

## **Supplement 2: Basic Splus**

### **1. Naming conventions:**

**Example (Splus):**

A1= 1

a1= 5

A1

a1

### **2. Data structure:**

**(a) Vector**

**(b) Matrix**

**(a) Vector:**

**Example (Splus):**

v1= c(1.1,2.2,3.3,4.4,5.5,6.6)

v1

v1[3] # the third element of v1

1:5

3:7

v1[2:4] # the elements of v1 from the second one to the  
# fourth one

v1[c(1,3,5)] # the first, the third, and the fifth elements of v1

v1[-c(1,3,4)] # the elements of v3 excluding the first, the third,  
# and the fourth elements

**(b) Matrices:**

There are several ways to construct a matrix.

1. We construct a vector first, then convert this vector into a matrix.

**Example (Splus):**

v3

dim(v3)=c(2,5) # specify v3 as a 2 by 5 matrix

v3

dim(v3)=NULL # the dimension of v3 is 0

v3

v5=matrix(v3,2,5,byrow=T) # the matrix v5 is filled by row

v5

## 2. Specify the matrix directly.

The following example is to construct the matrix  $\begin{bmatrix} 1 & 2 \\ 7 & 9 \\ 4 & 8 \end{bmatrix}$ .

### Example (Splus):

```
m1=matrix(0,3,2)
m1
m1[1,]=c(1,2)
m1[2,]=c(7,9)
m1[3,]=c(4,8)
m1
m1[1,2]
m1[,1]
```

## 3. Writing functions:

- (a) Control flow
- (b) Loops

### (a) Control flow:

If, else if, else:

```
if (case 1){ expressions}
else if (case 2){ expressions}
:
else{ last expressions}
```

### Example (Splus):

```
pig=function()
{
  if (x >= 0)
  {
    print("You are a pig")
  }
  else
  {
    print("I am a pig")
  }
}
x= -1
pig()
```

```
x= 3
```

```
print()
```

**Note:**

&&: and

||: or

**Note:**

Multiple conditions can be used within if statements. The syntax is

```
if (condition 1 && condition 2){ expressions }
```

```
if(condition 1 || condition 2){ expressions }
```

**(b) Loops:**

There are 3 statements for loop. The syntaxes are

I.     **for (variable in expression) { expressions }**

II.    **repeat{**

**expressions**

**if (condition) break**

**expressions**

**}**

III.   **while (condition){ expressions }**

**Example (Splus):**

```
for (j in 1:10)
{
    print(j) # loop using for
}
for (j in 2:7)
{
    print(j) # loop using for
    print("Hi")
}
```

```
j=1
```

```
repeat
```

```
{
```

```
    print(j)
```

```
    j=j+1 # loop using repeat
```

```
    if (j > 10) break
```

```
}
```

```

j=1
while (j < 11)
{
    print(j)                      # loop using while
    j=j+1
}

```

#### 4. Specifying argument lists:

**Example (Splus):**

```

pig=function(x)
{
    if (x >= 0){ print("You are a pig") }
    else{ print("I am a pig") }
}

pig(3)
pig(-2)

add=function(x)
{
    s=0
    for (j in 1:x)
    {
        s=s+j
    }
    print(s)
    s
}

add(10)
x1=add(10)
x1

arith1=function(x)
{
    a=sum(x)
    m=prod(x)
    list(suma=a,prod=m)
}

b=1:5
arith1(b)
s1=arith1(b)

```

```

s1$suma
s1$prodm
arith2=function(y,x=1:10)
{
  a=sum(x)
  m=prod(y)
  list(suma=a,prodm=m)
}
d=1:5
arith2(b,d)
arith2(d)
t.test
help(t.test)

```

## 5. Newton-Raphson method:

**Objective:** Find the solutions for *possibly nonlinear* equations

$$\begin{aligned}
 f_1(x) &= f_1(x_1, \dots, x_p) = \mathbf{0} \\
 f_2(x) &= f_2(x_1, \dots, x_p) = \mathbf{0} \\
 &\vdots \\
 f_p(x) &= f_p(x_1, \dots, x_p) = \mathbf{0} \\
 \Leftrightarrow \mathbf{f}(x) &= \begin{bmatrix} f_1(x_1, \dots, x_p) \\ f_2(x_1, \dots, x_p) \\ \vdots \\ f_p(x_1, \dots, x_p) \end{bmatrix} = \mathbf{0}
 \end{aligned}$$

**Newton-Raphson algorithm:**

$$\begin{aligned}
 \hat{x}_{t+1} &= \begin{bmatrix} \hat{x}_{(t+1)1} \\ \hat{x}_{(t+1)2} \\ \vdots \\ \hat{x}_{(t+1)p} \end{bmatrix} = \begin{bmatrix} \hat{x}_{t1} \\ \hat{x}_{t2} \\ \vdots \\ \hat{x}_{tp} \end{bmatrix} - \begin{bmatrix} \frac{\partial f_1(x)}{\partial x_1} & \frac{\partial f_1(x)}{\partial x_2} & \dots & \frac{\partial f_1(x)}{\partial x_p} \\ \frac{\partial f_2(x)}{\partial x_1} & \frac{\partial f_2(x)}{\partial x_2} & \dots & \frac{\partial f_2(x)}{\partial x_p} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_p(x)}{\partial x_1} & \frac{\partial f_p(x)}{\partial x_2} & \dots & \frac{\partial f_p(x)}{\partial x_p} \end{bmatrix}_{x=\hat{x}_t}^{-1} \begin{bmatrix} f_1(\hat{x}_t) \\ f_2(\hat{x}_t) \\ \vdots \\ f_p(\hat{x}_t) \end{bmatrix} \\
 &= \hat{x}_t - \left[ \frac{\partial f(x)}{\partial x} \right]_{x=\hat{x}_t}^{-1} f(\hat{x}_t), t = 0, 1, 2, \dots
 \end{aligned}$$

The converge criteria are:

1.  $\|\hat{x}_{N+1} - \hat{x}_N\| < \varepsilon_1$ ,  $\varepsilon_1$  is some pre-specified small number.
2.  $\|f(\hat{x}_N)\| < \varepsilon_2$ ,  $\varepsilon_2$  is some pre-specified small number.

**Example:**

$f(x) = x^2 - 1$ . Please find the solutions of  $f(x) = 0$  by Newton-Raphson method.

**[code:]**

```
newton1=function(initial,error)
{
    xold = initial
    repeat
    {
        f=xold^2-1
        df=2*xold
        xnew=xold-f/df
        cri1=abs(f)
        cri2=abs(xnew-xold)
        if(cri1 < error && cri2 < error) break
        xold=xnew
    }
    list(xnew)
}
solution=c(newton1(10,0.000001),newton1(-10,0.000001))
solution
```

**Example:**

Please write a program to find all solutions of the following equations to **5 decimal places** of accuracy using Newton-Raphson method:

$$7x^3 - 10x - y - 1 = 0$$
$$8y^3 - 11y + x - 1 = 0$$

with starting point  $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ ,  $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ , and  $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$  Please save the solutions as outputs in a list.

**[code:]**

```
newton2=function(initial,error)
{
    xold = initial
    repeat
    {
        f=c(7*xold[1]^3-10*xold[1]-xold[2]-1,
            8*xold[2]^3-11*xold[2]+xold[1]-1)
```

```

hm=matrix(c(21*xold[1]^2-10,1,-1,24*xold[2]^2-11),2,2)
xnew=xold-solve(hm)%*%f
cri1=sqrt(sum(f^2))
cri2=sqrt(sum((xnew-xold)^2))
if(cri1 < error && cri2 < error) break
xold=xnew
}
list(xnew)
}
solution=c(newton2(c(1,0),0.00001),
           newton2(c(1,1),0.00001),
           newton2(c(1,-1),0.00001))
solution

```