

The `calculator` and `calculus` packages*

Scientific calculations with L^AT_EX

Robert Fuster
Universitat Politècnica de València
`rfuster@mat.upv.es`

2012/06/10

Abstract

The `calculator` package allows us to use L^AT_EX as a calculator, with which we can perform many of the common scientific calculations (with the limitation in accuracy imposed by the T_EX arithmetic).

This package introduces several new instructions that allow you to do several calculations with integer and decimal numbers using L^AT_EX. Apart from add, multiply or divide, we can calculate powers, square roots, logarithms, trigonometric and hyperbolic functions . . .

In addition, the `calculator` package supports some elementary calculations with vectors in two and three dimensions and square 2×2 and 3×3 matrices.

The `calculus` package adds to the `calculator` package several utilities to use and define various functions and their derivatives, including elementary functions, operations with functions, polar coordinates and vector-valued real functions.

Contents

1	Introduction	3
I	The <code>calculator</code> package	5
2	Predefined numbers	5
3	Operations with numbers	5
3.1	Assignments and comparisons	5
3.2	Real arithmetic	6
3.2.1	The four basic operations	6
3.2.2	Powers with integer exponent	7
3.2.3	Absolute value, integer part and fractional part	7
3.2.4	Truncation and rounding	8

*This document corresponds to `calculator` and `calculus` 1.0a, dated 2012/06/10.

3.3	Integers	8
3.3.1	Integer division, quotient and remainder	9
3.3.2	Greatest common divisor and least common multiple	10
3.3.3	Simplifying fractions	10
3.4	Elementary functions	10
3.4.1	Square roots	10
3.4.2	Exponential and logarithm	10
3.4.3	Trigonometric functions	11
3.4.4	Hyperbolic functions	13
4	Matrix arithmetic	14
4.1	Vector operations	14
4.1.1	Assignments	14
4.1.2	Vector addition and subtraction	14
4.1.3	Scalar-vector product	15
4.1.4	Scalar product and euclidean norm	15
4.1.5	Unit vector parallel to a given vector (normalized vector)	16
4.1.6	Absolute value (in each entry of a given vector)	16
4.2	Matrix operations	16
4.2.1	Assignments	16
4.2.2	Transposed matrix	17
4.2.3	Matrix addition and subtraction	17
4.2.4	Scalar-matrix product	17
4.2.5	Matrix-vector product	18
4.2.6	Product of two square matrices	18
4.2.7	Determinant	19
4.2.8	Inverse matrix	19
4.2.9	Absolute value (in each entry)	20
4.2.10	Solving a linear system	20
II	The calculus package	20
5	What is a <i>function</i>?	21
6	Predefined functions	21
7	Operations with functions	22
8	Polynomial functions	24
9	Vector-valued functions (or parametrically defined curves)	25
10	Vector-valued functions in polar coordinates	26

11 Low-level instructions	26
11.1 The <code>\newfunction</code> declaration and its variants	26
11.2 Vector functions and polar coordinates	27
12 Implementation (calculator)	28
12.1 Internal lengths and special nmbers	28
12.2 Warning messages	28
12.3 Operations with numbers	30
12.4 Matrix arithmetics	47
13 Implementation (calculator)	60
13.1 Error and info messages	60
13.2 New functions	61
13.3 Polynomials	64
13.4 Elementary functions	66
13.5 Operations with functions	68

1 Introduction

The `calculator` package defines some instructions which allow us to realize algebraic operations (and to evaluate elementary functions) in our documents. The operations implemented by the `calculator` package include routines of assignment of variables, arithmetical calculations with real and integer numbers, two and three dimensional vector and matrix arithmetics and the computation of square roots, trigonometrical, exponential, logarithmic and hyperbolic functions. In addition, some important numbers, such as $\sqrt{2}$, π or e , are predefined.

The name of all these commands is spelled in capital letters (with very few exceptions: the commands `\DEGtoRAD` and `\RADtoDEG` and the control sequences that define special numbers, as `\numberPI`) and, in general, they all need one or more mandatory arguments, the first one(s) of which is(are) number(s) and the last one(s) is(are) the name(s) of the command(s) where the results will be stored.¹ The new commands defined in this way work in any \LaTeX mode.

By example, this instruction

```
\MAX{3}{5}{\solution}
```

stores 5 in the command `\solution`. In a similar way,

```
\FRACTIONSIMPLIFY{10}{12}{\numerator}{\denominator}
```

defines `\numerator` and `\denominator` as 5 i 6, respectively.

The *data* arguments should not be necessarily explicit numbers; it may also consist in commands the value of which is a number. This allows us to chain several calculations, since in the following example:

¹Logically, the control sequences that represent special numbers (as `\numberPI`) does not need any argument.

Ex. 1

$$\begin{aligned}\frac{2.5^2}{\sqrt{12}} + e^{3.4} &= \frac{6.25}{3.4641} + 29.96432 \\ &= 1.80421 + 29.96432 \\ &= 31.76854\end{aligned}$$

```
% \tempA=2,5^2
\ SQUARE{2.5}{\tempA}
% \tempB=sqrt(12)
\ SQUAREROOT{12}{\tempB}
% \tempC=exp(3,4)
\ EXP{3.4}{\tempC}
% \divisio=\tempA/tempB
\ DIVIDE{\tempA}{\tempB}{\divisio}
% \sol=\divisio+\tempC
\ ADD{\divisio}{\tempC}{\sol}
\begin{align*}
\frac{2.5^2}{\sqrt{12}}+\mathrm{e}^{3.4}
&= \frac{\tempA}{\tempB}+\tempC \\
&= \divisio+\tempC \\
&=\sol
\end{align*}
```

Observe that, in this example, we have followed exactly the same steps that we would do to calculate $\frac{2.5^2}{\sqrt{12}} + e^{3.4}$ with a standard calculator: We would calculate the square, the root and the exponential and, finally, we would divide and add the results.

It does not matter if the arguments *results* are or not predefined. But these commands act as declarations, so that its scope is local in environments and groups.

Ex. 2

The `\sol` command contains the square of 5:

$$5^2 = 25$$

Now, the `\sol` command is the square root of 5:

$$\sqrt{5} = 2.23605$$

On having gone out of the `center` environment, the command recovers its previous value: 25

```
\ SQUARE{5}\sol
The \texttt{\textbackslash sol}
command contains the square of $5$:
\[5^2=\sol\]
\begin{center}
\ SQUAREROOT{5}\sol
Now, the \texttt{\textbackslash sol}
command is the square root of $5$:
\[\sqrt{5}=\sol\]
\end{center}
On having gone out of the \texttt{center}
environment,
the command recovers its previous value:
\sol
```

The `calculus` package goes a step further and allows us to define and use in a user-friendly manner various functions and their derivatives.

For exemple, using the `calculus` package, you can define the $f(t) = t^2e^t - \cos 2t$ function as follows:

```
% \PRODUCTfunction{\SQUAREfunction}{\EXPfunction}{\tempfunctionA}
% \SCALEVARIABLEfunction{2}{\COSfunction}{\tempfunctionB}
% \SUBTRACTfunction{\tempfunctionA}{\tempfunctionB}{\Ffunction}
```

Then you can compute any value of the new function `\Ffunction` and its derivative: typing

```
\Ffunction{\num}{\sol}{\Dsol}
```

the values of $f(num)$ and $f'(num)$ will be stored in `\sol` and `\Dsol`.

File I

The calculator package

2 Predefined numbers

The calculator package predefines the following numbers:

<code>\numberPI</code>	$3.14159 \approx \pi$	<code>\numberHALFPI</code>	$1.57079 \approx \pi/2$
<code>\numberTHREEHALFPI</code>	$4.71237 \approx 3\pi/2$	<code>\numberTHIRDPI</code>	$1.0472 \approx \pi/3$
<code>\numberQUARTERPI</code>	$0.78539 \approx \pi/4$	<code>\numberFIFTHPI</code>	$0.62831 \approx \pi/5$
<code>\numberSIXTHPI</code>	$0.52359 \approx \pi/6$	<code>\numberTWOPI</code>	$6.28317 \approx 2\pi$
<code>\numberE</code>	$2.71828 \approx e$	<code>\numberINVE</code>	$0.36787 \approx 1/e$
<code>\numberETWO</code>	$7.38902 \approx e^2$	<code>\numberINVETWO</code>	$0.13533 \approx 1/e^2$
<code>\numberLOGTEN</code>	$2.30258 \approx \log 10$		
<code>\numberGOLD</code>	$1.61803 \approx \phi$	<code>\numberINVGOLD</code>	$0.61803 \approx 1/\phi$
<code>\numbersQRTTWO</code>	$1.41421 \approx \sqrt{2}$	<code>\numbersQRTTHREE</code>	$1.73205 \approx \sqrt{3}$
<code>\numbersQRTFIVE</code>	$2.23607 \approx \sqrt{5}$		
<code>\numberCOSXXX</code>	$0.86603 \approx \cos \pi/6$	<code>\numberCOSXLV</code>	$0.70711 \approx \cos \pi/4$

3 Operations with numbers

3.1 Assignments and comparisons

The first command we describe here is used to store a number in a control sequence. The other two commands in this section determine the maximum and minimum of a pair of numbers.

`\COPY{<num>}{<cmd>}` stores the number *num* to the command *cmd*.

Ex. 3

-1.256

```
\COPY{-1.256}{\sol}
\sol
```

`\MAX{<num1>}{<num2>}{<cmd>}` stores in *cmd* the maximum of the numbers *num1* and *num2*.

Ex. 4

$\max(1.256, 3.214) = 3.214$

```
\MAX{1.256}{3.214}{\sol}
\[\max(1.256,3.214)=\sol\]
```

`\MIN{<num1>}{<num2>}{<\cmd>}` stores in `\cmd` the minimum of `num1` and `num2`.

Ex. 5

1.256

`\MIN{1.256}{3.214}{\sol}`
`\sol`

3.2 Real arithmetic

3.2.1 The four basic operations

The following commands calculate the four arithmetical basic operations.

`\ADD{<num1>}{<num2>}{<\cmd>}` Sum of numbers `num1` and `num2`.

Ex. 6

$1.256 + 3.214 = 4.47$

`\ADD{1.256}{3.214}{\sol}`
`$1.256+3.214=\sol$`

`\SUBTRACT{<num1>}{<num2>}{<\cmd>}` Difference `num1-num2`.

Ex. 7

$1.256 - 3.214 = -1.95801$

`\SUBTRACT{1.256}{3.214}{\sol}`
`$1.256-3.214=\sol$`

`\MULTIPLY{<num1>}{<num2>}{<\cmd>}` Product `num1 × num2`.

Ex. 8

$1.256 \times 3.214 = 4.03677$

`\MULTIPLY{1.256}{3.214}{\sol}`
`$1.256\times3.214=\sol$`

`\DIVIDE{<num1>}{<num2>}{<\cmd>}` Quotient `num1 / num2`.²

Ex. 9

$1.256/3.214 = 0.39078$

`\DIVIDE{1.256}{3.214}{\sol}`
`$1.256/3.214=\sol$`

In addition, the `\LENGTHDIVIDE` command divides two lengths and returns a number.

`\LENGTHDIVIDE{<length1>}{<length2>}{<\cmd>}`

Ex. 10

One inch equals 2.54 centimeters.

`\LENGTHDIVIDE{1in}{1cm}{\sol}`
`One inch equals \sol$ centimeters.`

²This command uses a modified version of the division algorithm of Claudio Beccari.

3.2.2 Powers with integer exponent

`\SQUARE{⟨num⟩}{⟨cmd⟩}` Square of the number *num*.

Ex. 11

$$(-1.256)^2 = 1.57751$$

```
\SQUARE{-1.256}{\sol}
$(-1.256)^2=\sol$
```

`\CUBE{⟨num⟩}{⟨cmd⟩}` Cube of *num*.

Ex. 12

$$(-1.256)^3 = -1.98134$$

```
\CUBE{-1.256}{\sol}
$(-1.256)^3=\sol$
```

`\POWER{⟨num⟩}{⟨exp⟩}{⟨cmd⟩}` The *exp* power of *num*.

The exponent, *exp*, must be an integer (if you want to calculate powers with non integer exponents, use the `\EXP` command).

Ex. 13

$$\begin{aligned} (-1.256)^{-5} &= -0.31989 \\ (-1.256)^5 &= -3.1256 \\ (-1.256)^0 &= 1 \end{aligned}$$

```
\POWER{-1.256}{-5}{\sola}
\POWER{-1.256}{5}{\solb}
\POWER{-1.256}{0}{\solc}
\[
\begin{aligned}
(-1.256)^{-5}&=\sola \\
(-1.256)^5&=\solb \\
(-1.256)^0&=\solc
\end{aligned}
\]
```

3.2.3 Absolute value, integer part and fractional part

`\ABSVALUE{⟨num⟩}{⟨cmd⟩}` Absolute value of *num*.

Ex. 14

$$|-1.256| = 1.256$$

```
\ABSVALUE{-1.256}{\sol}
$\left\vert-1.256\right\vert=\sol$
```

`\INTEGERPART{⟨num⟩}{⟨cmd⟩}` Integer part of *num*.³

Ex. 15

The integer part of 1.256 is 1, but the integer part of -1.256 is -2 .

```
\INTEGERPART{1.256}{\sola}
\INTEGERPART{-1.256}{\solb}
The integer part of $1.256$ is $\sola$,
but the integer part of $-1.256$ is $\solb$.
```

³The integer part of x is the largest integer that is less than or equal to x .

`\FLOOR` is an alias of `\INTEGERPART`.

Ex. 16

The integer part of 1.256 is 1.

`\FLOOR{1.256}\sol`
The integer part of \$1.256\$ is \$\sol\$.

`\FRACTIONALPART{<num>}{<cmd>}` Fractional part of *num*.

Ex. 17

0.256
0.744

`\FRACTIONALPART{1.256}\sol`
`\sol`
`\FRACTIONALPART{-1.256}\sol`
`\sol`

3.2.4 Truncation and rounding

`\TRUNCATE[<n>]{<num>}{<cmd>}` truncates the number *num* to *n* decimal places.

`\ROUND[n]{<num>}{<cmd>}` rounds the number *num* to *n* decimal places.

The optional argument *n* may be 0, 1, 2, 3 or 4 (the default is 2).⁴

Ex. 18

1
1.25
1.2568

`\TRUNCATE[0]{1.25688}\sol`
`\sol`
`\TRUNCATE[2]{1.25688}\sol`
`\sol`
`\TRUNCATE[4]{1.25688}\sol`
`\sol`

Ex. 19

1
1.26
1.2569

`\ROUND[0]{1.25688}\sol`
`\sol`
`\ROUND[2]{1.25688}\sol`
`\sol`
`\ROUND[4]{1.25688}\sol`
`\sol`

3.3 Integers

The operations described here are subject to the same restrictions as those referring to decimal numbers. In particular, although \TeX does not have this restriction in its integer arithmetic, the largest integer that can be used is 16383.

⁴Note that `\TRUNCATE[0]` is equivalent to `\INTEGERPART` only for non-negative numbers.

3.3.1 Integer division, quotient and remainder

`\INTEGERDIVISION{<num1>}{<num2>}{<cmd1>}{<cmd2>}` stores in the `<cmd1>` and `<cmd2>` commands the quotient and the remainder of the integer division of the two integers `num1` and `num2`. The remainder is a non-negative number smaller than the divisor.⁵

Ex. 20

$$\begin{aligned} 435 &= 27 \times 16 + 3 \\ 27 &= 435 \times 0 + 27 \\ -435 &= 27 \times (-17) + 24 \\ 435 &= -27 \times (-16) + 3 \\ -435 &= -27 \times 17 + 24 \end{aligned}$$

```
\INTEGERDIVISION{435}{27}{\sola}{\solb}
$435=27\times\sola+\solb$
```

```
\INTEGERDIVISION{27}{435}{\sola}{\solb}
$27=435\times\sola+\solb$
```

```
\INTEGERDIVISION{-435}{27}{\sola}{\solb}
$-435=27\times(\sola)+\solb$
```

```
\INTEGERDIVISION{435}{-27}{\sola}{\solb}
$435=-27\times(\sola)+\solb$
```

```
\INTEGERDIVISION{-435}{-27}{\sola}{\solb}
$-435=-27\times\sola+\solb$
```

`\INTEGERQUOTIENT{<num1>}{<num2>}{<cmd>}` Integer part of the quotient of `num1` and `num2`. These two numbers are not necessarily integers.

Ex. 21

$$\begin{aligned} 16 \\ 0 \\ -17 \end{aligned}$$

```
\INTEGERQUOTIENT{435}{27}{\sol}
\sol
```

```
\INTEGERQUOTIENT{27}{435}{\sol}
\sol
```

```
\INTEGERQUOTIENT{-43.5}{2.7}{\sol}
\sol
```

`\MODULO{<num1>}{<num2>}{<cmd>}` Remainder of the integer division of `num1` and `num2`.

Ex. 22

$$\begin{aligned} 435 &\equiv 3 \pmod{27} \\ -435 &\equiv 24 \pmod{27} \end{aligned}$$

```
\MODULO{435}{27}{\sol}
\[
435 \equiv \sol \pmod{27}
\]
\MODULO{-435}{27}{\sol}
\[
-435 \equiv \sol \pmod{27}
\]
```

⁵The scientific computing systems (such as Matlab, Scilab or Mathematica) do not always return a non-negative residue —especially when the divisor is negative—. However, the most reasonable definition of integer quotient is this one: *the quotient of the division D/d is the largest number q for which $dq \leq D$* . With this definition, the remainder $r = D - dq$ is a non-negative number.

3.3.2 Greatest common divisor and least common multiple

`\GCD{⟨num1⟩}{⟨num2⟩}{⟨cmd⟩}` Greatest common divisor of the integers *num1* and *num2*.

Ex. 23

$$\gcd(435, 27) = 3$$

```
\GCD{435}{27}{\sol}
$\gcd(435,27)=\sol$
```

`\LCM{⟨num1⟩}{⟨num2⟩}{⟨cmd⟩}` Least common multiple of *num1* and *num2*.

Ex. 24

$$\operatorname{lcm}(435, 27) = 3915$$

```
\newcommand{\lcm}{\operatorname{lcm}}
\LCM{435}{27}{\sol}
$\lcm(435,27)=\sol$
```

3.3.3 Simplifying fractions

`\FRACTIONSIMPLIFY{⟨num1⟩}{⟨num2⟩}{⟨cmd1⟩}{⟨cmd2⟩}` stores in the `\cmd1` and `\cmd2` commands the numerator and denominator of the irreducible fraction equivalent to *num1*/*num2*.

Ex. 25

$$435/27 = 145/9$$

```
\FRACTIONSIMPLIFY{435}{27}{\sola}{\solb}
$435/27=\sola/\solb$
```

3.4 Elementary functions

3.4.1 Square roots

`\SQUAREROOT {⟨num⟩}{⟨cmd⟩}` Square root of the number *num*.

Ex. 26

$$\sqrt{1.44} = 1.2$$

```
\SQUAREROOT{1.44}{\sol}
$\sqrt{1.44}=\sol$
```

If the argument *num* is negative, the package returns a warning message.

Instead of `\SQUAREROOT`, you can use the alias `\SQRT`.

3.4.2 Exponential and logarithm

The `\EXP` and `\LOG` commands compute, by default, exponentials and logarithms of the natural base *e*. They admit, however, an optional argument to choose another base.

`\EXP {⟨num⟩}{⟨cmd⟩}` Exponential of the number *num*.

Ex. 27

$$\exp(0.5) = 1.64871$$

`\EXP{0.5}{\sol}`
`$\exp(0.5)=\sol$`

The argument *num* must be in the interval $[-9.704, 9.704]$.⁶

Moreover, the `\EXP` command accepts an optional argument, to compute expressions such as a^x :

`\EXP [⟨num1⟩]{⟨num2⟩}{⟨cmd⟩}` Exponential with base *num1* of *num2*. *num1* must be a positive number.

Ex. 28

$$10^{1.3} = 19.95209$$

$$2^{1/3} = 1.25989$$

`\EXP[10]{1.3}{\sol}`
`$10^{1.3}=\sol$`

`\EXP[2]{0.33333}{\sol}`
`$2^{1/3}=\sol$`

`\LOG {⟨num⟩}{⟨cmd⟩}` logarithm of the number *num*.

Ex. 29

$$\log 0.5 = -0.69315$$

`\LOG{0.5}{\sol}`
`$\log 0.5=\sol$`

`\LOG [⟨num1⟩]{⟨num2⟩}{⟨cmd⟩}` Logarithm in base *num1* of *num2*.

Ex. 30

$$\log_{10} 0.5 = -0.30103$$

`\LOG[10]{0.5}{\sol}`
`$\log_{10} 0.5=\sol$`

3.4.3 Trigonometric functions

The arguments, in functions `\SIN`, `\COS`, ..., are measured in radians. If you measure angles in degrees (sexagesimal or not), use the `\DEGREESSIN`, `\DEGREESCOS`, ... commands.

`\SIN {⟨num⟩}{⟨cmd⟩}` Sine of *num*.

`\COS {⟨num⟩}{⟨cmd⟩}` Cosine of *num*.

`\TAN {⟨num⟩}{⟨cmd⟩}` Tangent of *num*.

⁶9.704 is the logarithm of 16383, the largest number that supports the $\text{T}_{\text{E}}\text{X}$'s arithmetic.

`\COT {⟨num⟩}{⟨cmd⟩}` Cotangent of *num*.

Ex. 31

$$\begin{aligned}\sin \pi/3 &= 0.86601 \\ \cos \pi/3 &= 0.5 \\ \tan \pi/3 &= 1.73201 \\ \cot \pi/3 &= 0.57736\end{aligned}$$

`\SIN{⟨numberTHIRDPI⟩}{⟨sol⟩}`
`$\sin \pi/3=\sol$`

`\COS{⟨numberTHIRDPI⟩}{⟨sol⟩}`
`$\cos \pi/3=\sol$`

`\TAN{⟨numberTHIRDPI⟩}{⟨sol⟩}`
`$\tan \pi/3=\sol$`

`\COT{⟨numberTHIRDPI⟩}{⟨sol⟩}`
`$\cot \pi/3=\sol$`

`\DEGREESSIN {⟨num⟩}{⟨cmd⟩}` Sine of *num* sexagesimal degrees.

`\DEGREESCOS {⟨num⟩}{⟨cmd⟩}` Cosine of *num* sexagesimal degrees.

`\DEGREESTAN {⟨num⟩}{⟨cmd⟩}` Tangent of *num* sexagesimal degrees.

`\DEGREESCOT {⟨num⟩}{⟨cmd⟩}` Cotangent of *num* sexagesimal degrees.

Ex. 32

$$\begin{aligned}\sin 60^\circ &= 0.86601 \\ \cos 60^\circ &= 0.49998 \\ \tan 60^\circ &= 1.73201 \\ \cot 60^\circ &= 0.57736\end{aligned}$$

`\DEGREESSIN{60}{⟨sol⟩}`
`$\sin 60^\circ=\sol$`

`\DEGREESCOS{60}{⟨sol⟩}`
`$\cos 60^\circ=\sol$`

`\DEGREESTAN{60}{⟨sol⟩}`
`$\tan 60^\circ=\sol$`

`\DEGREESCOT{60}{⟨sol⟩}`
`$\cot 60^\circ=\sol$`

The latter commands support an optional argument that allows us to divide the circle in an arbitrary number of *degrees* (not necessarily 360).

`\DEGREESSIN [⟨degrees⟩]{⟨num⟩}{⟨cmd⟩}`

`\DEGREESCOS [⟨degrees⟩]{⟨num⟩}{⟨cmd⟩}`

`\DEGREESTAN [⟨degrees⟩]{⟨num⟩}{⟨cmd⟩}`

`\DEGREESCOT [⟨degrees⟩]{⟨num⟩}{⟨cmd⟩}`

By example, `\DEGREESCOS[400]{50}` computes the cosine of 50 gradians (a right angle has 100 gradians, the whole circle has 400 gradians), which are equivalent to 45 (sexagesimal) degrees or $\pi/4$ radians. Or to 1 *degree*, if we divide the circle into 8 parts!

Ex. 33

0.70709
0.70709
0.7071
0.70709

```
\DEGREESCOS[400]{50}{\sol}  
\sol  
  
\DEGREESCOS{45}{\sol}  
\sol  
  
\COS{\numberQUARTERPI}{\sol}  
\sol  
  
\DEGREESCOS[8]{1}{\sol}  
\sol
```

Moreover, we have a couple of commands to convert between radians and degrees,

`\DEGtoRAD {⟨num⟩}{⟨cmd⟩}` Equivalence in radians of *num* sexagesimal degrees.

`\RADtoDEG {⟨num⟩}{⟨cmd⟩}` Equivalence in sexagesimal degrees of *num* radians.

Ex. 34

1.0472

```
\DEGtoRAD{60}{\sol}  
\sol
```

and two other commands to reduce arguments to basic intervals:

`\REDUCERADIANSANGLE {⟨num⟩}{⟨cmd⟩}` Reduces the arc *num* to the interval $]-\pi, \pi]$.

`\REDUCEDEGREESANGLE {⟨num⟩}{⟨cmd⟩}` Reduces the angle *num* to the interval $]-180, 180]$.

Ex. 35

3.14159
90

```
\MULTIPLY{\numberTWOPI}{10}{\TWENTYPI}  
\ADD{\numberPI}{\TWENTYPI}{\TWENTYONEPI}  
\REDUCERADIANSANGLE{\TWENTYONEPI}{\sol}  
\sol  
  
\REDUCEDEGREESANGLE{3690}{\sol}  
\sol
```

3.4.4 Hyperbolic functions

`\SINH {⟨num⟩}{⟨cmd⟩}` stores in *cmd* the hyperbolic sine of *num*.

`\COSH {⟨num⟩}{⟨cmd⟩}` Hyperbolic cosine of *num*.

`\TANH {⟨num⟩}{⟨cmd⟩}` Hyperbolic tangent of *num*.

`\COTH {⟨num⟩}{⟨cmd⟩}` Hyperbolic cotangent of *num*.

Ex. 36

1.61328
1.89807
0.84995
1.17651

```
\SINH{1.256}\sol  
\sol  
  
\COSH{1.256}\sol  
\sol  
  
\TANH{1.256}\sol  
\sol  
  
\COTH{1.256}\sol  
\sol
```

4 Matrix arithmetic

The calculator package defines the commands described below to operate on vectors and matrices. We only work with two or three-dimensional vectors and 2×2 and 3×3 matrices. Vectors are represented in the form (a_1, a_2) or (a_1, a_2, a_3) ; ⁷ and, in the case of matrices, columns are separated *à la matlab* by semicolons: $(a_{11}, a_{12}; a_{21}, a_{22})$ or $(a_{11}, a_{12}, a_{13}; a_{21}, a_{22}, a_{23}; a_{31}, a_{32}, a_{33})$.

4.1 Vector operations

4.1.1 Assignments

`\VECTORCOPY($\langle x, y \rangle$) ($\langle \backslash cmd1, \backslash cmd2 \rangle$)` copy the entries of vector $(\langle x, y \rangle)$ to the $\backslash cmd1$ and $\backslash cmd2$ commands.

`\VECTORCOPY($\langle x, y, z \rangle$) ($\langle \backslash cmd1, \backslash cmd2, \backslash cmd3 \rangle$)` copy the entries of vector $(\langle x, y, z \rangle)$ to the $\backslash cmd1$, $\backslash cmd2$ and $\backslash cmd3$ commands.

Ex. 37

$(1, -1)$
 $(1, -1, 2)$

```
\VECTORCOPY(1,-1)(\sola,\solb)  
$(\sola,\solb)$  
  
\VECTORCOPY(1,-1,2)(\sola,\solb,\solc)  
$(\sola,\solb,\solc)$
```

4.1.2 Vector addition and subtraction

`\VECTORADD($\langle x_1, y_1 \rangle$) ($\langle x_2, y_2 \rangle$) ($\langle \backslash cmd1, \backslash cmd2 \rangle$)`

`\VECTORADD($\langle x_1, y_1, z_1 \rangle$) ($\langle x_2, y_2, z_2 \rangle$) ($\langle \backslash cmd1, \backslash cmd2, \backslash cmd3 \rangle$)`

`\VECTORSUB($\langle x_1, y_1 \rangle$) ($\langle x_2, y_2 \rangle$) ($\langle \backslash cmd1, \backslash cmd2 \rangle$)`

⁷But they are *column* vectors.

`\VECTORSUB(\langle x1,y1,z1\rangle)(\langle x2,y2,z2\rangle)(\langle \cmd1,\cmd2,\cmd3\rangle)`

Ex. 38

$$\begin{aligned}(1, -1, 2) + (3, 5, -1) &= (4, 4, 1) \\ (1, -1, 2) - (3, 5, -1) &= (-2, -6, 3)\end{aligned}$$

`\VECTORADD(1,-1,2)(3,5,-1)(\sola,\solb,\solc)`
`$(1,-1,2)+(3,5,-1)=(\sola,\solb,\solc)$`

`\VECTORSUB(1,-1,2)(3,5,-1)(\sola,\solb,\solc)`
`$(1,-1,2)-(3,5,-1)=(\sola,\solb,\solc)$`

4.1.3 Scalar-vector product

`\SCALARVECTORPRODUCT{\langle num\rangle}(\langle x,y\rangle)(\langle \cmd1,\cmd2\rangle)`

`\SCALARVECTORPRODUCT{\langle num\rangle}(\langle x,y,z\rangle)(\langle \cmd1,\cmd2,\cmd3\rangle)`

Ex. 39

$$\begin{aligned}2(3, 5) &= (6, 10) \\ 2(3, 5, -1) &= (6, 10, -2)\end{aligned}$$

`\SCALARVECTORPRODUCT{2}(3,5)(\sola,\solb)`
`$2(3,5)=(\sola,\solb)$`

`\SCALARVECTORPRODUCT{2}(3,5,-1)(\sola,\solb,\solc)`
`$2(3,5,-1)=(\sola,\solb,\solc)$`

4.1.4 Scalar product and euclidean norm

`\SCALARPRODUCT(\langle x1,y1\rangle)(\langle x2,y2\rangle){\langle \cmd\rangle}`

`\SCALARPRODUCT(\langle x1,y1,z1\rangle)(\langle x2,y2,z2\rangle){\langle \cmd\rangle}`

`\VECTORNORM(\langle x,y\rangle){\langle \cmd\rangle}`

`\VECTORNORM(\langle x,y,z\rangle){\langle \cmd\rangle}`

Ex. 40

$$\begin{aligned}(1, -1) \cdot (3, 5) &= -2 \\ (1, -1, 2) \cdot (3, 5, -1) &= -4 \\ \|(3, 4)\| &= 5 \\ \|(1, 2, -2)\| &= 3\end{aligned}$$

`\SCALARPRODUCT(1,-1)(3,5){\sol}`
`$(1,-1)\cdot(3,5)=\sol$`

`\SCALARPRODUCT(1,-1,2)(3,5,-1){\sol}`
`$(1,-1,2)\cdot(3,5,-1)=\sol$`

`\VECTORNORM(3,4)\sol`
`$$\left\|(3,4)\right\|=\sol$`

`\VECTORNORM(1,2,-2)\sol`
`$$\left\|(1,2,-2)\right\|=\sol$`

4.1.5 Unit vector parallel to a given vector (normalized vector)

`\UNITVECTOR($\langle x,y \rangle$)($\langle \backslash cmd1, \backslash cmd2 \rangle$)`
`\UNITVECTOR($\langle x,y,z \rangle$)($\langle \backslash cmd1, \backslash cmd2, \backslash cmd3 \rangle$)`

Ex. 41

$(0.59999, 0.79999)$
 $(0.33333, 0.66666, -0.66666)$

`\UNITVECTOR(3,4)(\sola,\solb)`
`(\sola,\solb)`
`\UNITVECTOR(1,2,-2)(\sola,\solb,\solc)`
`(\sola,\solb,\solc)`

4.1.6 Absolute value (in each entry of a given vector)

`\VECTORABSVLUE($\langle x,y \rangle$)($\langle \backslash cmd1, \backslash cmd2 \rangle$)`
`\VECTORABSVLUE($\langle x,y,z \rangle$)($\langle \backslash cmd1, \backslash cmd2, \backslash cmd3 \rangle$)`

Ex. 42

$(3, 4)$
 $(3, 4, 1)$

`\VECTORABSVLUE(3,-4)(\sola,\solb)`
`(\sola,\solb)`
`\VECTORABSVLUE(3,-4,-1)(\sola,\solb,\solc)`
`(\sola,\solb,\solc)`

4.2 Matrix operations

4.2.1 Assignments

`\MATRIXCOPY ($\langle a11,a12;a21,a22 \rangle$) ($\langle \backslash cmd11, \backslash cmd12; \backslash cmd21, \backslash cmd22 \rangle$)`

Use this command to store the matrix $\begin{bmatrix} a11 & a12 \\ a21 & 22 \end{bmatrix}$ in $\backslash cmm11, \backslash cmm12, \backslash cmm21, \backslash cmm22$.
The analogous 3×3 version is

`\MATRIXCOPY ($\langle a11,a12,a13; [\dots], a33 \rangle$) ($\langle \backslash cmd11, \backslash cmd12, \backslash cmd13; [\dots], \backslash cmd33 \rangle$)`

Ex. 43

$\begin{bmatrix} 1 & -1 & 2 \\ 3 & 0 & 5 \\ -1 & 1 & 4 \end{bmatrix}$

`\MATRIXCOPY(1, -1, 2;`
`3, 0, 5;`
`-1, 1, 4)%`
`(\sola,\solb,\solc;`
`\sold,\sole,\self;`
`\solg,\solh,\soli)`
`$\begin{bmatrix}`
`\sola & \solb & \solc \\\`
`\sold & \sole & \self \\\`
`\solg & \solh & \soli`
`\end{bmatrix}$`

Henceforth, we will present only the syntax for commands operating with 2×2 matrices. In all cases, the syntax is similar if we work with 3×3 matrices. In the examples, we will work with either 2×2 or 3×3 matrices.

4.2.2 Transposed matrix

`\TRANPOSEMATRIX (<a11,a12;a21,a22>) (<\cmd11,\cmd12;\cmd21,\cmd22>)`

Ex. 44

$$\begin{bmatrix} 1 & -1 \\ 3 & 0 \end{bmatrix}^T = \begin{bmatrix} 1 & 3 \\ -1 & 0 \end{bmatrix}$$

```
\TRANPOSEMATRIX(1,-1;3,0)%
(\sola,\solb;\solc,\sold)
$\begin{bmatrix}
1 & -1 \\ 3 & 0
\end{bmatrix}^T=\begin{bmatrix}
\sola & \solb \\ \solc & \sold
\end{bmatrix}$
```

4.2.3 Matrix addition and subtraction

`\MATRIXADD (<a11,a12;a21,a22>) (<b11,b12;b21,b22>) (<\cmd11,\cmd12;\cmd21,\cmd22>)`

`\MATRIXSUB (<a11,a12;a21,a22>) (<b11,b12;b21,b22>) (<\cmd11,\cmd12;\cmd21,\cmd22>)`

Ex. 45

$$\begin{bmatrix} 1 & -1 \\ 3 & 0 \end{bmatrix} + \begin{bmatrix} 3 & 5 \\ -3 & 2 \end{bmatrix} = \begin{bmatrix} 4 & 4 \\ 0 & 2 \end{bmatrix}$$

$$\begin{bmatrix} 1 & -1 \\ 3 & 0 \end{bmatrix} - \begin{bmatrix} 3 & 5 \\ -3 & 2 \end{bmatrix} = \begin{bmatrix} -2 & -6 \\ 6 & -2 \end{bmatrix}$$

```
\MATRIXADD(1,-1;3,0)(3,5;-3,2)%
(\sola,\solb;\solc,\sold)
$\begin{bmatrix}
1 & -1 \\ 3 & 0
\end{bmatrix}+
\begin{bmatrix}
3 & 5 \\ -3 & 2
\end{bmatrix}=\begin{bmatrix}
\sola & \solb \\ \solc & \sold
\end{bmatrix}$
```

```
\MATRIXSUB(1,-1;3,0)(3,5;-3,2)%
(\sola,\solb;\solc,\sold)
$\begin{bmatrix}
1 & -1 \\ 3 & 0
\end{bmatrix}-
\begin{bmatrix}
3 & 5 \\ -3 & 2
\end{bmatrix}=\begin{bmatrix}
\sola & \solb \\ \solc & \sold
\end{bmatrix}$
```

4.2.4 Scalar-matrix product

`\SCALARMATRIXPRODUCT{<num>} (<a11,a12;a21,a22>) (<\cmd11,\cmd12;\cmd21,\cmd22>)`

Ex. 46

$$3 \begin{bmatrix} 1 & -1 & 2 \\ 3 & 0 & 5 \\ -1 & 1 & 4 \end{bmatrix} = \begin{bmatrix} 3 & -3 & 6 \\ 9 & 0 & 15 \\ -3 & 3 & 12 \end{bmatrix}$$

```
\SCALARMATRIXPRODUCT{3}(1,-1,2;
                        3, 0,5;
                        -1, 1,4)%
(\sola,\solb,\solc;
\sold,\sole,\solf;
\solg,\solh,\soli)

$3\begin{bmatrix}
1 & -1 & 2 \\ 3 & 0 & 5 \\ -1 & 1 & 4
\end{bmatrix}
\end{bmatrix}
=\begin{bmatrix}
\sola & \solb & \solc \\
\sold & \sole & \solf \\
\solg & \solh & \soli
\end{bmatrix}$
```

4.2.5 Matrix-vector product

`\MATRIXVECTORPRODUCT ($\langle a_{11},a_{12};a_{21},a_{22} \rangle$) ($\langle x,y \rangle$) ($\langle \text{cmd1},\text{cmd2} \rangle$)`

Ex. 47

$$\begin{bmatrix} 1 & -1 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 3 \\ 5 \end{bmatrix} = \begin{bmatrix} -2 \\ 10 \end{bmatrix}$$

```
\MATRIXVECTORPRODUCT(1,-1;
                      0, 2)(3,5)(\sola,\solb)

$\begin{bmatrix}
1 & -1 \\ 0 & 2
\end{bmatrix}
\begin{bmatrix}
3 \\ 5
\end{bmatrix}
=\begin{bmatrix}
\sola \\ \solb
\end{bmatrix}$
```

4.2.6 Product of two square matrices

`\MATRIXPRODUCT ($\langle a_{11},a_{12};a_{21},a_{22} \rangle$) ($\langle b_{11},b_{12};b_{21},b_{22} \rangle$) ($\langle \text{cmd11},\text{cmd12};\text{cmd21},\text{cmd22} \rangle$)`

Ex. 48

$$\begin{bmatrix} 1 & -1 & 2 \\ 3 & 0 & 5 \\ -1 & 1 & 4 \end{bmatrix} \begin{bmatrix} 3 & 5 & -1 \\ -3 & 2 & -5 \\ 1 & -2 & 3 \end{bmatrix} = \begin{bmatrix} 8 & -1 & 10 \\ 14 & 5 & 12 \\ -2 & -11 & 8 \end{bmatrix}$$

```
\MATRIXPRODUCT(1,-1,2;3,0,5;-1,1,4)%
(3,5,-1;-3,2,-5;1,-2,3)%
(\sola,\solb,\solc;
\sold,\sole,\solf;
\solg,\solh,\soli)
\begin{multline*}
\begin{bmatrix}
1 & -1 & 2 \\ 3 & 0 & 5 \\ -1 & 1 & 4
\end{bmatrix}
\begin{bmatrix}
3 & 5 & -1 \\ -3 & 2 & -5 \\ 1 & -2 & 3
\end{bmatrix}
\\
= \begin{bmatrix}
8 & -1 & 10 \\ 14 & 5 & 12 \\ -2 & -11 & 8
\end{bmatrix}
\end{multline*}
```

4.2.7 Determinant

`\DETERMINANT (<a11,a12;a21,a22>) {<\cmd>}`

Ex. 49

$$\begin{vmatrix} 1 & -1 & 2 \\ 3 & 0 & 5 \\ -1 & 1 & 4 \end{vmatrix} = 18$$

```
\DETERMINANT(1,-1,2;3,0,5;-1,1,4){\sol}
\SpecialUsageIndex{\DETERMINANT}%
$\begin{vmatrix}
1 & -1 & 2 \\ 3 & 0 & 5 \\ -1 & 1 & 4
\end{vmatrix}=\sol$
```

4.2.8 Inverse matrix

`\INVERSEMATRIX (<a11,a12;a21,a22>) (<\cmd11,\cmd12;\cmd21,\cmd22>)`

Ex. 50

$$\begin{bmatrix} 1 & -1 \\ 3 & 5 \end{bmatrix}^{-1} = \begin{bmatrix} 0.625 & 0.125 \\ -0.375 & 0.125 \end{bmatrix}$$

```
\INVERSEMATRIX(1,-1;3,5){%
\sola,\solb;\solc,\sold)
$\begin{bmatrix}
1 & -1 \\ 3 & 5
\end{bmatrix}^{-1}=
\begin{bmatrix}
0.625 & 0.125 \\ -0.375 & 0.125
\end{bmatrix}
$
```

If the given matrix is singular, the calculator package returns a warning message and the `\cmd11`, ..., commands are marked as undefined.

4.2.9 Absolute value (in each entry)

`\MATRIXABSVLUE` ($\langle a11, a12; a21, a22 \rangle$) ($\langle \backslash cmd11, \backslash cmd12; \backslash cmd21, \backslash cmd22 \rangle$)

Ex. 51

$$\begin{bmatrix} 1 & 1 & 2 \\ 3 & 0 & 5 \\ 1 & 1 & 4 \end{bmatrix}$$

```
\MATRIXABSVLUE(1,-1,2;3,0,5;-1,1,4)%
(\sola,\solb,\solc;
\sold,\sole,\self;
\solg,\solh,\soli)
$\begin{bmatrix}
\sola & \solb & \solc \\
\sold & \sole & \self \\
\solg & \solh & \soli
\end{bmatrix}$
\end{bmatrix}$
```

4.2.10 Solving a linear system

`\SOLVELINEARSYSTEM` ($\langle a11, a12; a21, a22 \rangle$) ($\langle b1, b2 \rangle$) ($\langle \backslash cmd1, \backslash cmd2 \rangle$) solves the linear system $\begin{pmatrix} a11 & a12 \\ a21 & a22 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b1 \\ b2 \end{pmatrix}$ and stores the solution in ($\backslash cmd1, \backslash cmd2$).

Ex. 52

Solving the linear system

$$\begin{bmatrix} 1 & -1 & 2 \\ 3 & 0 & 5 \\ -1 & 1 & 4 \end{bmatrix} X = \begin{bmatrix} -4 \\ 4 \\ -2 \end{bmatrix}$$

we obtain $X = \begin{bmatrix} 3 \\ 5 \\ -1 \end{bmatrix}$

```
\SOLVELINEARSYSTEM(1,-1,2;3,0,5;-1,1,4)%
(-4,4,-2)%
(\sola,\solb,\solc)
Solving the linear system
\[
\begin{bmatrix}
1 & -1 & 2 \\
3 & 0 & 5 \\
-1 & 1 & 4
\end{bmatrix} \mathsf{X} = \begin{bmatrix}
-4 \\
4 \\
-2
\end{bmatrix}
\end{bmatrix}
\]
we obtain
$\mathsf{X} = \begin{bmatrix}
\sola \\
\solb \\
\solc
\end{bmatrix}$
```

If the given matrix is singular, the package `calculator` returns a warning message. When system is indeterminate, in the bi-dimensional case one of the solutions is computed; if the system is incompatible, then the `\sola`, ..., commands are marked as undefined. For three equations systems, only determinate systems are solved.⁸

⁸This is the only command that does not behave the same way with 2×2 and 3×3 matrices.

File II

The calculus package

5 What is a *function*?

From the point of view of this package, a *function* f is a pair of formulae: the first one calculates $f(t)$; the other, $f'(t)$. Therefore, any function is applied using three arguments: the value of the variable t , and two command names where $f(t)$ and $f'(t)$ will be stored. For example,

```
\SQUAREfunction{\num}{\sol}{\Dsol}
```

computes $f(t) = t^2$ and $f'(t) = 2t$ (where $t = \text{num}$), and stores the results in the commands `\sol` and `\Dsol`.⁹

Ex. 53

If $f(t) = t^2$, then

$$f(3) = 9 \text{ and } f'(3) = 6$$

```
\SQUAREfunction{3}{\sol}{\Dsol}
If $f(t)=t^2$, then
\[
  f(3)=\sol \mbox{ and } f'(3)=\Dsol
\]
```

For all functions defined here, you must use the following syntax:

```
\functionname{\num}{\cmd1}{\cmd2}
```

being num a number (or a command whose value is a number), and `\cmd1` and `\cmd2` two control sequence names where the values of the function and its derivative (in this number) will be stored.

The key difference between this *functions* and the instructions defined in the *calculator* package is the inclusion of the derivative; for example, the `\SQUARE{3}{\sol}` instruction computes, only, the square power of number 3, while `\SQUAREfunction{3}{\sol}{\Dsol}` finds, also, the corresponding derivative.

6 Predefined functions

The *calculus* package predefines the most commonly used elementary functions, and includes several utilities for defining new ones. The predefined functions are the following:

⁹Do not spect any control about the existence or differentiability of the function; if the function or the derivative are not well defined, a $\text{T}_{\text{E}}\text{X}$ error will occur.

<code>\ZEROfunction</code>	$f(t) = 0$	<code>\ONEfunction</code>	$f(t) = 1$
<code>\IDENTITYfunction</code>	$f(t) = t$	<code>\RECIPROCALfunction</code>	$f(t) = 1/t$
<code>\SQUAREfunction</code>	$f(t) = t^2$	<code>\CUBEfunction</code>	$f(t) = t^3$
<code>\SQRTfunction</code>	$f(t) = \sqrt{t}$		
<code>\EXPfunction</code>	$f(t) = \exp t$	<code>\LOGfunction</code>	$f(t) = \log t$
<code>\COSfunction</code>	$f(t) = \cos t$	<code>\SINfunction</code>	$f(t) = \sin t$
<code>\TANfunction</code>	$f(t) = \tan t$	<code>\COTfunction</code>	$f(t) = \cot t$
<code>\COSHfunction</code>	$f(t) = \cosh t$	<code>\SINHfunction</code>	$f(t) = \sinh t$
<code>\TANHfunction</code>	$f(t) = \tanh t$	<code>\COTHfunction</code>	$f(t) = \coth t$
<code>\HEAVISIDEfunction</code>	$f(t) = \begin{cases} 0 & \text{si } t < 0 \\ 1 & \text{si } t \geq 0 \end{cases}$		

In the following example, we use the `\LOGfunction` function to compute a table of the log function and its derivative.

Ex. 54

x	$\log x$	$\log' x$
1	0	1
2	0.69315	0.5
3	1.0986	0.33333
4	1.38629	0.25
5	1.60942	0.2
6	1.79176	0.16666

```

 $\begin{array}{c}
x \ \& \log x \ \& \log' x \\
\LOGfunction{1}{\logx}{\Dlogx} \\
1 \ \& \logx \ \& \Dlogx \\
\LOGfunction{2}{\logx}{\Dlogx} \\
2 \ \& \logx \ \& \Dlogx \\
\LOGfunction{3}{\logx}{\Dlogx} \\
3 \ \& \logx \ \& \Dlogx \\
\LOGfunction{4}{\logx}{\Dlogx} \\
4 \ \& \logx \ \& \Dlogx \\
\LOGfunction{5}{\logx}{\Dlogx} \\
5 \ \& \logx \ \& \Dlogx \\
\LOGfunction{6}{\logx}{\Dlogx} \\
6 \ \& \logx \ \& \Dlogx \\
\end{array}$ 

```

7 Operations with functions

We can define new functions using the following *operations* (the last argument is the name of the new function):

`\CONSTANTfunction{<num>}{<\Function>}` defines `\Function` as the constant function `num`.

Example. Definition of the $F(t) = 5$ function:

`\CONSTANTfunction{5}{\F}`

`\SUMfunction{<\function1>}{<\function2>}{<\Function>}` defines `\Function` as the sum of functions `\function1` and `\function2`.

Example. Definition of the $F(t) = t^2 + t^3$ function:

`\SUMfunction{\SQUAREfunction}{\CUBEfunction}{\F}`

`\SUBTRACTfunction{<\function1>}{<\function2>}{<\Function>}` defines `\Function` as the difference of functions `\function1` and `\function2`.

Example. Definition of the $F(t) = t^2 - t^3$ function:

`\SUBTRACTfunction{\SQUAREfunction}{\CUBEfunction}{\F}`

`\PRODUCTfunction{\function1}{\function2}{\Function}` defines `\Function` as the product of functions `\function1` and `\function2`

Example. Definition of the $F(t) = e^t \cos t$ function:

`\PRODUCTfunction{\EXPfunction}{\COSfunction}{\F}`

`\QUOTIENTfunction{\function1}{\function2}{\Function}` defines `\Function` as the quotient of functions `\function1` and `\function2`.

Example. Definition of the $F(t) = e^t / \cos t$ function:

`\QUOTIENTfunction{\EXPfunction}{\COSfunction}{\F}`

`\COMPOSITIONfunction{\function1}{\function2}{\Function}` defines `\Function` as the composition of functions `\function1` and `\function2`.

Example. Definition of the $F(t) = e^{\cos t}$ function:

`\COMPOSITIONfunction{\EXPfunction}{\COSfunction}{\F}`

(note than `\COMPOSITIONfunction{f}{g}{\F}` means $\F = f \circ g$).

`\SCALEfunction{\num}{\function}{\Function}` defines `\Function` as the product of number `\num` and function `\function`.

Example. Definition of the $F(t) = 3 \cos t$ function:

`\SCALEfunction{3}{\COSfunction}{\F}`

`\SCALEVARIABLEfunction{\num}{\function}{\Function}` scales the variable by factor `\num` and then applies the function `\function`.

Example. Definition of the $F(t) = \cos 3t$ function:

`\SCALEVARIABLEfunction{3}{\COSfunction}{\F}`

`\POWERfunction{\function}{\num}{\Function}` defines `\Function` as the power of function `\function` to the exponent `\num` (a positive integer). Example. Definition of the $F(t) = t^5$ function:

`\POWERfunction{\IDENTITYfunction}{5}{\F}`

`\LINEARCOMBINATIONfunction{\num1}{\function1}{\num2}{\function2}{\Function}` defines `\Function` as the linear combination of functions `\function1` and `\function2` multiplied, respectively, by numbers `\num1` and `\num2`.

Example. Definition of the $F(t) = 2t - 3 \cos t$ function:

`\LINEARCOMBINATIONfunction{2}{\IDENTITYfunction}{-3}{\COSfunction}{\F}`

By combining properly this operations and the predefined functions, many elementary functions can be defined.

Ex. 55

If

$$f(t) = 3t^2 - 2e^{-t} \cos t$$

then

$$f(5) = 74.99619$$

$$f'(5) = 29.99084$$

```
% exp(-t)
\SCALEVARIABLEfunction
{-1}\{EXPfunction}
{\NEGEXPfunction}

% exp(-t)cos(t)
\PRODUCTfunction
{\NEGEXPfunction}
{\COSfunction}
{\NEGEXPCOSfunction}

% 3t^2-2exp(-t)cos(t)
\LINEARCOMBINATIONfunction
{3}\{SQUAREfunction}
{-2}\{\NEGEXPCOSfunction}
{\myfunction}

\myfunction{5}\{sol}\{Dsol}

If
\[
f(t)=3t^2-2\mathrm{e}^{-t}\cos t
\]
then
\[
\begin{gathered}
f(5)=\sol\\
f'(5)=\Dsol
\end{gathered}
\]
```

8 Polynomial functions

Although polynomial functions can be defined using linear combinations of power functions, to facilitate our work, the `calculus` package includes the following commands to define more easily the polynomials of 1, 2, and 3 degrees: `\newlpoly` (new *linear* polynomial), `\newqpoly` (new *quadratic* polynomial), and `\newcpoly` (new *cubic* polynomial):

`\newlpoly{\Function}\{a}\{b}` stores the $p(t) = a + bt$ function in the `\Function` command.

`\newqpoly{\Function}\{a}\{b}\{c}` stores the $p(t) = a + bt + ct^2$ function in the `\Function` command.

`\newcpoly{\Function}\{a}\{b}\{c}\{d}` stores the $p(t) = a + bt + ct^2 + dt^3$ function in the `\Function` command.

Ex. 56

$$p'(2) = 8$$

```
% \mypoly=1-x^2+x^3
\newcpoly{\mypoly}{1}{0}{-1}{1}
\mypoly{2}{\sol}{\Dsol}
$p'(2)=\Dsol$
```

These declarations behave similarly to the declaration `\newcommand`: If the name you want to assign to the new function is that of an already defined command, the `calculus` package returns an error message and do not redefines this command. To obtain any alternative behavior, our package includes three other sets of declarations:

`\renewlpoly`, `\renewqpoly`, `\renewcpoly` redefine the already existing command `\Function`. If this command does not exist, then it is not defined and an error message occurs.

`\ensurelpoly`, `\ensureqpoly`, `\ensurecpoly` define a new function. If the command `\Function` already exists, it is not redefined.

`\forcelpoly`, `\forceqpoly`, `\forcecpoly` define a new function. If the command `\Function` already exists, it is redefined.

9 Vector-valued functions (or parametrically defined curves)

The instruction

```
\PARAMETRICfunction{\Xfunction}{\Yfunction}{\myvectorfunction}
```

defines the new vector-valued function $f(t) = (x(t), y(t))$.

The first and second arguments are a pair of functions already defined and, the third, the name of the new function we define. Once we have defined them, the new vector functions requires five arguments:

```
\myvectorfunction{\num}{\cmd1}{\cmd2}{\cmd3}{\cmd4}
```

where

- num is a number t ,
- `\cmd1` and `\cmd2` are two command names where the values of the $x(t)$ function and its derivative $x'(t)$ will be stored, and
- `\cmd3` and `\cmd4` will store $y(t)$ and $y'(t)$.

In short, in this context, a vector function is a pair of scalar functions.

Instead of `\PARAMETRICfunction` we can use the alias `\VECTORfunction`.

Ex. 57

For the $f(t) = (t^2, t^3)$ function we have

$$f(4) = (16, 64), f'(4) = (8, 48)$$

```
For the $f(t)=(t^2,t^3)$ function we have
\VECTORfunction
{\SQUAREfunction}{\CUBEfunction}{\F}

\F{4}{\solx}{\Dsolx}{\soly}{\Dsoly}

\[
f(4)=(\solx,\soly), f'(4)=(\Dsolx,\Dsoly)
\]
```

10 Vector-valued functions in polar coordinates

The following instruction:

```
\POLARfunction{\rfunction}{\Polarfunction}
```

declares the vector function $f(\phi) = (r(\phi) \cos \phi, r(\phi) \sin \phi)$. The first argument is the $r = r(\phi)$ function, (an already defined function). For example, we can define the *Archimedean spiral* $r(\phi) = 0,5\phi$, as follows:

```
\SCALEfunction{0.5}{\IDENTITYfunction}{\rfunction}
\POLARfunction{\rfunction}{\archimedes}
```

11 Low-level instructions

Probably, many users of the package will not be interested in the implementation of the commands this package includes. If this is your case, you can ignore this section.

11.1 The \newfunction declaration and its variants

All the functions predefined by this package use the \newfunction declaration. This control sequence works as follows:

```
\newfunction{\Function}{\Instructions to compute \y and \Dy from \t}
```

where the second argument is the list of the instructions you need to run to calculate the value of the function \mathbf{y} and the derivative \mathbf{Dy} in the \mathbf{t} point.

For example, if you want to define the $f(t) = t^2 + e^t \cos t$ function, whose derivative is $f'(t) = 2t + e^t(\cos t - \sin t)$, using the high-level instructions we defined earlier, you can write the following instructions:

```
\PRODUCTfunction{\EXPfunction}{\COSfunction}{\ffunction}
\SUMfunction{\SQUAREfunction}{\ffunction}{\Ffunction}
```

But you can also define this function using the \newfunction command as follows:

```
\newfunction{\Ffunction}{%
  \SQUARE{\t}{\tempA}          % A=t^2
  \EXP{\t}{\tempB}             % B=e^t
  \COS{\t}{\tempC}             % C=cos(t)
  \SIN{\t}{\tempD}             % D=sin(t)
  \MULTIPLY{2}{\t}{\tempE}      % E=2t
  \MULTIPLY{\tempB}{\tempC}{\tempC} % C=e^t cos(t)
  \MULTIPLY{\tempB}{\tempD}{\tempD} % D=e^t sin(t)
  \ADD{\tempA}{\tempC}{\y}      % y=t^2 + e^t cos(t)
  \ADD{\tempE}{\tempC}{\tempC}  % C=t^2 + e^t cos(t)
  \SUBTRACT{\tempC}{\tempD}{\Dy} % y'=t^2 + e^t cos(t) - e^t sin(t)
}
```

It must be said, however, that the `\newfunction` declaration behaves similarly to `\newcommand` or `\newlpoly`: If the name you want to assign to the new function is that of an already defined command, the `calculus` package returns an error message and does not redefine this command. To obtain any alternative behavior, our package includes three other versions of the `\newfunction` declarations: the `\renewfunction`, `\ensurefunction` and `\forcefunction` declarations. Each of these declarations behaves differently:

`\newfunction` defines a new function. If the command `\Function` already exists, it is not redefined and an error message occurs.

`\renewfunction` redefines the already existing command `\Function`. If this command does not exist, then it is not defined and an error message occurs.

`\ensurefunction` defines a new function. If the command `\Function` already exists, it is not redefined.

`\forcefunction` defines a new function. If the command `\Function` already exists, it is redefined.

11.2 Vector functions and polar coordinates

You can (re)define a vector function $f(t) = (x(t), y(t))$ using the `\newvectorfunction` declaration or any of its variants `\renewvectorfunction`, `\ensurevectorfunction` and `\forcevectorfunction`:

`\newvectorfunction{\Function}{\Instructions to compute \x, \Dx, \y and \Dy from \t}`

For example, you can define the function $f(t) = (t^2, t^3)$ in the following way:

```
\newvectorfunction{\F}{%
  \SQUARE{\t}{\x}      % x=t^2
  \MULTIPLY{2}{\t}{\Dx} % x'=2t
  \CUBE{\t}{\y}         % y=t^3
  \MULTIPLY{3}{\x}{\Dy} % y'=3t^2
}
```

Finally, to define the $r = r(\phi)$ function, in polar coordinates, we have the declarations `\newpolarfunction`, `\renewpolarfunction`, `\ensurepolarfunction` and `\forcepolarfunction`.

`\newpolarfunction{\Function}{\Instructions to compute \r and \Dr from \t}`

For example, you can define the *cardioid* curve $r(\phi) = 1 + \cos \phi$, using high level instructions,

```
\SUMfunction{\ONEfunction}{\COSfunction}{\ffunction} % y=1 + cos t
\POLARfunction{\ffunction}{\cardioid}
```

or, with the `\newpolarfunction` declaration,

```

\newpolarfunction{\cardioide}{%
  \COS{\t}{\r}
  \ADD{1}{\r}{\r}          % r=1+cos t
  \SIN{\t}{\Dr}
  \MULTIPLY{-1}{\Dr}{\Dr} % r'=-sin t
}

```

12 Implementation (calculator)

```

1 (*calculator)
2 \NeedsTeXFormat{LaTeX2e}
3 \ProvidesPackage{calculator}[2012/06/10 v.1.0a]

```

12.1 Internal lengths and special nmbers

`\cctr@lengtha` and `\cctr@lengthb` will be used in internal calculations and comparisons.

```

4 \newdimen\cctr@lengtha
5 \newdimen\cctr@lengthb

```

`\cctr@epsilon` `\cctr@epsilon` will store the closest to zero length in the \TeX arithmetic: one scaled point ($1\text{ sp} = 1/65536\text{ pt}$). This means the smallest positive number will be $0.00002 \approx 1/65536 = 1/2^{16}$.

```

6 \newdimen\cctr@epsilon
7 \cctr@epsilon=1sp

```

`\cctr@logmaxnum` The largest \TeX number is $16383.99998 \approx 2^{14}$; `\cctr@logmaxnum` is the logarithm of this number, $9.704 \approx \log 16384$.

```

8 \def\cctr@logmaxnum{9.704}

```

12.2 Warning messages

```

9 \def\cctr@Warndivzero#1#2{%
10   \PackageWarning{calculator}%
11     {Division by 0.\MessageBreak
12     I can't define #1/#2}}
13
14 \def\cctr@Warnnogcd{%
15   \PackageWarning{calculator}%
16     {gcd(0,0) is not well defined}}
17
18 \def\cctr@Warnnuposrad#1{%
19   \PackageWarning{calculator}%
20     {The argument in square root\MessageBreak
21     must be non negative\MessageBreak
22     I can't define sqrt(#1)}}
23
24 \def\cctr@Warnnointexp#1#2{%
25   \PackageWarning{calculator}%
26     {The exponent in power function\MessageBreak

```

```

27             must be an integer\MessageBreak
28             I can't define #1^#2}}
29
30 \def\cctr@Warnsingmatrix#1#2#3#4{%
31     \PackageWarning{calculator}{%
32         {Matrix (#1 #2 ; #3 #4) is singular\MessageBreak
33         Its inverse is not defined}}
34
35 \def\cctr@WarnsingTdmatrix#1#2#3#4#5#6#7#8#9{%
36     \PackageWarning{calculator}{%
37         {Matrix (#1 #2 #3; #4 #5 #6; #7 #8 #9) is singular\MessageBreak
38         Its inverse is not defined}}
39
40 \def\cctr@WarnIncLinSys{\PackageWarning{xpicture}{%
41     Incompatible linear system}}
42
43 \def\cctr@WarnIncTDLinSys{\PackageWarning{xpicture}{%
44     Incompatible or indeterminate linear system\MessageBreak
45     For 3x3 systems I can solve only determinate systems}}
46
47 \def\cctr@WarnIndLinSys{\PackageWarning{xpicture}{%
48     Indeterminate linear system.\MessageBreak
49     I will choose one of the infinite solutions}}
50
51 \def\cctr@WarnZeroLinSys{\PackageWarning{xpicture}{%
52     0x=0 linear system. Every vector is a solution!\MessageBreak
53     I will choose the (0,0) solution}}
54
55 \def\cctr@WarninfTan#1{%
56     \PackageWarning{calculator}{%
57         Undefined tangent.\MessageBreak
58         The cosine of #1 is zero and, then,\MessageBreak
59         the tangent of #1 is not defined}}
60
61 \def\cctr@WarninfCotan#1{%
62     \PackageWarning{calculator}{%
63         Undefined cotangent.\MessageBreak
64         The sine of #1 is zero and, then,\MessageBreak
65         the cotangent of #1 is not defined}}
66
67 \def\cctr@WarninfExp#1{%
68     \PackageWarning{calculator}{%
69         The absolute value of the variable\MessageBreak
70         in the exponential function must be less than
71         \cctr@logmaxnum\MessageBreak
72         (the logarithm of the max number I know)\MessageBreak
73         I can't define exp(#1)}}
74
75 \def\cctr@WarninfExpB#1#2{%
76     \PackageWarning{calculator}{%

```

```

77             The base\MessageBreak
78             in the exponential function must be positive.
79             \MessageBreak
80             I can't define #1^{#2}}
81
82 \def\cctr@Warninflog#1{%
83     \PackageWarning{calculator}{%
84         The value of the variable\MessageBreak
85         in the logarithm function must be positive\MessageBreak
86         I can't define log(#1)}}

```

12.3 Operations with numbers

Assignements and comparisons

`\COPY` `\COPY{<#1>}{<#2>}` defines the `#2` command as the number `#1`.

```
87 \def\COPY#1#2{\edef#2{#1}\ignorespaces}
```

`\GLOBALCOPY` Global version of `\COPY`. The new defined command `#2` is not changed outside groups.

```
88 \def\GLOBALCOPY#1#2{\xdef#2{#1}\ignorespaces}
```

`\@OUTPUTSOL` `\@OUTPUTSOL{<#1>}`: an internal macro to save solutions when a group is closed.

The global c.s. `\cctr@outa` preserves solutions. Whenever we use any temporary parameters in the definition of an instruction, we use a group to ensure the local character of those parameters. The instruction `\@OUTPUTSOL` is a bypass to export the solution.

```
89 \def\@OUTPUTSOL#1{\GLOBALCOPY{#1}{\cctr@outa}\endgroup\COPY{\cctr@outa}{#1}}
```

`\@OUTPUTSOLS` Analogous to `\@OUTPUTSOL`, preserving a pair of solutions.

```

90 \def\@OUTPUTSOLS#1#2{\GLOBALCOPY{#1}{\cctr@outa}
91     \GLOBALCOPY{#2}{\cctr@outb}\endgroup
92     \COPY{\cctr@outa}{#1}\COPY{\cctr@outb}{#2}}

```

`\MAX` `\MAX{<#1>}{<#2>}{<#3>}` defines the `#3` command as the maximum of numbers `#1` and `#2`.

```

93 \def\MAX#1#2#3{%
94     \ifdim #1\p@ < #2\p@
95         \COPY{#2}{#3}\else\COPY{#1}{#3}\fi\ignorespaces}

```

`\MIN` `\MIN{<#1>}{<#2>}{<#3>}` defines the `#3` command as the minimum of numbers `#1` and `#2`.

```

96 \def\MIN#1#2#3{%
97     \ifdim #1\p@ > #2\p@
98         \COPY{#2}{#3}\else\COPY{#1}{#3}\fi\ignorespaces}

```

Real arithmetic

`\ABSVALUE` `\ABSVALUE{<#1>}{<#2>}` defines the `#2` command as the absolute value of number `#1`.

```

99 \def\ABSVALUE#1#2{%
100     \ifdim #1\p@<\z@
101         \MULTIPLY{-1}{#1}{#2}\else\COPY{#1}{#2}\fi}

```

Product, sum and difference

\MULTIPLY **\MULTIPLY{<#1>}{<#2>}{<#3>}** defines the *#3* command as the product of numbers *#1* and *#2*.

```
102 \def\MULTIPLY#1#2#3{\cctr@lengtha=#1\p@
103     \cctr@lengtha=#2\cctr@lengtha
104     \edef#3{\expandafter\strip@pt\cctr@lengtha}\ignorespaces}
```

\ADD **\ADD{<#1>}{<#2>}{<#3>}** defines the *#3* command as the sum of numbers *#1* and *#2*.

```
105 \def\ADD#1#2#3{\cctr@lengtha=#1\p@
106     \cctr@lengthb=#2\p@
107     \advance\cctr@lengtha by \cctr@lengthb
108     \edef#3{\expandafter\strip@pt\cctr@lengtha}\ignorespaces}
```

\SUBTRACT **\SUBTRACT{<#1>}{<#2>}{<#3>}** defines the *#3* command as the difference of numbers *#1* and *#2*.

```
109 \def\SUBTRACT#1#2#3{\ADD{#1}{-#2}{#3}}
```

Divisions We define several kinds of *divisions*: the quotient of two real numbers, the integer quotient, and the quotient of two lengths. The basic algorithm is a lightly modified version of the Beccari's division.

\DIVIDE **\DIVIDE{<#1>}{<#2>}{<#3>}** defines the *#3* command as the quotient of numbers *#1* and *#2*.

```
110 \def\DIVIDE#1#2#3{%
111     \begingroup
    Absolute values of dividend and divisor
112     \ABSVALUE{#1}{\cctr@tempD}
113     \ABSVALUE{#2}{\cctr@tempd}
    The sign of quotient
114     \ifdim#1\p@<\z@\ifdim#2\p@>\z@\COPY{-1}{\cctr@sign}
115     \else\COPY{1}{\cctr@sign}\fi
116     \else\ifdim#2\p@>\z@\COPY{1}{\cctr@sign}
117     \else\COPY{-1}{\cctr@sign}\fi
118     \fi
```

Integer part of quotient

```
119     \@DIVIDE{\cctr@tempD}{\cctr@tempd}{\cctr@tempq}{\cctr@tempr}
120     \COPY{\cctr@tempq.}{\cctr@Q}
```

Fractional part up to five decimal places. **\cctr@ndec** is the number of decimal places already computed.

```
121     \COPY{0}{\cctr@ndec}
122     \@whilenum \cctr@ndec<5 \do{%
```

Each decimal place is calculated by multiplying by 10 the last remainder and dividing it by the divisor. But when the remainder is greater than 1638.3, an overflow occurs, because 16383.99998 is the greatest number. So, instead, we multiply the divisor by 0.1.

```
123     \ifdim\cctr@tempr\p@<1638\p@
```

```

124         \MULTIPLY{\cctr@temp{10}}{\cctr@tempD}
125     \else
126         \COPY{\cctr@temp{1}}{\cctr@tempD}
127         \MULTIPLY{\cctr@tempD}{0.1}{\cctr@tempD}
128     \fi
129     \@DIVIDE{\cctr@tempD}{\cctr@tempD}{\cctr@tempq}{\cctr@temp{1}}
130     \COPY{\cctr@tempq}{\cctr@tempq}{\cctr@tempq}
131     \ADD{1}{\cctr@tempq}{\cctr@tempq}%

```

Adjust the sign and return the solution.

```

132     \MULTIPLY{\cctr@sign}{\cctr@tempq}{\cctr@tempq}
133     \@OUTPUTSOL{\cctr@tempq}

```

\@DIVIDE The `\@DIVIDE(<#1>)(<#2>)(<#3>)(<#4>)` command computes $\#1/\#2$ and returns an integer quotient ($\#3$) and a real remainder ($\#4$).

```

134 \def\@DIVIDE#1#2#3#4{%
135     \@INTEGERDIVIDE{#1}{#2}{#3}
136     \MULTIPLY{#2}{#3}{#4}
137     \SUBTRACT{#1}{#4}{#4}}

```

\@INTEGERDIVIDE `\@INTEGERDIVIDE` divides two numbers (not necessarily integer) and returns an integer (this is the integer quotient only for nonnegative integers).

```

138 \def\@INTEGERDIVIDE#1#2#3{%
139     \cctr@lengtha=#1\p@
140     \cctr@lengthb=#2\p@
141     \ifdim\cctr@lengthb=\z@
142         \let#3\undefined
143         \cctr@Warndivzero#1#2%
144     \else
145         \divide\cctr@lengtha\cctr@lengthb
146         \COPY{\number\cctr@lengtha}{#3}
147     \fi\ignorespaces}

```

\LENGTHDIVIDE The quotient of two lengths must be a number (not a length). For example, one inch over one centimeter equals 2.54. `\LENGTHDIVIDE{<#1>}{<#2>}{<#3>}` stores in $\#3$ the quotient of the lengths $\#1$ and $\#2$.

```

148 \def\LENGTHDIVIDE#1#2#3{%
149     \begin{group}
150     \cctr@lengtha=#1
151     \cctr@lengthb=#2
152     \edef\cctr@tempa{\expandafter\strip@pt\cctr@lengtha}%
153     \edef\cctr@tempb{\expandafter\strip@pt\cctr@lengthb}%
154     \DIVIDE{\cctr@tempa}{\cctr@tempb}{#3}
155     \@OUTPUTSOL{\cctr@tempa}

```

Powers

\SQUARE `\SQUARE{<#1>}{<#2>}` stores $\#1$ squared in $\#2$.

```

156 \def\SQUARE#1#2{\MULTIPLY{#1}{#1}{#2}}

```


`\CUBE` `\CUBE{⟨#1⟩}{⟨#2⟩}` stores #1 cubed in #2.
157 `\def\CUBE#1#2{\MULTIPLY{#1}{#1}{#2}\MULTIPLY{#2}{#1}{#2}}`

`\POWER` `\POWER{⟨#1⟩}{⟨#2⟩}{⟨#3⟩}` stores in #3 the power $\#1^{\#2}$

```
158 \def\POWER#1#2#3{%
159     \begingroup
160     \INTEGERPART{#2}{\cctr@tempexp}
161     \ifdim \cctr@tempexp\p@<\#2\p@
162         \cctr@Warnnointexp{#1}{#2}
163         \let#3\undefined
164     \else
```

This ensures that power will be defined only if the exponent is an integer.

```
165         \@POWER{#1}{#2}{#3}\fi\@OUTPUTSOL{#3}}
```

```
166 \def\@POWER#1#2#3{%
167     \begingroup
168     \ifdim #2\p@<\z@
```

For negative exponents, $a^n = (1/a)^{-n}$.

```
169         \DIVIDE{1}{#1}{\cctr@tempb}
170         \MULTIPLY{-1}{#2}{\cctr@tempc}
171         \@POWER{\cctr@tempb}{\cctr@tempc}{#3}
172     \else
173         \COPY{0}{\cctr@tempa}
174         \COPY{1}{#3}
175         \@whilenum \cctr@tempa<\#2 \do {%
176             \MULTIPLY{#1}{#3}{#3}
177             \ADD{1}{\cctr@tempa}{\cctr@tempa}}%
178     \fi\@OUTPUTSOL{#3}}
```

Integer arithmetic and related things

`\INTEGERDIVISION` `\INTEGERDIVISION{⟨#1⟩}{⟨#2⟩}{⟨#3⟩}{⟨#4⟩}` computes the division $\#1/\#2$ and returns an integer quotient and a positive remainder.

```
179 \def\INTEGERDIVISION#1#2#3#4{%
180     \begingroup
181     \ABSVALUE{#2}{\cctr@tempd}
182     \@DIVIDE{#1}{#2}{#3}{#4}
183     \ifdim #4\p@<\z@
184         \ifdim #1\p@<\z@
185             \ifdim #2\p@<\z@
186                 \ADD{#3}{1}{#3}
187             \else
188                 \SUBTRACT{#3}{1}{#3}
189             \fi
190         \ADD{#4}{\cctr@tempd}{#4}
191     \fi\fi\@OUTPUTSOLS{#3}{#4}}
```

`\MODULO` `\MODULO{⟨#1⟩}{⟨#2⟩}{⟨#3⟩}` returns the remainder of division $\#1/\#2$.

```

192 \def\MODULO#1#2#3{%
193     \begingroup
194     \INTEGERDIVISION{#1}{#2}{\cctr@temp}{#3}\@OUTPUTSOL{#3}}

\INTEGERQUOTIENT \INTEGERQUOTIENT{<#1>}{<#2>}{<#3>} returns the integer quotient of division #1/#2.
195 \def\INTEGERQUOTIENT#1#2#3{%
196     \begingroup
197     \INTEGERDIVISION{#1}{#2}{#3}{\cctr@temp}\@OUTPUTSOL{#3}}

\INTEGERPART \INTEGERPART{<#1>}{<#2>} returns the integer part of #2.
198 \def\@INTEGERPART#1.#2.#3)#4{\ifnum #1=1 \COPY{0}{#4}
199     \else \COPY{#1}{#4}\fi}
200 \def\@INTEGERPART#1#2{\expandafter\@INTEGERPART#1..){#2}}
201 \def\INTEGERPART#1#2{\begingroup
202     \ifdim #1\p@<\z@
203         \MULTIPLY{-1}{#1}{\cctr@temp}
204         \INTEGERPART{\cctr@temp}{#2}
205     \ifdim #2\p@<\cctr@temp\p@
206         \SUBTRACT{-#2}{1}{#2}
207     \else \COPY{-#2}{#2}
208     \fi
209     \else
210         \@INTEGERPART{#1}{#2}
211     \fi\@OUTPUTSOL{#2}}

\FLOOR \FLOOR is an alias for \INTEGERPART.
212 \let\FLOOR\INTEGERPART

\FRACTIONALPART \FRACTIONALPART{<#1>}{<#2>} returns the fractional part of #2.
213 \def\@FRACTIONALPART#1.#2.#3)#4{\ifnum #2=11 \COPY{0}{#4}
214     \else \COPY{0.#2}{#4}\fi}
215 \def\@FRACTIONALPART#1#2{\expandafter\@FRACTIONALPART#1..){#2}}
216 \def\FRACTIONALPART#1#2{\begingroup
217     \ifdim #1\p@<\z@
218         \INTEGERPART{#1}{\cctr@tempA}
219         \SUBTRACT{#1}{\cctr@tempA}{#2}
220     \else
221         \@FRACTIONALPART{#1}{#2}
222     \fi\@OUTPUTSOL{#2}}

\TRUNCATE \TRUNCATE[<#1>]{<#2>}{<#3>} truncates #2 to #1 (0, 1, 2 (default), 3 or 4) digits.
223 \def\TRUNCATE{\@ifnextchar[\@TRUNCATE\@TRUNCATE}
224 \def\@TRUNCATE#1#2{\@TRUNCATE[2]{#1}{#2}}
225 \def\@TRUNCATE[#1]#2#3{%
226     \begingroup
227     \INTEGERPART{#2}{\cctr@tempa}
228     \ifdim \cctr@tempa\p@ = #2\p@
229         \expandafter\@@TRUNCATE#2.00000)[#1]{#3}
230     \else
231         \expandafter\@@TRUNCATE#200000.)[#1]{#3}

```

```

232 \fi
233 \@OUTPUTSOL{#3}}
234 \def\@@@TRUNCATE#1.#2#3#4#5#6.#7)[#8]#9{%
235 \ifcase #8
236 \COPY{#1}{#9}
237 \or\COPY{#1.#2}{#9}
238 \or\COPY{#1.#2#3}{#9}
239 \or\COPY{#1.#2#3#4}{#9}
240 \or\COPY{#1.#2#3#4#5}{#9}
241 \fi}

```

\ROUND \ROUND[<#1>]{<#2>}{<#3>} rounds #2 to #1 (0, 1, 2 (default), 3 or 4) digits.

```

242 \def\ROUND{\@ifnextchar[\@@ROUND\@ROUND}
243 \def\@ROUND#1#2{\@@ROUND[2]{#1}{#2}}
244 \def\@@ROUND[#1]#2#3{%
245 \begingroup
246 \ifdim#2\p@<\z@
247 \MULTIPLY{-1}{#2}{\cctr@temp}
248 \@@ROUND[#1]{\cctr@temp}{#3}\COPY{-#3}{#3}
249 \else
250 \@@TRUNCATE[#1]{#2}{\cctr@tempe}
251 \SUBTRACT{#2}{\cctr@tempe}{\cctr@tempe}
252 \POWER{10}{#1}{\cctr@tempb}
253 \MULTIPLY{\cctr@tempb}{\cctr@tempe}{\cctr@tempe}
254 \ifdim\cctr@tempe\p@<0.5\p@
255 \else
256 \DIVIDE{1}{\cctr@tempb}{\cctr@tempb}
257 \ADD{\cctr@tempe}{\cctr@tempb}{\cctr@tempe}
258 \fi
259 \@@TRUNCATE[#1]{\cctr@tempe}{#3}
260 \fi
261 \@OUTPUTSOL{#3}}

```

\GCD \GCD{<#1>}{<#2>}{<#3>} Greatest common divisor, using the Euclidean algorithm

```

262 \def\GCD#1#2#3{%
263 \begingroup
264 \ABSVALUE{#1}{\cctr@tempa}
265 \ABSVALUE{#2}{\cctr@tempb}
266 \MAX{\cctr@tempa}{\cctr@tempb}{\cctr@tempe}
267 \MIN{\cctr@tempa}{\cctr@tempb}{\cctr@tempa}
268 \COPY{\cctr@tempe}{\cctr@tempb}
269 \ifnum \cctr@tempa = 0
270 \ifnum \cctr@tempb = 0
271 \cctr@Warnnogcd
272 \let#3\undefined
273 \else
274 \COPY{\cctr@tempb}{#3}
275 \fi
276 \else

```

Euclidean algorithm: if $c \equiv b \pmod{a}$ then $\gcd(b, a) = \gcd(a, c)$. Iterating this property, we obtain $\gcd(b, a)$ as the last nonzero residual.

```

277      \@whilenum \ctr@tempa > \z@ \do {%
278          \COPY{\ctr@tempa}{#3}%
279          \MODULO{\ctr@tempb}{\ctr@tempa}{\ctr@tempc}%
280          \COPY\ctr@tempa\ctr@tempb%
281          \COPY\ctr@tempc\ctr@tempa}
282      \fi\ignorespaces\@OUTPUTSOL{#3}

```

\LCM $\text{\LCM}\{\langle\#1\rangle\}\{\langle\#2\rangle\}\{\langle\#3\rangle\}$ Least common multiple.

```

283 \def\LCM#1#2#3{%
284     \GCD{#1}{#2}{#3}%
285     \ifx #3\undefined \COPY{0}{#3}
286     \else
287         \DIVIDE{#1}{#3}{#3}
288         \MULTIPLY{#2}{#3}{#3}
289         \ABSVALUE{#3}{#3}
290     \fi}

```

\FRACTIONSIMPLIFY $\text{\FRACTIONSIMPLIFY}\{\langle\#1\rangle\}\{\langle\#2\rangle\}\{\langle\#3\rangle\}\{\langle\#4\rangle\}$ Fraction simplification: $\#3/\#4$ is the irreducible fraction equivalent to $\#1/\#2$.

```

291 \def\FRACTIONSIMPLIFY#1#2#3#4{%
292     \ifnum #1=\z@
293         \COPY{0}{#3}\COPY{1}{#4}
294     \else
295         \GCD{#1}{#2}{#3}%
296         \DIVIDE{#2}{#3}{#4}
297         \DIVIDE{#1}{#3}{#3}
298         \ifnum #4<0 \MULTIPLY{-1}{#4}{#4}\MULTIPLY{-1}{#3}{#3}\fi
299     \fi\ignorespaces}

```

Elementary functions

Square roots

\SQUAREROOT $\text{\SQUAREROOT}\{\langle\#1\rangle\}\{\langle\#2\rangle\}$ defines $\#2$ as the square root of $\#1$, using the Newton's method:

$$x_{n+1} = x_n - (x_n^2 - \#1)/(2x_n).$$

```

300 \def\SQUAREROOT#1#2{%
301     \begingroup
302     \ifdim #1\p@ = \z@
303         \COPY{0}{#2}
304     \else
305         \ifdim #1\p@ < \z@
306             \let#2\undefined
307             \ctr@Warnnuposrad{#1}%
308         \else

```

We take $\#1$ as the initial approximation.

```

309         \COPY{#1}{#2}

```

`\ctr@lengthb` will be the difference of two successive iterations.
 We start with `\ctr@lengthb=5\p@` to ensure almost one iteration.

```

310      \ctr@lengthb=5\p@
Successive iterations
311      \@whilenum \ctr@lengthb>\ctr@epsilon \do {%
Copy the actual approximation to \ctr@tempw
312      \COPY{#2}{\ctr@tempw}
313      \DIVIDE{#1}{\ctr@tempw}{\ctr@tempz}
314      \ADD{\ctr@tempw}{\ctr@tempz}{\ctr@tempz}
315      \DIVIDE{\ctr@tempz}{2}{\ctr@tempz}
Now, \ctr@tempz is the new approximation.
316      \COPY{\ctr@tempz}{#2}
Finally, we store in \ctr@lengthb the difference of the two last approximations, finishing the
loop.
317      \SUBTRACT{#2}{\ctr@tempw}{\ctr@tempw}
318      \ctr@lengthb=\ctr@tempw\p@
319      \ifnum
320      \ctr@lengthb<\z@ \ctr@lengthb=-\ctr@lengthb
321      \fi}
322      \fi\fi\@OUTPUTSOL{#2}}

```

`\SQRT` `\SQRT` is an alias for `\SQUAREROOT`.

```

323 \let\SQRT\SQUAREROOT

```

Trigonometric functions For a variable close enough to zero, the sine and tangent functions are computed using some continued fractions. Then, all trigonometric functions are derived from well-known formulas.

`\SIN` `\SIN{<#1>}{<#2>}`. Sine of $\#1$.

```

324 \def\SIN#1#2{%
325   \begingroup
Exact sine for  $t \in \{\pi/2, -\pi/2, 3\pi/2\}$ 
326   \ifdim #1\p@=-\numberHALFPI\p@ \COPY{-1}{#2}
327   \else
328     \ifdim #1\p@=\numberHALFPI\p@ \COPY{1}{#2}
329     \else
330       \ifdim #1\p@=\numberTHREEHALFPI\p@ \COPY{-1}{#2}
331       \else
If  $|t| > \pi/2$ , change  $t$  to a smaller value.
332       \ifdim#1\p@<-\numberHALFPI\p@
333         \ADD{#1}{\numberTWOPI}{\ctr@tempb}
334         \SIN{\ctr@tempb}{#2}
335       \else
336       \ifdim #1\p@<\numberHALFPI\p@

```

Compute the sine.

```

337          \@BASICSINE{#1}{#2}
338      \else
339          \ifdim #1\p<\numberTHREEHALFPI\p@
340              \SUBTRACT{\numberPI}{#1}{\cctr@tempb}
341              \SIN{\cctr@tempb}{#2}
342          \else
343              \SUBTRACT{#1}{\numberTWOPI}{\cctr@tempb}
344              \SIN{\cctr@tempb}{#2}
345      \fi\fi\fi\fi\fi\fi\@OUTPUTSOL{#2}}

```

\@BASICSINE \@BASICSINE{<#1>}{<#2>} applies this approximation:

$$\sin x = \frac{x}{1 + \frac{x^2}{2 \cdot 3 - x^2 + \frac{2 \cdot 3x^2}{4 \cdot 5 - x^2 + \frac{4 \cdot 5x^2}{6 \cdot 7 - x^2 + \dots}}}}$$

```

346 \def\@BASICSINE#1#2{%
347     \begingroup
348     \ABSVALUE{#1}{\cctr@tempa}
349     \ifdim\cctr@tempa\p@=\z@ \COPY{0}{#2}
350     \else
351         For  $t$  very close to zero,  $\sin t \approx t$ .
352         \ifdim \cctr@tempa\p@<0.009\p@ \COPY{#1}{#2}
353         \else

```

Compute the continued fraction.

```

353         \SQUARE{#1}{\cctr@tempa}
354         \DIVIDE{\cctr@tempa}{42}{#2}
355         \SUBTRACT{1}{#2}{#2}
356         \MULTIPLY{#2}{\cctr@tempa}{#2}
357         \DIVIDE{#2}{20}{#2}
358         \SUBTRACT{1}{#2}{#2}
359         \MULTIPLY{#2}{\cctr@tempa}{#2}
360         \DIVIDE{#2}{6}{#2}
361         \SUBTRACT{1}{#2}{#2}
362         \MULTIPLY{#2}{#1}{#2}
363     \fi\fi\@OUTPUTSOL{#2}}

```

\COS \COS{<#1>}{<#2>}. Cosine of #1: $\cos t = \sin(t + \pi/2)$.

```

364 \def\COS#1#2{%
365     \begingroup
366     \ADD{\numberHALFPI}{#1}{\cctr@tempc}
367     \SIN{\cctr@tempc}{#2}\@OUTPUTSOL{#2}}

```

`\TAN` `\TAN{⟨#1⟩}{⟨#2⟩}`. Tangent of $\#1$.

```

368 \def\TAN#1#2{%
369     \begingroup
    Tangent is infinite for  $t = \pm\pi/2$ 
370     \ifdim #1\p@=-\numberHALFPI\p@
371         \cctr@Warninftan{#1}
372         \let#2\undefined
373     \else
374         \ifdim #1\p@=\numberHALFPI\p@
375             \cctr@Warninftan{#1}
376             \let#2\undefined
377         \else
    If  $|t| > \pi/2$ , change  $t$  to a smaller value.
378         \ifdim #1\p@<-\numberHALFPI\p@
379             \ADD{#1}{\numberPI}{\cctr@tempb}
380             \TAN{\cctr@tempb}{#2}
381         \else
382             \ifdim #1\p@<\numberHALFPI\p@
    Compute the tangent.
383             \@BASICTAN{#1}{#2}
384         \else
385             \SUBTRACT{#1}{\numberPI}{\cctr@tempb}
386             \TAN{\cctr@tempb}{#2}
387         \fi\fi\fi\fi\@OUTPUTSOL{#2}}

```

`\@BASICTAN` `\@BASICTAN{⟨#1⟩}{⟨#2⟩}` applies this approximation:

$$\tan x = \frac{1}{\frac{1}{x} - \frac{3}{x - \frac{5}{x - \frac{7}{x - \frac{9}{x - \frac{11}{x - \dots}}}}}}$$

```

388 \def\@BASICTAN#1#2{%
389     \begingroup
390     \ABSVALUE{#1}{\cctr@tempa}
    Exact tangent of zero.
391     \ifdim\cctr@tempa\p@=\z@ \COPY{0}{#2}
392     \else
    For  $t$  very close to zero,  $\tan t \approx t$ .
393     \ifdim\cctr@tempa\p@<0.04\p@
394         \COPY{#1}{#2}
395     \else

```

Compute the continued fraction.

```

396          \DIVIDE{#1}{11}{#2}
397          \DIVIDE{9}{#1}{\cctr@tempa}
398          \SUBTRACT{\cctr@tempa}{#2}{#2}
399          \DIVIDE{1}{#2}{#2}
400          \DIVIDE{7}{#1}{\cctr@tempa}
401          \SUBTRACT{\cctr@tempa}{#2}{#2}
402          \DIVIDE{1}{#2}{#2}
403          \DIVIDE{5}{#1}{\cctr@tempa}
404          \SUBTRACT{\cctr@tempa}{#2}{#2}
405          \DIVIDE{1}{#2}{#2}
406          \DIVIDE{3}{#1}{\cctr@tempa}
407          \SUBTRACT{\cctr@tempa}{#2}{#2}
408          \DIVIDE{1}{#2}{#2}
409          \DIVIDE{1}{#1}{\cctr@tempa}
410          \SUBTRACT{\cctr@tempa}{#2}{#2}
411          \DIVIDE{1}{#2}{#2}
412          \fi\fi\@OUTPUTSOL{#2}}

```

`\COT` `\COT{<#1>}{<#2>}`. Cotangent of $\#1$: If $\cos t = 0$ then $\cot t = 0$; if $\tan t = 0$ then $\cot t = \infty$. Otherwise, $\cot t = 1/\tan t$.

```

413 \def\COT#1#2{%
414     \begingroup
415     \COS{#1}{#2}
416     \ifdim #2\p@ = \z@
417     \COPY{0}{#2}
418     \else
419     \TAN{#1}{#2}
420     \ifdim #2\p@ = \z@
421     \cctr@Warninfcotan{#1}
422     \let#2\undefined
423     \else
424     \DIVIDE{1}{#2}{#2}
425     \fi\fi\@OUTPUTSOL{#2}}

```

`\DEGtoRAD` `\DEGtoRAD{<#1>}{<#2>}`. Convert degrees to radians.

```

426 \def\DEGtoRAD#1#2{\DIVIDE{#1}{57.29578}{#2}}

```

`\RADtoDEG` `\RADtoDEG{<#1>}{<#2>}`. Convert radians to degrees.

```

427 \def\RADtoDEG#1#2{\MULTIPLY{#1}{57.29578}{#2}}

```

`\REDUCERADIANSANGLE` Reduces to the trigonometrically equivalent arc in $]-\pi, \pi]$.

```

428 \def\REDUCERADIANSANGLE#1#2{%
429     \COPY{#1}{#2}
430     \ifdim #1\p@ < -\numberPI\p@
431         \ADD{#1}{\numberTWOPI}{#2}
432         \REDUCERADIANSANGLE{#2}{#2}
433     \fi
434     \ifdim #1\p@ > \numberPI\p@

```



```

435          \SUBTRACT{#1}{\numberTWOPI}{#2}
436          \REDUCERADIANSANGLE{#2}{#2}
437      \fi
438      \ifdim #1\p@ = -180\p@ \COPY{\numberPI}{#2} \fi}

```

\REDUCEDEGREESANGLE Reduces to the trigonometrically equivalent angle in $]-180, 180]$.

```

439 \def\REDUCEDEGREESANGLE#1#2{%
440     \COPY{#1}{#2}
441     \ifdim #1\p@ < -180\p@
442         \ADD{#1}{360}{#2}
443         \REDUCEDEGREESANGLE{#2}{#2}
444     \fi
445     \ifdim #1\p@ > 180\p@
446         \SUBTRACT{#1}{360}{#2}
447         \REDUCEDEGREESANGLE{#2}{#2}
448     \fi
449     \ifdim #1\p@ = -180\p@ \COPY{180}{#2} \fi}

```

Trigonometric functions in degrees Four next commands compute trigonometric functions in *degrees*. By default, a circle has 360 degrees, but we can use an arbitrary number of divisions using the optional argument of these commands.

\DEGREESSIN **\DEGREESSIN[$\langle\#1\rangle\rangle\{\langle\#2\rangle\}\{\langle\#3\rangle\}$** . Sine of $\#2$ *degrees*.
450 \def\DEGREESSIN{\@ifnextchar[\@@DEGREESSIN\@DEGREESSIN}

\DEGREESCOS **\DEGREESCOS[$\langle\#1\rangle\rangle\{\langle\#2\rangle\}\{\langle\#3\rangle\}$** . Cosine of $\#2$ *degrees*.
451 \def\DEGREESCOS{\@ifnextchar[\@@DEGREESCOS\@DEGREESCOS}

\DEGREESTAN **\DEGREESTAN[$\langle\#1\rangle\rangle\{\langle\#2\rangle\}\{\langle\#3\rangle\}$** . Tangent of $\#2$ *degrees*.
452 \def\DEGREESTAN{\@ifnextchar[\@@DEGREESTAN\@DEGREESTAN}

\DEGREESCOT **\DEGREESCOT[$\langle\#1\rangle\rangle\{\langle\#2\rangle\}\{\langle\#3\rangle\}$** . Cotangent of $\#2$ *degrees*.
453 \def\DEGREESCOT{\@ifnextchar[\@@DEGREESCOT\@DEGREESCOT}

\@DEGREESSIN **\@DEGREESSIN** computes the sine in sexagesimal *degrees*.

```

454 \def\@DEGREESSIN#1#2{%
455     \begingroup
456     \ifdim #1\p@=-90\p@ \COPY{-1}{#2}
457     \else
458         \ifdim #1\p@=90\p@ \COPY{1}{#2}
459         \else
460             \ifdim #1\p@=270\p@ \COPY{-1}{#2}
461         \else
462             \ifdim #1\p@<-90\p@
463                 \ADD{#1}{360}{\cctr@tempb}
464                 \DEGREESSIN{\cctr@tempb}{#2}
465             \else
466                 \ifdim #1\p@<90\p@
467                     \DEGtoRAD{#1}{\cctr@tempb}

```

```

468             \@BASICSINE{\cctr@tempb}{#2}
469         \else
470             \ifdim #1\p@<270\p@
471                 \SUBTRACT{180}{#1}{\cctr@tempb}
472                 \DEGREESSIN{\cctr@tempb}{#2}
473             \else
474                 \SUBTRACT{#1}{360}{\cctr@tempb}
475                 \DEGREESSIN{\cctr@tempb}{#2}
476     \fi\fi\fi\fi\fi\fi\@OUTPUTSOL{#2}}

```

\@DEGREESCOS \@DEGREESCOS computes the cosine in sexagesimal *degrees*.

```

477 \def\@DEGREESCOS#1#2{%
478     \begingroup
479     \ADD{90}{#1}{\cctr@tempc}
480     \DEGREESSIN{\cctr@tempc}{#2}\@OUTPUTSOL{#2}}

```

\@DEGREESTAN \@DEGREESTAN computes the tangent in sexagesimal *degrees*.

```

481 \def\@DEGREESTAN#1#2{%
482     \begingroup
483     \ifdim #1\p@=-90\p@
484         \cctr@Warninftan{#1}
485         \let#2\undefined
486     \else
487         \ifdim #1\p@=90\p@
488             \cctr@Warninftan{#1}
489             \let#2\undefined
490         \else
491             \ifdim #1\p@<-90\p@
492                 \ADD{#1}{180}{\cctr@tempb} \DEGREESTAN{\cctr@tempb}{#2}
493             \else
494                 \ifdim #1\p@<90\p@
495                     \DEGtoRAD{#1}{\cctr@tempb}
496                     \@BASICTAN{\cctr@tempb}{#2}
497                 \else
498                     \SUBTRACT{#1}{180}{\cctr@tempb}
499                     \DEGREESTAN{\cctr@tempb}{#2}
500             \fi\fi\fi\fi\@OUTPUTSOL{#2}}

```

\@DEGREESCOT \@DEGREESCOT computes the cotangent in sexagesimal *degrees*.

```

501 \def\@DEGREESCOT#1#2{%
502     \begingroup
503     \DEGREESCOS{#1}{#2}
504     \ifdim #2\p@ = \z@
505         \COPY{0}{#2}
506     \else
507         \DEGREESTAN{#1}{#2}
508         \ifdim #2\p@ = \z@
509             \cctr@Warninfcotan{#1}
510             \let#2\undefined
511         \else

```

```

512      \DIVIDE{1}{#2}{#2}
513      \fi\fi\@OUTPUTSOL{#2}}

```

For an arbitrary number of *degrees*, we normalise to 360 degrees and, then, call the former functions.

`\@DEGREESSIN` `\@DEGREESSIN` computes the sine. A circle has *#1 degrees*.

```

514 \def\@DEGREESSIN[#1]#2#3{\@CONVERTDEG{#1}{#2}
515      \@DEGREESSIN{\@DEGREES}{#3}}

```

`\@DEGREESCOS` `\@DEGREESCOS` computes the sine. A circle has *#1 degrees*.

```

516 \def\@DEGREESCOS[#1]#2#3{\@CONVERTDEG{#1}{#2}
517      \DEGREESCOS{\@DEGREES}{#3}}

```

`\@DEGREESTAN` `\@DEGREESTAN` computes the sine. A circle has *#1 degrees*.

```

518 \def\@DEGREESTAN[#1]#2#3{\@CONVERTDEG{#1}{#2}
519      \DEGREESTAN{\@DEGREES}{#3}}

```

`\@DEGREESCOT` `\@DEGREESCOT` computes the sine. A circle has *#1 degrees*.

```

520 \def\@DEGREESCOT[#1]#2#3{\@CONVERTDEG{#1}{#2}
521      \DEGREESCOT{\@DEGREES}{#3}}

```

`\@CONVERTDEG` `\@CONVERTDEG` normalises to sexagesimal degrees.

```

522 \def\@CONVERTDEG#1#2{\DIVIDE{#2}{#1}{\@DEGREES}
523      \MULTIPLY{\@DEGREES}{360}{\@DEGREES}}

```

Exponential functions

`\EXP` `\EXP[⟨#1⟩]{⟨#2⟩}{⟨#3⟩}` computes the exponential $\#3 = \#1^{\#2}$. Default for *#1* is number e.

```

524 \def\EXP{\@ifnextchar[\@EXP\@EXP}

```

`\@EXP` `\@EXP[⟨#1⟩]{⟨#2⟩}{⟨#3⟩}` computes $\#3 = \#1^{\#2}$

```

525 \def\@EXP[#1]#2#3{%
526      \begingroup
      #1 must be a positive number.
527      \ifdim #1\p<\cctr@epsilon
528          \cctr@Warninfexpb{#1}{#2}
529          \let#3\undefined
530      \else
       $a^b = \exp(b \log a)$ .
531      \LOG{#1}{\cctr@log}
532      \MULTIPLY{#2}{\cctr@log}{\cctr@log}
533      \@EXP{\cctr@log}{#3}
534      \fi\@OUTPUTSOL{#3}}

```

\@EXP \@EXP{<#1>}{<#2>} computes $\#3 = e^{\#2}$

```

535 \def\@EXP#1#2{%
536     \begingroup
537     \ABSVALUE{#1}{\cctr@absval}

    If  $|t|$  is greater than \cctr@logmaxnum then  $\exp t$  is too large.
538     \ifdim \cctr@absval\p@>\cctr@logmaxnum\p@
539         \cctr@Warninfexp{#1}
540         \let#2\undefined
541     \else
542         \ifdim #1\p@ < \z@

```

We call \@BASICEXP when $t \in [-6, 3]$. Otherwise we use the equality $\exp t = (\exp t/2)^2$.

```

543         \ifdim #1\p@ > -6.00002\p@
544             \@BASICEXP{#1}{#2}
545         \else
546             \DIVIDE{#1}{2}{\cctr@expt}
547             \@EXP{\cctr@expt}{\cctr@expy}
548             \SQUARE{\cctr@expy}{#2}
549         \fi
550     \else
551         \ifdim #1\p@ < 3.00002\p@
552             \@BASICEXP{#1}{#2}
553         \else
554             \DIVIDE{#1}{2}{\cctr@expt}
555             \@EXP{\cctr@expt}{\cctr@expy}
556             \SQUARE{\cctr@expy}{#2}
557         \fi
558 \fi\fi\@OUTPUTSOL{#2}}

```

\@BASICEXP \@BASICEXP{<#1>}{<#2>} applies this approximation:

$$\exp x \approx 1 + \frac{2x}{2 - x + \frac{x^2/6}{1 + \frac{x^2/60}{1 + \frac{x^2/140}{1 + \frac{x^2/256}{1 + \frac{x^2}{396}}}}}}$$

```

559 \def\@BASICEXP#1#2{%
560     \begingroup
561     \SQUARE{#1}{\cctr@tempa}
562     \DIVIDE{\cctr@tempa}{396}{#2}
563     \ADD{1}{#2}{#2}
564     \DIVIDE{\cctr@tempa}{#2}{#2}
565     \DIVIDE{#2}{256}{#2}
566     \ADD{1}{#2}{#2}
567     \DIVIDE{\cctr@tempa}{#2}{#2}
568     \DIVIDE{#2}{140}{#2}

```

```

569      \ADD{1}{#2}{#2}
570      \DIVIDE\cctr@tempa{#2}{#2}
571      \DIVIDE{#2}{60}{#2}
572      \ADD{1}{#2}{#2}
573      \DIVIDE\cctr@tempa{#2}{#2}
574      \DIVIDE{#2}{6}{#2}
575      \ADD{2}{#2}{#2}
576      \SUBTRACT{#2}{#1}{#2}
577      \DIVIDE{#1}{#2}{#2}
578      \MULTIPLY{2}{#2}{#2}
579      \ADD{1}{#2}{#2}\@OUTPUTSOL{#2}}

```

Hyperbolic functions

\COSH **\COSH.** Hyperbolic cosine: $\cosh t = (\exp t + \exp(-t))/2$.

```

580 \def\COSH#1#2{%
581   \begingroup
582   \ABSVALUE{#1}{\cctr@absval}
583   \ifdim \cctr@absval\p@>\cctr@logmaxnum\p@
584     \cctr@Warninfexp{#1}
585     \let#2\undefined
586   \else
587     \EXP{#1}{\cctr@exp}
588     \MULTIPLY{-1}{#1}{\cctr@minust}
589     \EXP{\cctr@minust}{\cctr@expminux}
590     \ADD{\cctr@exp}{\cctr@expminux}{#2}
591     \DIVIDE{#2}{2}{#2}
592   \fi\@OUTPUTSOL{#2}}

```

\SINH **\SINH.** Hyperbolic sine: $\sinh t = (\exp t - \exp(-t))/2$.

```

593 \def\SINH#1#2{%
594   \begingroup
595   \ABSVALUE{#1}{\cctr@absval}
596   \ifdim \cctr@absval\p@>\cctr@logmaxnum\p@
597     \cctr@Warninfexp{#1}
598     \let#2\undefined
599   \else
600     \EXP{#1}{\cctr@exp}
601     \MULTIPLY{-1}{#1}{\cctr@minust}
602     \EXP{\cctr@minust}{\cctr@expminux}
603     \SUBTRACT{\cctr@exp}{\cctr@expminux}{#2}
604     \DIVIDE{#2}{2}{#2}
605   \fi\@OUTPUTSOL{#2}}

```

\TANH **\TANH.** Hyperbolic tangent: $\tanh t = \sinh t / \cosh t$.

```

606 \def\TANH#1#2{%
607   \begingroup
608   \ABSVALUE{#1}{\cctr@absval}
609   \ifdim \cctr@absval\p@>\cctr@logmaxnum\p@
610     \cctr@Warninfexp{#1}

```

```

611         \let#2\undefined
612     \else
613         \SINH{#1}{\cctr@tanhnum}
614         \COSH{#1}{\cctr@tanhden}
615         \DIVIDE{\cctr@tanhnum}{\cctr@tanhden}{#2}
616     \fi\@OUTPUTSOL{#2}}

```

\COTH \COTH. Hyperbolic cotangent $\coth t = \cosh t / \sinh t$.

```

617 \def\COTH#1#2{%
618     \begingroup
619     \ABSVALUE{#1}{\cctr@absval}
620     \ifdim \cctr@absval\p@>\cctr@logmaxnum\p@
621         \cctr@Warninfexp{#1}
622         \let#2\undefined
623     \else
624         \SINH{#1}{\cctr@tanhden}
625         \COSH{#1}{\cctr@tanhnum}
626         \DIVIDE{\cctr@tanhnum}{\cctr@tanhden}{#2}
627     \fi\@OUTPUTSOL{#2}}

```

Logarithm

\LOG \LOG[⟨#1⟩]{⟨#2⟩}{⟨#3⟩} computes the logarithm $\#3 = \log_{\#1} \#2$. Default for #1 is number e.

```

628 \def\LOG{\@ifnextchar[\@LOG\@LOG}

```

\@LOG \@LOG{⟨#1⟩}{⟨#2⟩} computes $\#2 = \log \#1$

```

629 \def\@LOG#1#2{%
630     \begingroup
631     The argument  $t$  must be positive.
632     \ifdim #1\p@<\cctr@epsilon
633         \cctr@Warninflog{#1}
634         \let#2\undefined
635     \else
636         \ifdim #1\p@ > \numberETWO\p@
637             If  $t > e^2$ ,  $\log t = \log e + \log(t/e) = 1 + \log(t/e)$ 
638             \DIVIDE{#1}{\numberE}{\cctr@ae}
639             \@LOG{\cctr@ae}{#2}
640             \ADD{1}{#2}{#2}
641         \else
642             \ifdim #1\p@ < 1\p@
643                 If  $t < 1$ ,  $\log t = \log(1/e) + \log(te) = -1 + \log(te)$ 
644                 \MULTIPLY{\numberE}{#1}{\cctr@ae}
645                 \LOG{\cctr@ae}{#2}
646                 \SUBTRACT{#2}{1}{#2}
647             \else

```

For $t \in [1, e^2]$ we call \@@BASICLOG.

```
645 \@@BASICLOG{#1}{#2}
646 \fi\fi\fi\@OUTPUTSOL{#2}}
```

\@@LOG \@@LOG[⟨#1⟩]{⟨#2⟩}{⟨#3⟩} computes $\#3 = \log_{\#1} \#2 = \log(\#2)/\log(\#1)$

```
647 \def\@@LOG[#1]#2#3{\begingroup
648 \@LOG{#1}{\cctr@loga}
649 \@LOG{#2}{\cctr@logx}
650 \DIVIDE{\cctr@logx}{\cctr@loga}{#3}\@OUTPUTSOL{#3}}
```

\@BASICLOG \@BASICLOG{⟨#1⟩}{⟨#2⟩} applies the Newton's method to calculate $x = \log t$:

$$x_{n+1} = x_n + \frac{t}{e^{x_n}} - 1$$

```
651 \def\@BASICLOG#1#2{\begingroup
652 % We take $\textit{\langle#1\rangle-1}$ as the initial approximation.
653 % \begin{macrocode}
654 \SUBTRACT{#1}{1}{\cctr@tempw}
    We start with \cctr@lengthb=5\p@ to ensure almost one iteration.
655 \cctr@lengthb=5\p@
656 \cctr@lengtha=\cctr@epsilon%
    Successive iterations
657 \@whilenum \cctr@lengthb>\cctr@lengtha \do {%
658 \COPY{\cctr@tempw}{\cctr@tempoldw}
659 \EXP{\cctr@tempw}{\cctr@tempxw}
660 \DIVIDE{#1}{\cctr@tempxw}{\cctr@tempxw}
661 \ADD{\cctr@tempw}{\cctr@tempxw}{\cctr@tempw}
662 \SUBTRACT{\cctr@tempw}{1}{\cctr@tempw}
663 \SUBTRACT{\cctr@tempw}{\cctr@tempoldw}{\cctr@tempdif}
664 \cctr@lengthb=\cctr@tempdif\p@
665 \ifnum
        \cctr@lengthb<\z@ \cctr@lengthb=-\cctr@lengthb
666 \fi}%
667 \COPY{\cctr@tempw}{#2}\@OUTPUTSOL{#2}}
```

12.4 Matrix arithmetics

Vector operations

\VECTORSIZE The *size* of a vector is 2 or 3. \VECTORSIZE(⟨#1⟩){⟨#2⟩} stores in #2 the size of (⟨#1⟩).

Almost all vector commands needs to know the vector size.

```
669 \def\VECTORSIZE(#1)#2{\expandafter\@VECTORSIZE(#1,,){#2}}
670 \def\@VECTORSIZE(#1,#2,#3,#4)#5{\ifx$#3$\COPY{2}{#5}
671 \else\COPY{3}{#5}\fi\ignorespaces}
```

\VECTORCOPY \VECTORCOPY(⟨#1,#2⟩)(⟨#3,#4⟩) stores #1 and #2 in #3 and #4.

\VECTORCOPY(⟨#1,#2,#3⟩)(⟨#4,#5#6⟩) stores #1, #2 and #3 in #4 and #5 and #6.

```
672 \def\@VECTORCOPY(#1,#2)(#3,#4){%
```

```

673     \COPY{#1}{#3}\COPY{#2}{#4}}
674
675 \def\@@@VECTORCOPY(#1,#2,#3)(#4,#5,#6){%
676     \COPY{#1}{#4}\COPY{#2}{#5}\COPY{#3}{#6}}
677
678 \def\VECTORCOPY(#1)(#2){%
679     \VECTORSIZE(#1){\cctr@size}
680     \ifnum\cctr@size=2
681         \@@VECTORCOPY(#1)(#2)
682     \else \@@@VECTORCOPY(#1)(#2)\fi}

```

\VECTORGLOBALCOPY \VECTORGLOBALCOPY is the global version of \VECTORCOPY

```

683 \def\@@@VECTORGLOBALCOPY(#1,#2)(#3,#4){%
684     \GLOBALCOPY{#1}{#3}\GLOBALCOPY{#2}{#4}}
685
686 \def\@@@VECTORGLOBALCOPY(#1,#2,#3)(#4,#5,#6){%
687     \GLOBALCOPY{#1}{#4}\GLOBALCOPY{#2}{#5}\GLOBALCOPY{#3}{#6}}
688
689 \def\VECTORGLOBALCOPY(#1)(#2){%
690     \VECTORSIZE(#1){\cctr@size}
691     \ifnum\cctr@size=2
692         \@@@VECTORGLOBALCOPY(#1)(#2)
693     \else \@@@VECTORGLOBALCOPY(#1)(#2)\fi}

```

\@OUTPUTVECTOR

```

694 \def\@@@OUTPUTVECTOR(#1,#2){%
695     \VECTORGLOBALCOPY(#1,#2)(\cctr@outa,\cctr@outb)
696     \endgroup\VECTORCOPY(\cctr@outa,\cctr@outb)(#1,#2)}
697
698 \def\@@@OUTPUTVECTOR(#1,#2,#3){%
699     \VECTORGLOBALCOPY(#1,#2,#3)(\cctr@outa,\cctr@outb,\cctr@outc)
700     \endgroup\VECTORCOPY(\cctr@outa,\cctr@outb,\cctr@outc)(#1,#2,#3)}
701
702 \def\@@@OUTPUTVECTOR(#1){\VECTORSIZE(#1){\cctr@size}
703     \ifnum\cctr@size=2
704         \@@@OUTPUTVECTOR(#1)
705     \else \@@@OUTPUTVECTOR(#1)\fi}

```

\SCALARPRODUCT Scalar product of two vectors.

```

706 \def\@@@SCALARPRODUCT(#1,#2)(#3,#4)#5{%
707     \MULTIPLY{#1}{#3}{#5}
708     \MULTIPLY{#2}{#4}\cctr@tempa
709     \ADD{#5}{\cctr@tempa}{#5}}
710
711 \def\@@@SCALARPRODUCT(#1,#2,#3)(#4,#5,#6)#7{%
712     \MULTIPLY{#1}{#4}{#7}
713     \MULTIPLY{#2}{#5}\cctr@tempa
714     \ADD{#7}{\cctr@tempa}{#7}
715     \MULTIPLY{#3}{#6}\cctr@tempa
716     \ADD{#7}{\cctr@tempa}{#7}}

```



```

717
718 \def\SCALARPRODUCT(#1)(#2)#3{%
719     \begin{group}
720     \VECTORSIZE(#1){\cctr@size}
721     \ifnum\cctr@size=2
722         \@@SCALARPRODUCT(#1)(#2){#3}
723     \else \@@@SCALARPRODUCT(#1)(#2){#3}\fi\@OUTPUTSOL{#3}}

```

\VECTORADD Sum of two vectors.

```

724 \def\@@VECTORADD(#1,#2)(#3,#4)(#5,#6){%
725     \ADD{#1}{#3}{#5}
726     \ADD{#2}{#4}{#6}}
727
728 \def\@@@VECTORADD(#1,#2,#3)(#4,#5,#6)(#7,#8,#9){%
729     \ADD{#1}{#4}{#7}
730     \ADD{#2}{#5}{#8}
731     \ADD{#3}{#6}{#9}}
732
733 \def\VECTORADD(#1)(#2)(#3){%
734     \VECTORSIZE(#1){\cctr@size}
735     \ifnum\cctr@size=2
736         \@@VECTORADD(#1)(#2)(#3)
737     \else \@@@VECTORADD(#1)(#2)(#3)\fi}

```

\VECTORSUB Difference of two vectors.

```

738 \def\@@VECTORSUB(#1,#2)(#3,#4)(#5,#6){%
739     \VECTORADD(#1,#2)(-#3,-#4)(#5,#6)}
740
741 \def\@@@VECTORSUB(#1,#2,#3)(#4,#5,#6)(#7,#8,#9){%
742     \VECTORADD(#1,#2,#3)(-#4,-#5,-#6)(#7,#8,#9)}
743
744 \def\VECTORSUB(#1)(#2)(#3){%
745     \VECTORSIZE(#1){\cctr@size}
746     \ifnum\cctr@size=2
747         \@@VECTORSUB(#1)(#2)(#3)
748     \else \@@@VECTORSUB(#1)(#2)(#3)\fi}

```

\VECTORABSVALUE Absolute value of a each entry of a vector.

```

749 \def\@@VECTORABSVALUE(#1,#2)(#3,#4){%
750     \ABSVALUE{#1}{#3}\ABSVALUE{#2}{#4}}
751
752 \def\@@@VECTORABSVALUE(#1,#2,#3)(#4,#5,#6){%
753     \ABSVALUE{#1}{#4}\ABSVALUE{#2}{#5}\ABSVALUE{#3}{#6}}
754
755 \def\VECTORABSVALUE(#1)(#2){%
756     \VECTORSIZE(#1){\cctr@size}
757     \ifnum\cctr@size=2
758         \@@VECTORABSVALUE(#1)(#2)
759     \else \@@@VECTORABSVALUE(#1)(#2)\fi}

```

`\SCALARVECTORPRODUCT` Scalar-vector product.

```

760 \def\@@SCALARVECTORPRODUCT#1(#2,#3)(#4,#5){%
761     \MULTIPLY{#1}{#2}{#4}
762     \MULTIPLY{#1}{#3}{#5}}
763
764 \def\@@@SCALARVECTORPRODUCT#1(#2,#3,#4)(#5,#6,#7){%
765     \MULTIPLY{#1}{#2}{#5}
766     \MULTIPLY{#1}{#3}{#6}
767     \MULTIPLY{#1}{#4}{#7}}
768
769 \def\SCALARVECTORPRODUCT#1(#2)(#3){%
770     \VECTORSIZE(#2){\cctr@size}
771     \ifnum\cctr@size=2
772         \@@SCALARVECTORPRODUCT{#1}(#2)(#3)
773     \else \@@@SCALARVECTORPRODUCT{#1}(#2)(#3)\fi}

```

`\VECTORNORM` Euclidean norm of a vector.

```

774 \def\VECTORNORM(#1)#2{%
775     \begingroup
776     \SCALARPRODUCT{#1}(#1){\cctr@tempa}
777     \SQUAREROOT{\cctr@tempa}{#2}\@OUTPUTSOL{#2}}

```

`\UNITVECTOR` Unitary vector parallel to a given vector.

```

778 \def\UNITVECTOR(#1)(#2){%
779     \begingroup
780     \VECTORNORM{#1}{\cctr@tempa}
781     \DIVIDE{1}{\cctr@tempa}{\cctr@tempa}
782     \SCALARVECTORPRODUCT{\cctr@tempa}{#1}(#2)\@OUTPUTVECTOR{#2}}

```

Matrix operations

Here, we need to define some internal macros to simulate commands with more than nine arguments.

`\@TDMATRIXCOPY` This command copies a 3×3 matrix to the commands `\cctr@solAA`, `\cctr@solAB`, ..., `\cctr@solCC`.

```

783 \def\@TDMATRIXCOPY(#1,#2,#3;#4,#5,#6;#7,#8,#9){%
784     \COPY{#1}{\cctr@solAA}
785     \COPY{#2}{\cctr@solAB}
786     \COPY{#3}{\cctr@solAC}
787     \COPY{#4}{\cctr@solBA}
788     \COPY{#5}{\cctr@solBB}
789     \COPY{#6}{\cctr@solBC}
790     \COPY{#7}{\cctr@solCA}
791     \COPY{#8}{\cctr@solCB}
792     \COPY{#9}{\cctr@solCC}}

```

`\@TDMATRIXSOL` This command copies the commands `\cctr@solAA`, `\cctr@solAB`, ..., `\cctr@solCC` to a 3×3 matrix. This macro is used to store the results of a matrix operation.

```

793 \def\@TDMATRIXSOL(#1,#2,#3;#4,#5,#6;#7,#8,#9){%
794     \COPY{\cctr@solAA}{#1}
795     \COPY{\cctr@solAB}{#2}
796     \COPY{\cctr@solAC}{#3}
797     \COPY{\cctr@solBA}{#4}
798     \COPY{\cctr@solBB}{#5}
799     \COPY{\cctr@solBC}{#6}
800     \COPY{\cctr@solCA}{#7}
801     \COPY{\cctr@solCB}{#8}
802     \COPY{\cctr@solCC}{#9}}

```

\@TDMATRIXGLOBALSOL

```

803 \def\@TDMATRIXGLOBALSOL(#1,#2,#3;#4,#5,#6;#7,#8,#9){%
804     \GLOBALCOPY{\cctr@solAA}{#1}
805     \GLOBALCOPY{\cctr@solAB}{#2}
806     \GLOBALCOPY{\cctr@solAC}{#3}
807     \GLOBALCOPY{\cctr@solBA}{#4}
808     \GLOBALCOPY{\cctr@solBB}{#5}
809     \GLOBALCOPY{\cctr@solBC}{#6}
810     \GLOBALCOPY{\cctr@solCA}{#7}
811     \GLOBALCOPY{\cctr@solCB}{#8}
812     \GLOBALCOPY{\cctr@solCC}{#9}}

```

\@TDMATRIXNOSOL This command undefines a 3×3 matrix when a matrix problem has no solution.

```

813 \def\@TDMATRIXNOSOL(#1,#2,#3;#4,#5,#6;#7,#8,#9){%
814     \let#1\undefined
815     \let#2\undefined
816     \let#3\undefined
817     \let#4\undefined
818     \let#5\undefined
819     \let#6\undefined
820     \let#7\undefined
821     \let#8\undefined
822     \let#9\undefined
823 }

```

\@@TDMATRIXSOL This command stores or undefines the solution.

```

824 \def\@@TDMATRIXSOL(#1,#2,#3;#4,#5,#6;#7,#8,#9){%
825     \ifx\cctr@solAA\undefined
826         \@TDMATRIXNOSOL(#1,#2,#3;#4,#5,#6;#7,#8,#9)%
827     \else
828         \@TDMATRIXSOL(#1,#2,#3;#4,#5,#6;#7,#8,#9)\fi}

```

\@NUMBERSOL This command stores the scalar solution of a matrix operation.

```

829 \def\@NUMBERSOL#1{\COPY{\cctr@sol}{#1}}

```

\MATRIXSIZE Size (2 or 3) of a matrix.

```

830 \def\MATRIXSIZE(#1)#2{\expandafter\@MATRIXSIZE(#1;){#2}}
831 \def\@MATRIXSIZE(#1,#2;#3;#4)#5{\ifx$#3$\COPY{2}{#5}
832     \else\COPY{3}{#5}\fi\ignorespaces}

```

\MATRIXCOPY Store a matrix in 4 or 9 commands.

```

833 \def\@MATRIXCOPY(#1,#2;#3,#4)(#5,#6;#7,#8){%
834   \COPY{#1}-{#5}\COPY{#2}-{#6}\COPY{#3}-{#7}\COPY{#4}-{#8}}
835
836 \def\@MATRIXCOPY(#1,#2,#3;#4,#5,#6;#7,#8,#9){%
837   \@TDMATRIXCOPY(#1,#2,#3;#4,#5,#6;#7,#8,#9)
838   \@TDMATRIXSOL}
839
840 \def\MATRIXCOPY(#1)(#2){%
841   \MATRIXSIZE(#1){\ctr@size}
842   \ifnum\ctr@size=2
843     \@MATRIXCOPY(#1)(#2)
844   \else \@MATRIXCOPY(#1)(#2)\fi}

```

\MATRIXGLOBALCOPY Global version of \MATRIXCOPY.

```

845 \def\@MATRIXGLOBALCOPY(#1,#2;#3,#4)(#5,#6;#7,#8){%
846   \GLOBALCOPY{#1}-{#5}\GLOBALCOPY{#2}-{#6}\GLOBALCOPY{#3}-{#7}\GLOBALCOPY{#4}-{#8}}
847
848 \def\@MATRIXGLOBALCOPY(#1,#2,#3;#4,#5,#6;#7,#8,#9){%
849   \@TDMATRIXCOPY(#1,#2,#3;#4,#5,#6;#7,#8,#9)
850   \@TDMATRIXGLOBALSOL}
851
852 \def\MATRIXGLOBALCOPY(#1)(#2){%
853   \MATRIXSIZE(#1){\ctr@size}
854   \ifnum\ctr@size=2
855     \@MATRIXGLOBALCOPY(#1)(#2)
856   \else \@MATRIXGLOBALCOPY(#1)(#2)\fi}

```

\@OUTPUTMATRIX

```

857 \def\@OUTPUTMATRIX(#1,#2;#3,#4){%
858   \MATRIXGLOBALCOPY(#1,#2;#3,#4)(\ctr@outa,\ctr@outb;\ctr@outc,\ctr@outd)
859   \endgroup\MATRIXCOPY(\ctr@outa,\ctr@outb;\ctr@outc,\ctr@outd)(#1,#2;#3,#4)}
860
861 \def\@OUTPUTMATRIX(#1,#2,#3;#4,#5,#6;#7,#8,#9){%
862   \MATRIXGLOBALCOPY(#1,#2,#3;#4,#5,#6;#7,#8,#9)(%
863     \ctr@outa,\ctr@outb,\ctr@outc;
864     \ctr@outd,\ctr@oute,\ctr@outf;
865     \ctr@outg,\ctr@outh,\ctr@outi)
866   \endgroup\MATRIXCOPY(%
867     \ctr@outa,\ctr@outb,\ctr@outc;
868     \ctr@outd,\ctr@oute,\ctr@outf;
869     \ctr@outg,\ctr@outh,\ctr@outi)(#1,#2,#3;#4,#5,#6;#7,#8,#9)}
870
871 \def\@OUTPUTMATRIX(#1){\MATRIXSIZE(#1){\ctr@size}
872   \ifnum\ctr@size=2
873     \@OUTPUTMATRIX(#1)
874   \else \@OUTPUTMATRIX(#1)\fi}

```

\TRANPOSEMATRIX Matrix transposition.

```

875 \def\@TRANPOSEMATRIX(#1,#2;#3,#4)(#5,#6;#7,#8){%

```

```

876 \COPY{#1}{#5}\COPY{#3}{#6}\COPY{#2}{#7}\COPY{#4}{#8}}
877
878 \def\@@@TRANPOSEMATRIX(#1,#2,#3;#4,#5,#6;#7,#8,#9){%
879 \@TDMATRIXCOPY(#1,#4,#7;#2,#5,#8;#3,#6,#9)
880 \@TDMATRIXSOL}
881
882 \def\TRANPOSEMATRIX(#1)(#2){%
883 \begin{group}
884 \MATRIXSIZE(#1){\cctr@size}
885 \ifnum\cctr@size=2
886 \@@TRANPOSEMATRIX(#1)(#2)
887 \else \@@@TRANPOSEMATRIX(#1)(#2)\fi\@OUTPUTMATRIX(#2)}

```

\MATRIXADD Sum of two matrices.

```

888 \def\@@MATRIXADD(#1;#2)(#3;#4)(#5,#6;#7,#8){%
889 \VECTORADD(#1)(#3)(#5,#6)
890 \VECTORADD(#2)(#4)(#7,#8)}
891
892 \def\@@@MATRIXADD(#1;#2;#3)(#4;#5;#6){%
893 \VECTORADD(#1)(#4)(\cctr@solAA,\cctr@solAB,\cctr@solAC)
894 \VECTORADD(#2)(#5)(\cctr@solBA,\cctr@solBB,\cctr@solBC)
895 \VECTORADD(#3)(#6)(\cctr@solCA,\cctr@solCB,\cctr@solCC)
896 \@TDMATRIXSOL}
897
898 \def\MATRIXADD(#1)(#2)(#3){%
899 \begin{group}
900 \MATRIXSIZE(#1){\cctr@size}
901 \ifnum\cctr@size=2
902 \@@MATRIXADD(#1)(#2)(#3)
903 \else \@@@MATRIXADD(#1)(#2)(#3)\fi\@OUTPUTMATRIX(#3)}

```

\MATRIXSUB Difference of two matrices.

```

904 \def\@@MATRIXSUB(#1;#2)(#3;#4)(#5,#6;#7,#8){%
905 \VECTORSUB(#1)(#3)(#5,#6)
906 \VECTORSUB(#2)(#4)(#7,#8)}
907
908 \def\@@@MATRIXSUB(#1;#2;#3)(#4;#5;#6){%
909 \VECTORSUB(#1)(#4)(\cctr@solAA,\cctr@solAB,\cctr@solAC)
910 \VECTORSUB(#2)(#5)(\cctr@solBA,\cctr@solBB,\cctr@solBC)
911 \VECTORSUB(#3)(#6)(\cctr@solCA,\cctr@solCB,\cctr@solCC)
912 \@TDMATRIXSOL}
913
914 \def\MATRIXSUB(#1)(#2)(#3){%
915 \begin{group}
916 \MATRIXSIZE(#1){\cctr@size}
917 \ifnum\cctr@size=2
918 \@@MATRIXSUB(#1)(#2)(#3)
919 \else \@@@MATRIXSUB(#1)(#2)(#3)\fi\@OUTPUTMATRIX(#3)}

```

\MATRIXABSVALUE Absolute value (of each entry) of a matrix.

```

920 \def\@@MATRIXABSVLUE(#1;#2)(#3;#4){%
921     \VECTORABSVLUE(#1)(#3)\VECTORABSVLUE(#2)(#4)}
922
923 \def\@@@MATRIXABSVLUE(#1;#2;#3)(#4;#5;#6){%
924     \VECTORABSVLUE(#1)(#4)\VECTORABSVLUE(#2)(#5)\VECTORABSVLUE(#3)(#6)}
925
926 \def\MATRIXABSVLUE(#1)(#2){%
927     \begin{group}
928     \MATRIXSIZE(#1){\cctr@size}
929     \ifnum\cctr@size=2
930         \@@MATRIXABSVLUE(#1)(#2)
931     \else \@@@MATRIXABSVLUE(#1)(#2)\fi\@OUTPUTMATRIX(#2)}

```

\MATRIXVECTORPRODUCT Matrix-vector product.

```

932 \def\@@MATRIXVECTORPRODUCT(#1;#2)(#3)(#4,#5){%
933     \SCALARPRODUCT(#1)(#3){#4}
934     \SCALARPRODUCT(#2)(#3){#5}}
935
936 \def\@@@MATRIXVECTORPRODUCT(#1;#2;#3)(#4)(#5,#6,#7){%
937     \SCALARPRODUCT(#1)(#4){#5}
938     \SCALARPRODUCT(#2)(#4){#6}
939     \SCALARPRODUCT(#3)(#4){#7}}
940
941 \def\MATRIXVECTORPRODUCT(#1)(#2)(#3){%
942     \begin{group}
943     \MATRIXSIZE(#1){\cctr@size}
944     \ifnum\cctr@size=2
945         \@@MATRIXVECTORPRODUCT(#1)(#2)(#3)
946     \else \@@@MATRIXVECTORPRODUCT(#1)(#2)(#3)\fi\@OUTPUTVECTOR(#3)}

```

\VECTORMATRIXPRODUCT Vector-matrix product.

```

947 \def\@@VECTORMATRIXPRODUCT(#1)(#2,#3;#4,#5)(#6,#7){%
948     \SCALARPRODUCT(#1)(#2,#4){#6}
949     \SCALARPRODUCT(#1)(#3,#5){#7}}
950
951 \def\@@@VECTORMATRIXPRODUCT(#1,#2,#3)(#4;#5;#6)(#7){%
952     \SCALARVECTORPRODUCT{#1}{#4)(#7)
953     \SCALARVECTORPRODUCT{#2}{#5)(\cctr@tempa,\cctr@tempb,\cctr@tempc)
954     \VECTORADD(#7)(\cctr@tempa,\cctr@tempb,\cctr@tempc)(#7)
955     \SCALARVECTORPRODUCT{#3}{#6)(\cctr@tempa,\cctr@tempb,\cctr@tempc)
956     \VECTORADD(#7)(\cctr@tempa,\cctr@tempb,\cctr@tempc)(#7)}
957
958 \def\VECTORMATRIXPRODUCT(#1)(#2)(#3){%
959     \begin{group}
960     \VECTORSIZE(#1){\cctr@size}
961     \ifnum\cctr@size=2
962         \@@VECTORMATRIXPRODUCT(#1)(#2)(#3)
963     \else \@@@VECTORMATRIXPRODUCT(#1)(#2)(#3)\fi\@OUTPUTVECTOR(#3)}

```

\SCALARMATRIXPRODUCT Scalar-matrix product.

```

964 \def\@@SCALARMATRIXPRODUCT#1(#2;#3)(#4,#5;#6,#7){%
965     \SCALARVECTORPRODUCT{#1}(#2)(#4,#5)
966     \SCALARVECTORPRODUCT{#1}(#3)(#6,#7)}
967
968 \def\@@@SCALARMATRIXPRODUCT#1(#2;#3;#4){%
969     \SCALARVECTORPRODUCT{#1}(#2)(\cctr@solAA,\cctr@solAB,\cctr@solAC)
970     \SCALARVECTORPRODUCT{#1}(#3)(\cctr@solBA,\cctr@solBB,\cctr@solBC)
971     \SCALARVECTORPRODUCT{#1}(#4)(\cctr@solCA,\cctr@solCB,\cctr@solCC)
972     \@TDMATRIXSOL}
973
974 \def\SCALARMATRIXPRODUCT#1(#2)(#3){%
975     \begin{group}
976     \MATRIXSIZE{#2}{\cctr@size}
977     \ifnum\cctr@size=2
978         \@@SCALARMATRIXPRODUCT{#1}(#2)(#3)
979     \else \@@@SCALARMATRIXPRODUCT{#1}(#2)(#3)\fi\@OUTPUTMATRIX{#3}}

```

\MATRIXPRODUCT Product of two matrices.

```

980 \def\@@MATRIXPRODUCT(#1)(#2,#3;#4,#5)(#6,#7;#8,#9){%
981     \MATRIXVECTORPRODUCT{#1}(#2,#4)(#6,#8)
982     \MATRIXVECTORPRODUCT{#1}(#3,#5)(#7,#9)}
983
984 \def\@@@MATRIXPRODUCT(#1;#2;#3)(#4){%
985     \VECTORMATRIXPRODUCT{#1}(#4)(\cctr@solAA,\cctr@solAB,\cctr@solAC)
986     \VECTORMATRIXPRODUCT{#2}(#4)(\cctr@solBA,\cctr@solBB,\cctr@solBC)
987     \VECTORMATRIXPRODUCT{#3}(#4)(\cctr@solCA,\cctr@solCB,\cctr@solCC)
988     \@TDMATRIXSOL}
989
990 \def\MATRIXPRODUCT(#1)(#2)(#3){%
991     \begin{group}
992     \MATRIXSIZE{#1}{\cctr@size}
993     \ifnum\cctr@size=2
994         \@@MATRIXPRODUCT{#1}(#2)(#3)
995     \else \@@@MATRIXPRODUCT{#1}(#2)(#3)\fi\@OUTPUTMATRIX{#3}}

```

\DETERMINANT Determinant of a matrix.

```

996 \def\@@DETERMINANT(#1,#2;#3,#4)#5{%
997     \MULTIPLY{#1}{#4}{#5}
998     \MULTIPLY{#2}{#3}{\cctr@tempa}
999     \SUBTRACT{#5}{\cctr@tempa}{#5}}
1000
1001 \def\@@@DETERMINANT(#1,#2,#3;#4,#5,#6;#7,#8,#9){%
1002     \DETERMINANT{#5,#6;#8,#9}{\cctr@det}\MULTIPLY{#1}{\cctr@det}{\cctr@sol}
1003     \DETERMINANT{#6,#4;#9,#7}{\cctr@det}\MULTIPLY{#2}{\cctr@det}{\cctr@det}
1004     \ADD{\cctr@sol}{\cctr@det}{\cctr@sol}
1005     \DETERMINANT{#4,#5;#7,#8}{\cctr@det}\MULTIPLY{#3}{\cctr@det}{\cctr@det}
1006     \ADD{\cctr@sol}{\cctr@det}{\cctr@sol}
1007     \@NUMBERSOL}
1008
1009 \def\DETERMINANT(#1)#2{%

```

```

1010 \beginngroup
1011 \MATRIXSIZE(#1){\cctr@size}
1012 \ifnum\cctr@size=2
1013 \@@DETERMINANT(#1){#2}
1014 \else \@@DETERMINANT(#1){#2}\fi\@OUTPUTSOL{#2}}

```

\INVERSEMATRIX Inverse of a matrix.

```

1015 \def\@@INVERSEMATRIX(#1,#2;#3,#4)(#5,#6;#7,#8){%
1016 \ifdim \cctr@det\p@ <\cctr@epsilon % Matrix is singular
1017 \let#5\undefined
1018 \let#6\undefined
1019 \let#7\undefined
1020 \let#8\undefined
1021 \cctr@Warnsingmatrix{#1}{#2}{#3}{#4}%
1022 \else \COPY{#1}{#8}
1023 \COPY{#4}{#5}
1024 \MULTIPLY{-1}{#3}{#7}
1025 \MULTIPLY{-1}{#2}{#6}
1026 \DIVIDE{1}{\cctr@det}{\cctr@det}
1027 \SCALARMATRIXPRODUCT{\cctr@det}{#5,#6;#7,#8}(#5,#6;#7,#8)
1028 \fi}
1029
1030 \def\@@@INVERSEMATRIX(#1,#2,#3;#4,#5,#6;#7,#8,#9){%
1031 \ifdim \cctr@det\p@ <\cctr@epsilon % Matrix is singular
1032 \@TDMATRIXNOSOL(\cctr@solAA,\cctr@solAB,\cctr@solAC;
1033 \cctr@solBA,\cctr@solBB,\cctr@solBC;
1034 \cctr@solCA,\cctr@solCB,\cctr@solCC)
1035 \cctr@WarnsingTdmatrix{#1}{#2}{#3}{#4}{#5}{#6}{#7}{#8}{#9}%
1036 \else
1037 \@ADJMATRIX(#1,#2,#3;#4,#5,#6;#7,#8,#9)
1038 \@SCLRDIVECT{\cctr@det}{\cctr@solAA,\cctr@solAB,\cctr@solAC}(%
1039 \cctr@solAA,\cctr@solAB,\cctr@solAC)
1040 \@SCLRDIVECT{\cctr@det}{\cctr@solBA,\cctr@solBB,\cctr@solBC}(%
1041 \cctr@solBA,\cctr@solBB,\cctr@solBC)
1042 \@SCLRDIVECT{\cctr@det}{\cctr@solCA,\cctr@solCB,\cctr@solCC}(%
1043 \cctr@solCA,\cctr@solCB,\cctr@solCC)
1044 \fi
1045 \@@TDMATRIXSOL}
1046
1047 \def\@SCLRDIVECT#1(#2,#3,#4)(#5,#6,#7){%
1048 \DIVIDE{#2}{#1}{#5}\DIVIDE{#3}{#1}{#6}\DIVIDE{#4}{#1}{#7}}
1049
1050 \def\@ADJMATRIX(#1,#2,#3;#4,#5,#6;#7,#8,#9){%
1051 \DETERMINANT(#5,#6;#8,#9){\cctr@solAA}
1052 \DETERMINANT(#6,#4;#9,#7){\cctr@solBA}
1053 \DETERMINANT(#4,#5;#7,#8){\cctr@solCA}
1054 \DETERMINANT(#8,#9;#2,#3){\cctr@solAB}
1055 \DETERMINANT(#1,#3;#7,#9){\cctr@solBB}
1056 \DETERMINANT(#2,#1;#8,#7){\cctr@solCB}
1057 \DETERMINANT(#2,#3;#5,#6){\cctr@solAC}

```



```

1058      \DETERMINANT(#3,#1,#6,#4){\ctr@solBC}
1059      \DETERMINANT(#1,#2,#4,#5){\ctr@solCC}}
1060
1061 \def\INVERSEMATRIX(#1)(#2){%
1062     \begin{group}
1063     \DETERMINANT(#1){\ctr@det}
1064     \ABSVALUE{\ctr@det}{\ctr@@det}
1065     \MATRIXSIZE(#1){\ctr@size}
1066     \ifnum\ctr@size=2
1067         \@@INVERSEMATRIX(#1)(#2)
1068     \else
1069         \@@@INVERSEMATRIX(#1)(#2)\fi\@OUTPUTMATRIX(#2)}

```

\SOLVELINEARSYSTEM Solving a linear system (two equations and two unknowns or three equations and three unknowns).

```

1070 \def\@INCSYS#1#2{\ctr@WarnIncLinSys
1071     \let#1\undefined\let#2\undefined}
1072
1073 \def\@SOLPART#1#2#3#4{\ctr@WarnIndLinSys
1074     \DIVIDE{#1}{#2}{#3}
1075     \COPY{0}{#4}}
1076
1077 \def\@TDINCSYS(#1,#2,#3){\ctr@WarnIncTDLinSys
1078     \let#1\undefined
1079     \let#2\undefined
1080     \let#3\undefined}
1081
1082 \def\@SOLVELINEARSYSTEM(#1,#2,#3,#4)(#5,#6)(#7,#8){%
1083     \DETERMINANT(#1,#2,#3,#4)\ctr@deta
1084     \DETERMINANT(#5,#2,#6,#4)\ctr@detb
1085     \DETERMINANT(#1,#5,#3,#6)\ctr@detc
1086     \ABSVALUE{\ctr@deta}{\ctr@@deta}
1087     \ABSVALUE{\ctr@detb}{\ctr@@detb}
1088     \ABSVALUE{\ctr@detc}{\ctr@@detc}
1089     \ifdim \ctr@@deta\p@>\ctr@epsilon% Regular matrix. Determinate system
1090         \DIVIDE{\ctr@detb}{\ctr@deta}{#7}
1091         \DIVIDE{\ctr@detc}{\ctr@deta}{#8}
1092     \else % Singular matrix \ctr@deta=0
1093         \ifdim \ctr@@detb\p@>\ctr@epsilon% Incompatible system
1094             \@INCSYS#7#8
1095         \else
1096             \ifdim \ctr@@detc\p@>\ctr@epsilon% Incompatible system
1097                 \@INCSYS#7#8
1098             \else
1099                 \MATRIXABSVALUE(#1,#2,#3,#4)(\ctr@tempa,\ctr@tempb;
1100                     \ctr@tempc,\ctr@tempd)
1101                 \ifdim \ctr@tempa\p@>\ctr@epsilon
1102                     % Indeterminate system
1103                     \@SOLPART{#5}{#1}{#7}{#8}
1104                 \else

```

```

1105         \ifdim \cctr@tempb\p@ > \cctr@epsilon
1106             % Indeterminate system
1107             \@SOLPART{#5}{#2}{#8}{#7}
1108         \else
1109             \ifdim \cctr@tempc\p@ > \cctr@epsilon
1110                 % Indeterminate system
1111                 \@SOLPART{#6}{#3}{#7}{#8}
1112             \else
1113                 \ifdim \cctr@tempd\p@ > \cctr@epsilon
1114                     % Indeterminate system
1115                     \@SOLPART{#6}{#4}{#8}{#7}
1116                 \else
1117                     \VECTORNORM(#5,#6){\cctr@tempa}
1118                     \ifdim \cctr@tempa\p@ > \cctr@epsilon
1119                         % Incompatible system
1120                         \@INCSYS#7#8
1121                     \else
1122                         \cctr@WarnZeroLinSys
1123                         \COPY{0}{#7}\COPY{0}{#8}
1124                         % 0x=0 Indeterminate system
1125                     \fi\fi\fi\fi\fi\fi\fi\fi}
1126
1127 \def\@@@SOLVELINEARSYSTEM(#1)(#2)(#3){%
1128     \DETERMINANT(#1){\cctr@det}
1129     \ABSVALUE{\cctr@det}{\cctr@@det}
1130     \ifdim\cctr@@det\p@<\cctr@epsilon
1131         \@TDINCSYS(#3)
1132     \else
1133         \@ADJMATRIX(#1)
1134         \MATRIXVECTORPRODUCT(\cctr@solAA,\cctr@solAB,\cctr@solAC;
1135                               \cctr@solBA,\cctr@solBB,\cctr@solBC;
1136                               \cctr@solCA,\cctr@solCB,\cctr@solCC)(#2)(#3)
1137         \@SCLRDIVVECT{\cctr@det}{#3}(#3)
1138     \fi}
1139
1140 \def\SOLVELINEARSYSTEM(#1)(#2)(#3){%
1141     \begin{group}
1142     \MATRIXSIZE(#1){\cctr@size}
1143     \ifnum\cctr@size=2
1144         \@@@SOLVELINEARSYSTEM(#1)(#2)(#3)
1145     \else
1146         \@@@SOLVELINEARSYSTEM(#1)(#2)(#3)
1147     \fi\@OUTPUTVECTOR(#3)}

```

Predefined numbers

\numberPI The number π

```
1148 \def\numberPI{3.14159}
```

\numberTWOPI 2π

$\backslash\text{numberPI}\{2\}\{\backslash\text{numberTWOPI}\}$
 $\backslash\text{numberHALFPI} \quad \pi/2$
 $\backslash\text{numberTHREEHALFPI} \quad 3\pi/2$
 $\backslash\text{numberTHIRDPI} \quad \pi/3$
 $\backslash\text{numberQUARTERPI} \quad \pi/4$
 $\backslash\text{numberFIFTHPI} \quad \pi/5$
 $\backslash\text{numberSIXTHPI} \quad \pi/6$
 $\backslash\text{numberE} \quad \text{The number } e$
 $\backslash\text{numberINVE} \quad 1/e$
 $\backslash\text{numberETWO} \quad e^2$
 $\backslash\text{numberINVETWO} \quad 1/e^2$
 $\backslash\text{numberLOGTEN} \quad \log 10$
 $\backslash\text{numberGOLD} \quad \text{The golden ratio } \phi$
 $\backslash\text{numberINVGOLD} \quad 1/\phi$
 $\backslash\text{numberSQRTTWO} \quad \sqrt{2}$
 $\backslash\text{numberSQRTHREE} \quad \sqrt{3}$
 $\backslash\text{numberSQRTFIVE} \quad \sqrt{5}$

```

\numberCOSXLV  cos 45° (or cos  $\pi/4$ )
1166 \def\numberCOSXLV{0.70711}

\numberCOSXXX  cos 30° (or cos  $\pi/6$ )
1167 \def\numberCOSXXX{0.86603}

1168 </calculator>

```

13 Implementation (calculator)

```

1169 <*calculus>
1170 \NeedsTeXFormat{LaTeX2e}
1171 \ProvidesPackage{calculus}
1172 [2012/06/10 v.1.0a]

```

This package requires the calculator package.

```
1173 \RequirePackage{calculator}
```

13.1 Error and info messages

For scalar functions

Error message to be issued when you attempt to define, with `\newfunction`, an already defined command:

```

1174 \def\ccls@ErrorFuncDef#1{%
1175     \PackageError{calculus}%
1176     {\noexpand#1 command already defined}
1177     {The \noexpand#1 control sequence is already defined\MessageBreak
1178     If you want to redefine the \noexpand#1 command as a
1179     function\MessageBreak
1180     please, use the \noexpand\renewfunction command}}

```

Error message to be issued when you attempt to redefine, with `\renewfunction`, an undefined command:

```

1181 \def\ccls@ErrorFuncUnDef#1{%
1182     \PackageError{calculus}%
1183     {\noexpand#1 command undefined}
1184     {The \noexpand#1 control sequence is not currently defined\MessageBreak
1185     If you want to define the \noexpand#1 command as a function\MessageBreak
1186     please, use the \noexpand\newfunction command}}

```

Info message to be issued when `\ensurefunction` does not changes an already defined command:

```

1187 \def\ccls@InfoFuncEns#1{%
1188     \PackageInfo{calculus}%
1189     {\noexpand#1 command already defined\MessageBreak
1190     the \noexpand\ensurefunction command will not redefine it}}

```

For polar functions

```

1191 \def\ccls@ErrorPFuncDef#1{%
1192     \PackageError{calculus}%

```

```

1193      {\noexpand#1 command already defined}
1194      {The \noexpand#1 control sequence is already defined\MessageBreak
1195       If you want to redefine the \noexpand#1
1196       command as a polar function\MessageBreak
1197       please, use the \noexpand\renewpolarfunction command}}
1198
1199 \def\ccls@ErrorPFuncUnDef#1{%
1200   \PackageError{calculus}%
1201     {\noexpand#1 command undefined}
1202     {The \noexpand#1 control sequence
1203      is not currently defined.\MessageBreak
1204      If you want to define the \noexpand#1 command as a polar
1205      function\MessageBreak
1206      please, use the \noexpand\newpolarfunction command}}
1207
1208 \def\ccls@InfoPFuncEns#1{%
1209   \PackageInfo{calculus}%
1210   {\noexpand#1 command already defined\MessageBreak
1211    the \noexpand\ensurepolarfunction command does not redefine it}}

```

For vector functions

```

1212 \def\ccls@ErrorVFuncDef#1{%
1213   \PackageError{calculus}%
1214     {\noexpand#1 command already defined}
1215     {The \noexpand#1 control sequence is already defined\MessageBreak
1216      If you want to redefine the \noexpand#1 command as a vector
1217      function\MessageBreak
1218      please, use the \noexpand\renewvectorfunction command}}
1219
1220 \def\ccls@ErrorVFuncUnDef#1{%
1221   \PackageError{calculus}%
1222     {\noexpand#1 command undefined}
1223     {The \noexpand#1 control sequence is not currently
1224      defined.\MessageBreak
1225      If you want to define the \noexpand#1 command as a vector
1226      function\MessageBreak
1227      please, use the \noexpand\newvectorfunction command}}
1228
1229 \def\ccls@InfoVFuncEns#1{%
1230   \PackageInfo{calculus}%
1231   {\noexpand#1 command already defined\MessageBreak
1232    the \noexpand\ensurevectorfunction command does not redefine it}}

```

13.2 New functions

New scalar functions

`\newfunction` The `\newfunction{#1}{#2}` instruction defines a new function called #1. #2 is the list of instructions to calculate the function y and his derivative Dy from the t variable.

```

1233 \def\newfunction#1#2{%
1234   \ifx #1\undefined

```

```

1235         \ccls@deffunction{#1}{#2}
1236     \else
1237         \ccls@ErrorFuncDef{#1}
1238     \fi}

```

\renewfunction **\renewfunction** redefines #1, as a new function, if this command is already defined.

```

1239 \def\renewfunction#1#2{%
1240     \ifx #1\undefined
1241         \ccls@ErrorFuncUnDef{#1}
1242     \else
1243         \ccls@deffunction{#1}{#2}
1244     \fi}

```

\ensurefunction **\ensurefunction** defines the new function #1 (only if this macro is undefined).

```

1245 \def\ensurefunction#1#2{%
1246     \ifx #1\undefined\ccls@deffunction{#1}{#2}
1247     \else
1248         \ccls@InfoFuncEns{#1}
1249     \fi}

```

\forcefunction **\forcefunction** defines (if undefined) or redefines (if defined) the new function #1.

```

1250 \def\forcefunction#1#2{%
1251     \ccls@deffunction{#1}{#2}}

```

\ccls@deffunction The private **\ccls@deffunction** command makes the real work. The new functions will have three arguments: ##1, a number, ##2, the value of the new function in that number, and ##3, the derivative.

```

1252 \def\ccls@deffunction#1#2{%
1253     \def#1##1##2##3{%
1254         \begingroup
1255         \def\t{##1}%
1256         #2
1257         \xdef##2{\y}%
1258         \xdef##3{\Dy}%
1259     \endgroup}\ignorespaces}

```

New polar functions

\newpolarfunction The **\newpolarfunction{#1}{#2}** instruction defines a new polar function called #1. #2 is the list of instructions to calculate the radius r and his derivative Dr from the t arc variable.

```

1260 \def\newpolarfunction#1#2{%
1261     \ifx #1\undefined
1262         \ccls@defpolarfunction{#1}{#2}
1263     \else
1264         \ccls@ErrorPFuncDef{#1}
1265     \fi}

```

\renewpolarfunction **\renewpolarfunction** redefines #1 if already defined.

```

1266 \def\renewpolarfunction#1#2{%

```

```

1267      \ifx #1\undefined
1268          \ccls@ErrorPFuncUnDef{#1}
1269      \else
1270          \ccls@defpolarfunction{#1}{#2}
1271      \fi}

```

\ensurepolarfunction **\ensurepolarfunction** defines (only if undefined) #1.

```

1272 \def\ensurepolarfunction#1#2{%
1273     \ifx #1\undefined\ccls@defpolarfunction{#1}{#2}
1274     \else
1275         \ccls@InfoPFuncEns{#1}
1276     \fi}

```

\forcepolarfunction **\forcepolarfunction** defines (if undefined) or redefines (if defined) #1.

```

1277 \def\forcepolarfunction#1#2{%
1278     \ccls@defpolarfunction{#1}{#2}}

```

\ccls@defpolarfunction The private **\ccls@defpolarfunction** command makes the real work. The new functions will have three arguments: #1, a number (the polar radius), #2, #3, #4, and #5, the x and y component functions and its derivatives at #1.

```

1279 \def\ccls@defpolarfunction#1#2{%
1280     \def#1#1#2#3#4#5{%
1281         \begingroup
1282             \def\t{#1}
1283             #2
1284             \COS{\t}\ccls@cost
1285             \MULTIPLY\r\ccls@cost{x}
1286             \SIN{\t}\ccls@sint
1287             \MULTIPLY\r\ccls@sint{y}
1288             \MULTIPLY\ccls@cost\Dr\Dx
1289             \SUBTRACT{Dx}{y}{Dx}
1290             \MULTIPLY\ccls@sint\Dr\Dy
1291             \ADD{Dy}{x}{Dy}
1292             \xdef#2{x}
1293             \xdef#3{Dx}
1294             \xdef#4{y}
1295             \xdef#5{Dy}
1296         \endgroup\ignorespaces}

```

New vector functions

\newvectorfunction The **\newvectorfunction{#1}{#2}** instruction defines a new vector (parametric) function called #1. #2 is the list of instructions to calculate x , y , Dx and Dy from the t arc variable.

```

1297 \def\newvectorfunction#1#2{%
1298     \ifx #1\undefined
1299         \ccls@defvectorfunction{#1}{#2}
1300     \else
1301         \ccls@ErrorVFuncDef{#1}
1302     \fi}

```

`\renewvectorfunction` `\renewvectorfunction` redefines #1 if already defined.

```

1303 \def\renewvectorfunction#1#2{%
1304     \ifx #1\undefined
1305         \ccls@ErrorVFuncUnDef{#1}
1306     \else
1307         \ccls@defvectorfunction{#1}{#2}
1308     \fi}

```

`\ensurevectorfunction` `\ensurevectorfunction` defines (only if undefined) #1.

```

1309 \def\ensurevectorfunction#1#2{%
1310     \ifx #1\undefined\ccls@defvectorfunction{#1}{#2}
1311     \else
1312         \ccls@InfoVFuncEns{#1}
1313     \fi}

```

`\forcevectorfunction` `\forcevectorfunction` defines (if undefined) or redefines (if defined) #1.

```

1314 \def\forcevectorfunction#1#2{%
1315     \ccls@defvectorfunction{#1}{#2}}

```

`\ccls@defvectorfunction` The private `\ccls@defvectorfunction` command makes the real work. The new functions will have three arguments: #1, a number, #2, #3, #4, and #5, the x and y component functions and its derivatives at #1.

```

1316 \def\ccls@defvectorfunction#1#2{%
1317     \def###1##2###3##4##5{%
1318         \begingroup
1319         \def\t{##1}
1320         #2
1321         \xdef##2{\x}
1322         \xdef##3{\Dx}
1323         \xdef##4{\y}
1324         \xdef##5{\Dy}
1325         \endgroup\ignorespaces}

```

13.3 Polynomials

Linear (first degree) polynomials

`\newlpoly` The `\newlpoly{#1}{#2}{#3}` instruction defines the linear polynomial $\#1 = \#2 + \#3t$.

```

1326 \def\newlpoly#1#2#3{%
1327     \newfunction{#1}{%
1328         \ccls@lpoly{#2}{#3}}

```

`\renewlpoly` We define also the `\renewlpoly`, `\ensurelpoly` and `\forcelpoly` variants.

```

1329 \def\renewlpoly#1#2#3{%
1330     \renewfunction{#1}{%
1331         \ccls@lpoly{#2}{#3}}

```


\ensurelpoly

```
1332 \def\ensurelpoly#1#2#3{%
1333     \ensurefunction{#1}{%
1334         \ccls@lpoly{#2}{#3}}}
```

\forcelpoly

```
1335 \def\forcelpoly#1#2#3{%
1336     \forcefunction{#1}{%
1337         \ccls@lpoly{#2}{#3}}}
```

\ccls@lpoly The \ccls@lpoly{#1}{#2} macro defines the new polynomial function.

```
1338 \def\ccls@lpoly#1#2{%
1339     \MULTIPLY{#2}{\t}{\y}
1340     \ADD{\y}{#1}{\y}
1341     \COPY{#2}{\Dy}}
```

Quadratic polynomials

\newqpoly The \newqpoly{#1}{#2}{#3}{#4} instruction defines the quadratic polynomial
 $\#1 = \#2 + \#3t + \#4t^2$.

```
1342 \def\newqpoly#1#2#3#4{%
1343     \newfunction{#1}{%
1344         \ccls@qpoly{#2}{#3}{#4}}}
```

\renewqpoly

```
1345 \def\renewqpoly#1#2#3#4{%
1346     \renewfunction{#1}{%
1347         \ccls@qpoly{#2}{#3}{#4}}}
```

\ensureqpoly

```
1348 \def\ensureqpoly#1#2#3#4{%
1349     \ensurefunction{#1}{%
1350         \ccls@qpoly{#2}{#3}{#4}}}
```

\forceqpoly

```
1351 \def\forceqpoly#1#2#3#4{%
1352     \forcefunction{#1}{%
1353         \ccls@qpoly{#2}{#3}{#4}}}
```

\ccls@qpoly The \ccls@qpoly{#1}{#2} macro defines the new polynomial function.

```
1354 \def\ccls@qpoly#1#2#3{%
1355     \MULTIPLY{\t}{#3}{\y}
1356     \MULTIPLY{2}{\y}{\Dy}
1357     \ADD{#2}{\Dy}{\Dy}
1358     \ADD{#2}{\y}{\y}
1359     \MULTIPLY{\t}{\y}{\y}
1360     \ADD{#1}{\y}{\y}}
```

Cubic polynomials

`\newcpoly` The `\newcpoly{#1}{#2}{#3}{#4}{#5}` instruction defines the cubic polynomial

$$\#1 = \#2 + \#3t + \#4t^2 + \#5t^3.$$

```
1361 \def\newcpoly#1#2#3#4#5{%
1362     \newfunction{#1}{%
1363         \ccls@cpoly{#2}{#3}{#4}{#5}}}
```

`\renewcpoly`

```
1364 \def\renewcpoly#1#2#3#4#5{%
1365     \renewfunction{#1}{%
1366         \ccls@cpoly{#2}{#3}{#4}{#5}}}
```

`\ensurecpoly`

```
1367 \def\ensurecpoly#1#2#3#4#5{%
1368     \ensurefunction{#1}{%
1369         \ccls@cpoly{#2}{#3}{#4}{#5}}}
```

`\forcecpoly`

```
1370 \def\forcecpoly#1#2#3#4#5{%
1371     \forcefunction{#1}{%
1372         \ccls@cpoly{#2}{#3}{#4}{#5}}}
```

`\ccls@cpoly` The `\ccls@cpoly{#1}{#2}` macro defines the new polynomial function.

```
1373 \def\ccls@cpoly#1#2#3#4{%
1374     \MULTIPLY{\t}{#4}{\y}
1375     \MULTIPLY{3}{\y}{\Dy}
1376     \ADD{#3}{\y}{\y}
1377     \MULTIPLY{2}{#3}{\ccls@temp}
1378     \ADD{\ccls@temp}{\Dy}{\Dy}
1379     \MULTIPLY{\t}{\y}{\y}
1380     \MULTIPLY{\t}{\Dy}{\Dy}
1381     \ADD{#2}{\y}{\y}
1382     \ADD{#2}{\Dy}{\Dy}
1383     \MULTIPLY{\t}{\y}{\y}
1384     \ADD{#1}{\y}{\y}
1385 }
```

13.4 Elementary functions

`\ONEfunction` The `\ONEfunction`: $y(t) = 1$, $y'(t) = 0$

```
1386 \newfunction{\ONEfunction}{%
1387     \COPY{1}{\y}
1388     \COPY{0}{\Dy}}
```

`\ZEROfunction` The `\ZEROfunction`: $y(t) = 0$, $y'(t) = 0$

```
1389 \newfunction{\ZEROfunction}{%
1390     \COPY{0}{\y}
1391     \COPY{0}{\Dy}}
```

`\IDENTITYfunction` The `\IDENTITYfunction`: $y(t) = t$, $y'(t) = 1$

```
1392 \newfunction{\IDENTITYfunction}{%
1393     \COPY{\t}{\y}
1394     \COPY{1}{\Dy}}
```

`\RECIPROCALfunction` The `\RECIPROCALfunction`: $y(t) = 1/t$, $y'(t) = -1/t^2$

```
1395 \newfunction{\RECIPROCALfunction}{%
1396     \DIVIDE{1}{\t}{\y}
1397     \SQUARE{\y}{\Dy}
1398     \MULTIPLY{-1}{\Dy}{\Dy}}
```

`\SQUAREfunction` The `\SQUAREfunction`: $y(t) = t^2$, $y'(t) = 2t$

```
1399 \newfunction{\SQUAREfunction}{%
1400     \SQUARE{\t}{\y}
1401     \MULTIPLY{2}{\t}{\Dy}}
```

`\CUBEfunction` The `\CUBEfunction`: $y(t) = t^3$, $y'(t) = 3t^2$

```
1402 \newfunction{\CUBEfunction}{%
1403     \SQUARE{\t}{\Dy}
1404     \MULTIPLY{\t}{\Dy}{\y}
1405     \MULTIPLY{3}{\Dy}{\Dy}}
```

`\SQRTfunction` The `\SQRTfunction`: $y(t) = \sqrt{t}$, $y'(t) = 1/(2\sqrt{t})$

```
1406 \newfunction{\SQRTfunction}{%
1407     \SQRT{\t}{\y}
1408     \DIVIDE{0.5}{\y}{\Dy}}
```

`\EXPfunction` The `\EXPfunction`: $y(t) = \exp t$, $y'(t) = \exp t$

```
1409 \newfunction{\EXPfunction}{%
1410     \EXP{\t}{\y}
1411     \COPY{\y}{\Dy}}
```

`\COSfunction` The `\COSfunction`: $y(t) = \cos t$, $y'(t) = -\sin t$

```
1412 \newfunction{\COSfunction}{%
1413     \COS{\t}{\y}
1414     \SIN{\t}{\Dy}
1415     \MULTIPLY{-1}{\Dy}{\Dy}}
```

`\SINfunction` The `\SINfunction`: $y(t) = \sin t$, $y'(t) = \cos t$

```
1416 \newfunction{\SINfunction}{%
1417     \SIN{\t}{\y}
1418     \COS{\t}{\Dy}}
```

`\TANfunction` The `\TANfunction`: $y(t) = \tan t$, $y'(t) = 1/(\cos t)^2$

```
1419 \newfunction{\TANfunction}{%
1420     \TAN{\t}{\y}
1421     \COS{\t}{\Dy}
1422     \SQUARE{\Dy}{\Dy}
1423     \DIVIDE{1}{\Dy}{\Dy}}
```

`\COTfunction` The `\COTfunction`: $y(t) = \cot t$, $y'(t) = -1/(\sin t)^2$

```
1424 \newfunction{\COTfunction}{%
1425     \COTAN{t}{y}
1426     \SIN{t}{Dy}
1427     \SQUARE{Dy}{Dy}
1428     \DIVIDE{-1}{Dy}{Dy}}
```

`\COSHfunction` The `\COSHfunction`: $y(t) = \cosh t$, $y'(t) = \sinh t$

```
1429 \newfunction{\COSHfunction}{%
1430     \COSH{t}{y}
1431     \SINH{t}{Dy}}
```

`\SINHfunction` The `\SINHfunction`: $y(t) = \sinh t$, $y'(t) = \cosh t$

```
1432 \newfunction{\SINHfunction}{%
1433     \SINH{t}{y}
1434     \COSH{t}{Dy}}
```

`\TANHfunction` The `\TANHfunction`: $y(t) = \tanh t$, $y'(t) = 1/(\cosh t)^2$

```
1435 \newfunction{\TANHfunction}{%
1436     \TANH{t}{y}
1437     \COSH{t}{Dy}
1438     \SQUARE{Dy}{Dy}
1439     \DIVIDE{1}{Dy}{Dy}}
```

`\COTHfunction` The `\COTHfunction`: $y(t) = \coth t$, $y'(t) = -1/(\sinh t)^2$

```
1440 \newfunction{\COTHfunction}{%
1441     \COTANH{t}{y}
1442     \SINH{t}{Dy}
1443     \SQUARE{Dy}{Dy}
1444     \DIVIDE{-1}{Dy}{Dy}}
```

`\LOGfunction` The `\LOGfunction`: $y(t) = \log t$, $y'(t) = 1/t$

```
1445 \newfunction{\LOGfunction}{%
1446     \LOG{t}{y}
1447     \DIVIDE{1}{t}{Dy}}
```

`\HEAVISIDEfunction` The `\HEAVISIDEfunction`: $y(t) = \begin{cases} 0 & \text{if } t < 0 \\ 1 & \text{if } t \geq 0 \end{cases}$, $y'(t) = 0$

```
1448 \newfunction{\HEAVISIDEfunction}{%
1449     \ifdim t\p@<\z@ \COPY{0}{y}\else\COPY{1}{y}\fi
1450     \COPY{0}{Dy}}
```

13.5 Operations with functions

`\CONSTANTfunction` `\CONSTANTfunction` defines #2 as the constant function $f(t) = \#1$.

```
1451 \def\CONSTANTfunction#1#2{%
1452     \def#2##1##2##3{%
1453         \xdef##2{#1}%
1454         \xdef##3{0}}}
```

`\SUMfunction` `\SUMfunction` defines #3 as the sum of functions #1 and #2.

```

1455 \def\SUMfunction#1#2#3{%
1456     \def#3##1##2##3{%
1457         \begingroup
1458             #1{##1}{\ccls@SUMf}{\ccls@SUMDf}%
1459             #2{##1}{\ccls@SUMg}{\ccls@SUMDg}%
1460             \ADD{\ccls@SUMf}{\ccls@SUMg}{\ccls@SUMfg}
1461             \ADD{\ccls@SUMDf}{\ccls@SUMDg}{\ccls@SUMDfg}
1462             \xdef##2{\ccls@SUMfg}%
1463             \xdef##3{\ccls@SUMDfg}%
1464         \endgroup}\ignorespaces}

```

`\SUBTRACTfunction` `\SUBTRACTfunction` defines #3 as the difference of functions #1 and #2.

```

1465 \def\SUBTRACTfunction#1#2#3{%
1466     \def#3##1##2##3{%
1467         \begingroup
1468             #1{##1}{\ccls@SUBf}{\ccls@SUBDf}%
1469             #2{##1}{\ccls@SUBg}{\ccls@SUBDg}%
1470             \SUBTRACT{\ccls@SUBf}{\ccls@SUBg}{\ccls@SUBfg}
1471             \SUBTRACT{\ccls@SUBDf}{\ccls@SUBDg}{\ccls@SUBDfg}
1472             \xdef##2{\ccls@SUBfg}%
1473             \xdef##3{\ccls@SUBDfg}%
1474         \endgroup}\ignorespaces}

```

`\PRODUCTfunction` `\PRODUCTfunction` defines #3 as the product of functions #1 and #2.

```

1475 \def\PRODUCTfunction#1#2#3{%
1476     \def#3##1##2##3{%
1477         \begingroup
1478             #1{##1}{\ccls@PROf}{\ccls@PRODf}%
1479             #2{##1}{\ccls@PROg}{\ccls@PRODg}%
1480             \MULTIPLY{\ccls@PROf}{\ccls@PROg}{\ccls@PROfg}
1481             \MULTIPLY{\ccls@PROf}{\ccls@PRODg}{\ccls@PROfDg}
1482             \MULTIPLY{\ccls@PRODf}{\ccls@PROg}{\ccls@PRODfg}
1483             \ADD{\ccls@PROfDg}{\ccls@PRODfg}{\ccls@PRODfg}
1484             \xdef##2{\ccls@PROfg}%
1485             \xdef##3{\ccls@PRODfg}%
1486         \endgroup}\ignorespaces}

```

`\QUOTIENTfunction` `\QUOTIENTfunction` defines #3 as the quotient of functions #1 and #2.

```

1487 \def\QUOTIENTfunction#1#2#3{%
1488     \def#3##1##2##3{%
1489         \begingroup
1490             #1{##1}{\ccls@QUOf}{\ccls@QUODf}%
1491             #2{##1}{\ccls@QUOg}{\ccls@QUODg}%
1492             \DIVIDE{\ccls@QUOf}{\ccls@QUOg}{\ccls@QUOfg}
1493             \MULTIPLY{\ccls@QUOf}{\ccls@QUODg}{\ccls@QUOfDg}
1494             \MULTIPLY{\ccls@QUODf}{\ccls@QUOg}{\ccls@QUODfg}
1495             \SUBTRACT{\ccls@QUODfg}{\ccls@QUOfDg}{\ccls@QUOnum}
1496             \SQUARE{\ccls@QUOg}{\ccls@qsquaretempg}
1497             \DIVIDE{\ccls@QUOnum}{\ccls@qsquaretempg}{\ccls@QUODfg}

```

```

1498             \xdef##2{\ccls@QUOfg}%
1499             \xdef##3{\ccls@QUODfg}%
1500         \endgroup}\ignorespaces}

```

\COMPOSITIONfunction \COMPOSITIONfunction defines #3 as the composition of functions #1 and #2.

```

1501 \def\COMPOSITIONfunction#1#2#3{% #3=#1(#2)
1502     \def###1##2##3{%
1503         \begingroup
1504             #2{##1}{\ccls@COMg}{\ccls@COMDg}%
1505             #1{\ccls@COMg}{\ccls@COMf}{\ccls@COMDf}%
1506             \MULTIPLY{\ccls@COMDg}{\ccls@COMDf}{\ccls@COMDf}
1507             \xdef##2{\ccls@COMf}%
1508             \xdef##3{\ccls@COMDf}%
1509         \endgroup}\ignorespaces}

```

\SCALEfunction \SCALEfunction defines #3 as the product of number #1 and function #2.

```

1510 \def\SCALEfunction#1#2#3{%
1511     \def###1##2##3{%
1512         \begingroup
1513             #2{##1}{\ccls@SCFf}{\ccls@SCFDf}%
1514             \MULTIPLY{#1}{\ccls@SCFf}{\ccls@SCFaf}
1515             \MULTIPLY{#1}{\ccls@SCFDf}{\ccls@SCFDaf}
1516             \xdef##2{\ccls@SCFaf}%
1517             \xdef##3{\ccls@SCFDaf}%
1518         \endgroup}\ignorespaces}

```

\SCALEVARIABLEfunction \SCALEVARIABLEfunction scales the variable by number #1 and applies function #2.

```

1519 \def\SCALEVARIABLEfunction#1#2#3{%
1520     \def###1##2##3{%
1521         \begingroup%
1522             \MULTIPLY{#1}{##1}{\ccls@SCVat}
1523             #2{\ccls@SCVat}{\ccls@SCVf}{\ccls@SCVDf}%
1524             \MULTIPLY{#1}{\ccls@SCVDf}{\ccls@SCVDf}
1525             \xdef##2{\ccls@SCVf}%
1526             \xdef##3{\ccls@SCVDf}%
1527         \endgroup}\ignorespaces}

```

\POWERfunction \POWERfunction defines #3 as the power of function #1 to exponent #2.

```

1528 \def\POWERfunction#1#2#3{%
1529     \def###1##2##3{%
1530         \begingroup
1531             #1{##1}{\ccls@POWf}{\ccls@POWDf}%
1532             \POWER{\ccls@POWf}{#2}{\ccls@POWfn}
1533             \SUBTRACT{#2}{1}{\ccls@nminusone}
1534             \POWER{\ccls@POWf}{\ccls@nminusone}{\ccls@POWDfn}
1535             \MULTIPLY{#2}{\ccls@POWDfn}{\ccls@POWDfn}
1536             \MULTIPLY{\ccls@POWDfn}{\ccls@POWDf}{\ccls@POWDfn}
1537             \xdef##2{\ccls@POWfn}%
1538             \xdef##3{\ccls@POWDfn}%
1539         \endgroup}\ignorespaces}

```

LINEARCOMBINATIONfunction `\LINEARCOMBINATIONfunction` defines the new function #5 as the linear combination #1#2+#3#4.
#1 and #3 are two numbers. #1 and #3 are two functions.

```
1540 \def\LINEARCOMBINATIONfunction#1#2#3#4#5{%
1541     \def##1##2##3{%
1542         \begingroup
1543             #2{##1}{\ccls@LINf}{\ccls@LINDf}%
1544             #4{##1}{\ccls@LING}{\ccls@LINDg}%
1545             \MULTIPLY{#1}{\ccls@LINf}{\ccls@LINf}
1546             \MULTIPLY{#3}{\ccls@LING}{\ccls@LING}
1547             \MULTIPLY{#1}{\ccls@LINDf}{\ccls@LINDf}
1548             \MULTIPLY{#3}{\ccls@LINDg}{\ccls@LINDg}
1549             \ADD{\ccls@LINf}{\ccls@LING}{\ccls@LINafbg}
1550             \ADD{\ccls@LINDf}{\ccls@LINDg}{\ccls@LINDafbg}
1551             \xdef##2{\ccls@LINafbg}%
1552             \xdef##3{\ccls@LINDafbg}%
1553         \endgroup}\ignorespaces}
```

\POLARfunction `\POLARfunction` defines the polar curve #2. #1 is a previously defined function.

```
1554 \def\POLARfunction#1#2{%
1555     \PRODUCTfunction{#1}{\COSfunction}{\ccls@polarx}
1556     \PRODUCTfunction{#1}{\SINfunction}{\ccls@polary}
1557     \PARAMETRICfunction{\ccls@polarx}{\ccls@polary}{#2}}
```

\PARAMETRICfunction `\PARAMETRICfunction` defines the parametric curve #3. #1 and #2 are the components functions (two previously defined functions).

```
1558 \def\PARAMETRICfunction#1#2#3{%
1559     \def##1##2##3##4##5{%
1560         #1{##1}{##2}{##3}
1561         #2{##1}{##4}{##5}}}
```

\VECTORfunction `\VECTORfunction`: an alias of `\PARAMETRICfunction`.

```
1562 \let\VECTORfunction\PARAMETRICfunction
```

```
1563 % </calculus>
```

Change History

v1.0	General: First public version 1	autoinstallable. calculus.dtx embedded in calculus.dtx 1
v1.0a	General: calculator.dtx modified to make it	

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols		
\#	652	\@MATRIXPRODUCT . . 980, 994
\@@@DETERMINANT . 1001, 1014		\@MATRIXSUB 904, 918
\@@@INVERSEMATRIX 1030, 1069		\@MATRIXVECTORPRODUCT 932, 945
\@@@MATRIXABSVVALUE . 923, 931		\@@OUTPUTMATRIX . . . 857, 873
\@@@MATRIXADD 892, 903		\@@OUTPUTVECTOR . . . 694, 704
\@@@MATRIXCOPY 836, 844		\@@ROUND 242–244, 248
\@@@MATRIXGLOBALCOPY 848, 856		\@@SCALARMATRIXPRODUCT 964, 978
\@@@MATRIXPRODUCT . 984, 995		\@@SCALARPRODUCT . . 706, 722
\@@@MATRIXSUB 908, 919		\@@SCALARVECTORPRODUCT 760, 772
\@@@MATRIXVECTORPRODUCT 936, 946		\@@SOLVELINEARSYSTEM 1082, 1144
\@@@OUTPUTMATRIX . . 861, 874		\@@TDMATRIXSOL . . . <u>824</u> , 1045
\@@@OUTPUTVECTOR . . 698, 705		\@@TRANPOSEMATRIX . 875, 886
\@@@SCALARMATRIXPRODUCT 968, 979		\@@TRUNCATE 223–225, 250, 259
\@@@SCALARPRODUCT . 711, 723		\@@VECTORABSVVALUE . 749, 758
\@@@SCALARVECTORPRODUCT 764, 773		\@@VECTORADD 724, 736
\@@@SOLVELINEARSYSTEM 1127, 1146		\@@VECTORCOPY 672, 681
\@@@TRANPOSEMATRIX 878, 887		\@@VECTORGLOBALCOPY 683, 692
\@@@TRUNCATE . . 229, 231, 234		\@@VECTORMATRIXPRODUCT 947, 962
\@@@VECTORABSVVALUE . 752, 759		\@@VECTORSUB 738, 747
\@@@VECTORADD 728, 737		\@ADJMATRIX . 1037, 1050, 1133
\@@@VECTORCOPY 675, 682		\@BASICEXP . . . 544, 552, <u>559</u>
\@@@VECTORGLOBALCOPY 686, 693		\@BASICLOG 645, <u>651</u>
\@@@VECTORMATRIXPRODUCT 951, 963		\@BASICSINE . . . 337, <u>346</u> , 468
\@@@VECTORSUB 741, 748		\@BASICTAN . . . 383, <u>388</u> , 496
\@DEGREESCOS 451, <u>516</u>		\@CONVERTDEG 514, 516, 518, 520, <u>522</u>
\@DEGREESCOT 453, <u>520</u>		\@DEGREES 515, 517, 519, 521–523
\@DEGREESSIN 450, <u>514</u>		\@DEGREESCOS 451, <u>477</u>
\@DEGREESTAN 452, <u>518</u>		\@DEGREESCOT 453, <u>501</u>
\@DETERMINANT . . . 996, 1013		\@DEGREESSIN . . 450, <u>454</u> , 515
\@EXP 524, <u>525</u>		\@DEGREESTAN 452, <u>481</u>
\@FRACTIONALPART . 213, 215		\@DIVIDE . . 119, 129, <u>134</u> , 182
\@INTEGERPART 198, 200		\@EXP 524, 533, <u>535</u>
\@INVERSEMATRIX 1015, 1067		\@FRACTIONALPART . . 215, 221
\@LOG 628, <u>647</u>		\@INCSYS 1070, 1094, 1097, 1120
\@MATRIXABSVVALUE . 920, 930		\@INTEGERDIVIDE . . . 135, <u>138</u>
\@MATRIXADD 888, 902		\@INTEGERPART 200, 210
\@MATRIXCOPY 833, 843		\@LOG 628, <u>629</u> , 648, 649
\@MATRIXGLOBALCOPY 845, 855		
		\@MATRIXSIZE 830, 831
		\@NUMBERSOL <u>829</u> , 1007
		\@OUTPUTMATRIX 857, 887, 903, 919, 931, 979, 995, 1069
		\@OUTPUTSOL . <u>89</u> , 133, 155, 165, 178, 194, 197, 211, 222, 233, 261, 282, 322, 345, 363, 367, 387, 412, 425, 476, 480, 500, 513, 534, 558, 579, 592, 605, 616, 627, 646, 650, 668, 723, 777, 1014
		\@OUTPUTSOLS <u>90</u> , 191
		\@OUTPUTVECTOR 694, 782, 946, 963, 1147
		\@POWER 165, 166, 171
		\@ROUND 242, 243
		\@SCLRDIVVECT 1038, 1040, 1042, 1047, 1137
		\@SOLPART 1073, 1103, 1107, 1111, 1115
		\@TDINCSYS 1077, 1131
		\@TDMATRIXCOPY 783, 837, 849, 879
		\@TDMATRIXGLOBALSOL <u>803</u> , 850
		\@TDMATRIXNOSOL <u>813</u> , 826, 1032
		\@TDMATRIXSOL 793, 828, 838, 880, 896, 912, 972, 988
		\@TRUNCATE 223, 224
		\@VECTORSIZE 669, 670
		A
		\ABSVVALUE 7, <u>99</u> , 112, 113, 181, 264, 265, 289, 348, 390, 537, 582, 595, 608, 619, 750, 753, 1064, 1086–1088, 1129
		\ADD 6, <u>105</u> , 109, 131, 177, 186, 190, 257, 314, 333,

366, 379, 431, 442,	\ccls@nminusone . 1533, 1534	\cctr@@deta 1086, 1089
463, 479, 492, 563,	\ccls@polarx 1555, 1557	\cctr@@detb 1087, 1093
566, 569, 572, 575,	\ccls@polary 1556, 1557	\cctr@@detc 1088, 1096
579, 590, 638, 661,	\ccls@POWdf 1531, 1536	\cctr@absval 537,
709, 714, 716, 725,	\ccls@POWdfn 1534–1536, 1538	538, 582, 583, 595,
726, 729–731, 1004,	\ccls@POWf . 1531, 1532, 1534	596, 608, 609, 619, 620
1006, 1291, 1340,	\ccls@POWfn 1532, 1537	\cctr@ae . . 636, 637, 641, 642
1357, 1358, 1360,	\ccls@PROdf 1478, 1482	\cctr@det . . . 1002–1006,
1376, 1378, 1381,	\ccls@PROdfg 1482, 1483, 1485	1026, 1027, 1038,
1382, 1384, 1460,	\ccls@PRODg 1479, 1481	1040, 1042, 1063,
1461, 1483, 1549, 1550	\ccls@PROf . 1478, 1480, 1481	1064, 1128, 1129, 1137
	\ccls@PROfdg 1481, 1483	\cctr@deta
C	\ccls@PROfg 1480, 1484	. 1083, 1086, 1090–1092
\ccls@COMdf . 1505, 1506, 1508	\ccls@PROg . 1479, 1480, 1482	\cctr@detb . 1084, 1087, 1090
\ccls@COMdg 1504, 1506	\ccls@qpoly 1344,	\cctr@detc . 1085, 1088, 1091
\ccls@COMf 1505, 1507	1347, 1350, 1353, <u>1354</u>	\cctr@epsilon . . . 6, 311,
\ccls@COMg 1504, 1505	\ccls@qsquaretempg 1496, 1497	527, 631, 656, 1016,
\ccls@cost . 1284, 1285, 1288	\ccls@QUODf 1490, 1494	1031, 1089, 1093,
\ccls@cpoly 1363,	\ccls@QUODfg	1096, 1101, 1105,
1366, 1369, 1372, <u>1373</u>	. 1494, 1495, 1497, 1499	1109, 1113, 1118, 1130
\ccls@deffunction 1235,	\ccls@QUODg 1491, 1493	\cctr@expminusx
1243, 1246, 1251, <u>1252</u>	\ccls@QUOf . 1490, 1492, 1493 589, 590, 602, 603
\ccls@defpolarfunction .	\ccls@QUOfDg 1493, 1495	\cctr@expt 546, 547, 554, 555
. 1262,	\ccls@QUOfg 1492, 1498	\cctr@expx 587, 590, 600, 603
1270, 1273, 1278, <u>1279</u>	\ccls@QUOg	\cctr@expy 547, 548, 555, 556
\ccls@defvectorfunction	. 1491, 1492, 1494, 1496	\cctr@lengtha 4, 102–105,
. 1299,	\ccls@QUOnum 1495, 1497	107, 108, 139, 145,
1307, 1310, 1315, <u>1316</u>	\ccls@SCFaf 1514, 1516	146, 150, 152, 656, 657
\ccls@ErrorFuncDef 1174, 1237	\ccls@SCFDaf 1515, 1517	\cctr@lengthb 5, 106, 107,
\ccls@ErrorFuncUnDef . . .	\ccls@SCFDf 1513, 1515	140, 141, 145, 151,
. 1181, 1241	\ccls@SCFf 1513, 1514	153, 310, 311, 318,
\ccls@ErrorPFuncDef	\ccls@SCVat 1522, 1523	320, 655, 657, 664, 666
. 1191, 1264	\ccls@SCVdf . 1523, 1524, 1526	\cctr@log 531–533
\ccls@ErrorPFuncUnDef . .	\ccls@SCVf 1523, 1525	\cctr@loga 648, 650
. 1199, 1268	\ccls@sint . 1286, 1287, 1290	\cctr@logmaxnum . . 8, 71,
\ccls@ErrorVFuncDef	\ccls@SUBDf 1468, 1471	538, 583, 596, 609, 620
. 1212, 1301	\ccls@SUBDfg 1471, 1473	\cctr@logx 649, 650
\ccls@ErrorVFuncUnDef . .	\ccls@SUBDg 1469, 1471	\cctr@minust 588, 589, 601, 602
. 1220, 1305	\ccls@SUBf 1468, 1470	\cctr@ndec . . . 121, 122, 131
\ccls@InfoFuncEns 1187, 1248	\ccls@SUBfg 1470, 1472	\cctr@outa 89,
\ccls@InfoPFuncEns 1208, 1275	\ccls@SUBg 1469, 1470	90, 92, 695, 696, 699,
\ccls@InfoVFuncEns 1229, 1312	\ccls@SUMdf 1458, 1461	700, 858, 859, 863, 867
\ccls@LINAfbg . . . 1549, 1551	\ccls@SUMDfg 1461, 1463	\cctr@outb
\ccls@LINDafbg . . 1550, 1552	\ccls@SUMDg 1459, 1461	91, 92, 695, 696, 699,
\ccls@LINDf . 1543, 1547, 1550	\ccls@SUMf 1458, 1460	700, 858, 859, 863, 867
\ccls@LINDg . 1544, 1548, 1550	\ccls@SUMfg 1460, 1462	\cctr@outc 699,
\ccls@LINF . 1543, 1545, 1549	\ccls@SUMg 1459, 1460	700, 858, 859, 863, 867
\ccls@LING . 1544, 1546, 1549	\ccls@temp 1377, 1378	\cctr@outd 858, 859, 864, 868
\ccls@lpoly 1328,	\cctr@@det 1016,	\cctr@oute 864, 868
1331, 1334, 1337, <u>1338</u>	1031, 1064, 1129, 1130	\cctr@outf 864, 868

\cctr@outg	865, 869	\cctr@solCC 792, 802, 812, 895,	\cctr@tempr 119, 123, 124, 126, 129
\cctr@outh	865, 869		911, 971, 987, 1034,	\cctr@tempw	312–
\cctr@outi	865, 869		1042, 1043, 1059, 1136		314, 317, 318, 654,
\cctr@Q	120, 130, 132	\cctr@tanhden 614, 615, 624, 626		658, 659, 661–663, 668
\cctr@sign	114–117, 132		\cctr@tanhnum	\cctr@tempwx	659–661
\cctr@size 679, 680, 690, 691,	 613, 615, 625, 626	\cctr@tempz	313–316
	702, 703, 720, 721,	\cctr@temp	194, 197, 203–	\cctr@Warndivzero	9, 143
	734, 735, 745, 746,		205, 247, 248, 776, 777	\cctr@WarnIncLinSys	40, 1070
	756, 757, 770, 771,	\cctr@tempA	218, 219	\cctr@WarnIncTDLinSys	43, 1077
	841, 842, 853, 854,	\cctr@tempa	152,	\cctr@WarnIndLinSys	47, 1073
	871, 872, 884, 885,		154, 173, 175, 177,	\cctr@Warninfcotan 61, 421, 509
	900, 901, 916, 917,		227, 228, 264, 266,	\cctr@Warninfexp	67,
	928, 929, 943, 944,		267, 269, 277–281,		539, 584, 597, 610, 621
	960, 961, 976, 977,		348, 349, 351, 353,	\cctr@Warninfexpb	75, 528
	992, 993, 1011, 1012,		354, 356, 359, 390,	\cctr@Warninflog	82, 632
	1065, 1066, 1142, 1143		391, 393, 397, 398,	\cctr@WarninfTan 55, 371, 375, 484, 488
\cctr@sol	829, 1002, 1004, 1006		400, 401, 403, 404,	\cctr@Warnnogcd	14, 271
\cctr@solAA	784,		406, 407, 409, 410,	\cctr@Warnnointexp	24, 162
	794, 804, 825, 893,		561, 562, 564, 567,	\cctr@Warnnuposrad	18, 307
	909, 969, 985, 1032,		570, 573, 708, 709,	\cctr@Warnsingmatrix	30, 1021
	1038, 1039, 1051, 1134		713–716, 780–782,	\cctr@WarnsingTMatrix 35, 1035
\cctr@solAB 785, 795, 805, 893,	\cctr@tempb	953–956, 998, 999,		\cctr@WarnZeroLinSys
	909, 969, 985, 1032,		1099, 1101, 1117, 1118		51, 1122
	1038, 1039, 1054, 1134			\COMPOSITIONfunction	23, 1501
\cctr@solAC 786, 796, 806, 893,			\CONSTANTfunction	22, 1451
	909, 969, 985, 1032,			\COPY	5, 87,
	1038, 1039, 1057, 1134				89, 92, 95, 98, 101,
\cctr@solBA 787, 797, 807, 894,				114–117, 120, 121,
	910, 970, 986, 1033,				126, 130, 146, 173,
	1040, 1041, 1052, 1135				174, 198, 199, 207,
\cctr@solBB 788, 798, 808, 894,				213, 214, 236–240,
	910, 970, 986, 1033,				248, 268, 274, 278,
	1040, 1041, 1055, 1135				280, 281, 285, 293,
\cctr@solBC 789, 799, 809, 894,				303, 309, 312, 316,
	910, 970, 986, 1033,				326, 328, 330, 349,
	1040, 1041, 1058, 1135				351, 391, 394, 417,
\cctr@solCA 790, 800, 810, 895,				429, 438, 440, 449,
	911, 971, 987, 1034,				456, 458, 460, 505,
	1042, 1043, 1053, 1136				658, 668, 670, 671,
\cctr@solCB 791, 801, 811, 895,				673, 676, 784–792,
	911, 971, 987, 1034,				794–802, 829, 831,
	1042, 1043, 1056, 1136				832, 834, 876, 1022,
					1023, 1075, 1123,
					1341, 1387, 1388,
					1390, 1391, 1393,
					1394, 1411, 1449, 1450

<code>\COS</code> 11, <u>364</u> , 415, 1284, 1413, 1418, 1421	1388, 1391, 1394, 1397, 1398, 1401, 1403–1405, 1408, 1411, 1414, 1415, 1418, 1421–1423, 1426–1428, 1431, 1434, 1437–1439, 1442–1444, 1447, 1450	<code>\INVERSEMATRIX</code> 19, <u>1015</u>
<code>\COSfunction</code> . 21, <u>1412</u> , 1555		L
<code>\COSH</code> 13, <u>580</u> , 614, 625, 1430, 1434, 1437		<code>\LCM</code> 10, <u>283</u>
<code>\COSHfunction</code> 21, <u>1429</u>		<code>\LENGTHDIVIDE</code> 6, <u>148</u>
<code>\COT</code> 12, <u>413</u>		<code>\LINEARCOMBINATIONfunction</code> 23, <u>1540</u>
<code>\COTAN</code> 1425		<code>\LOG</code> . . 11, 531, <u>628</u> , 642, 1446
<code>\COTANH</code> 1441		<code>\LOGfunction</code> 21, <u>1445</u>
<code>\COTfunction</code> 21, <u>1424</u>	E	
<code>\COTH</code> 13, <u>617</u>	<code>\ensurecpoly</code> 25, <u>1367</u>	M
<code>\COTHfunction</code> 21, <u>1440</u>	<code>\ensurefunction</code> 27, 1190, 1245, 1333, 1349, 1368	<code>\MATRIXABSTRACTVALUE</code> 20, <u>920</u> , 1099
<code>\CUBE</code> 7, <u>157</u>	<code>\ensurepelpoly</code> 25, <u>1332</u>	<code>\MATRIXADD</code> 17, <u>888</u>
<code>\CUBEfunction</code> 21, <u>1402</u>	<code>\ensurepolarfunction</code> 27, 1211, <u>1272</u>	<code>\MATRIXCOPY</code> 16, <u>833</u> , 859, 866
D	<code>\ensureqpoly</code> 25, <u>1348</u>	<code>\MATRIXGLOBALCOPY</code> 845, 858, 862
<code>\DEGREESCOS</code> 12, <u>451</u> , 503, 517	<code>\ensurevectorfunction</code> 27, 1232, <u>1309</u>	<code>\MATRIXPRODUCT</code> 18, <u>980</u>
<code>\DEGREESCOT</code> 12, <u>453</u> , 521	<code>\EXP</code> 11, <u>524</u> , 587, 589, 600, 602, 659, 1410	<code>\MATRIXSIZE</code> 830, 841, 853, 871, 884, 900, 916, 928, 943, 976, 992, 1011, 1065, 1142
<code>\DEGREESSIN</code> 12, <u>450</u> , 464, 472, 475, 480	<code>\EXPfunction</code> 21, <u>1409</u>	<code>\MATRIXSUB</code> 17, <u>904</u>
<code>\DEGREESTAN</code> 12, <u>452</u> , 492, 499, 507, 519	F	<code>\MATRIXVECTORPRODUCT</code> 18, <u>932</u> , 981, 982, 1134
<code>\DEGtoRAD</code> . 13, <u>426</u> , 467, 495	<code>\FLOOR</code> 8, <u>212</u>	<code>\MAX</code> 5, <u>93</u> , 266
<code>\DETERMINANT</code> 19, <u>996</u> , 1051–1059, 1063, 1083–1085, 1128	<code>\forcecpoly</code> 25, <u>1370</u>	<code>\MIN</code> 6, <u>96</u> , 267
<code>\DIVIDE</code> 6, <u>110</u> , 154, 169, 256, 287, 296, 297, 313, 315, 354, 357, 360, 396, 397, 399, 400, 402, 403, 405, 406, 408, 409, 411, 424, 426, 512, 522, 546, 554, 562, 564, 565, 567, 568, 570, 571, 573, 574, 577, 591, 604, 615, 626, 636, 650, 660, 781, 1026, 1048, 1074, 1090, 1091, 1150, 1152–1155, 1157, 1396, 1408, 1423, 1428, 1439, 1444, 1447, 1492, 1497	<code>\forcefunction</code> 27, <u>1250</u> , 1336, 1352, 1371	<code>\MODULO</code> 9, <u>192</u> , 279
<code>\Dr</code> 1288, 1290	<code>\forcelpoly</code> 25, <u>1335</u>	<code>\MULTIPLY</code> . . . 6, 101, <u>102</u> , 124, 127, 132, 136, 156, 157, 170, 176, 203, 247, 253, 288, 298, 356, 359, 362, 427, 523, 532, 578, 588, 601, 641, 707, 708, 712, 713, 715, 761, 762, 765–767, 997, 998, 1002, 1003, 1005, 1024, 1025, 1149, 1151, 1285, 1287, 1288, 1290, 1339, 1355, 1356, 1359, 1374, 1375, 1377, 1379, 1380, 1383, 1398, 1401, 1404, 1405, 1415, 1480–1482, 1493, 1494, 1506, 1514, 1515, 1522, 1524, 1535, 1536, 1545–1548
<code>\Dx</code> . . 1288, 1289, 1293, 1322	<code>\forcepolarfunction</code> 27, <u>1314</u>	
<code>\Dy</code> . . . 1258, 1290, 1291, 1295, 1324, 1341, 1356, 1357, 1375, 1378, 1380, 1382,	<code>\forceqpoly</code> 25, <u>1351</u>	
	<code>\forcevectorfunction</code> 27, <u>1314</u>	
	<code>\FRACTIONALPART</code> 8, <u>213</u>	
	<code>\FRACTIONSIMPLIFY</code> . 10, <u>291</u>	
	G	
	<code>\GCD</code> 10, 262, 284, 295	
	<code>\GLOBALCOPY</code> . . . 88, 89–91, 684, 687, 804–812, 846	
	H	
	<code>\HEAVISIDEfunction</code> . 21, <u>1448</u>	
	I	
	<code>\IDENTITYfunction</code> . 21, <u>1392</u>	
	<code>\ifcase</code> 235	
	<code>\INTEGERDIVISION</code> 9, <u>179</u> , 194, 197	
	<code>\INTEGERPART</code> 7, 160, <u>198</u> , 212, 218, 227	
	<code>\INTEGERQUOTIENT</code> . . . 9, <u>195</u>	

N		P		\SINH 13, <u>593</u> , 613, 624, 1431, 1433, 1442	
\NeedsTeXFormat 2, 1170		\PackageError 1175, 1182, 1192, 1200, 1213, 1221		\SINHfunction 21, <u>1432</u>	
\newcpoly 24, <u>1361</u>		\PackageInfo 1188, 1209, 1230		\SOLVELINEARSYSTEM . 20, <u>1070</u>	
\newfunction . . 27, 1186, 1233, 1327, 1343, 1362, 1386, 1389, 1392, 1395, 1399, 1402, 1406, 1409, 1412, 1416, 1419, 1424, 1429, 1432, 1435, 1440, 1445, 1448		\PackageWarning . 10, 15, 19, 25, 31, 36, 40, 43, 47, 51, 56, 62, 68, 76, 83		\SQRT 10, <u>323</u> , 1407	
\newlpoly 24, <u>1326</u>		\PARAMETRICfunction 25, 1557, <u>1558</u> , 1562		\SQRTfunction 21, <u>1406</u>	
\newpolarfunction 27, 1206, <u>1260</u>		\POLARfunction 26, <u>1554</u>		\SQUARE 7, <u>156</u> , 353, 548, 556, 561, 1158, 1159, 1397, 1400, 1403, 1422, 1427, 1438, 1443, 1496	
\newqpoly 24, <u>1342</u>		\POWER 7, <u>158</u> , 252, 1532, 1534		\SQUAREfunction . . . 21, <u>1399</u>	
\newvectorfunction 27, 1227, <u>1297</u>		\POWERfunction 23, <u>1528</u>		\SQUAREROOT 10, <u>300</u> , 323, 777	
\numberCOSXLV 5, <u>1166</u>		\PRODUCTfunction 23, <u>1475</u> , 1555, 1556		\SUBTRACT 6, <u>109</u> , 137, 188, 206, 219, 251, 317, 340, 343, 355, 358, 361, 385, 398, 401, 404, 407, 410, 435, 446, 471, 474, 498, 576, 603, 643, 654, 662, 663, 999, 1289, 1470, 1471, 1495, 1533	
\numberCOSXXX 5, <u>1167</u>		\ProvidesPackage . . . 3, 1171		\SUBTRACTfunction . 22, <u>1465</u>	
\numberE 5, 636, 641, <u>1156</u> , 1157, 1158			Q	\SUMfunction 22, <u>1455</u>	
\numberETWO 5, 635, <u>1158</u>			\QUOTIENTfunction . 23, <u>1487</u>		
\numberFIFTHPI 5, <u>1154</u>			R		
\numberGOLD 5, <u>1161</u>			\r 1285, 1287		
\numberHALFPI . . . 5, 326, 328, 332, 336, 366, 370, 374, 378, 382, <u>1150</u>			\RADtoDEG 13, <u>427</u>		
\numberINVE . . . 5, <u>1157</u> , 1159			\RECIPROCALfunction 21, <u>1395</u>		
\numberINVETWO 5, <u>1159</u>			\REDUCEDEGREESANGLE 13, <u>439</u>		
\numberINGOLD 5, <u>1162</u>			\REDUCERADIANSANGLE 13, <u>428</u>		
\numberLOGTEN 5, <u>1160</u>			\renewcpoly 25, <u>1364</u>		
\numberPI 5, 340, 379, 385, 430, 434, 438, <u>1148</u> , 1149–1155			\renewfunction 27, 1180, 1239, 1330, 1346, 1365		
\numberQUARTERPI . . 5, <u>1153</u>			\renewlpoly 25, <u>1329</u>		
\numberSIXTHPI 5, <u>1155</u>			\renewpolarfunction 27, 1197, <u>1266</u>		
\numberSQRTFIVE . . . 5, <u>1165</u>			\renewqpoly 25, <u>1345</u>		
\numberSQRTTHREE . . 5, <u>1164</u>			\renewvectorfunction 27, 1218, <u>1303</u>		
\numberSQRTTWO 5, <u>1163</u>			\ROUND 8, <u>242</u>		
\numberTHIRDPI 5, <u>1152</u>			S		
\numberTHREEHALFPI 5, 330, 339, <u>1151</u>			\SCALARMATRIXPRODUCT 17, <u>964</u> , 1027		
\numberTWOPI 5, 333, 343, 431, 435, <u>1149</u>			\SCALARPRODUCT 15, <u>706</u> , 776, 933, 934, 937–939, 948, 949		
			\SCALARVECTORPRODUCT 15, 760, 782, 952, 953, 955, 965, 966, 969–971		
			\SCALEfunction 23, <u>1510</u>		
			\SCALEVARIABLEfunction 23, <u>1519</u>		
			\SIN 11, <u>324</u> , 367, 1286, 1414, 1417, 1426		
			\SINfunction . 21, <u>1416</u> , 1556		
	O				
\ONEfunction 21, 1386					
\or 237–240					
				</	

V	\VECTORNORM 15, <u>774</u> , 780, 1117	1355, 1356, 1358–
\VECTORABSVALUE	\VECTORSIZE <u>669</u> ,	1360, 1374–1376,
. 16, <u>749</u> , 921, 924	679, 690, 702, <u>720</u> ,	1379, 1381, 1383,
\VECTORADD 14,	734, 745, 756, 770, 960	1384, 1387, 1390,
<u>724</u> , 739, 742, 889,	\VECTORSUB 14, 15,	1393, 1396, 1397,
890, 893–895, 954, 956	<u>738</u> , 905, 906, 909–911	1400, 1404, 1407,
\VECTORCOPY 14, <u>672</u> , 696, 700	X	1408, 1410, 1411,
\VECTORfunction . . . 25, <u>1562</u>	\x . . . 1285, 1291, 1292, 1321	1413, 1417, 1420,
\VECTORGLOBCOPY	Y	1425, 1430, 1433,
. <u>683</u> , 695, 699	\y 1257, 1287, 1289, 1294,	1436, 1441, 1446, 1449
\VECTORMATRIXPRODUCT . . .	1323, 1339, 1340,	Z
. <u>947</u> , 985–987	\ZEROfunction 21, <u>1389</u>	