Exercises 2. Matrices

1. Suppose

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

(a) Check that $\mathbf{A}^3 = \mathbf{0}$ where $\mathbf{0}$ is a 3×3 matrix with every entry equal to 0.

(b) Replace the third column of $\mathbf{A}$ by the sum of the second and third columns.

2. Create the following matrix $\mathbf{B}$ with 15 rows:

$$\mathbf{B} = \begin{bmatrix} 10 & -10 & 10 \\ 10 & -10 & 10 \\ \cdots & \cdots & \cdots \\ 10 & -10 & 10 \end{bmatrix}$$

Calculate the 3×3 matrix $\mathbf{B}^T\mathbf{B}$. (Look at the help for `crossprod`.)

3. Create a 6×6 matrix `matE` with every entry equal to 0. Check what the functions `row` and `col` return when applied to `matE`. Hence create the 6×6 matrix:

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

4. Look at the help for the function `outer`. Hence create the following patterned matrix:

$$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 5 & 6 & 7 \\ 4 & 5 & 6 & 7 & 8 \end{pmatrix}$$

5. Create the following patterned matrices. In each case, your solution should make use of the special form of the matrix—this means that the solution should easily generalise to creating a larger matrix with the same structure and should not involve typing in all the entries in the matrix.

(a) $$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 & 0 \\ 2 & 3 & 4 & 0 & 1 \\ 3 & 4 & 0 & 1 & 2 \\ 4 & 0 & 1 & 2 & 3 \end{pmatrix}$$

(b) $$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 8 & 9 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 9 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{pmatrix}$$

(c) $$\begin{pmatrix} 0 & 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 \\ 1 & 0 & 8 & 7 & 6 & 5 & 4 & 3 & 2 \\ 2 & 1 & 0 & 8 & 7 & 6 & 5 & 4 & 3 \\ 3 & 2 & 1 & 0 & 8 & 7 & 6 & 5 & 4 \\ 4 & 3 & 2 & 1 & 0 & 8 & 7 & 6 & 5 \\ 5 & 4 & 3 & 2 & 1 & 0 & 8 & 7 & 6 \\ 6 & 5 & 4 & 3 & 2 & 1 & 0 & 8 & 7 \\ 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 & 8 \\ 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \end{pmatrix}$$

6. Solve the following system of linear equations in five unknowns

$$\begin{aligned} x_1 + 2x_2 + 3x_3 + 4x_4 + 5x_5 &= 7 \\ 2x_1 + x_2 + 2x_3 + 3x_4 + 4x_5 &= -1 \\ 3x_1 + 2x_2 + x_3 + 2x_4 + 3x_5 &= -3 \\ 4x_1 + 3x_2 + 2x_3 + x_4 + 2x_5 &= 5 \\ 5x_1 + 4x_2 + 3x_3 + 2x_4 + x_5 &= 17 \end{aligned}$$

by considering an appropriate matrix equation $\mathbf{Ax} = \mathbf{y}$.

Make use of the special form of the matrix $\mathbf{A}$. The method used for the solution should easily generalise to a larger set of equations where the matrix $\mathbf{A}$ has the same structure; hence the solution should not involve typing in every number of $\mathbf{A}$.

7. Create a 6×10 matrix of random integers chosen from 1, 2,..., 10 by executing the following two lines of code:

```
set.seed(75)
aMat <- matrix( sample(10, size=60, replace=T), nr=6)
```

(a) Find the number of entries in each row which are greater than 4.
(b) Which rows contain exactly two occurrences of the number seven?
(c) Find those pairs of columns whose total (over both columns) is greater than 75. The answer should be a matrix with two columns; so, for example, the row $(1, 2)$ in the output matrix means that the sum of columns 1 and 2 in the original matrix is greater than 75. Repeating a column is permitted; so, for example, the final output matrix could contain the rows $(1, 2)$, $(2, 1)$ and $(2, 2)$.
What if repetitions are not permitted? Then, only $(1, 2)$ from $(1, 2)$, $(2, 1)$ and $(2, 2)$ would be permitted.

8. Calculate

(a) $\displaystyle\sum_{i=1}^{20}\sum_{j=1}^{5}\frac{i^4}{(3+j)}$ **(b)** (Hard) $\displaystyle\sum_{i=1}^{20}\sum_{j=1}^{5}\frac{i^4}{(3+ij)}$ **(c)** (Even harder!) $\displaystyle\sum_{i=1}^{10}\sum_{j=1}^{i}\frac{i^4}{(3+ij)}$

Answers to Exercises 2

1. **(a)**

```
( tmp <- matrix( c(1,5,-2,1,2,-1,3,6,-3),nr=3) )
tmp%*%tmp%*%tmp
```

The brackets round the first line ensure the matrix `tmp` is displayed so that we can check that it has been entered correctly.

(b) `tmp[,3] <- tmp[,2]+tmp[,3]`

2.

```
tmp <- matrix(c(10,-10,10), b=T, nc=3, nr=15)
t(tmp)%*%tmp
```

or `crossprod(tmp)`

3.

```
matE <- matrix(0,nr=6,nc=6)
matE[ abs(col(matE)-row(matE))==1 ] <- 1
```

4. `outer(0:4,0:4,"+")`

5. **(a)** `outer(0:4,0:4,"+")%%5`
(b) `outer(0:9,0:9,"+")%%10`
(c) `outer(0:8,0:8,"-")%%9`
Other solutions are possible: for example `matrix(0:4+rep(0:4,times=rep(5,5)),nc=5)` also solves part **(a)**.

6. We have

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} 7 \\ -1 \\ -3 \\ 5 \\ 17 \end{bmatrix} \quad \text{and} \quad \mathbf{A} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 1 & 2 & 3 & 4 \\ 3 & 2 & 1 & 2 & 3 \\ 4 & 3 & 2 & 1 & 2 \\ 5 & 4 & 3 & 2 & 1 \end{bmatrix}$$

Appropriate R code is

```
yVec <- c(7,-1,-3,5,17)
AMat <- matrix(0,nr=5, nc=5)
AMat <- abs(col(AMat)-row(AMat))+1
```

To solve for $\mathbf{x}$, calculate $\mathbf{A}^{-1}\mathbf{y}$, by using the function `solve` to find the inverse of $\mathbf{A}$.
Either `solve(AMat)%*%yVec` which returns the values in $\mathbf{x}$ as a matrix with one column;
or `solve(AMat,yVec)` which returns the values in $\mathbf{x}$ as a vector
or `solve(AMat,matrix(yVec,nc=1) )` which returns the values in $\mathbf{x}$ as a matrix with one column.
If the result of any of these three expressions is saved as `xVec`, then we can check the solution is correct by evaluating `AMat%*%xVec` which returns the values in $\mathbf{y}$ as a matrix with one column in all three cases.

7. **(a)** `apply(aMat, 1, function(x){sum(x>4)})`
(b) `which( apply(aMat,1,function(x){sum(x==7)==2}) )`
(c) Here are two solutions:

```
aMatColSums <- colSums(aMat)
cbind( rep(1:10,rep(10,10)), rep(1:10,10) ) [outer(aMatColSums,aMatColSums,"+")>75,]
```

or

```
aMatColSums <- colSums(aMat)
which( outer(aMatColSums,aMatColSums,"+")>75, arr.ind=T )
```

If we wish to exclude repeats, we can use code such as

```
aMatColSums <- colSums(aMat)
logicalMat <- outer(aMatColSums,aMatColSums,"+")>75
logicalMat[lower.tri(logicalMat,diag=T)] <- F
which(logicalMat, arr.ind=T)
```

8. **(a)** `sum( (1:20)^4 ) * sum( 1/(4:8) )` *or* `sum(outer((1:20)^4,4:8,"/"))`
The answer is 639,215.
(b) `sum( (1:20)^4 / (3 + outer(1:20,1:5,"*")))`
The answer is 89,912.021.
(c) `sum( outer(1:10,1:10,function(i,j){ (i>=j)*i^4/(3+i*j) }) )`
The answer is 6,944.7434.