

# Lab 1: Basics of Matrix Algebra in R

*Prof. Gaston Sanchez*

*Stat 154, Fall 2017*

## 1) Basic Vector and Matrix manipulations in R

If you have no or minimal experience working with matrices in R, please go over this section. If you are already familiar with this material, skip to the next section.

Consider the following vector `x`:

```
x <- 1:9
```

Use the vector `x` as input of the function `matrix()` to create the following matrix:

	[,1]	[,2]	[,3]
[1,]	1	4	7
[2,]	2	5	8
[3,]	3	6	9

Using `x` and `matrix()`, how would you generate a matrix like this:

	[,1]	[,2]	[,3]
[1,]	1	2	3
[2,]	4	5	6
[3,]	7	8	9

Use `diag()` to create the following identity matrix  $\mathbf{I}_n$  of dimensions  $n \times p = (5, 5)$ :

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	1	0	0	0	0
[2,]	0	1	0	0	0
[3,]	0	0	1	0	0
[4,]	0	0	0	1	0
[5,]	0	0	0	0	1

Consider the following vectors `a1`, `a2`, `a3`:

```
a1 <- c(2, 3, 6, 7, 10)
a2 <- c(1.88, 2.05, 1.70, 1.60, 1.78)
a3 <- c(80, 90, 70, 50, 75)
```

Column-bind the vectors `a1`, `a2`, `a3` to form the following matrix `A`:

```

      a1  a2 a3
1  2 1.88 80
2  3 2.05 90
3  6 1.70 70
4  7 1.60 50
5 10 1.78 75

```

Consider the following vectors b1, b2, b3:

```

b1 <- c(1, 4, 5, 8, 9)
b2 <- c(1.22, 1.05, 3.60, 0.40, 2.54)
b3 <- c(20, 40, 30, 80, 100)

```

Row-bind the vectors b1, b2, b3 to form the following matrix B:

```

      1      2      3      4      5
b1  1.00  4.00  5.0  8.0  9.00
b2  1.22  1.05  3.6  0.4  2.54
b3 20.00 40.00 30.0 80.0 100.00

```

Now compute the matrix products:

- $AB$
- $BA$
- $A^T B^T$
- $B^T A^T B$

R comes with the data frame `iris` which contains five columns:

- `Sepal.Length`
- `Sepal.Width`
- `Petal.Length`
- `Petal.Width`
- `Species` (this is a factor)

```
head(iris)
```

```

##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4          0.2  setosa
## 2          4.9         3.0          1.4          0.2  setosa
## 3          4.7         3.2          1.3          0.2  setosa
## 4          4.6         3.1          1.5          0.2  setosa
## 5          5.0         3.6          1.4          0.2  setosa
## 6          5.4         3.9          1.7          0.4  setosa

```

Take the first four columns of `iris` (i.e. quantitative variables) and form a linear combination with coefficients 1, 2, 3, 4, that is:

$$1 \times \text{Sepal.Length} + 2 \times \text{Sepal.Width} + 3 \times \text{Petal.Length} + 4 \times \text{Petal.Width}$$

Try to obtain this linear combination via a matrix multiplication.

Using matrix functions like transpose `t()` and product `%*%`, write a function `vnorm()` that computes the Euclidean norm (i.e. length) of a vector:  $\|\mathbf{x}\| = \sqrt{\mathbf{x}^T \mathbf{x}}$ .

```
# test vnorm() with 1:5
v <- 1:5
vnorm(v)
```

Given the vector `v <- 1:5`, use `vnorm()` to create a vector `u` such that `u` is of unit norm:  $\|\mathbf{u}\| = 1$ . Recall the a unit vector  $u$  is obtained from a vector  $v$  as:

$$u = \frac{v}{\|v\|}$$

Write a function `is_square()` to check whether the provided matrix is a square matrix.

Write a function `mtrace()` to compute the trace of a square matrix. Use your `is_square()` function to make sure that the input is a square matrix. The trace is defined as the sum of the elements on the diagonal:  $tr(\mathbf{A}) = \sum_{i=1}^n a_{ii}$

Given two square matrices  $\mathbf{A}$  and  $\mathbf{B}$ , verify that your function `mtrace()` is a linear mapping:

- $tr(\mathbf{A} + \mathbf{B}) = tr(\mathbf{A}) + tr(\mathbf{B})$
- $tr(c\mathbf{A}) = c \times tr(\mathbf{A})$ , where  $c$  is a scalar

Trace of products: Given two matrices  $\mathbf{X}$  and  $\mathbf{Y}$  of adequate size, verify that:

$$tr(\mathbf{X}^T \mathbf{Y}) = tr(\mathbf{X} \mathbf{Y}^T) = tr(\mathbf{Y}^T \mathbf{X}) = tr(\mathbf{Y} \mathbf{X}^T)$$


---

## 2) Transformation and Scaling Operations

In this section you will be using the built in data.frame `mtcars` to practice transforming and scaling operations:

```
head(mtcars)
```

##	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
## Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
## Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
## Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
## Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
## Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
## Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

Create a matrix `M` with variables `mpg`, `disp`, `hp`, `drat`, and `wt`.

Use `apply()` to compute the vector containing the means of the columns in `M`

Compute a matrix `Mc` of mean-centered data applying the function `scale()` on `M` (do NOT `scale = TRUE`).

Confirm that variables in `Mc` are mean-centered by calculating the vector of column-means

Use the function `sweep()` to mean-center `M` by *sweeping out* the vector of column means. Compare this result with `Mc` (you should get the same values).

Compute a vector of column maxima.

Use `sweep()` to scale the columns by dividing by the column maxima.

Compute a matrix in which all columns are scaled to have minimum = 0 and maximum = 1

Write a function `dummify()` that takes a factor or a character vector, and which returns a matrix with dummy indicators. Assuming that the factor has  $k$  categories (i.e.  $k$  levels), include an argument `all` that lets you specify whether to return  $k$  binary indicators, or  $k - 1$  indicators. You should be able to call `dummify()` like this:

```
cyl <- factor(mtcars$cyl)
# all categories
CYL1 <- dummify(cyl, all = TRUE)
# minus one category
CYL2 <- dummify(cyl, all = FALSE)
```

Write a function `crosstable()` that takes two factors, and which returns a cross-table between those factors. To create this function you should use your function `dummify()` to compute two dummy matrices, and then multiple them.

```
cyl <- factor(mtcars$cyl)
gear <- factor(mtcars$gear)
xtb <- crosstable(cyl, gear)
```

You should get a table like this:

	3	4	5
4	1	8	2
6	2	4	1
8	12	0	2