


Simple matrix operations with R

The goal of this exercise is simply to show you how to use R as a matrix desk calculator. There is a script, N:\psy6140\R\matrixR.R containing the lines below, but it is just as easy to type them in to a new R script or the R Studio console. Note that R maintains a “command history”: at the prompt (>) use the arrow keys (↑ and ↓) to scroll through previous commands or look in the History tab. Recall that you can get help on any function via help() or ?

1. Start R Studio from the Start menu or the icon on the desktop.  I recommend that you get into the habit of typing your commands in an R script (File -> New file -> R script, or Ctrl+shift+N).
2. **Creating vectors:** `c()` combines elements into a vector; `seq()` generates a vector as a sequence; `:` is a shorthand; `rep()` repeats elements. Note that an expression assigned to a name (via `<-`) doesn't print the result, but an unassigned expression does print. (`#` indicates a comment; you don't need to type them.)

```
vecA <- c(1, 2, 3, 4)
vecA
vecA <- 1:4           # same
vecA
# same, using seq(), and print also
(vecA <- seq(1,4))
```

```
# using rep() to repeat
vecB <- rep( 1:2, each=3)
vecB
(vecC <- rep( 5:7, each=2))
```

3. **Creating matrices:** matrices are mostly created with the `matrix()` function. Row and or column names can be assigned using `rownames()` and `colnames()`.

```
A <- matrix(vecA, nrow=2, ncol=2, byrow=TRUE)
# default is by columns; no need to name args if given in order
matrix(vecA, 2, 2)
(B <- matrix(vecB, nrow=2, ncol=3, byrow=TRUE))
(C <- matrix(vecC, nrow=3, ncol=2, byrow=TRUE))

# can also supply row & column labels
rownames(A) <- c("Male", "Female")
colnames(B) <- c("Lo", "Med", "Hi")
B
# when typing elements directly, also use c()
(D <- matrix(c( 5, 6, 1, 2, 7, 8), 3, 2, byrow=TRUE))
# or, use scan() to avoid typing all those ', 's
# NB: need a blank line at the end
E <- matrix(scan(), nrow=3, byrow=TRUE)
5 7
3 -1
4 2
```

4. **Joining vectors and matrices:** Use `rbind()` to join by rows, `cbind()` to join by columns.

```
rbind(A,C)
rbind(A, other=c(7,7))
cbind(A,B)
```

5. **Transpose (`t()`), sum (`+`) and matrix multiplication (`%*%`)**

```
B+C # error
B+t(C)
A %*% B
A %*% t(B)
t(B) %*% B
# any number can play...
A %*% B %*% t(B) %*% t(A)
```

6. **Subscripts:** Parts of matrices and vectors can selected using subscripts, enclosed in `[]`.

```
X <- matrix(1:16, 4, 4)
rownames(X) <- c("Al", "Bee", "Cy", "Di")
colnames(X) <- paste("Var", 1:4, sep="")
X
X[1,3] # element
X[1,] # one row
X[,2] # one col
X[1:3,] # several rows
X[,c(1,4)] # several cols
X[1:3, c(1,4)] # extract submatrix
(X[1,] <- X[1,4:1]) # reassign elements
diag(X) <- 1 # change diagonal elements
X
```

7. **Row, column means and other functions:** In addition to standard functions (like `mean()`, `sd()`, `range()`), functions can be “applied” to a matrix over a given margin (1 for rows, 2 for columns).

```
mean(X)
colMeans(X)
rowMeans(X)

apply(X, 1, sum) # sum over rows -> col sums
apply(X, 1, mean)
apply(X, 1, sd)
apply(X, 1, range)

apply(X, 2, sum) # sum over cols -> row sums
apply(X, 2, mean)
apply(X, 2, sd)
```

8. **Some matrix functions:**

```
options(digits=4) # set display options
var(X) # variance-covariance matrix
```

matrixR

```
cor(X)           # correlation matrix
diag(X)          # extract diagonal of matrix
diag(c(1, 2, 3)) # make a diagonal matrix
crossprod(X)     # same as t(X) %*% X
xpx <- crossprod(X)
det(xpx)         # determinant
solve(xpx)       # matrix inverse
```

9. `matlib` functions: The `matlib` package provides a bunch more. I will be using them as we proceed.

```
# install.packages("matlib") # may need to install
library(matlib) # load the package
help(matlib) # access help
R(X) # rank of a matrix
tr(X) # trace of a matrix
len(X) # length of vector or matrix columns
```

When you want to end your R session, save your R script to a file and then type: `q()`, or just close the RStudio window.