Mérnöki programozás 9

Szerkesztette: dr. Vass Péter Tamás

Octave

Függvények hívása

Más programfejlesztő környezetekhez hasonlóan, az Octave is biztosít számos beépített könyvtári függvényt a felhasználók számára.

Ezek hívásával nagyon fontos és alapvető számítások egyszerű végrehajtása válik lehetővé.

Az Octave függvények általános jellemzői:

- egyedi azonosító nevük van,
- bemeneti paramétereik lehetnek, melyeket zárójelek közé téve adunk meg a függvény neve után,
- a bemenet paraméter skalár és mátrix is lehet, a függvénytől függően,
- a függvényeknek akár több kimeneti paramétere is lehet,
- a kimeneti paraméterek értékeit értékadás segítségével változóknak adhatjuk át.

Octave

Függvények hívása

Az egyes függvények szerepét, bemeneti és kimeneti paramétereinek számát és értelmezését a dokumentáció tartalmazza.

Példa a négyzetgyök függvény hívására egy skalár bemeneti változóval, az eredmény egy másik változóban tároljuk el értékadással:

```
>> x=2;
>> z=sqrt(x)
z =  1.4142
```

Példa a négyzetgyök függvény hívására egy vektor bemeneti változóval

```
>> v=[2, 3, 4];
>> z=sqrt(v)
z =
   1.4142   1.7321   2.0000
```

Octave

Függvények hívása

Példa a négyzetgyök függvény hívására egy mátrix bemeneti változóval

```
>> X=[2, 3, 4; 1, 5 9];
>> Z=sqrt(X)
Z =
    1.4142    1.7321    2.0000
    1.0000    2.2361    3.0000
```

A Octave negatív számból is von négyzetgyököt, az eredmény komplex szám lesz

```
>> X=[-2, 3, 4; 1, 5 -9];
>> Z=sqrt(X)
Z =
   0.00000 + 1.41421i   1.73205 + 0.00000i   2.00000 + 0.00000i
   1.00000 + 0.00000i   2.23607 + 0.00000i   0.00000 + 3.00000i
```

Octave

Függvények hívása

Példa mátrix inverzét számító függvény hívására:

```
>> A=[2, -1, 2; 4, -4, 6; 6, -7, 8];
>> invA=inv(A)
invA =
   1.25000  -0.75000   0.25000
   0.50000   0.50000  -0.50000
  -0.50000   1.00000  -0.50000
```

Példa két bemeneti paraméterrel hívott függvényre:

```
>> n=2;
>> m=3;
>> Y=ones(n,m)
Y =
   1   1   1
   1   1   1
```

Octave

Függvények hívása

Példa két kimeneti paraméterrel visszatérő függvényre:

```
>> A=[2, -1, 2; 4, -4, 6; 6, -7, 8];
>> [n, m]=size(A)
n =  3
m =  3
```

A beépített függvények betűrend szerinti listáját és a magyarázatukhoz kapcsolódó linkeket a Documentation ablak Function Index linkjére kattintva érhetjük el.

Octave

Matematikai műveletek

Összeadás

```
a + b
```

ha az operandusok mátrixok, akkor a soraik és oszlopaik számának meg kell egyeznie.

Elemenkénti összeadás

```
a .+ b
```

Megegyezik az összeadás + operátorral.

Kivonás

```
a – b
```

ha az operandusok mátrixok, akkor a soraik és oszlopaik számának meg kell egyeznie.

Elemenkénti kivonás

```
a .- b
```

Megegyezik a kivonás operátorral

Octave

Matematikai műveletek

Mátrix szorzás

```
a * b
```

az a mátrix oszlopai és a b mátrix sorai számának meg kell egyeznie.

Elemenkénti szorzás

```
a .* b
```

Skalárra és mátrixra egyaránt alkalmazható. Ha az operandusok mátrixok, akkor a soraik és oszlopaik számának meg kell egyeznie.

Octave

Matematikai műveletek

Mátrix komplex konjugált transzponáltja

```
X'
```

ha a mátrix elemei között vannak komplex számok, akkor azokra vonatkozóan nem csak a transzponálás, hanem a komplex konjugált képzés is végbemegy.

```
>> X=[0, 2+3i, -1; 2, -1+2i, 5; -3.5-i, 1, 3-2i]
X =
   0.00000 + 0.00000i   2.00000 + 3.00000i  -1.00000 + 0.00000i
   2.00000 + 0.00000i  -1.00000 + 2.00000i   5.00000 + 0.00000i
  -3.50000 - 1.00000i   1.00000 + 0.00000i   3.00000 - 2.00000i
>> X'
ans =
   0.00000 - 0.00000i   2.00000 - 0.00000i  -3.50000 + 1.00000i
   2.00000 - 3.00000i  -1.00000 - 2.00000i   1.00000 - 0.00000i
  -1.00000 - 0.00000i   5.00000 - 0.00000i   3.00000 + 2.00000i
```

Octave

Matematikai műveletek

Mátrix transzponáltja

```
X.'
```

a mátrix transzponáltját számítja ki, nincs komplex konjugált képzés a komplex elemek esetén.

```
>> X=[0, 2+3i, -1; 2, -1+2i, 5; -3.5-i, 1, 3-2i]
X =
   0.00000 + 0.00000i   2.00000 + 3.00000i  -1.00000 + 0.00000i
   2.00000 + 0.00000i  -1.00000 + 2.00000i   5.00000 + 0.00000i
  -3.50000 - 1.00000i   1.00000 + 0.00000i   3.00000 - 2.00000i
>> X.'
ans =
   0.00000 + 0.00000i   2.00000 + 0.00000i  -3.50000 - 1.00000i
   2.00000 + 3.00000i  -1.00000 + 2.00000i   1.00000 + 0.00000i
  -1.00000 + 0.00000i   5.00000 + 0.00000i   3.00000 - 2.00000i
```

Octave

Matematikai műveletek

Mátrix jobboldali osztása mátrixxal

```
a / b
```

az alábbi kifejezéssel azonos eredményt szolgáltat

```
(inv(b') * a')'
```

amiben `inv` az utána következő zárójelpáron belüli mátrix inverzét előállító beépített könyvtári függvény azonosítója, az aposztróf pedig az előtte álló mátrix transzponáltját szolgáltatja. Példa:

```
>> a=[1, 1; -2, 6];
>> b=[3, -5; 7, 2];
>> c=a/b
c =
  -0.12195   0.19512
  -1.12195   0.19512
>> c=(inv(b') * a')'
c =
  -0.12195   0.19512
  -1.12195   0.19512
```

Octave

Matematikai műveletek

Mátrix elemenkénti jobboldali osztása mátrixxal

```
a ./ b
```

Példa:

```
>> a=[1, 1; -2, 6];
>> b=[3, -5; 7, 2];
>> c=a./b
c =
   0.33333  -0.20000
  -0.28571   3.00000
```

Mátrix baloldali osztása mátrixxal

```
a \ b
```

az alábbi kifejezéssel azonos eredményt szolgáltat

```
inv(a) * b
```

Octave

Matematikai műveletek

Mátrix baloldali osztása mátrixxal

Példa:

```
>> a=[1, 1; -2, 6];
>> b=[3, -5; 7, 2];
>> c=a\b
c =
   1.3750  -4.0000
   1.6250  -1.0000
>> c=inv(a)*b
c =
   1.3750  -4.0000
   1.6250  -1.0000
```

Octave

Matematikai műveletek

Mátrix elemenkénti baloldali osztása mátrixxal

```
a .\ b
```

Példa:

```
>> a=[1, 1; -2, 6];
>> b=[3, -5; 7, 2];
>> c=a.\b
c =
   3.00000  -5.00000
  -3.50000   0.33333
```

Skalár hatványozása skalár kitevővel

`a^b` vagy `a**b`

Octave

Matematikai műveletek

Négyzetes mátrix hatványozása pozitív egész kitevővel

`a^b` vagy `a**b`

Példa:

```
>> a=[1, 1; -2, 6];
>> c=a^3
c =
   -15    41
   -82   190
>> c=a*a*a
c =
   -15    41
   -82   190
```

Octave

Matematikai műveletek

Mátrix elemenkénti hatványozása mátrixxal

`a.^b` vagy `a.**b`

mindkét mátrixnak azonos méretűnek kell lenni.

Példa:

```
>> a=[1, 1; -2, 6];
>> b=[3, -5; 7, 2];
>> c=a.^b
c =
     1     1
  -128    36
```

Skalár hatványozása mátrixxal

Példa:

```
>> c=2.^a
c =
    2.00000    2.00000
    0.25000   64.00000
```

Octave

Összehasonlítási műveletek

Az összehasonlítási (relációs) műveletek skalárok és mátrix között egyaránt végrehajthatók.

Mátrixok esetében elemenként történik az összehasonlítás, ennek megfelelően a két operandus méretének meg kell egyeznie.

Ha az összehasonlítás eredménye igaz, akkor az értéke 1, ha pedig hamis, akkor 0 lesz.

`a < b`	igaz, ha a értéke kisebb, mint b értéke, egyébként hamis
`a <= b`	igaz, ha a értéke kisebb, mint b értéke, vagy azzal egyenlő, egyébként hamis
`a > b`	igaz, ha a értéke nagyobb, mint b értéke, egyébként hamis
`a >= b`	igaz, ha a értéke nagyobb, mint b értéke, vagy azzal egyenlő, egyébként hamis

Octave

Összehasonlítási műveletek

`a == b`	igaz, ha a értéke egyenlő b értékével, egyébként hamis
`a != b`	igaz, ha a értéke nem egyenlő b értékével,
`a ~= b`	egyébként hamis

Példa:

```
>> a=[1, 2; 3, 4];
>> b=[1, 3; 2, 4];
>> a == b
ans =
   1   0
   0   1
```

Octave

Logikai műveletek

Mátrixok esetében elemenként történik a logikai műveletek végrehajtása, ennek megfelelően a két operandus méretének meg kell egyeznie.

&	logikai és operátor
|	logikai (megengedő) vagy operátor
!	logikai tagadás (negáció) operátor

Példa:

```
>> a=[1, 0; 0, 1];
>> b=[1, 0; 2, 3];
>> a & b
ans =
   1   0
   0   1
```

Octave

Értékadó kifejezések

Az értékadás egy olyan kifejezés, ami egy új értéket rendel hozzá egy változóhoz.

Az értékadás operátora az = jel.

Egy értékadó kifejezéssel egyszerre több változóhoz is hozzárendelhetjük ugyanazt az értéket.

Példa:

```
>> a=b=c=[0, 1; 1, 0];
>> a
a =
   0   1
   1   0
>> b
b =
   0   1
   1   0
>> c
c =
   0   1
   1   0
```

Octave

Értékadó kifejezések

A C nyelvhez hasonlóan a matematikai alapműveletek és az értékadás művelete összevonható, és egyszerűbb formában írható.

```
+=      -=      *=      /=
```

Példa:

```
>> a=[1, 2; 3, 4];
>> a+=2
a =

   3   4
   5   6
```

Octave

Inkrementálás és dekrementálás

A C nyelvhez hasonlóan az Octave programozási nyelvében is definiálva van az eggyel növelés (inkrementálás) és az eggyel csökkentés (dekrementálás) operátora.

`++a` megnöveli az "a" változó értékét eggyel, és a kifejezés értéke "a" új értéke lesz,

`a++` megnöveli az "a" változó értékét eggyel, de a kifejezés értéke "a" régi értéke marad,

`--a` csökkenti az "a" változó értékét eggyel, és a kifejezés értéke "a" új értéke lesz,

`a--` csökkenti az "a" változó értékét eggyel, de a kifejezés értéke "a" régi értéke marad,

Octave

Inkrementálás és dekrementálás

Példa:

```
>> a=[1, 2; 3, 4];
>> ++a
ans =
   2   3
   4   5
>> a++
ans =
   2   3
   4   5
>> a
a =
   3   4
   5   6
```

Octave

Műveletek precedenciája

A C nyelvhez hasonlóan az Octave esetében is különböző precedencia szintekhez tartoznak az egyes műveletek.
A magasabb precedenciájú műveletek végrehajtása hamarabb megy végbe egy több műveletet magában foglaló kifejezésben, mint az alacsonyabb precedenciájú műveleteké.
Zárójelek alkalmazásával azonban felülbírálhatjuk a műveletek végrehajtási sorrendjét.
Azonos precedenciájú műveletek esetében a műveletek általában balról jobbra haladva értékelődnek ki pl.
az `a - b + c` kifejezés csoportosítási sorrendje `(a - b) + c`
De az értékadási műveletek jobbról balra értékelődnek ki pl.
az `a = b = 5` többszörös értékadásban a sorrend `a = (b = 5)`

Octave

Vezérlési utasítások

If utasítás szerkezete

1. változat

```
if (feltétel)
      utasítás(ok)
endif
```

2. változat

```
if (feltétel)
      utasítás(ok)
else
      utasítás(ok)
endif
```

Octave

Vezérlési utasítások

If utasítás szerkezete

3. változat

```
if (feltétel_1)
      utasítás(ok)
elseif (feltétel_2)
      utasítás(ok)
else
      utasítás(ok)
endif
```

Octave

Vezérlési utasítások

switch utasítás szerkezete

```
switch (kifejezés)
      case konstans_kifejezés1
            utasítás(ok)
      case konstans_kifejezés1
            utasítás(ok)
      ...
      otherwise
            utasítás(ok)
endswitch
```

Octave

Vezérlési utasítások

while ciklus szerkezete

```
while (feltétel)
      utasítás(ok)
endwhile
```

do-until ciklus szerkezete

```
do
      utasítás(ok)
until (feltétel)
```

for ciklus szerkezete

```
for ciklusváltozó = kifejezés
      utasítás(ok)
endfor
```

A kifejezés általában tartomány, vektor vagy mátrix.

Octave

Egyéb utasítások

A C nyelvben megismertekkel megegyező módon vannak értelmezve a `break` és `continuous` utasítások, melyeket a ciklusok belsejében használhatunk.

Saját függvények definíciója

Bemenő és kimenő paraméter nélküli függvény definíciójának szerkezete

```
function fgvnev
      utasítás(ok)
endfunction
```

Kimenő paraméter nélküli függvény definíciójának szerkezete

```
function fgvnev(par. lista)
      utasítás(ok)
endfunction
```

Octave

Saját függvények definíciója

Egyetlen kimenő paraméterrel visszatérő függvény definíciójának szerkezete

```
function valtozonev=fgvnev(par. list)
      utasítás(ok)
endfunction
```

Több kimenő paraméterrel visszatérő függvény definíciójának szerkezete

```
function [par. list]=fgvnev(par. list)
      utasítás(ok)
endfunction
```

Octave

Grafikonok

Kétváltozós grafikonok rajzolása

```
>> x = [-10:0.1:10];
>> y = sin(x);
>> plot(x, y);
```

tengelyfeliratok és grafikon cím

```
>> xlabel("x");
>> ylabel("y");
>> title("y=sin(x)");
```

felirat elhelyezése a grafikonon

```
text(pi/2, 0.9, "amplitudo");
```

rácsháló bekapcsolása

```
>> grid on
```

Octave

Grafikonok

két görge rajzolása egy grafikonba

```
>> x = [-10:0.1:10];
>> y = sin(x);
>> z = cos(x);
>> plot(x, y, x, z, "--r");
>> xlabel("x");
>> ylabel("y");
>> title("szinusz es koszinusz fuggvenyek");
>> grid on
>> legend("sin(x)", "cos(x)");
>> legend("sin(x)", "cos(x)", "location", "west");
```

Octave

Grafikonok

vonalstílusok

```
-     folytonos vonal (alapértelmezett)
--    szaggatott vonal
:     pontozott vonal
-.    pontvonal
```

színek

```
k     fekete
r     piros
g     zöld
b     kék
m     bíbor
c     türkiz
w     fehér
```