Возможности символьного вычисления в среде MatLab

Презентацию подготовили
студенты группы Эсн/б-35-о:
Максименко Т.С.
Малиновский К.Н.

Что такое символьные выражения?

В процессе символьных вычислений используются переменные и константы особого типа, так называемые символьные объекты. Хотя обычно в коде MATLAB тип переменных определяется динамически и нет нужды объявлять его явно, для символьных объектов дело обстоит иначе. Для объявления символьных переменных служит команда *syms*, которая в качестве аргументов принимает имена переменных, перечисленные через пробел. Например, так:

```
>> syms x y
>> syms a b real % объявляемые объекты обозначают вещественные переменные
```

Объявление символьных констант осуществляется при помощи функции *sym*. Она может принимать в качестве аргумента строку, содержащую специальные переменные, численное выражение или вызов функции, как в примерах ниже:

```
>> sym_pi = sym('pi')
>> sym_delta = sym('1/10')
>> sym_sqroot2 = sym('sqrt(2)')
```

Использование символьных констант полезно тем, что вычисления с ними производятся точно (т.е. без вычислительных погрешностей) до тех пор, пока не потребуется вычислить некоторое числовое значение. Заметим, что при выводе содержимого рабочего пространства командой *whos* символьные переменные и константы отображаются как представители класса *sym object*.

После объявления, с символьными переменными можно обращаться примерно так же, как и с обычными числовыми. В частности, для них определены операторы + - * / ^, с помощью которых можно составлять символьные выражения:

```
>> syms s t A
>> f = s^2 + 4*s + 5
f =
s^2 + 4*s + 5
>> g = s + 2
g =
s + 2
>> h = f*g
h =
(s^2 + 4*s + 5)*(s + 2)
>> z = exp(-s*t)
z =
exp(-s*t)
>> y = A*exp(-s*t)
y =
A*exp(-s*t)
```

В приведенных командах символьные переменные s, t, A используются для составления символьных выражений, создавая новые символьные переменные f, g, h, z, y. При этом последние автоматически объявляются символьными и их значение не вычисляется и никак не преобразуется.

Примеры

- **Раскрытие скобок**

```
>> syms s;
>> A = s + 2;
>> B = s + 3;
>> C = A*B
C =
(s+2)*(s+3)
>> C = expand(C)
   C =
   s^2 + 5*s + 6
```

- **Разложение на множители**

```
>> syms s;
>> D = s^2 + 6*s + 9;
>> D = factor(D)
D =
(s+3)^2
>> p = s^3 - 2*s^2 - 3*s + 10;
>> P = factor(p)
P =
(s+2)*(s^2 - 4*s + 5)
```

- **Сокращение общего множителя**

```
>> syms s;
>> H = (s^3 + 2*s^2 + 5*s + 10)/(s^2 + 5);
>> H = simplify(H)
H =
s+2
>> factor(s^3 + 2*s^2 + 5*s + 10)
ans =
(s+2)*(s^2 + 5)
```

- **Подстановка переменной**

  Путь имеется выражение $H(s) = \dfrac{s+3}{s^2+6s+8}$ и требуется вычислить $G(s) = H(s)|_{s=s+2}$.

```
>> syms s;
>> H = (s + 3)/(s^2 + 6*s + 8);
>> G = subs(H, s, s+2)
G =
(s+5)/((s+2)^2 + 6*s + 20)
>> G = collect(G)
G =
(s+5)/(s^2 + 10*s + 24)
```

  Таким образом, $G(s) = \dfrac{s+5}{s^2+10s+24}$.

4.2.1 Вычисления значения выражения

`double(S)` Преобразует символьную матрицу S, заменяя ее элементы их численными значениями. Матрица S не должна содержать символьных переменных

Как следует из описания функции, в символическом выражении перед вычислением его численного значения необходимо сделать все возможные подстановки, чтобы избавиться от свободных символьных переменных. Рассмотрим пример:

```
>> syms s;
>> E = s^3 -14*s^2 + 65*s - 100);
>> F = subs(E, s, 7.1)
F =
13671/1000
>> G = double(F)
G =
13.6710
```

4.2.2 Рисование при помощи функции *ezplot*

Символическое выражение может быть нарисовано непосредственно при помощи функции *ezplot*:

`ezplot(f)`	Рисует график функции $f(x)$, где f является математической функцией переменной x. Интервал, на котором изображается график, по умолчанию равен $[-2\pi, 2\pi]$.
`ezplot(f,xmin,xmax)`	Рисует график на заданном интервале.

Например, график кубического многочлена

$$A(s) = s^3 + 4s^2 - 7s - 10$$

на интервале $[-1, 3]$ можно нарисовать следующим образом:

```
>> syms s;
>> A = s^3 + 4*s^2 - 7*s - 10;
>> ezplot(A, -1, 3), ylabel('A(s)')
```

Отметим, что название оси абсцисс и всего графика определяется автоматически. Они могут быть изменены обычным способом (см. команды *xlabel, title*).

4.3 Решение уравнений и систем уравнений

4.3.1 Решение уравнений при помощи функции *solve*

MATLAB можно использовать для решения алгебраических и трансцендентных уравнений и систем уравнений, заданных в виде массива символьных выражений. Основной инструмент для этого - функция *solve*. Она может вызываться в разной форме:

```
solve(E1, E2, ..., EN)
solve(E1, E2, ..., EN, var1, var2, ..., varN)
```

Здесь *E1, E2, ..., EN* — символьные выражения или пременные, в которых они содержаться, а *var1, ..., varN* — перменные, относительно которых следует разрешить систему уравнений *E1=0, E2=0, ..., EN=0*. Разумеется, первая форма вызова этой функции допустима лишь в случае, когда нет неоднозначностей относительно того, что именно следует найти. Функция *solve* возвращает единственное символьное выражение, если уравнение (система уравнений) имеет единственное решение и вектор решений в противном случае. Если уравнение содержит периодические функции и может поэтому иметь бесконечное число решение, функция ограничивается тем, что возвращает корни за один период в окрестности нуля.

Рассмотрим примеры.

- **Единственное решение**

```
>> syms s;
>> E = s + 2;
>> s = solve(E);
s =
-2
```

- **Несколько решений**

```
>> syms s;
>> D = s^2 + 6*s + 9;
>> s = solve(D)
s =
[  -3]
[  -3]
```

- **Уравнение с параметрами**

```
>> syms theta x z;
>> E = z*cos(theta) - x;
>> theta = solve(E, theta)
theta =
acos(x/z)
```

- **Уравнение с комплексными корнями**

```
>> syms x;
>> E = exp(2*x) + 4*exp(x) - 32;
>> x = solve(E)
x =
[ log(-8)]
[  log(4)]
>> log(-8)
ans =
   2.0794 + 3.1416i
>> log(4)
ans =
   1.3863
```

- **Уравнение с комплексными корнями**

```
>> syms x;
>> E = exp(2*x) + 4*exp(x) - 32;
>> x = solve(E)
x =
[ log(-8)]
[  log(4)]
>> log(-8)
ans =
   2.0794 + 3.1416i
>> log(4)
ans =
   1.3863
```

- **Уравнение с бесконечным числом решений**

```
>> E = cos(2*theta) - sin(theta);
>> solve(E)
theta =
[ -1/2*pi]
[  1/6*pi]
[  5/6*pi]
```

Напомним, что численное (т.е. приближенное) значение решения можно получить, воспользовавшись функцией *double*.

4.4.1 Дифференцирование

Чтобы найти производную символьного выражения, следует воспользоваться одной из следующих форм функции *diff*:

`diff(S)`	Дифференцирует выражение S по независимым переменным, определенным функцией *findsym*
`diff(S, t)`	Дифференцирует выражение S по переменной t
`diff(S,n)`	n раз дифференцирует выражение S
`diff(S, t, n)`	n раз дифференцирует выражение S по переменной t

Приведем несколько простых примеров.

```
>> syms s n;
>> p = s^3 + 4*s^2 - 7*s - 10;
>> d = diff(p)
d =
3*s^2+8*s-7
>> e = diff(p, 2)
e =
6*s+8
>> g = s^n;
>> h = diff(g)
h =
s^n*n/s
>> h = simplify(h)
h =
s^(n-1)*n
>>
>> f = exp(-(x^2)/2);
>> diff(f)
ans =
-x*exp(-1/2*x^2)
```