Компьютерные технологии в научных исследованиях и образовании

Юрий Анатольевич Богоявленский, заведующий кафедрой Информатики и математического обеспечения, к.т.н., доцент, ybgv

Скрипты и функции

Файлы скриптов

Публикация файлов скриптов

Дескрипторы функций

Анонимные функции

Функции поставляемые с дистрибутивом `octave`

Файл скрипта содержит последовательность команд `octave` не начинающихся с ключевого слова `function`. Считывая его `octave` выполняет их так, как будто они вводились в командной строке.

В отличие от файла функции переменные, определенные в файла сценария на имеет локальной области видимости, а принадлежат той же области видимости, в которой файл прочитан.

В одном файле сценария можно определить более одной функции и загрузить (но не выполнить) их все одновременно. Для этого первая команда в файле (не считая комментарии символы белого пробела) должна отличаться от ключевого слова `function`. Это может быть любое выражение, можно, например использовать инструкцию, не имеющую никакого эффекта, например задать константу: `1;`. Пример.

```
ybgv@ybgv-home:~> octave -q -p ~/MyOct
octave:1> scr_exm
+ echo on
+
```

```
+ X=[1 3 5 7 5 3 1 100 15 2]
X =

     1     3     5     7     5     3     1   100    15     2

+
+ global alpha=pi
+
+ whos
Variables visible from the current scope:

variables in scope: top scope

  Attr   Name        Size                     Bytes  Class
  ====   ====        ====                     =====  =====
         X           1x10                        80  double
     g   alpha       1x1                          8  double

Total is 11 elements using 88 bytes

+
+ newwhos
+ whos_line_format ("%la:5; %ln:6; %cs:14:6:1; %lt:6; %rb:12; %lc:-1;\n");
+
+
+ whos
Variables visible from the current scope:

variables in scope: top scope

Attr   Name        Size            Type           Bytes Class
====   ====        ====            ====           ===== =====
```

```
        X           1x10           matrix            80 double
    g   alpha       1x1            scalar             8 double

Total is 11 elements using 88 bytes


+
+ function retval = avg (v)
+
+     retval = 0;
+     if (nargin != 1)
+       error ("avg: параметр не один, вызов: avg (vector)");
+     endif
+
+     if (isvector (v))
+       retval = sum (v) / length (v);
+
+     else
+     error ("avg: параметр не вектор");
+     endif
+
+     return
+ endfunction
+
+ function [max, idx] = scvmax (v)
+     global alpha
+     idx = 1;
+     max = v (idx);
+     for i = 2:length (v)
+       if (v (i) > max)
+         max = v (i);
+           idx = i;
+       endif
```

```
+     endfor
+     alpha
+     alpha = 2+3i
+     whos
+     return
+ endfunction
+
+ avg(X)
ans = 14.200
+
+ scvmax(X)
alpha = 3.1416
alpha =  2 + 3i
Variables visible from the current scope:

variables in scope: scvmax: /home/ybgv/MyOct/scr_exm.m

Attr   Name        Size            Type                   Bytes Class
====   ====        ====            ====                   ===== =====
c  g   alpha       1x1             complex scalar            16 double
       i           1x1             scalar                     8 double
  f    idx         1x1             scalar                     8 double
  f    max         1x1             scalar                     8 double
  f    v           1x10            matrix                    80 double

Total is 14 elements using 120 bytes

ans = 100
+
+ whos
Variables visible from the current scope:

variables in scope: top scope
```

```
Attr   Name        Size            Type                    Bytes Class
====   ====        ====            ====                    ===== =====
       X           1x10            matrix                     80 double
c  g   alpha       1x1             complex scalar             16 double
       ans         1x1             scalar                      8 double

Total is 12 elements using 104 bytes


+
octave:2>
```

Чтение и компиляция файла скрипта во внутреннюю форму происходят если задать интерпретатору `octave` имя файла без расширения `.m`. При этом файл должен быть в одном из каталогов из списка каталогов «`load path`». Отметим, что правила для поиска файлов сценариев такие же как и для поиска файлов функций.

Заметим также, что если в первой строке файла находитя ключевое слово `function`, то `octave` попытается ее скомпилировать и выполнить. Если при этом после завершения определения функции будут обнаружены строки команд, то о каждой из них будт выдано предупреждающее сообщение.

Запустить на выполнение файл не имеющий расширения .m или не находящийся одном из каталогов из списка каталогов «`load path`» можно с помощью функций:

```
source (file)
source (file, context)
```

которые загружают файл и выполняют содержащиеся в нем команды. Если параметр `context` не задан, то они выполняются в текущей области видимости, если он задан как `caller`, то они выполняются в области видимости функции, вызвавшей текущую функцию. Если же этот параметр

задан как base, то команды файла выполняются в исходной области видимости (top scope).

Публикация файлов скриптов

Функция publish запускает файл скрипта и формирует документ, описывающий процесс его выполнения. В документ включаются все полученные при выполнении выходные данные и рисунки.

Команда publish ("полное_имя_файла") генерирует документ в формате html, размещая его в подкаталог html текущего рабочего каталога. Если этот подкаталог отсутствует, он создается автоматически. Подчеркнем, что команды файла выполняются в отдельной области видимости. См. пример генерации документа для рассмотренного выше файла scr_exm.m, полученного командой:

```
octave:2> publish MyOct/scr_exm.m
ans = MyOct/html/scr_exm.html
octave:3>
```

Отметим, что в этом документе видно, что скрипт выполняется в области видимости:

```
variables in scope: eval_code_helper:
/usr/share/octave/7.1.0/m/miscellaneous/publish.m
```

Формат html поддерживается по умолчанию, доступны также форматы latex и pdf. Мы не рассматриваем детали генерации отчетов, а также способ использования octave совместно с технологией Jupyter notebooks которые даны в [Док, с. 224 — 232].

Дескрипторы функций и анонимные функции

Многие функции octave выполняют операции над функциями (например численное интегрирование) и в этом случае имя функции, над

которой выполняется операция, нужно передать как параметр функции, выполняющей операцию. Эта задача решается с помощью переменных, имеющих тип function handle (дескриптор функции). Значение такой переменной содержательно интерпретируется как указатель на код (адрес кода) функции и присваивается переменной в выражении присваивания с помощью операции @:

```
имя_типа_function handle = @имя_функции
```

Пример численного интегрирования функции sin(x) на интервале [0,pi].

```
ybgv@ybgv-home:~> octave -q -p ~/MyOct
octave:1>
octave:1> newwhos
octave:2>
octave:2> quad (sin(x), 0, pi)
error: 'x' undefined near line 1, column 11
octave:3> quad (sin, 0, pi)
error: Invalid call to sin.  Correct usage is:

 -- sin (X)

Additional help for built-in functions and operators is
available in the online version of the manual.  Use the command
'doc <topic>' to search the manual index.

Help and information about Octave is also available on the WWW
at https://www.octave.org and via the help@octave.org
mailing list.
octave:4> f=@sin
f = @sin
octave:5> quad (f, 0, pi)
ans = 2
```

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

variables in scope: top scope

Attr   Name        Size            Type                    Bytes Class
====   ====        ====            ====                    ===== =====
       ans         1x1             scalar                      8 double
       f          1x1       function handle            0 function_handle

Total is 2 elements using 8 bytes

octave:7>
```

Вызвать функцию для конкретных значений фактических параметров можно либо с помощью функции `feval`, либо указав имя переменной, имеющей тип дескриптора функции со следующим за ним списком фактических параметров, который при отсутствии последних должен задаваться как `()`. Продолжение предыдущего примера:

```
octave:7> feval(f,pi/4)
ans = 0.7071
octave:8> f (pi/4)
ans = 0.7071
octave:9>
```

Функция `is_function_handle (x)` выдаст значение `true`, если `x` имеет тип дескриптора функции. Мы на рассматриваем еще несколько функций,

дающх данные о свойствах дескрипторв функций, которые представлены в [Док, с. 233 — 234]

Анонимные функции

Анонимные функции определяются в виде:

```
@(argument-list) expression
```

Напомним, что `(argument-list)` это разделенный запятыми список формальных параметров функции. Выражение `expression` вычисляет значение функции. Подчеркнем, что все переменные, не перечисленные в `(argument-list)`.

Анонимные функции полезны для определения простых безымянных функций из выражений или для «обертывания» других функций, чтобы адаптировать их для использования такими функциями, как, например, вычисление определенного интеграла `quad`.

Примеры.

```
ybgv@ybgv-home:~> octave -q -p ~/MyOct
octave:1>
octave:1> # вычисление определенного интеграла
octave:1> # от функции x^2 на интревале [0, 10]
octave:1>
octave:1> f = @(x) x.^2;
octave:2> quad (f, 0, 10)
ans = 333.33
octave:3>
octave:3> # вычисление определенного интеграла функции sin(x)
octave:3> # на интервале [0,pi] с помощью анонимной обертки
octave:3>
```

```
octave:3> quad (@(x) sin (x), 0, pi)
ans = 2
octave:4>
octave:4> # адаптация неполной бета функции с тремя параметрами
octave:4> # betainc (x, a, b) для вычисление ее определенного
octave:4> # интеграла на интервале [0,0.4]
octave:4>
octave:4> a = 1;
octave:5> b = 2;
octave:6> quad (@(x) betainc (x, a, b), 0, 0.4)
ans = 0.1387
octave:7>
```

Заметим, что производительность при использовании дескрипторов существующих функций `octave` существенно выше, чем использование их анонимных оберток. Так код:

```
quad (@sin, 0, pi)
```

будет работать в пять раз быстрее, чем код:

```
quad (@(x) sin (x), 0, pi)
```

Ме не рассматриваем приведенный в [Док, с. 235] альтернативный синтаксис обращения к функциям.

Функции поставляемые с дистрибутивом `octave`

Многие стандартные функции Octave поставляются в виде файлов функций. Они сгруппированы по темам в подкаталогах `octave-home/share/octave/version/m`. Ниже приведен список подкаталогов файлов функций с указанием их предназначения для `octave v.7.1`.

Имя подкаталога	Предназначение группы функций

@ftp	Класс для объекта ftp.
+containers	Классы контейнеров.
audio	Воспроизведение и запись звуков.
deprecated	Устаревшие функции, которые будут удалены
elfun	Элементарные функции, в т.ч. тригонометрические.
general	Различные матричные манипуляции, такие как flipud, rot90 и triu, а также другие базовые функции, такие как ismatrix, narginchk и т.п.
geometry	Триангуляция Делоне.
gui	Построение элементов графического интерфейса.
help	Справочная система.
image	Обработки изображений. Требуется система X Window.
io	Ввод-вывод.
java	Интеграция со средой Java.
linear-algebra	Линейная алгебра.
miscellaneous	Различные функции, не входящие в группы.
ode	Решение обыкновенных дифференциальных уравнений.
optimization	Минимизация, оптимизация, поиск корней функций.
path	Управление путем к каталогу, который octave использует для поиска функций.
pkg	Менеджер пакетов для установки внешних пакетов функций в octave.
plot	Отображение и печать двух и трехмерных графиков.
polynomial	Манипулирование многочленами.
prefs	Реализация предпочтений пользователя.
set	Операции над множествами.
signal	Приложения обработки сигналов.

`sparse`	Обработка разреженных матриц.
`specfun`	Специальные функции, например функция бесселя.
`special-matrix`	Создание специальных матриц, например Гильберта или Вандермонда.
`startup`	Управление загрузочным файлом `octave`.
`statistics`	Математическая статистика.
`strings`	Обработка строк.
`testfun`	Модульные тесты.
`time`	Обработка времени и даты.

Мы не рассматриваем представленные в [Док, с. 239 — 268] функции вывода сообщение об ошибках и предупреждениях, а также функции организации отладки.