Penggunaan R untuk Operasi Matriks

disusun oleh Bagus Sartono [*last update*: September 2018]

Departemen Statistika FMIPA – Institut Pertanian Bogor

Membuat dan Mengakses Objek Matriks di R

Membuat Matriks dan Mengaksesnya

Andaikan kita ingin membuat matriks berukuran 3 x 2 di R dengan nama A sebagai berikut

$$A = \begin{bmatrix} 1 & 2 \\ 5 & 6 \\ 3 & 3 \end{bmatrix},$$

maka perintah yang diberikan adalah

```
> A = matrix(c(1, 2, 5, 6, 3, 3), ncol=2, byrow=TRUE)
```

Perhatikan cara menuliskan unsur dari matriks A pada c(1, 2, 5, 6, 3, 3) adalah dengan cara menuliskan unsur setiap baris secara berurutan. Jika perintah di atas dijalankan, maka pada workspace R akan tersimpan objek A berupa matriks berukuran 3 x 2. Untuk menampilkan dan melihat isi dari matriks A, cukup berikan perintah

```
> A
```

dan kita akan memperoleh output

```
     [,1] [,2]
[1,]    1    2
[2,]    5    6
[3,]    3    3
```

Output di atas dapat juga diperoleh dengan memberikan perintah

```
> print(A)
     [,1] [,2]
[1,]    1    2
[2,]    5    6
[3,]    3    3
```

Selain itu, kita juga dapat menggunakan cara lain untuk membuat matriks A, yaitu dengan menuliskan unsur kolom per kolom sebagai berikut

```
> A = matrix(c(1, 5, 3, 2, 6, 3), ncol=2, byrow=FALSE)
```

yang memberikan hasil persis sama dengan sebelumnya.

Guna mengetahui ukuran atau ordo dari matriks A kita bisa memberikan perintah

```
> dim(A)
```

dan akan didapatkan output

```
[1] 3 2
```

yang berarti A memiliki 3 baris dan 2 kolom.

Terdapat pula perintah untuk hanya memberikan nilai banyaknya baris dan banyaknya kolom seperti yang diilustrasikan berikut ini

```
> nrow(A)
[1] 3
> ncol(A)
[1] 2
```

Suatu saat ada kemungkinan kita ingin mendapatkan nilai unsur yang ada ada pada baris dan/atau kolom tertentu. Berikut ini akan dipaparkan bagaimana memperoleh hal tersebut. Seandaikan kita ingin memperoleh a_{22} yang tidak adalah adalah unsur pada baris ke-2 dan kolom ke-2, maka perintah yang digunakan beserta hasilnya adalah sebagai berikut

```
> A[2,2]
[1] 6
```

Perintah yang diberikan untuk mengakses unsur pada baris dan kolom tertentu adalah dengan menyebutkan nama matriks dan kemudian menyebutkan nomor baris dan nomor kolom yang dipisah tanda koma dan diletakkan di dalam kurung siku [ ] seperti di atas. Jelas bahwa untuk mendapatkan unsur pada baris ke-3 dan kolom pertama maka perintah yang diberikan adalah A[3, 1].

Selanjutnya kita juga dapat mengambil kolom atau baris tertentu. Perintah dan hasil untuk mengambil kolom pertama adalah

```
> A[ , 1]
[1] 1 5 3
```

Sedangkan perintah dan hasil untuk mengambil baris ketiga adalah

```
> A[3, ]
[1] 3 3
```

Perhatikan bahwa jika lokasi baris kita kosongkan dan hanya berisi nomor kolom maka kita memperoleh unsur kolom yang diinginkan untuk semua baris. Sebaliknya jika nomor kolom yang dikosongkan, maka kita akan mendapatkan unsur baris di semua kolom.

Kita dapat pula mengakses beberapa kolom/baris tertentu, misalnya perintah

```
> A[c(1,3), ]
```

akan menghasilkan baris pertama dan ketiga dari matriks A atau

```
     [,1] [,2]
[1,]    1    2
[2,]    3    3
```

Perintah

```
> A[c(1,2),]
```

dan

```
> A[1:2,]
```

memiliki makna yang sama yaitu mengambil baris pertama dan kedua dari matriks A.

Tentu saja hasil-hasil di atas bisa kita simpan menjadi objek/variabel baru untuk selanjutnya dipergunakan di waktu mendatang. Misalnya saja kita akan simpan a_{22} menjadi variabel dengan nama b dan a_{31} disimpan dengan nama c dan kemudian keduanya kita jumlahkan. Perintah yang dilakukan dan hasilnya adalah sebagai berikut

```
> b = A[2, 2]
> c = A[3, 1]
> b + c
[1] 9
```

Beberapa fungsi yang berguna

R menyediakan beberapa fungsi yang berguna dan sering digunakan untuk dapat memperoleh informasi dari matriks yang kita miliki. Pada bagian ini akan diulas beberapa fungsi tersebut.

sum( )

Fungsi ini berguna untuk mendapatkan jumlah unsur dari matriks, seperti ilustrasi di bawah ini

```
> B=matrix(c(1,2,3,3,1,4,3,2,2), byrow=TRUE, ncol=3)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    3    1    4
[3,]    3    2    2
> sum(B)
[1] 21
> sum(B[1,])
[1] 6
> sum(B[,1])
[1] 7
```

rowSums( ) dan colSums( )

Kedua fungsi ini menghasilkan vektor yang berisi jumlah unsur di setiap baris dan jumlah unsur di setiap kolom. Ilustrasinya sebagai berikut

```
> B=matrix(c(1,2,3,3,1,4,3,2,2), byrow=TRUE, ncol=3)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    3    1    4
[3,]    3    2    2
> rowSums(B)
[1] 6 8 7
> colSums(B)
[1] 7 5 9
```

apply( )

Fungsi lain untuk menghasilkan jumlah baris dan kolom adalah apply( ). Cara penggunaannya adalah sebagai berikut

```
> B=matrix(c(1,2,3,3,1,4,3,2,2), byrow=TRUE, ncol=3)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    3    1    4
[3,]    3    2    2
> apply(B, 1, sum)
[1] 6 8 7
> apply(B, 2, sum)
[1] 7 5 9
```

Kode 1 dan 2 di fungsi apply( ) adalah untuk menentukan apakah sum (penjumlahan) dilakukan berdasarkan baris ataukah kolom.

diag( )

Fungsi ini dapat digunakan untuk mengambil unsur diagonal dari sebuah matriks seperti yang dapat dilihat pada program di bawah ini.

```
> B=matrix(c(1,2,3,3,1,4,3,2,2), byrow=TRUE, ncol=3)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    3    1    4
[3,]    3    2    2
> diag(B)
[1] 1 1 2
```

Fungsi diag() juga dapat digunakan untuk mengganti unsur diagonal suatu matriks dengan cara berikut

```
> B=matrix(c(1,2,3,3,1,4,3,2,2), byrow=TRUE, ncol=3)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    3    1    4
[3,]    3    2    2
> diag(B) = c(4, 4, 2)
> B
     [,1] [,2] [,3]
[1,]    4    2    3
[2,]    3    4    4
[3,]    3    2    2
```

Latihan Penggunaan R

1. Perhatikan matriks X berikut

$$X = \begin{bmatrix} 3 & 4 & 5 & 1 \\ 1 & 3 & 4 & 2 \\ 1 & 4 & 5 & 2 \\ 6 & 4 & 3 & 3 \end{bmatrix}$$

   a. Buatlah perintah di R untuk membuat matriks di atas dengan nama X
   b. Bagaimana perintah untuk memperoleh unsur x_{23}?
   c. Bagaimana perintah untuk memperoleh baris kedua dan keempat matriks X?
   d. Bagaimana perintah untuk memperoleh baris kedua dan keempat serta kolom pertama dan ketiga?

2. Seseorang memberikan perintah di R dalam bentuk

```
B = matrix(c(3, 4, 5, 6, 3, 8, 9, 4), ncol=2, byrow=TRUE)
```

   Tentukan hasil dari perintah-perintah di bawah ini
   a. B[3, 2]
   b. B[1, 1]
   c. B[1,]
   d. B[3:4,]

   e. B[2:4, 1]
   f. sum(B)
   g. sum(colSums(B))

3. Seseorang memberikan perintah di R dalam bentuk

   ```
   D = matrix(c(3, 4, 5, 6, 3, 8, 9, 4, 9), ncol=3, byrow=TRUE)
   ```

   Tentukan hasil dari perintah-perintah di bawah ini

   a. sum(rowSums(D))
   b. sum(diag(D))
   c. sum(B[2,])
   d. sum(B[1:2, 2:3])

Beberapa Operasi Matriks

Penjumlahan dan Pengurangan Matriks

Andaikan terdapat dua buah matriks dengan ukuran yang sama yaitu 3 x 3 sebagai berikut

$$B = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 1 & 4 \\ 3 & 2 & 2 \end{bmatrix} \text{ dan } C = \begin{bmatrix} 2 & 3 & 3 \\ 1 & 4 & 1 \\ 2 & 4 & 3 \end{bmatrix}$$

perintah di R untuk menjumlahkan keduanya menjadi matriks D dan kemudian menampilkan hasilnya adalah

```
> B=matrix(c(1,2,3,3,1,4,3,2,2), byrow=TRUE, ncol=3)
> C=matrix(c(2,3,3,1,4,1,2,4,3), byrow=TRUE, ncol=3)
> D = B + C
> D
     [,1] [,2] [,3]
[1,]    3    5    6
[2,]    4    5    5
[3,]    5    6    5
```

Sedangkan hasil pengurangannya adalah

```
> E = B - C
> E
     [,1] [,2] [,3]
[1,]   -1   -1    0
[2,]    2   -3    3
[3,]    1   -2   -1
```

Putaran (Transpose) Matriks

Untuk melakukan putaran atau transpose terhadap matriks di R, kita dapat menggunakan fungsi t() seperti yang diilustrasikan berikut

```
> B=matrix(c(1,2,3,3,1,4,3,2,2), byrow=TRUE, ncol=3)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    3    1    4
[3,]    3    2    2
> t(B)
     [,1] [,2] [,3]
[1,]    1    3    3
[2,]    2    1    2
[3,]    3    4    2
```

Ilustrasi lain adalah

```
> D=matrix(c(1,2,3,3,1,4), byrow=TRUE, ncol=3)
> D
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    3    1    4
> t(D)
     [,1] [,2]
[1,]    1    3
[2,]    2    1
[3,]    3    4
```

Perkalian Matriks dengan Skalar

Perkalian matriks dengan skalar akan menghasilkan matriks baru dimana unsur dari matriks baru tersebut merupakan perkalian dari masing-masing unsur matriks asal dengan skalar yang digunakan. Operator perkalian pada R adalah "*" (asterisk), dan ilustrasinya adalah sebagai berikut

```
> D=matrix(c(1,2,3,3,1,4), byrow=TRUE, ncol=3)
> D
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    3    1    4

> 3 * D
     [,1] [,2] [,3]
[1,]    3    6    9
[2,]    9    3   12

> 2.5 * D
     [,1] [,2] [,3]
[1,]  2.5  5.0  7.5
[2,]  7.5  2.5 10.0
```

Perkalian Matriks

Perkalian matriks tidak dapat dilakukan pada sembarang matriks. Hanya matriks yang memiliki ukuran yang sesuai yang bisa dikalikan. Perkalian matriks dapat dilakukan jika banyaknya kolom dari matriks pertama sama dengan banyaknya baris dari matriks kedua. Perhatikan bahwa jika

$$B = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 1 & 4 \\ 3 & 2 & 2 \end{bmatrix} \text{ dan } C = \begin{bmatrix} 2 & 3 & 3 \\ 1 & 4 & 1 \\ 2 & 4 & 3 \end{bmatrix}$$

maka perkalian B dengan C dapat dikerjakan. Namun untuk

$$B = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 1 & 4 \\ 3 & 2 & 2 \end{bmatrix} \text{ dan } D = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 1 & 4 \end{bmatrix}$$

tidak dapat dilakukan perkalian B dengan D karena B memiliki kolom sebanyak 3 yang tidak sama dengan banyaknya baris D yaitu 2.

Operator perkalian matriks pada R adalah “%*%” seperti yang diilustrasikan di bawah ini

```
> B=matrix(c(1,2,3,3,1,4,3,2,2), byrow=TRUE, ncol=3)
> C=matrix(c(2,3,3,1,4,1,2,4,3), byrow=TRUE, ncol=3)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    3    1    4
[3,]    3    2    2
> C
     [,1] [,2] [,3]
[1,]    2    3    3
[2,]    1    4    1
[3,]    2    4    3
> B %*% C
     [,1] [,2] [,3]
[1,]   10   23   14
[2,]   15   29   22
[3,]   12   25   17
```

serta

```
> B=matrix(c(1,2,3,3,1,4,3,2,2), byrow=TRUE, ncol=3)
> D=matrix(c(1,2,3,3,1,4), byrow=TRUE, ncol=3)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    3    1    4
[3,]    3    2    2
> D
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    3    1    4
> B %*% D
Error in B %*% D : non-conformable arguments
```

<u>Latihan Penggunaan R</u>

1. Perhatikan matriks X berikut

$$X = \begin{bmatrix} 3 & 4 & 5 & 1 \\ 1 & 3 & 4 & 2 \\ 1 & 4 & 5 & 2 \\ 6 & 4 & 3 & 3 \end{bmatrix} \text{dan } Y = \begin{bmatrix} 7 & 2 & 5 & 1 \\ 1 & 5 & 4 & 3 \\ 2 & 4 & 3 & 2 \\ 1 & 4 & 1 & 3 \end{bmatrix}$$

Tuliskan perintah program di R untuk hal-hal berikut

a. membuat matriks di atas dengan nama X dan Y
b. menjumlahkan kedua matriks
c. menggandakan matriks, yaitu XY
d. menentukan matriks hasil operasi X^TY
e. menunjukkan bahwa $(XY)^T = Y^TX^T$

Membuat Matriks Khusus

Ada beberapa matriks yang dapat dibuat di R dengan lebih cepat, tidak menggunakan perintah yang sebelumnya dikerjakan yaitu menyebutkan semua unsur matriks. Beberapa yang bisa dilakukan akan dipaparkan di bawah ini.

Matriks Nol

Sebuah matriks disebut sebagai matriks nol jika seluruh unsurnya bernilai nol. Berikut perintah yang dapat digunakan untuk menghasilkan matriks nol berukuran 4 x 3.

```
> A = matrix (rep(0, 12), ncol=3)
> A
     [,1] [,2] [,3]
[1,]    0    0    0
[2,]    0    0    0
[3,]    0    0    0
[4,]    0    0    0
```

Perintah rep(0, 12) menghasilkan barisan bilangan 0 sebanyak 12 buah yang kemudian diubah menjadi matriks tiga kolom. Angka 12 digunakan karena kita akan membuat matriks 4 x 3 yang memerlukan 12 unsur. Dengan logika yang sama, program berikut ini menghasilkan matriks nol berukuran 5 x 6.

```
> B = matrix (rep(0, 30), ncol=6)
> B
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    0    0    0    0    0    0
[2,]    0    0    0    0    0    0
[3,]    0    0    0    0    0    0
[4,]    0    0    0    0    0    0
[5,]    0    0    0    0    0    0
```

Matriks Satuan

Matriks satuan merupakan matriks yang seluruh unsurnya adalah 1 (satu). Perintah di R untuk membuat matriks satuan sangat mirip dengan matriks nol dengan mengubah saja nilai 0 dengan 1. Berikut ini perintah menghasilkan dua buah matriks satuan di R yang berukuran 4 x 3 dan 5 x 6.

```
> D = matrix (rep(1, 12), ncol=3)
> D
     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    1
[3,]    1    1    1
[4,]    1    1    1
> C = matrix (rep(1, 30), ncol=6)
> C
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    1    1    1    1    1
[2,]    1    1    1    1    1    1
[3,]    1    1    1    1    1    1
[4,]    1    1    1    1    1    1
[5,]    1    1    1    1    1    1
```

Matriks Diagonal

Matriks diagonal merupakan matriks persegi yang semua unsur di luar diagonalnya bernilai nol. Berikut adalah perintah menghasilkan matriks diagonal 3 x 3 dengan unsur diagonal bernilai 4, 5, dan 10.

```
> E = diag(c(4,5,10))
> E
     [,1] [,2] [,3]
[1,]    4    0    0
[2,]    0    5    0
[3,]    0    0   10
```

Matriks Identitas

Matriks identitas adalah matriks diagonal yang seluruh unsur diagonalnya adalah 1 (satu). Berikut dua alternatif membuat matriks identitas berukuran 4 x 4

```
> #cara pertama
> F = diag(c(1,1,1,1))
> F
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    1    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1
> #cara kedua
> G = diag(1, nrow=4, ncol=4)
> G
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    1    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1
> #cara ketiga
> H = diag(rep(1, 4))
> H
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    1    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1
```

Latihan Penggunaan R

Tuliskan perintah program di R untuk memperoleh matriks-matriks berikut:

a. Matriks identitas berukuran 6 x 6
b. Vektor kolom satuan dengan enam unsur
c. Matriks diagonal dengan unsur diagonal: 1, 2, 3, 4, dan 5

Menghitung Determinan dan Pangkat (Rank) Matriks

Determinan

Determinan dari suatu matriks persegi dapat dihitung di R menggunakan fungsi det(). Berikut ini beberapa ilustrasi memperoleh determinan dari berbagai matriks

```
> A = matrix(c(1,0,2,0,1,2,2,2,1), ncol=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    1    0    2
[2,]    0    1    2
[3,]    2    2    1
> det(A)
[1] -7
```

```
> B = matrix(c(1, 2, 3, 2, 3, 4, 3, 5, 7), ncol=3, byrow=TRUE)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    2    3    4
[3,]    3    5    7
> det(B)
[1] 0
```

Pangkat (rank)

Pangkat (rank) dari suatu matriks persegi dapat dihitung di R menggunakan fungsi rankMatrix() yang ada pada library "Matrix" atau menggunakan salah satu output dari fungsi qr(). Berikut ini beberapa ilustrasi memperoleh pangkat (rank) dari berbagai matriks

```
> A = matrix(c(1,0,2,0,1,2,2,2,1), ncol=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    1    0    2
[2,]    0    1    2
[3,]    2    2    1
> qr(A)$rank
[1] 3
> library(Matrix)
> rankMatrix(A)
[1] 3
attr(,"method")
[1] "tolNorm2"
attr(,"useGrad")
[1] FALSE
attr(,"tol")
[1] 6.661338e-16
```

Ilustrasi lainnya:

```
> B = matrix(c(1, 2, 3, 2, 3, 4, 3, 5, 7), ncol=3, byrow=TRUE)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    2    3    4
[3,]    3    5    7
> qr(B)$rank
[1] 2
> rankMatrix(B)
[1] 2
attr(,"method")
[1] "tolNorm2"
attr(,"useGrad")
[1] FALSE
attr(,"tol")
[1] 6.661338e-16
```

<u>Adjoint</u>

Untuk memperoleh adjoin dari suatu matriks persegi kita dapat menyusun suatu fungsi di R. Fungsi yang bisa digunakan salah satunya adalah sebagai berikut

```
minor <- function(A, i, j) {
  det( A[-i,-j] )
  }

cofactor <- function(A, i, j) {
  (-1)^(i+j) * minor(A,i,j)
  }

adjoint <- function(A) {
  n <- nrow(A)
  B <- matrix(NA, n, n)
  for( i in 1:n )
    for( j in 1:n )
      B[j,i] <- cofactor(A, i, j)
  B
}
```

Penggunaan dari fungsi di atas dilakukan dengan memanggil menggunakan program berikut

```
> A = matrix(c(1,0,2,0,1,2,2,2,1), ncol=3, byrow=TRUE)
> adjoint(A)
     [,1] [,2] [,3]
[1,]   -3    4   -2
[2,]    4   -3   -2
[3,]   -2   -2    1
```

Latihan Penggunaan R

1. Perhatikan matriks X berikut

$$X = \begin{bmatrix} 3 & 4 & 5 & 1 \\ 1 & 3 & 4 & 2 \\ 1 & 4 & 5 & 2 \\ 6 & 4 & 3 & 3 \end{bmatrix} \text{ dan } Y = \begin{bmatrix} 7 & 2 & 5 & 1 \\ 1 & 5 & 4 & 3 \\ 2 & 4 & 3 & 2 \\ 1 & 4 & 1 & 3 \end{bmatrix}$$

Tuliskan perintah program di R untuk hal-hal berikut

a. memperoleh adjoint matrik X
b. memperoleh determinan matriks Y
c. memperoleh pangrkat/rank matriks X
d. menunjukkan bahwa pangkat dari matriks X sama dengan pangkat dari matriks X^T
e. menunjukkan bahwa determinan dari matriks Y sama dengan determinan dari matriks Y^T
f. menunjukkan bahwa determinan dari matriks XY sama dengan perkalian dari determinan X dan determinan Y

Operasi Baris Elementer

Karena tidak tersedia fungsi yang telah disediakan atau dibuat oleh orang lain dan disimpan dalam library di R, operasi baris elementer dapat dilakukan dengan secara mandiri kita buat fungsinya. Pada paparan ini disajikan tiga buah fungsi untuk melakukan tiga jenis operasi baris elementer.

Operasi menukar dua buah baris

Fungsi yang dapat dibuat untuk melakukan operasi penukaran dua baris adalah sebagai berikut:

```
tukarbaris <- function(A, baris1, baris2){
  A1 <- A[baris1,]
  A2 <- A[baris2,]
  A[baris1,] <- A2
  A[baris2,] <- A1
  return(A)
}
```

Untuk memanggil fungsi di atas kita tinggal menyebut nama fungsinya yaitu "tukarbaris" dan memberikan tiga argumen/input yaitu (1) matriks yang akan dioperasikan, (2) nomor baris pertama yang akan ditukar, dan (3) nomor baris kedua yang akan ditukar. Misalkan kita akan menukar baris kedua dan baris ketiga dari matriks B, maka perintah yang diberikan serta hasil yang didapatkan sebagai berikut.

```
> B = matrix(c(1, 2, 3, 2, 3, 4, 3, 5, 7), ncol=3, byrow=TRUE)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    2    3    4
[3,]    3    5    7
> tukarbaris(B,2,3)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    3    5    7
[3,]    2    3    4
```

Operasi menggandakan baris tertentu dari sebuah matriks dengan skalar

Berikut ini adalah fungsi dengan nama "penggandabaris" yang dapat digunakan untuk menggandakan baris tertentu dari sebuah matriks dengan skalar.

```
penggandabaris <- function(A, baris, pengganda){
  A[baris,] <- A[baris,] * pengganda
  return (A)
}
```

Untuk memanggil fungsi di atas kita tinggal menyebut nama fungsinya yaitu "penggandabaris" dan memberikan 3 (tiga) argumen/input yaitu (1) matriks yang akan dioperasikan, (2) nomor baris yang akan digandakan, dan (3) bilangan/skalar penggandanya. Misalkan kita akan menggandakan baris kedua dari matriks B dengan 6, maka perintah yang diberikan serta hasil yang didapatkan sebagai berikut.

```
> B = matrix(c(1, 2, 3, 2, 3, 4, 3, 5, 7), ncol=3, byrow=TRUE)
> B
     [,1] [,2] [,3]
```

```
[1,]    1    2    3
[2,]    2    3    4
[3,]    3    5    7
> penggandabaris(B, 2, 6)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]   12   18   24
[3,]    3    5    7
```

Operasi menambahkan baris tertentu dengan k kali baris lain
Berikut ini adalah fungsi dengan nama "tambahganda" yang dapat digunakan untuk menambahkan baris tertentu dengan k kali baris lain.

```
tambahganda <- function(A, baris1, baris2, pengganda){
  A[baris1, ] <- A[baris1, ] + pengganda * A[baris2, ]
  return(A)
}
```

Untuk memanggil fungsi di atas kita tinggal menyebut nama fungsinya yaitu "tambahganda" dan memberikan 4 (empat) argumen/input yaitu (1) matriks yang akan dioperasikan, (2) nomor baris yang akan diubah, dan (3) nomor baris yang akan digandakan dan ditambahkan, (4) bilangan/skalar penggandanya. Misalkan kita akan mengubah baris kedua dengan menambahkan -2 kali baris pertama dari matriks B, maka perintah yang diberikan serta hasil yang didapatkan sebagai berikut.

```
> B = matrix(c(1, 2, 3, 2, 3, 4, 3, 5, 7), ncol=3, byrow=TRUE)
> B
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    2    3    4
[3,]    3    5    7
> tambahganda(B, 2, 1, -2)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    0   -1   -2
[3,]    3    5    7
```

Penerapan invers matriks dalam bidang persandian

Salah satu penggunaan matriks adalah penerapannya di bidang kriptografi atau persandian dimana orang berupaya untuk merahasiakan pesan yang mereka kirimkan kepada seseorang dan mengubah naskah sehingga tidak mudah dibaca orang lain yang tidak memiliki kunci sandinya. Mengubah naskah menjadi kode yang sulit dibaca disebut sebagai kegiatan enkripsi, sedangkan kode kembali menjadi naskah yang dapat dibaca dikenal sebagai dekripsi. Berikut ini akan dipaparkan cara melakukan enkripsi sederhana menggunakan perkalian matriks dan selanjutnya mendekripsi kode menggunakan invers matriksnya.

Andaikan setiap abjad dikodekan dalam bentuk bilangan sebagai berikut:

0 = _	7 = G	14 = N	21 = U
1 = A	8 = H	15 = O	22 = V
2 = B	9 = I	16 = P	23 = W
3 = C	10 = J	17 = Q	24 = X
4 = D	11 = K	18 = R	25 = Y
5 = E	12 = L	19 = S	26 = Z
6 = F	13 = M	20 = T	

dan pesan yang akan dikirimkan adalah "AYO_PERGI". Pesan tersebut jika dinyatakan dalam bilangan akan terdiri atas 9 bilangan yaitu: 1, 25, 15, 0, 16, 5, 18, 7, 9. Dalam susunan matriks 3 x 3 kita dapat tuliskan sebagai

$$\text{pesan asli} = \begin{bmatrix} 1 & 25 & 15 \\ 0 & 16 & 5 \\ 18 & 7 & 9 \end{bmatrix}$$

Seandainya saja digunakan matriks A berukuran 3 x 3 untuk mengubah pesan dimana matriks A dan inversnya adalah sebagai berikut

$$A = \begin{bmatrix} 1 & 3 & 3 \\ 1 & 4 & 3 \\ 1 & 3 & 4 \end{bmatrix} \text{ dan } A^{-1} = \begin{bmatrix} 7 & -3 & -3 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

Yang dilakukan adalah bukan mengirim matriks pesan asli tetapi yang dikirimkan adalah matriks pesan asli yang dikalikan dengan matriks A seperti di bawah ini

$$\text{pesan yang dikirim} = \begin{bmatrix} 1 & 3 & 3 \\ 1 & 4 & 3 \\ 1 & 3 & 4 \end{bmatrix} \begin{bmatrix} 1 & 25 & 15 \\ 0 & 16 & 5 \\ 18 & 7 & 9 \end{bmatrix} = \begin{bmatrix} 55 & 64 & 57 \\ 55 & 70 & 62 \\ 73 & 71 & 66 \end{bmatrix}$$

Perhatikan bahwa matriks berisi pesan yang dikirim tidak mudah dibaca oleh orang lain. Lalu bagaimana penerima pesan akan membaca matriks tersebut. Dia harus melakukan dekripsi menggunakan kunci berupa matriks A^{-1}. Matriks kunci ini tentulah matriks yang dirahasiakan berdua oleh pengirim dan penerima pesan agar orang lain tidak mudah membaca. Penerima pesan selanjutnya akan membaca dengan cara

$$\text{pesan terbaca} = \begin{bmatrix} 7 & -3 & -3 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 55 & 64 & 57 \\ 55 & 70 & 62 \\ 73 & 71 & 66 \end{bmatrix} = \begin{bmatrix} 1 & 25 & 15 \\ 0 & 16 & 5 \\ 18 & 7 & 9 \end{bmatrix}$$

yang persis sama dengan matriks pesan asli yang dikirimkan. Setelah itu dengan mudah dia akan membaca kode bilangan dalam matriks ini menjadi “AYO_PERGI”.

Invers Matriks / Matriks Kebalikan

Mendapatkan matriks invers

Untuk memperoleh matriks invers dari suatu matriks persegi yang tidak singular, kita dapat menggunakan fungsi solve( ). Berikut ini ilustrasi memperoleh matriks invers dari matriks tak singular

```
> A <- matrix(c(1, 2, 3, 1, 3, 4, 1, 4, 3), nrow=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    1    3    4
[3,]    1    4    3

> A.invers <- solve(A)
> A.invers
     [,1] [,2] [,3]
[1,]  3.5   -3  0.5
[2,] -0.5    0  0.5
[3,] -0.5    1 -0.5
```

Sekarang perhatikan jika fungsi solve() kita terapkan pada matriks yang singular.

```
> B <- matrix(c(1, 2, 1, 2), nrow=2, byrow=TRUE)
> B
     [,1] [,2]
[1,]    1    2
[2,]    1    2

> solve(B)
Error in solve.default(B) :
  Lapack routine dgesv: system is exactly singular: U[2,2] = 0
```

Karena kita tahu bahwa determinan dari matriks B adalah nol, maka B adalah matriks singular yang tidak mempunya invers. Sehingga, ketika solve() diterapkan pada matriks B, akan diperoleh pesan error seperti di atas.

Ilustrasi lain dari penggunaan fungsi solve() adalah di bawah ini.

```
> D <- matrix(c(1, 0,-3, 2, 5,-4), nrow=2, byrow=TRUE)
> D
     [,1] [,2] [,3]
[1,]    1    0   -3
[2,]    2    5   -4

> solve(D)
Error in solve.default(D) : 'a' (2 x 3) must be square
```

Perhatikan bahwa matriks D berukuran 2 x 3 dan bukanlah matriks persegi. Fungsi solve() tidak dapat bekerja dan akan memberikan error karena matriks yang dioperasikan tidak memenuhi persyaratan dari sisi ukurannya.

Sifat-sifat matriks invers

Berikut ini akan dipaparkan demonstrasi empirik dari terkait dengan sifat-sifat yang melekat pada matriks invers.

1. Deteminan dari matriks invers adalah satu per determinan dari matriks asalnya. Berikut ilustrasinya.

```
> A <- matrix(c(1, 2, 3, 1, 3, 4, 1, 4, 3), nrow=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    1    3    4
[3,]    1    4    3
> det(A) #1
[1] -2


> A.invers <- solve(A)
> A.invers
     [,1] [,2] [,3]
[1,]  3.5   -3  0.5
[2,] -0.5    0  0.5
[3,] -0.5    1 -0.5
> det(A.invers)
[1] -0.5
```

2. Invers dari matriks invers adalah matriks itu sendiri. Berikut ilustrasinya

```
> A <- matrix(c(1, 2, 3, 1, 3, 4, 1, 4, 3), nrow=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    1    3    4
[3,]    1    4    3

> A.invers <- solve(A)
> A.invers
     [,1] [,2] [,3]
[1,]  3.5   -3  0.5
[2,] -0.5    0  0.5
[3,] -0.5    1 -0.5

> solve(A.invers)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    1    3    4
[3,]    1    4    3
```

3. Invers dari putaran matriks A sama dengan putaran dari invers matriks A. Berikut ilustrasinya

```
> A <- matrix(c(1, 2, 3, 1, 3, 4, 1, 4, 3), nrow=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    1    3    4
[3,]    1    4    3

> solve(t(A))
     [,1] [,2] [,3]
[1,]  3.5 -0.5 -0.5
[2,] -3.0  0.0  1.0
```

```
[3,]  0.5  0.5 -0.5

> t(solve(A))
     [,1] [,2] [,3]
[1,]  3.5 -0.5 -0.5
[2,] -3.0  0.0  1.0
[3,]  0.5  0.5 -0.5
```

4. jika dua matriks bersifat tak singular dan berukuran sama, inverse dari perkalian keduanya sama dengan perkalian dari invers masing-masing

```
> A <- matrix(c(1, 2, 3, 1, 3, 4, 1, 4, 3), nrow=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    1    3    4
[3,]    1    4    3

> C <- matrix(c( 1, 0, 3, -2, 3, 1, 1, 1, 2), nrow=3, byrow=TRUE)
> C
     [,1] [,2] [,3]
[1,]    1    0    3
[2,]   -2    3    1
[3,]    1    1    2

> solve(A %*% C)
      [,1] [,2]  [,3]
[1,] -2.05  2.4 -0.85
[2,] -2.15  2.2 -0.55
[3,]  1.85 -1.8  0.45

> solve(C) %*% solve(A)
      [,1] [,2]  [,3]
[1,] -2.05  2.4 -0.85
[2,] -2.15  2.2 -0.55
[3,]  1.85 -1.8  0.45
```

Matriks Kebalikan Umum

Mendapatkan matriks invers umum

Untuk memperoleh matriks kebalikan umum dari suatu matriks menggunakan R kita dapat menggunakan fungsi ginv() yang ada pada package MASS. Berikut ini ilustrasi untuk memperoleh matriks kebalikan umum menggunakan fungsi tersebut.

```
> library(MASS)
 > A <- matrix(c( 1, 0,-1, 1, 0, 2, 2, 2, -1, 4, 5, 3), nrow=3, byrow=TRUE)
> A
     [,1] [,2] [,3] [,4]
[1,]    1    0   -1    1
[2,]    0    2    2    2
[3,]   -1    4    5    3

> B <- ginv(A)
> B
            [,1]        [,2]          [,3]
[1,]  0.27777778  0.11111111 -5.555556e-02
[2,]  0.05555556  0.05555556  5.555556e-02
[3,] -0.22222222 -0.05555556  1.111111e-01
[4,]  0.33333333  0.16666667  2.775558e-17
```

Menggunakan perintah di atas, kita dapatkan bahwa B adalah salah satu matriks kebalikan umum bagi A. Berikut ini hanyalah perintah untuk memastikan bahwa B adalah memang suatu matriks kebalikan umum yang memenuhi persamaan ABA = A.

```
> round(A %*% B %*% A)
   [,1] [,2] [,3] [,4]
[1,]   1   0  -1   1
[2,]   0   2   2   2
[3,]  -1   4   5   3
```

Contoh lain dari proses memperoleh suatu matriks kebalikan umum dari sebuah matriks dan memeriksanya adalah sebagai berikut.

```
> C <- matrix(c(1, 3, 5, 2,-1,-4,  4, 2, 0, 3,-2,-7), nrow=4, byrow=TRUE)
> D <- ginv(C)
> D
           [,1]         [,2]        [,3]         [,4]
[1,] 0.06873823  0.038606403 0.133709981  0.050847458
[2,] 0.06026365  0.001883239 0.071563089 -0.005649718
[3,] 0.05178908 -0.034839925 0.009416196 -0.062146893

> round(C %*% D %*% C)
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2   -1   -4
[3,]    4    2    0
[4,]    3   -2   -7
```

Menyelesaikan Sistem Persamaan Linear

Andaikan kita memiliki sebuah sistem persamaan linear

```
x  + y  + 2z = 1
2x + y       = 2
x  + 2y + 2z = 3
```

dan ingin memperoleh penyelesaian baginya, yaitu mencari berapa nilai x, y, dan z yang memenuhi sistem persamaan tersebut, berikut ini adalah penggunaan R untuk hal tersebut. Sebelumnya terlebih dahulu kita akan mengubah sistem persamaan di atas dalam notasi matriks dalam bentuk:

$$\begin{bmatrix} 1 & 1 & 2 \\ 2 & 1 & 0 \\ 1 & 2 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

atau Ax = b, dengan

$$A = \begin{bmatrix} 1 & 1 & 2 \\ 2 & 1 & 0 \\ 1 & 2 & 2 \end{bmatrix} \text{ dan } b = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Sehingga kita tuliskan dulu

```
> A <- matrix(c(1, 1, 2, 2, 1, 0, 1, 2, 2), nrow=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    1    1    2
[2,]    2    1    0
[3,]    1    2    2

> b <- c(1, 2, 3)
> b
[1] 1 2 3
```

Solusi bagi sistem persamaan tersebut selanjutnya diperoleh dengan cara

```
> solve(A,b)
[1]  0.0  2.0 -0.5
```

yang berarti solusinya adalah x = 0, y = 2, dan z = -0.5.

Karena A adalah matriks tak singular, maka kita juga dapat mencari penyelesaian sistem persamaan linear di atas dengan cara mencari invers dari matriks A dan selanjutnya mengalikannya dengan vektor b seperti yang diilustrasikan pada program berikut

```
> A1 <- solve(A)
> A1
      [,1]  [,2]  [,3]
[1,]  0.50  0.50 -0.50
[2,] -1.00  0.00  1.00
[3,]  0.75 -0.25 -0.25

> A1 %*% b
     [,1]
[1,]  0.0
[2,]  2.0
[3,] -0.5
```

Cara di atas memberikan hasil yang persis sama.

Nilai Karakteristik dan Vektor Karakteristik (Akar Ciri dan Vektor Ciri)

Nilai karakteristik dan vektor karakteristik dari suatu matriks persegi dapat dihasilkan oleh sebuah fungsi saja di R. Fungsi yang dapat dipergunakan untuk hal tersebut adalah eigen( ). Berikut ini adalah ilustrasi untuk memperoleh nilai karakteristik dan vektor karakteristik dari matriks

$$A = \begin{bmatrix} 2 & 2 & 0 \\ 2 & 1 & 1 \\ -7 & 2 & -3 \end{bmatrix}$$

Perintah di R yang dapat diberikan adalah

```
> A <- matrix(c(2, 2, 0, 2, 1, 1, -7, 2, -3), ncol=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    2    2    0
[2,]    2    1    1
[3,]   -7    2   -3
> karakteristik <- eigen(A)
```

Dengan perintah di atas kita akan peroleh nilai karakteristik dan vektor karakteristik yang disimpan pada objek dengan nama "karakteristik". Untuk menampilkan nilai dan vektor karakteristik yang dihasilkan, program yang digunakan adalah:

```
> karakteristik$values
[1] -4  3  1
> karakteristik$vectors
            [,1]       [,2]       [,3]
[1,]  0.07474351 -0.6666667 -0.4364358
[2,] -0.22423053 -0.3333333  0.2182179
[3,]  0.97166562  0.6666667  0.8728716
```

Sekarang perhatikan matriks lain yaitu

$$B = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

Matriks B bersifat simetrik. Nilai dan vektor karakteristik dari matriks B dapat diperoleh dengan cara

```
> B <- matrix(c(2, 1, 1, 1, 2, 1, 1, 1, 2), ncol=3, byrow=TRUE)
> B
     [,1] [,2] [,3]
[1,]    2    1    1
[2,]    1    2    1
[3,]    1    1    2
> karakteristik.B <- eigen(B)
> #nilai karakteristik
> karakteristik.B$values
[1] 4 1 1
```

```
> #vektor karakteristik
> karakteristik.B$vectors
           [,1]       [,2]       [,3]
[1,] -0.5773503  0.0000000  0.8164966
[2,] -0.5773503 -0.7071068 -0.4082483
[3,] -0.5773503  0.7071068 -0.4082483
```

Perhatikan bahwa vektor karakteristik yang berpadanan dengan nilai karakteristik 4 ada di kolom pertama sehingga kita bisa ambil sebagai

```
> v.4 <- karakteristik.B$vectors[,1]
```

Sedangkan vektor karakteristik yang berpadanan dengan nilai karakteristik 1 (yang pertama) ada di kolom kedua, dan kita dapatkan

```
> v.1 <- karakteristik.B$vectors[,2]
```

Seandainya kedua vektor dikalikan (inner product) maka kita dapatkan hasil kalinya adalah 0 seperti di bawah ini yang mengindikasikan bahwa kedua vektor bersifat orthogonal (ingat bahwa vektor karakteristik dari matriks simetrik yang berpadanan dengan nilai karakteristik yang berbeda bersifat ortogonal)

```
> t(v.4) %*% v.1
              [,1]
[1,] -1.110223e-16
```

Sifat-Sifat Nilai Karakteristik (Akar Ciri)

Pada bagian ini akan didemonstrasikan keterpenuhan beberapa sifat nilai dan vektor karakteristik secara empirik menggunakan ilustrasi program R.

1. Nilai karakteristik dari matriks cA adalah c kali nilai karakteristik matriks A

```
> A <- matrix(c(2, 2, 0, 2, 1, 1, -7, 2, -3), ncol=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    2    2    0
[2,]    2    1    1
[3,]   -7    2   -3
> eigen(A)$values
[1] -4  3  1
> B = 2 * A
> eigen(B)$values
[1] -8  6  2
```

2. Nilai karakteristik dari matriks A^2 adalah kuadrat nilai karakteristik matriks A

```
> A <- matrix(c(2, 2, 0, 2, 1, 1, -7, 2, -3), ncol=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    2    2    0
[2,]    2    1    1
[3,]   -7    2   -3
> eigen(A)$values
[1] -4  3  1

> C <- A %*% A
> eigen(C)$values
[1] 16  9  1
```

Nilai 16, 9, dan 1 merupakan kuadrat dari nilai karakteristik asalnya yaitu -4, 3, dan 1.

3. Nilai karakteristik dari matriks A + cI adalah nilai karakteristik A ditambah dengan c, seperti yang didemonstrasikan berikut ini

```
> A <- matrix(c(2, 2, 0, 2, 1, 1, -7, 2, -3), ncol=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    2    2    0
[2,]    2    1    1
[3,]   -7    2   -3
> eigen(A)$values
[1] -4  3  1

> D <- A + 2 * diag(c(1,1,1))
> eigen(D)$values
[1]  5  3 -2
```

4. Perkalian dari nilai karakteristik suatu matriks, sama dengan determinan matriks tersebut. Berikut ilustrasinya.

```
> A <- matrix(c(2, 2, 0, 2, 1, 1, -7, 2, -3), ncol=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    2    2    0
[2,]    2    1    1
[3,]   -7    2   -3
> eigen(A)$values
[1] -4  3  1

> det(A)
[1] -12
```

Nilai karakteristik dari A adalah -4, 3, dan 1 yang kalau dikalikan seluruhnya menjadi 12, dan itu sama dengan nilai determinan matriks A.

5. Penjumlahan dari nilai karakteristik suatu matriks sama dengan teras/trace dari matriks yang bersangkutan. Berikut ilustrasinya.

```
> A <- matrix(c(2, 2, 0, 2, 1, 1, -7, 2, -3), ncol=3, byrow=TRUE)
> A
     [,1] [,2] [,3]
[1,]    2    2    0
[2,]    2    1    1
[3,]   -7    2   -3
> eigen(A)$values
[1] -4  3  1
> sum(eigen(A)$values)
[1] 1.776357e-15
> sum(diag(A))
[1] 0
```