Politechnika Gdańska
Wydział Elektrotechniki i Automatyki
Katedra Inżynierii Systemów Sterowania

Teoria sterowania

MATLAB – instrukcje warunkowe, logiczne, pętle

Materiały pomocnicze do ćwiczeń laboratoryjnych 1 – Część 5

Opracowanie:
Michał Grochowski, dr inż.
Robert Piotrowski, dr inż.

Gdańsk

Instrukcje MATLAB'a

MATLAB jest wyposażony w instrukcje sterujące realizacją poleceń o składni zapożyczonej z języka C. Podobne instrukcje istnieją w innych językach programowania. MATLAB posiada polecenia, które umożliwiają definiowanie własnych funkcji. Istnieją w nim instrukcje, które zapewniają interaktywną współpracę napisanego programu aplikacyjnego z użytkownikiem.

Podstawowymi instrukcjami MATLAB'a są:

- polecenia – te które omawialiśmy do tej pory i inne,
- instrukcje iteracyjne (pętle) **for** i **while**,
- instrukcje warunkowe **if**, **switch**,
- instrukcje **return**, **break**

Instrukcja for ("dla")

Ogólna postać instrukcji **for** MATLAB'a przedstawia się następująco:

for zmienna-iterowana=macierz-wartości,
    ciąg-instrukcji,
end

Weźmy następujący przykład, w którym nadawane są wartości elementom wektora x równe odwrotności wartości indeksu elementu.

```
» N=10;
» for i=1:N
    x(i)=1/i;
  end
```

Średnik kończący ciąg instrukcji wewnątrz pętli powoduje, że nie jest wyświetlana odpowiedź po wykonaniu każdej iteracji. Wektor x możemy wyświetlić po zakończeniu wszystkich iteracji pętli.

```
» x

x =

  Columns 1 through 7
    1.0000    0.5000    0.3333    0.2500    0.2000    0.1667    0.1429

  Columns 8 through 10
    0.1250    0.1111    0.1000
```

Wyniki działania pętli moglibyśmy prześledzić nie umieszczając średnika po ostatniej instrukcji pętli. Ilustruje to poniższy przykład:

```
» M=5;
» for j=1:M
     y(j)=1/j
  end;
y =
    1

y =
   1.0000    0.5000

y =
   1.0000    0.5000    0.3333

y =
   1.0000    0.5000    0.3333    0.2500

y =
   1.0000    0.5000    0.3333    0.2500    0.2000
```

W powyższych przykładach, gdyby N lub M były mniejsze od 1, zapis byłby poprawny, ale ciąg instrukcji wewnątrz pętli nie zostałby wykonany. Gdyby wektory x lub y nie istniały lub miały mniejszą liczbę elementów niż N lub M, wówczas zostałyby one automatycznie zdefiniowane lub rozszerzone.

Podsumowując: działanie instrukcji **for** polega na wykonaniu ciągu-instrukcji dla kolejnych wartości zmiennej-iterowanej. Wartościami tymi są kolejne kolumny pobrane z macierzy-wartości. Jeżeli jako macierz-wartości podany zostanie wektor wierszowy to instrukcje zostaną wykonane dla kolejnych elementów pobranych z tego wektora. O uszeregowaniu wartości, które staną się wartością zmiennej-iterowanej decyduje programista-użytkownik.

Można tworzyć w MATLAB'ie pętle zagnieżdżone. Ilustruje to poniższy przykład:

```
» A=[];
» M=3;
» N=5;
» for i=1:M
     for j=1:N
        A(i,j)=1/(i+j-1);
     end
  end
»
» A

A =
   1.0000    0.5000    0.3333    0.2500    0.2000
   0.5000    0.3333    0.2500    0.2000    0.1667
   0.3333    0.2500    0.2000    0.1667    0.1429
```

Należy zwrócić uwagę na to, że każda pętla **for** musi być zakończona instrukcją **end**. W przeciwnym wypadku MATLAB nie wykona żadnego działania czekając na kolejną instrukcję pętli.

Często jest tak, że postawione zadanie można rozwiązać nie tylko jednym sposobem. Należy wówczas dążyć do zastosowania sposobu najprostszego. Zilustrujemy to następującym przykładem. Mamy obliczyć tzw. macierz Vandermonde'a dla danego wektora. Kolumny tej macierzy są kolejnymi potęgami tego wektora. Ostatnia kolumna jest potęgą zerową, przedostatnia pierwszą itd. Pierwsza kolumna tej macierzy jest potęgą danego wektora równą jego rozmiarowi. Zadanie to można rozwiązać stosując dwukrotnie instrukcję **for**:

```
» A=[];
» t=[-1,0,1,3,5]'

t =
    -1
     0
     1
     3
     5

» n=max(size(t))

n =
     5

» for j=1:n
      for i=1:n
          A(i,j)=t(i)^(n-j);
      end
  end
» A

A =
      1     -1     1    -1     1
      0      0     0     0     1
      1      1     1     1     1
     81     27     9     3     1
    625    125    25     5     1
```

Można jednak rozwiązać to zadanie za pomocą pojedynczej pętli stosując indeksowanie wsteczne.

```
» A=[]

A =
     []

» A(:,n)=ones(n,1)
```

```
A =
    0    0    0    0    1
    0    0    0    0    1
    0    0    0    0    1
    0    0    0    0    1
    0    0    0    0    1

» for j=n-1:-1:1
     A(:,j)=t.*A(:,j+1);
  end



» A

A =
     1    -1     1    -1     1
     0     0     0     0     1
     1     1     1     1     1
    81    27     9     3     1
   625   125    25     5     1
```

Instrukcja while (“dopóki”)

Instrukcja while (dopóki) odpowiada analogicznym instrukcjom z języków programowania takich jak C czy Pascal i ma następującą postać:

while wyrażenie-warunkowe,
ciąg-instrukcji,
end

Instrukcja ta powoduje wykonywanie ciągu-instrukcji dopóki wartość wyrażenia-warunkowego ma wartość logiczną TRUE (PRAWDA), to znaczy wtedy, gdy macierz będąca wartością wyrażenia warunkowego ma wszystkie elementy niezerowe.

Weźmy prosty przykład w którym wykorzystamy funkcję MATLAB’a - prod(X). Aby dowiedzieć się co realizuje ta funkcja skorzystamy z pomocy:

```
» help prod

 PROD Product of the elements.
      For vectors, PROD(X) is the product of the elements of X.
      For matrices, PROD(X) is a row vector with the product over
      each column.
```

Wykorzystamy tę funkcję i instrukcję while do dla określenia dla jakiej wartości n wartość wyrażenia n! jest liczbą stucyfrową:

```
» n=1;
» while prod(1:n)<1.e100
```

```
    n=n+1;
end
```

Odczytamy wynik:

```
» n

n =
   70
```

Bardziej praktycznym przykładem zastosowania instrukcji **while** jest jej wykorzystanie do obliczenia eksponenty macierzy, czyli policzenia:

$$e^{A} = I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + ...$$

Uzasadnione jest wykonywanie sumowania szeregu dopóki jego wyrazy są wystarczająco duże. Można na przykład sumować tyle wyrazów tego szeregu, ile potrzeba, aby wynik nie ulegał zmianie przy uwzględnieniu dokładności arytmetyki komputera. Niech A będzie daną macierzą, E - pożądanym wynikiem eksponenty A, F - pojedynczym wyrazem szeregu a k - jego indeksem. Polecenia w pętli będą wykonywane dopóty, dopóki F nie stanie się tak mała, że dodanie jej do E nie zmieni E. W zaproponowanej niżej postaci instrukcji **while** dla rozwiązania zadania zostanie użyta funkcja MATLAB'a norm(X,1). Informację o tej funkcji możemy uzyskać korzystając z polecenia help:

```
» help norm

 NORM        Matrix or vector norm.
        For matrices..
         NORM(X) is the largest singular value of X, max(svd(X)).
         NORM(X,2) is the same as NORM(X).
         NORM(X,1) is the 1-norm of X, the largest column sum,
                  = max(sum(abs((X)))).
         NORM(X,inf) is the infinity norm of X, the largest row sum,
                  = max(sum(abs((X')))).
         NORM(X,'inf') is same as NORM(X,inf).
         NORM(X,'fro') is the F-norm, sqrt(sum(diag(X'*X))).
         NORM(X,P) is available for matrix X only if P is 1, 2, inf or 'fro'.
```

Zadanie rozwiążemy dla przykładowej macierzy A.

```
» A=[2,5;1,3]

A =
    2    5
    1    3
```

Propozycja rozwiązania zadania jest następująca:

```
» E=zeros(size(A))

E =
    0    0
```

```
    0     0
```

```
» F=eye(size(A))

F =
    1     0
    0     1
```

```
» k=1

k =
    1
```

```
» while norm(E+F-E,1)>0
       E=E+F;
       F=A*F/k;
       k=k+1;
  end
```

Wynik otrzymany wynosi:

```
» E

E =
   47.8358  130.0844
   26.0169   73.8527
```

Otrzymany wynik możemy sprawdzić korzystając z funkcji MATLAB'a **expm(X)**:

```
» eA=expm(A)

eA =
   47.8358  130.0844
   26.0169   73.8527
»
```

Jak widać obydwa rezultaty są identyczne.

Instrukcja warunkowa

Instrukcja warunkowa w MATLAB'ie ma postać:

```
if wyrażenie-warunkowe1
   ciąg-instrukcji1
elseif wyrażenie-warunkowe2
   ciąg-instrukcji2
   …
```

```
else
  ciąg-instrukcjiN
end
```

Wykonanie instrukcji **if** polega na wykonaniu ciągu-instrukcji, związanego z wyrażeniem-warunkowym, jeżeli jego wartość jest TRUE (PRAWDA). Jeżeli nie zachodzi żaden z warunków, wykonywany jest ciąg instrukcji po słowie kluczowym **else**. Sekwencje **elseif** … i **else** są opcjonalne.

Przykład pokazuje w jaki sposób za pomocą instrukcji warunkowej można rozbić obliczenia na trzy różne przypadki.

```
» A=[1,-3,3;-3,-4,1;1,2,-1]

A =
    1   -3    3
   -3   -4    1
    1    2   -1

» n=2

n =
    2

» if n<0
    A=-A
  elseif rem(n,2)==0
    A=2*A
  else
    B=invA;
  end
```

Sprawdzimy prawidłowość wykonania instrukcji. Macierz A powinna mieć wszystkie elementy pomnożone przez 2, macierz B powinna być pusta.

```
>> A

A =
    2   -6    6
   -6   -8    2
    2    4   -2

» B

B =
    []
```

Instrukcja switch

Instrukcja **switch** sprawdza wartość wyrażenia i w zależności od wyniku przypisuje wykonanie pewnych instrukcji (case 1, case 2, …, otherwise). Instrukcja **switch** ma postać:

```
switch wyrażenie
case 1
    instrukcja 1
case 2
    instrukcja 2
otherwise
    instrukcja 3
end
```

Instrukcja break

Instrukcja **break** powoduje przerwanie wykonywania pętli, przy czym opuszczony jest tylko jeden poziom zagłębienia pętli.

Instrukcja return

Instrukcja **return** powoduje bezwarunkowe opuszczenie danej funkcji lub skryptu i powrót do miejsca jej/jego wywołania.

Bibliografia

Brzózka J., Dorobczyński L. *Matlab – środowisko obliczeń naukowo – technicznych.* Wydawnictwo MIKOM, 2005.

Mrozek B., Mrozek Z. *Matlab i Simulink. Poradnik użytkownika. Wydanie II.* Wydawnictwo HELION, 2004.

Zalewski A., Cegieła R. *Matlab – obliczenia numeryczne i ich zastosowania.* Wydawnictwo NAKOM, 1996.