Computer Science 260, Spring 1999

Introduction to Scientific, Symbolic, and Graphical Computation

Instructor: Radford M. Neal

The Subject of this Course

How can we use mathematical models to answer questons about the world, or to do things that we want?

- Planning for retirement: How fast will the money I save grow from investments? How long am I likely to live? How much will it cost to live during my retirement?

- Global climate change: How do human and natural changes in environment affect the climate? How much carbon dioxide will be produced by humans in the future, under various government policies?

- Computer Graphics: How can we synthesize images whose appearance mimics that of real or imaginary scenes? How can we let designers create such scenes easily?

Two Steps to Solving a Problem

1. Formulate the problem in a mathematical form. We'll talk about this only a bit, now and then, but if you go wrong here, the rest will obviously be nonsense!

2. Get answers to interesting questions using this mathematical model. There are two main approaches:

    - *Symbolic* methods that yield explict formulas.

    - *Numerical* methods that produce an answer for a particular case.

    Methods of either sort can be *exact* or *approximate*.

    Finding both symbolic and numerical solutions is often easier with the help of a computer.

The Maple Programming Language

We will be using the Maple V programming language. Some characteristics of Maple differ from those of languages such as C or Turing:

- Maple is an *interactive* language. Eg, you can use it as a calculator:

```
> 125+7;

                    132
```

- Maple supports *symbolic* computation:

```
> expand((x+10)*(x-1));

                     2
                    x  + 9 x - 10
```

- Because Maple is *interpreted*, it is slow compared to *compiled* languages such as C or Turing, when used to do things that these other languages could do just as well.

Detailed Topics Covered

We will look in some detail at the following:

- Representing numbers on a computer.
- Computing approximations to functions.
- Interpolating data points.
- Representing curves and surfaces.
- Rendering curves for graphical display.
- Simulating dynamical systems.
- Finding solutions of equations.
- Computing integrals.
- Synthesizing and filtering signals.

Of course, there are many other mathematical methods that we won't have time to mention.

Approximating Numbers & Functions

Or, How Does Your Calculator Work?

During numerical calculations, we will often have to use *approximate* representations of numbers. How is this best done?

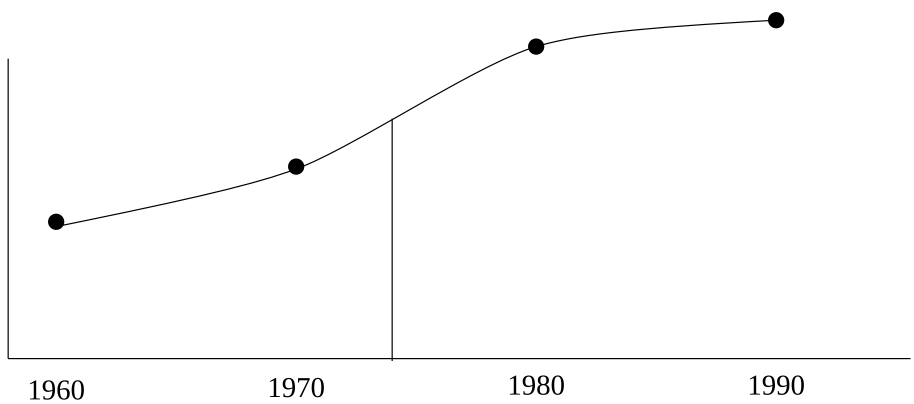
We will also often need to calculate the values of functions such as square root, logarithm, or sine. How is this possible?

Unlike your calculator, or most computer languages, Maple can represent numbers to as many digits of accuracy as you ask, and can approximate functions to that accuracy:

```
> Digits:=30;

          Digits := 30

> evalf(Pi);

          3.14159265358979323846264338328

> evalf(sqrt(12345.6789));

          111.111106055555544405416661434
```

Interpolating Data Points

In many applications, we need to *interpolate* between observed data points:

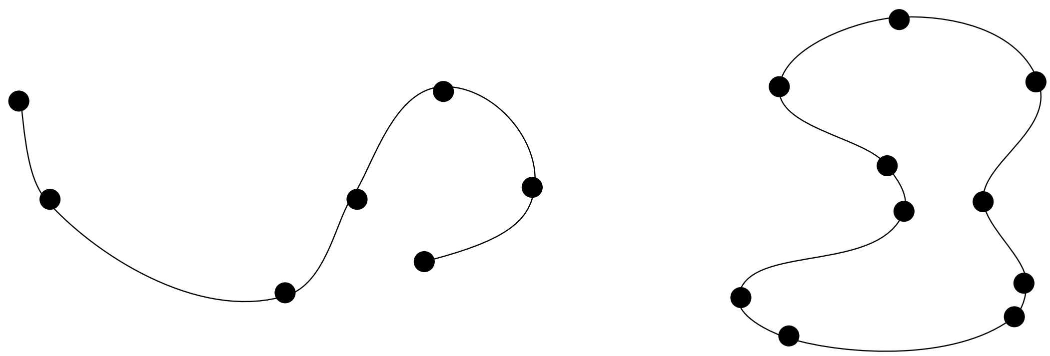


For instance, we might know the amount of carbon dioxide emitted into the atmosphere in 1960, 1970, 1980, and 1990, and want to know how much was emitted in 1974.

There are many ways of trying to do this. We will look at a few.

Drawing Curves with Interpolation

Interpolation can also be used to draw pictures using a computer (here, using "xfig"):

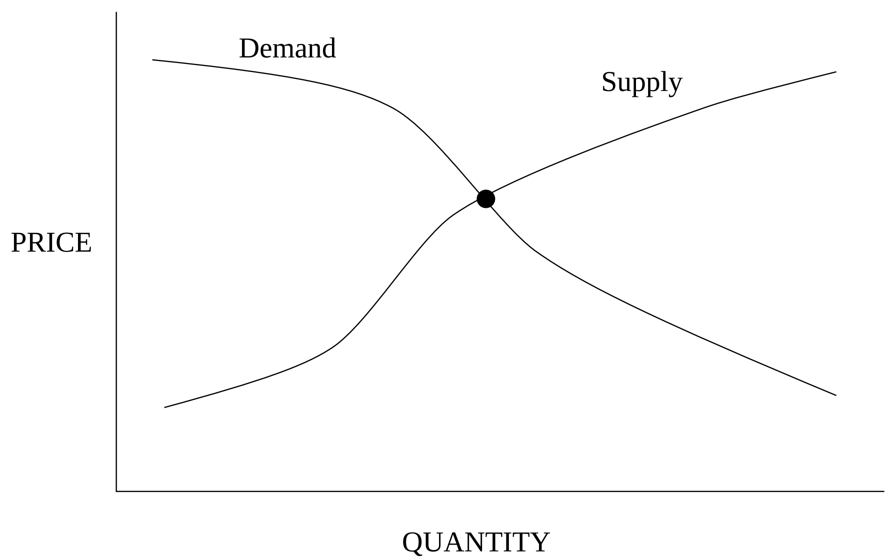


In this use, we use a *parametric* representation of a curve, in terms of functions for the x and y coordinates of points on the curve. We then apply interpolation to each of these functions.

In this application, we aren't trying to model reality. But we do want the scheme we use to behave in a way that will make it easy to create the pictures we want.

Solving Equations

To find the price at which a good is expected to trade, and economist might solve an equation equating supply and demand:

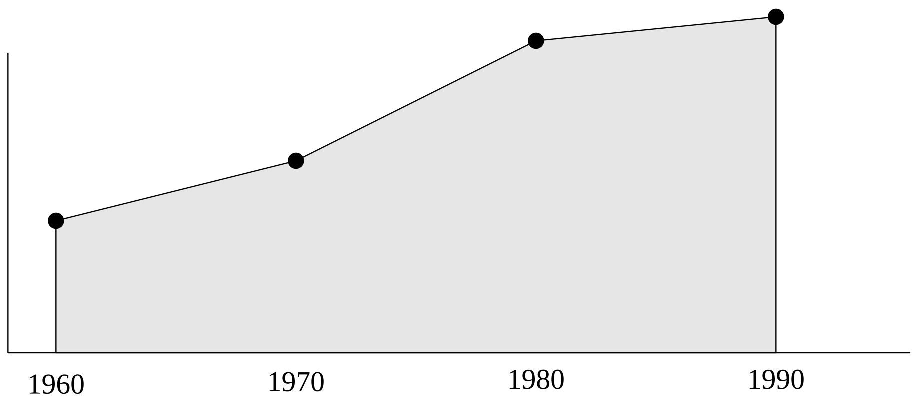


An equivalent problem is that of finding the points where a function is zero. Finding the points where a function is minimized or maximized is a closely related problem.

You are probably used to solving equations symbolicly, by hand, but it is often necessary to solve them numerically, using a computer.

Integrating Functions

Problems involving areas, volumes, totals, and averages often reduce to computing the (definite) integral of a function:

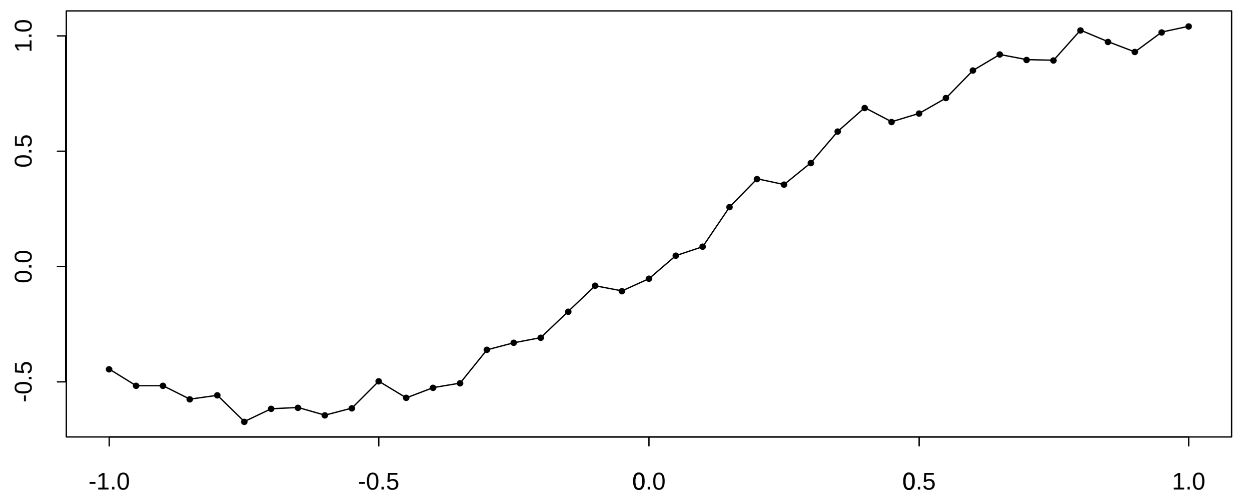


Here, we see how finding the total amount of carbon dioxide emitted from 1970 to 1990 can be seen as calculation of an integral. This is done above using straight-line interpolation, but there are better ways, as we'll see.

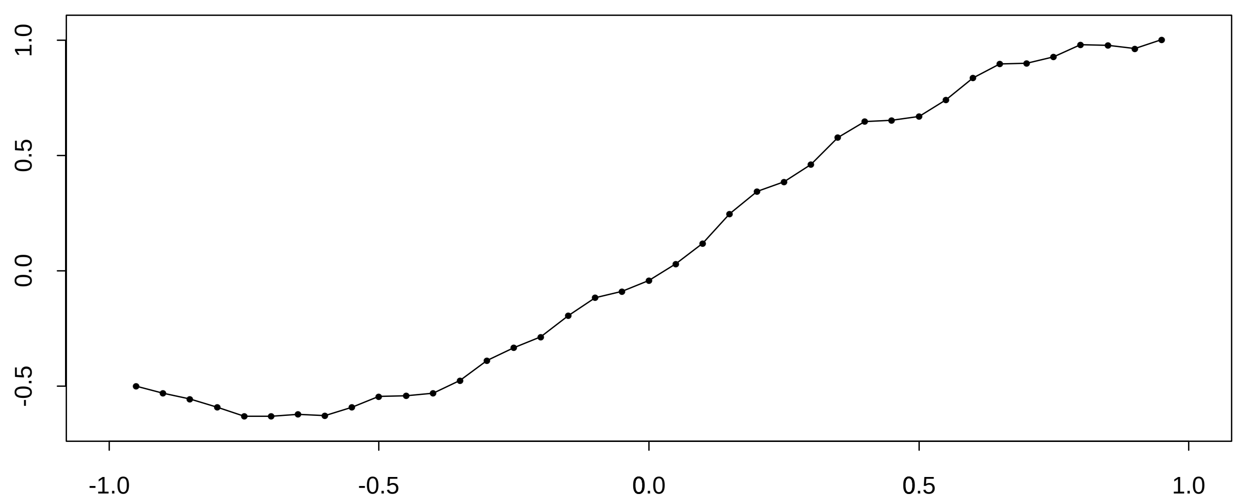
Filtering Signals

A signal (eg, music) may be *filtered*, to suppress noise, or emphasize certain aspects.

A noisy signal:



After applying a moving average filter:



Some General Themes

Many of the methods we will look at for solving these problems share some general characteristics:

- They often involve approximating functions by *polynomials*, or later by *sine waves*.

- They often involve *linear* operations, for which $L(a+b) = L(a) + L(b)$.

- Linear operations can be represented in terms of their effect on simple objects — eg, as given by *basis functions*, or *impulse response functions*.

Exact and Approximate Numerical Calculations in Maple

Maple can calculate numerical results *exactly*, as fractions, and *approximately*, as "floating point" numbers:

```
> 1/2 + 1/3;
                              5/6

> Digits:=5;
                          Digits := 5

> 1.0/2.0 + 1.0/3.0;
                             .83333

> Digits:=10;
                          Digits := 10

> 1.0/2.0 + 1.0/3.0;
                          .8333333333

> 1.0/3.0 + 1.0/3.0;
                          .6666666666

> evalf(1/3+1/3);
                          .6666666667
```

Saving Values in Variables

We can save the result of a computation in a Maple variable, then refer to it later:

```
> a := 1/7;
                                    a := 1/7

> b := (1+a)^7;
                                       2097152
                                  b := -------
                                       823543

> evalf(b);
                                  2.546499697

> pow := 100;
                                   pow := 100

> (1 + 1.0/pow)^pow;
                                  2.704813829
```

Symbolic Calculations in Maple

When we refer to a variable that hasn't been given a value, Maple treats it as unknown, and does symbolic computations using it:

```
> (x^5-1)/(x-1);
                                      5
                                     x  - 1
                                     ------
                                     x - 1

> simplify((x^5-1)/(x-1));
                               4    3    2
                              x  + x  + x  + x + 1
> (x+1)^5;
                                          5
                                    (x + 1)

> expand((x+1)^5);
                         5      4       3       2
                        x  + 5 x  + 10 x  + 10 x  + 5 x + 1

> b := x^2+1;
                                        2
                                  b := x  + 1

> subs(x=5,b);
                                       26
```

Example Problem: Paying a Mortgage

Let's see how Maple can be used to figure out mortgage payments, both numerically and symbolically.

We start with a debt of D dollars. Every year, interest on the remaining debt is charged at rate r (eg, $r = 0.06$ for 6% interest), and we make a payment of A dollars against the debt.

If P is the amount owed at the start of a year, the amount owed at the end of the year (after interest and payment) will be

$$P' = P + rP - A$$

Some questions:

- Given D, r, and A, how much will still be owed after n years?
- Given D and r, how big must A be so we pay off the mortgage in exactly n years?

A Maple Procedure to Find the Debt Remaining After n Years

We can express the way the mortgage works in a Maple *procedure* for calculating the debt that remains after n years:

```
mortgage := proc (D,r,A,n)

    local P, i;

    P := D;

    for i from 1 to n do
        P := P + r*P - A;
    od;

    P;

end;
```

Using the Procedure for Numerical Calculations

We can use this procedure to numerically calculate answers for specific values of D, r, A, and n:

```
> mortgage(10000,0.06,1000,1);
                                        9600.00

> mortgage(10000,0.06,1000,10);
                                      4727.682022

> mortgage(10000,0.06,1000,20);
                                      -4714.236482
```

This sort of calculation doesn't provide any easy way to figure out what A must be in order for the debt to be zero after n years.

It also doesn't give much insight into the way the debt goes down over the years.

Using the Procedure for Symbolic Computations

Let's try calculations using unknowns rather than actual numbers (except for n):

```
> mortgage(D,r,A,1);
                                    D + r D - A

> mortgage(D,r,A,2);
                         D + r D - 2 A + r (D + r D - A)

> mortgage(D,r,A,3);
D + r D - 3 A + r (D + r D - A) + r (D + r D - 2 A + r (D + r D - A))

> simplify(mortgage(D,r,A,3));
                                 2              3      2
              D + 3 r D - 3 A + 3 r  D - 3 r A + r  D - r  A

> simplify(mortgage(D,r,A,10));
                        2                  3           2          4
D - 10 A + 10 r D + 45 r  D - 45 r A + 120 r  D - 120 r  A + 210 r  D

         3          5          4          6          5         7
  - 210 r  A + 252 r  D - 252 r  A + 210 r  D - 210 r  A - 45 r  A

        8          7          6      9       10         8         9
  + 45 r  D + 120 r  D - 120 r  A - r  A + r    D - 10 r  A + 10 r  D
```

This seems a bit complicated, but let's go on anyway...

Solving for the Appropriate Payment Amount

Let's now ask Maple to figure out symbolically what the payment amount (A) must be in order to pay off the debt in a given number of years (n):

```
> needed_amount := solve(mortgage(D,r,A,5)=0,A);

                                           2        3       4    5
                         D (1 + 5 r + 10 r  + 10 r  + 5 r  + r )
        needed_amount := ---------------------------------------
                                              2       3    4
                              5 + 10 r + 10 r  + 5 r  + r
```

We can substitute specific values into this formula to get numerical answers:

```
> subs(D=10000,r=0.06,needed_amount);
                             2373.964005
```

A check shows that this answer is pretty close to being right:

```
> mortgage(10000,0.06,2373.964005,5);
                                        -5
                                 -.3 10
```

How Clever is Maple in this Example?

Maple has managed to solve this problem symbolically. But there is a better solution, that we might find manually.

We can rewrite the amount owed after the year in terms of the amount P owed at the start of the year as

$$P' = (1+r)P - A$$

We can now see that the amount owed after n years startign with a debt of D will be

$$D(1+r)^n - A[1+(1+r)+(1+r)^2+\cdots+(1+r)^{n-1}]$$

The sum in square brackets reduces to $((1+r)^n - 1)/r$. When we equate this to zero and then solve for A, we get

$$A = D\,\frac{r(1+r)^n}{(1+r)^n - 1}$$

How Does the Manual Solution Compare to Maple's Solution?

This solution turns out to be the same as Maple's. We can check this as follows:

```
> manual_answer := D*(r*(1+r)^5)/((1+r)^5-1);
                                                  5
                                         D r (1 + r)
                        manual_answer := ------------
                                              5
                                         (1 + r)  - 1

> simplify (needed_amount - manual_answer);

                                   0
```

The manual solution is simpler and more intelligible.

On the other hand, it required some cleverness. We can't always be clever, and sometimes there is no clever way to do things.