Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«Владимирский государственный университет
имени Александра Григорьевича и Николая Григорьевича Столетовых»
(ВлГУ)**

Факультет радиофизики, электроники и медицинской техники

Кафедра Электротехники и электроэнергетики

Шмелёв В.Е.

Математические задачи электроэнергетики

Конспект лекций

по дисциплине «Математические задачи электроэнергетики» для студентов ВлГУ, обучающихся по направлению 13.03.02 «Электроэнергетика и электротехника»

Владимир – 2015

Целями изучения дисциплины являются: изучение математического программного обеспечения ПК, которое очень удобно использовать в электротехнических и электроэнергетических расчётах широкого класса при практически неограниченных требуемых объёмах вычислений; формирование готовности участвовать в исследовании отдельных компонентов систем электроэнергетики и электротехники.

Необходимость в больших объёмах вычислений может возникать при проектировании систем электроснабжения с большим количеством потребителей электроэнергии, при проектировании электроэнергетических систем с большим количеством генерирующих мощностей, линий, подстанций и узлов нагрузки потребителей, при проектировании больших разветвлённых электрических сетей, при проведении поверочных расчётов, связанных с анализом конкретных установившихся и переходных режимов уже спроектированных объектов электротехники и электроэнергетики, при проведении статистических и оптимизационных технико-экономических расчётов этих объектов.

Представление однофазных электрических нагрузок в синусоидальном режиме импедансами или адмиттансами. Алгоритмическая реализация в MATLAB

Существует много способов расчётных представлений электрических нагрузок. Здесь рассмотрим способы представления детерминированных однофазных электрических нагрузок в синусоидальных режимах методами теории цепей в линеаризованных схемах замещения.

Если для активно- индуктивной однофазной нагрузки задано номинальное действующее значение напряжения $U_н$, номинальная активная мощность $P_н$, номинальный коэффициент мощности $\cos(\varphi_н)$, и нагрузка линейна по своим свойствам, то её можно представить комплексным сопротивлением (импедансом) или комплексной проводимостью (адмиттансом):

$$Y_н = \frac{P_н \cdot \left(1 - j\sqrt{\cos(\varphi_н)^{-2} - 1}\right)}{U_н^2}, \qquad (1)$$

где $\sqrt{\cos(\varphi_н)^{-2} - 1} = \mathrm{tg}(\varphi_н)$ – коэффициент реактивной мощности;

$\cos(\varphi_н) = \left(\mathrm{tg}(\varphi_н)^2 + 1\right)^{-0.5}$; $\sin(\varphi_н) = \left(\mathrm{tg}(\varphi_н)^{-2} + 1\right)^{-0.5}$,

импеданс нагрузки определяется выражением

$$Z_н = \frac{U_н^2}{P_н \cdot \left(1 - j\sqrt{\cos(\varphi_н)^{-2} - 1}\right)}. \qquad (2)$$

Если вместо номинальной активной мощности $P_н$ задана полная мощность $S_н$, то вместо формул (1) и (2) адмиттанс и импеданс нагрузки будут определять следующие фирмулы:

$$Y_н = \frac{S_н \cdot \left(\cos(\varphi_н) - j\sqrt{1-\cos(\varphi_н)^2} \right)}{U_н^2}, \tag{3}$$

$$Z_н = \frac{U_н^2}{S_н \cdot \left(\cos(\varphi_н) - j\sqrt{1-\cos(\varphi_н)^2} \right)}, \tag{4}$$

Если нагрузка активно- ёмкостная, то в формулах (3) и (4) слагаемое с множителем j (мнимая единица) будет со знаком плюс.

Если вместо $\cos(\varphi_н)$ задан $\mathrm{tg}(\varphi_н)$, то формулы (1) – (4) примут вид

$$Y_н = \frac{P_н \cdot (1 - j\,\mathrm{tg}(\varphi_н))}{U_н^2}, \tag{5}$$

$$Z_н = \frac{U_н^2}{P_н \cdot (1 - j\,\mathrm{tg}(\varphi_н))}, \tag{6}$$

$$Y_н = \frac{S_н \cdot \left(\left(\mathrm{tg}(\varphi_н)^2 + 1\right)^{-0.5} - j\left(\mathrm{tg}(\varphi_н)^{-2} + 1\right)^{-0.5} \right)}{U_н^2}, \tag{7}$$

$$Z_н = \frac{U_н^2}{S_н \cdot \left(\left(\mathrm{tg}(\varphi_н)^2 + 1\right)^{-0.5} - j\left(\mathrm{tg}(\varphi_н)^{-2} + 1\right)^{-0.5} \right)}, \tag{8}$$

Алгоритмическая реализация формул (1) – (8) в системе MATLAB выносится на самостоятельное изучение.

В случае жёсткого задания комплексной мощности нагрузки (независимо от фактического значения напряжения на её зажимах) уравнение, связывающее комплексное напряжение с комплексным током, является нелинейным: действующие значения тока и напряжения обратно пропорциональны друг другу, причём разность фаз между ними постоянна. В этом случае анализ режимов объектов электротехники и электроэнергетики может проводиться с помощью итерационных процедур, связанных с линеаризацией свойств нагрузки вблизи значений напряжения, получаемых на текущих итерационных шагах.

Разложение синусоидального режима работы трёхфазной цепи на симметричные составляющие при случайном задании её параметров. Алгоритмическая реализация в MATLAB

Из курса теоретических основ электротехники известно, что любой несимметричный режим работы трёхфазной цепи можно разложить на симметричные составляющие прямой, обратной и нулевой последовательности. Расчётные технологии, основанные на таком разложении, называют методом симметричных составляющих.

Несимметрия режимов может быть обусловлена несимметрией трёхфазной системы ЭДС и несимметрией распределения комплексных сопротивлений и проводимостей по фазам. В первом случае симметричные составляющие режима не связаны друг с другом и могут независимо рассчитываться по однолинейным схемам замещения без всяких допущений. Во втором случае симметричные составляющие режима связаны друг с другом, и их разделение может выполняться лишь приближённо, с допущениями. Но и в этом случае разложение режима на симметричные составляющие может быть полезно для определения показателей качества режима.

Несимметрия параметров цепи может носить случайный характер, значит, для решения некоторых статистических задач для задания этих параметров можно применять генераторы случайных чисел.

Рассмотрим две трёхфазные цепи, соединённые по схемам «звезда-звезда» и «треугольник-треугольник» (рис. 1 и 2).

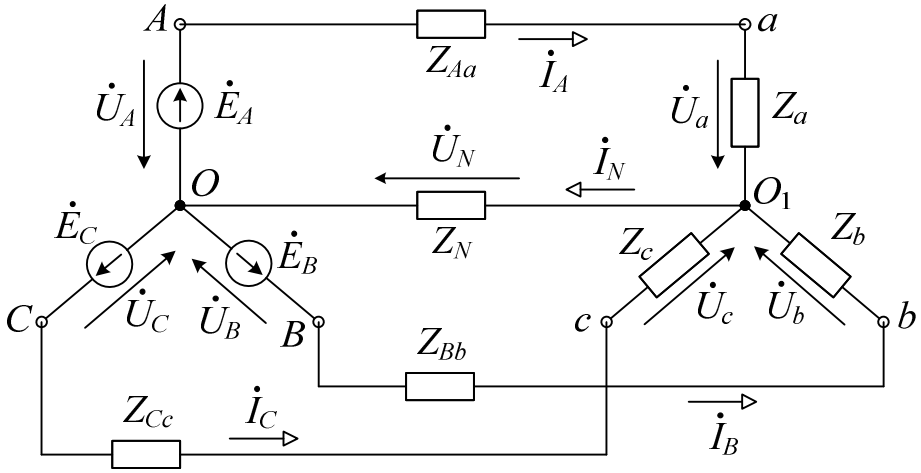

Рис. 1. Цепь «звезда-звезда»

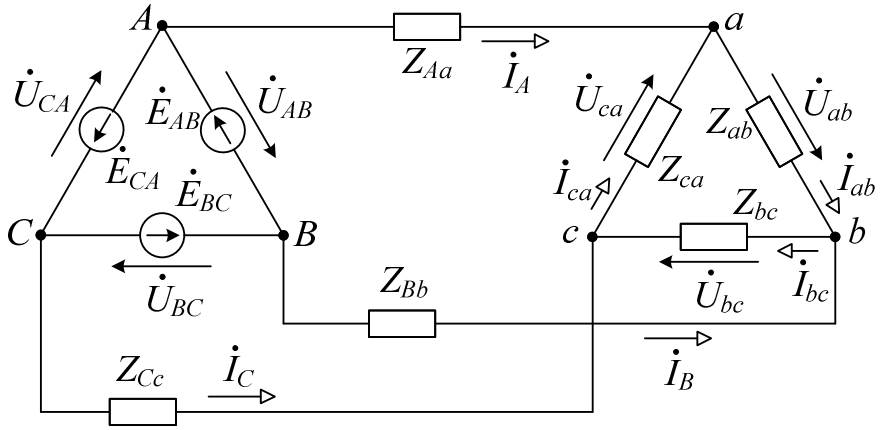

Рис. 2. Цепь «треугольник-треугольник»

Для простоты задачи зададим симметричную систему ЭДС, детерминированные значения сопротивлений подводящих проводов. Фазные нагрузки зададим случайным образом но-

минальными полными мощностями с заданным среднеквадратичным относительным отклонением (нормальный закон в логарифмическом масштабе), номинальными значениями разности фаз между напряжениями и токами с заданным среднеквадратическим отклонением (нормальный закон).

$$S_{Ан} = S_{Анн} \cdot \exp(\delta S_{Ан} \cdot \mathrm{randn}(1,1)), \quad \varphi_{Ан} = \varphi_{Анн} + \Delta\varphi_{Ан} \cdot \mathrm{randn}(1,1).$$

В двух других фазах «номинальные» полные мощности зададим аналогично.

Приведём пример сценария вместе с заданием исходных данных.

```
% Исходные данные:
Uf=220; % Действующее значение фазной ЭДС на стороне источника, В
Zp=0.2+0.05i; % Импедансы подводящих проводов, Ом
Slnn=[210;300;120]; % Полные nn мощности фаз нагрузки при номинальном напряжении, ВА
Flnn=[45;30;60]; % nn разности фаз напряжений и токов, град
dSnn=0.15; % Относительное СКО "номинальных" полных мощностей фаз
DFlnn=15; % СКО "номинальных" разностей фаз
Sn=Slnn.*exp(dSnn*randn).*exp(1i*(Flnn+DFlnn*randn)*pi/180); % "Ном" мощности фаз
[SE,SEall,Sp,SP,Slo,SLo,Isa,Usa]=sym_sos_zz(Uf,Zp,Sl,cl);

sl=sqrt(1-cl.^2).*sign(cl); % Коэффициенты реактивной мощности фаз нагрузки
% Комплексные мощности фаз нагрузки при номинальном напряжении, ВА:
SL=Sl.*complex(abs(cl),sl);
ZL=conj(Uf^2./SL); % Импедансы фаз нагрузки, Ом
% Пронумеруем ветви трёхфазной цепи.
% 1 - ветвь EA
% 2 - ветвь EB
% 3 - ветвь EC
% 4 - ветвь ZAa
% 5 - ветвь ZBb
% 6 - ветвь ZCc
% 7 - нейтральный провод ZN
% 8 - ветвь Za
% 9 - ветвь Zb
% 10 - ветвь Zc
Zv=[0;0;0;Zp;Zp;Zp;ZL]; % Диагональ импедансов ветвей
a=complex(-0.5,sqrt(0.75)); % Оператор поворота в симметричной трёхфазной системе
Ev=[Uf*[1;a^2;a];0;0;0;0;0;0;0];
% Матрица главных контуров трёхфазной цепи:
B=[1,0,0,1,0,0,1,1,0,0;0,1,0,0,1,0,1,0,1,0;0,0,1,0,0,1,1,0,0,1];
```

```
Zk=B*diag(Zv)*B.'; % Матрица контурных импедансов, Ом
Ek=B*Ev; % Матрица контурных ЭДС, В
Ik=Zk\Ek; % Комплексные токи в фазах нагрузки, А
SE=Ev(1:3).*conj(Ik); % Комплексные мощности источников, ВА
SEall=sum(SE); % Суммарная комплексная мощность, генерируемая источниками, ВА
IN=sum(Ik); % Комплексный ток в нейтральном проводе, А
% Комплексные мощности потерь в подводящих проводах, ВА:
Sp=conj([Ik;IN]).*[Ik;IN]*Zp;
SP=sum(Sp); % Суммарная комплексная мощность потерь в подводящих проводах, ВА
Ulo=Ik.*ZL; % Комплексные фазные напряжения нагрузки, В
Slo=conj(Ik).*Ulo; % Комплексные мощности фаз нагрузки, ВА
SLo=sum(Slo); % Суммарная комплексная мощность фаз нагрузки, ВА
% Симметричные составляющие тока для фазы a, А:
Isa=[1,a,a^2;1,a^2,a;1,1,1]*Ik/3;
% Симметричные составляющие напряжения для фазы a, В:
Usa=[1,a,a^2;1,a^2,a;1,1,1]*Ulo/3;
disp('Активные мощности, генерируемые источниками фазных ЭДС, Вт')
disp(mat2str(real(SE.'),5))
disp('Реактивные мощности, генерируемые источниками фазных ЭДС, ВАр')
disp(mat2str(imag(SE.'),5))
disp(['Суммарная активная мощность, генерируемая источниками ',num2str(real(SEall),'%0.7g'),'
Вт'])
disp(['Суммарная реактивная мощность, генерируемая источниками ',num2str(imag(SEall),'%0.7g'),'
ВАр'])
disp('Потери активной мощности в фазных проводах a, b, c, и нейтрали, Вт')
disp(mat2str(real(Sp.'),5))
disp('Потери реактивной мощности в фазных проводах a, b, c, и нейтрали, ВАр')
disp(mat2str(imag(Sp.'),5))
disp(['Суммарная активная мощность, теряемая в проводах ',num2str(real(SP),'%0.7g'),' Вт'])
disp(['Суммарная реактивная мощность, теряемая в проводах ',num2str(imag(SP),'%0.7g'),' ВАр'])
disp('Активные мощности фаз нагрузки, Вт')
disp(mat2str(real(Slo.'),5))
disp('Реактивные мощности фаз нагрузки, ВАр')
disp(mat2str(imag(Slo.'),5))
disp(['Суммарная активная мощность фаз нагрузки ',num2str(real(SLo),'%0.7g'),' Вт'])
disp(['Суммарная реактивная мощность фаз нагрузки ',num2str(imag(SLo),'%0.7g'),' ВАр'])
disp('Симметричные составляющие тока для фазы a, А')
disp(mat2str(Isa,5))
```

```
disp('Симметричные составляющие напряжения на нагрузке для фазы a, В')
disp(mat2str(Usa,5))
```

Вычислительные технологии разложения несинусоидальных периодических процессов на гармонические составляющие

Любая периодическая функция, имеющая на конечном интервале конечное число разрывов первого рода и конечное число максимумов и минимумов: $f(t)=f(t+n\cdot T)$, - может быть представлена в виде бесконечного тригонометрического (гармонического) ряда:

$$f(t)=A_0+\sum_{k=1}^{\infty}A_{km}\cdot\sin\left(k\omega t+\psi_k\right)=$$

$$=A_0+\sum_{k=1}^{\infty}A_{km}^{|}\cdot\sin\left(k\omega t\right)+\sum_{k=1}^{\infty}A_{km}^{||}\cdot\cos\left(k\omega t\right)=$$

$$=A_0+\frac{1}{2j}\cdot\sum_{k=1}^{\infty}\left(\dot{A}_{km}e^{jk\omega t}-\overset{*}{A}_{km}\,e^{-jk\omega t}\right)=$$

$$A_0+\frac{1}{j\sqrt{2}}\cdot\sum_{k=1}^{\infty}\left(\dot{A}_{k}e^{jk\omega t}-\overset{*}{A}_{k}\,e^{-jk\omega t}\right),$$

$$A_0=\frac{1}{T}\cdot\int_0^T f(t)\,\mathrm{d}t;\quad \dot{A}_{km}=\frac{2j}{T}\cdot\int_0^T f(t)\cdot e^{-jk\omega t}\,\mathrm{d}t;$$

$$A_{km}^{|}=\frac{2}{T}\cdot\int_0^T f(t)\cdot\sin\left(k\omega t\right)\mathrm{d}t=\frac{1}{\pi k}\cdot\int_0^{2\pi k} f(t)\cdot\sin\left(k\omega t\right)\mathrm{d}\left(k\omega t\right);$$

$$A_{km}^{||}=\frac{2}{T}\cdot\int_0^T f(t)\cdot\cos\left(k\omega t\right)\mathrm{d}t=\frac{1}{\pi k}\cdot\int_0^{2\pi k} f(t)\cdot\cos\left(k\omega t\right)\mathrm{d}\left(k\omega t\right);$$

Известно много приближённых методов разложения функций в гармонический ряд Фурье. Практически удаётся ограничить разложение в ряд несколькими гармониками, сумма которых приближённо отображает заданную функцию.

Если функция нечётная, т.е. $f(t)=-f(-t)$, то ряд Фурье состоит только из синусных составляющих, и мнимые части комплексных представлений всех гармоник равны нулю.

Если функция чётная, т.е. $f(t)=f(-t)$, то ряд Фурье состоит только из косинусных составляющих, и действительные части комплексных представлений всех гармоник равны нулю.

Если на полупериоде функции выполняется условие антипериодичности $f(t)=-f(t+T/2)$, то гармонический ряд состоит из гармоник только нечётного порядка.

Пример разложения периодического сигнала в ряд Фурье

Рассмотрим прямоугольное колебание $u(t)=U_m\cdot\mathrm{sign}(\sin(\omega t))$.

Такого вида сигнал иначе называют меандром.

В системе MATLAB график (рис.1) построим следующими операторами:

```
t=linspace(-1,2,4501);
u=sign(sin(t*pi*2));
plot(t,u)
```

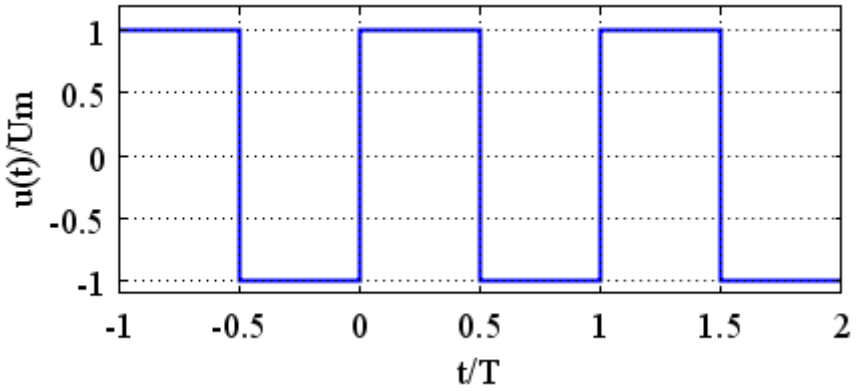

Рис. 1. Прямоугольное колебание - меандр

Постоянная составляющая: $U_0 = \frac{1}{T} \cdot \int_0^T u(t)\,\mathrm{d}t = 0$.

Комплексная амплитуда *k*-й гармоники:

$$\dot{U}_{km} = \frac{j}{\pi k} \int_0^{2\pi k} u(t) \cdot e^{-jk\omega t}\,\mathrm{d}(k\omega t) =$$

$$= \frac{jU_m}{\pi k} \cdot \left(\int_0^{\pi k} e^{-jk\omega t}\,\mathrm{d}(k\omega t) - \int_{\pi k}^{2\pi k} e^{-jk\omega t}\,\mathrm{d}(k\omega t) \right) =$$

$$= \frac{2U_m}{\pi k} \cdot \left(1 - (-1)^k\right).$$

Рассчитаем гармоники с номерами
k = от 1 до 21 во временн**о**й области:

```
t=linspace(-1,2,4501); % Моменты времени
u=sign(sin(t*pi*2)); % Сигнал (напряжение)
k=1:2:21; % Номера гармоник
Ukm=4/pi./k; % Комплексные амплитуды гармоник
ug=sin(k.'*t*pi*2);
ug=repmat(Ukm.',1,length(t)).*ug;
% ug - мгновенные значения всех гармоник
ugc=cumsum(ug);
plot(t,ugc([1 3 11],:))
```

В результате выполнения этой последовательности операторов будет построен график (временн**а**я диаграмма) (рис.2) трёх вариантов суммирования гармоник меандра во временн**о**й области.

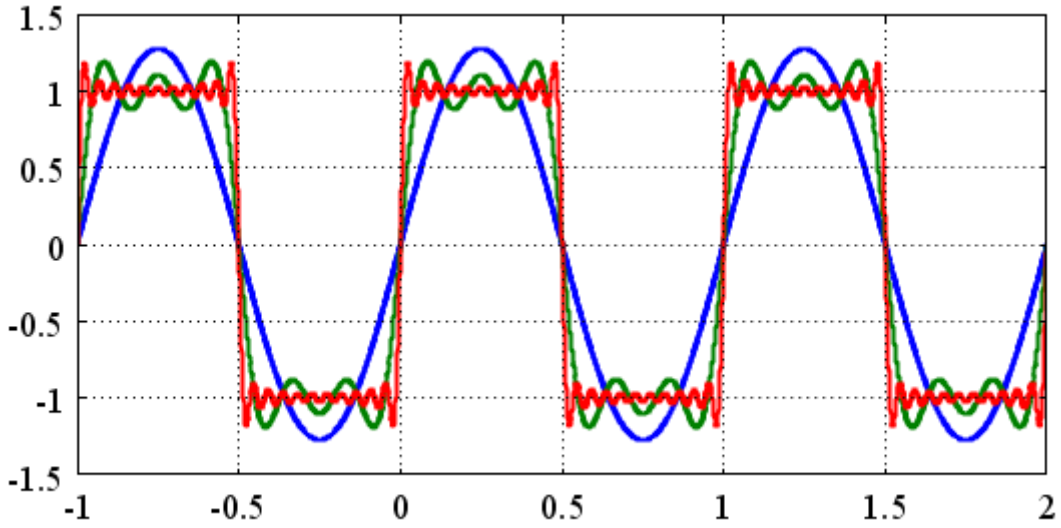

Рис. 2. Представление меандра в виде суммы нескольких гармоник во временн**о**й области: синий – только первая гармоника; зелёный – сумма гармоник 1,3,5; красный – сумма гармоник от 1 до 21.

Расчёт установившихся периодических процессов в линейных цепях

Разложение периодических функций на гармоники позволяет рассматривать действительный несинусоидальный установившийся режим в линейных цепях как совокупность взаимно налагающихся синусоидальных режимов кратных частот. Для каждой гармоники несинусоидальных источников ЭДС и тока должны быть известны: комплексное представление для амплитудного или действующего значения, а также частота $k\omega$ или номер гармоники k. Если цепь линейна, то действие каждой гармоники каждого из источников можно рассматривать отдельно. При этом для каждой гармоники источника цепь следует представить в виде эквивалентной схемы (схемы замещения) с соответствующими комплексными параметрами. Эквивалентные схемы, составленные для каждой из кратных частот, получаются при этом взаимно независимыми. В каждой схеме токи и напряжения имеют ту же частоту, что и все активные параметры – ЭДС и токи источников тока. Найденные при этом токи и напряжения определяют частичных рабочий режим цепи. Результирующие токи и напряжения для какого-либо участка цепи могут быть определены путём суммирования во временн**о**й области всех гармоник по принципу наложения.

Для каждого частичного синусоидального режима справедливы законы Ома и Кирхгофа в комплексно-матричной форме.

Таким образом, расчёт несинусоидальных режимов для линейных цепей сводится к расчёту ряда синусоидальных режимов кратных частот и суммированию гармоник.

Пример расчёта периодического несинусоидального режима

Рассмотрим последовательную *RC*-цепочку (рис.3), питаемую от идеального источника ЭДС в форме меандра с амплитудой 1В, частотой 1 рад/мс и нулевой начальной фазой: $e(t)=\mathrm{sign}(\sin(t))$. Здесь время измеряется в миллисекундах.

Пусть C = 1мкФ, R = 1 кОм.

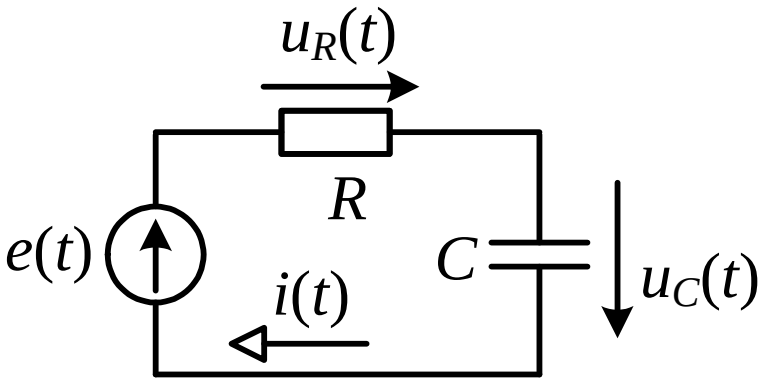

Рис. 3. Схема анализируемой цепи

На *k*-й гармонике тока и напряжения комплексное сопротивление контура цепи равно

$$Z_k = R + \left(jk\omega C \right)^{-1}.$$

Комплексная амплитуда *k*-й гармоники тока равна

$$\dot{I}_{km} = \frac{\dot{E}_{km}}{Z_k} = \frac{2E_m}{\pi k} \cdot \left(1 - (-1)^k\right) \cdot \frac{jk\omega C}{1 + jk\omega RC};$$

$$\dot{U}_{Ckm} = \frac{2E_m}{\pi k \cdot \left(1 + jk\omega RC\right)} \cdot \left(1 - (-1)^k\right).$$

Последняя формула определяет комплексную амплитуду *k*-й гармоники напряжения на конденсаторе. Выполним следующую последовательность операторов:

```
t=linspace(-1,2,4501)*pi*2; % Моменты времени
k=1:2:21; % Номера гармоник
Ekm=4/pi./k; % Компл. амплитуды гармоник ЭДС
Zc=(1i*k).^(-1);
R=1;
Ikm=Ekm./(R+Zc);
Uckm=Ikm.*Zc;
ig=repmat(real(Ikm).',1,length(t)).*sin(k.'*t);
ig=ig+repmat(imag(Ikm).',1,length(t)).*cos(k.'*t);
ucg=repmat(real(Uckm).',1,length(t)).*sin(k.'*t);
ucg=ucg+repmat(imag(Uckm).',1,length(t)).*cos(k.'*t);
ii=sum(ig);
uc=sum(ucg);
plot(t,ii)
figure
plot(t,uc)
Ikm.', Uckm.'
```

В результате выполнения этой последовательности операторов получим графики тока $i(t)$ и напряжения на конденсаторе $u_C(t)$ в двух фигурах (рис.4,5), а также значения комплексных амплитуд гармоник этих величин в командном окне.

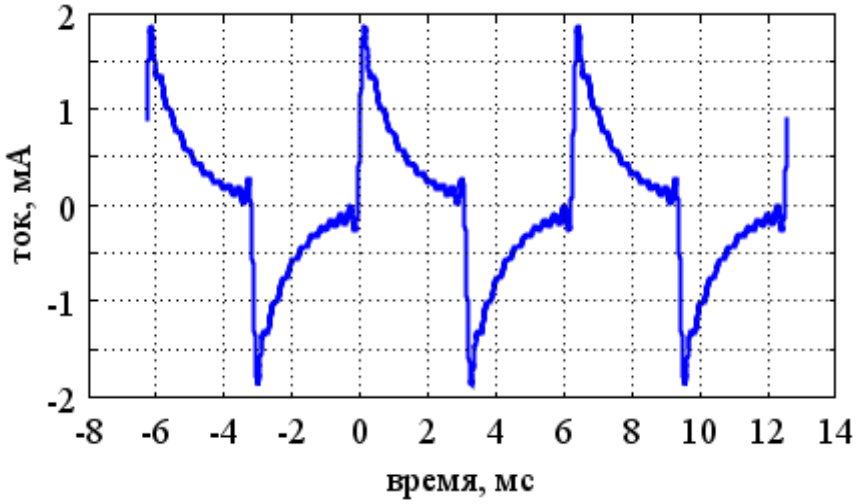

Рис. 4. Рассчитанная осциллограмма тока, представленная суммой гармоник с номерами от 1 до 21.

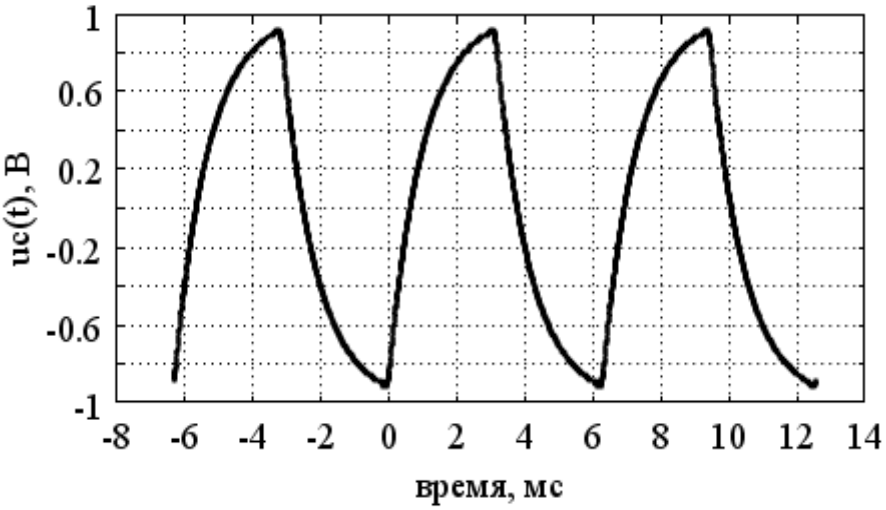

Рис. 5. Рассчитанная осциллограмма напряжения на конденсаторе, представленная суммой гармоник с номерами от 1 до 21.

Комплексные амплитуды нечётных гармоник тока:

```
ans =
   0.63661977236758 +   0.63661977236758i
   0.38197186342055 +   0.12732395447352i
   0.24485375860292 +   0.04897075172058i
   0.17825353626292 +   0.02546479089470i
   0.13974580369044 +   0.01552731152116i
   0.11480028682038 +   0.01043638971094i
   0.09736537695034 +   0.00748964438080i
   0.08450704942932 +   0.00563380329529i
   0.07463818020861 +   0.00439048118874i
   0.06682748991704 +   0.00351723631142i
   0.06049328153719 +   0.00288063245415i
```

Комплексные амплитуды нечётных гармоник напряжения на конденсаторе:

```
ans =
   0.63661977236758 -   0.63661977236758i
   0.04244131815784 -   0.12732395447352i
```

0.00979415034412 - 0.04897075172058i
0.00363782727067 - 0.02546479089470i
0.00172525683568 - 0.01552731152116i
9.48762700995e-004 - 0.01043638971094i
5.76126490830e-004 - 0.00748964438080i
3.75586886353e-004 - 0.00563380329529i
2.58263599338e-004 - 0.00439048118874i
1.85117700601e-004 - 0.00351723631142i
1.37172974007e-004 - 0.00288063245415i

Действующие значения несинусоидальных токов и напряжений

Действующие значения периодических токов и напряжений определяют по формуле:

$$I=\left(\frac{1}{T}\cdot\int_0^T\left(i(t)\right)^2\mathrm{d}t\right)^{0.5};\qquad U=\left(\frac{1}{T}\cdot\int_0^T\left(u(t)\right)^2\mathrm{d}t\right)^{0.5}.$$

$$I=\left(\frac{1}{T}\cdot\int_0^T\left(I_0+\frac{1}{j\sqrt{2}}\cdot\sum_{k=1}^{\infty}\left(\dot{I}_k\cdot e^{jk\omega t}-\overset{*}{\dot{I}}_k\cdot e^{-jk\omega t}\right)\right)^2\mathrm{d}t\right)^{0.5}=$$

$$=\left(\frac{1}{T}\cdot\int_0^T I_0^2\,\mathrm{d}t+\sum_{k=1}^{\infty}\frac{1}{T}\cdot\int_0^T\left(\dot{I}_k\cdot e^{jk\omega t}\cdot\overset{*}{\dot{I}}_k\cdot e^{-jk\omega t}\right)\mathrm{d}t\right)^{0.5}=\left(\sum_{k=0}^{\infty}I_k^2\right)^{0.5};\quad U=\left(\sum_{k=0}^{\infty}U_k^2\right)^{0.5}.$$

Действующее значение несинусоидального периодического тока или напряжения равно квадратному корню из суммы квадратов действующих значений всех гармоник.

Коэффициенты, характеризующие форму электрических сигналов

Коэффициент формы: $k_{\text{ф}}=\dfrac{A}{A_{\text{ср}}}$, где $A=\left(\dfrac{1}{T}\cdot\int_0^T\left(a(t)\right)^2\mathrm{d}t\right)^{0.5}$, $A_{\text{ср}}=\dfrac{1}{T}\cdot\int_0^T\left|a(t)\right|\mathrm{d}t.$

Для синусоидального сигнала $k_{\text{фsin}}=\dfrac{\pi}{2\sqrt{2}}\approx 1.11$

Коэффициентом амплитуды называют отношение максимального значения сигнала $A_{\max}$ к его действующему значению A:

$$k_{\text{а}}=\frac{A_{\max}}{A}.\qquad \text{Для синусоидального сигнала}\quad k_{\text{аsin}}=\sqrt{2}\approx 1.414.$$

Для сигналов, не имеющих постоянной составляющей, определяют ещё два коэффициента.

$$\text{Коэффициент искажения}\quad k_{\text{и}}=\frac{A_1}{A}.\qquad \text{Для синусоидального сигнала}\quad k_{\text{исin}}=1.$$

Коэффициент гармоник – это отношение действующего значения высших гармоник к действующему значению первой (основной) гармоники:

$$k_{\text{г}} = \frac{\sqrt{A^2 - A_1^2}}{A_1} = \sqrt{\left(\frac{A}{A_1}\right)^2 - 1} = \sqrt{k_{\text{и}}^{-2} - 1}.$$

Существует и другой вариант коэффициента гармоник, определяемый как отношение действующего значения высших гармоник к действующему значению сигнала:

$$k_{\text{г}}^{|} = \frac{\sqrt{A^2 - A_1^2}}{A} = \sqrt{1 - \left(\frac{A_1}{A}\right)^2} = \sqrt{1 - k_{\text{и}}^2}.$$

Для синусоидального сигнала $k_{\text{г}\sin} = k_{\text{г}\sin}^{|} = 0.$

Названные коэффициенты применяются в измерительной и преобразовательной технике. Коэффициент гармоник является также одним из показателей качества электроэнергии, доводимой до потребителей.

Мощность при несинусоидальных токах и напряжениях

Если напряжение и ток разложить в ряд Фурье, то мгновенная мощность будет определяться формулой: $p(t) = u(t) \cdot i(t) =$

$$= \left(U_0 + \frac{1}{j\sqrt{2}} \cdot \sum_{k=1}^{\infty} \left(\dot{U}_k \cdot e^{jk\omega t} - \overset{*}{\dot{U}}_k \cdot e^{-jk\omega t}\right)\right) \cdot \left(I_0 + \frac{1}{j\sqrt{2}} \cdot \sum_{q=1}^{\infty} \left(\dot{I}_q \cdot e^{jq\omega t} - \overset{*}{\dot{I}}_q \cdot e^{-jq\omega t}\right)\right) =$$

$$= U_0 \cdot I_0 + \frac{U_0}{j\sqrt{2}} \cdot \sum_{q=1}^{\infty} \left(\dot{I}_q \cdot e^{jq\omega t} - \overset{*}{\dot{I}}_q \cdot e^{-jq\omega t}\right) + \frac{I_0}{j\sqrt{2}} \cdot \sum_{k=1}^{\infty} \left(\dot{U}_k \cdot e^{jk\omega t} - \overset{*}{\dot{U}}_k \cdot e^{-jk\omega t}\right) -$$

$$- \frac{1}{2} \sum_{k=1}^{\infty} \sum_{q=1}^{\infty} \left(\dot{U}_k \cdot e^{jk\omega t} - \overset{*}{\dot{U}}_k \cdot e^{-jk\omega t}\right) \cdot \left(\dot{I}_q \cdot e^{jq\omega t} - \overset{*}{\dot{I}}_q \cdot e^{-jq\omega t}\right).$$

Активная мощность $P = \frac{1}{T} \cdot \int_0^T p(t)\,\mathrm{d}t$:

$$P = U_0 \cdot I_0 + \frac{1}{2} \cdot \sum_{k=1}^{\infty} \left(\dot{U}_k \cdot \overset{*}{\dot{I}}_k + \overset{*}{\dot{U}}_k \cdot \dot{I}_k\right) =$$

$$= U_0 \cdot I_0 + \sum_{k=1}^{\infty} \mathrm{Re}\left(\dot{U}_k \cdot \overset{*}{\dot{I}}_k\right) =$$

$$= U_0 \cdot I_0 + \sum_{k=1}^{\infty} U_k \cdot I_k \cdot \cos\left(\varphi_k\right) = \sum_{k=0}^{\infty} P_k .$$

Активная мощность, потребляемая участком цепи при несинусоидальных периодических токах и напряжениях, равна сумме активных мощностей всех гармоник.

В пределах каждой гармоники используют понятие полной и комплексной мощности так же как и при анализе цепей синусоидального тока. Здесь выполняется условие баланса комплексных мощностей.

Баланс активных мощностей справедлив также для всей совокупности гармоник.

Суммирование реактивных мощностей различных гармоник не имеет смысла.

В преобразовательной технике допускается формальное суммирование реактивных мощностей различных гармоник. Такую сумму называют реактивной мощностью несинусоидального периодического режима. В этом случае считается, что полная мощность несинусоидального режима, равная произведению действующих значений напряжения и тока, состоит из трёх ортогональных составляющих:

$$S = U \cdot I = \sqrt{P^2 + Q^2 + N^2}, \qquad (1)$$

где P – активная мощность, Q – реактивная мощность, N – мощность искажения.
В преобразовательной технике и в технике электропривода реактивную мощность иначе называют *мощностью сдвига*.
В электротехнической литературе встречается также понятие реактивной мощности, основанное на следующем соотношении:

$$Q^{|} = \frac{1}{2\pi} \cdot \int_0^T u(t) \cdot \frac{\mathrm{d}\, i(t)}{\mathrm{d}\, t} \mathrm{d}\, t \qquad (2)$$

Реактивная мощность, определяемая формулой (2) отличается от реактивной мощности, входящей в (1) тем, что

$$Q = \sum_{k=1}^{\infty} Q_k\ , \qquad Q^{|} = \sum_{k=1}^{\infty} k \cdot Q_k\ .$$

Отношение активной мощности к полной называется *коэффициентом мощности*:
$\chi = P/S$.

Краткая характеристика программного обеспечения (ПО) персональных компьютеров (ПК)

Всё очень разнообразное ПО ПК можно разделить на две очень большие группы: 1. Системное ПО, 2. Прикладное ПО.

1. Системное ПО

К системному ПО относятся операционные системы (ОС) и системно-независимые программы.

Операционной системой называется комплекс программ, получающий управление в результате начальной загрузки компьютера и управляющий процессом взаимодействия пользователя (оператора) с оборудованием ПК, работающими прикладными программами и задачами.

В настоящее время получили большое распространение ОС двух больших семейств: Windows и Linux. В системах Windows имеется очень развитой графический интерфейс пользователя (GUI), поэтому редко возникает необходимость использования командной строки и консольных команд для запуска прикладных программ и выполнения настроечных и служебных действий. Достоинство – простота работы пользователя. В народе говорят, что Windows – ОС для домохозяек. Недостаток этого семейства ОС – в процессе работы постоянно занят большой объём оперативной памяти (реальной и виртуальной) и дискового пространства для временных файлов.

В системах Linux основным режимом является режим командной строки (консольный режим). GUI реализован в графических оболочках (обычно это X-Window). Для этих ОС характерна высокая экономичность использования вычислительных ресурсов, реальная мультизадачность и многопользовательский режим работы ПК. Существует много бес-

платных версий этих ОС. Как правило, работа пользователя требует серьёзной подготовки, поэтому такие ОС менее популярны, чем Windows.

Системно-независимыми называют программы, предназначенные для выполнения конкретных действий, и получающие управление в результате начальной загрузки ПК. Завершение работы таких программ приводит к остановке ПК, и если надо продолжить работу, то необходимо опять провести начальную загрузку. Обычные пользователи очень редко применяют такие программы. Системно-независимые программы обычно используются для тестирования основного и периферийного оборудования, а также для разметки и форматирования томов данных, если операционная система ещё не установлена.

2. Прикладное ПО

Прикладными называются программы которые могут выполняться только под управлением операционных систем. По функциональному назначению прикладное ПО условно можно разделить на несколько больших групп: служебные программы (утилиты), офисное ПО, игры и развлечения, издательские системы, программы разработки проектно-сметной документации и САПР, специализированное и профессиональное ПО, математическое ПО, средства разработки прикладного ПО.

2.1. Утилиты выполняют служебные функции: тестирование оборудования, разметка и форматирование томов данных, файл-менеджеры, архиваторы, просмотрщики и редакторы файлов простейших форматов и др.

2.2. Офисное ПО – редакторы документов общего и офисного назначения, имеющих достаточно сложную структуру. Сюда можно отнести программный комплекс Microsoft Office, графические редакторы и просмотрщики общего назначения (Adobe Acrobat, Adobe Photo Shop, пакет Corel, ACD See и др.).

2.4. Издательские системы – редакторы документов сложной структуры, предназначенных для выпуска полиграфической продукции (газет, журналов, книг и др.).

2.5. Сюда можно отнести AutoCAD, КОМПАС, Pro Ingener, P-CAD, P-Spice и др. CAD-системы.

2.6. Сюда можно отнести программы фирмы 1С: бухгалтерия, предприятие и др.

2.7. Программы для выполнения сложных числовых и символьных вычислений без разработки собственных прикладных программ. Сюда можно отнести MathCAD, MATLAB, Mathematica, Statistika, Maple, программы структурного моделирования (LabView), программы решения задач математической физики и расчёта физических полей (ANSYS, Elcut, JUMP, COMSOL Multiphysics, LS-DYNA и др.).

2.8. Сюда можно отнести компиляторы, компоновщики задач, отладчики вместе с редакторами исходных текстов программ. Сюда можно отнести Delphi, C++, ассемблеры…

Между перечисленными группами нет чётких границ: одну и ту же программу можно отнести к разным группам.

В рамках данной дисциплины мы будем изучать основы программных комплексов MATLAB и COMSOL Multiphysics.

Система инженерных и научных расчётов MATLAB

MATLAB – это система инженерных и научных расчётов, базирующаяся на матричных вычислениях. Разработчик – фирма MathWorks (США).

Основным объектом арифметических операций является числовая матрица. Поддерживаются также и другие типы данных (структуры, массивы ячеек, типы объектно- ориентированного программирования). Матрица является частным случаем массива.

В технике любого программирования *массивом* называют упорядоченное множество, т.е. такое множество, каждый элемент которого пронумерован. Если каждому элементу множества соответствует только один уникальный номер, то такой массив называют *одномерным*. Если каждому элементу множества соответствует уникальная пара номеров, то такой массив называют *двумерным*. Номера элементов массива называют *индексами*. Двумерные массивы иначе называют матрицами. Одномерные массивы можно интерпретировать как матрицы-строки и матрицы-столбцы. Если каждому элементу множества соответствует уникальная тройка номеров, то такой массив называют трёхмерным. Существуют в общем случае многомерные массивы, которые в вычислительных алгоритмах можно интерпретировать как *суперматрицы*. Если каждым элементом массива является число, то такой массив называют *числовым*. Для числовых матриц в математике определены арифметические операции: сложение, вычитание, умножение, деление и др. Эти операции естественным образом поддерживаются системой MATLAB. Их мы рассмотрим позже.

В системе MATLAB имеется свой язык программирования. Каждый оператор должен выполняться непосредственно из командного окна или автоматически после запуска программы (скрипт-файла). Система MATLAB является интерпретирующей, но не компилирующей, но скорость выполнения программы незначительно меньше скорости выполнения откомпилированных программ.

Важнейшей отличительной особенностью MATLAB по сравнению с остальным математическим ПО является высокая эффективность использования математического сопроцессора ПК, выполняющего арифметические операции с плавающей точкой.

Операционная среда MATLAB

MATLAB запускается как обычное приложение Windows с графическим интерфейсом пользователя (GUI). Коротко рассмотрим этот интерфейс. Более подробно он описан, например, в [1].

В окне MATLAB есть свой рабочий стол (Desktop). На рабочем столе находится строка главного меню, панель инструментов, командное окно, окно истории команд, окно отображения текущей директории, окно отображения переменных рабочей области (Workspace) и другие окна. Эти окна могут быть открыты по мере необходимости. В системе MATLAB все действия происходят в результате выполнения команд меню и панели инструментов, а также при выполнении операторов MATLAB. В командном окне имеется командная строка и область показа результатов выполнения операторов MATLAB. Операторы выполняются сразу после ввода в командную строку или автоматически из m-файла после его запуска. Содержимое m-файла условно можно назвать программой (исходным текстом). В окне Workspace обычно показываются имена переменных, значения, типы, объём памяти и размеры. Рабочей областью (Workspace) называется область памяти ПК, где размещены переменные MATLAB. Переменной MATLAB называется объект данных в рабочей области, имеющий собственное имя в соответствии с правилами синтаксиса языка программирования. Имя переменной – последовательность латинских букв и цифр, начинающаяся с буквы. В версии 6.5 длина имени переменной не должна превышать 63 символа [2].

Система MATLAB логически состоит из ядра, пакетов расширения и подсистем структурного моделирования Simulink, Stateflow и др. Ядро, пакеты расширения и подсистемы представляют собой отдельные лицензируемые единицы программного продукта. В ядро входят программы, поддерживающие операционную и вычислительную среду MATLAB, а также его встроенные и основные функции. Пакеты расширения (Toolboxes) представ-

ляют собой пакеты m-, p-, mex- и др. файлов для решения задач в конкретных областях математики и техники. Эти файлы выполняются под управлением ядра MATLAB. Подсистемы Simulink, Stateflow и др. также состоят из ядра и библиотек блоков (Blocksets) структурных и виртуальных моделей. Состав программного пакета MATLAB более подробно описан в многочисленной литературе, например, в [1, 2].

Типы данных в системе MATLAB

Все переменные, хранящиеся в рабочей области MATLAB, являются массивами. Массив является основным типом данных. Все другие типы являются его подтипами. Структуру подчинённости типов данных изобразим в виде диаграммы [1] (рис. 1).

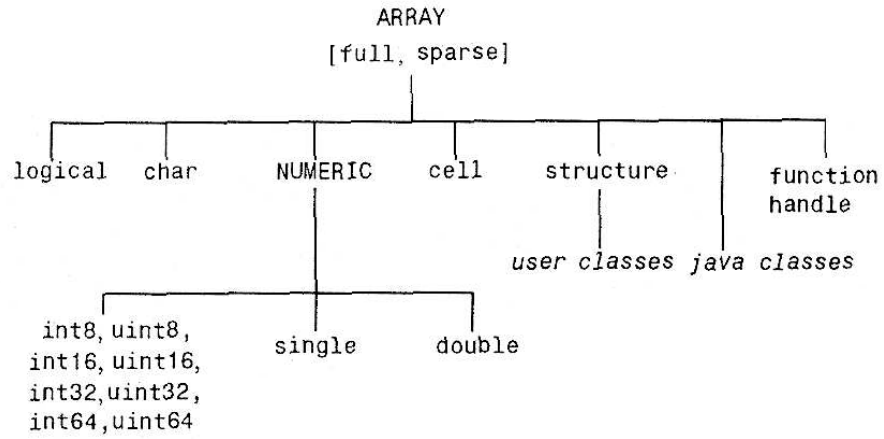

Рис. 1. Типы данных

ARRAY – массив. В квадратных скобках (рис. 1) показаны возможные атрибуты этого типа данных. **full** – полный массив (в памяти ЭВМ хранятся все элементы массива). **sparse** – разреженный массив, или иначе разреженная матрица (в памяти ЭВМ хранятся только ненулевые элементы массива и такие элементы, которые при дальнейших вычислениях могут стать ненулевыми). При реализации многих вычислительных методов разреженные матрицы могут давать существенную экономию оперативной памяти ЭВМ.

logical – логический тип. Каждый элемент такого массива может принимать только одно из двух возможных значений: логический нуль (false) или логическая единица (true).

char – массив текстовых символов (букв, цифр и др.). Обычно этот тип данных используется для формирования различных сообщений и текстовых пояснений в выходных текстовых файлах.

NUMERIC – числовой массив.

int8 – числовой массив, каждый элемент которого представляет собой целое число со знаком, занимающее в памяти ЭВМ 8 бит (1 байт). Диапазон возможных значений – от -128 до +127.

uint8 – числовой массив, каждый элемент которого представляет собой целое число без знака, занимающее в памяти ЭВМ 8 бит (1 байт). Диапазон возможных значений – от 0 до 255.

int16 – числовой массив, каждый элемент которого представляет собой целое число со знаком, занимающее в памяти ЭВМ 16 бит (2 байта). Диапазон возможных значений – от -32768 до +32767.

uint16 – числовой массив, каждый элемент которого представляет собой целое число без знака, занимающее в памяти ЭВМ 16 бит (2 байта). Диапазон возможных значений – от 0 до 65535.

int32 – числовой массив, каждый элемент которого представляет собой целое число со знаком, занимающее в памяти ЭВМ 32 бит (4 байта). Диапазон возможных значений – от -2147483648 до +2147483647.
uint32 – числовой массив, каждый элемент которого представляет собой целое число без знака, занимающее в памяти ЭВМ 32 бит (4 байта). Диапазон возможных значений – от 0 до 4294967295.
int64 – числовой массив, каждый элемент которого представляет собой целое число со знаком, занимающее в памяти ЭВМ 64 бит (8 байтов). Диапазон возможных значений – от -9223372036854775808 до +9223372036854775807.
uint64 – числовой массив, каждый элемент которого представляет собой целое число без знака, занимающее в памяти ЭВМ 64 бит (8 байтов). Диапазон возможных значений – от 0 до 18446744073709551615.
Эти типы данных называются целочисленными с фиксированной точкой. Начиная с версии 6.5, для этих типов определены арифметические операции. До версии 6.5 эти типы использовались только для кодирования графической информации и не могли применяться для вычислений.
single – числовой массив, каждый элемент которого представляет собой действительное или комплексное число с плавающей точкой. Если все элементы – действительные числа, то массив имеет атрибут real, и каждый элемент занимает в памяти ЭВМ 4 байта. Если все элементы – комплексные числа, то массив имеет атрибут complex, и каждый элемент занимает в памяти ЭВМ 8 байтов.
double – числовой массив, каждый элемент которого представляет собой действительное или комплексное число с плавающей точкой. Если все элементы – действительные числа, то массив имеет атрибут real, и каждый элемент занимает в памяти ЭВМ 8 байтов. Если все элементы – комплексные числа, то массив имеет атрибут complex, и каждый элемент занимает в памяти ЭВМ 16 байтов. Этот тип данных является основным для системы MATLAB. По умолчанию все числовые константы считаются принадлежащими этому типу данных. Комплексные числа представляются в памяти ЭВМ двумя действительными числами: действительной частью и мнимой частью.
Машинное представление действительных чисел с плавающей точкой на ПК и диапазоны их возможных значений описаны в [2]. Тип single применяется, когда не требуется высокая точность представления чисел с плавающей точкой. В этом случае для хранения массива требуется оперативной памяти в два раза меньше, чем при использовании типа double.
cell – массив ячеек. Элементы такого массива могут иметь разные типы. С математической точки зрения такой массив можно считать гиперматрицей. Этот тип полезен для организации сложных вычислений.
structure – массив, каждый элемент которого является структурой. Каждая структура имеет именованные поля, значения которых могут принадлежать любому из описанных типов данных. Именованные поля могут иметь значения, принадлежащие разным типам данных. В других языках программирования такие структуры называются записями.
user classes – типы данных объектно-ориентированного программирования, определяемые пользователями и создаваемые на шаблонах, принадлежащих типу **structure**.
Остальные типы данных в данной дисциплине рассматриваться не будут.

Константы языка MATLAB

В языке программирования MATLAB предусмотрены константы типов **char** и **double**.

Константа типа **char** представляет собой последовательность текстовых символов, заключённых в апострофы. Например, значением константы

'word'

является массив-строка, первым элементом которого является символ (в данном случае – буква) w , вторым элементом – буква o , третьим элементом – буква r , четвёртым – буква d .

Действительная константа типа double в общем случае имеет следующий синтаксис.

±пц1.**пц2E±пц3**

Здесь первый **±** – знак представляемого действительного числа: - (минус), + (плюс) или отсутствие символа. Минус соответствует отрицательному числу, плюс или отсутствие символа – положительному числу. **пц1** – последовательность десятичных цифр, кодирующая целую часть дробного числа, умножаемого на порядковый множитель. . – символ «точка», называемый десятичной точкой. **пц2** – последовательность десятичных цифр, кодирующая дробную часть числа, умножаемого на порядковый множитель. Обязательно должна быть записана либо целая, либо дробная часть этого дробного числа. Если дробной части нет, то десятичная точка может отсутствовать. **E** – буква E или e, выполняющая роль начала кода порядкового множителя. **±пц3** – десятичный код целого показателя степени числа 10 в порядковом множителе. **±** – знак показателя степени: - (минус), + (плюс) или отсутствие символа. Минус соответствует отрицательному показателю, плюс или отсутствие символа – положительному показателю. **пц3** – последовательность десятичных цифр (не более трёх), кодирующая абсолютную величину показателя степени. Если порядкового множителя нет, то синтаксическая конструкция **E±пц3** может отсутствовать. В этом случае показатель степени принимается равным нулю.

Комплексная константа типа double в общем случае имеет следующий синтаксис.

±пц1.**пц2E±пц3±пц4**.**пц5E±пц6i**

Здесь **пц4**, **пц5**, **пц6** имеют тот же смысл, что и **пц1**, **пц2**, **пц3**, но относятся к мнимой части представляемого комплексного числа. **i** – буква i или j, указывающая на то, что второе действительное число кодирует мнимую часть. Если действительная часть равна нулю, то синтаксическая конструкция **±пц1**.**пц2E±пц3** может отсутствовать. В этом случае синтаксический элемент ± перед **пц4** может отсутствовать, если мнимая часть положительная.

Приведём некоторые простейшие примеры применения констант типа **double**.

Оператор

a1=31

приводит к присваиванию переменной с именем a1 действительного числа 31.

Оператор

a1=3.1E1

приводит к присваиванию переменной с именем a1 действительного числа $31=3.1\cdot10^{1}$.

Оператор

a1=3.1E-1

приводит к присваиванию переменной с именем a1 действительного числа $0.31=3.1\cdot10^{-1}$.

Оператор

a1=–3+4i

приводит к присваиванию переменной с именем a1 комплексного числа $-3+j4$.

Оператор

a1=-3.1E-2+4.2E+2i

приводит к присваиванию переменной с именем a1 комплексного числа $0.031+j420=3.1\cdot10^{-2}+j4.2\cdot10^{2}$.

Выражения в языке MATLAB

Выражением называется синтаксическая конструкция, представляющая последовательность операндов, разделённых знаками операций. В результате выполнения (вычисления) выражение возвращает значение, которое может быть присвоено переменной или её элементу. Роль операндов могут выполнять константы, переменные, их элементы и подмассивы, синтаксические конструкции, соответствующие вызовам функций (подпрограмм) MATLAB. Последние могут возвращать значения (выходные параметры), которые могут использоваться интерпретатором MATLAB для вычисления выражений. Прежде чем показывать примеры выражений, нужно рассмотреть перечень операций и приоритеты их выполнения в выражениях.

Основные операции в выражениях MATLAB

1. Унарный (одноместный) плюс.

+a или иначе uplus(a)

Здесь a – единственный операнд (например, переменная).

Для числовых данных унарный плюс не выполняет никаких действий и является пустой операцией. Для типов данных объектно-ориентированного программирования эта операция (как и все другие) может быть переопределена, т.е. может быть не пустой.

2. Унарный (одноместный) минус.

-a или иначе uminus(a)

Эта операция возвращает значение, противоположное значению операнда a. Она изменяет знак числового операнда.

3. Сложение массивов.

a+b или иначе plus(a,b)

Эта операция приводит к почленному сложению массивов a, b. Размеры обоих массивов должны быть одинаковыми. Один из этих операндов может быть скаляром (массивом, состоящим из одного элемента). В этом случае такая операция интерпретируется как сложение матрицы с числом.

4. Вычитание массивов.

a-b или иначе minus(a,b)

Эта операция приводит к почленному вычитанию массивов a, b. Размеры обоих массивов должны быть одинаковыми. Один из этих операндов может быть скаляром (массивом, состоящим из одного элемента). В этом случае такая операция интерпретируется как вычитание матрицы из числа или числа из матрицы.

5. Почленное умножение массивов.

a.*b или иначе times(a,b)

Эта операция приводит к почленному умножению массивов a, b. Размеры обоих массивов должны быть одинаковыми. Один из этих операндов может быть скаляром (массивом, состоящим из одного элемента). В этом случае такая операция интерпретируется как умножение матрицы на число.

6. Матричное умножение.

a*b или иначе mtimes(a,b)

Эта операция приводит к умножению матрицы a на матрицу b в соответствии с правилами, определёнными в линейной алгебре. Число столбцов матрицы a должно быть равно

числу строк матрицы b. Один из этих операндов может быть скаляром (массивом, состоящим из одного элемента). В этом случае такая операция интерпретируется как умножение матрицы на число.

7. Правое почленное деление массивов.

a./b или иначе rdivide(a,b)

Эта операция приводит к почленному делению массива a на массив b. Размеры обоих массивов должны быть одинаковыми. Один из этих операндов может быть скаляром (массивом, состоящим из одного элемента). В этом случае такая операция интерпретируется как деление матрицы на число или деление числа на матрицу.

8. Левое почленное деление массивов.

a.\b или иначе ldivide(a,b)

Эта операция приводит к почленному делению массива b на массив a. Размеры обоих массивов должны быть одинаковыми. Один из этих операндов может быть скаляром (массивом, состоящим из одного элемента). В этом случае такая операция интерпретируется как деление матрицы на число или деление числа на матрицу.

9. Правое деление матриц.

a/b или иначе mrdivide(a,b)

Эта операция приводит к вычислению матричного выражения $a \cdot b^{-1}$ в соответствии с правилами, определёнными в линейной алгебре.

10. Левое деление матриц.

a\b или иначе mldivide(a,b)

Эта операция приводит к вычислению матричного выражения $a^{-1} \cdot b$ в соответствии с правилами, определёнными в линейной алгебре.

11. Почленное возведение в степень массивов.

a.^b или иначе power(a,b)

Те же замечания, что и для других почленных операций.

12. Матричное возведение в степень.

a^b или иначе mpower(a,b)

Один из операндов обязательно должен быть скаляром. Матричное возведение в степень выполняется в соответствии с правилами, определёнными в линейной алгебре.

13. Операции отношения.

a<b операция «меньше»

a<=b операция «меньше или равно»

a>b операция «больше»

a>=b операция «больше или равно»

a==b операция «равно»

a~=b операция «не равно»

Операции отношения (сравнения) выполняются почленно. Размеры операндов должны быть одинаковы. Один из этих операндов может быть скаляром (массивом, состоящим из одного элемента). В этом случае скаляр «размножается» до размеров другого операнда, затем выполняется почленная операция. Результат операции – массив логического типа того же размера. Где равенство или неравенство справедливо, там записываются логические единицы, а где нет, там записываются логические нули.

14. Логические операции.

~a логическое отрицание (логическое «не»), не a

a&b логическое «и»
a|b логическое «или»
a&&b укороченное «и»
a||b укороченное «или»

Те же замечания, что и для операций отношения. Укороченные логические операции применяются только к скалярам.

15. Операция сечения массива.

a:b

Результат выполнения этой двуместной операции – массив-строка [a,a+1,a+2 … a+n], где n≥0 – максимально возможное целое число, такое, что a+n≤b. Если a<b, то результат выполнения операции – пустой массив.

a:d:b

Результат выполнения этой трёхместной операции – массив-строка [a,a+d,a+2*d … a+n*d], где n≥0 – максимально возможное целое число, такое, что a+n*d≤b при d>0 и a+n*d≥b при d<0. Если a<b при d>0 и a>b при d<0, то результат выполнения операции – пустой массив. Если d=0, то в любом случае результат выполнения операции – пустой массив.

Если операндами этой операции являются массивы, то операция сечения массива применяется к первым элементам операндов.

Приоритеты операций

Для формирования выражений в операторах MATLAB применяются комбинации различных операций: арифметических, логических и операций отношения. Порядок вычисления таких комбинированных выражений зависит от приоритетов используемых операций. Если все операции имеют одинаковый приоритет, то они выполняются последовательно слева направо. В противном случае их выполнение подчиняется правилам приоритета, приведённым в табл.1. Первый приоритет – наивысший, 11-й приоритет – самый низший.

Таблица 1

Приоритеты	Операции
1	() – круглые скобки
2	.' – простое транспонирование матрицы; ' – транспонирование матрицы с комплексным сопряжением; .^ – почленное возведение в степень; ^ – матричное возведение в степень
3	Одноместные операции: - – унарный минус; + – унарный плюс; ~ – логическое отрицание
4	Операции умножения и деления: .* – почленное умножение; ./ – почленное правое деление; .\ – почленное левое деление; * – матричное умножение; / – правое деление матриц; \ – левое деление матриц
5	+ – сложение; - – вычитание

6	: – операция сечения массива
7	Операции отношения: a<b –меньше; a<=b – меньше или равно; a>b – больше; a>=b – больше или равно; a==b – равно; a~=b – не равно
8	& – логическое «и»
9	| – логическое «или»
10	&& – укороченное логическое «и»
11	|| – укороченное логическое «или»

Операция **:** предназначена для формирования массива-строки с равномерным шагом.

Операторы MATLAB

Самым главным оператором языка MATLAB является оператор присваивания:

Переменная=выражение

В конце оператора может стоять перевод строки или запятая или точка с запятой. В последних двух случаях на одной строке может быть записано несколько операторов. Здесь **Переменная** – имя переменной MATLAB, которой нужно присвоить значение; **выражение** – выражение, записываемое по правилам синтаксиса языка MATLAB, значение которого вычисляется, когда управление получает этот оператор. Значение выражения можно присваивать также элементу (элементам) массива или полю переменной структурного типа. Оператор, записанный в представленном формате, называется оператором явного присваивания. Существует также оператор неявного присваивания:

выражение

При неявном присваивании значение выражения присваивается переменной с именем ans.

Ещё одной разновидностью оператора присваивания является оператор вызова функции, возвращающей более одного выходного параметра:

[varo1,varo2, …, varoN]=**имяфунк**(expri1,expri2, …, expriM)

Здесь varo1,varo2, …, varoN – имена фактических выходных параметров функции с именем **имяфунк**. Этим переменным присваиваются значения, возвращаемые функцией. expri1,expri2, …, expriM – фактические входные параметры вызываемой функции. Это могут быть выражения, вычисляемые системой MATLAB.

Если функция не возвращает выходных параметров, то её вызов имеет следующий формат:

имяфунк(expri1,expri2, …, expriM)

Этот формат вызова функции может также использоваться, если функция возвращает только один выходной параметр. В этом случае вызов функции может играть роль операнда в выражении при явном и неявном присваивании.

Если при отсутствии выходных параметров все входные параметры имеют тип char, и они заданы в явном виде (константами), то вызов такой функции может быть произведён командой, например,

format long g

Это эквивалентно следующему оператору:

format('long','g')

Условный оператор

Этот оператор представляет собой синтаксическую конструкцию из ключевых слов if, else, elseif, end и операторов MATLAB. Самая простая конструкция:

if **логвыр**
  последовательность_операторов
end

Здесь **логвыр** – вычисляемое логическое выражение. Если его значение равно логической единице (1 или true), то управление будет передано строкам, которые обозначены здесь как последовательность_операторов. Ключевое слово end обозначает конец этой последовательности. Если значение **логвыр** равно логическому нулю (0 или false), то управление будет передано оператору, следующему за end, значит, последовательность_операторов выполняться не будет. Если **логвыр** представляет собой массив любой размерности, то последовательность_операторов получит управление только в том случае, когда все члены логического массива будут равны true.

Вторая форма условного оператора имеет вид:

if **логвыр**
  последовательность_операторов1
else
  последовательность_операторов0
end

Если **логвыр**=true, то будет выполняться только последовательность_операторов1, а если **логвыр**=false, то будет выполняться только последовательность_операторов0.

Третья форма условного оператора имеет вид:

if **логвыр1**
  последовательность_операторов1
elseif **логвыр2**
  последовательность_операторов2
else
  последовательность_операторов0
end

Если **логвыр1**=true, то управление получает последовательность_операторов1, а когда она выполнится, управление передаётся строке, следующей за end. Если **логвыр1**=false, то будет вычисляться **логвыр2**. Если **логвыр2**=true, то управление получает последовательность_операторов2, а когда она выполнится, управление передаётся строке, следующей за end. Если **логвыр2**=false, то управление получает последовательность_операторов0, и ключевое слово end не будет влиять на дальнейшие передачи управления.

Оператор переключения

Синтаксис этого оператора имеет вид:

switch **выражение**
 case значение1
  последовательность_операторов1
 case значение2
  последовательность_операторов2
…
 case значениеn

```
  последовательность_операторовn
 otherwise
  последовательность_операторов0
end
```

Здесь **выражение** – это обязательно скаляр или строка символов (char). Если **выражение** = значение1, то выполняется последовательность_операторов1, затем управление передаётся строке, следующей за end. Если **выражение** = значение2, то выполняется последовательность_операторов2, затем управление передаётся строке, следующей за end… Если **выражение** = значениеn, то выполняется последовательность_операторовn, затем управление передаётся строке, следующей за end. Если **выражение** не равно ни одному из значений, перечисленных в строках case, то выполняется последовательность_операторов0, и ключевое слово end не будет влиять на дальнейшие передачи управления.

Оператор цикла с определённым числом операций

Синтаксис:

```
for переменная_цикла=массив
 последовательность_операторов
end
```

Если **массив** одномерный, то последовательность_операторов выполнится столько раз, сколько элементов имеет **массив**. На первом шаге переменная_цикла принимает значение **массив**(1), на втором шаге – **массив**(2) и т.д. Если **массив** двумерный, то на первом шаге переменная_цикла принимает значение **массив**(:,1), т.е. первый столбец, на втором шаге – **массив**(:,2) и т.д.

Оператор цикла с неопределённым числом операций

Синтаксис:

```
while логическое_выражение
 последовательность_операторов
end
```

Здесь последовательность_операторов выполняется многократно, пока логическое_выражение = true. Если логическое_выражение – массив, то оно считается истинным, если все элементы массива имеют значение, равное логической единице. Если массив пустой, то логическое_выражение считается ложным.

Оператор принудительного завершения шага цикла

Синтаксис:

```
continue
```

Этот оператор применяется в теле циклов for или while. Выполнение этого оператора приводит к передаче управления строке end, после чего цикл переходит на следующий шаг (выполнение цикла продолжается со следующего шага).

Оператор принудительного завершения цикла

Синтаксис:

```
break
```

Этот оператор применяется в теле циклов for или while. Выполнение этого оператора приводит к передаче управления строке, следующей за end, т.е. к прекращению выполнения цикла. Если цикл вложенный, то оператор break может применяться только в самом внутреннем цикле.

Оператор возврата в вызывающую программу

return

В результате его выполнения происходит возврат управления в вызывающую программу. Этот оператор применяется также для выхода из режима keyboard.

Специальные символы в языке MATLAB

Здесь кратко расскажем о назначении специальных символов в синтаксических конструкциях языка MATLAB.

Квадратные скобки

[ ]

1) Формирование массивов;
2) Указание списка фактических выходных параметров вызова функции;
3) Указание списка формальных выходных параметров в заголовке m-функции.

Фигурные скобки

{ }

1) Формирование массивов ячеек;
2) Обращение к элементу массива ячеек.

Круглые скобки

( )

1) Принудительное изменение приоритетов операций в выражениях (см. табл.1);
2) Указание списка индексов элемента (элементов) массива;
3) Указание списка фактических входных параметров вызова функции;
4) Указание списка формальных входных параметров в заголовке m-функции.

Знак равенства

=

1) Знак присваивания в операторах;
2) Входит в состав знаков операций отношения.

Апостроф

'

1) Входит в состав знаков операций транспонирования;
2) Используется в синтаксисе констант типа char.

Точка

.

1) Десятичная точка в числовых константах;
2) Входит в состав знаков почленных операций над массивами;
3) Отделение имени переменной от имени её поля при работе со структурными типами данных.

Две точки

..

Переход по дереву каталогов на один уровень вверх.

Три и более точек

…

Признак продолжения оператора в следующей командной строке или в следующей строке m-файла.

Запятая

,

1) Разделение операторов языка MATLAB;
2) Разделитель в списке индексов элементов массива;
3) Разделитель элементов массива в пределах строки (применяется внутри квадратных скобок);
4) Разделитель в любом списке параметров функции.

Точка с запятой

;

1) Разделение операторов языка MATLAB с подавлением вывода результатов в командное окно;
2) Разделитель строк при формировании массива (применяется внутри квадратных скобок).

Двоеточие

:

Знак операции сечения массива.

Знак процента

%

Конец интерпретируемой части командной строки или строки m-файла и начало комментария.

Восклицательный знак

!

Указатель ввода команды операционной системы.

Пробел

1) Разделитель ключевых слов в операторах MATLAB;
2) Разделитель элементов массива в пределах строки (применяется внутри квадратных скобок).

Формирование массивов с помощью квадратных скобок

Чтобы сформировать массив, надо внутри квадратных скобок просто перечислить его элементы. Разделителем элементов в пределах одной строки формируемого массива является запятая или пробел, разделителем строк в массиве является точка с запятой. Все строки в формируемой матрице должны иметь одинаковое количество элементов.

Пример формирования массива можно привести из курса «Теоретические основы электротехники» (ТОЭ, теория цепей). Пусть имеется трёхфазная цепь, схема которой изображена на рис. 2. Утолщёнными линиями там выделены ветви дерева (пассивные ветви 2, 4, а также фазные источники ЭДС $\dot{E}_A, \dot{E}_B, \dot{E}_C$), обычными линиями показаны ветви связи (пассивные ветви 1, 3, 5, 6).

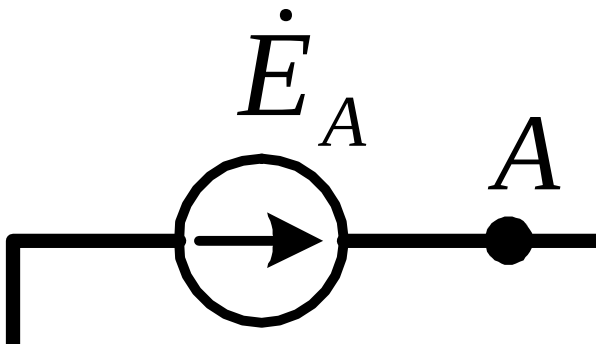

Рис. 2. Пример схемы электрической цепи

Матрица главных контуров этой цепи равна

$$[B] = \begin{array}{c} 1 \\ 3 \\ 5 \\ 6 \\ \\ \end{array} \begin{array}{c} \left[\begin{array}{ccccccccc} 1 & -1 & 0 & 1 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & -1 \end{array}\right], \\ \begin{array}{ccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & A & B & C \end{array} \end{array}$$

где цифры перед открывающей скобкой – номера ветвей связи, образующих главные контуры, соответствующие строкам матрицы [*B*]. Цифры и буквы под значением матрицы – обозначения ветвей, соответствующих столбцам матрицы [*B*].

Чтобы ввести эту матрицу в MATLAB, нужно выполнить следующий оператор:

```
B=[1 -1 0 1 0 0 1 -1 0; 0 0 1 -1 0 0 -1 0 1; 0 0 0 0 1 0 0 1 -1; 0 1 0 0 0 1 0 1 -1];
```

Точка с запятой в конце оператора поставлена, чтобы в командном окне не отображался результат его выполнения. В рабочей области MATLAB будет создана переменная типа double с именем B. Значением этой переменной будет матрица, состоящая из 4 строк и 9 столбцов.

Выделение подматриц. Склейка матриц

Эти операции рассмотрим на примере топологической матрицы B, соответствующей схеме, изображённой на рис. 2. Пусть эта матрица уже введена в рабочую область MATLAB. Выделим из этой матрицы три подматрицы: 1) блок, соответствующий первым трём главным контурам и только пассивным ветвям; 2) блок, соответствующий последнему главному контуру и только пассивным ветвям; 3) блок, соответствующий всем главным контурам и только активным ветвям. Соответствующие блоки присвоим отдельным переменным MATLAB. Для этого выполним следующую последовательность операторов:

```
Bp1=B(1:3,1:6);  % 1-й из этих блоков
Bp2=B(4,1:6);  % 2-й из этих блоков
Ba=B(:,7:9);  % 3-й из этих блоков
```

Из этого примера видно, что операцию сечения массива можно применять в индексных выражениях для выделения блоков матриц (а в общем случае – массивов любой размерности). Второй и третий операторы в данном случае можно записать по-другому (от этого результат не изменится):

```
Bp2=B(end,1:6);  % 2-й из этих блоков
Ba=B(:,7:end);  % 3-й из этих блоков
```

Ключевое слово end в индексных выражениях обозначает максимально возможное значение соответствующего индекса (последняя строка или последний столбец матрицы). Если

индексное выражение состоит только из двоеточия, то оно равносильно индексному выражению 1:end.
Теперь склеим выделенные блоки и сравним результат с исходной матрицей B:
co=B==[[Bp1;Bp2],Ba]
В результате выполнения этого оператора переменной с именем co будет присвоена логическая матрица того же размера, что и B. Каждый элемент матрицы co будет равен логической единице. Результат выполнения оператора будет показан в командном окне. Здесь потребовалось применить вложенные квадратные скобки, т.к. операция горизонтальной склейки массивов (её знак – запятая или пробел) выполняется с более высоким приоритетом, чем операция вертикальной склейки (её знак – точка с запятой).
Возможно также выделение блоков с изменением последовательности чередования строк и столбцов. Из матрицы B выделим две подматрицы: блок, соответствующий ветвям дерева, и блок, соответствующий ветвям связи:

```
D=[2 4 7 8 9];  % список ветвей дерева
S=[1 3 5 6];  % список ветвей связи
BD=B(:,D)  % подматрица ветвей дерева
BS=B(:,S)  % подматрица дополнения дерева
```

Из этого примера видно, что у матрицы главных контуров подматрица дополнения дерева является единичной квадратной матрицей.

Наиболее важные функции для работы с массивами

Базовая информация

disp – отображение текста или массива в командном окне MATLAB.
isempty – проверка, является ли массив пустым.
isequal – проверка массивов на равенство.
isequalwithequalnans – то же, элементы, равные NaN, тоже проверяются на равенство.
isfinite – проверка элементов массива на конечные значения.
isfloat – проверка, состоит ли массив из чисел с плавающей точкой.
isinf – проверка элементов массива на бесконечные значения.
isinteger – проверка, состоит ли массив из чисел с фиксированной точкой.
islogical – проверка, является ли массив логическим.
isnan – проверка элементов массива на нечисловые значения NaN.
isnumeric – проверка, является ли массив числовым.
isscalar – проверка, является ли входной параметр скаляром.
issparse – проверка, является ли массив разреженным (применяется к матрицам).
isvector – проверка, является ли массив одномерным (строкой или столбцом).
length – максимальный размер массива или длина одномерного массива.
max – максимальный элемент вдоль заданной размерности массива или максимум из двух входных параметров.
min – минимальный элемент вдоль заданной размерности массива или минимум из двух входных параметров.
ndims – размерность (число измерений) массива.
numel – число элементов массива.
size – размеры массива.

Элементарные матрицы и массивы

blkdiag – формирование блочно-диагональной матрицы из входных параметров.

diag – формирование диагональной матрицы из одномерного массива или выделение диагонали из матрицы.
eye – формирование единичной матрицы.
linspace – формирование массива-строки с равномерным шагом в линейном масштабе.
meshgrid – формирование двумерной или трёхмерной сетки из одномерных массивов.
ones – формирование матрицы единиц.
rand – формирование псевдослучайного массива с равномерной плотностью распределения в диапазоне от 0 до 1.
randn – формирование псевдослучайного массива с нормальным законом распределения с математическим ожиданием 0 и среднеквадратичным отклонением 1.
zeros – формирование матрицы нулей.

Операции над массивами

cross – векторное произведение в трёхмерном пространстве.
cumprod – кумулятивное произведение массива.
cumsum – кумулятивная сумма массива.
dot – скалярное произведение массивов.
idivide – почленное целочисленное деление массивов.
kron – произведение Кронекера.
prod – произведение элементов массива вдоль заданной размерности.
sum – сумма элементов массива вдоль заданной размерности.
tril – нижняя треугольная часть матрицы.
triu – верхняя треугольная часть матрицы.

Манипуляции массивами

circshift – циклический сдвиг элементов вдоль заданных размерностей.
flipdim – зеркальное отражение массива вдоль заданной размерности.
fliplr – зеркальное отражение массива по горизонтали (вдоль второй размерности).
flipud – зеркальное отражение массива по вертикали (вдоль первой размерности).
ipermute – обратная перестановка размерностей многомерного массива.
permute – перестановка размерностей многомерного массива.
repmat – формирование массива путём размножения блока вдоль заданных размерностей.
reshape – изменение размеров массива без изменения числа его элементов.
rot90 – поворот матрицы на 90 градусов.
shiftdim – сдвиг размерностей массива.
sort – сортировка элементов массива вдоль заданной размерности в порядке возрастания или убывания.
sortrows – сортировка строк массива в порядке возрастания элементов столбца.
squeeze – удаление всех единичных размерностей многомерного массива.

Простой пример работы с блоками

Опять вернёмся к матрице главных контуров электрической цепи, схема которой изображена на рис. 2. Здесь представим алгоритм определения матрицы главных сечений по известной матрице главных контуров. Пусть в рабочей области имеется матрица главных контуров B, список ветвей дерева D и список ветвей связи S (см. предыдущий пример). Выполним следующую последовательность операторов:

```
[s,v]=size(B);  % s - число ветвей связи (главных контуров), v - число ветвей
d=v-s;  % число ветвей дерева
if s~=length(S)|d~=length(D)
```

```
  Q=NaN; % ошибка: массивы S и D имеют неправильные размеры
elseif B(:,S)~=eye(s)
  Q=repmat(NaN,d,v); % ошибка: подматрица дополнения дерева не является единичной
else
  Q=zeros(d,v);
  Q(:,[D S])=[eye(d),-B(:,D).'];
end
disp(Q)
```

В результате в командное окно MATLAB будет выдано значение матрицы главных сечений:

```
     1     1     0     0     0    -1     0     0     0
    -1     0     1     1     0     0     0     0     0
    -1     0     1     0     0     0     1     0     0
     1     0     0     0    -1    -1     0     1     0
     0     0    -1     0     1     1     0     0     1
```

Проанализировав схему на рис. 2, убеждаемся, что матрица главных сечений рассчитана верно.

Примеры применения функций для работы с массивами

Пример 1

Пусть имеется генеральный план предприятия, на котором показаны потребители электроэнергии, питаемые от главной понизительной подстанции. Последнюю обычно располагают вблизи центра нагрузки потребителей. Координаты этого центра определяют как средневзвешенные значения координат потребителей. В качестве весовых коэффициентов при усреднении координат принимают номинальную активную мощность, номинальную реактивную мощность или номинальную полную мощность. Предположим, что координаты потребителей записаны в одномерные массивы-строки x и y (массив горизонтальных координат и массив вертикальных координат).

Десять потребителей распределим псевдослучайным образом на квадратной площадке 300×300 м с равномерной плотностью распределения. Это можно сделать с помощью функции rand. Чтобы наиболее естественно смоделировать случайный характер выходных значений функции rand, установим начальное состояние генератора псевдослучайных чисел от часов компьютера. Всё это проделаем с помощью следующей последовательности операторов:

```
clo=clock
rand('twister',clo(end)/60*2^32)
x=rand(1,10)*300
y=rand(1,10)*300
plot(x,y,'bo','linewidth',2)
grid on
axis equal
```

Первый из этих операторов присваивает массиву с именем clo текущее состояние компьютерных часов: clo(1) – календарный номер текущего года; clo(2) – месяц; clo(3) – текущий календарный день; clo(4) – часы; clo(5) – минуты; clo(6) – секунды (в версии 7.4.0 (R2007a) – с точностью до тысячной доли). Второй оператор устанавливает начальное состояние генератора псевдослучайных чисел по счётчику секунд компьютерных часов.

Третий и четвёртый операторы формируют псевдослучайные десятиэлементные массивы-строки координат электропотребителей на заданной площадке. Остальные операторы записаны для наглядного графического отображения положения потребителей на площадке. Пятый оператор в фигуре № 1 строит график положения потребителей: маркеры – синие кружочки двойной толщины. Шестой оператор наносит координатную сетку. Последний оператор устанавливает одинаковый масштаб по горизонтальной и вертикальной оси. Символы «точка с запятой» не поставлены в конце операторов, чтобы результаты их выполнения были показаны в командном окне, например, такие:

```
clo =
       2008          8         24          0         11       12.546
x =
  Columns 1 through 9
      40.803      117.59      230.16      48.203      250.68      280.71      46.105      217.41
91.006
  Column 10
      11.423
y =
  Columns 1 through 9
      32.109      275.71      64.598      5.7866      263.55      206.08      23.227      166.73
105.53
  Column 10
      122.51
```

В окне Figure 1 выполним команду меню Edit/ Copy Figure. Графика этой фигуры запишется в системный буфер обмена. Вставим содержимое буфера в документ Micrisoft Word. В результате получим рис. 3.

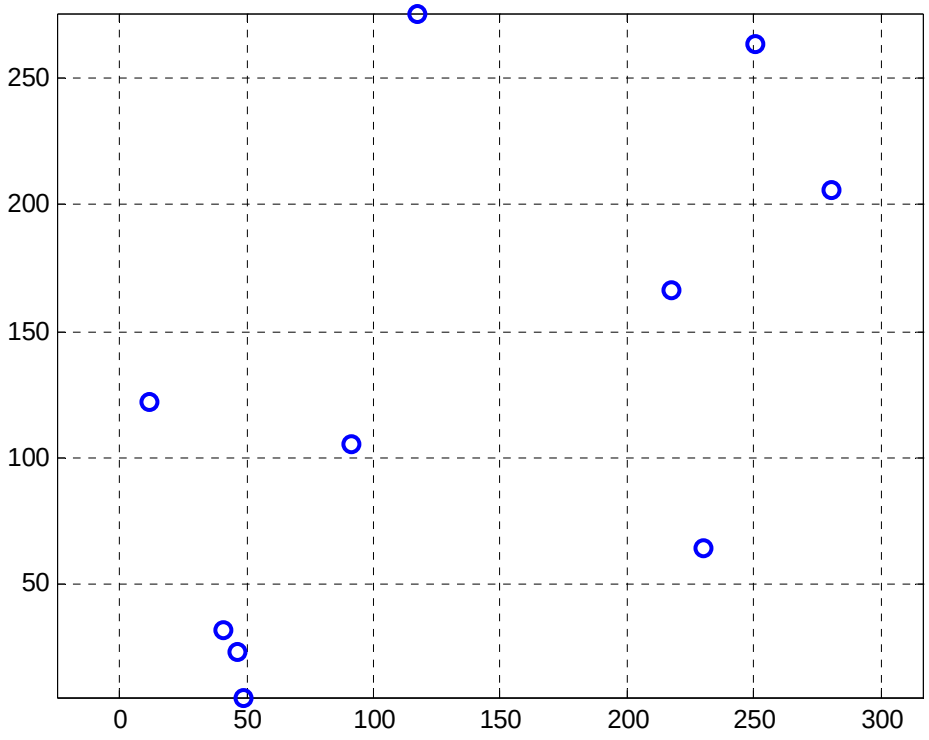

Рис. 3. Фигура с расположением электропотребителей на площадке

Псевдослучайным образом зададим номинальные активные мощности потребителей с равномерной плотностью распределения в диапазоне от 0 до 500 кВт. Для этого выполним следующий оператор:

```
P=rand(1,10)*500
```

Результат выполнения оператора:

```
P =
  Columns 1 through 9
    297.48    46.578    200.75    290.14    484.28    209.51    30.091    499.3
237.09
  Column 10
    407.29
```

Для вычисления координат центра нагрузки потребителей выполним следующую последовательность операторов:

```
PP=sum(P) % Суммарная номинальная активная мощность потребителей, кВт
xc=dot(P,x)/PP % Горизонтальная координата центра нагрузки
yc=dot(P,y)/PP % Вертикальная координата центра нагрузки
```

Результаты выполнения операторов:

```
PP =
    2702.5
xc =
    145.86
yc =
    135.69
```

Чтобы нанести рассчитанный центр нагрузки на график в фигуре 1, выполним следующую последовательность операторов:

```
hold on
plot(xc,yc,'kh','linewidth',2,'MarkerSize',15)
```

В фигуре 1 увидим график, показанный на рис. 4. Здесь центр нагрузки потребителей по активной мощности отмечен большой шестиконечной звездой.

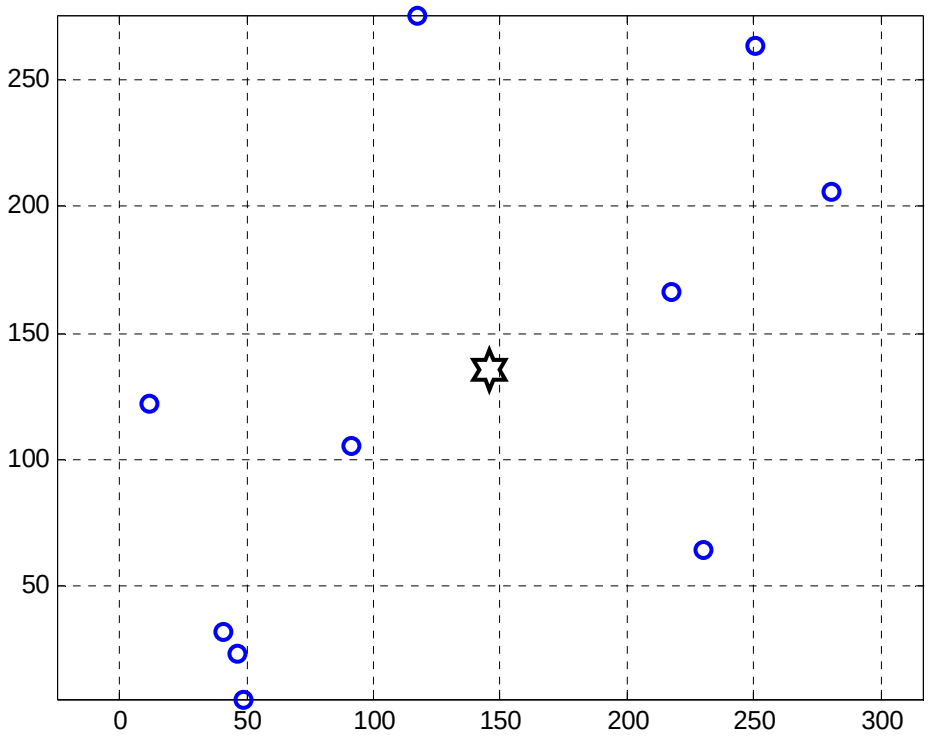

Рис. 4. Расположение электропотребителей и центра нагрузки

Пример 2

Пусть некоторая однофазная нагрузка питается от электронного преобразователя частоты. Формы напряжения и тока искажены по отношению к синусоиде:

```
clo=clock;
randn('state',clo(end)/60*2^32);
t=linspace(0,10,2001);   % Моменты времени, мс (здесь один период напряжения)
deu=[randn(1,70) 0]; deu(end)=deu(1);
tde=linspace(0,10,71);
u=100*sin(2*pi/10*t)+25*interp1(tde,deu,t,'cubic');  % Мгновенное напряжение, В
dei=[randn(1,70) 0]; deu(end)=deu(1);
ii=10*sin(2*pi/10*t-pi/6)+1.5*interp1(tde,dei,t,'cubic');   % Мгновенный ток, А
```

Здесь представлена последовательность операторов, определяющая мгновенные значения синусоидального напряжения с периодом 10 мс, нулевой начальной фазой и амплитудой 100 В, на которое наложено псевдослучайное искажение с нормальным законом распределения и стандартной девиацией 25 В. Искажение не нарушает условие периодичности напряжения. Последние два оператора определяют мгновенные значения синусоидального тока с тем же периодом, начальной фазой $-\pi/6$ рад и амплитудой 10 А, на который наложено псевдослучайное искажение с нормальным законом распределения и стандартной девиацией 1.5 А. Искажение не нарушает условие периодичности тока. Здесь использована стандартная функция системы MATLAB с именем interp1 – одномерная интерполяция таблично заданных функций.

Осциллограмму выходного напряжения преобразователя построим следующей последовательностью операторов:

```
plot(t,u,'k-','linewidth',2)
```

grid on
xlabel('время, мс')
ylabel('напряжение, В')
График будет построен в фигуре 1 (рис. 5).

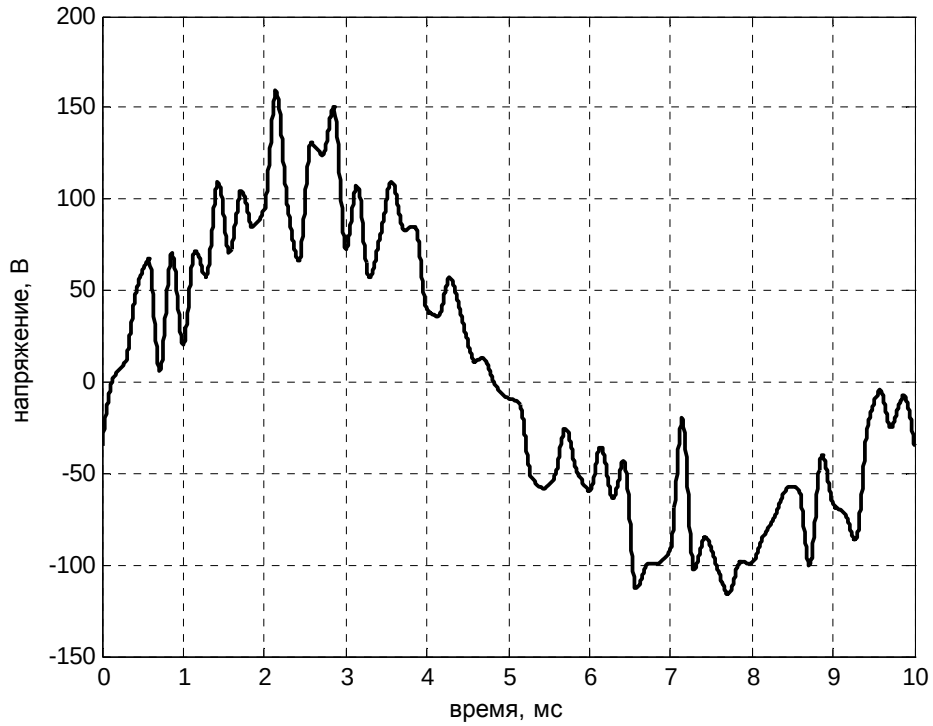

Рис. 5. Фигура 1 с осциллограммой напряжения

Осциллограмму выходного тока преобразователя построим следующей последовательностью операторов:
figure
plot(t,ii,'b-','linewidth',2)
grid on
xlabel('время, мс')
ylabel('ток, А')
График будет построен в фигуре 2 (рис. 6).

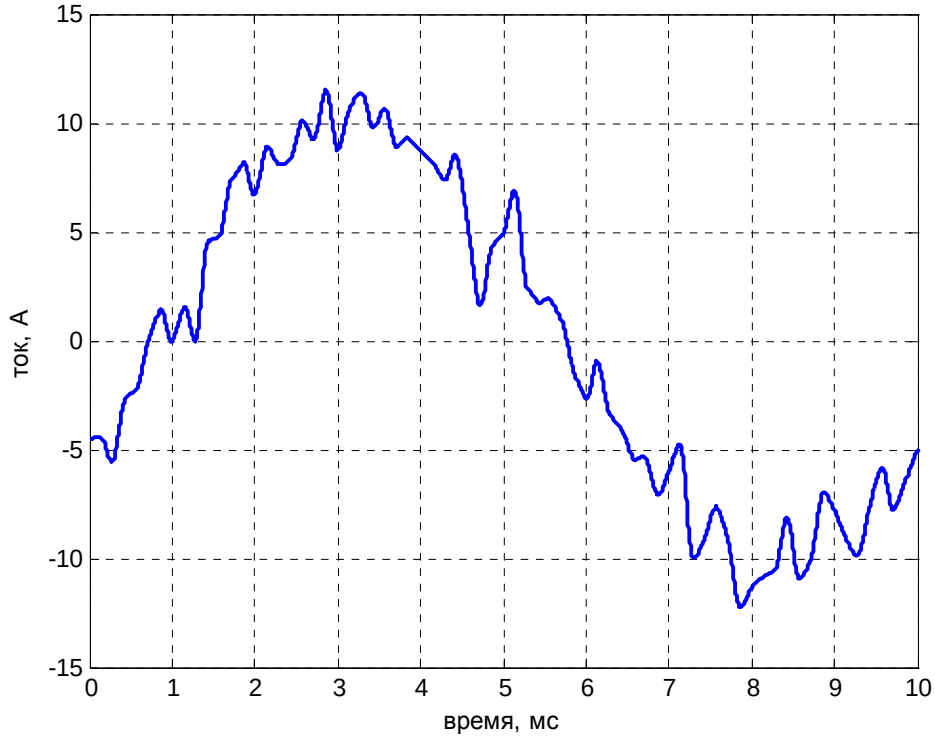

Рис. 6. Фигура 2 с осциллограммой тока

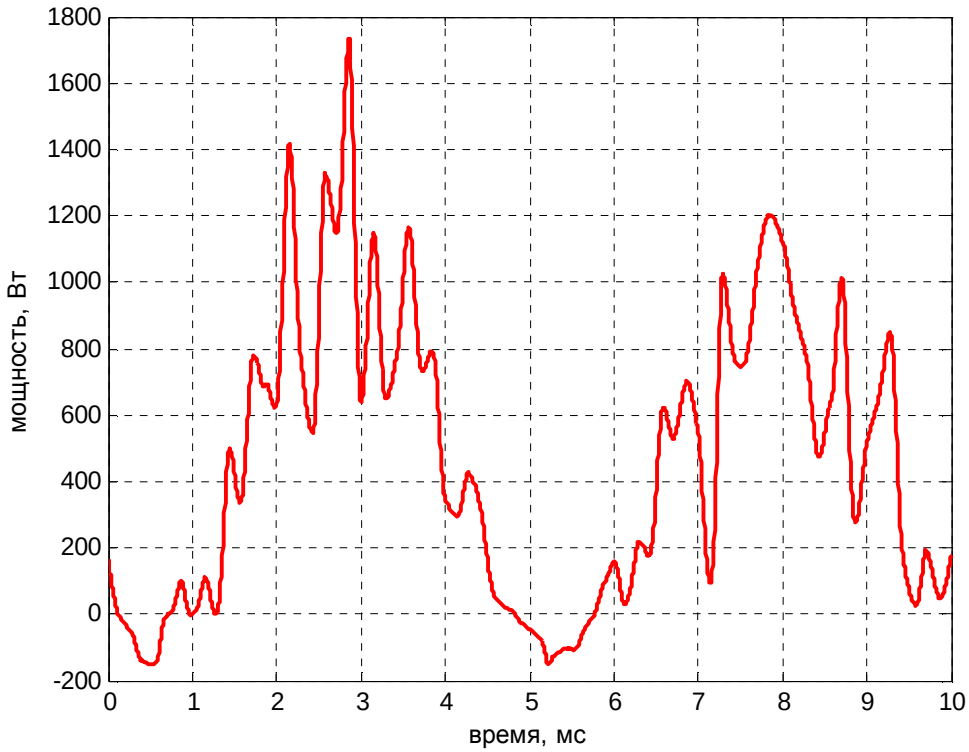

Рис. 7. Фигура 3 с осциллограммой мгновенной мощности

Осциллограмму мгновенной потребляемой мощности нагрузки рассчитаем и построим следующей последовательностью операторов:

```
p=u.*ii; % массив мгновенных значений мощности, Вт
figure
plot(t,p,'r-','linewidth',2)
```

grid on
xlabel('время, мс')
ylabel('мощность, Вт')
График будет построен в фигуре 3 (рис. 7).
Активную мощность определим по формуле

$$P = \frac{1}{T}\int_0^T p(t)\,\mathrm{d}t, \qquad (1)$$

где T – период, $p(t)$ – мгновенная мощность.
Реактивную мощность определим по формуле

$$Q = \frac{1}{2\pi}\int_0^T u(t)\cdot\frac{\mathrm{d}\,i(t)}{\mathrm{d}t}\,\mathrm{d}t, \qquad (2)$$

где $u(t)$ – мгновенное напряжение, $i(t)$ – мгновенный ток.
Полную мощность определим по формуле

$$S = U \cdot I, \qquad (3)$$

где U – действующее значение напряжения; I – действующее значение тока. Эти величины определяются по формулам

$$U = \sqrt{\frac{1}{T}\int_0^T u^2(t)\,\mathrm{d}t}, \quad I = \sqrt{\frac{1}{T}\int_0^T i^2(t)\,\mathrm{d}t}. \qquad (4)$$

Мощность искажения определим по формуле

$$N = \sqrt{S^2 - P^2 - Q^2}. \qquad (5)$$

Для расчёта по формулам (1) – (5) этих важных энергетических параметров режима работы преобразователя частоты выполним следующую последовательность операторов:

```
T=t(end)-t(1); % период тока и напряжения, мс
dt=diff(t); % конечная разность массивов моментов времени, мс
P=(dot(p(1:end-1),dt)+dot(p(2:end),dt))/T/2 % активная мощность, Вт
di=diff(ii); % конечная разность массива мгновенных токов, А
Q=(dot(u(1:end-1),di)+dot(u(2:end),di))/pi/4 % реактивная мощность, ВАр
U=((dot(u(1:end-1).^2,dt)+dot(u(2:end).^2,dt))/T/2)^0.5 % действ зн-е напряж-я, В
I=((dot(ii(1:end-1).^2,dt)+dot(ii(2:end).^2,dt))/T/2)^0.5 % действ значение тока, А
S=U*I % полная электрическая мощность, ВА
N=(S^2-P^2-Q^2)^0.5 % мощность искажения в нагрузке, ВА
hi=P/S % коэффициент мощности преобразователя в данном режиме
```

В результате в командное окно будут выданы значения энергетических параметров режима работы вентильного преобразователя частоты:

```
P =
    458.72
Q =
    255.86
U =
    74.122
I =
    7.2812
S =
```

539.7

N =

124.04

hi =

0.84996

Для расчёта постоянной составляющей напряжения и тока выполним следующие операторы:

```
U0=(dot(u(1:end-1),dt)+dot(u(2:end),dt))/T/2 % постоянная составляющая напряжения, В
I0=(dot(ii(1:end-1),dt)+dot(ii(2:end),dt))/T/2 % постоянная составляющая тока, А
```

Результаты:

U0 =

1.8001

I0 =

-0.13675

Для расчёта комплексных действующих значений гармоник напряжения и тока выполним следующую последовательность операторов:

```
om=2*pi/T; % круговая частота основной гармоники, рад/с
nt=length(t); % всего временных отсчётов
ng=41; % столько гармоник будем рассчитывать
fm=repmat((1:ng).',1,nt); % распределяем память под экспоненциальные множители
fm=exp(1i*fm*om.*repmat(t,ng,1)); % массив экспоненциальных множителей
ur=repmat(u,ng,1); % мгновенные напряжения размножили на число гармоник
ir=repmat(ii,ng,1); % мгновенные токи размножили на число гармоник
dtr=repmat(dt,ng,1);
Ug=1i*(dot(ur(:,1:end-1).*fm(:,1:end-1),dtr,2)+...
  dot(ur(:,2:end).*fm(:,2:end),dtr,2))/T/2^0.5;
Ig=1i*(dot(ir(:,1:end-1).*fm(:,1:end-1),dtr,2)+...
  dot(ir(:,2:end).*fm(:,2:end),dtr,2))/T/2^0.5;
disp('Комплексные действующие значения гармоник напряжения и тока')
disp('номер гармоники                напряжение                ток')
disp([(1:ng).' Ug Ig])
```

Результаты:

Комплексные действующие значения гармоник напряжения и тока

номер гармоники	напряжение	ток
1	71.189 + 0.4552i	6.3379 - 3.3758i
2	0.50474 - 6.3661i	0.0058845 - 0.29718i
3	-0.36821 + 2.146i	-0.28404 + 0.015594i
4	-0.096871 - 0.74825i	-0.055838 - 0.076527i
5	2.5965 + 2.2612i	-0.118 - 0.12454i
6	1.0003 - 0.58938i	0.35912 + 0.012687i
7	1.4498 + 2.7358i	0.039518 + 0.040729i
8	-3.8302 + 0.99034i	0.050392 - 0.050684i
9	0.76816 - 7.2688i	-0.35586 - 0.13495i
10	-0.1276 - 1.1662i	-0.0085304 + 0.0328i
11	-0.58653 + 0.89965i	-0.31114 + 0.073597i
12	-0.51006 - 6.1258i	0.14025 + 0.18888i

13	-3.0142 - 4.1841i	-0.069389 - 0.021723i
14	-1.2655 + 4.2522i	0.010794 + 0.051862i
15	1.2249 - 3.8516i	0.076361 - 0.32518i
16	3.3803 - 1.4902i	0.39462 + 0.12805i
17	-0.22254 - 1.2232i	0.0070363 - 0.0073574i
18	-1.1379 + 2.434i	0.041858 + 0.33133i
19	0.40633 + 3.7433i	-0.016063 - 0.086814i
20	1.4011 + 0.49707i	0.12716 + 0.16411i
21	-2.773 + 1.2266i	-0.16773 + 0.15872i
22	-3.0799 - 0.16571i	-0.15053 - 0.020883i
23	0.63857 + 1.9376i	0.14696 - 0.060618i
24	3.0746 - 0.096081i	0.041392 + 0.071183i
25	-0.10822 + 0.16585i	0.027746 + 0.15521i
26	0.4032 - 2.126i	-0.07841 - 0.050791i
27	0.082487 - 1.6351i	-0.093488 + 0.10768i
28	-0.50291 + 3.7815i	0.0384 + 0.085481i
29	-0.16854 - 2.6659i	0.10201 - 0.054326i
30	1.0476 - 5.8509i	0.095843 - 0.23552i
31	-1.5783 - 0.41658i	-0.019206 + 0.087454i
32	-0.64916 - 0.94984i	-0.014114 - 0.10283i
33	-0.27463 - 1.293i	-0.038275 - 0.027495i
34	-1.2001 + 1.4447i	-0.027849 - 0.0076833i
35	-0.054953 + 3.0099i	-0.0041241 + 0.040993i
36	1.0484 + 1.3101i	0.042924 - 0.0099506i
37	0.26883 - 0.76459i	0.0342 - 0.019683i
38	0.57769 - 0.90694i	0.027834 - 0.065053i
39	0.8771 - 0.32089i	0.015072 + 0.046179i
40	-0.74946 - 3.4392i	-0.04834 - 0.13349i
41	0.18041 - 1.3082i	-0.051994 - 0.028092i

Одними из важнейших показателей качества вентильных преобразователей частоты являются коэффициенты гармоник напряжения и тока при некоторых заданных режимах. Этот коэффициент определяется как отношение действующего значения всей совокупности неосновных гармоник к действующему значению основной гармоники. Чтобы рассчитать эти коэффициенты для переменной составляющей тока и напряжения, выполним следующую последовательность операторов:

```
unog=u-U0-2^0.5*(real(Ug(1))*sin(om*t)+imag(Ug(1))*cos(om*t));
inog=ii-I0-2^0.5*(real(Ig(1))*sin(om*t)+imag(Ig(1))*cos(om*t));
Unog=((dot(unog(1:end-1).^2,dt)+dot(unog(2:end).^2,dt))/T/2)^0.5
Inog=((dot(inog(1:end-1).^2,dt)+dot(inog(2:end).^2,dt))/T/2)^0.5
U1=abs(Ug(1)) % действующее значение 1-й гармоники напряжения, В
I1=abs(Ig(1)) % действующее значение 1-й гармоники тока, А
kgu=Unog/U1 % коэффициент гармоник напряжения
kgi=Inog/I1 % коэффициент гармоник тока
```

Результаты:

```
Unog =
    20.561
```

Inog =

1.1969

U1 =

71.191

I1 =

7.1809

kgu =

0.28881

kgi =

0.16668

Здесь Unog – действующее значение всей совокупности неосновных гармоник напряжения; Inog – действующее значение всей совокупности неосновных гармоник тока. Коэффициенты гармоник количественно показывают, насколько сильно периодическое напряжение или ток отличаются от синусоиды. В данном случае осциллограмма тока (рис.6) меньше отличается от синусоиды, чем осциллограмма напряжения (рис. 5). Коэффициент гармоник является одним из важнейших показателей качества электроэнергии в централизованных и автономных системах электроснабжения.

Наиболее важные функции линейной алгебры

Анализ матриц

cond – число обусловленности матрицы по отношению к обращению.

condeig – число обусловленности матрицы по отношению к собственным значениям.

det – определитель матрицы.

norm – нормы одномерных массивов и матриц.

normest – оценка L_2–нормы матрицы.

rank – ранг матрицы.

trace – сумма диагональных элементов.

Линейные уравнения

chol – разложение на треугольные множители методом Холесского.

cholinc – неполное разложение Холесского для разреженной матрицы.

funm – вычисление матричной функции.

ilu – неполное LU-разложение для разреженной матрицы.

inv – вычисление обратной матрицы (обращение матрицы).

linsolve – решение системы линейных алгебраических уравнений (СЛАУ).

lu – LU-разложение матрицы.

luinc – неполное LU-разложение для разреженной матрицы.

pinv – псевдообращение матрицы методом Мура-Пенроуза.

Собственные и сингулярные значения

eig – нахождение собственных значений и собственных векторов.

eigs – нахождение самых больших собственных значений и собственных векторов разреженной матрицы.

gsvd – обобщённое сингулярное разложение.

hess – приведение матрицы к Хессенберговой форме (Хессенбергово разложение).

poly – построение полинома по указанным корням.

polyeig – решение полиномиальной матричной задачи на собственные значения.

schur – разложение Шура.

sqrtm – матричный квадратный корень.

svd – сингулярное разложение матрицы

Матричная экспонента и логарифм

expm – матричная экспонента.

logm – матричный логарифм.

Элементарные математические функции

Тригонометрические функции

acos – арккосинус.

acosd – арккосинус в градусах.

acosh – гиперболический арккосинус.

acot – арккотангенс.

acotd – арккотангенс в градусах.

acoth – гиперболический арккотангенс.

acsc – арккосеканс.

acscd – арккосеканс в градусах.

acsch – гиперболический арккосеканс.

asec – арксеканс.

asecd – арксеканс в градусах.

asech – гиперболический арксеканс.

asin – арксинус.

asind – арксинус в градусах.

asinh – гиперболический арксинус.

atan – арктангенс.

atan2 – четырёхквадрантный арктангенс.

atand – арктангенс в градусах.

atanh – гиперболический арктангенс.

cos – косинус.

cosd – косинус с аргументом в градусах.

cosh – гиперболический косинус.

cot – котангенс.

cotd – котангенс с аргументом в градусах.

coth – гиперболический котангенс.

csc – косеканс.

cscd – косеканс с аргументом в градусах.

csch – гиперболический косеканс.

hypot – квадратный корень из суммы квадратов модулей аргументов.

sec – секанс.

secd – секанс с аргументом в градусах.

sech – гиперболический секанс.

sin – синус.

sind – синус с аргументом в градусах.

sinh – гиперболический синус.

tan – тангенс.

tand – тангенс с аргументом в градусах.

tanh – гиперболический тангенс.

Показательные и логарифмические функции

exp – экспонента e^x.

log – натуральный логарифм.
log10 – десятичный логарифм.
nthroot – действительный корень *n*-й степени из действительного числа.
reallog – натуральный логарифм для неотрицательных действительных массивов.
realpow – степень массива только для действительных выходных значений
realsqrt – квадратный корень для неотрицательных действительных массивов.
sqrt – квадратный корень.

Функции для комплексных чисел

abs – модуль комплексных или действительных значений.
angle – главный аргумент комплексных чисел (в радианах).
complex – объединение действительных и мнимых частей в комплексные числа.
conj – комплексное сопряжение (изменение знака мнимой части).
cplxpair – сортировка комплексного массива так, чтобы числа располагались комплексно-сопряжёнными парами.
i или j – мнимая единица.
imag – мнимая часть комплексных чисел.
isreal – проверка, являются ли все элементы массива действительными числами.
real – действительная часть комплексных чисел.
sign – сигнум комплексных или действительных чисел.
unwrap – коррекция фазовых углов для построения непрерывных фазовых характеристик.

Округление и определение остатков

ceil – округление в сторону +∞.
fix – округление в сторону нуля.
floor – округление в сторону -∞.
mod – остаток от деления нацело:
M=mod(X,Y)
Функция возвращает M=X-floor(X./Y).*Y
Для M=mod(X,0) функция возвращает M=X
Для M=mod(X,X) функция возвращает M=0
Знак M всегда совпадает со знаком Y для ненулевых Y.
rem – остаток от деления нацело:
M=rem(X,Y)
Функция возвращает M=X-fix(X./Y).*Y
Для M=rem(X,0) функция возвращает M=NaN
Для M=mod(X,X) функция возвращает M=0 при X~=0
Знак M всегда совпадает со знаком X для ненулевых Y при Y~=X.
round – округление в сторону ближайшего целого числа:
если дробная часть числа равна 0.5, то округление производится в сторону удаления от нуля.

Функции дискретной математики

factor – разложение целых неотрицательных чисел на простые множители.
factorial – факториал целых неотрицательных чисел.
gcd – определение наибольших общих делителей.
isprime – проверка, являются ли числа простыми.
lcm – наименьшее общее кратное.

nchoosek – число сочетаний или перечисление сочетаний.
perms – все возможные перестановки.
primes – генерация списка простых чисел.
rat, rats – представление дробного числа как отношение целых чисел.

Функции работы с полиномами

conv – умножение полиномов.
deconv – деление полиномов.
poly – формирование полинома по указанным корням или формирование характеристического полинома матрицы.
polyder – дифференцирование полинома или дробно-рациональной функции.
polyfit – полиномиальная аппроксимация таблично заданной функции.
polyint – аналитическое интегрирование полинома (первообразная полинома).
polyval – вычисление значений полинома.
polyvalm – вычисление значения полинома матричного аргумента.
residue – разложение дробно-рациональной функции на простые дроби или наоборот.
roots – вычисление корней полинома.

Интерполяция

interp1 – одномерная интерполяция.
interp1q – кусочно-линейная одномерная интерполяция.
interp2 – двумерная интерполяция.
interp3 – трёхмерная интерполяция.
interpn – *n*-мерная интерполяция.
pchip – построение кусочно-кубического интерполирующего полинома или табличное представление одномерной кусочно-кубической интерполяции.
ppval – вычисление кусочного полинома.
spline – интерполяция таблично-заданной функции кубическим сплайном
unmkpp – запись в отдельные переменные полей структуры, представляющей кусочный полином.

Преобразование декартовой, цилиндрической и сферической систем координат

cart2pol – преобразование декартовых координат в полярные (цилиндрические).
cart2sph – преобразование декартовых координат в сферические.
pol2cart – преобразование цилиндрических координат в декартовые.
sph2cart – преобразование сферических координат в декартовые.

Нелинейные численные методы

Обыкновенные дифференциальные уравнения с начальными условиями (ODE, IVP)

deval – вычисление уже полученного решения ODE.
ode23, ode45, ode113, ode15s, ode23s, ode23t, ode23tb – решение системы обыкновенных дифференциальных уравнений с начальными условиями.
odeget – получить текущие параметры решателей.
odeset – установить параметры решателей.

Обыкновенные дифференциальные уравнения с задержками по времени (DDE)

dde23 – решение DDE с фиксированными задержками.
ddeget – получить текущие параметры решателей.
ddesd – решение DDE с общим случаем задержек.
ddeset – установить параметры решателей.
deval – вычисление уже полученного решения DDE.

Обыкновенные дифференциальные уравнения с граничными условиями (ODE, BVP)

bvp4c – решение BVP для ODE.

bvpget – получить текущие параметры решателей.

bvpset – установить параметры решателей.

deval – вычисление уже полученного решения BVP для ODE.

Дифференциальные уравнения с частными производными (PDE)

pdepe – решение системы одномерных эллиптических или параболических уравнений с граничными и начальными условиями.

pdeval – вычисление решения PDE по выходным данным функции pdepe.

Оптимизация

fminbnd – отыскание минимума функции одной переменной в фиксированном интервале.

fminsearch – отыскание минимума функции нескольких переменных с использованием метода, не требующего дифференцирования.

fzero – отыскание корня функции одной переменной.

optimget – получить текущие параметры решателей.

optimset – установить параметры решателей.

Численное интегрирование (квадратура)

dblquad – вычисление двойного интеграла.

quad – вычисление определённого интеграла методом Симпсона.

quadl – вычисление определённого интеграла методом Лобатто.

quadv – векторизованная квадратура.

triplequad – вычисление тройного интеграла.

Специальные функции

airy – функции Эйри.

besselh – функции Бесселя третьего рода (функции Ханкеля).

besseli – модифицированные функции Бесселя первого рода.

besselj – функции Бесселя первого рода.

besselk – модифицированные функции Бесселя второго рода.

bessely – функции Бесселя второго рода.

beta – бета-функция.

betainc – неполная бета-функция.

betaln – натуральный логарифм бета-функции.

ellipj – эллиптические функции Якоби.

ellipke – полные эллиптические интегралы первого и второго рода.

erf, erfc, erfcx, erfinv, erfcinv – функции ошибок (интегралы от Гауссова распределения).

expint – интегральная экспонента.

gamma, gammainc, gammaln – гамма-функции.

legendre – присоединённые функции Лежандра.

psi – пси-функции (полигамма-функции).

Математические константы [2]

eps – машинный эпсилон арифметики с плавающей точкой.

i, j – мнимая единица.

inf – бесконечность.

intmax – максимальное значение указанного целочисленного типа.

intmin – минимальное значение указанного целочисленного типа

NaN – не число

pi – число π.
realmax – максимальное положительное число с плавающей точкой.
realmin – минимальное положительное число с плавающей точкой.

Наиболее важные функции для работы с массивами типа char

Формирование массивов символов

blanks – формирование строки символов, состоящей из пробелов.
cellstr – преобразование матрицы символов в массив строковых ячеек.
char – преобразование объекта к типу char.
sprintf – запись форматированных данных в строку символов.
strcat – горизонтальная склейка матриц символов.
strvcat – вертикальная склейка матриц символов.

Идентификация массивов символов

class – создание объекта заданного класса или определение класса указанного объекта.
isa – проверка принадлежности объекта указанному классу (типу).
iscellstr – проверка, является ли объект массивом строковых ячеек.
ischar – проверка, является ли объект массивом типа char.
isletter – проверка элементов массива типа char на принадлежность к категории букв алфавита.
isscalar – проверка, является ли объект скаляром.
isspace – проверка элементов массива типа char на равенство пробелу.
isstrprop – проверка элементов массива типа char на принадлежность указанной категории.
isvector – проверка, является ли массив одномерным.

Манипуляции строками символов

deblank – удаление пробелов и других пустых символов из конца строки или из каждой строки массива строковых ячеек.
lower – приведение букв к нижнему регистру.
strjust – выравнивание строк.
strrep – поиск и замена подстроки.
strtrim – удаление начальных или конечных пустых символов.
upper – приведение букв к верхнему регистру.

Анализ строк символов

findstr – поиск подстроки.
strfind – поиск подстроки.

Вычисление строк

eval – выполнение строки как выражения или оператора MATLAB.
evalin – выполнение (вычисление) выражения MATLAB в указанной рабочей области.

Сравнение строк

strcmp, strcmpi – сравнение строк.

Логические функции

all – true, если все элементы массива вдоль указанной размерности ненулевые.
and – операция «логическое и».
any – true, если все хотя бы один элемент массива вдоль указанной размерности ненулевой.
false – логический нуль.
find – поиск индексов ненулевых элементов.

iskeyword – проверка, является ли строка символов ключевым словом MATLAB.
isvarname – проверка, является ли строка символов именем переменной MATLAB.
logical – преобразование числовых значений к логическому типу.
not – операция «логическое отрицание».
or – операция «логическое или».
true – логическая единица.
xor – операция «логическое исключающее или».

Функции операций отношения

eq – операция ==
ge – операция >=
gt – операция >
le – операция <=
lt – операция <
ne – операция ~=

Дата и время

calendar – календарь для указанного месяца.
clock – текущее состояние компьютерных часов (см. примеры применения функций для работы с массивами).
cputime – процессорное время, прошедшее с момента запуска MATLAB.
date – текущая дата в виде строки символов.
weekday – день недели.

Некоторые примеры применения встроенных функций MATLAB

Пусть имеется идеализированный диод, участок ВАХ которого в открытом состоянии представлен таблицей. В MATLAB эта таблица представлена двумя массивами-строками: U – массив значений напряжения в порядке возрастания, I – массив значений тока. Если напряжение на зажимах диода меньше U(1), то диод считается закрытым, т.е протекающий через него ток равен нулю. Экспериментально исследуем стандартные методы интерполяции ВАХ, поддерживаемые функцией intrep1. Ещё посмотрим, как эта функция будет экстраполировать ВАХ за пределы диапазона напряжений, представленного массивом U. Возьмём из справочника любой диод, приближённо считая, что его свойства являются статическими.

```
U=[0.28 0.4 0.48 0.5]; % Массив напряжений из справочника, В
I=[0 100 500 600]; % Массив токов из справочника, мА
```

Если график этой функции мы построим функцией plot, то получим ломаную линию в поле axes (рис.8):

```
plot(U,I,'k-','linewidth',2)
```

Параметр 'k-' означает чёрный цвет линии (буква k) и сплошной стиль линии (знак -). Пара параметров 'linewidth',2 означает, двойную толщину линии.

Оператор

```
grid on
```

нанесёт на поле axes графика координатную сетку.

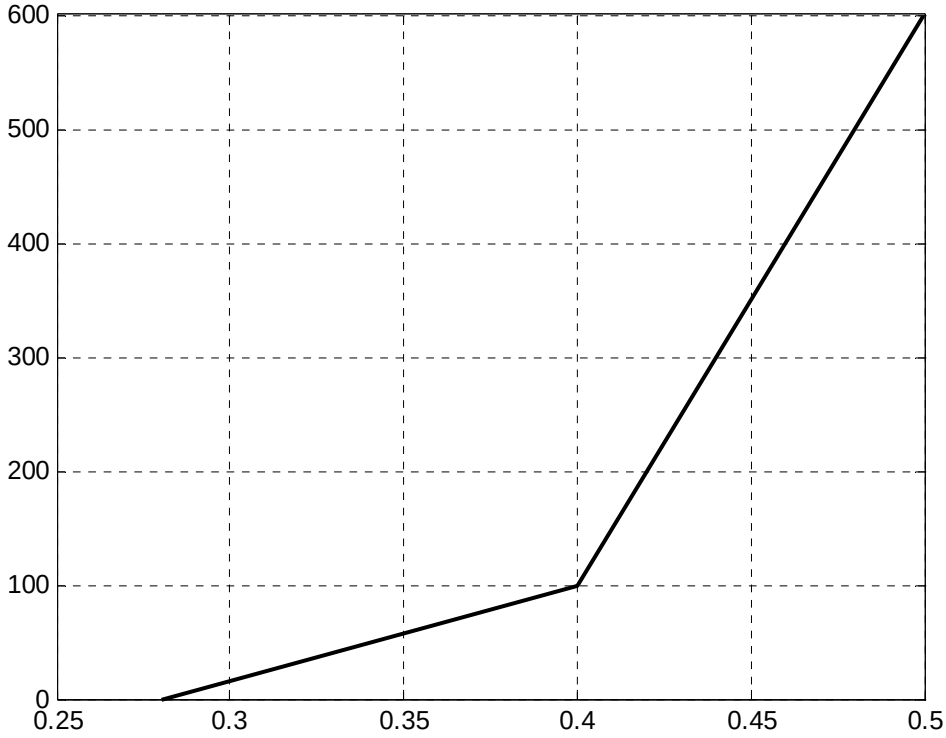

Рис. 8. График ВАХ без интерполяции

Перед интерполяцией добавим в таблицу ВАХ нулевую точку:

```
U0=[0 U];
I0=[0 I];
```

Построим равномерную сетку напряжений в диапазоне от -20 до +0.6 В. Пусть в этой сетке будет 2001 точка:

```
Ui=linspace(-20,0.6,2001);
```

Посмотрев описание функции interp1 по справочной системе, увидим, что эта функция поддерживает следующие методы интерполяции:

'nearest' – кусочно-постоянная интерполяция;

'linear' – кусочно-линейная интерполяция (по умолчанию);

'spline' – кубическая сплайновая интерполяция;

'pchip' или 'cubic' – кусочно-кубическая Эрмитова интерполяция;

'v5cubic' – кусочно-кубическая интерполяция, используемая в MATLAB 5 (экстраполяция невозможна).

Метод 'nearest' даёт разрывную ВАХ, поэтому его использовать не будем. 'v5cubic' не позволяет экстраполировать функции, поэтому её тоже использовать не будем.

```
Ii=interp1(U0,I0,Ui,'linear','extrap');
plot(Ui,Ii,'k-','linewidth',2)
grid on
```

Данная последовательность операторов приведёт к построению графика, показанного на рис. 9. Такая характеристика годится для моделирования процессов в диодных выпрямителях.

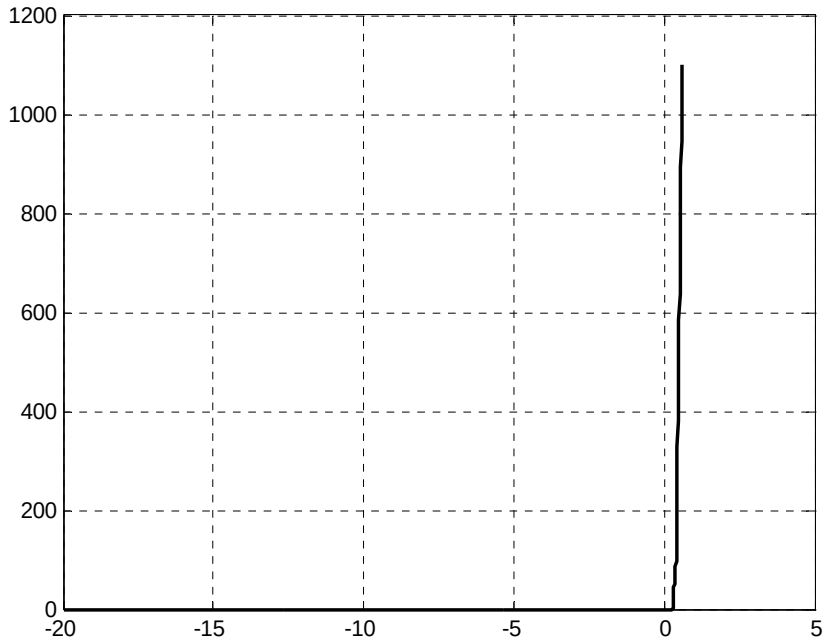

Рис. 9. Кусочно-линейно интерполированная и экстраполированная ВАХ диода

```
Ii=interp1(U0,I0,Ui,'spline','extrap');
plot(Ui,Ii,'k-','linewidth',2)
grid on
```

Данная последовательность операторов приведёт к построению графика, показанного на рис. 10. Такая характеристика не годится для моделирования процессов в диодных выпрямителях.

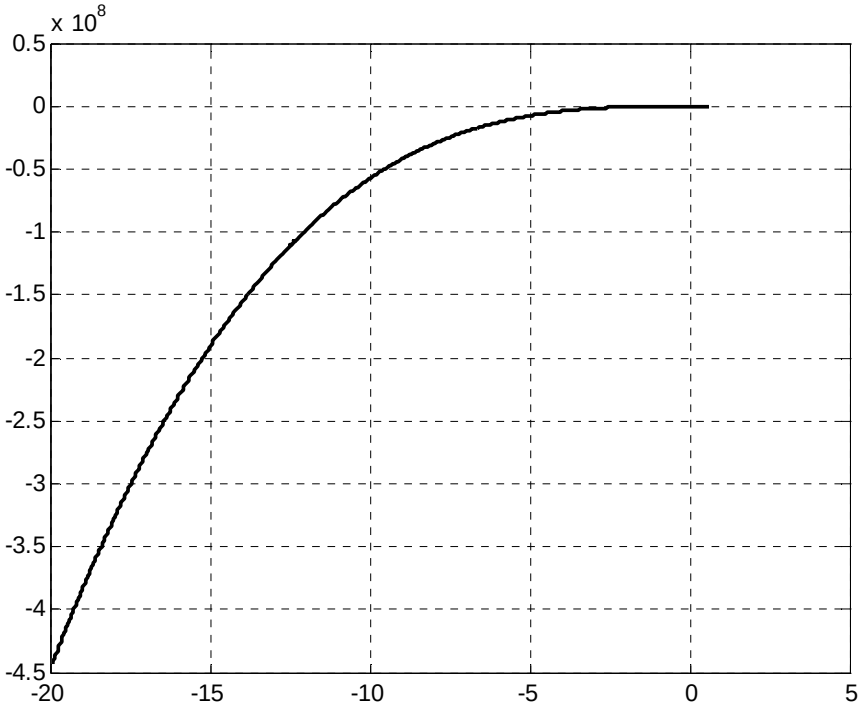

Рис. 10. ВАХ диода, экстраполированная кубическим сплайном

Оператором

```
axis([0 0.6 -50 800])
```

покажем участок ВАХ в области положительных напряжений (рис. 11). Получили немонотонную характеристику, хотя физически она является монотонной. Здесь мы убедились в проявлении численной неустойчивости метода сплайновой интерполяции применительно к данной конкретной ВАХ.

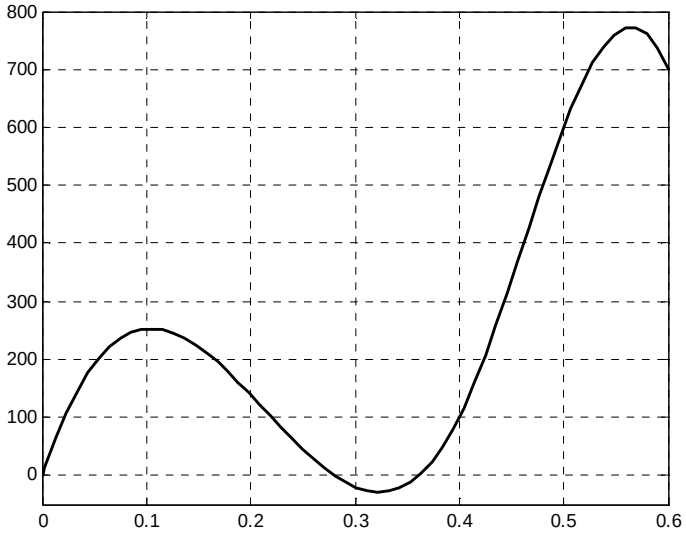

Рис. 11. ВАХ диода, интерполированная кубическим сплайном в области положительных напряжений

```
Ii=interp1(U0,I0,Ui,'cubic','extrap');
plot(Ui,Ii,'k-','linewidth',2)
grid on
```

Данная последовательность операторов приведёт к построению графика, показанного на рис. 12. Такая характеристика годится для моделирования процессов в диодных выпрямителях.

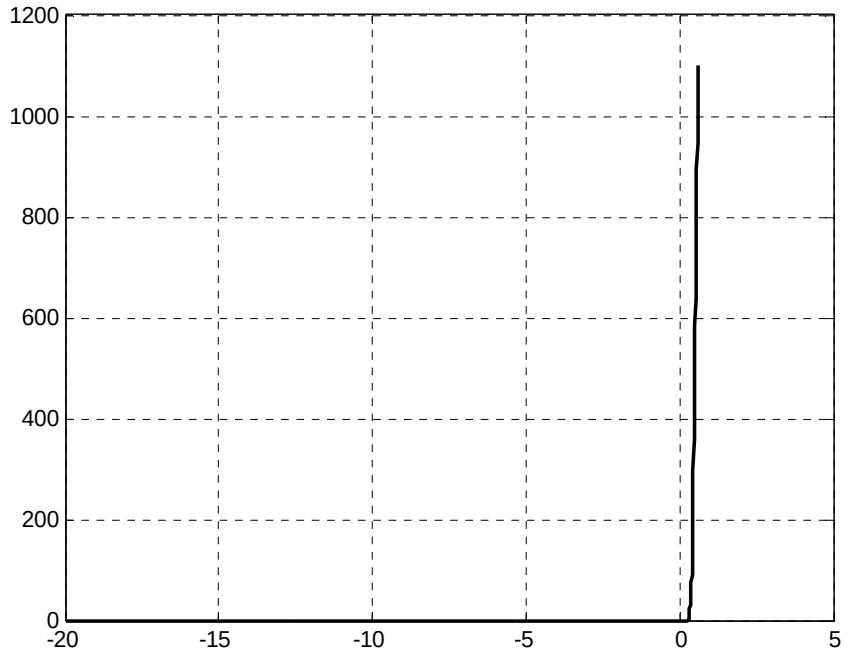

Рис. 12. Кусочно-кубически экстраполированная ВАХ диода

Оператором
axis([0 0.6 0 1100])
покажем участок ВАХ в области положительных напряжений (рис. 13). Видна устойчивость метода кусочно-кубической интерполяции.

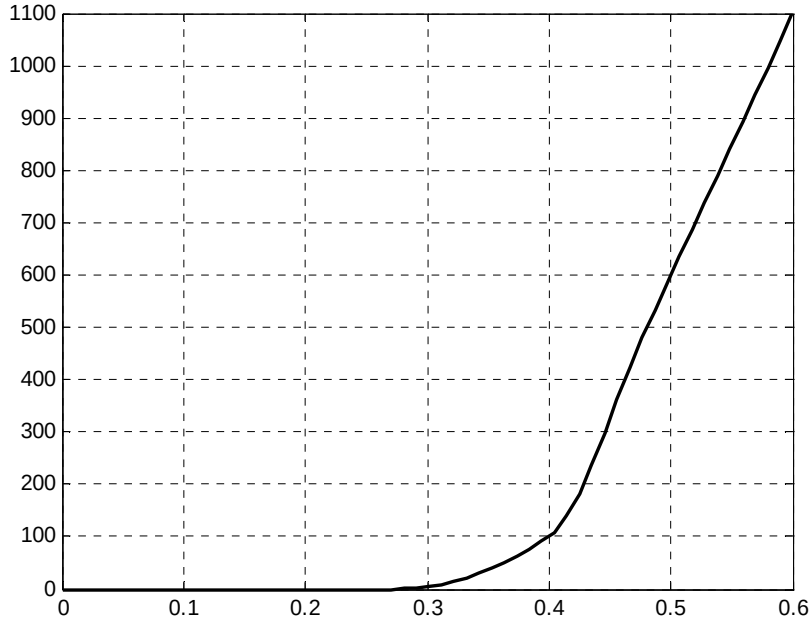

Рис. 13. Кусочно-кубически интерполированная ВАХ диода в области положительных напряжений

Пусть этот диод служит в качестве однополупериодного выпрямителя (рис. 14). ЭДС источника изменяется во времени по синусоидальному закону с амплитудой 20 В. Сопротивление нагрузки $R_н$ = 20 Ом = 0.02 кОм. С помощью функции interp1 рассчитаем и построим временную диаграмму напряжения на нагрузке $u_н(t)$ для трёх периодов изменения ЭДС источника.

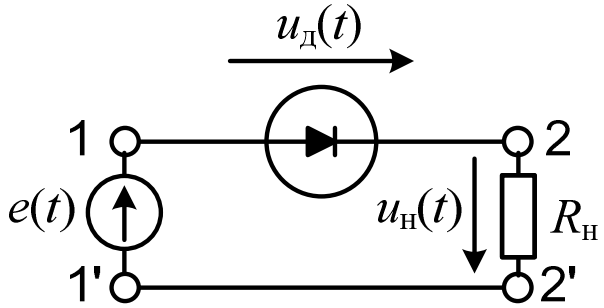

Рис. 14. Однополупериодный выпрямитель, работающий на резистивную нагрузку

В соответствии с законом Ома и вторым законом Кирхгофа

$$u_н(t) = R \cdot i(t); \qquad e(t) = u_н(t) + u_д(t),$$

поэтому для расчёта и построения временной диаграммы выполним следующую последовательность операторов.

```
R=0.02; % Сопротивление нагрузки, кОм
Un=R*Ii; % Массив напряжений нагрузки для функции преобразования напряжений
Ei=Ui+Un; % Массив значений ЭДС для функции преобразования напряжений
t=linspace(0,3,1501); % Массив значений времени, делённых на период
```

```
ei=20*sin(2*pi*t); % Массив ЭДС в разные моменты времени
ui=interp1(Ei,Un,ei,'cubic','extrap'); % Массив напряжений нагрузки в разные моменты времени
plot(t,ei,'k-',t,ui,'r-','linewidth',2)
xlabel('время, делённое на период')
ylabel('ЭДС и напряжение нагрузки, В')
grid on
```

В результате будет построена совмещённая временная диаграмма ЭДС источника и напряжения на нагрузке (рис. 15).

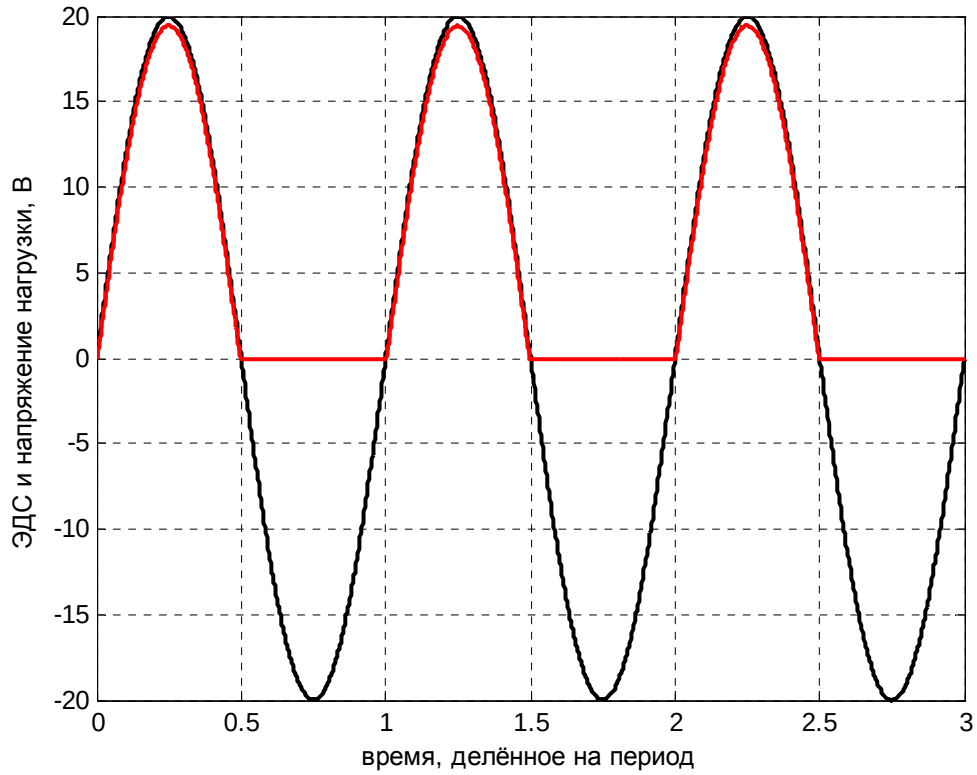

Рис. 15. Временная диаграмма ЭДС и напряжения нагрузки

Пусть имеется группа симметричных трёхфазных приёмников электроэнергии, питаемых от общей сети низкого напряжения. Даны номинальные полные мощности приёмников и их номинальные коэффициенты мощности. Линейное напряжение равно 380 В. Каждый приёмник питает отдельная кабельная линия с погонным комплексным сопротивлением каждого провода (0.3+0.1j) Ом/км. Длина каждой линии дана. Каждый из приёмников работает совместно с конденсаторной установкой (КУ). Даны значения тангенса угла разности фаз между напряжением и током при совместной работе приёмника и КУ без учёта потерь активной мощности в КУ. Конденсаторные установки работают в непосредственной близости от приёмников, и для них даны тангенсы угла диэлектрических потерь. Определить потери активной и реактивной мощности в каждой линии и во всех линиях, потери активной мощности в каждой КУ и во всех КУ.

Операторы, определяющие исходные данные задачи:

```
clo=clock;
rand('state',clo(end)/60*2^32);
N=11;  % Число приёмников в группе
S=10*rand(1,N)  % Номинальные полные мощности приёмников, кВА
```

```
cofi=0.9+0.1*rand(1,N)  % Номинальные коэффициенты мощности приёмников
taficomp=0.1+0.01*randn(1,N);  % тангенсы угла разности фаз . . .
Dli=100*rand(1,N);  % Длины питающих линий, м
tade=0.01*rand(1,N);  % Тангенсы углов диэлектрических потерь установок
```

Задача решается путём выполнения следующей последовательности операторов:

```
P=S.*cofi % Номинальные активные мощности приёмников, кВт
Q=sqrt(S.^2-P.^2) % Номинальные реактивные мощности приёмников, кВАр
Qcomp=P.*taficomp % Реактивные мощности приёмников после компенсации, кВАр
Qcond=Q-Qcomp % Реактивные мощности, генерируемые КУ, кВАр
Pcond=Qcond.*tade % Активные мощности, теряемые в конденсаторных установках, кВт
Pco=P+Pcond % Активные мощности приёмников вместе с потерями в КУ, кВт
Sco=sqrt(Pco.^2+Qcomp.^2) % Полные мощности приёмников вместе с КУ, кВА
Uli=380; % Линейное напряжение приёмников, В
Ili=Sco/Uli/sqrt(3) % Токи подводящих линий, кА
Zli=0.3+0.1i; % Погонное комплексное сопротивление проводов линий, Ом/км
Sli=3*Zli*Dli.*Ili.^2 % Комплексные мощности потерь в линиях, кВА
Pconda=sum(Pcond) % Активная мощность потерь во всех КУ, кВт
Slia=sum(Sli) % Комплексная мощность потерь во всех линиях, кВА
```

В командное окно будут выданы результаты выполнения операторов:

```
P =
  Columns 1 through 9
     2.1707       6.6241       8.2349       7.3415       6.6003       3.5163       1.5365       4.5856
4.6962
  Columns 10 through 11
     3.9197       3.794
Q =
  Columns 1 through 9
    0.51822       1.8058      0.92258       3.2906       2.4354       1.3929      0.38392      0.60841
1.9143
  Columns 10 through 11
     1.6076      0.66655
Qcomp =
  Columns 1 through 9
    0.20768      0.55208      0.83381      0.75526      0.58436      0.39351      0.17192      0.45683
0.48499
  Columns 10 through 11
    0.39882      0.37231
Qcond =
  Columns 1 through 9
    0.31054       1.2537      0.08877       2.5354       1.8511      0.99935        0.212      0.15157
1.4293
  Columns 10 through 11
     1.2088      0.29424
Pcond =
  Columns 1 through 8
```

```
   0.0023666    0.0013924   0.00022933    0.0013979     0.018153    0.0084484   0.00084917
0.00023822
  Columns 9 through 11
   0.0093104    0.0055992    0.0013363
Pco =
  Columns 1 through 8
      2.1731       6.6255       8.2351       7.3428       6.6185       3.5248       1.5373       4.5858
  Columns 9 through 11
      4.7055       3.9253       3.7953
Sco =
  Columns 1 through 8
       2.183       6.6485       8.2772       7.3816       6.6442       3.5467       1.5469       4.6085
  Columns 9 through 11
      4.7305       3.9455       3.8135
Ili =
  Columns 1 through 8
   0.0033167     0.010101     0.012576     0.011215     0.010095    0.0053886    0.0023503
0.0070019
  Columns 9 through 11
   0.0071872    0.0059946    0.0057941
Sli =
  Columns 1 through 4
  0.00033454 + 0.00011151i    0.0087707 +  0.0029236i    0.0059292 +  0.0019764i
0.0026678 + 0.00088928i
  Columns 5 through 8
   0.0059192 +  0.0019731i   0.00031647 + 0.00010549i   0.00032854 + 0.00010951i
0.0042235 +  0.0014078i
  Columns 9 through 11
   0.0040015 +  0.0013338i     0.002084 + 0.00069465i  4.5104e-005 +1.5035e-005i
Pconda =
    0.049321
Slia =
     0.03462 +    0.01154i
```

Активная мощность потерь в конденсаторных установках составила 49.321 Вт. Активная мощность потерь в линиях 34.62 Вт, реактивная мощность потерь 11.54 ВАр.

Наиболее важные функции анализа данных

Математическая статистика

corrcoef – коэффициенты корреляции.

cov – ковариация.

mean – среднее арифметическое массива вдоль указанной размерности.

std – стандартная девиация (среднеквадратичное отклонение от среднего).

var – квадрат стандартной девиации (дисперсия).

Производные и интегралы

cumtrapz – кумулятивное численное интегрирование методом трапеций.

diff – конечная разность массива вдоль заданной размерности.

gradient – численное определение градиента.

trapz – численное интегрирование методом трапеций.

m-файлы

Часто используемые последовательности операторов MATLAB можно сохранять в виде текстовых файлов с расширением m (имя.m).

Существует два типа m-файлов:

- вычислительные сценарии (script-файлы),
- m-функции.

Script-файл – это просто последовательность операторов MATLAB. Эти файлы не позволяют передавать входные и выходные параметры. Все переменные создаются и обрабатываются в главной рабочей области MATLAB. После завершения выполнения m-файла все созданные переменные и данные остаются в главной рабочей области MATLAB и доступны для дальнейшей обработки.

m-функции позволяют передавать входные и выходные параметры. Переменные создаются и обрабатываются в локальной рабочей области, которая уничтожается после завершения работы m-функции. В некоторых случаях возможно сохранение отдельных переменных.

Возможна также работа с глобальными переменными.

m-файл может вначале содержать комментарии. Если эти комментарии оформлены по стандартной форме, то они могут обрабатываться справочной системой MATLAB. После этих комментариев могут следовать операторы MATLAB. Комментарии, которые могут создаваться в этой части m-файла, не обрабатываются справочной системой.

Комментарии в начале m-файла записываются в следующем виде:

% имя - назначение файла

% Справочная описательная последовательность комментариев

Если это script-файл, то дальше просто следуют операторы MATLAB. Если это m-функция, то первый оператор должен быть заголовком функции:

function [выхпараметры]=имя(вхпараметры)

function – ключевое слово заголовка;

выхпараметры – список выходных формальных параметров, разделённых запятыми;

вхпараметры – список входных формальных параметров, разделённых запятыми.

Синтаксис входных и выходных формальных параметров такой же, как у имён переменных MATLAB. Если выходной параметр один, то в этом списке квадратные скобки можно опустить. Если выходных параметров нет, то список и знак равенства отсутствуют. Если входных параметров нет, то список и круглые скобки должны отсутствовать.

Ниже представлен текст m-функции, которая определяет матрицу главных сечений Q и матрицу главных контуров электрической цепи по известной матрице узловых соединений A. У этой функции имя getqbm, три входных и два выходных параметра.

```
% getqbm - Функция вычисляет топологические матрицы Q, B по известной матрице A
% [Q,B] = getqbm(A,D,S)
% A - матрица узловых соединений цепи;
% D - массив номеров ветвей дерева;
% S - массив номеров ветвей связи;
function [Q,B] = getqbm(A,D,S)
[d,v]=size(A); % d - число ветвей дерева; v – число ветвей
s=v-d; % число ветвей связи (главных контуров)
% Проверим размеры списков ветвей дерева
```

```
% Если они неверные, то выходным параметрам присвоим NaN
% и вернём управление вызывающей программе
if (length(D)~=d)|(length(S)~=s), Q=NaN; B=NaN; return; end;
% Проверим, правильно ли заданы ветви дерева
% Если нет, то выходным параметрам присвоим матрицы размеров (d,v) и (s,v)
% и вернём управление вызывающей программе
if det(A(:,D))==0 % если в заданном дереве есть хотя бы один контур
 Q=repmat(NaN,d,v); B=repmat(NaN,s,v);
 return; % возврат управления в вызывающую программу
end; % if det(A(:,D))==0
Q(1:d,[D,S])= A(:,D)\A(:,[D,S]);
B(1:s,[S,D])=[eye(s),-Q(:,S).'];
```

Пример составления m-файла разложения трёхфазной системы токов на симметричные составляющие

Пусть имеется несимметричная трёхфазная система токов:

```
IA=(5+5*rand(1,1))*exp(5i*randn(1,1)*pi/180); % Ток 1-й фазы, А
IB=(5+5*rand(1,1))*exp((-120i+5i*randn(1,1))*pi/180); % Ток 2-й фазы, А
IC=(5+5*rand(1,1))*exp((120i+5i*randn(1,1))*pi/180); % Ток 3-й фазы, А
```

Определить симметричные составляющие прямой, обратной и нулевой последовательности трёхфазной системы токов в фазе А.

Ниже представлен текст m-функции, возвращающей симметричные составляющие.

```
% s_sost_3ph – разложение трёхфазной системы на симметричные составляющие
% [X1A,X2A,X0]=s_sost_3ph(XA,XB,XC)
% XA - комплексный ток или напряжение первой фазы (A)
% XB - комплексный ток или напряжение второй фазы (B)
% XC - комплексный ток или напряжение третьей фазы (C)
% X1A - симметричная составляющая прямой последовательности для фазы A
% X2A - симметричная составляющая обратной последовательности для фазы A
% X0 - симметричная составляющая нулевой последовательности
function [X1A,X2A,X0]=s_sost_3ph(XA,XB,XC)
a1=exp(2i*pi/3); % комплексный оператор поворота
a2=conj(a1); % комплексно сопряженный оператор поворота
X1A=(XA+a1*XB+a2*XC)/3; % составляющая прямой последовательности
X2A=(XA+a2*XB+a1*XC)/3; % составляющая обратной последовательности
X0=(XA+XB+XC)/3; % составляющая нулевой последовательности
```

Для проверки правильности работы m-функции выполним следующую последовательность операторов:

```
IA=(5+5*rand(1,1))*exp(5i*randn(1,1)*pi/180); % Ток 1-й фазы, А
IB=(5+5*rand(1,1))*exp((-120i+5i*randn(1,1))*pi/180); % Ток 2-й фазы, А
IC=(5+5*rand(1,1))*exp((120i+5i*randn(1,1))*pi/180); % Ток 3-й фазы, А
disp('Фазные токи:')
disp([num2str(IA,'%0.5g'),', ',num2str(IB,'%0.5g'),', ',num2str(IC,'%0.5g')])
[I1A,I2A,I0]=s_sost_3ph(IA,IB,IC)
```

Результаты выполнения последовательности операторов будут выведены в командное окно:

```
Фазные токи:
9.7857-0.053847i, -4.6394-7.7137i, -4.3225+5.6437i
I1A =
      8.6115 +    0.23557i
I2A =
      0.8996 +    0.41854i
I0 =
     0.27459 -    0.70795i
```