第四章 方程求解

1 代数方程(组)求解

1.1 常用求解工具—solve

求解代数方程或代数方程组，使用 Maple 中的 **solve** 函数. 求解关于 x 的方程 eqn=0 的命令格式为：

solve(eqn, x);

求解关于变量组 vars 的方程组 eqns 的命令为：

solve(eqns, vars);

```
> eqn:=(x^2+x+2)*(x-1);
```

$$eqn := (x^2 + x + 2)(x - 1)$$

```
> solve(eqn,x);
```

$$1, -\frac{1}{2} + \frac{1}{2} I\sqrt{7}, -\frac{1}{2} - \frac{1}{2} I\sqrt{7}$$

当然, solve 也可以求解含有未知参数的方程：

```
> eqn:=2*x^2-5*a*x=1;
```

$$eqn := 2x^2 - 5ax = 1$$

```
> solve(eqn,x);
```

$$\frac{5}{4}a + \frac{1}{4}\sqrt{25a^2 + 8}, \frac{5}{4}a - \frac{1}{4}\sqrt{25a^2 + 8}$$

solve 函数的第一个参数是有待求解的方程或方程的集合，当然也可以是单个表达式或者表达式的集合，如下例：

```
> solve(a+ln(x-3)-ln(x),x);
```

$$3\frac{e^a}{-1 + e^a}$$

对于第二个参数, Maple 的标准形式是未知变量或者变量集合，当其被省略时，函数 **indets** 自动获取未知变量. 但当方程中含有参数时，则会出现一些意想不到的情况：

```
> solve(a+ln(x-3)-ln(x));
```

$$\{x = x,\ a = -\ln(x-3) + \ln(x)\}$$

很多情况下，我们知道一类方程或方程组有解，但却没有解决这类方程的一般解法，或者说没有解析解．比如，一般的五次或五次以上的多项式，其解不能写成解析表达式．Maple 具备用所有一般算法尝试所遇到的问题，在找不到解的时候，Maple 会用 RootOf 给出形式解．

```
> x^7-2*x^6-4*x^5-x^3+x^2+6*x+4;
```

$$x^7 - 2x^6 - 4x^5 - x^3 + x^2 + 6x + 4$$

```
> solve(%);
```

$$1+\sqrt{5},\ 1-\sqrt{5},\ \text{RootOf}(\_Z^5 - \_Z - 1,\ index = 1),\ \text{RootOf}(\_Z^5 - \_Z - 1,\ index = 2),\ \text{RootOf}(\_Z^5 - \_Z - 1,\ index = 3),\ \text{RootOf}(\_Z^5 - \_Z - 1,\ index = 4),\ \text{RootOf}(\_Z^5 - \_Z - 1,\ index = 5)$$

```
> solve(cos(x)=x,x);
```

$$\text{RootOf}(\_Z - \cos(\_Z))$$

对于方程组解的个数可用 **nops** 命令获得，如：

```
> eqns:={seq(x[i]^2=x[i],i=1..7)};
```

$$eqns := \{x_1^{\,2} = x_1,\ x_2^{\,2} = x_2,\ x_3^{\,2} = x_3,\ x_4^{\,2} = x_4,\ x_5^{\,2} = x_5,\ x_6^{\,2} = x_6,\ x_7^{\,2} = x_7\}$$

```
> nops({solve(eqns)});
```

$$128$$

但是，有时候，Maple 甚至对一些“显而易见”的结果置之不理，如：

```
> solve(sin(x)=3*x/Pi,x);
```

$$\text{RootOf}(3\ \_Z - \sin(\_Z)\,\pi)$$

此方程的解为$\pm\frac{\pi}{6}$，0，但 Maple 却对这个超越方程无能为力，即便使用 **allvalues** 求解也只有下述结果：

```
> allvalues(%);
```

$$\text{RootOf}(3\ \_Z - \sin(\_Z)\,\pi,\ 0.)$$

另外一个问题是，Maple 在求解方程之前,会对所有的方程或表达式进行化简，而不管表达式的类型，由此而产生一些低级的错误：

```
> (x-1)^2/(x^2-1);
```

$$\frac{(x-1)^2}{x^2-1}$$

```
> solve(%);
```

$$1$$

但是，大量实验表明，**solve** 的确是一个实用的方程求解工具，但是也不可盲目相信它给出的一切结果，特别是对于非线性方程而言，对于给出的结果需要加以验证.

下面通过几个例子说明在 Maple 中非线性方程组的求解问题.

例：求解方程组：$\begin{cases} x^2+y^2=25 \\ x^2-9=y \end{cases}$

```
> eqns:={x^2+y^2=25,y=x^2-5};
```

$$eqns := \{y = x^2 - 5, x^2 + y^2 = 25\}$$

```
> vars:={x,y};
```

$$vars := \{x, y\}$$

```
> solve(eqns,vars);
```

$$\{x = 0, y = -5\}, \{x = 0, y = -5\}, \{y = 4, x = 3\}, \{y = 4, x = -3\}$$

也可用下面的语句一步求出：

```
> solve({x^2+y^2=25,y=x^2-5},{x,y});
```

$$\{x = 0, y = -5\}, \{x = 0, y = -5\}, \{y = 4, x = 3\}, \{y = 4, x = -3\}$$

这个问题非常简单，但通常遇到的非线性问题却不是这么简单，例如要求解方程组：$x^2+y^2=1, \sqrt{x+y}=x-y$

```
> eqns:={x^2+y^2=1,sqrt(x+y)=x-y};
  vars:={x,y};
```

$$eqns := \{x^2 + y^2 = 1, \sqrt{x+y} = x - y\}$$

$$vars := \{x, y\}$$

```
> sols:=solve(eqns,vars);
```

$$sols := \{y = \text{RootOf}(2\_Z^2 + 4\_Z + 3, -1.000000000 - .7071067812\ I),$$
$$x = -\text{RootOf}(2\_Z^2 + 4\_Z + 3, -1.000000000 - .7071067812\ I) - 2\}, \{x = 1, y = 0\}$$

可以看出，方程解的形式是以集合的序列给出的，序列中的每一个集合是方程的一组解，这样就很利于我们用 **subs** 把解代入原方程组进行检验：

```
> subs(sols[2],eqns);
```

$$\{1 = 1\}$$

```
> sols2:=allvalues(sols[1]);
```

$$sols2 := \{x = -1 + \frac{1}{2} I\sqrt{2}, y = -1 - \frac{1}{2} I\sqrt{2}\}$$

```
> simplify(subs(sols2,eqns));
```

$$\{I\sqrt{2} = I\sqrt{2}, 1 = 1\}$$

1.2 其他求解工具

1.2.1 数值求解

对于求代数方程的数值解问题, Maple 提供了函数 **fsolve**, **fsolve** 的使用方法和 **solve** 很相似:

fsolve(eqns, vars, options);

其中, eqns 表示一个方程、方程组或者一个程序, vars 表示一个未知量或者未知量集合, options 控制解的参数(诸如：complex: 复根; maxsols=n：只找到 n 阶最小根; intervals：在给定闭区间内求根, 等).

```
> fsolve(x^5-x+1,x);
```

-1.167303978

```
> fsolve(x^5-x+1,x,complex);
```

-1.167303978 , -.1812324445 − 1.083954101 *I*, -.1812324445 + 1.083954101 *I*, .7648844336 − .3524715460 *I*, .7648844336 + .3524715460 *I*

```
> fsolve(x^3-3*x+1,x,0..1);
```

.3472963553

对于多项式方程, **fsolve** 在默认情况下以给出所有的实数解, 如果附加参数 **complex**, 就可以给出所有的解. 但对于更一般的其他形式的方程, **fsolve** 却往往只满足于得到一个解：

```
> eqn:=sin(x)=x/2;
```

$$eqn := \sin(x) = \frac{1}{2}x$$

```
> fsolve(eqn);
```

0.

```
> fsolve(eqn,x,0.1..infinity);
```

1.895494267

```
> fsolve(eqn,x,-infinity..-0.1);
```

-1.895494267

函数 **fsolve** 主要基于两个算法, 通常使用牛顿法, 如果牛顿法无效, 它就改而使用切线法. 为了使 **fsolve** 可以求得所有的实根, 我们通常需要确定这些根所在的区间. 对于单变量多项式, 函数 **realroot** 可以获得多项式的所有实根所在的区间.

```
> 4+6*x+x^2-x^3-4*x^5-2*x^6+x^7;
```

$$4 + 6x + x^2 - x^3 - 4x^5 - 2x^6 + x^7$$

```
> realroot(%);
```

$$[[0, 2], [2, 4], [-2, -1]]$$

函数 **realroot** 还有一个可选参数，它是用来限制区间的最大长度的，为了保证使用数值求解方法时收敛，我们可以用它限制区间的最大长度：

```
> realroot(%%,1/1000);
```

$$\left[\left[\frac{1195}{1024}, \frac{299}{256}\right], \left[\frac{3313}{1024}, \frac{1657}{512}\right], \left[\frac{-633}{512}, \frac{-1265}{1024}\right]\right]$$

求解方程或方程组的整数解时使用函数 **isolve**，它常常被用来求解不定方程. 例如著名的“百钱买百鸡”问题♣的求解过程为：

```
> isolve({x+y+z=100,5*x+3*y+z/3=100});
```

$$\{z = 75 + 3\ \_Z1, x = 4\ \_Z1, y = 25 - 7\ \_Z1\}$$

据此可得满足该问题的三组解为：

$$\{x, y, z\}=\{4, 18, 78\}, \{x, y, z\}=\{8, 11, 81\}, \{x, y, z\}=\{12, 4, 84\}$$

1.2.2 整数环中的方程(组)求解

利用 Maple 中的函数 **msolve(eqns, vars, n)**，可以在模 ***n*** 的整数环中求解方程(组)eqns.

例：在 Z_7 中求解 **Pell** 方程 $y^7 = x^3 - 28$

```
> msolve(y^7=x^3-28,7);
```

$$\{x = 3, y = 6\}, \{x = 4, y = 1\}, \{y = 0, x = 0\}, \{x = 1, y = 1\}, \{y = 6, x = 6\}, \{x = 2, y = 1\}, \{y = 6, x = 5\}$$

再如下例：

```
> msolve(y^4=x^3+32,5);
```

$$\{x = 2, y = 0\}, \{x = 4, y = 1\}, \{x = 4, y = 2\}, \{x = 4, y = 3\}, \{x = 4, y = 4\}$$

1.2.3 递归方程的求解

在 Maple 中，可以求解有限差分方程(也称递归方程)，所需调用的函数是 **rsolve**，该函数使用的是一些比较通用的方法，例如产生函数法、z 变换法以及一些基于变量替换和特征方程的方法. 作为例子，求解 Fibonacci 多项式:

```
> eq:=f(n)=f(n-1)+2*f(n-2);
```

$$eq := \mathrm{f}(n) = \mathrm{f}(n-1) + 2\,\mathrm{f}(n-2)$$

```
> rsolve({eq,f(0)=1,f(1)=1},f(n));
```

$$\frac{1}{3}(-1)^n + \frac{2}{3}2^n$$

当然，并不是所有的递归形式的函数方程的解可以写成解析形式，如果不能，Maple 将保留原来的调用形式. 此时，可用 **asympt** 函数获得它的渐进表达式，也就是 1/n 的级数解. 例如，对于一个具有超越形式的递归函数方程，仍然可以得到解的渐进形式：

♣ **百钱买百鸡问题**：用 100 元钱买 100 只鸡，大公鸡 5 元钱 1 只，大母鸡 3 元钱 1 只，小鸡 1 元钱 3 只，问如何买法？

```
> rsolve(u(n+1)=ln(u(n)+1),u(n));
```

$$\text{rsolve}(\text{u}(n+1)=\ln(\text{u}(n)+1),\text{u}(n))$$

```
> asympt(%,n,5);
```

$$2\,\frac{1}{n}+\frac{\_C+\frac{2}{3}\ln(n)}{n^2}+\frac{\frac{1}{9}-\frac{1}{3}\_C+\frac{1}{2}\_C^2-\left(-\frac{2}{3}\_C+\frac{2}{9}\right)\ln(n)+\frac{2}{9}\ln(n)^2}{n^3}+\text{O}\left(\frac{1}{n^4}\right)$$

1.2.4 不等式(组)求解

求解一元不等式方程(组)使用命令**solve**：

```
> solve((x-1)*(x-2)*(x-3)<0,x);
```

$$\text{RealRange}(-\infty,\text{Open}(1)),\text{RealRange}(\text{Open}(2),\text{Open}(3))$$

```
> solve((x-1+a)*(x-2+a)*(x-3+a) < 0, {x});
```

$$\{x<1-a\},\{2-a<x,x<3-a\}$$

```
> solve(exp(x)>x+1);
```

$$\text{RealRange}(-\infty,\text{Open}(0)),\text{RealRange}(\text{Open}(0),\infty)$$

```
> solve({x^2*y^2=0,x-y=1,x<>0});
```

$$\{y=0,x=1\},\{y=0,x=1\}$$

对于由不等式方程组约束的最优问题的求解使用“线性规则”工具包**simplex**:

```
> with(simplex):
> cnsts:={3*x+4*y-3*z<=23, 5*x-4*y-3*z<=10,7*x+4*y+11*z<=30};
```

$$cnsts := \{3x+4y-3z\le 23,\ 5x-4y-3z\le 10,\ 7x+4y+11z\le 30\}$$

```
> obj:=-x+y+2*z;
```

$$obj := -x+y+2z$$

```
> maximize(obj,cnsts union {x>=0,y>=0,z>=0});
```

$$\{z=\frac{1}{2},y=\frac{49}{8},x=0\}$$

2 常微分方程求解

微分方程求解是数学研究与应用的一个重点和难点. Maple 能够显式或隐式地解析地求解许多微分方程求解. 在常微分方程求解器 dsolve 中使用了一些传统的技术例如 laplace 变换和积分因子法等, 函数 pdesolve 则使用诸如特征根法等经典方法求解偏微分方程. 此外, Maple 还提供了可作摄动解的所有工具, 例如 Poincare-Lindstedt 法和高阶多重尺度法.

帮助处理常微分方程(组)的各类函数存于 Detools 软件包中, 函数种类主要有：可视化类的函数, 处理宠加莱动态系统的函数, 调整微分方程的函数, 处理积分因子、李对称

法和常微分方程分类的函数，微分算子的函数，利用可积性与微分消去的方法简化微分方程的函数，以及构造封闭解的函数等．更重要的是其提供的强大的图形绘制命令Deplot 能够帮助我们解决一些较为复杂的问题.

2.1 常微分方程的解析解

求解常微分方程最简单的方法是利用求解函数 **dsolve**．命令格式为：

dsolve(ODE);

dsolve(ODE, y(x), extra_args);

dsolve({ODE, ICs}, y(x), extra_args);

dsolve({sysODE, ICs}, {funcs}, extra_args);

其中，ODE—常微分方程，y(x)—单变量的任意变量函数，Ics—初始条件，{sysODE}—ODE 方程组的集合，{funcs}—变量函数的集合，extra_args—依赖于要求解的问题类型.

例如，对于一阶常微分方程 $xy' = y\ln(xy) - y$ 可用 **dsolve** 直接求得解析解：

```
> ODE:=x*diff(y(x),x)=y(x)*ln(x*y(x))-y(x);
```

$$ODE := x\left(\frac{\partial}{\partial x}\mathrm{y}(x)\right) = \mathrm{y}(x)\ln(x\,\mathrm{y}(x)) - \mathrm{y}(x)$$

```
> dsolve(ODE,y(x));
```

$$\mathrm{y}(x) = \frac{\mathrm{e}^{\left(\frac{x}{\_C1}\right)}}{x}$$

可以看出，**dsolve** 的第一个参数是待求的微分方程，第二个参数是未知函数．需要注意的是，无论在方程中还是作为第二个参数，未知函数必须用函数的形式给出(即：必须加括号，并在其中明确自变量)，这一规定是必须的，否则 Maple 将无法区分方程中的函数、自变量和参变量，这一点和我们平时的书写习惯不一致．为了使其与我们的习惯一致，可用 **alias** 将函数用别称表示：

```
> alias(y=y(x));
> ODE:=x*diff(y,x)=y*ln(x*y)-y;
```

$$ODE := x\left(\frac{\partial}{\partial x}y\right) = y\ln(x\,y) - y$$

```
> dsolve(ODE,y);
```

$$y = \frac{\mathrm{e}^{\left(\frac{x}{\_C1}\right)}}{x}$$

函数 **dsolve** 给出的是微分方程的通解，其中的任意常数是用下划线起始的内部变量表示的.

在 Maple 中，微分方程的解是很容易验证的，只需要将解代入到原方程并化简就可以了.

```
> subs(%,ODE);
```

$$x\left(\frac{\partial}{\partial x}\frac{e^{\left(\frac{x}{\_C1}\right)}}{x}\right)=\frac{e^{\left(\frac{x}{\_C1}\right)}\ln\left(e^{\left(\frac{x}{\_C1}\right)}\right)}{x}-\frac{e^{\left(\frac{x}{\_C1}\right)}}{x}$$

```
> assume(x,real): assume(_C1,real):
> simplify(%);
```

$$-\frac{e^{\left(\frac{x\sim}{\_C1\sim}\right)}(-x\sim+\_C1\sim)}{x\sim\ \_C1\sim}=-\frac{e^{\left(\frac{x\sim}{\_C1\sim}\right)}(-x\sim+\_C1\sim)}{x\sim\ \_C1\sim}$$

```
> evalb(%);
```

$$true$$

evalb 函数的目的是对一个包含关系型运算符的表达式使用三值逻辑系统求值，返回的值是 true, false 和 FAIL. 如果无法求值，则返回一个未求值的表达式. 通常包含关系型运算符“=, <>, <, <=, >, >=”的表达式在 Maple 中看作是代数方程或者不等式. 然而，作为参数传递给 evalb 或者出现在 if 或 while 语句的逻辑表达式中时，它们会被求值为 true 或 false. 值得注意的是, evalb 不化简表达式，因此在使用 evalb 之前应将表达式化简，否则可能会出错.

再看下面常微分方程的求解： $y'=\sqrt{y^2+1}$

```
> alias(y=y(x)):
> ODE:=diff(y,x)=sqrt(y^2+1);
```

$$ODE:=\frac{\partial}{\partial x}y=\sqrt{y^2+1}$$

```
> dsolve(ODE,y);
```

$$y=\sinh(x+\_C1)$$

函数 **dsolve** 对于求解含有未知参变量的常微分方程也完全可以胜任:

```
> alias(y=y(x)):
> ODE:=diff(y,x)=-y/sqrt(a^2-y^2);
```

$$ODE:=\frac{\partial}{\partial x}y=-\frac{y}{\sqrt{a^2-y^2}}$$

```
> sol:=dsolve(ODE,y);
```

$$sol:=x+\sqrt{a^2-y^2}-\frac{a^2\ln\left(\frac{2a^2+2\sqrt{a^2}\sqrt{a^2-y^2}}{y}\right)}{\sqrt{a^2}}+\_C1=0$$

由此可见，对于不能表示成显式结果的微分方程解，Maple 尽可能将结果表示成隐

式解. 另外, 对于平凡解 y=0 常常忽略, 这一点应该引起注意.

dsolve 对于求解微分方程初值问题也十分方便的:

```
> ODE:=diff(u(t),t$2)+omega^2*u(t)=0;
```

$$ODE := \left(\frac{\partial^2}{\partial t^2}\,u(t)\right) + \omega^2\,u(t) = 0$$

```
> dsolve({ODE,u(0)=u0,D(u)(0)=v0},u(t));
```

$$u(t) = \frac{v0\,\sin(\omega\,t)}{\omega} + u0\,\cos(\omega\,t)$$

2.2 利用积分变换求解微分方程

对于特殊的微分方程, 我们还可以指定 **dsolve** 利用积分变换方法求解, 只需要在 **dsolve** 中加入可选参数 **method=transform** 即可. 其中 **transform** 是积分变换, 可以是 **laplace**、**fourier**、**fouriercos** 或者 **fouriersin** 变换.

作为例子, 我们来看一个具有阻尼的振子在阶跃冲击(Heaviside 函数)下的响应:

```
> ODE:=diff(u(t),t$2)+2*d*omega*diff(u(t),t)+omega^2*u(t)=Heaviside(t);
```

$$ODE := \left(\frac{\partial^2}{\partial t^2}\,u(t)\right) + 2\,d\,\omega\left(\frac{\partial}{\partial t}\,u(t)\right) + \omega^2\,u(t) = \text{Heaviside}(t)$$

```
> initvals:=(u(0)=u[0],D(u)(0)=v[0]);
```

$$initvals := u(0) = u_0,\ D(u)(0) = v_0$$

```
> solution:=dsolve({ODE,initvals},u(t),method=laplace);
```

$$solution := u(t) = \frac{\frac{1}{\omega} + e^{(-t\,d\,\omega)}\left(\frac{(\omega^2 u_0 - 1)\cosh(t\sqrt{d^2\omega^2-\omega^2})}{\omega} + \frac{(\omega\,v_0 + d\,\omega^2 u_0 - d)\sinh(t\sqrt{d^2\omega^2-\omega^2})}{\sqrt{d^2\omega^2-\omega^2}}\right)}{\omega}$$

Maple 给出了问题的通解, 但没有区分自由振动(d=0)、欠阻尼(0<d<1)、临界阻尼(d=1)和过阻尼(d>1)的情况. 下面加以区分求解:

```
> assume(omega>0):
> simplify(subs(d=0,solution));
```

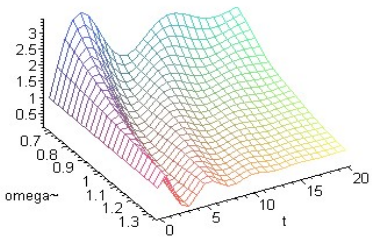
$$u(t) = \frac{1 + \cos(t\,\omega)\,\omega^2 u_0 - \cos(t\,\omega) + v_0\,\sin(t\,\omega)\,\omega}{\omega^2}$$

```
> K:=subs(d=1/5,u[0]=1,v[0]=1,solution);
```

$$K := u(t) = \frac{\frac{1}{\omega} + e^{(-1/5\,t\,\omega)}\left(\frac{(\omega^2-1)\cosh\left(t\sqrt{-\frac{24}{25}\omega^2}\right)}{\omega} + \frac{\left(\omega + \frac{1}{5}\omega^2 - \frac{1}{5}\right)\sinh\left(t\sqrt{-\frac{24}{25}\omega^2}\right)}{\sqrt{-\frac{24}{25}\omega^2}}\right)}{\omega}$$

```
> with(plots):
```

```
> plot3d(rhs(%%),omega=2/3..4/3,t=0..20,style=hidden,orientation=[-30,45],axes=framed);
```



对于 d=1 的情况，可可用下式获得结果：

```
> limit(rhs(solution),d=1);
```

$$\frac{(\omega^2 u_0 + \omega^2 v_0 t - 1 + \omega^3 u_0 t - t\omega + e^{(t\omega)})\, e^{(-t\omega)}}{\omega^2}$$

再如下例：

```
> diff(u(t),t$2)+3*diff(u(t),t)+2*u(t)=exp(-abs(t));
```

$$\left(\frac{\partial^2}{\partial t^2}\,u(t)\right) + 3\left(\frac{\partial}{\partial t}\,u(t)\right) + 2\,u(t) = e^{(-|t|)}$$

```
> dsolve(%,u(t),method=fourier);
```

$$u(t) = \frac{2}{3}e^{(-2t)}\,\text{Heaviside}(t) + \frac{1}{6}e^{t}\,\text{Heaviside}(-t) + e^{(-t)}\,t\,\text{Heaviside}(t) - \frac{1}{2}e^{(-t)}\,\text{Heaviside}(t)$$

2.3 常微分方程组的求解

函数 dsolve 不仅可以用来求解单个常微分方程，也可以求解联立的常微分方程组. 特别是对于线性微分方程组，由于数学上具有成熟的理论，Maple 的求解也是得心应手. 其命令格式为：

dsolve({eqn1, eqn2, …, ini_conds}, {vars});

其中，**ini_conds** 是初始条件.

```
> eqn1:={diff(x(t),t)=x(t)+y(t),diff(y(t),t)=y(t)-x(t)};
```

$$eqn1 := \left\{\frac{\partial}{\partial t}\,x(t) = x(t) + y(t),\ \frac{\partial}{\partial t}\,y(t) = y(t) - x(t)\right\}$$

```
> dsolve(eqn1,{x(t),y(t)});
```

$$\{x(t) = e^{t}\,(\_C1\,\sin(t) + \_C2\,\cos(t)),\ y(t) = e^{t}\,(\_C1\,\cos(t) - \_C2\,\sin(t))\}$$

```
> eqn2:=2*diff(x(t),t$2)+2*x(t)+y(t)=2*t;
```

$$eqn2 := 2\left(\frac{\partial^2}{\partial t^2}\,x(t)\right) + 2\,x(t) + y(t) = 2\,t$$

```
> eqn3:=diff(y(t),t$2)+2*x(t)+y(t)=t^2+1;
```

$$eqn3 := \left(\frac{\partial^2}{\partial t^2} y(t)\right) + 2\,x(t) + y(t) = t^2 + 1$$

> **dsolve({eqn2, qn3, x(0)=0, D(x)(0)=1, y(0)=0, D(y)(0)=0}, {x(t),y(t)});**

$$\{x(t) = \frac{1}{8}\sin(\sqrt{2}\,t)\sqrt{2} + \frac{1}{12}t^3 - \frac{1}{48}t^4 + \frac{3}{4}t,$$

$$y(t) = \frac{1}{4}\sin(\sqrt{2}\,t)\sqrt{2} - \frac{1}{2}t + \frac{1}{2}t^2 - \frac{1}{6}t^3 + \frac{1}{24}t^4\}$$

2.4 常微分方程的级数解法

1) 泰勒级数解法

当一个常微分方程的解析解难以求得时，可以用 Maple 求得方程解的级数近似，这在大多数情况下是一种非常好的方法. 级数解法是一种半解析半数值的方法.

泰勒级数法的使用命令为：

dsolve({ODE,Ics}, y(x), **'series');** 或 dsolve({ODE,Ics}, y(x), **'type=series');**

下面求解物理摆的大幅振动方程：$l\theta = -g\sin\theta$，其中 l 是摆长，θ 是摆角，g 是重力加速度.

> **ODE:=l*diff(theta(t),t$2)=-g*sin(theta(t));**

$$ODE := l\left(\frac{\partial^2}{\partial t^2}\theta(t)\right) = -g\sin(\theta(t))$$

> **initvals:=theta(0)=0,D(theta)(0)=v[0]/l;**

$$initvals := \theta(0) = 0, D(\theta)(0) = \frac{v_0}{l}$$

> **sol:=dsolve({ODE,initvals},theta(t),type=series);**

$$sol := \theta(t) = \frac{v_0}{l}t - \frac{1}{6}\frac{g\,v_0}{l^2}t^3 + \frac{1}{120}\frac{g\,v_0\,(v_0^2 + g\,l)}{l^4}t^5 + O(t^6)$$

> **Order:=11:**

> **sol:=dsolve({ODE,initvals},theta(t),type=series);**

$$sol := \theta(t) = \frac{v_0}{l}t - \frac{1}{6}\frac{g\,v_0}{l^2}t^3 + \frac{1}{120}\frac{g\,v_0\,(v_0^2 + g\,l)}{l^4}t^5 - \frac{1}{5040}\frac{g\,v_0\,(11\,g\,l\,v_0^2 + g^2\,l^2 + v_0^4)}{l^6}t^7 + \frac{1}{362880}\frac{g\,v_0\,(57\,g\,v_0^4\,l + 102\,g^2\,v_0^2\,l^2 + g^3\,l^3 + v_0^6)}{l^8}t^9 + O(t^{11})$$

2) 幂级数解法

对于一个符号代数系统来说，幂级数是必不可少的微分方程求解工具. 幂级数求解函数 **powsolve** 存于工具包 **powseries** 中. 但是，这一求解函数的使用范围很有限，它只可以用来求解多项式系数的线性常微分方程或方程组,其求解命令为：powseries[function](prep)或直接载入软件包后用 function(prep), prep 为求解的线性微分方程及其初值.

例：求解： $xy' + y'' + 4x^2 y = 0$

```
> ODE:=x*diff(y(x),x$2)+diff(y(x),x)+4*x^2*y(x)=0;
```

$$ODE := x\left(\frac{\partial^2}{\partial x^2}\mathrm{y}(x)\right) + \left(\frac{\partial}{\partial x}\mathrm{y}(x)\right) + 4x^2\,\mathrm{y}(x) = 0$$

```
> dsolve(ODE,y(x));
```

$$\mathrm{y}(x) = \_C1\ \mathrm{BesselJ}\left(0, \frac{4}{3}x^{(3/2)}\right) + \_C2\ \mathrm{BesselY}\left(0, \frac{4}{3}x^{(3/2)}\right)$$

```
> initvals:=y(0)=y0,D(y)(0)=0;
```

$$initvals := \mathrm{y}(0) = y0,\ \mathrm{D}(y)(0) = 0$$

```
> with(powseries):
> sol:=powsolve({ODE,initvals});
```

$$sol := \mathbf{proc}\ (powparm)\ \ldots\ \mathbf{end\ proc}$$

```
> tpsform(sol,x,16);
```

$$y0 - \frac{4}{9}y0\,x^3 + \frac{4}{81}y0\,x^6 - \frac{16}{6561}y0\,x^9 + \frac{4}{59049}y0\,x^{12} - \frac{16}{13286025}y0\,x^{15} + \mathrm{O}(x^{16})$$

也可以用 **powsolve** 给出的函数直接获得用递归形式定义的幂级数系数，不过参数必须用***_k***，这是 **powsolve** 使用的临时变量.

```
> sol(_k);
```

$$-4\,\frac{\mathrm{a}(\_k - 3)}{\_k^2}$$

例：求解一维谐振子的解： $y'' + (\varepsilon - x^2)y = 0$

```
> alias(y=y(x)):
> ODE:=diff(y,x$2)+(epsilon-x^2)*y=0;
```

$$ODE := \left(\frac{\partial^2}{\partial x^2}y\right) + (\varepsilon - x^2)\,y = 0$$

```
> H:=powsolve(ODE);
```

$$H := \mathbf{proc}\ (powparm)\ \ldots\ \mathbf{end\ proc}$$

```
> tpsform(H,x,8);
```

$$C0 + C1\,x - \frac{1}{2}\varepsilon\,C0\,x^2 - \frac{1}{6}\varepsilon\,C1\,x^3 + \left(\frac{1}{24}\varepsilon^2\,C0 + \frac{1}{12}C0\right)x^4 + \left(\frac{1}{120}\varepsilon^2\,C1 + \frac{1}{20}C1\right)x^5 + \left(-\frac{1}{30}\varepsilon\left(\frac{1}{24}\varepsilon^2\,C0 + \frac{1}{12}C0\right) - \frac{1}{60}\varepsilon\,C0\right)x^6 + \left(-\frac{1}{42}\varepsilon\left(\frac{1}{120}\varepsilon^2\,C1 + \frac{1}{20}C1\right) - \frac{1}{252}\varepsilon\,C1\right)x^7 + \mathrm{O}(x^8)$$

```
> H(_k);
```

$$\_\frac{\varepsilon\, a(\_k-2)-a(\_k-4)}{\_k\,(\_k-1)}$$

2.5 常微分方程的数值解法

在对微分方程的解析解失效后，可以求助于数值方法求解微分方程. 数值求解的好处是只要微分方程的条件足够多时一般都可求得结果，然而所得结果是否正确则必须依赖相关数学基础加以判断. 调用函数 **dsolve** 求常微分方程初值问题的数值解时需加入参数 **type=numeric**.

另一方面，常微分方程初值问题数值求解还可以选择算法，加入参数“method=方法参数”即可，方法参数主要有：

rkf45：4~5 阶变步长 Runge-Kutta-Fehlberg 法

dverk78：7~8 阶变步长 Runge-Kutta-Fehlberg 法

classical：经典方法，包括向前欧拉法，改进欧拉法，2、3、4 阶龙格库塔法，Sdams-Bashford 方法等

gear：吉尔单步法

mgear：吉尔多步法

2.5.1 变步长龙格库塔法

下面用 4~5 阶 Runge-Kutta-Fehlberg 法求解 van der Pol 方程：

$$\begin{cases} y''-(1-y^2)y'+y=0 \\ y(0)=0, y'(0)=-0.1 \end{cases}$$

```
> ODE:=diff(y(t),t$2)-(1-y(t)^2)*diff(y(t),t)+y(t)=0;
```

$$ODE := \left(\frac{\partial^2}{\partial t^2} y(t)\right) - (1 - y(t)^2)\left(\frac{\partial}{\partial t} y(t)\right) + y(t) = 0$$

```
> initvals:=y(0)=0,D(y)(0)=-0.1;
```

$$initvals := y(0) = 0, D(y)(0) = -.1$$

```
> F:=dsolve({ODE,initvals},y(t),type=numeric);
```

$$F := \mathbf{proc}\ (rkf45\_x)\ \ldots\ \mathbf{end\ proc}$$

此时，函数返回的是一个函数，可以在给定的数值点上对它求值：

```
> F(0);
```

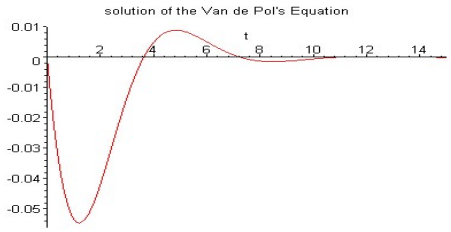
$$\left[t = 0., y(t) = 0., \frac{\partial}{\partial t} y(t) = -.1\right]$$

```
> F(1);
```

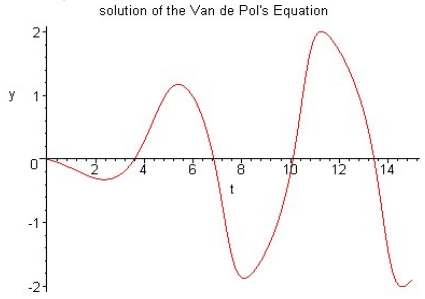
$$\left[t = 1., y(t) = -.144768589749425608, \frac{\partial}{\partial t} y(t) = -.178104066128215944\right]$$

可以看到, F 给出的是一个包括 t、y(t)、D(y)(t)在内的有序表，它对于每一个时间点可以给出一组数值表达式. 有序表的每一项是一个等式，可对其作图描述.

```
> plot('rhs(F(t)[2])', t=0..15, title="solution of the Van de Pol's Equation");
```




```
> plots[odeplot](F,[t,y(t)],0..15,title="solution of the Van de Pol's Equation");
```



2.5.2 吉尔法求解刚性方程

在科学和工程计算中，常常会遇到这样一类常微分方程问题，它可以表示成方程组：$y' = f(t,y), y(t_0) = y_0$，称其为刚性方程，其解的分量数量相差很大，分量的变化速度也相差很大. 如果用常规方法求解，为了使变量有足够高的精度，必须取很小的步长，而为了使慢变分量达到近似的稳态解，则需要很长的时间，这样用小步长大时间跨度的计算，必定造成庞大的计算量，而且会使误差不断积累. 吉尔法是专门用来求解刚性方程的一种数值方法.

```
> ODE:=diff(u(t),t)=-2000*u(t)+999.75*v(t)+1000.25,diff(v(t),t)=u(t)-v(t);
```

$$ODE := \frac{\partial}{\partial t}\mathrm{u}(t) = -2000\ \mathrm{u}(t) + 999.75\ \mathrm{v}(t) + 1000.25,\ \frac{\partial}{\partial t}\mathrm{v}(t) = \mathrm{u}(t) - \mathrm{v}(t)$$

```
> initvals:=u(0)=0,v(0)=-2;
```

$$initvals := \mathrm{u}(0) = 0,\ \mathrm{v}(0) = -2$$

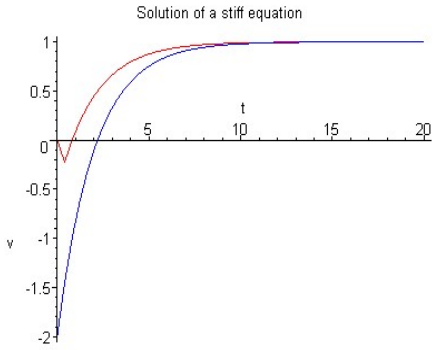
```
> ansl:=dsolve({ODE,initvals},{u(t),v(t)},type=numeric,method=gear);
```

$$ansl := \mathbf{proc}\ (x\_gear)\ \ldots\ \mathbf{end\ proc}$$

```
> ansl(10,0);
```

$$[t = 10.,\ \mathrm{u}(t) = .989893921726687442\ ,\ \mathrm{v}(t) = .979787842765888594\ ]$$

```
> p1:=plots[odeplot] (ansl,[t,u(t)],0..20,color=red):
  p2:=plots[odeplot] (ansl,[t,v(t)],0..20,color=blue):
  plots[display] ({p1,p2}, title="Solution of a stiff equation");
```



2.5.3 经典数值方法

Maple 中常微分方程数值解法中有一类被称作是“经典”(**classical**)方法. 当然, 称其为经典方法不是因为它们常用或是精度高, 而是因为它们的形式简单, 经常被用于计算方法课上的教学内容. 它们是一些常见的固定步长方法, 在 **dsolve** 中用参数 **method=classical**[方法名称], 如果不特别指出, 将默认采用向前欧拉法. 主要有：

foreuler：向前欧拉法(默认)

hunform：Heun 公式法(梯形方法, 改进欧拉法)

imply：改进多项式法

rk2：二阶龙格库塔法

rk3：三阶龙格库塔法

rk4：四阶龙格库塔法

adambash：Adams-Bashford 方法(预测法)

abmoulton：Adams-Bashford-Moulton 方法(预测法)

下面给出微分方程数值方法的参数表：

参数名	参数类型	参数用途	参数用法
initial	浮点数的一维数组	指定初值向量	
number	正整数	指定向量个数	
output	'procedurelist'(默认)或'listprocedure'	指定生成单个函数或多个函数的有序表	Procedurelis：单个函数, 返回有序表 Listprocedure：函数的有序表
procedure	子程序名	用子程序形式指定第一尖常微分方程组的右边部分	参数 1：未知函数的个数 参数 2：自变量 参数 3：函数向量 参数 4：导函数向量
start	浮点数	自变量起始值	
startinit	布尔量(默认 FALSE)	指定数值积分是否总是从起始值开始	对 dverk78 不适用
value	浮点数向量(一维数组)	指定需要输出函数值的自变量数值点	如果给定, 结果是一个 2×2 的矩阵. 元素[1,1]是一个向量, 含自变量名和函数名称; 元素[2,1]是一个数值矩阵, 其中第一列 value 的输入相同, 其他列中是相应的函数值

另外，还有一些特殊的附加参数：

maxfun：整数类型，用于最大的函数值数量，默认值 50000，为负数时表示无限制

corrections：正整数类型，指定每步修正值数量，在 **abmoulton** 中使用，建议值≤4

stepsize：浮点数值，指定步长

下面看一个简单的例子：

```
> ODE:=diff(y(x),x)=y(x)-2*x/y(x);
```

$$ODE := \frac{\partial}{\partial x}\mathrm{y}(x) = \mathrm{y}(x) - \frac{2x}{\mathrm{y}(x)}$$

```
> initvals:=y(0)=1;
```

$$initvals := \mathrm{y}(0) = 1$$

```
> sol1:=dsolve({ODE,initvals},y(x),numeric,method=classical,stepsize=0.1,start=0);
```

$$sol1 := \mathbf{proc}\ (x\_classical)\ \ldots\ \mathbf{end\ proc}$$

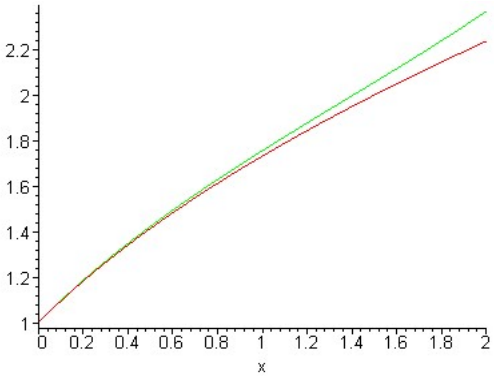
而其解析解为：

```
> sol2:=dsolve({diff(y(x),x)=y(x)-2*x/y(x), y(0)=1}, y(x));
```

$$sol2 := \mathrm{y}(x) = \sqrt{2x+1}$$

将两者图形同时绘制在同一坐标系中比较，可以发现，在经过一段时间后，欧拉法的数值结果会产生较大误差.

```
> plot({rhs(sol2),'rhs(sol1(x)[2])'},x=0..2);
```



求解微方程，无论使用什么方法或者加入什么选项，求解完成后必须利用相关数学知识进行逻辑判断，绝对不对简单迷信 Maple 给出的结果，否则很有可能得到一个对于方程本身也许还看得过去，但在数学或者物理意义上不合理的解.

2.6 摄动法求解常微分方程

由于微分方程求解的复杂性，一般微分方程常常不能求得精确解析解，需要借助其它方法求得近似解或数值解，或者两种方法兼而有之. 摄动法是重要的近似求解方法.

摄动法又称小参数法，它处理含小参数 ε 的系统，一般当 ε=0 时可求得解 x_0. 于是可把原系统的解展成 ε 的幂级数 $x = x_0 + x_1\varepsilon + x_2\varepsilon^2 + \cdots$，若这个级数当 $\varepsilon \to 0$ 时一

致收敛，则称正则摄动，否则称奇异摄动. 摄动法的种类繁多，最有代表性的是庞加莱—林斯泰特(**Poicare-Lindstedt**)法，在此，我们以该方法求解 **van der Pol** 方程：

$$y'' - \varepsilon(1-y^2)y' + y = 0$$

当 ε =0 时该方程退化为数学单摆的常微分方程，当 ε =1 时为 3.5 讨论的情况，对任意 ε，该微分方程拥有一个渐进稳定的周期解，称为极限环.

由于 van der Pol 方程中没有显式的时间依赖项，不失一般性，设初值为 y(0)=0. 在庞加莱—林斯泰特法中，时间通过变换拉伸：

$$\tau = \omega t, \text{ 其中 } \omega = \sum_{i=0}^{\infty} \omega_i \varepsilon^i$$

对于 $y(\tau)$, van der Pol 方程变为：

$$\omega^2 y'' - \omega\varepsilon(1-y^2)y' + y = 0$$

```
restart:
diff(y(t),t$2)-epsilon*(1-y(t)^2)*diff(y(t),t)+y(t)=0;
```

$$\left(\frac{\partial^2}{\partial t^2} y(t)\right) - \varepsilon\,(1-y(t)^2)\left(\frac{\partial}{\partial t} y(t)\right) + y(t) = 0$$

```
> ODE:=DEtools[Dchangevar]({t=tau/omega,y(t)=y(tau)},%,t,tau);
```

$$ODE := \omega^2\left(\frac{\partial^2}{\partial \tau^2} y(\tau)\right) - \varepsilon\,(1-y(\tau)^2)\,\omega\left(\frac{\partial}{\partial \tau} y(\tau)\right) + y(\tau) = 0$$

```
> e_order:=6:
> macro(e=epsilon,t=tau):
> alias(seq(y[i]=eta[i](tau),i=0..e_order)):
> e:=()->e:
> for i from 0 to e_order do
  eta[i]:=t->eta[i](t)
  od:
> omega:=1+sum('w[i]*e^i','i'=1..e_order);
```

$$\omega := 1 + w_1\,\varepsilon + w_2\,\varepsilon^2 + w_3\,\varepsilon^3 + w_4\,\varepsilon^4 + w_5\,\varepsilon^5 + w_6\,\varepsilon^6$$

```
> y:=sum('eta[i]*e^i','i'=0..e_order);
```

$$y := \eta_0 + \eta_1\,\varepsilon + \eta_2\,\varepsilon^2 + \eta_3\,\varepsilon^3 + \eta_4\,\varepsilon^4 + \eta_5\,\varepsilon^5 + \eta_6\,\varepsilon^6$$

```
> deqn:=simplify(collect(ODE,e),{e^(e_order+1)=0}):
```

> **for i from 0 to e_order do**
ode[i]:=coeff(lhs(deqn),e,i)=0
od:
> **ode[0];**

$$y_0 + \left(\frac{\partial^2}{\partial \tau^2} y_0\right) = 0$$

> **ode[1];**

$$\left(\frac{\partial^2}{\partial \tau^2} y_1\right) - \left(\frac{\partial}{\partial \tau} y_0\right) + y_1 + 2\, w_1 \left(\frac{\partial^2}{\partial \tau^2} y_0\right) + \left(\frac{\partial}{\partial \tau} y_0\right) {y_0}^2 = 0$$

> **ode[2];**

$$\left(\frac{\partial}{\partial \tau} y_0\right) w_1 {y_0}^2 + 2 \left(\frac{\partial}{\partial \tau} y_0\right) y_0 y_1 - \left(\frac{\partial}{\partial \tau} y_1\right) + \left(\frac{\partial^2}{\partial \tau^2} y_2\right) + y_2 + \left(\frac{\partial}{\partial \tau} y_1\right) {y_0}^2 - \left(\frac{\partial}{\partial \tau} y_0\right) w_1 + 2\, w_1 \left(\frac{\partial^2}{\partial \tau^2} y_1\right) + 2 \left(\frac{\partial^2}{\partial \tau^2} y_0\right) w_2 + \left(\frac{\partial^2}{\partial \tau^2} y_0\right) {w_1}^2 = 0$$

> **dsolve({ode[0],eta[0](0)=0,D(eta[0])(0)=C[1]},eta[0](t));**

$$y_0 = C_1 \sin(\tau)$$

> **eta[0]:=unapply(rhs(%),t);**

$$\eta_0 := \tau \to C_1 \sin(\tau)$$

> **ode[1];**

$$\left(\frac{\partial^2}{\partial \tau^2} y_1\right) - C_1 \cos(\tau) + y_1 - 2\, w_1\, C_1 \sin(\tau) + {C_1}^3 \cos(\tau) \sin(\tau)^2 = 0$$

> **map(combine,ode[1],'trig');**

$$\left(\frac{\partial^2}{\partial \tau^2} y_1\right) - C_1 \cos(\tau) + y_1 - 2\, w_1\, C_1 \sin(\tau) + \frac{1}{4} {C_1}^3 \cos(\tau) - \frac{1}{4} {C_1}^3 \cos(3\,\tau) = 0$$

> **ode[1]:=map(collect,%,[sin(t),cos(t)]);**

$$ode_1 := -2\, w_1\, C_1 \sin(\tau) + \left(-C_1 + \frac{1}{4} {C_1}^3\right) \cos(\tau) + \left(\frac{\partial^2}{\partial \tau^2} y_1\right) + y_1 - \frac{1}{4} {C_1}^3 \cos(3\,\tau) = 0$$

> **dsolve({ode[1],eta[1](0)=0,D(eta[1])(0)=C[2]},eta[1](t),method=laplace);**

$$y_1 = \left(-\frac{1}{8} C_1 (C_1 - 2) (C_1 + 2)\, \tau + w_1\, C_1 + C_2\right) \sin(\tau) + \left(\frac{1}{32} {C_1}^3 - C_1\, \tau\, w_1\right) \cos(\tau) - \frac{1}{32} {C_1}^3 \cos(3\,\tau)$$

> **map(collect,%,[sin(t),cos(t),t]);**

$$y_1 = \left(-\frac{1}{8} C_1 (C_1 - 2) (C_1 + 2)\, \tau + w_1\, C_1 + C_2\right) \sin(\tau) + \left(\frac{1}{32} {C_1}^3 - C_1\, \tau\, w_1\right) \cos(\tau) - \frac{1}{32} {C_1}^3 \cos(3\,\tau)$$

```
> solve({coeff(lhs(ode[1]),sin(t))=0,coeff(lhs(ode[1]),cos(t))=0});
```

$$\{C_1 = 0, w_1 = w_1\}, \{C_1 = 2, w_1 = 0\}, \{C_1 = -2, w_1 = 0\}$$

```
> w[1]:=0:C[1]:=-2:
> ode[1];
```

$$\left(\frac{\partial^2}{\partial \tau^2} y_1\right) + y_1 + 2\cos(3\,\tau) = 0$$

```
> dsolve({ode[1],eta[1](0)=0,D(eta[1])(0)=C[2]},eta[1](t),method=laplace);
```

$$y_1 = \frac{1}{4}\cos(3\,\tau) - \frac{1}{4}\cos(\tau) + C_2\sin(\tau)$$

```
> eta[1]:=unapply(rhs(%),tau);
```

$$\eta_1 := \tau \to \frac{1}{4}\cos(3\,\tau) - \frac{1}{4}\cos(\tau) + C_2\sin(\tau)$$

```
> map(combine,ode[2],'trig'):
> ode[2]:=map(collect,%,[sin(t),sin(3*t),cos(t),cos(3*t)]);
```

$$ode_2 := \left(\frac{1}{4} + 4\,w_2\right)\sin(\tau) + \frac{5}{4}\sin(5\,\tau) + \left(\frac{\partial^2}{\partial \tau^2} y_2\right) - \frac{3}{2}\sin(3\,\tau) + 2\,C_2\cos(\tau) - 3\,C_2\cos(3\,\tau) + y_2 = 0$$

```
> solve({coeff(lhs(ode[2]),sin(t))=0,coeff(lhs(ode[2]),cos(t))=0});
```

$$\{C_2 = 0, w_2 = \frac{-1}{16}\}$$

```
> assign(%):
> dsolve({ode[2],eta[2](0)=0,D(eta[2])(0)=C[3]},eta[2](t),method=laplace):
> eta[2]:=unapply(rhs(%),t);
```

$$\eta_2 := \tau \to -\frac{3}{16}\sin(3\,\tau) + \left(\frac{29}{96} + C_3\right)\sin(\tau) + \frac{5}{96}\sin(5\,\tau)$$

```
> for i from 0 to e_order do
    map(combine,ode[i],'trig'):
    ode[i]:=map(collect,%,[seq(sin((2*j+1)*t),j=0..i),seq(cos((2*j+1)*t),j=0..i)]):
    solve({coeff(lhs(ode[i]),sin(t))=0,coeff(lhs(ode[i]),cos(t))=0}):
    assign(%):
    dsolve({ode[i],eta[i](0)=0,D(eta[i])(0)=C[i+1]},eta[i](t),method=laplace):
    collect(%,[seq(sin((2*j+1)*t),j=0..i),seq(cos((2*j+1)*t),j=0..i)]):
    eta[i]:=unapply(rhs(%),t);
od:
> omega;
```

$$1 - \frac{1}{16}\varepsilon^2 + \frac{17}{3072}\varepsilon^4 + \frac{35}{884736}\varepsilon^6$$

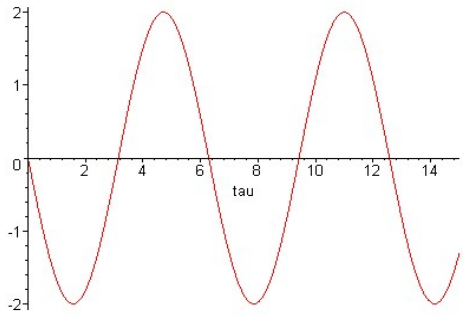
> **y(t):**

> **y:=unapply(simplify(y(t),{e^e_order=0}),t):**

> **e:=1:y(t);**

$$-\frac{1037927}{552960}\sin(\tau)+\frac{1519}{540}\sin(\tau)\cos(\tau)^6-\frac{81}{32}\sin(\tau)\cos(\tau)^4-\frac{61}{80}\sin(\tau)\cos(\tau)^8+\frac{13}{256}\sin(\tau)\cos(\tau)^2+\frac{5257957}{3317760}\cos(\tau)^3+\frac{5533}{7200}\cos(\tau)^{11}+\frac{912187}{129600}\cos(\tau)^7-\frac{799991}{1105920}\cos(\tau)$$
$$-\frac{1212373}{259200}\cos(\tau)^5-\frac{114941}{28800}\cos(\tau)^9$$

> **plot(y(t),t=0..15);**



在该问题的求解过程中，前半部分我们按照交互式命令方式输入，也就是把数学逻辑推理的过程“翻译”成 Maple 函数，而在后半部分，则采用程序设计方式书写了数学推导过程，这是应用 Maple 解决实际问题的两种方式. 前一种方法只需了解 Maple 函数即可应用，而后一种程序设计方式则需掌握 Maple 程序设计语言. 但是，不论是那一种方式，数学基础总是最重要的.

3 偏微分方程求解初步

Maple 中偏微分方程求解器为 **pdsolve**，该函数及其它偏微分方程求解工具存于软件包 **PDEtools** 中. 函数 **pdsolve** 能够很快的辨认出偏微分方程是否为可以用标准方法求解的类型，如果无法判别，则 **pdsolve** 采用一种启发式的算法尝试偏微分方程按特征结构分离出来. **pdsolve** 的策略就是寻找给定偏微分方程的通解，如不能成功则寻找可以完全分离的变量，因此，该函数返回的结果可能为：

(i) 通解；

(ii) 近似的通解(即包含任意函数但又不足以得到通解的解)；

(iii) 变量分离的非耦合的常微分方程.

如不能完全分离变量，则函数再次调用自身，如仍失败则返回未完全分离的变量并给出一个警告信息. 其命令格式为：

pdsolve(PDE, f);

其中, PDE 为偏微分方程, f 为被求解函数.

下面通过几个例子说明 pdsolve 的用法：

> **PDE1:=diff(u(x,t),[t$2])-1/v^2*diff(u(x,t),[x$2]);**

$$PDE1 := \left(\frac{\partial^2}{\partial t^2}\,u(x,t)\right) - \frac{\frac{\partial^2}{\partial x^2}\,u(x,t)}{v^2}$$

> **pdsolve(PDE1,u(x,t));**

$$u(x,t) = \_F1(-t - v\,x) + \_F2(t - v\,x)$$

> **restart:**

> **PDE2:=a*x*diff(u(x,t),x)+b*t*diff(u(x,t),t)=0;**

$$PDE2 := a\,x\left(\frac{\partial}{\partial x}\,u(x,t)\right) + b\,t\left(\frac{\partial}{\partial t}\,u(x,t)\right) = 0$$

> **pdsolve(PDE2,u(x,t));**

$$u(x,t) = \_F1\left(\frac{t}{x^{\left(\frac{b}{a}\right)}}\right)$$

> **PDE3:=diff(u(x,t),x$2)-t^2*diff(u(x,t),t$2)-t*diff(u(x,t),t)=0;**

$$PDE3 := \left(\frac{\partial^2}{\partial x^2}\,u(x,t)\right) - t^2\left(\frac{\partial^2}{\partial t^2}\,u(x,t)\right) - t\left(\frac{\partial}{\partial t}\,u(x,t)\right) = 0$$

> **pdsolve(PDE3,u(x,t));**

$$u(x,t) = \_F1(t\,e^{x}) + \_F2(t\,e^{(-x)})$$

> **restart:**

> **heatPDE:=diff(u(x,t),t)=diff(u(x,t),[x$2]);**

$$heatPDE := \frac{\partial}{\partial t}\,u(x,t) = \frac{\partial^2}{\partial x^2}\,u(x,t)$$

> **pdsolve(heatPDE,u(x,t));**

$$(u(x,t) = \_F1(x)\,\_F2(t))\ \&\text{where}\left[\left\{\frac{\partial^2}{\partial x^2}\,\_F1(x) = \_c_1\,\_F1(x), \frac{\partial}{\partial t}\,\_F2(t) = \_c_1\,\_F2(t)\right\}\right]$$

> **dsolve(diff(F2(t),t)=c[1]*F2(t),F2(t));**

$$F2(t) = \_C1\,e^{(c_1\,t)}$$

> **dsolve(diff(F1(x),[x$2])=c[1]*F1(x));**

$$F1(x) = \_C1\,e^{\left(\sqrt{c_1}\,x\right)} + \_C2\,e^{\left(-\sqrt{c_1}\,x\right)}$$

> **result:=rhs(%)*rhs(%%);**

$$result := \left(\_C1\,e^{\left(\sqrt{c_1}\,x\right)} + \_C2\,e^{\left(-\sqrt{c_1}\,x\right)}\right)\_C1\,e^{(c_1\,t)}$$

> **subs(u(x,t)=result,heatPDE):expand(%);**

$$\_C1^2\, c_1\, e^{(c_1 t)}\, e^{(\sqrt{c_1} x)} + \frac{\_C1\, c_1\, e^{(c_1 t)}\, \_C2}{e^{(\sqrt{c_1} x)}} =$$

$$\_C1^2\, c_1\, e^{(c_1 t)}\, e^{(\sqrt{c_1} x)} + \frac{\_C1\, c_1\, e^{(c_1 t)}\, \_C2}{e^{(\sqrt{c_1} x)}}$$

> **lhs(%)-rhs(%);**

$$0$$