Chapter 2. Numbers in Scilab

1. Natural integers:

An integer in Scilab always contains the "." character, even if the decimal point is optional during input. Example:

```
-->  n = 5
 n =
       5.
```

2. Representation of real numbers:

The basic objects in the Scilab language are real numbers, which are stored as double-precision floating-point numbers following the **IEEE 754** standard. Implementation of IEEE Standard for Floating-Point Arithmetic can be found at https://en.wikipedia.org/wiki/IEEE_754. In this format, each number occupies **64** bits: **1** bit for the sign (positive or negative), **11** bits for the exponent (which sets the scale), and **52** bits for the mantissa (the significant digits). Although up to **18** digits may appear on the screen, only the first **15–16** digits are truly accurate. Scilab displays real numbers in decimal notation, using an optional decimal point "**.**" and the signs "**+**" or "**–**". In scientific notation, Scilab uses the letter "**D**" to indicate the power-of-ten scaling factor. For example, **-123456.789123456789** is represented as **-1.23456789123456789D+5**, where the mantissa is **1.23456789123456789**, the sign is negative, **D+5** indicates the exponent (meaning $\mathbf{10^5}$), the signed exponent is **5** (positive), and the sign is negative for the entire number.

In Scilab, real numbers have an absolute value ranging from about **2×10^{-308}** to **1.8×10^{308}**. The value range can change depending on the version of Scilab, the computer's hardware, and system configuration. For example, when the value exceeds the upper limit (around **`1.8D+308`**), such as **`2D+308`**, Scilab displays **`%inf`**, which represents infinity:

```
-->2D+308
ans=
     %inf
```

The result of a calculation operation is by default displayed with **7** digits after the decimal point (ten characters: the sign and the decimal point included). Example, by default, Scilab displays the number **`-1.23456789123456789D+5`** such as **`-123456.79`**, but maintains full precision internally for accurate calculations.

```
--> -1.23456789123456789D+5
ans=
     -123456.79
```

If more precision (more significant decimals) is needed, the **`format`** function is used. For example:

```
---> format(25)
---> -1.23456789123456789D+5
ans  =
     - 123456.78912345679418650
```

To return to the default format, type:

```
---> format('v', 10)
```

Notice that the valid range for the **`format`** function in Scilab is **[2, 25]**.

To obtain the scientific notation, the **`format e`** function is used. For example:

```
---> x=0.476190547
--> format e
---> x
ans=
     4.762D-01
```

Scilab provides the usual mathematical operations (addition, subtraction, multiplication, division) and elementary functions such as sine, cosine, exponential, etc.

Addition	Subtraction	Multiplication	Division	Exponentiation
+	**-**	*****	**/**	**^ or ****

The evaluation of an expression is executed from left to right, considering the operation priority indicated in the following table:

Operation	Parentheses ( )	Exponentiation	Multiplication and division	Addition and subtraction
Priority	1	2	3	4

For this example:

```
--> (%e^8-5)*(6+9/2)
 ans =
   31247.559
```

The order of calculation is shown below:

$$\underbrace{\left(\underbrace{\underbrace{e^8}_{1} - 5}_{2}\right)}_{1} * \underbrace{\left(\underbrace{6 + \underbrace{9/2}_{1}}_{2}\right)}_{2}$$

In the expression **`(%e^8 - 5) * (6 + 9/2)`**, Scilab follows the standard order of operations. First, it evaluates the parentheses: **`(%e^8 - 5)`** and

`(6 + 9/2)`. Next, it computes the exponent **%e^8**, then performs the division **9/2 = 4.5**. After that, it completes the addition and subtraction: **6 + 4.5 = 10.5** and `%e^8- 5`. Finally, it multiplies the two results to give **(e^8 – 5) × 10.5**.

Some of the most commonly used functions include the following:

Function	**Meaning**
`sin(x)`	sine of x (in radians)
`cos(x)`	cosine of x (in radians)
`tan(x)`	tangent of x (in radians)
`sind(x), cosd(x), tand(x)`	sine, cosine, tangent of x (in degrees)
`asin(x)`	arcsine of x
`acos(x)`	arccosine of x
`atan(x)`	arctangent of x
`sqrt(x)`	square root of x
`abs(x)`	absolute value of x
`exp(x)`	exponential of x
`log(x)`	natural logarithm
`log10(x)`	logarithm to the base 10
`round(x)`	rounds a number to the nearest integer
`floor(x)`	rounds a number down to the nearest integer
`ceil(x)`	rounds a number up to the nearest integer
`sign(x)`	gives the sign of x

3. Complex numbers:

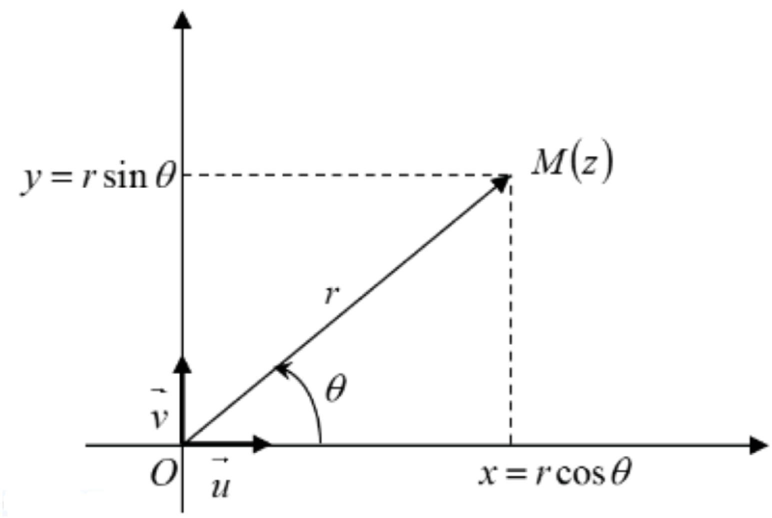
Scilab provides complex numbers that are stored as pairs of real numbers. Complex numbers consist of a real part and an imaginary part. In Scilab, complex numbers are represented using `%i` for the imaginary part. Complex numbers can be written in Cartesian form as **x + i y** or in polar form as **$re^{i\theta}$**, where **x**, **y**, **r**, **θ** are real numbers. Example:

```
--> z1 = 2 + 3 * %i
z 1=
      2. + 3.i
--> z2=-3*exp(4*%i)
z2=
      1.9609309+2.2704075i
```

Scilab allows mathematical operations with complex numbers, such as addition, subtraction, multiplication, and division. Example:

```
--> z1^2

 ans  =

       -5. + 12.i

--> z1 + z2

 ans  =

       3.9609309 + 5.2704075i

--> z1 - z2

 ans  =

       0.0390691 + 0.7295925i

--> z1 * z2

 ans  =

       -2.8893607 + 10.423608i

--> z1 / z2

 ans  =

       1.1925649 + 0.1491086i
```

In the $\mathbb{R}^2$ plane, a specific point **M(x,y)** is represented by the complex number **z = x + i y**, where **x** corresponds to the projection onto the real axis and **y** corresponds to the projection onto the imaginary axis.



The value $\mathbf{r} = |\mathbf{z}|$ (the modulus of **z**) and the argument are given by:

$$\begin{cases} \mathbf{r} = \sqrt{\mathbf{x}^2 + \mathbf{y}^2} \\ \theta = \tan^{-1}\left(\dfrac{\mathbf{y}}{\mathbf{x}}\right) \end{cases}$$

.

Since $\mathbf{x} = \mathbf{r}\cos(\theta)$ and $\mathbf{y} = \mathbf{r}\sin(\theta)$, then $\mathbf{z} = \mathbf{r}(\cos(\theta) + \mathbf{i}\sin(\theta)) = \mathbf{r}\mathbf{e}^{\mathbf{i}\theta}$.

Let **z** be a complex number defined in Scilab

```
--> z = 2 + 3*%i
```

Some of the most commonly used functions on complex numbers include:

Function	Result	Description
`real(z)`	2	Returns the real part of z
`imag(z)`	3	Returns the imaginary part of z
`abs(z)`	3.605513	Returns the modulus of z
`atan(imag(z),real(z))`	0.9827937	Returns the angle θ
`conj(z)`	2 - 3i	The conjugate of z
`imult(z)`	-3 + 2i	The multiplication by i