

ERNIC KAMERICH

Maple

指南

唐兢 李静 译



CHEP

高等教育出版社



Springer

施普林格出版社

图书在版编目 (CIP) 数据

Maple 指南 / (荷兰) 恩里克 (Ernic, K.) 著; 唐兢, 李静 译 - 北京: 高等教育出版社; 海德堡: 施普林格出版社, 2000.6

ISBN 7-04-008695-6

LM... II.①恩... ②唐... III.Maple 语言-软件-教材 IV.P312

中国版本图书馆 CIP 数据核字 (2000) 第 60863 号

Maple 指南

唐兢 李静 译

出版发行 高等教育出版社 施普林格出版社

社 址 北京市东城区沙滩后街 55 号

邮政编码 100009

电 话 010-64054588

传 真 010-64014048

网 址 <http://www.hep.edu.cn>

经 销 新华书店北京发行所

印 刷 北京民族印刷厂

开 本 880×1230 1/32

版 次 2000 年 6 月第 1 版

印 张 10.625

印 次 2000 年 6 月第 1 次印刷

字 数 280 000

定 价 22.00 元

©China Higher Education Press Beijing and Springer-Verlag Berlin Heidelberg 2000

版权所有 侵权必究

简要命令查询一览表

这个一览表仅包含最基本的 MAPLE 命令.

需要键入的命令元素用打字机字体 (typewriter font) 或黑体 (**bold font**) 表示.

用户可选择的代表 MAPLE 表达式的命令元素, 以斜体字 (*italic font*) 表示.

1 特殊符号

； 命令终止符

： 同上，但在屏幕上的不显示结果

:= 用于赋值

= 等号，用在方程中，不用于赋值

%1 缩略符 (在显示结果中)

%, %, %% 在 MAPLE V Release 5 中代表最后一个、倒数第二个和倒数第三个结果 (在命令行中)

", ", " " 在 V.5 之前的版本中代表最后一个、倒数第二个和倒数第三个结果

" " 一对双引号，在 V.5 版本中用来把一行字符变成字符串

' ' 一对向后单引号，在 V.5 版本中把一行字符当作符号，而在早期版本中则把一行字符当作字符串

' ' 一对向前单引号，禁止求值

- 用于字符串连接, 或在浮点数中使用
- .. 范围
- * 强制性乘法记号
- ! 阶乘符号
- & 特殊运算符前缀, 如 &*
- ~ 后缀, 表明一个变量的一条性质 (在显示结果中)
- > 用于定义函数, 此符号后的表达式用文字给出
- @ 函数的复合
- @@ 一个函数的重复复合
- () 与数学中通常的括号作用一样, 也可表示函数的参数
- [] 列表
- { } 集合
- ' 语句、列表、集合及函数参数间的分隔符
- 下划线, 用于内部名称
- \$ 重复
- :: 检验过程参数的类型
- ? 在线帮助

2 一般命令

constants	常数 e 输入为 <code>exp(1)</code> , π 为 <code>Pi</code> , i 为 <code>I</code>
eval(<i>expr</i>,1)	对 <i>expr</i> 求值, 仅做一步
assume(<i>expr</i>,<i>prop</i>)	对变量或表达式 <i>expr</i> 假设性质 <i>prop</i> , 否则未定义
<i>expr</i> mod <i>n</i>	表达式 <i>expr</i> 对整数 <i>n</i> 求模
sqrt(<i>expr</i>)	表达式 <i>expr</i> 的平方根
with	加载软件包
readlib	从标准库读入过程

3 代换, 子表达式

subs(<i>old=new</i>,<i>expr</i>)	在 <i>expr</i> 中以 <i>new</i> 替代 <i>old</i> . 如果 <i>expr</i> 代表一过程或矩阵的名称等, 则用 subs(<i>old=new</i>,eval(<i>expr</i>)) 对于同时代换, 可使用一个方程列表
eval(<i>expr</i>, <i>old=new</i>)	作用同上, 但不替换形式参数, 如 <code>diff(f(x),x)</code> 中的 <code>x</code>
simplify (<i>expr</i>, <i>equation</i>)	根据边际关系 <i>equation</i> 化简 <i>expr</i> , 可以带可选的第三个参数 [<code>< vars ></code>], 表示对哪些变量化简
op(<i>expr</i>)	生成 <i>expr</i> 的操作数; 不能用于序列
op(<i>n</i>,<i>expr</i>)	生成 <i>expr</i> 的第 <i>n</i> 个操作数; 不能用于序列

select from sequence	取决于索引: 如果 <i>sols</i> 是个序列, 则 <i>sols</i> [2] 生成序列 <i>sols</i> 的第二个元素
subsop(<i>n=expr1</i>,<i>expr2</i>)	以 <i>expr1</i> 替代 <i>expr2</i> 的第 <i>n</i> 个操作数
map(<i>proced</i>,<i>expr</i>)	对 <i>expr</i> 所有的操作数 (可能为列表或集合) 应用过程 <i>proced</i> , 并根据 <i>expr</i> 的结构组成结果
map2(<i>proced</i>, <i>expr1</i>, <i>expr2</i>)	对以 <i>expr1</i> 为第一参数, <i>expr2</i> 的操作数为第二参数应用程序 <i>proced</i> , 并根据 <i>expr2</i> 的结构组成结果
coeff(<i>poly</i>, <i>x</i>, <i>n</i>)	给出多项式 <i>poly</i> 中 x^n 项的系数
numer	生成一个比例式的分子
denom	生成一个比例式的分母
lhs, rhs	生成一个方程的左边或右边的 (表达式)

4 处理数字及公式

evalc(<i>expr</i>)	把一个复数代数表达式 <i>expr</i> 转换成 $a + bI$ 的形式, 其中 <i>a</i> 和 <i>b</i> 是实表达式, 假设 <i>expr</i> 表达式中的所有名字都是实数
convert(<i>expr</i>, polar)	把一个复数表达式 <i>expr</i> 转换成典型的极坐标形式 $\text{polar}(a, \phi)$, 也可以写作 $ae^{i\phi}$
collect(<i>poly</i>, <i>x</i>)	将 <i>poly</i> 记作 <i>x</i> 的多项式
expand(<i>expr</i>)	在 <i>expr</i> 中展开幂、作乘法和函数调用

- factor(*expr*)** 对 *expr* 作有理数上的因式分解
- normal(*expr*)** 转换代数表达式 *expr* 为商式, 应用一些基本化简规则
- combine(*expr*,*option*)** 与函数展开的作用正相反, 合并类型由选项 *option* 给出.
可能的选项有:
exp, ln, power, trig, Psi,
radical, abs, signum, plus,
atatsign, conjugate, plot,
product, range, polylog, cmbplus,
cmbtms, cmbpwr, polylog
- simplify(*expr*,*option*)** 根据给定选项 *option* 的规则化简 *expr*.
可能的选项有:
power, commonpow, radical,
RootOf, polar, infinity, max,
atsign, atatsign
函数集: trig, arctrig,
或函数名如:
exp, ln, Dirac, hypergeom 等等,
函数的复合: atsign,
重复的复合: atatsign
- convert(*expr*,*option*)** 根据选项操作; 许多选项可以用; 可参考在线帮助
- convert(*expr*,RootOf)** 将 *expr* 中的根式变成 RootOf 表达式
- convert(*expr*,radical)** 如果可能的话, 通过选择 allvalues 中的一个值, 将 *expr* 中的每个 RootOf 表达式都转变成一个根式

<code>convert(expr,rational)</code>	将 $expr$ 中出现的所有浮点数转换成有理数; 可选的第三选项为 <code>exact</code>
<code>testeq(expr1=expr2)</code>	对一给定的等式执行一次数值检验. 不能对根式和 <code>RootOf</code> 表达式用此命令

5 作图和数值计算

<code>graphics</code>	二维作图用 <code>plot</code> 命令, 三维作图用 <code>plot3d</code> . 在程序包 <code>plots</code> 和 <code>DEtools</code> 中还有其他的过程
<code>plot(expr,x=a..b)</code>	对关于 x 的函数的代数表达式 $expr$ 作图, x 的范围从 a 到 b . 可以使用多个选项和变量
<code>plot3d(expr,x=a..b,y=c..d)</code>	同上, 作三维图
<code>plots[display]({ })</code>	把几个图合并成一个图
<code>evalf(expr,n)</code>	用浮点数求 $expr$ 中所有实数的近似值. 计算执行到第 n 位数字
<code>Digits</code>	当没有特别指明时, 为确定浮点计算精度的变量

6 解方程

<code>solve(equa,x)</code>	精确求解关于变量 x 的方程 $equa$
<code>solve({equa1,equa2},{x,y})</code>	精确求解关于变量 x 和 y 的方程组 $\{equa1,equa2\}$ (也可求解更大的方程组)

<code>fsolve(equa, x)</code>	求关于 x 的方程 $equa$ 其中一个解的近似值 (如果为多项式: 则求所有解的近似值), 此命令也可用于方程组
<code>fsolve(equa, x, x = a..b)</code>	在 $[a, b]$ 上, 求关于 x 的方程 $equa$ 的解的近似值
<code>lhs, rhs</code>	生成一个方程的左, 右边表达式
<code>allvalues(expr)</code>	如果 $expr$ 中的 <code>RootOf</code> 表达式作为多项式方程求解, 则生成所有可能的值; 如果没有选项 <code>independent</code> , 则用相同的值替代同样的 <code>RootOf</code> 表达式
<code>dsolve</code>	解微分方程: 见微积分类命令

7 微积分

<code>diff(f(x), x)</code>	对表达式 $f(x)$ 关于变量 x 求微分, 其结果为表达式
<code>D(f)</code>	对函数 f 微分, 其结果为函数
<code>diff(f(x, y), x, y, y)</code>	计算 $\frac{\partial^3}{\partial x \partial y^2} f(x, y)$
<code>D[1,2,2](f)</code>	计算函数 $(x, y) \mapsto \frac{\partial^3}{\partial x \partial y^2} f(x, y)$
<code>int(expr, x)</code>	计算 $\int expr \, dx$
<code>int(expr, x = a..b)</code>	计算 $\int_a^b expr \, dx$
<code>sum(expr, k = a..b)</code>	计算 $\sum_a^b expr$
<code>product(expr, k = a..b)</code>	计算 $\prod_a^b expr$

infinity	在 MAPLE 中用来表示无穷 (∞)
numerical integration	命令行 <code>evalf(Int(expr, x = a..b))</code> 只求 $\int_a^b \text{expr} dx$ 的数值近似, 而不对积分作符号计算
series(expr, x = c, n)	求表达式 <i>expr</i> 作为 <i>x</i> 的函数在 $x = c$ 点的级数展开式. 展开阶数可由最后的参数 <i>n</i> 确定, 不过可能比 <i>n</i> 要低. 结果通常为级数类型
convert(sr, polynomial)	假定表达式 <i>sr</i> 是级数类型, 将 <i>sr</i> 转换成一个 (广义的) 多项式, 省略了阶次项
Order	在阶数没有特别设定时, 用以确定级数计算阶数的变量
limit(expr, x = a)	计算极限 $\lim_{x \rightarrow a} \text{expr}$; <i>a</i> 可以为正无穷 (infinity) 或负无穷 (-infinity); 可加选项: left, right, real 或 complex
dsolve	解微分方程或微分方程组
differential equation	表示形如 $\text{diff}(f(x), x, x) = -f(x)$ 或 $D(D(f))(x) = -f(x)$ 的微分方程
initial conditions	初始条件, 如: $f(0)=1,$ $D(f)(0)=a,$ 等

- > 箭头：为减号 “-” 和大于号 “>” 的组合，用于构成函数，例如：

`x- >x^2,`

`(x,y)- >x*cos(y)`

箭头右边的表达式用文字给出。

对一元以上的函数，参数必须用圆括号括上

unapply(*expr*,*vars*) 用计算 *expr* 的结果所描述的变量 *vars* 构造一个函数

?inifcns 生成 MAPLE 已有的全部数学函数的清单

piecewise 产生分段定义的表达式

8 线性代数

start linear algebra 读入 linalg 软件包启动线性代数功能：
`with(linalg)`

create 生成矩阵或者向量，例如
`matrix([[1,2,3][4,5,6]])`
`vector([1,2,3])`

diagonal matrix 对角矩阵，如 `diag(a,b,c)`

evaluate 用 `eval(A)` 计算矩阵 *A* (或向量) 的值，如含有指定的名字，则用 `map(eval,A)`

matrix arithmetic	矩阵算术运算, 总是用命令 <code>evalm</code> , 如 <code>evalm(5*A^2+B&*C)</code> 用 <code>&*</code> 表示矩阵与矩阵 (或向量) 的乘法, 用 <code>*</code> 表示数量乘法
transpose	对向量或矩阵作转置运算
dotprod(v, w)	作 v 和 w 的点积
copy(A)	在内存中生成一个与 A 有相同数据项的 新矩阵或向量对象
det(A)	计算 A 的行列式
eigenvals, eigenvects	计算方阵的特征值与特征向量
colspace, rowspace	计算由矩阵的列或行向量张成的线性空 间的一组基
kernel	计算矩阵的核
concat, stack	把两个矩阵左右合并或上下合并

如何使用这本书

如果你对 MAPLE 知道得很少或根本没什么经验,你最好仔细地读一下前五章,当然也可以先跳过一些章节.如果你在阅读的同时在计算机上尝试使用 MAPLE 就更好了;尝试运行一些例子,积累一点经验,但如果在什么地方卡住了,你不必耽误时间.因为随着进一步学习,你可能很快便能弄懂那个问题.

如果你已对 MAPLE 有些经验,前五章可以帮助你更有效地使用 MAPLE,还有助于你进一步理解它的基本思想.

如果你已经读了前五章,或早已了解 MAPLE 的基本思想,你就可以往下学习与你的特定兴趣相关的章节.其中一些章节是在数学领域给予指导,另一些章节则是在符号操作领域.每章都有一个前言,在前言中你可以了解在该章能学到什么.

当你使用 MAPLE 并遇到问题时,你可以借助目录获得帮助.要是你没读本书前面几章也没关系;许多交叉的参考条目使得必要时很容易找到另外的信息.此外,书中还有一个错误信息列表,为你显示并解释本书示例中的错误信息.

如果你不是每天使用 MAPLE,或许会忘记 MAPLE 的指令.因此最常用的命令都列入本书开头的简要命令查询一览表中.你也可以自己扩展那个表.

一个好的习惯是:当你使用计算机时不要停止思考,即使你使用像 MAPLE 这样的强有力的工具.在这本书的许多例子里,使用 MAPLE 时你会看到普通的数学常识将怎样适时地给你帮助.

使用计算机系统做计算的一个重要方面就是计算结果的可靠性问题.本书在相应的地方对这个问题做了讨论,且介绍了核对与检验结果的方法.可是,即便某处的一个计算是可靠的,都没人敢确保不会出现未知的故障.检验与核对结果一般来讲是个好习惯,更不用说使用这样一个复杂的系统就更得有这种好习惯.

本书基于 MAPLE V Release 5, 凡与 Release 3 和 Release 4 相关的区别均予以指明. MAPLE 的输出是按 MAPLE 的视窗版本生成的输出风格打印的 (如 Microsoft Windows, Mac, X 终端和工作站), 选择字体和断行也都如此.

有关 MAPLE 的其他信息

你可以从位于

<http://www.maplesoft.com/home.html>

的 MAPLE 信息服务器 (MAPLE Info Server) 中获得有关 MAPLE 的所有信息、附加软件 (MAPLE Share Library 提供的新软件包)、演示及 MAPLE 的支持信息. 在同一个网址上你还可以找到有关使用 MAPLE 的书目.

你还可以下载补丁程序; 这与大多数软件是一样的, 新的版本一般总是有缺陷, 不过这些缺陷总会被开发人员找到; 从这个网址上可以得到修复缺陷的补丁程序.

有关 MAPLE 的讨论一直在 MAPLE 的用户组中通过电子邮件进行着, 人们在网上问些简单或复杂的问题, 对 MAPLE 的一些问题进行讨论, 而另一些人则回答. 你发一个电子邮件就可以加入该组, 地址为:

majordomo@daisy.uwaterloo.ca,

正文中应含有一行:

`subscribe maple-list [<address>].`

致 谢

首先我要感谢 MAPLE 的全体开发人员,也要感谢开发小组人员之外的、对数学算法及 MAPLE 资源作出贡献的那些人们,是他们创造了这个在数学计算方面功能如此强大的工具.

我还要感谢 Catholic University Nijmegen 数学学院,给我提供了撰写此书的机会和必要的支持,以及感谢计算机系提供给我耐心和细致的帮助.

这本书的写成,还应归功于那些我把他们引向 MAPLE 的人们,在教他们使用 MAPLE 的过程中通过他们问我问题,跟我反映他们的困难,极大地帮助了我,使我受益匪浅.此外,MAPLE 用户小组研讨过的问题和思想是使我获益的另一个源泉,我对他们所作的贡献也感谢不尽.我还要感谢计算机代数专家 A.Heck 博士,我们经常就 MAPLE 及如何讲授它的使用进行富有建设性的争论.感谢 Nijmegen 大学的 A.H.M.Levelt 教授鼓励我写这本书.

感谢 Nijmegen 大学的 J.M.G.Ingelaat 博士, A.G.M.Janner 教授, R.M.Corless, D.Redfern, B.Barber 和所有读过这本书,并对它作过有益评价的人们.

本书用 $\text{T}_{\text{E}}\text{X}$ 写出.我得感谢 V.Eijkhout 博士,他把 $\text{T}_{\text{E}}\text{X}$ 介绍给我,并在一些问题上帮助我.还要感谢 Yunliang Yu,是他创造了功能强大的 MAPLE 软件包,可以把 MAPLE 表达式转换成 $\text{T}_{\text{E}}\text{X}$;我在本书中就使用了这个软件包,用于模拟 MAPLE 命令的屏幕输出.

最后,我还要感谢所有鼓励我写这本书的朋友们.

Ernic Kamberich

尽管有这些努力及帮助, 本书肯定还有需要改善的地方. 所以, 如果您还有什么建议、更正或有关这本书的其他评论, 我将非常欢迎您的批评指正. 联络地址如下:

Dr. B.N.P. Kamerich
Fac. of Math. and Comp. Sci.
Catholic University Nijmegen
Toernooiveld 1
6525 ED Nijmegen
The Netherlands

Fax: 0031-243652140

email: ernic@sci.kun.nl