Лекция 3. Пакет GNU Octave.

Типы данных

Диапазоны

Класс `single` и типы `float`

Целые типы данных

Целочисленная арифметика

Логический тип данных

Арифметические операции с данными разных типов

Проверки типа числовых объектов

Строковые данные

Переменные

Статус переменных

Диапазоны

Тип данных диапазон (range) позволяет определить сложную константу, представляющую собой серию числовых констант значения которых даются формулой $a_{i+1} = a_i + h$.

Константа типа диапазон задается выражением $a_1:h:a_{max}$, где a_1 — значение первого элемента, h — необязательный шаг (по умолчанию имеет значение 1), a_{max} — максимально допустимое значение элемента серии. Можно использовать любые арифметические выражения и вызовы функци для задания a_1, h и a_{max}.

Отметим важную особенность. По умолчанию `octave` не преобразует константу диапазон в матричные типы данных (вектор строка), что

позволяет не выделять память для диапазонов с большим числом элементов, например `1:10000`.

На отсутствие такого преобразования указывает значение `1 (true)`, выдаваемое функцией `optimize_range()` без аргументов. Вывод этой функцией значение `0 (false)` указывает что в `octave` установлен режим автоматического преобразования констант диапазона к матричному типу — вектору строке. Переключение режима осуществляется выполнением функции `optimize_range` с аргументом `1` или `0`. Пользователь всегда может явно преобразовать константу диапазона к матричному типу задав ее внутри символов `[]`.

Пример.

```
ybgv@ybgv-home:~> octave -q
octave:1>
octave:1> optimize_range ()
ans = 1
octave:2> # нет преобразования к матричному типу
octave:2>
octave:2> x=1:5
x =

   1   2   3   4   5

octave:3> size(x),typeinfo(x),class(x)
ans =

   1   5

ans = double_range
ans = double
octave:4> y=1:3:5
y =

   1   4

octave:5> size(y),typeinfo(y),class(y)
ans =

   1   2

ans = double_range
```

```
ans = double
octave:6> z=0:0.1:2
z =

 Columns 1 through 7:

           0     0.1000     0.2000     0.3000     0.4000     0.5000
0.6000

 Columns 8 through 14:

     0.7000     0.8000     0.9000     1.0000     1.1000     1.2000
1.3000

 Columns 15 through 21:

     1.4000     1.5000     1.6000     1.7000     1.8000     1.9000
2.0000

octave:7> size(z),typeinfo(z),class(z)
ans =

    1   21

ans = double_range
ans = double
octave:8> w=[0:0.1:2]
w =

 Columns 1 through 8:

         0    0.1000    0.2000    0.3000    0.4000    0.5000    0.6000
0.7000

 Columns 9 through 16:

    0.8000    0.9000    1.0000    1.1000    1.2000    1.3000    1.4000
1.5000

 Columns 17 through 21:

   1.6000   1.7000   1.8000   1.9000   2.0000

octave:9> size(w),typeinfo(w),class(w)
ans =

    1   21
```

```
ans = matrix
ans = double
octave:10> whos
Variables visible from the current scope:

variables in scope: top scope

  Attr   Name        Size                     Bytes  Class
  ====   ====        ====                     =====  =====
         ans         1x6                          6  char
         w           1x21                       168  double
         x           1x5                         24  double
         y           1x2                         24  double
         z           1x21                        24  double

Total is 55 elements using 246 bytes

octave:11>  optimize_range  (0)    #  включить  автопреобразовние  к
матричному типу
octave:12> z=0:0.1:2
z =

 Columns 1 through 8:

         0    0.1000    0.2000    0.3000    0.4000    0.5000    0.6000
0.7000

 Columns 9 through 16:

    0.8000    0.9000    1.0000    1.1000    1.2000    1.3000    1.4000
1.5000

 Columns 17 through 21:

   1.6000   1.7000   1.8000   1.9000   2.0000

octave:13> size(z),typeinfo(z),class(z)
ans =

    1   21

ans = matrix
ans = double
octave:15> whos
Variables visible from the current scope:
```

```
variables in scope: top scope

  Attr   Name        Size                     Bytes  Class
  ====   ====        ====                     =====  =====
         ans         1x6                          6  char
         w           1x21                       168  double
         x           1x5                         24  double
         y           1x2                         24  double
         z           1x21                       168  double

Total is 55 elements using 390 bytes

octave:16>
```

Класс `single` и типы `float`

В стандарте IEEE 754 определены и 32 битовые числа с плавающей запятой обычной точности. Преобразование данных к этому типу выполнят функция `single(expr)`.

Пример.

```
ybgv@ybgv-home:~> octave -q
octave:1>
octave:1> x=5.2
x = 5.2000
octave:2> class(x),typeinfo(x)
ans = double
ans = scalar
octave:3> y=single(x)
y = 5.2000
octave:4> format long
octave:5> x
x = 5.200000000000000
octave:6> y
y = 5.1999998
octave:7> class(y),typeinfo(y)
ans = single
ans = float scalar
octave:8> A=[3.5,4.6;2.3,3.4]
A =

   3.500000000000000   4.600000000000000
```

```
   2.300000000000000   3.400000000000000

octave:9> class(A),typeinfo(A)
ans = double
ans = matrix
octave:10> B=single(A)
B =

   3.5000000   4.5999999
   2.3000000   3.4000001

octave:11> class(B),typeinfo(B)
ans = single
ans = float matrix
octave:12> whos
Variables visible from the current scope:

variables in scope: top scope

  Attr   Name        Size                     Bytes  Class
  ====   ====        ====                     =====  =====
         A           2x2                         32  double
         B           2x2                         16  single
         ans         1x12                        12  char
         x           1x1                          8  double
         y           1x1                          4  single

Total is 22 elements using 72 bytes

octave:13>
```

Целые типы данных

В octave поддерживаются знаковые и беззнаковые целые длиной nn битов, где nn принимает значения 8, 16, 32, или 64. Функции intnn(expr) преобразуют expr к целому со знаком соответствующей длины, а untnn(expr) — к целому без знака. Функции intmax ("type") и intmin ("type") где "type" строка вида "intnn" или "uintnn" с конкретно указанным значением nn, даст соответственно наибольшее или наименьшее значения величин указанного целого типа. Функции

intmax(expr) и intmin(expr) дадут такие значения для типа, который имеет выражение expr.

Целочисленная арифметика

Арифметические операции — +, -, *, и / выполняются над целыми одного тип, т.е. можно сложить два 32 битовых целых и нельзя — 32 битовое целое с 16 битовым. Отметим, что операция деления округляет результат до ближайшего целого. Так деление int32 (5) / int32 (8) даст 1. В [Док, с. 61] описана функция idivide, позволяющая задать четыре варианта округления при целочисленном делении.

Мы не рассматриваем в лекциях представленные в В [Док, с. 62-65] битовые операции.

Логический тип данных

Данные этого типа имаеют значения true (истина) или false (ложь). Определены операции: & (Логическое И), | (Логическое Или) и ! (Логическое отрицание) выполняющиеся по обычным правилам логики.

Логические константы true и false принадлежат классу logical и имют тип bool. Тем не менее их можно применять в арифметических выражениях. В этом случае они будут иметь значение типа Fl64 — 1 для true и 0 — для false. Функция logical (x) будет преобразовывать любое ненулевое значение в значение true, а нулевое — в значение false.

Пример.

```
octave:1>
octave:1> a=true(),b=false()
a = 1
b = 0
octave:2> typeinfo(a),class(a)
```

```
ans = bool
ans = logical
octave:3> typeinfo(b),class(b)
ans = bool
ans = logical
octave:4> c=5+a-10*b
c = 6
octave:5> typeinfo(c),class(c)
ans = scalar
ans = double
octave:6> d=int32(5)+a-int32(10)*b
d = 6
octave:7> typeinfo(d),class(d)
ans = int32 scalar
ans = int32
octave:8> whos
Variables visible from the current scope:

variables in scope: top scope

  Attr   Name        Size                     Bytes  Class
  ====   ====        ====                     =====  =====
         a           1x1                          1  logical
         ans         1x5                          5  char
         b           1x1                          1  logical
         c           1x1                          8  double
         d           1x1                          4  int32

Total is 9 elements using 19 bytes

octave:9>
```

В [Док, с. 65] даны функции присвоения логических значений элементам массивов.

Арифметические операции с данными разных типов

В таблице показаны допустимые пары операндов разных типов (порядок операндов не имеет значения) с указанием типа получаемого результата (OP — знак арифметической операции).

Типы операндов	Тип

	результата
double OP single	single
double OP integer	integer
double OP char	double
double OP logical	double
single OP integer	integer
single OP char	single
single OP logical	single

Пример.

```
octave:13> a=3,b=int8(2)
a = 3
b = 2
octave:14> typeinfo(a),class(a)
ans = scalar
ans = double
octave:15> typeinfo(b),class(b)
ans = int8 scalar
ans = int8
octave:16> c=a+b
c = 5
octave:17> typeinfo(c),class(c)
ans = int8 scalar
ans = int8
octave:18> c=b+a
c = 5
octave:19> typeinfo(c),class(c)
ans = int8 scalar
ans = int8
octave:20>
```

Подчеркнем, что для ясности программ предпочтительно не смешивать типы операндов в выражениях.

Проверки типа числовых объектов

В octave реализовано много функций для проверки типа выражений. Эти функции имеют обобщенное имя istype(expr) и дают результат логического типа 1 (true) если аргумент функции — expr имеет тип type и 0 (false) в противном случае. Возможные значения type представлены в [Док, с. 66 — 70].

Строковые данные

Строковые константы задаются конструкциями вида "строка" или 'строка'. Отметим, что предпочтительно использование двойных кавычек, т. к. символ одинарной кавычки используется для обозначения операции транспонирования матрицы. Работа со строками и богатый набор функций для нее даны в [Док, с. 71 — 110].

Переменные

Имя переменной это последовательность букв, цифр и знаков подчеркивания, оно не может начинаться с цифры. Длина имен переменных не ограничена. Прописные и строчные буквы считаются разными символами.

Такие имена, как __foo_bar_baz__, которые начинаются и заканчиваются двумя символами подчеркивания, зарезервированы для внутреннего использования в octave. Вы не должны использовать их в коде, который вы пишете, за исключением доступа к документированным внутренним переменным octave и встроенным символьным константам.

Если интерпретатору octave задается вычисление выражения без присваивания его значения переменной, то это значение присваивается встроенной переменной ans. В то же время присвоения значения любой переменной не изменяет значения переменной ans.

В octave определены глобальные, локальные и постоянные (persistent) переменные. Мы рассмотрим их подробнее при изучении функций.

Статус переменных

В octave встроены функции, дающие информацию о свойствах перемененных определенных в текущем интерактивном сеансе или в файле с программой.

Команда who выводит список всех определенных переменных, команда выводит этот же список и, по умолчанию, значения следующих данных о каждой переменной: атрибут, имя, размерность, объем памяти в байтах и класс. Для атрибута определены следующие значения:

Атрибут	Описание атрибута
Символ пробела	Локальная переменная
c	Комплексный тип
f	Формальный параметр функции
g	Глобальная переменная
p	Постоянная (persistent) переменная

Если в текущей сессии octave определено много переменных, то можно воспользоваться описанными в [Док, с. 142 — 143] вариантами команд who и whos, позволяющими задавать образцы для поиска переменных, данные о которых нужно вывести.

Команда whos_line_format, описанная в [Док, с. 144] позволяет изменить формат и содержание данных о переменных, выводимых по умолчанию командой whos. Рассмотрим ее работу на примере.

Пример.

```
ybgv@ybgv-home:~> octave -q
octave:1> a=6
a = 6
octave:2> b=int16(31)
b = 31
octave:3> c=4+3i
c =  4 + 3i
octave:4> d=[1 1;2 2]
d =

   1   1
   2   2

octave:5> whos_line_format ()
ans =   %la:5; %ln:6; %cs:16:6:1;  %rb:12;  %lc:-1;

octave:6> whos
Variables visible from the current scope:

variables in scope: top scope

  Attr   Name        Size                     Bytes  Class
  ====   ====        ====                     =====  =====
         a           1x1                          8  double
         ans         1x46                        46  char
         b           1x1                          2  int16
  c      c           1x1                         16  double
         d           2x2                         32  double

Total is 53 elements using 104 bytes

octave:7> cd oct-m-files
octave:8> dir
.           ..          newwhos.m
octave:9> echo on
octave:10> newwhos
+  whos_line_format  ("%la:5;  %ln:6;  %cs:14:6:1;  %lt:6;  %rb:12;
%lc:-1;\n");
+
octave:11> whos_line_format ()
ans = %la:5; %ln:6; %cs:14:6:1; %lt:6; %rb:12; %lc:-1;

octave:12> whos
Variables visible from the current scope:
```

```
variables in scope: top scope

Attr   Name        Size            Type                    Bytes Class
====   ====        ====            ====                    ===== =====
       a           1x1             scalar                      8 double
       ans         1x49            sq_string                  49 char
       b           1x1             int16 scalar                2 int16
c      c           1x1             complex scalar             16 double
       d           2x2             matrix                     32 double

Total is 56 elements using 107 bytes

octave:13>
```

В [Док, с. 145 — 150] даны следующие команды управления переменными.

Имя команды	Функция
exist	Проверка существования имени
clear	Удаление переменных из памяти
clearvars	Удаление имен из памяти
pack	Для совестимости с Matlab. Не действует
type	Определение типа и категории имени
which	Определние типа имено
what	Вывод содержимого каталога