Elementary Concepts in Linear Algebra

Number Representations

1

a) Convert $1\,110\,000\,101$ from base 2 into base 10 by hand.

Solution:

$$
\begin{aligned}
&1 \cdot 2^8 + 1 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \\
=&1 \cdot 256 + 1 \cdot 128 + 1 \cdot 64 + 0 \cdot 32 + 0 \cdot 16 + 0 \cdot 8 + 1 \cdot 4 + 0 \cdot 2 + 1 \\
=&901 \text{ in base } 10
\end{aligned}
$$

1

b) Convert 852 from base 10 into base 2 by hand.

Solution:

Start	**Integer division by 2**	**Remainder**
852	426	0
426	213	0
213	106	1
106	53	0
53	26	1
26	13	0
13	6	1
6	3	0
3	1	1
1	0	1

Result: $1\,101\,010\,100$ in base 2

3

c) Repeat the above base 2 conversions by utilizing R.

Solution:

```
library(sfsmisc)
strtoi(1110000101, base=2)

## [1] 901

digitsBase(852, base=2)

## Class 'basedInt'(base = 2) [1:1]
##       [,1]
##  [1,]    1
##  [2,]    1
##  [3,]    0
##  [4,]    1
##  [5,]    0
##  [6,]    1
##  [7,]    0
##  [8,]    1
##  [9,]    0
## [10,]    0
```

1

d) Convert $756\,001$ from base 8 into base 10 by hand.

Solution:

$$
\begin{aligned}
&7 \cdot 8^5 + 5 \cdot 8^4 + 6 \cdot 8^3 + 0 \cdot 8^2 + 0 \cdot 8^1 + 1 \\
&7 \cdot 32768 + 5 \cdot 4096 + 6 \cdot 512 + 0 \cdot 64 + 0 \cdot 8 + 1 \\
=&252\,929 \text{ in base 10}
\end{aligned}
$$

1

e) Convert 9530 from base 10 into base 8 by hand.

Solution:

Start	Integer division by 8	Remainder
9530	1191	2
1191	148	7
148	18	4
18	2	2
2	0	2

Result: 22 472 in base 8

3

f) Repeat the above base 2 conversions by utilizing R.

Solution:

```
library(sfsmisc)
strtoi(756001, base=8)

## [1] 252929

digitsBase(9530, base=8)

## Class 'basedInt'(base = 8) [1:1]
##      [,1]
## [1,]    2
## [2,]    2
## [3,]    4
## [4,]    7
## [5,]    2
```

10

g) Write a function `dec_to_bin` that takes a decimal number as input and outputs its binary representation in a vector of 1s and 0s. Instead of using the build-in conversion from R, implement this with help of an appropriate loop. Then test its correctness by calculating the binary representation of 852.

Solution:

```
dec_to_bin <- function(decimal) {
  binary <- c()
  while (decimal > 0) {
    if (decimal %% 2 == 0) {
      binary <- c(0, binary)
    } else {
      binary <- c(1, binary)
    }
    decimal <- floor(decimal / 2)
  }
```

```
  return(binary)
}

dec_to_bin(852) # should return 1,101,010,100

##  [1] 1 1 0 1 0 1 0 1 0 0
```

Polynomials

4

h) Write a function `my_power` in R that calculates the exponentiation of two numbers b^n using a for-loop. Assume that $n \in \mathbb{N}$. Finally, use this function to calculate 7^3 and 5^0.

Solution:

```
my_power <- function(base, exponent) {
  result <- 1
  if (exponent != 0) {
    for(i in 1:exponent) {
      result <- result * base
    }
  }
  return(result)
}

my_power(5, 0)  # should return 1

## [1] 1

my_power(17, 3) # should return 343

## [1] 4913
```

4

i) Create a function `horner` that evaluates a polynomial at given point $x_0 = 3$ by utilizing the Horner scheme for a given vector of polynomial coefficients. For example, the polynomial $2x^4 - 4x^3 - 5x^2 + 7x + 11$ corresponds to a vector representation `c(2, -4, -5, 7, 11)` in R.

Solution:

```
horner <- function(coeffs, x0) {
        value <- 0

        for (i in 1:length(coeffs)) {
          value <- value * x0 + coeffs[i]
        }

        return(value)
}

horner(c(2, -4, -5, 7, 11), 3)

## [1] 41

# double-check with built-in R
2*3^4 - 4*3^3 - 5*3^2 + 7*3 + 11

## [1] 41
```

Linear Algebra

2

j) Calculate the dot product for the following two vectors by hand and with the help of R.

$$\boldsymbol{x} = \begin{bmatrix} 5 \\ -1 \\ 2 \end{bmatrix} \quad \text{and} \quad \boldsymbol{y} = \begin{bmatrix} 8 \\ 7 \\ -3 \end{bmatrix}.$$

Solution:

$$\begin{bmatrix} 5 \\ -1 \\ 2 \end{bmatrix} \cdot \begin{bmatrix} 8 \\ 7 \\ -3 \end{bmatrix} = 5 \cdot 8 + -1 \cdot 7 + 2 \cdot -3 = 40 - 7 - 6 = 27$$

```
drop(c(5, -1, 2) %*% c(8, 7, -3))

## [1] 27
```

2

k) Write a function `dot_product` in R to calculate the dot product of the vectors from the previous exercise. Use a for loop to calculate the result.

Solution:

```
dot_product <- function(vector1, vector2) {
  result <- 0
  for (i in 1:length(vector1)) {
    result <- result + vector1[i]*vector2[i]
  }
  return(result)
}

x <- c(5, -1, 2)
y <- c(8, 7, -3)

dot_product(x,y)

## [1] 27
```

2

l) Write a function `mult_matrix_vector` in R to multiply a matrix by a vector. Use a for loop to calculate the result. Apply your function to A and $\boldsymbol{x}$ defined by

$$A = \begin{bmatrix} 4 & 3 \\ -1 & 7 \\ 0 & 12 \end{bmatrix} \quad \text{and} \quad \boldsymbol{x} = \begin{bmatrix} 3 \\ -2 \end{bmatrix}$$

and then compare to the built-in functionality within R.

Solution:

```
mult_matrix_vector <- function(matrix, vector) {
  result <- c()
  for (row_i in 1:nrow(matrix)) {
    row <- matrix[row_i, ]
    row_result <- dot_product(row, vector)
    result <- c(result, row_result)
```

```
  }
  return(result)
}

A <- matrix(c(4, -1, 0, 3, 7, 12), nrow=3, ncol=2)
x <- c(3, -2)

mult_matrix_vector(A, x)

## [1]   6 -17 -24

mult_matrix_vector(A, x) - A %*% x # post check

##      [,1]
## [1,]    0
## [2,]    0
## [3,]    0
```

The result should be $A\boldsymbol{x} = \begin{bmatrix} 6 \\ -17 \\ -24 \end{bmatrix}$.

2

m) Write a function `mult_matrix_matrix` in R to multiply two matrices. Use a for loop to calculate the result. Apply your function to A and B with

$$A = \begin{bmatrix} 4 & 3 \\ -1 & 7 \\ 0 & 12 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 3 & 2 & 5 & 1 \\ -2 & 0 & 4 & 1 \end{bmatrix}.$$

Hint: The result should be $AB = \begin{bmatrix} 6 & 8 & 32 & 7 \\ -17 & -2 & 23 & 6 \\ -24 & 0 & 48 & 12 \end{bmatrix}$.

Solution:

```
mult_matrix_matrix <- function(matrix1, matrix2) {
  result_rows <- nrow(matrix1)
  result_cols <- ncol(matrix2)
  result <- c()
  for (col_i in 1:ncol(matrix2)) {
    col <- matrix2[, col_i]
```

```
    col_result <- mult_matrix_vector(matrix1, col)
    result <- c(result, col_result)
  }
  return(matrix(result, nrow=result_rows, ncol=result_cols))
}

A <- matrix(c(4, -1, 0, 3, 7, 12), nrow=3, ncol=2)
B <- matrix(c(3, -2, 2, 0, 5, 4, 1, 1), nrow=2, ncol=4)

mult_matrix_matrix(A, B)

##      [,1] [,2] [,3] [,4]
## [1,]    6    8   32    7
## [2,]  -17   -2   23    6
## [3,]  -24    0   48   12
```

3

n) Consider the vector

$$\boldsymbol{x} = \begin{bmatrix} -3 \\ -4 \\ -1 \\ 0 \\ 2 \end{bmatrix}.$$

What are its L^1-, L^2 and 4-norm?

Solution:

```
x <- c(-3, -4, -1, 0, 2)

sum(abs(x)) # L1-norm

## [1] 10

sqrt(sum(x^2)) # L2-norm

## [1] 5.477226

(sum(abs(x)^4))^(1/4) # 4-norm

## [1] 4.337613
```

2

o) Calculate the inverse (or pseudoinverse, depending on what is necessary) for the following matrix. Double-check subsequently.

$$A = \begin{bmatrix} 3 & 7 \\ -1 & 2 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 2 & 8 \\ 3 & 12 \end{bmatrix}$$

Solution:

```
library(MASS)

A <- matrix(c(3,-1, 7,2), ncol=2)
A

##      [,1] [,2]
## [1,]    3    7
## [2,]   -1    2

det(A)  # != 0, thus A is invertible

## [1] 13

solve(A)  # inverse

##            [,1]       [,2]
## [1,] 0.15384615 -0.5384615
## [2,] 0.07692308  0.2307692

A %*% solve(A) - diag(2) # post check

##              [,1] [,2]
## [1,] 0.000000e+00    0
## [2,] 2.775558e-17    0

B <- matrix(c(2,3, 8,12), ncol=2)
B

##      [,1] [,2]
## [1,]    2    8
## [2,]    3   12

det(B)  # == 0, thus B is not invertible
```

```
## [1] 0

ginv(B)  # pseudo-inverse

##             [,1]       [,2]
## [1,] 0.009049774 0.01357466
## [2,] 0.036199095 0.05429864

B*ginv(B) - diag(2)  # post-check

##            [,1]       [,2]
## [1,] -0.9819005  0.1085973
## [2,]  0.1085973 -0.3484163
```

1

p) Find the determinant, eigenvalues and eigenvectors to the matrix

$$A = \begin{bmatrix} 3 & 3 \\ 1 & 1 \end{bmatrix}$$

by hand.

Solution:
To find the eigenvalues, we solve the characteristic equation

$$\begin{aligned} &\det A - \lambda I = 0 \\ \Leftrightarrow &\det \begin{bmatrix} 3-\lambda & 3 \\ 1 & 1-\lambda \end{bmatrix} = 0 \\ \Leftrightarrow &3 - 4\lambda + \lambda^2 - 3 = 0 \\ \Rightarrow &\lambda_1 = 4, \lambda_2 = 0 \end{aligned}$$

To find the corresponding eigenvectors, we solve the equation $(A - \lambda I)\boldsymbol{x} = 0$. Given

$\lambda_1 = 4$, we obtain

$$\begin{aligned}
&\begin{bmatrix} 3-4 & 3 \\ 1 & 1-4 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -1 & 3 \\ 1 & -3 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\
\Rightarrow & -1x_1 + 3x_2 = 0 \\
\Leftrightarrow & x_1 = 3x_2 \\
\Rightarrow & \boldsymbol{v}_1 = \begin{bmatrix} 1 \\ 3 \end{bmatrix}.
\end{aligned}$$

Given $\lambda_2 = 0$, we obtain

$$\begin{aligned}
&\begin{bmatrix} 3-0 & 3 \\ 1 & 1-0 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 3 & 3 \\ 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\
\Rightarrow & 3x_1 + 3x_2 = 0 \\
\Leftrightarrow & x_1 = -1x_2 \\
\Rightarrow & \boldsymbol{v}_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.
\end{aligned}$$

1

q) Calculate the determinates, eigenvalues and eigenvectors of the two matrices

$$A_1 = \begin{bmatrix} 3 & 3 \\ 1 & 1 \end{bmatrix} \quad \text{and} \quad A_2 = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}$$

with the help of R.

Solution:

```
A1 <- matrix(c(3, 1, 3, 1), nrow=2)
A1

##      [,1] [,2]
## [1,]    3    3
## [2,]    1    1

det(A1)

## [1] 0

eigen(A1)
```

```
## eigen() decomposition
## $values
## [1] 4 0
##
## $vectors
##           [,1]       [,2]
## [1,] 0.9486833 -0.7071068
## [2,] 0.3162278  0.7071068
```

```
A2 <- matrix(c(1, 0, 2, 1), nrow=2)
A2
```

```
##      [,1] [,2]
## [1,]    1    2
## [2,]    0    1
```

```
det(A2)
```

```
## [1] 1
```

```
eigen(A2)
```

```
## eigen() decomposition
## $values
## [1] 1 1
##
## $vectors
##      [,1]          [,2]
## [1,]    1 -1.000000e+00
## [2,]    0  1.110223e-16
```

1

r) Are the following two matrices positive or negative definite?

$$A_1 = \begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix} \quad \text{and} \quad A_2 = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

Solution:

```
library(matrixcalc)

A1 <- matrix(c(3, -1, -1, 3), nrow=2)
```

```r
A1
```

```
##      [,1] [,2]
## [1,]    3   -1
## [2,]   -1    3
```

```r
is.negative.definite(A1)
```

```
## [1] FALSE
```

```r
is.positive.definite(A1)
```

```
## [1] TRUE
```

```r
A2 <- matrix(c(1, 0, 0, 2), nrow=2)
A2
```

```
##      [,1] [,2]
## [1,]    1    0
## [2,]    0    2
```

```r
is.negative.definite(A2)
```

```
## [1] FALSE
```

```r
is.positive.definite(A2)
```

```
## [1] TRUE
```