НТУУ „Київський Політехнічний Інститут ім. Ігоря Сікорського”
Інститут прикладного системного аналізу
Кафедра „Математичних методів системного аналізу”

КЕРІВНИЦТВО ПО РОБОТІ З МАТРИЦЯМИ ТА ВЕКТОРАМИ В СИСТЕМІ EVIEWS

Автор документа:

студент гр. КА-45 Дегтярьов А. О.

Викладачі:

інженер I категорії, асистент Терентьєв О.М.

доцент, д.т.н. Кузнєцова Н.В.

проф., д.т.н. Бідюк П. І.

КИЇВ – 2019

Для зручної навігації по документу використовуйте гіперсилки зміста, зажати кнопку Ctrl + натиснути ліву кнопку миші.

ЗМІСТ

Вступ

У системі **EViews** є можливість створення наступних об’єктів

1. ***Matrix*** – матриця.

2. ***Rowvector* –** вектор-строка.

3. ***Scalar*** – скаляр.

4. ***Sym* –** симетрична матриця (елементи матриці розташовані у нижній трикутній формі).

5. ***Vector* –** вектор-стовпчик.

Ці об’єкти у системі **EViews** називаються матричними об’єктами, хоча і не всі з них є матрицями. Кожен матричний об’єкт повинен бути оголошен до початку використання.

Стандартна форма оголошення має наступну структуру:

“тип матричного об’єкту” (“розмір”) “ім’я”

Розмір задається у круглих дужках.

При створенні, різні матричні об’єкти потребують різну інформацію про розмір об’єкта, що створюється.

При створенні об’єкту ***Matrix*** треба задати кількість рядків та стовпчиків. При оголошенні об’єкта S***ym*** треба задати лише одне число, яке буде визначати кількість строк і стовпчиків одночасно (симетрична матриця може бути тільки квадратною). Оголошення об’єктів V***ector***, R***owvector*** вимагає визначення кількості елементів. ***Scalar*** не вимагає інформації про розмір. У випадку, коли розмір не задається при оголошенні матричного об’єкту, то буде створений матричний об’єкт лише з одним елементом.

Приклад:

matrix(3,10) xdata

sym(9) moments

vector(11) betas

rowvector(5) xob

Ці команди створюють матрицю *xdata* розміром 3×10, симетричну матрицю ***moments*** 9×9, вектор-стовпчик ***betas*** (9), вектор-строку ***xob***(5). У цьому випадку елементи всіх цих об’єктів дорівнюють нулю.

Для того щоб змінити розмір існуючого матричного об’єкту, можна цей об’єкт переоголосити. У разі використання оператора присвоєння з вже існуючим матричним об’єктом, при необхідності, розмір об’єкту може бути змінений.

Приклад:

sym(10) bigz

matrix zdata

matrix(10,2) zdata

zdata = bigz

Спочатку матриця ***zdata*** має всього один елемент, потім переоголошується як матриця 10×2. Оператор присвоєння в останньому рядку змінить розмір zdata так, щоб матриця ***zdata*** містила в собі елементи ***bigz***. У нашому випадку розмір матриці ***zdata*** буде 10×10.

Значення елементів матриці

Є три способи надання значень елементам матриці:

1. Задати значення окремим елементам матриці.

2. Заповнити матрицю, використовуючи список значень.

3. Прирівняти одну матрицю іншій.

Робота з елементами матриці:

matrix(2,2) a

a(1,1) = 1

a(2,1) = 4

Отримаємо:

$$A = \begin{pmatrix} 1 & 0 \\ 4 & 0 \end{pmatrix}$$

Для визначення великої кількості елементів можна використовувати цикли:

```
vector(10) y
matrix (10,10) x
for !i = 1 to 10
    y(!i) = !i
    for !j = 1 to 10
        x(!i,!j) = !i + !j
    next
next
```

Процедура Fill

Ця процедура використовується для надання значень списку чисел згідно з вказаним порядком. За замовченням, процедура заповнює матрицю по стовпчиках.

Приклад:

```
vector(3) v
v.fill 0.1, 0.2, 0.3
matrix(2,4) x
x.fill 1, 2, 3, 4, 5, 6, 7, 8
```

$$V = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.3 \end{pmatrix} \quad X = \begin{pmatrix} 1 & 3 & 5 & 7 \\ 2 & 4 & 6 & 8 \end{pmatrix}$$

Якщо останній рядок записати наступним чином:

x.fill(b=r) 1,2,3,4,5,6,7,8

то отримаємо матрицю X :

$$X = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{pmatrix}$$

Якщо значення елементів повторюються, то в процедурі ***fill*** можна використати параметр ***«l»***:

Приклад:

matrix(3,3) y

y.fill(l) 1, 0, -1

$$Y = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}$$

Інші приклади надання значень елементам матриці:

<u>Приклад 1:</u>

matrix(5,8) first

scalar second

vector(10) third

first = 5

second = first(2,2)

third = first(3,5)

Створюється матриця 5×8 ***first***, скаляр ***second***, вектор-стовпчик (10) ***third***. Всі елементи цих об’єктів дорівнюють нулю. У 4-му рядку всі елементи матриці first прирівнюються 5, потім **second = 5** та **third = 5**.

Все це можна було записати так:

matrix(5,8) first = 5

scalar second = first(2,2)

vector(10) third = first(3,5)

<u>Приклад 2:</u>

vector(3) x

x(1) = 1

x(2) = 2

x(3) = 3

vector y = x

matrix z = x

У цьому прикладі розмір вектора Y дорівнює 3, матриця Z має розмір 3×1, $Z(1,1)=1, Z(2,1)=2, Z(3,1)=3$.

<u>Приклад 3:</u>

vector(7) y = 2

scalar value = 4

matrix(10,10) w = value

w = y

matrix(2,3) x = 1

rowvector(10) t = 100

x = t

W визначена як матриця 10×10, всі елементи якої дорівнюють 4, але у четвертому рядку переоголошується як матриця 7×1 з двійок.

$X -$ спочатку матриця 2×3 із одиниць, потім 1×10 всі елементи якої дорівнюють 100.

Математичні операції над матрицями

У системі **EViews** передбачена робота зі стандартними математичними операціями над матрицями.

Унарний мінус(-)

matrix jneg = -jpos

Унарний мінус міняє кожний елемент матриці на протилежний.

$$JPOS = \begin{pmatrix} 1 & 2 \\ -3 & -4 \end{pmatrix} \qquad JNEG = \begin{pmatrix} -1 & -2 \\ 3 & 4 \end{pmatrix}$$

Додавання(+)

Можна додавати матричні об'єкти однакового типу і розміру. Результатом буде матричний об'єкт такого ж типу і розміру.

matrix(3,4) a

matrix(3,4) b

matrix sum = a + b

Віднімання(-)

Обернена операція додавання.

matrix(3,4) a

matrix(3,4) b

matrix dif = a – b

Добуток(*)

Можна використовувати операцію множення двох матричних об'єктів, якщо кількість стовпчиків першого об'єкту дорівнює кількості строк другого.

matrix(5,9) a

matrix(9,22) b

*matrix prod = a * b*

У цьому випадку розмірність ***PROD*** буде 5×22

Ділення(/)

Можна поділити матричний об'єкт на скаляр.

matrix z = orig/3

У цьому випадку кожен елемент матриці буде поділений на 3.

Операції порівняння(=, >, >=, <, <=, <>)

Два матричних об'єкти можуть бути порівняні між собою завдяки операціям порівняння. Результатом буде скаляр/логічне число. Порівнюються всі відповідні елементи матричного об'єкту, і якщо якась пара не відповідає умові порівняння, повертається значення **нуль** або **false**. У іншому випадку повертається значення **один** або **true**.

```
if result <> value then
	run crect
endif
```

Основні функції для роботи з матрицями

Функція colplace

Синтаксис: *colplace(m, v, n)*

Аргумент 1: matrix, m

Аргумент 2: vector, v

Аргумент 3: integer, n

Вставляє стовпчик v у матрицю m під номером n. Кількість строк m та v повинні дорівнювати.

Приклад:

colplace(m1,v1,3)

В матрицю $m1$ буде вставлений стовпчик під номером 3, який дорівнює вектору $v1$.

Аналогічно:

функція rowplace

rowplace (m2,r2,3)

Аргумент 1: matrix, m2

Аргумент 2: rowvector, r2

Аргумент 3: integer, n

В матрицю $m2$ буде вставлена строка під номером 3.

Функція @columnextract

Синтаксис: *@columnextract(m, n)*

Аргумент 1:	matrix чи sym, m

Argument 2:	integer, n

Результат:	vector

Вибирає вектор-стовпчик зі ствпчика під номером n об’єкту m.

Приклад:

vector v1 = @columnextract(m1,3)

Вектору $v1$ прирівнюються значення третього стовпчика матриці m1.

Аналогічно:

@rowextract (m, n)

Rowvector r1 = @rowextract(m2,2)

Функція @columns

Синтаксис:	*@columns(o)*

Аргумент:	matrix, vector, rowvector, sym, scalar, чи series

Результат:	integer

Повертає кількість стовпчиків об’єкта.

Приклад:

scalar sc2 = @columns(m1)

vec1(2) = @columns(s1)

Аналогічно:

Функція @rows (o)

Повертає кількість строк об’єкта.

Функція @convert

Синтаксис: *@convert(o, smp)*

Аргумент 1: series чи group

Аргумент 2: (не обов'язковий) sample, smp

Результат: vector чи matrix

Якщо o ряд, ***@convert*** поверне вектор, значення якого дорівнюють значенням o, які потрапляють у sample ***smp***.

Приклад:

vector v2 = @convert(ser1)

vector v3 = @convert(ser2,smp1)

Якщо o група, ***@convert*** поверне матрицю з елементів o, які потрапляють у sample ***smp***.

Приклад:

matrix m1=@convert(grp1)

matrix m2=@convert(grp1, smp1)

Функція @cor

Синтаксис 1: *@cor(v1, v2)*

Аргумент 1: vector, rowvector, чи series, v1

Аргумент 2: vector, rowvector, чи series, v2

Результат: scalar

Синтаксис 2: *@cor(o)*

Аргумент: matrix чи group, o

Результат: sym

Якщо параметрами є два ряди чи вектори $v1$ та $v2$, ***@cor*** поверне значення кореляції між цими двома векторами/рядами.

scalar sc1 = @cor(v1,v2)

s1(1,2) = @cor(v1,r1)

Якщо параметром є матриця чи група o, ***@cor*** обчислює кореляцію між стовпчиками цієї матриці/групи. Результатом є симетрична матриця.

sym x = @cor(m1)

Функція @cov

Синтаксис 1: *@cov(v1, v2)*

Аргумент 1: vector, rowvector, чи series, v1

Аргумент 2: vector, rowvector, чи series, v2

Результат: scalar

Синтаксис 2: @cov(o)

Аргумент: matrix object or group, o

Результат: sym

Якщо параметрами є два ряди чи вектори $v1$ та $v2$, ***@cov*** поверне значення коваріації між цими двома векторами/рядами.

scalar sc1 = @cov(v1,v2)

s1(1,2) = @cov(v1,r1)

Якщо параметром є матриця чи група *o*, ***@cov*** обчислює коваріацію між стовпчиками цієї матриці/групи. Результатом є симетрична матриця..

sym x = @cov(m1)

Функція @det

Синтаксис: *@det(m)*

Аргумент: matrix чи sym, m

Результат: scalar

Обчислює детермінант квадратної матриці m.

Приклад:

scalar sc1 = @det(m1)

vec4(2) = @det(s2)

Функція @filledmatrix

Синтаксис: *@filledmatrix(n1, n2, n3)*

Аргумент 1: integer, n1

Аргумент 2: integer, n2

Аргумент 3: scalar, n3

Результат: matrix

Повертає матрицю з $n1$ строками та $n2$ стовпчиками, в якій кожен елемент дорівнює значенню $n3$.

Приклад:

matrix m2 = @filledmatrix(3,2,7)

Створює матрицю 3×2, у якій кожен елемент дорівнює 7.

$$M2 = \begin{pmatrix} 7 & 7 \\ 7 & 7 \\ 7 & 7 \end{pmatrix}$$

Аналогічно функції:

@filledrowvector (Створює вектор-строку, заповнений вказаним числом)

@filledsym (Створює симетричну матрицю, заповнену вказаним числом)

@filledvector (Створює вектор-стовпчик, заповнений вказаним числом)

Функція @identity

Синтаксис: *@identity(n)*

Аргумент: integer, n

Результат: matrix

Повертає одничну матрицю n×n.

Приклад:

matrix i1 = @identity(4)

$$I1=\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}$$

Функція @implode

Синтаксис: *@implode(m)*

Аргумент: square matrix, m

Результат: sym

Формує симетричну матрицю, копіюючи нижній трикутник вхідної квадратної матриці m.

Приклад:

sym s2 = @implode(m1)

$$M1=\begin{pmatrix}1&2&3\\4&5&6\\7&8&9\end{pmatrix}\quad S2=\begin{pmatrix}1&&\\4&5&\\7&8&9\end{pmatrix}$$

Функція @inverse

Синтаксис: *@inverse(m)*

Аргумент: square matrix чи sym, m

Результат: matrix чи sym

Повертає обернену матрицю. Добутком початкової матриці з оберненою є одинична матриця.

Приклад:

matrix m2 = @inverse(m1)

sym s2 = @inverse(s1)

sym s3 = @inverse(@implode(m2))

Функція mtos

Синтаксис: *mtos(o1, o2, smp)*

Аргумент 1: vector, rowvector, matrix, sym, o1

Аргумент 2: series чи group, o2

Аргумент 3: (не обов'язковий) sample, smp

Якщо o1 вектор-стовпчик чи вектор-строка, mtos заповнює ряд o2 значеннями з o1, у діапазоні sample smp, якщо він вказаний. Якщо ряда з вказаним ім'ям не існує то він буде створений.

Приклад:

mtos(r1,ser1)

mtos(r1,ser3,smp1)

Якщо $o1$ матриця, mtos заповнює рупу рядів $o2$ значеннями з 01, у діапазоні sample ***smp***, якщо він вказаний. Якщо такої групи не існує, то команда створить і групу, і ряди які в ній знаходяться.

Приклад:

mtos(m1,grp1)

mtos(m1,grp2,smp1)

Обернена функція:

stom (Сворює матрицю/вектор з групи/ряда)

stomna (Сворює матрицю/вектор з групи/ряда, у випадку коли група/ряд містить значення NA)

Функція @norm

Синтаксис: *@norm(o, n)*

Аргумент 1: vector, rowvector, matrix, sym, чи series, o

Аргумент 2: (не обов'язковий) integer, n

Результат: scalar

Повертає норму матриці.

Приклад:

scalar sc1 = @norm(m1)

scalar sc2 = @norm(v1,1)

Функція @rank

Синтаксис: *@rank(o, n)*

Аргумент 1: vector, rowvector, matrix, sym, чи series, o

Аргумент 2: (не обов'язковий) integer, n

Результат: integer

Повертає ранг матриці. Обчислення виконується шляхом підрахування кількості вироджених елементів матриці, абсолютне значення яких менше рівня толерантності, значення якого задається аргументом n. Якщо n не вказано, **EViews** використовує значення яке дорівнює добутку найбільшої розмірності матриці, норми матриці та епсілон (найменше число).

scalar rank1 = @rank(m1)

scalar rank2 = @rank(s1)

Функція @solvesystem

Синтаксис: *@solvesystem(o, v)*

Аргумент 1: matrix чи sym, o

Аргумент 2: vector, v

Результат: vector

Повертає вектор-стовпчик x, який є розв'язком системи рівнянь виду $M \cdot x = v$, де матриця M передається через аргумент o.

Приклад:

vector v2 = @solvesystem(m1,v1)

Функція @trace

Синтаксис: *@trace(m)*

Аргумент: matrix чи sym, m

Результат: scalar

Повертає суму діагональних елементів квадратної матриці.

Приклад:

scalar sc1 = @trace(m1)

Функція @transpose

Синтаксис: *@transpose(o)*

Аргумент: matrix, vector, rowvector, чи sym, o

Результат: matrix, rowvector, vector, чи sym

Нормує транспоновану матрицю.

Приклад:

matrix m2 = @transpose(m1)

rowvector r2 = @transpose(v1)

$$M1 = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \quad M2 = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}$$