

## Maple Lecture 17. Working with Functions

The language in which we write Maple procedure is imperative, similar to C, although its syntax is more like that of Pascal (simplified Algol, survived in Ada). However, a Maple worksheet is very much like a functional program. A functional program is a single expression, executed by evaluating the expression. We focus on what is computed, not on how it should be computed.

The material in this lecture is based upon [1, Sections 8.7 and 8.8].

### 17.1 Anonymous Functions

Anonymous functions are utilities without name. They occur mostly in conjunction with functions like map.

```
[> sq := seq(x[i], i=0..10);          # create a sequence of variables
```

Observe that map does not apply to sequences, we have to wrap our sequence into a list:

```
[> lsq := map(x->x^3, [sq]);          # raise everything to the power three
```

The  $x \rightarrow x^3$  is an anonymous function.

```
[> p := convert(lsq, '+' );
```

Now we raise  $x[i]$  to the power  $i$ :

```
[> q := map(t -> t^(op(op(1,t))), p);
```

Selecting the terms of degree higher than 10:

```
[> select(t -> degree(t)>10, q);
```

The opposite of select is remove :

```
[> remove(t -> degree(t)>10, q);
```

### 17.2 Operations on Functions

The usual arithmetical operations (like addition, subtraction, multiplication, division) are defined in a direct manner. For the composition operator, we use the at operator, denoted by @.

Suppose the trajectory of a projectile in space is modeled by a parabola, subject to the following constraints. At time  $t = 0$ , it is launched from the ground and at time  $t = 45$  it hits the ground 120 miles further. Create the function  $f(t)$  which gives the altitude of the projectile in function of time, assuming constant horizontal speed.

First we have the parabola:

```
[> y := x -> x*(120-x);
```

```
[> plot(y, 0..120);
```

The assumption of constant horizontal speed implies that  $x$  is just a rescaling of  $t$ , i.e.: when  $t = 45$ ,  $x$  must be 120:

```
[> xt := t -> 120/45*t;
```

```
[> xt(0); xt(45);
```

Then the altitude is given as the composition of  $y$  after  $xt$ :

```
[> f := y@xt;
```

```
[> f(0); f(22.5); f(45);
```

Because of the assumption of horizontal constant speed, the peak is reached at  $t = 22.5$ .

As an exercise, we can let the projectile travel twice as slow :

```
[> halft := t -> t/2;
```

```
[> slowf := f@halft;
```

```
[> f(20) = slowf(40); f(45) = slowf(90);
```

### 17.3 Moving along a Spiral with Constant Speed

This application is in the same spirit as done in the previous section, but no longer possible to do by hand. We will create a new function from the numerical solution of a differential equation.

Suppose we want to model the motion along a spiral in the plane with constant speed. The representation of a spiral is very easy in polar coordinates:

```
[> plot([t,t,t=0..2*Pi],coords=polar);
```

The transition to cartesian (or rectangular coordinates) is direct:

```
[> x := t*cos(t); y := t*sin(t);
[> plot([x,y,t=0..2*Pi]);
```

Let us first compute the speed using the current representation (assuming  $t$  as independent variable):

```
[> dx := diff(x,t); dy := diff(y,t);
[> v := sqrt(dx^2+dy^2);
[> v := simplify(v);
[> plot(v,t=0..20*Pi);
```

From the plot we see that the speed increases with  $t$ . The acceleration is constant though. If we use this simple representation of a spiral we must travel faster and faster as time goes. We do not want this. We want a function of time which tells us or any moment in time our position on the spiral, moving with constant speed.

To obtain such a function, we need to choose other variables. Let us look at the speed when  $t$  is  $T(s)$ , a function of time. So we replace  $x(t)$  by  $T(s)*\cos(T(s))$  and  $y(t)$  by  $T(s)*\sin(T(s))$ . Then the square of the speed becomes

```
[> V := diff(T(s)*cos(T(s)),s)^2 + diff(T(s)*sin(T(s)),s)^2;
[> fV := factor(V);
```

Via the chain rule we see the square of the speed coming back in, but multiplied with the square of the derivative of  $T(s)$ . To find  $T$ , we need to solve a differential equation. Unfortunately, the solution to the differential equation does not come in a nice symbolic form, we ask Maple to solve it numerically.

```
[> equ := diff(T(s),s) = 1/sqrt(T(s)^2+1);
[> sol := dsolve({equ,T(0)=0},T(s),numeric);
[> sol(2);
[> fT := s->rhs(sol(s)[2]);
[> fT(2);
[> fx := unapply(x,t); fy := unapply(y,t);
[> hfx := fx@fT; hfy := fy@fT;
[> plot([hfx,hfy,0..7*Pi]);
```

From the plot we can already see that we are moving slower than on the other plots. But let us verify a bit more precisely – from calculus we recall that speed is distance traveled over time:

```
[> (sqrt((hfx(100)-hfx(99.9))^2 + (hfy(100)-hfy(99.9))^2))/0.1;
[> (sqrt((hfx(10)-hfx(9.9))^2 + (hfy(10)-hfy(9.9))^2))/0.1;
[> (sqrt((hfx(1)-hfx(0.9))^2 + (hfy(1)-hfy(0.9))^2))/0.1;
```

We see that the numerical derivative approaches 1. So we have created the coordinate functions for moving along a spiral at constant speed equal to one.

## 17.4 Assignments

1. Type `l := [seq(rand() mod 100, i=1..10)]`; to generate a list of 10 random numbers between 0 and 99. Give the Maple commands for the following operations
  - (a) divide every element in the list by 100;
  - (b) convert the list to a list of floating-point numbers of 3 decimal places;
  - (c) select all elements in the list larger than 0.5;
  - (d) compute the sum of the elements in the list.
2. The command `isprime()` applied to a number returns *true* if the number is prime and *false* otherwise. Use `isprime()` to make a list of all primes in the range 1..1000.
3. Type `p := randpoly([x,y], degree=7, dense)`; to generate a dense polynomial of degree 7 in x and y. Give the Maple command to remove all terms in p whose degree is less than the degree of p.
4. Type `p := randpoly(x, degree=10, dense)`; to generate a dense polynomial of degree 10 in x. Give the Maple command to select from p those terms whose coefficient is positive.
5. The Riemann Zeta function is defined by the command `Zeta(z)`, where **z** is any algebraic expression. Use a macro to define `fzeta`, which gives a floating point approximation (using the standard precision in Digits) for the value of the Riemann Zeta function. For example, `fzeta(3.0)` should yield 1.202056903.
6. Suppose the spilling at an oil pipe produces a perfect circle on the floor. The area of the circle is  $\pi r^2$ , with the radius  $r = r(t)$  increasing in time  $t$ .  
Use composition of two functions to model the growing of the area of the circle as a function of time, assuming the speed at which the spilling happens remains constant.

## References

- [1] A. Heck. *Introduction to Maple*. Springer-Verlag, third edition, 2003.