_d - M_c.$$

7



$$F = i_d w, \quad \Phi = \Lambda(F),$$

$$U_c = i_d (r_B + r_{\text{я}}) + w \frac{d\Phi}{dt} + c_e \omega \Phi,$$

$$J \frac{d\omega}{dt} = c_M \Phi i_d - M_c.$$

Таблица 3

Параметры объектов моделирования

Номер варианта	$r_B$ , Ом	$r_{\text{я}}$ , Ом	$w$ , вит	$L_{\text{я}}$ , Гн	$R_0$ , Ом	$c_e$	$c_M$	$\Phi_0$ , % $\Phi_H$	$J$ , кг·м <sup>2</sup>
1	145	0.3	4000	0.01	4.0	205	200	—	0.35
2	145	0.3	4000	0.01	—	205	200	—	0.35
3	415	1.39	8600	0.02	—	300	300	—	0.12
4	97.2	0.05	3000	0.03	4.0	130	120	10	1.2
5	145	0.3	4000	0.01	—	205	200	—	0.35
6	97.2	0.05	3000	0.03	—	130	120	—	1.2
7	0.14	0.3	60	—	—	290	230	—	0.1

Таблица 4

## Входные, выходные и нормировочные переменные

Номер варианта	$\Phi_H$ , Вб	$\omega_H$ , с <sup>-1</sup>	$i_H$ , А	$M_{BH}$ , Н·м	$M_{CH}$ , Н·м	$U_{BH}$ , В	$U_{CH}$ , В	Вход
1	0.007	100	50	70	–	220	–	$U_\theta, M_\theta, R_0$
2	0.007	100	50	70	–	220	220	$U_\theta, M_\theta, U_c$
3	0.005	100	30	50	–	–	220	$U_c, M_\theta$
4	0.01	100	100	300	–	–	–	$M_\theta, R_0$
5	0.007	100	50	–	50	220	220	$M_c, U_c$
6	0.01	100	100	–	300	220	220	$U_\theta, M_c$
7	0.01	100	50	–	470	–	220	$M_c, U_c$

Таблица 5

## Кривые намагничивания

Номер варианта	Параметр	Значения									
		1	2	3	4	5	6	7	8	9	10
1, 2	$F, A \cdot w$	0	0.2	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8
	$\Phi/\Phi_H$	0	0.3	0.52	0.67	0.78	0.86	0.92	0.96	1.01	1.02
3	$F, A \cdot w$	0	0.5	1.0	1.5	2.0	2.5	3.0	3.5	–	–
	$\Phi/\Phi_H$	0	0.4	0.8	0.98	1.08	1.17	1.19	1.21	–	–
4, 6	$F, A \cdot w$	0	1.0	1.5	2.0	2.5	3.0	3.5	4.0	5.0	–
	$\Phi/\Phi_H$	0	0.66	0.93	1.18	1.38	1.5	1.6	1.67	1.79	–
5	$F, A \cdot w$	0	0.2	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8
	$\Phi/\Phi_H$	0	0.35	0.57	0.72	0.83	0.91	0.97	1.01	1.06	1.07
7	$I, A$	0	20	30	40	60	80	100	120	140	–
	$\Phi/\Phi_H$	0	0.58	0.8	0.91	1.07	1.21	1.35	1.47	1.58	–

В таблице 5 значение  $F$  поделено на 4000 для вариантов 1, 2, 3, 5 и на 1000 для вариантов 4, 6.

## СПИСОК ОБОЗНАЧЕНИЙ

ГПТ – генератор постоянного тока;

ДПТ – двигатель постоянного тока;

НВ – независимое возбуждение;

ПВ – последовательное возбуждение;

ПрВ – параллельное возбуждение;

СВ – самовозбуждение;  
ОВ – обмотка возбуждения;  
МНК – метод наименьших квадратов;  
СНЛАУ – система нелинейных алгебраических уравнений;  
СНДУ – система нелинейных дифференциальных уравнений;  
СЛАУ – система линейных алгебраических уравнений;  
СЛДУ – система линейных дифференциальных уравнений;  
ПФ — передаточная функция;  
АЧХ, ФЧХ — амплитудно- и фазо-частотные характеристики;  
ППП – пакеты прикладных программ.

## **I. АППРОКСИМАЦИЯ ОБРАТНОЙ КРИВОЙ НАМАГНИЧИВАНИЯ ЭЛЕКТРИЧЕСКОЙ МАШИНЫ НА ОСНОВЕ МЕТОДА НАИМЕНЬШИХ КВАДРАТОВ**

**Цель:** аппроксимировать нелинейную зависимость  $F(\Phi)$  заданную таблично, в промежуточных точках; аппроксимирующую функцию найти в виде полинома заданной степени; оценить зависимость точности аппроксимации от степени полинома.

### **Постановка задачи**

В дальнейших исследованиях понадобится функция  $\Phi^{-1}(F)$ , т.е. функции  $F(\Phi)$ , заданной таблично для некоторого числа точек  $N$ . При отыскании статических режимов, при численном интегрировании СНЛДУ, а также при линеаризации необходимо иметь значения  $F$  при любом значении  $\Phi$  из диапазона  $[0, \Phi_{\max}]$ , а не только в конечном числе точек. Поэтому нужно найти некоторую функцию  $p(\Phi)$ , определенную на всем отрезке  $[0, \Phi_{\max}]$ , обладающую свойством  $p(\Phi_i) \approx F(\Phi_i)$  в каждой  $i$ -й точке таблицы и аппроксимирующую таблицу в промежуточных точках. В данной работе использована функция  $p(\Phi)$  в виде полинома заданной степени  $n$ :

$$p(x) = c_0 + c_1x + \dots + c_nx^n,$$

где  $x$  - переменная.

Значение  $n$  выбирают из условия обеспечения требуемой точности. Чем больше  $n$ , тем выше точность аппроксимации. При заданном значении  $n$  полином  $p$  определяют его коэффициенты  $c_0, \dots, c_n$ . Метод наименьших квадра-



тов (МНК) позволяет аналитически определить коэффициенты полинома, когда критерием точности является функционал вида:

$$I(p) = I(c_0, \dots, c_n) = \sum_{i=1}^N (F_i - p(\Phi_i))^2, \quad (2)$$

где  $N$  - количество точек в таблице. В выражении (2) разность

$$\varepsilon_i = F_i - p(\Phi_i)$$

называется невязкой ( $i$ -ой точки), поэтому значение функционала  $I$  есть сумма квадратов значений невязок по всем  $N$  точкам. Метод наименьших квадратов дает формулу для определения коэффициентов полинома, при которых значение функционала  $I$  будет наименьшим.

### Содержание

Рассмотреть таблицу  $F(\Phi)$ , которую следует переписать в виде  $\bar{F}(\bar{\Phi})$ , где  $\bar{\Phi}$  – нормированный магнитный поток  $\bar{\Phi} = \Phi / \Phi_H$ ,  $\bar{F}$  – нормированная МДС  $\bar{F} = F / F_H$ . Такое преобразование необходимо с целью обеспечения хорошей обусловленности матриц в формуле МНК. Аппроксимации подлежит таблица  $\bar{F}(\bar{\Phi})$ . Ее аппроксимирующий полином обозначить через  $\bar{p}$ . После нахождения этого полинома коэффициенты  $p$  легко определить путем перехода от  $\bar{\Phi}$  к  $\Phi$  и от  $\bar{F}$  к  $F$ .

1. Написать и отладить программу расчета коэффициентов полинома  $\bar{p}$ , т.е.  $\bar{c}_0, \dots, \bar{c}_n$  затем рассчитать коэффициенты полинома  $p$ .

Предварительно нужно выбрать значение степени полинома  $n$ . Следует учитывать, что график  $\bar{F}(\bar{\Phi})$  симметричен относительно начала координат ( $\bar{F}(\bar{\Phi})$  – нечетная функция), следовательно, полином  $\bar{p}$  будет иметь нулевые коэффициенты при четных степенях переменной, а также  $\bar{c}_0 = 0$ . Поэтому рассчитывают только коэффициенты  $\bar{c}_1, \bar{c}_3, \dots, \bar{c}_n$ , где  $n$  – нечетное.

Алгоритм расчета коэффициентов полинома  $\bar{p}$ :

1) построить матрицу вида  $G = [G_1, G_3, \dots, G_n]$ , в которой каждый столбец  $G_i$  имеет вид:

$$G_i = \begin{bmatrix} \bar{\Phi}_1^i \\ \vdots \\ \bar{\Phi}_N^i \end{bmatrix},$$

где показатель степени  $i = 1, 3, 5, \dots, n$ ;

2) рассчитать вектор  $C = [G^T G]^{-1} [G^T \bar{F}]$ , где  $\bar{F}$  – вектор-столбец из таблицы  $\bar{F}(\bar{\Phi})$ . Компоненты вектора  $C$  есть коэффициенты полинома  $\bar{p}$  при нечетных степенях, причем последний элемент вектора  $C$  есть коэффициент при старшей степени. В языке MATLAB полином представлен вектором, где первый элемент – это коэффициент при старшей степени. Поэтому при  $n=3$  полином  $\bar{p}$  формируется в программе так:  $p=[c(2), 0, c(1), 0]$ ;

3) сформировать полином  $p$ .

4) построить графики  $F(\Phi)$  и  $p(\Phi)$  и оценить качество аппроксимации.

### Рекомендации по программированию

Понадобятся две функции языка MATLAB:

1)  $x = \text{inv}(y)$  – обращает квадратную невырожденную матрицу  $y$ , результат помещает в  $x$ ;

2)  $z = \text{polyval}(p, x)$  – вычисляет значение полинома  $p$  (заданного как вектор коэффициентов) на каждом элементе вектора  $x$ , результат помещает в соответствующий элемент вектора  $y$ , т.е. реализует расчет значений функции  $y = p(x)$  для всех точек из  $x$ .

### Содержание пояснительной записки

1. Формулировка цели работы.
2. Таблицы  $F(\Phi)$  и  $p(\Phi)$ .
3. Алгоритм расчета и соответствующий ему текст программы.
4. Значения коэффициентов аппроксимирующих полиномов  $\bar{p}$  и  $p$  для  $n=3$  и  $n=5$ .
5. Графики  $F(\Phi)$  и  $p(\Phi)$  и выводы о качестве аппроксимации.

## II. ИССЛЕДОВАНИЕ ПЕРЕХОДНЫХ ПРОЦЕССОВ ДИНАМИЧЕСКОЙ СИСТЕМЫ

**Цель:** исследовать характер переходных процессов, используя численное интегрирование СНДУ объекта; построить модель динамической системы в среде SIMULINK.

### Постановка задачи

Исходные уравнения (1) описывают полностью динамику объекта. Исследование переходных процессов подразумевает расчет изменения во вре-

мени всех переменных в уравнениях (1), что может быть осуществлено любым численным методом интегрирования. Однако, для решения численным методом необходимо: во-первых, избавиться от алгебраических уравнений (исключив промежуточные переменные); во-вторых, записать дифференциальные уравнения в канонической форме Коши. Принято представлять уравнения динамической системы в виде векторных дифференциального уравнения состояния и алгебраического уравнения выхода:

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)); \\ y(t) = g(x(t), u(t)). \end{cases} \quad (3)$$

где  $x \in R^n$ ,  $u \in R^l$ ; функция  $g(x, u)$  действует из  $R^{n+l}$  в  $R^m$  и является в общем случае нелинейной. Переменная  $y \in R^m$  есть вектор выходных координат, т.е. переменных, значения которых доступны измерению. Непосредственно измерить все компоненты вектора  $x$  в большинстве случаев невозможно. Измерениям доступна, как правило, лишь некоторая функция переменных состояния и управления. Значение этой функции в любой момент времени  $t$ , т.е.  $y(t)$ , есть вектор измеряемых переменных – информация, по которой можно судить о том, что происходит в системе.

**Пример.** Для приведения модели (1) к обобщенной форме необходимо осуществить ряд преобразований. Выразим переменную  $F$  через  $\Phi$ ,  $F = \Lambda^{-1}(\Phi)$ , а зависимость  $F = \Lambda^{-1}(\Phi)$  выразим через полином  $p(\Phi)$ , полученный в работе №1. Далее  $i_\phi$  выразим через  $u_\phi$  и  $r_\phi$ . Затем произведем нормирование переменных (относительно номинальных значений параметров):

$$\bar{\Phi} = \Phi / \Phi_H, \bar{i}_T = i_T / i_{TH}, \bar{\omega} = \omega / \omega_H, \bar{u}_B = u_B / u_{BH}, \bar{M}_B = M_B / M_{BH}.$$

В итоге, преобразованная форма записи уравнений имеет вид:

$$\left\{ \begin{aligned} \frac{d\bar{\Phi}(t)}{dt} &= \frac{1}{\Phi_H w} \left[ -\frac{r_B}{w} (F_H \bar{p}(\bar{\Phi}) - \bar{i}_T w_c i_{TH}) + u_{BH} \bar{u}_B \right]; \\ \frac{d\bar{i}_T(t)}{dt} &= \frac{1}{L_\pi i_{TH}} [c_e \Phi_H \omega_H \bar{\Phi} \bar{\omega} - i_{TH} (r_\pi + R_0) - \\ &\quad - \frac{w_c}{w} (\frac{r_B}{w} (F_H \bar{p}(\bar{\Phi}) - \bar{i}_T w_c i_{TH}) + u_{BH} \bar{u}_B)]; \\ \frac{d\bar{\omega}(t)}{dt} &= \frac{1}{J \omega_H} [M_{BH} \bar{M}_B - c_M \Phi_H i_{TH} \bar{\Phi} \bar{i}_T]; \\ i_B &= \frac{F_H}{w} \bar{p}(\bar{\Phi}) - \frac{i_{TH} w_c}{w} \bar{i}_T; \\ \bar{F} &= \bar{p}(\bar{\Phi}), \end{aligned} \right.$$

где в качестве выходных переменных приняты  $i_b$  и  $\bar{F}$ .

Вводя обозначения:

$$x = [x_1 \ x_2 \ x_3]^T = [\bar{\Phi} \ \bar{i}_2 \ \omega]^T,$$

$$u = [u_1 \ u_2 \ u_3]^T = [\bar{u}_e \ \bar{M}_e \ R_0]^T,$$

$$y = [y_1 \ y_2]^T = [i_e \ \bar{F}]^T,$$

можно перейти к описанию СНДУ в унифицированном виде (3).

Стандартные методы и пакеты программ для исследования динамических систем ориентированы на такое представление. При этом функция  $g(x, u)$  может быть сформирована как угодно.

Для исследования переходного процесса необходимо выбрать статический режим – равновесное состояние системы, характеризующееся неизменными во времени значениями  $x(t) \equiv x^{(0)}$  и  $u(t) \equiv u^{(0)}$ . Переходный процесс в системе возникает, когда вектор входов принимает новое постоянное значение  $u^{(1)} \neq u^{(0)}$ . В этом случае происходит изменение вектора  $x$  от значения  $x^{(0)}$  к значению  $x^{(1)}$ . Переходный процесс в статический режим  $(u^{(1)}, x^{(1)})$  практически заканчивается за конечное время. Поэтому построение кривых переходного процесса заключается в следующем: для СНДУ (3) вектор  $x^{(0)}$  принимают в качестве начальных условий  $x(t_0) = x^{(0)}$ , далее численным методом находят решение этой системы на отрезке  $[t_0, t_1]$  при входном векторе  $u(t) \equiv u^{(1)}$  на  $[t_0, t_1]$ . По графикам  $x(t)$  и  $y(t)$  оценивают время переходного процесса и его характер (апериодичность, колебательность и пр.).

### Содержание

1. Записать СНДУ вида (3) в нормированном виде. Выходные переменные (вектор  $y$ ) выбираются по указанию преподавателя.

2. Написать и отладить программу решения СНДУ численным методом и построения графиков переходных процессов на языке MATLAB либо построить ее модель в среде SIMULINK (по указанию преподавателя).

3. Выбрать статический режим  $(u^{(0)}, x^{(0)})$ , из которого начинается переходный процесс. По умолчанию - номинальный режим.

4. Выбрать (по указанию преподавателя) статические режимы  $(u^{(i)}, x^{(i)})$   $i = 1, 2, \dots$ , в которые будет осуществлен переход системы. Режимы задать

следующим образом: 1)  $((u^{(1)}, x^{(1)})) : u^{(1)}$  отличается от  $u^{(0)}$  только первой компонентой (на 15-20%); 2)  $((u^{(2)}, x^{(2)})) : u^{(2)}$  отличается от  $u^{(0)}$  только второй компонентой и т.д. Дополнительно следует выбрать один – два режима, где векторы входа изменены по всем компонентам относительно значения  $u^{(0)}$ .

5. С помощью программы либо модели для каждого статического режима рассчитать графики  $x(t)$  и  $y(t)$  при переходе системы в этот режим. При этом шаг и время интегрирования подобрать экспериментально из условия обеспечения устойчивого решения и окончания переходных процессов.

6. Построить графики переходных процессов.

7. Построить фазовые портреты  $x_i(x_j), i \neq j$  для нескольких режимов.

8. Изменить исходные данные – увеличить в 10 раз значение индуктивности якоря. Выполнить пункты 5, 6, 7.

### Рекомендации по программированию

Понадобится следующая функция языка MATLAB, реализующая решение СНДУ численными методами:

`[y, x] = nsim('ff', 'gg', t, u, x0, 'method');`

где 'ff' – имя файла с программой вычисления значения вектор-функции правых частей  $f(x, u)$ ; 'gg' – имя файла с программой вычисления значения вектор-функции выходов  $g(x, u)$ ;  $t$  – вектор-строка моментов времени, для которых рассчитываются значения  $x(t)$  и  $u(t)$ ;  $u$  – матрица, столбцы которой есть значения вектора входа  $u$  в моменты времени, заданные в векторе-строке  $t$ ;  $x0$  – вектор начальных условий; 'method' – численный метод интегрирования, возможные варианты которого: 'euler' – метод Эйлера, 'eulermod' – модифицированный метод Эйлера и 'rk4s' – метод Рунге-Кутты 4-го порядка точности;  $X$  – матрица, столбцы которой есть значения вектора состояния  $x$  в моменты времени, заданные в векторе-строке  $t$ ;  $y$  – матрица, столбцы которой есть значения вектора выхода  $y$  в моменты времени, заданные в векторе-строке  $t$ .

Ниже приведена структура программы, позволяющей построить графики переходных процессов при переходе в номинальный режим при нулевых начальных условиях:

```
% определение констант и присвоение им значений
gw=97.2;
```

```

global rw;
.....
% определение аппроксимирующего полинома
p = [0.23; 0; 0.15; 0];
global p;
% формирование набора значений моментов времени
t = 0.0:0.01:1.0; % набор значений t из [0 с, 1 с] с шагом 0.01 с
% формирование вектора u(t), соответствующего номинальному
режиму
u = [ones(size(t)); % строка из значений 1-го входа
     ones(size(t)); % строка из значений 2-го входа
     ones(size(t))]; % строка из значений 3-го входа
% задание начальных значений вектора x
x0 = [0; 0; 0]; % это соответствует
% интегрирование с шагом 0.01 с
[y, x] = nsim('fun_F', 'fun_G', t,u,x0, 'euler')
% построение графиков x(t), y(t)
.....
% вычисление значения функции f(x, u)
function z = fun_F(x, u, t)
global p; ...
z = [.....]; % z – вектор-столбец значений правых частей

% вычисление значения функции g(x, u)
function z = fun_G(x, u, t)
global p; ...
z = [.....]; % z – вектор-столбец значений выходных пере-
менных

```

### **Содержание пояснительной записки**

1. Формулировка цели работы.
2. Исходные уравнения объекта и система (3) относительно нормированных переменных (без подстановки значений параметров), а также все промежуточные выкладки.
3. Система (3) с численными значениями параметров объекта.
4. Программа на языке MATLAB или модель в среде SIMULINK.

5. Графики переходных процессов и фазовых траекторий.
6. Оценка характера переходных процессов и выводы.

## II. ИССЛЕДОВАНИЕ СТАТИЧЕСКИХ РЕЖИМОВ ДИНАМИЧЕСКОЙ СИСТЕМЫ

**Цель:** преобразовать исходную систему уравнений в СНЛАУ, описывающую статические режимы; рассчитать статические характеристики динамической системы при помощи средств пакета MATLAB.

### Постановка задачи

Статический режим динамической системы – это ее равновесное состояние, соответствующее окончанию переходных процессов. Например, изменение напряжения возбуждения на новое постоянное значение вызывает изменение МДС, магнитного потока, тока и напряжения генератора и т.д. Переходный процесс заканчивается новыми установившимися значениями этих величин, т.е. новым статическим режимом. Статический режим будет описывать система алгебраических уравнений, т.е. уравнений, куда не входят производные, так как последние в статическом режиме равны нулю.

Все статические режимы могут быть описаны СНЛАУ, записанной в обобщенной форме относительно компонент векторов  $u$  и  $x$ :

$$\begin{aligned} & f(x, u) = 0, \\ & \left\{ \begin{array}{l} f_1(x, u) = 0; \\ \dots \\ f_n(x, u) = 0. \end{array} \right. \end{aligned} \tag{4}$$

или

Имея описание нелинейной динамической системы в виде (3), легко получить обобщенную форму записи СНЛАУ, при этом алгебраическое уравнение из (3) не рассматривается.

**Пример.** Для объекта (1) описание статических режимов после исключения производных, сокращения постоянных множителей и упрощения выглядит следующим образом:

$$\left\{ \begin{array}{l} -\frac{r_B}{w} (F_H \bar{p}(\bar{\Phi}) - \bar{i}_T w_c i_{TH}) + u_{BH} \bar{u}_B = 0; \\ c_e \Phi_H \omega_H \bar{\Phi} \bar{\omega} - i_{TH} (r_y + R_0) = 0; \\ M_{BH} \bar{M}_B - c_M \Phi_H i_{TH} \bar{\Phi} \bar{i}_T = 0. \end{array} \right.$$

Эту систему легко переписать через переменные  $x_1, x_2, x_3, u_1, u_2, u_3$ .

Значение вектора  $x$  в статическом режиме может быть найдено как решение СНЛАУ, когда  $u$  задано, а  $x$  является неизвестным. Решение СНЛАУ ищут, как правило, численными методами, в данном случае методом Ньютона. Для реализации метода Ньютона необходимо найти матрицу частных производных:

$$G(x, u) = \begin{bmatrix} \partial f_1 / \partial x_1 & \partial f_1 / \partial x_2 & \dots & \partial f_1 / \partial x_n \\ \partial f_2 / \partial x_1 & \partial f_2 / \partial x_2 & \dots & \partial f_2 / \partial x_n \\ \dots & \dots & \dots & \dots \\ \partial f_n / \partial x_1 & \partial f_n / \partial x_2 & \dots & \partial f_n / \partial x_n \end{bmatrix}. \quad (5)$$

Множеству всех возможных значений вектора  $u$  соответствует множество значений вектора  $x$ . Множество пар вида  $(u, x)$  есть множество всех статических режимов данной динамической системы. На практике для описания этого множества пользуются статическими характеристиками. Статической характеристикой называется зависимость вида

$$x(u_i), u_j = u_j^{(0)} = \text{const}, j \neq i.$$

Это могут быть три графика  $\Phi(u_B), i_T(u_B), \omega(u_B)$  при некоторых постоянных значениях  $M_B$  и  $R_0$ .

Статическую характеристику можно построить следующим образом:

1) выбирают варьируемую переменную, например,  $u_2$ . Определяют диапазон ее изменения и в этом диапазоне более или менее равномерно выбирают  $M$  значений  $u_2^{(1)}, \dots, u_2^{(M)}$ . Значения остальных входных переменных фиксируются:  $u_1 = u_1^{(0)}, u_3 = u_3^{(0)}$ .

2) для каждого вектора вида  $[u_1^{(0)}, u_2^{(i)}, u_3^{(0)}], i = 1, \dots, M$  решают СНЛАУ и находят вектор  $[x_1^{(i)}, x_2^{(i)}, x_3^{(i)}], i = 1, \dots, M$ .

Зависимости  $x_1(u_2), x_2(u_2), x_3(u_2)$  при  $u_1 = u_1^{(0)}$  и  $u_3 = u_3^{(0)}$  представляют собой статические характеристики объекта. Потом можно выбрать новые значения для  $u_1$  и  $u_3$  и построить новые характеристики. Затем выбирают новую варьируемую переменную, например  $u_3$ , и весь процесс повторяют.

## Содержание

1. Найти и записать описание множества статических режимов объек-



та в форме СНЛАУ (4) для нормированных переменных.

2. Записать аналитическое выражение для матрицы частных производных  $G(x, u)$  (5).

3. Определить состав рассчитываемых статических характеристик (по указанию преподавателя).

4. Для каждой статической характеристики выбрать: диапазон изменения и набор промежуточных значений варьируемой входной переменной; фиксированные значения остальных входных переменных (по умолчанию принимаются номинальные значения).

5. Написать программу на языке MATLAB, осуществляющую расчет статических характеристик с помощью решения СНЛАУ методом Ньютона.

6. С помощью написанной программы рассчитать и построить графики статических характеристик.

### Рекомендации по программированию

Понадобится следующая функция языка MATLAB, реализующая решение СНЛАУ методом Ньютона:

```
x=newton('fun_F', 'fun_G' , x0, u, eps)
```

где 'fun\_F' – имя функции, вычисляющей  $f(x, u)$ , 'fun\_G' - имя функции, вычисляющей матрицу частных производных  $G(x, u)$ , - точка начального приближения, u - вектор значений  $u$ , eps - точность метода, - вектор значений решения СНЛАУ. Функция newton не входит в состав пакетов прикладных программ среды MATLAB.

Ниже приведена наиболее простая структура программы, позволяющей построить статические характеристики типа  $x_1(u_2), x_2(u_2), x_3(u_2)$  при  $u_1 = 1$  и  $u_3 = 1$ .

```
% определение констант и присвоение им значений
rw=97.2;
global rw;
.....
% определение аппроксимирующего полинома
p = [0.23; 0; 0.15; 0];
global p
% формирование набора значений вектора входов при
% постоянных значениях первой и третьей координат
u2 = 1.2:-0.2:0.2; % набор значений u2 с шагом 0.2
```

```

% формирование значений вектора u
u = [ones(size(u2));
      u2;
      ones(size(u2))];
% задание начальных значений вектора x
x0 = [1; 1; 1];
% цикл расчета значений вектора x
xx=[];
for i=1:length(u2),
    x=newton('fun_F', 'fun_G', x0, u(:,i), eps);
    xx=[xx x];
    x0=x;
end
% построение графиков x1(u2), x2(u2), x3(u2)
.....
% вычисление значения функции f(x, u)
function z = fun_F(x, u)
z = [.....]; % формируется вектор-столбец из трех компонент

% вычисление значения функции G(x, u)
function z = fun_G(x, u)
z = [.....]; % формируется матрица строения 3 на 3

```

### Содержание пояснительной записки

1. Исходные уравнения объекта (в форме СНДУ) и СНЛАУ относительно нормированных переменных в общем виде.
2. Выражение для матрицы  $G(x, u)$  для нормированных переменных.
3. Система алгебраических уравнений и матрица  $G(x, u)$  с численными значениями.
4. Перечень статических характеристик, подлежащих расчету. По каждой характеристике: состав и значения фиксированных входных переменных, набор значений варьируемых входных переменных, значение  $x_0$  на момент начала расчетов.
5. Программа на языке MATLAB.
6. Графики статических характеристик и табл. результатов расчетов.
7. Сопоставление статических характеристик с результатами модели-

рования, полученными в лабораторной работе №2.

8. Выводы.

#### IV. ИССЛЕДОВАНИЕ ЛИНЕАРИЗОВАННОЙ МАТЕМАТИЧЕСКОЙ МОДЕЛИ

**Цель работы:** Провести линеаризацию СНДУ в окрестности статического режима, получить линеаризованную математическую модель, описывающей динамику системы при малых отклонениях входных переменных от рассматриваемого статического режима; исследовать линеаризованную модель на ее соответствие нелинейной модели; исследовать устойчивость системы и характер переходных процессов.

##### Постановка задачи

В этой работе исследуется новый тип математической модели – математическая модель, линеаризованная в окрестности статического режима.

Рассматривают СНДУ, записанную в форме Коши:

$$\begin{aligned}\dot{x}(t) &= f(x(t), u(t)); \\ y(t) &= g(x(t), u(t)).\end{aligned}\tag{6}$$

Выбирают статический режим  $(x^{(0)}, u^{(0)})$ , в окрестности которого производится линеаризация.

Далее определяют матрицы частных производных:  
матрица состояния:

$$A = \frac{\partial f(x, u)}{\partial x} \Big|_{x=x^{(0)}, u=u^{(0)}} = \begin{bmatrix} \frac{\partial f_1(x, u)}{\partial x_1} \Big|_{x=x^{(0)}, u=u^{(0)}} & \dots & \frac{\partial f_1(x, u)}{\partial x_n} \Big|_{x=x^{(0)}, u=u^{(0)}} \\ \vdots & \dots & \vdots \\ \frac{\partial f_n(x, u)}{\partial x_1} \Big|_{x=x^{(0)}, u=u^{(0)}} & \dots & \frac{\partial f_n(x, u)}{\partial x_n} \Big|_{x=x^{(0)}, u=u^{(0)}} \end{bmatrix};$$

Аналогичным образом:

матрица входов: 
$$B = \frac{\partial f(x, u)}{\partial u} \Big|_{x=x^{(0)}, u=u^{(0)}};$$

матрица выходов: 
$$C = \frac{\partial g(x, u)}{\partial x} \Big|_{x=x^{(0)}, u=u^{(0)}};$$

матрица обхода:

$$D = \frac{\partial g(x, u)}{\partial u} \Big|_{x^{(0)}, u^{(0)}}.$$

Матрицы  $A$ ,  $B$ ,  $C$ ,  $D$  имеют постоянные коэффициенты, зависящие от статического режима, т.е. от  $u^{(0)}$  и  $x^{(0)}$ . Для СНДУ рассматриваемого типа коэффициенты матриц могут быть определены аналитически, т.к. для каждой частной производной аналитическое выражение записать достаточно просто. Если аналитическое выражение для какого-либо коэффициента записать невозможно или сложно, то этот коэффициент можно определить численно, например:

$$\frac{\partial f_2(x, u)}{\partial u_3} \Big|_{x=x^{(0)}, y=y^{(0)}} \approx \frac{1}{\delta} \left[ f_2(x_1^{(0)}, \dots, x_n^{(0)}, u_1^{(0)}, u_2^{(0)}, u_3^{(0)} + \delta, u_4^{(0)}, \dots, u_l^{(0)}) - \right. \\ \left. - f_2(x_1^{(0)}, \dots, x_n^{(0)}, u_1^{(0)}, \dots, u_l^{(0)}) \right],$$

где значение  $\delta$  должно быть малым. Система линейных уравнений вида:

$$\begin{cases} \Delta \dot{x}(t) = A \cdot \Delta x(t) + B \cdot \Delta u(t); \\ y(t) = C \cdot \Delta x(t) + D \cdot \Delta u(t). \end{cases} \quad (7)$$

записанная относительно векторов, обозначаемых через  $\Delta x = x(t) - x^{(0)}$ ,  $\Delta u = u(t) - u^{(0)}$ ,  $\Delta y(t) = y(t) - y^{(0)}$  называется системой, полученной из системы (6) путем линеаризации последней в окрестности статического режима  $(u^{(0)}, x^{(0)})$ .

Эта система обладает следующим замечательным качеством. Если рассмотреть решение системы (6) при некоторых начальных условиях  $x(t_0) = x^*$  и при некотором изменении вектора входов  $u(t)$  на интервале  $t \in [t_0, t_1]$ , то полученное решение  $x(t)$  обладает свойством:

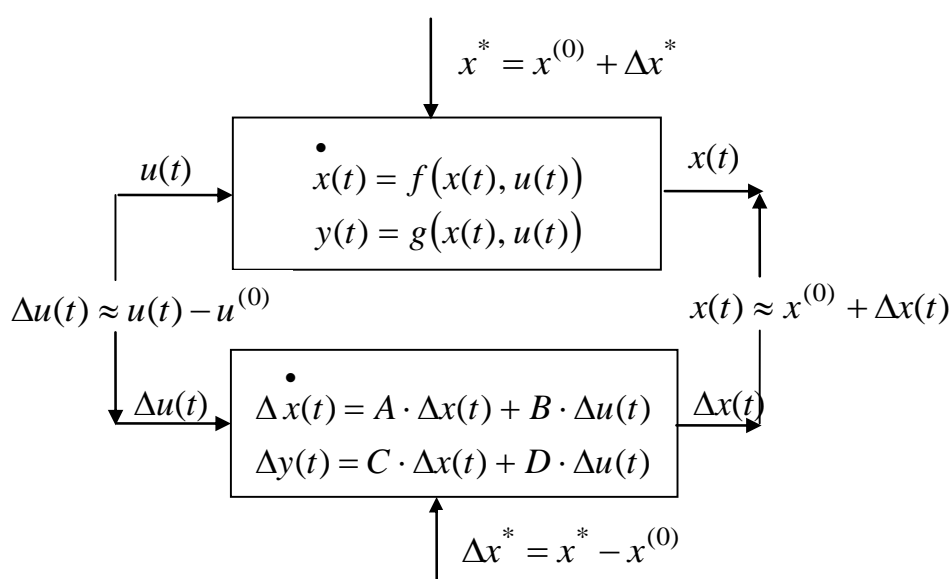
$$x(t) \approx x^* + \Delta x(t), t \in [t_0, t_1], \quad (8)$$

где  $\Delta x(t)$  – решение системы (7) при начальных условиях  $\Delta x(t_0) = x^* - x^{(0)}$  и при входном векторе  $\Delta u(t_0) = u(t) - u^{(0)}$ ,  $t \in [t_0, t_1]$ . Другими словами, соответствие математических моделей (6) и (7) выражено соотношениями:

$$\begin{aligned} x_i(t) &\approx x_i^{(0)} + \Delta x_i(t), \quad i = 1, \dots, n; \\ u_i(t) &\approx u_i^{(0)} + \Delta u_i(t), \quad i = 1, \dots, l; \\ y_i(t) &\approx y_i^{(0)} + \Delta y_i(t), \quad i = 1, \dots, m; \end{aligned} \quad (9)$$

Переменные  $\Delta x_i(t)$ ,  $\Delta u_i(t)$  называются приращениями (или отклонениями) переменных  $x_i$ ,  $u_i$  относительно значений  $x_i^{(0)}$ ,  $u_i^{(0)}$ .

Таким образом, в окрестности статического режима динамику исходного объекта (6) с большой точностью описывает динамика объекта (7). Знак приближенного равенства в (8) означает, что при значительном отклонении значений вектора  $u(t)$  от значения  $u^{(0)}$ , т.е. при больших значениях компонент вектора  $\Delta u_i(t)$ , схожесть будет меньше. В результате процесс исследования динамики нелинейного объекта (6) в окрестности его статического режима может быть осуществлен по схеме, показанной на рис. 1.



*Рис. 1.*

Описание в виде (7) является общепринятым. Практически все стандартные методы и программные средства для работы с линейными системами используют такое представление. В частности, пакет MATLAB ориентирован именно на такое представление, когда объект исследования описан в пространстве состояний.

СЛДУ есть частный случай СНДУ, поэтому к объекту (7) применимо все, что изложено в работах №2 и №3 по отношению к нелинейным объектам, т.е. статические режимы и переходные процессы можно определить теми же методами. Однако для линейных объектов набор методов исследования значительно проще и разнообразнее.

В этой работе рассчитывают только переходные процессы в линейной системе (7) и исследуют их соответствие переходным процессам в нелиней-

ной системе (6), согласно соотношениям (8). Именно, если в работе №2 был рассчитан переходный процесс из статического режима  $(u^{(0)}, x^{(0)})$  в статический режим  $(u^{(1)}, x^{(1)})$ , то для линеаризованной системы это означает переход из статического режима  $(\Delta u^{(0)}, \Delta x^{(0)})$  в статический режим  $(\Delta u^{(1)}, \Delta x^{(1)})$ . При этом  $\Delta u^{(0)} = 0 \in R^l$ ,  $\Delta x^{(0)} = 0 \in R^n$ ,  $\Delta u^{(1)} = u^{(1)} - u^{(0)}$ ,  $\Delta x^{(1)} = x^{(1)} - x^{(0)}$ . Расчет переходного процесса для линейной системы означает расчет решения СЛДУ (7), т.е. вектор-функции  $\Delta x(t)$  на  $t \in [t_0, t_1]$ , при начальных условиях  $\Delta x(t_0) = \Delta x^{(0)}$  и при входной вектор-функции  $\Delta u(t) = \Delta u^{(1)}$  на  $[t_0, t_1]$ . Решение  $\Delta x(t)$  соответствует решению  $x(t)$  для системы (6). Решение  $\Delta x(t)$  может быть определено любым численным методом. Однако для линейных систем используют специальные методы, позволяющие осуществлять интегрирование с большим шагом.

Использование линеаризованной модели дает следующие преимущества при исследовании динамической системы.

1. Решение СЛДУ всегда проще найти, в ряде случаев оно может быть найдено аналитически.

2. Для линейных систем определены такие понятия, как передаточные функции и частотные характеристики. Для оценки устойчивости движения в окрестности положения равновесия достаточно рассчитать собственные числа матрицы  $A$ . Кроме того, для линейных систем справедлив принцип суперпозиции, что позволяет, например, использовать методы частотного анализа.

3. Для линейных систем методы теории автоматического управления разработаны наиболее полно. Это позволяет строить законы управления для линеаризованных объектов и с учетом преобразования (8) переносить их на исходные нелинейные объекты.

## Содержание

1. Найти аналитические выражения для коэффициентов матриц  $A, B, C, D$ .
2. Для статического режима  $(u^{(0)}, x^{(0)})$ , из которого рассчитывали переходные процессы в п. II настоящих методических указаний, рассчитать значения коэффициентов матриц.
3. Написать и отладить программу на языке MATLAB для расчета переходных процессов в линейной системе.

4. Рассчитать переходные процессы из статического режима ( $\Delta u^{(0)} = 0$ ,  $\Delta x^{(0)} = 0$ ) в статические режимы ( $\Delta u^{(1)}$ ,  $\Delta x^{(1)}$ ) и т.д., соответствующие в работе №2 статическим режимам ( $u^{(1)}$ ,  $x^{(1)}$ ) и т.д. Построить графики переходных процессов.

5. Проверить соответствие графиков  $\Delta x(t)$  и  $x(t)$ ,  $\Delta u(t)$  и  $u(t)$ ,  $\Delta y(t)$  и  $y(t)$  по соотношениям (11).

6. Оценить устойчивость системы по собственным числам матрицы  $A$ .

### Рекомендации по программированию

Понадобится следующая функция языка MATLAB реализующая решение СЛДУ численным методом:

`[y, x] = lsim(a, b, c, d, t, x0, u) ;`

где  $a$ ,  $b$ ,  $c$ ,  $d$  – матрицы  $A$ ,  $B$ ,  $C$ ,  $D$  соответственно;  $t$  – вектор-строка моментов времени, для которых рассчитывают значения  $x(t)$  и  $u(t)$ ;  $x0$  – вектор начальных условий;  $u$  – матрица, столбцы которой есть значения вектора входа  $u$  в моменты времени, заданные в векторе-строке  $t$ . Эта функция предназначена только для линейных систем, представленных в виде (7).

Ниже приведен фрагмент программы, позволяющей рассчитать графики  $\Delta x(t)$  и  $\Delta u(t)$  :

```
% формирование набора значений моментов времени
t = 0.0:0.01:1.0; % набор значений t из [0 с, 1 с]
% с шагом 0.01 с
% формирование значений вектора u(t), t из [0 с, 1 с]
u =[u1_*ones (t); % строка из значений u1_
u2_*ones (t); % строка из значений u2_
u3_*ones (t); % строка из значений u3_
% задание начальных значений вектора x
x0 = [0; 0; 0]; % при нормированных переменных это
% соответствует номинальному режиму
[y, x] = lsim(a, b, c, d, t, x0, u) ;
```

Расчет коэффициентов характеристического полинома матрицы  $A$  и его корней, а также вывод графика расположения корней на комплексной плоскости можно осуществить следующим образом:

```
p = poly (a);
lmd = roots (p);
```

plot (real (lmd), imag (lmd), '\*'), grid, pause

### Содержание пояснительной записки

1. Формулировка цели работы.
2. Аналитические выражения для уравнений (6) и матриц системы (7) относительно нормированных переменных (без подстановки значений параметров), а также все промежуточные выкладки.
3. Матрицы системы (7) после подстановки численных значений параметров объекта.
4. Программа на языке MATLAB.
5. Графики переходных процессов. Для каждого переходного процесса указать статический режим, из которого процесс начался, а также статический режим, которым переходный процесс окончился. Сравнить, согласуются ли переходные процессы с результатами работы №2.
6. Оценка степени соответствия переходных процессов.
7. Коэффициенты характеристического полинома матрицы  $A$  и график расположения корней полинома на комплексной плоскости.
8. Выводы.

## V. РАСЧЕТ ПЕРЕДАТОЧНЫХ ФУНКЦИЙ И ЧАСТОТНЫХ ХАРАКТЕРИСТИК ДИНАМИЧЕСКОЙ СИСТЕМЫ

**Цель работы:** определить ПФ линеаризованной модели от каждого входа к каждой переменной состояния при условии, что в начальный момент времени система находится в состоянии равновесия; исследовать переходные процессы и частотные характеристики линейной системы.

### Постановка задачи.

Для объекта (7) (при нулевых начальных условиях) матричная ПФ от векторного входа  $\Delta u$  к векторному выходу  $\Delta y$ :

$$W(s) = C(sI - A)^{-1} \cdot B + D$$

представляет собой матрицу строения  $m \times l$ . Ее элементами являются дробно-рациональные функции комплексной переменной – скалярные передаточные функции  $W_{\Delta y_j / \Delta u_i}(s)$ , от компонент вектора  $\Delta u$  к компонентам вектора  $\Delta y$ .

Матричная ПФ  $W(s)$  соответствует схеме на рисунке 2.



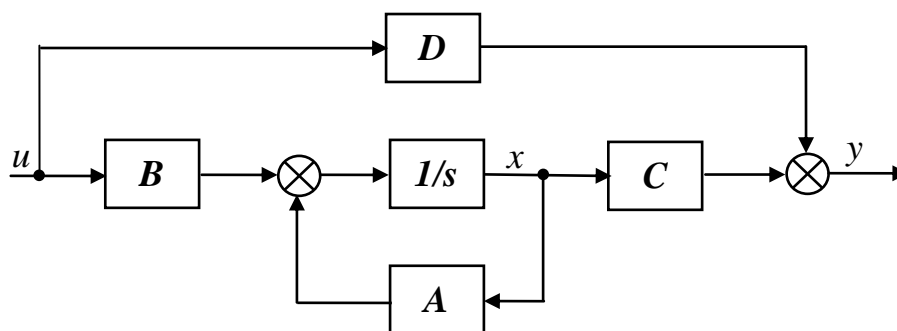


Рис. 2. Передаточная функция  $W(s)$

ПФ позволяют, в частности, рассчитывать частотные характеристики объекта и анализировать реакцию объекта на типовые воздействия.

В этой работе основной задачей является расчет матричной ПФ объекта (7) и расчет частотных характеристик по всем скалярным ПФ. Также рассматривают переходные характеристики, т.е. реакцию системы на единичное ступенчатое воздействие. Реакция системы на другие воздействия не рассматривается. Последовательность расчета следующая:

1) считать вектор выхода равным вектору состояния и сформировать матрицу  $C$  как единичную матрицу строения  $n \times n$ . Матрицу  $D$  сформировать нулевой строения  $n \times l$ .

2) рассчитать  $W(s)$  и выбрать из нее  $i$ -й столбец, т.е. скалярные ПФ  $W_{\Delta y_1 / \Delta u_i}(s), \dots, W_{\Delta y_n / \Delta u_i}(s)$ ;

3) рассчитать  $x_j(t)$ ,  $j = 1, \dots, n$  как результаты прохождения входного сигнала  $\Delta u_i(t)$  через соответствующие скалярные ПФ.

Поскольку сигнал  $\Delta u_i(t)$  представляет собой ступенчатое воздействие, то  $x_j(t)$  подобна переходной характеристике соответствующей ПФ, т.е. реакции на единичное ступенчатое воздействие. Отличие состоит только в постоянном множителе, т.к. скачок  $\Delta u_i(t)$  не равен единице в общем случае.

Очевидно, что для исследования только  $x(t)$  матрицу  $C$  выбрали единичной. Когда вектор выхода другой, то следует корректно сформировать другие матрицы  $C$  и  $D$ .

## Содержание

1. Рассчитать коэффициенты ПФ  $W_{\Delta y_j / \Delta u_i}(s)$  от каждого входа к каждому выходу.

2. Рассчитать АЧХ и ФЧХ для каждой скалярной ПФ.

3. Выбрать из работ №2 и №4 переходные процессы, в которых конечный статический режим отличается от начального изменением только одной компоненты вектора входов  $u_i$  (или  $\Delta u_i$ ),  $i=1, \dots, l$ . Проверить по графикам АЧХ от каждого входа, что коэффициенты передачи к каждому выходу (значение АЧХ в нуле) равны коэффициентам передачи для линейной системы (т.е. отношениям  $|\Delta y|/|\Delta u|$  на момент окончания переходных процессов).

4. Для тех же случаев построить переходные характеристики для каждой выходной координаты. Убедиться, что переходный процесс соответствует по форме переходному процессу, рассчитанному в лабораторной работе №4.

### Рекомендации по программированию

Необходимо использовать функции библиотеки LTI языка MATLAB, в которой линейные динамические системы описываются в виде LTI-объекта:

`sys = ss(A,B,C,D)` – формирование LTI-модели в пространстве состояния;

`sys = tf(num,den)` – формирование LTI-модели в пространстве ПФ, num, den – соответственно полиномы числителя и знаменателя.

Примечание: функции ss и tf также служат для преобразования LTI-моделей из одной формы описания в другую, для этого модель нужно указать единственным аргументом.

`bode(sys)` – расчет и построение логарифмических частотных характеристик (диаграмм Боде);

`step(sys)` – расчет и построение переходной характеристики.

Более подробную информацию об указанных функциях библиотеки LTI можно узнать, набрав в командной строке `help имя_функции`.

### Содержание пояснительной записки

1. Формулировка цели работы.
2. Значения коэффициентов всех скалярных ПФ из матричной ПФ  $W_{\Delta y_j / \Delta u_i}(s)$ . Значения корней знаменателя этих ПФ. Значения корней числителей этих ПФ.
3. Графики АЧХ и ФЧХ для каждой ПФ.
4. Графики переходных характеристик.
5. Тексты написанных и использованных программ.
6. Характеристика результатов и выводы.

## ОСНОВЫ РАБОТЫ В ИНТЕГРИРОВАННОЙ СРЕДЕ MATLAB

### Общие сведения

MATLAB – это высокопроизводительная инструментальная система для математической поддержки научно-технической деятельности. Система обеспечивает выполнение вычислений и визуализацию их результатов с использованием программирования в удобной интегрированной среде в форме, которая близка к математической.

Система MATLAB является интерактивной средой, в которой основной математической формой представления данных служит матрица. Подобная ориентация системы делает ее особо значимой для решения прикладных задач с использованием матричных методов.

В MATLAB важная роль отводится специализированным пакетам прикладных программ (ППП), называемых *toolboxes*. Они носят проблемно ориентированный характер и применяются для высокоэффективного решения частных задач с использованием специализированных вычислительных методов.

Система MATLAB состоит из шести основных элементов:

1. Интегрированная среда – набор интерфейсных инструментов, находящихся на рабочем столе системы, с которыми работает пользователь при выполнении отдельных команд или при разработке приложений с написанием программного кода. Среда включает в себя средства управления переменными в рабочем пространстве MATLAB, вводом и выводом данных, а также средства формирования, редактирования контроля и отладки программных текстов и сохранения их в виде файлов.

2. Язык программирования MATLAB – язык высокого уровня, позволяющий оперировать с векторами и матрицами, как с единым целым. Он позволяет управлять ходом вычислений, использовать функции и структуры данных, осуществлять ввод и вывод данных. Существенная роль отводится средствам объектной технологии.

3. Управляемая графика – графическая подсистема системы MATLAB, которая включает в себя команды высокого уровня для визуализации двух- и трехмерных данных, обработки изображений, выполнения анимации и формирования иллюстрированной графики. Графическая подсистема также

включает в себя команды низкого уровня, позволяющие полностью контролировать внешний вид графических объектов. С ее помощью осуществляется разработка графического интерфейса (GUI) пользователя для управления функционирования создаваемых приложений.

4. Библиотека математических функций – обширная коллекция вычислительных алгоритмов, охватывающих практически все разделы современных численных методов. Библиотека позволяет работать с элементарными и специальными функциями, комплексными числами, полиномами и матрицами.

5. Пакеты прикладных программ. Совокупность ППП по существу неограниченно расширяет возможности системы MATLAB за счет проблемной специализации вычислительных методов для решения прикладных задач в различных научно-технических областях. В настоящее время версия 6.5 системы включает в себя около 40 ППП.

6. Подсистема Simulink моделирования динамических объектов - интерактивное инструментальное средство, непосредственно взаимодействующее с MATLAB, которое позволяет формировать компьютерные модели нелинейных динамических систем и выполнять имитационное моделирование протекающих в них процессов. Компьютерные модели формируются в визуальном режиме с использованием компонентного подхода. Результаты моделирования могут быть переданы в MATLAB для анализа или синтеза.

### **Элементарные операции с матрицами**

*Представление чисел.* Числа в MATLAB могут быть целыми, дробными, с фиксированной и плавающей точкой, комплексными. Ниже приведены примеры представления чисел:

<i>вещественные числа</i>	<i>комплексные числа</i>
1	3i
-2	-2j
4.321	2+4i
0.000000001	-0.0001j
987.6543e-21	-5.4321i
-123.567e10	-123.45+1.2e-4i

*Переменные.* Система MATLAB выполняет различные действия над переменными, каждая из которых имеет некоторое имя. Имена назначаются пользователем или системой и формируются как произвольные последова-

тельности латинских букв и цифр (допустим также символ подчеркивания), начинающиеся обязательно с буквы. При этом система различает прописные и строчные буквы. Указание имени новой переменной в левой части оператора присваивания приводит к её автоматическому размещению в оперативной памяти, трактуемой в среде как «рабочее пространство» (WorkSpace). Удалить переменную из рабочего пространства можно только с помощью команды `clear`.

*Системные переменные.* В системе по умолчанию определены некоторые часто используемые переменные, основными из которых являются следующие:

`i` или `j` – мнимая единица  $\sqrt{-1}$ ;

`pi` – число  $\pi=3.141592653589793\dots$ ;

`inf` – значение машинной бесконечности;

`ans` – переменная, в которой хранится результат выполнения последней операции (создается автоматически, когда не определены выходные аргументы какой-либо команды);

`NaN` – указание на нечисловой характер данных (Not-a-Number);

Для очистки рабочего пространства используются следующие варианты команды `clear`:

`clear` – уничтожение всех переменных;

`clear x` – уничтожение переменной `x`;

`clear a, b, c` – уничтожение нескольких указанных переменных.

*Задание матриц.* Для задания некоторых матриц стандартного вида в систему MATLAB включены функции:

1. `zeros(m,n)` - нулевая матрица размера  $n \times m$ ;

2. `eye(n)` - единичная матрица размера  $n \times n$ ;

3. `ones(n,m)` - матрица размера  $n \times m$ , все компоненты которой равны 1.

4. `rand(n,m)` - матрица размера  $n \times m$  со случайными компонентами, которые равномерно распределены на отрезке  $[0,1]$ .

Прочие вектора и матрицы задаются вручную по следующим правилам: значения элементов матрицы перечисляются в квадратных скобках, элементы в строке разделяются пробелами или запятыми, строки разделяются точкой с запятой (;). Ниже приведены примеры формирования векторов и матриц.

`V1=[1 2 3]` – вектор-строка из трех элементов со значениями 1, 2 и 3.

`V2=[1;4;7]` – вектор-столбец из трех элементов со значениями 1, 4 и 7.

$M=[1\ 2\ 3; 4\ 5\ 6; 7\ 8\ 9]$  – матрица размера  $3 \times 3$  следующего вида:

$$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}.$$

Совершенно аналогично задаются блочные матрицы (состоящие из векторов и матриц).

Для определения размерности вектора служит функция `length(V)`, размерность матрицы можно определить при помощи функции `size(M)`.

Для указания отдельной компоненты вектора или матрицы используются выражения вида  $V(i)$  или  $M(i, j)$ .

*Оператор перечисления (:).* Этот оператор широко используется в различных формах для упрощения ряда манипуляций с матрицами. В частности:

$t = t1:h:t2$  – формирование вектора-строки, представляющего собой числовую последовательность от  $t1$  до  $t2$  с шагом  $h$  (по умолчанию шаг равен единице, для убывающей последовательности шаг должен быть отрицательным);

$A(n1:n2,m1:m2)$  – выделение блока из состава матрицы  $A$ , ограниченного строками  $n1$  и  $n2$  и столбцами  $m1$  и  $m2$  (в случае, если перечисляются все строки или столбцы, можно указать просто оператор перечисления без чисел). Например, выражение  $A(4,:)$  выделяет 4-ю строку матрицы  $A$ .

При возникновении необходимости удаления отдельных строк или столбцов матрицы, используется пустая матрица (`[]`). Например, выражение  $M(2,:)=[]$  удаляет 2-ю строку матрицы  $M$ .

*Простейшие операции с матрицами.* Операции сложения, вычитания и умножения матриц выполняются в соответствии с обычными правилами линейной алгебры в естественной форме записи с операторами (+, -, \*). Размерности матриц должны быть согласованы. Согласования не требуют только арифметические операции между матрицей и числом. Знак точки (.) перед знаком операции означает, что операция над матрицами будет производиться над каждым элементом матрицы отдельно – поэлементно.

Операция транспонирования матрицы выполняется с использованием символа (`'`).

Для вычисления определителя квадратной матрицы используется команда (функция) `det`. Операция обращения квадратной матрицы выполняется

с помощью команды `inv`. Для поиска собственных чисел квадратной матрицы используется команда `eig`.

### Операции с полиномами

*Формирование.* Полиномы  $P(s) = p_n s^n + p_{n-1} s^{n-1} + \dots + p_1 s + p_0$  в среде MATLAB представляются в виде векторов-строк  $\mathbf{p} = [p_n \ p_{n-1} \ \dots \ p_1 \ p_0]$ , содержащих коэффициенты полиномов, начиная с коэффициента при старшей степени.

*Сложение и вычитание* полиномов выполняется по обычным правилам сложения матриц, поэтому предварительно полиномы должны быть приведены к одинаковой размерности путем добавления нулей слева к полиному меньшей степени (например,  $\mathbf{p} = [0 \ 0 \ \mathbf{p}]$ ).

*Умножение* полиномов обеспечивается функцией `conv()` с двумя аргументами полиномами сомножителями.

*Деление* полиномов выполняется с помощью функции `deconv()`, которая имеет два полинома аргумента (делимое и делитель) и возвращает также два полинома: частное и остаток от деления.

*Вычисление значения полинома в точке* обеспечивается функцией `polyval()`, которая имеет два аргумента: вектор коэффициентов полинома и точку, в которой он вычисляется. Вторым аргументом функции может быть матрицей. Тогда она возвратит матрицу той же размерности, компоненты которой будут равны значениям полинома в точках, определяемых компонентами второго аргумента. В общем случае аргументы могут принимать комплексные значения.

*Построение производной от полинома* выполняется функцией `polyder()` с одним аргументом.

*Формирование полинома по его корням* можно выполнить с использованием функции `poly` с единственным аргументом – вектором, содержащим заданные корни, в общем случае комплексные. Функция `poly` также позволяет определить характеристический полином квадратной матрицы, если она указана в качестве аргумента функции.

*Поиск корней полинома* реализуется с помощью функции `roots()`. Аргументом при ее вызове является полином, а найденные корни возвращаются в виде компонентов выходного вектора-столбца.

## Графика в системе MATLAB

Одно из достоинств системы MATLAB – большое количество графических средств, начиная от команд построения простых графиков функций одной переменной и кончая комбинированными и презентационными графиками с элементами анимации. Здесь будут рассмотрены простейшие функции.

Вывод графика на экран в декартовой системе осуществляется командой `plot`. Эта команда может использоваться в нескольких вариантах.

При вызове функции с двумя аргументами `plot(X, Y)`, где  $X$ ,  $Y$  – векторы одинаковой размерности, будет построен график зависимости  $Y(X)$  в отдельном специальном окне, называемом `figure`. В окне расположен ряд элементов, которые используются для размещения выводимых графических образов, а также ряд интерфейсных инструментов, с помощью которых можно манипулировать размещенным в окне изображением.

Вызов функции с несколькими параметрами в виде `plot(x1,y1,x2,y2, ...)` приведет к построению нескольких графиков в одних декартовых осях, каждый из которых задан вектором аргументов и вектором значений. При этом графики различаются по цветам согласно принятой по умолчанию последовательности: синий – зеленый – красный и т.д.

Общий формат функции `plot(x1, y1, S1, x2, y2, S2...)` позволяет управлять типом линии, ее цветом и типом маркера за счет символьной константы  $S$ . Возможные значения константы можно посмотреть в таблице с помощью команды `help plot`. Пример использования функции `plot` для управления графиком приведен ниже.

```
x1 = 0:0.01:10;  
x2 = 0:10;  
y1 = sin(x1);  
y2 = sin(x2);  
plot (x1,y1, 'r--', x2, y2, 'k*'), grid
```

В примере (рис. 3) первый график построен красным цветом и пунктирной линией, второй выводится в виде изолированных черных звездочек.

Далее рассмотрим ряд вспомогательных графических функций:

`grid (grid on| off)` – координатная сетка;

`figure (n)` – смена текущего графического окна,  $n$  – номер окна. Также данная функция может применяться для создания новых графических окон.



`hold (hold on| off)` – наложение графиков в осях текущего графического окна. Наложение будет производиться до тех пор, пока не встретится команда `hold off` или пользователь не сменит текущее графическое окно.

`title('текст')` – добавление заголовка к графику;

`xlabel('текст'), ylabel('текст')` – добавление подписей к осям;

`loglog()`, `semilogx()`, `semilogy()` – построение графиков в логарифмических или полулогарифмических осях. Функции используются вместо `plot` по аналогичным правилам.

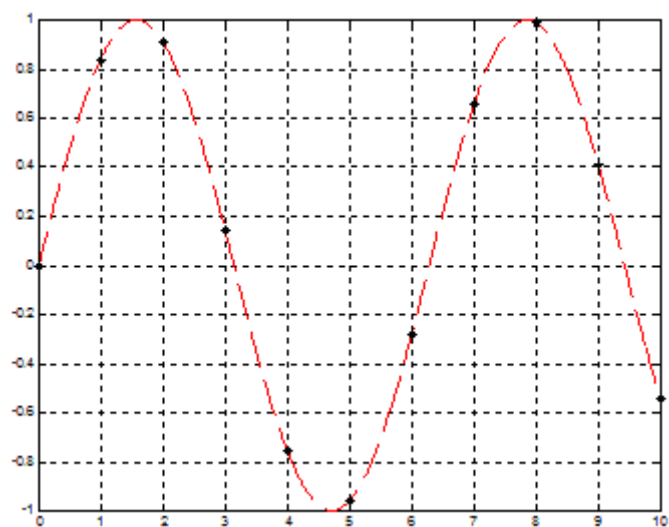


Рис. 3. Пример построения графиков функции `plot()`.

Для построения графиков в логарифмических и полулогарифмических осях имеет смысл задавать вектора в виде логарифмической, а не равномерно возрастающей последовательности. Это выполняется командой `logspace(x1, x2, N)`, где искомый вектор меняется в диапазоне от  $10^{x1}$  до  $10^{x2}$  и содержит  $N$  точек.

## Основные элементы языка программирования MATLAB

В языке программирования MATLAB чаще всего рассматриваются следующие элементы:

- константы и переменные;
- выражения;
- комментарии;
- операторы;
- скрипты (сценарии)
- функции пользователя.

Рассматриваемая система программирования является типичным интерпретатором. Это означает, что каждая инструкция (оператор или команда) программы распознается и тут же исполняется. При этом MATLAB не создает конечных исполняемых модулей. И, как следствие, для выполнения программ необходима среда MATLAB. Однако следует отметить, что для программ на языке MATLAB созданы компиляторы, транслирующие их в коды на языках программирования C и C++.

*Константы* в программах бывают трех типов:

- а) числовые (см. стр. 29);
- б) строковые – любая цепочка символов, заключенная в апострофы;
- в) логические – false и true, причем система понимает эти два зарезервированных слова как числа 1 и 0 соответственно.

*Переменные* в программах не подлежат предварительному описанию. Они всегда описываются по контексту и относятся к тому или иному типу данных автоматически в зависимости от присваиваемых ими значений. Каждая переменная имеет имя, присваиваемое программистом, которое, как и в командах, должно состоять из латинских букв, цифр и знака подчеркивания, причем первый символ – обязательно буква.

Перед использованием в правой части операторов присваивания имя соответствующей переменной должно быть известно в программе, точнее говоря, оно должно находиться в рабочем пространстве. Этого можно достичь двумя путями: или заданием значения новой переменной в командном окне или предварительным присваиванием ему значений в предшествующих операторах программного кода.

*Выражения.* Как и в любых других языках, выражения в программе на MATLAB – это имена переменных, констант и функций, объединенные знаками допустимых операций. В языке принято выделять следующие типы выражений:

1) Арифметические выражения. Формируются с использованием числовых переменных и констант и знаков арифметических операций тех же, что мы рассматривали для командной строки.

2) Строковые выражения. Строковые выражения допустимы только в двух вариантах: указание отдельной строковой константы или вызов функций, оперирующих со строками..

3) Выражения отношений. Эти выражения формируются с использованием следующих операций:

< меньше; <= меньше или равно; > больше;  
>= больше или равно; == равно; ~= не равно.

По отношению к однотипным массивам указанные операции применяются поэлементно.

4) Логические выражения. Формируются с использованием следующих операций:

& логическое «И»; | логическое «ИЛИ»; ~ логическое «НЕ»;

*Комментарии* в программном коде – это произвольные цепочки символов, расположенные после знака «%» до конца текущей строки.

*Основные операторы языка*

1. Оператор присваивания. Является простейшим оператором языка. Его смысл полностью эквивалентен аналогичным операторам таких языков, как Си и Паскаль, однако допускается при вызове функции присваивание значений нескольким переменных. Тогда они должны быть перечислены через запятую в квадратных скобках, например `[m, n] = size(A)`.

В зависимости от типа выражения в правой части, операторы присваивания делятся на арифметические, строковые, отношений и логические.

2. Условный оператор `if` в общем случае осуществляет проверку нескольких условий и записывается следующим образом:

```
if условие1
    операторы1
elseif условие2
    операторы2
elseif условие3
    операторы3
.....
else
    операторы4
end
```

Альтернативные ветви не являются обязательными частями оператора.

3. Оператор цикла типа `for ... end` используются для организации вычислений с заданным числом повторяющихся циклов. Конструкция такого оператора имеет вид:

```
for v=M
    операторы
end
```

Параметр **M** – это чаще всего вектор. В большинстве случаев он представляется с использованием оператора «:» в виде **s:d:e**, где **s** – начальное значение управляющей переменной цикла (**v**), **d** – приращение этой переменной, **e** – конечное значение управляющей переменной. На каждом шаге цикла переменная **v** последовательно принимает значения, соответствующие компонентам вектора **M**. Цикл выполняется до тех пор, пока **v<=d**.

Возможен также вариант, когда **M** не вектор, а матрица. Тогда цикл будет работать иначе: на каждом шаге управляющая переменная будет вектором, последовательно совпадающим со столбцами матрицы **M**. В этом случае в цикле будет столько шагов, сколько имеется столбцов в указанной матрице.

4. Оператор цикла типа **while** выполняется до тех пор, пока выполняется условие, указанное в заголовке:

```
while условие
    операторы
end
```

Если есть необходимость в досрочном прерывании выполнения цикла, используется оператор **break**.

#### *Скрипты и функции пользователя*

Все *m*-файлы, содержащие исходные коды на языке MATLAB, делятся на два типа: файлы-сценарии и файлы-функции.

Файлы-сценарии (или *script*-файлы) являются просто записью серии команд, которые могли бы быть набраны и выполнены в командном окне MATLAB. Главная их особенность состоит в том, что файлы-сценарии не имеют входных и выходных параметров. В них используются переменные из общего рабочего пространства MATLAB. В процессе выполнения они не компилируются в двоичный код. Обычно файл-сценарий имеет следующую структуру:

```
% Основной комментарий
```

```
% Дополнительный комментарий
```

```
Тело скрипта, состоящее из любой совокупности операторов.
```

Основным комментарием является первая строка текстовых комментариев, а дополнительным – последующие строки. Основной комментарий выводится при выполнении команд **lookfor** и «**help имя\_каталога**». Полный комментарий выводится при выполнении команды «**help имя\_файла**».

*M*-файл типа «функция» является типичным элементом языка программирования MATLAB. Файлы-функции обязательно начинаются с объявления

function, после которого указывается имя переменной (или имена нескольких переменных) – выходного параметра, имя самой функции и список ее входных параметров.

Все переменные, используемые в теле файла-функции, являются локальными, т.е. действуют только в пределах тела функции. При этом переменные общего рабочего пространства внутри функции не видны.

Структура файла-функции с одним выходным параметром выглядит следующим образом:

```
function var=f_name(список_параметров) ← заголовок
% Основной комментарий
% Дополнительный комментарий
Тело функции, состоящее из любой совокупности операторов
var=выражение
```

Замечание 1: оператор `var=выражение` (и сама переменная `var`) используются в тех случаях, когда требуется, чтобы функция возвращала некоторый результат.

Замечание 2: Если выходных параметров больше одного, то необходимо использовать конструкцию типа: `function [var1, var2, ...]=f_name(список_параметров)`. Если функция, имеющая несколько выходных параметров, входит в состав математического выражения, то для выполнения вычислений в выражении будет использоваться первый из выходных параметров. Как было сказано выше, в файлах-функциях используются локальные переменные. Но наряду с ними нередко возникает необходимость в использовании данных, находящихся в рабочем пространстве MATLAB, или передача данных из одной функции в другую не через выходные параметры. В этих случаях используется понятие глобальных переменных, описываемых командой

```
global var1 var2 ...;
```

Здесь переменные `var1` и `var2` определены как глобальные. Чтобы несколько программных модулей могли совместно использовать глобальную переменную, ее идентификатор должен быть объявлен как `global` во всех этих модулях. Если в функции должны использоваться общие переменные, их необходимо также объявить глобальными в функции.

## Основы моделирования динамических систем в среде SIMULINK

Как было отмечено выше, система SIMULINK предназначена для имитационного моделирования динамических систем. Процессы формирования моделей и моделирования носят интерактивный характер, т.е. пользователь может «на ходу» менять параметры модели и наблюдать за реакцией. Из SIMULINK имеется доступ ко всем средствам анализа пакета MATLAB. Эта возможность в совокупности с библиотекой блоков и возможностью создания своих библиотек делает SIMULINK одним из наиболее популярных средств моделирования динамических объектов.

В качестве иллюстративного примера на рисунке 4 изображены модель и результаты моделирования изменения синусоидального сигнала при прохождении через интегратор.

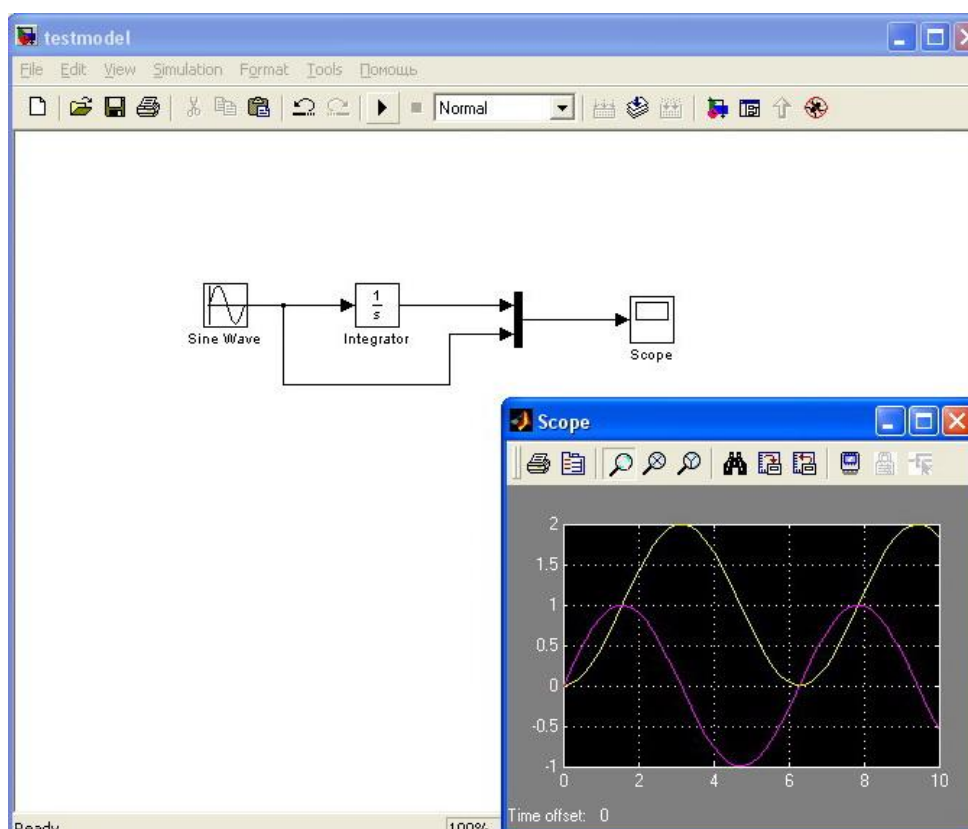


Рис. 4. Фрагмент рабочего окна SIMULINK

Библиотека блоков SIMULINK представляет собой набор визуальных объектов, используя которые можно собирать произвольную конструкцию. Блоки легко можно копировать и настраивать индивидуально (как правило, путем двойного нажатия мыши на изображении блока). Для удобства работы пользователя библиотека разбита на семь неизменяемых разделов:


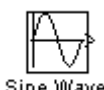





- Sources – источники сигналов;

- Sinks – получатели сигналов;
- Discrete – блоки моделирования дискретных систем;
- Continuous – блоки моделирования линейных систем;
- Discontinuous – блоки моделирования нелинейных систем;
- Math Operations – блоки математических операций;
- Signal Routing – блоки коммутации сигналов.

Также в состав SIMULINK входит множество специализированных библиотек блоков, предназначенных для конкретных инженерных задач.

Ниже в таблице 6 приводятся наиболее часто используемые для моделирования динамических систем блоки. Соединительные связи строятся при помощи мыши.

Таблица 6

Блок	Изображение	Библиотека	Настраиваемые параметры
Ступенчатое воздействие		Sources	Время включения; высота «ступеньки»
Гармоническое воздействие		Sources	Амплитуда, фаза, частота
Окно для временных диаграмм		Sinks	Масштаб, автомасштаб
Интегратор		Continuous	Начальное условие
Усилитель (пропорциональное звено)		Math operations	Коэффициент усиления
Сумматор (вычитатель)		Math operations	Количество входов, знаки входов
Мультиплексор		Signal Routing	Количество входов

Старт имитационного моделирования осуществляется через меню simulation, либо через соответствующую дублирующую пиктограмму. Также меню simulation позволяет задавать общие параметры моделирования, такие как время, шаг, метод интегрирования и пр.

## СПИСОК ЛИТЕРАТУРЫ

1. А.Н. Мирошников, С.Н. Румянцев. Моделирование систем управления технических средств транспорта. Учебное издание. ГЭТУ.- СПб.:Элмор, 1999.
2. Компьютерное моделирование систем управления движением морских подвижных объектов / Е.И. Веремей, В.М. Корчанов, М.В. Коровкин, С.В. Погожев. – СПб.: НИИ Химии СПбГУ, 2002. – 370 с.
3. Ю. Кетков, А. Кетков, М. Шульц. MATLAB 6.x: программирование численных методов. С-Пб.: БХВ-Петербург, 2004. – 690 с.
4. Моделирование электромеханических систем средствами MATLAB. Методические указания по лабораторным работам по дисциплине «Моделирование систем управления» / Сост.: О. Ю. Лукомская, А. Л. Стариченков, А. Г. Шпекторов. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2009. 40 с.

Редактор

---

Подписано в печать

Формат 60×84 1/16.

Бумага офсетная. Печать офсетная. Печ. л. .

Гарнитура «Times New Roman». Тираж экз. Заказ

---

Издательство СПбГЭТУ «ЛЭТИ»

197376, С.-Петербург, ул. Проф. Попова, 5