Computational Mathematics/Information Technology

Dr Oliver Kerr

2009–10

Introduction

We have covered a variety of topics in this course:

- Curve sketching
- Root finding
- Financial Functions
- Curve fitting
  - Polynomial fitting
  - Splines
  - Regression
- Markov chains
  - Google page ranking
- Numerical Integration
- Minitab

There are several areas that we haven't covered in the field of computational mathematics:

Chapters in Dr Bowtell's notes but not covered explicitly here:

- Polynomial Approximations
- Simultaneous Equations

Other topics:

- Computer Algebra
- Numerical solution of ODEs
- Numerical solution of PDEs
- ...

Polynomial Approximations

We have already seen how to fit a polynomial of order n through $N+1$ points.

Many of the techniques derived for finding polynomials as described in Dr Bowtell's notes are designed to make life easier if you are doing things by hand, or to avoid rounding errors if you are doing calculations to a limited accuracy. Many of these are not as necessary with up-to-date computers (64 bits $\approx$ 15 significant figures).

Polynomial approximations may come in other forms for example **Chebyshev Polynomials**:

$$T_0(x) = 1, \qquad T_1(x) = x$$

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x).$$

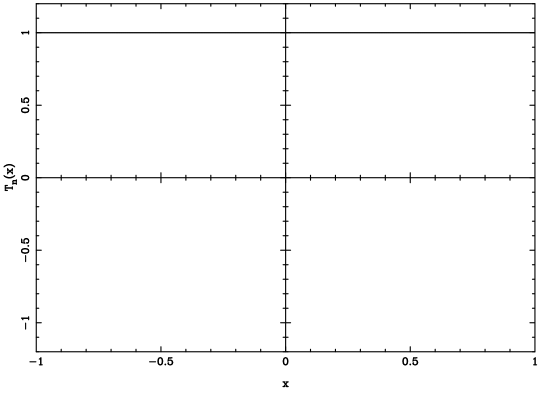
So

$$T_2(x) = 2x^2 - 1, \quad T_3(x) = 4x^3 - 3x, \quad \dots$$

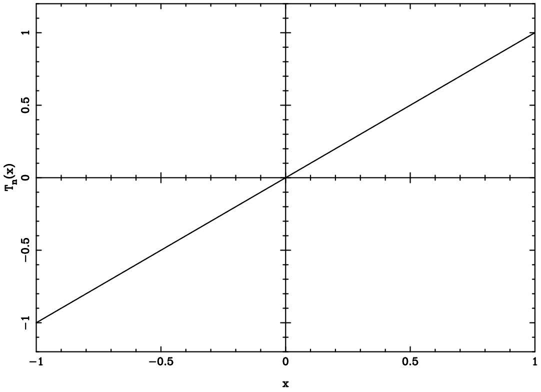
It can be shown that

$$T_n(x) = \cos(n \cos^{-1}(x))$$

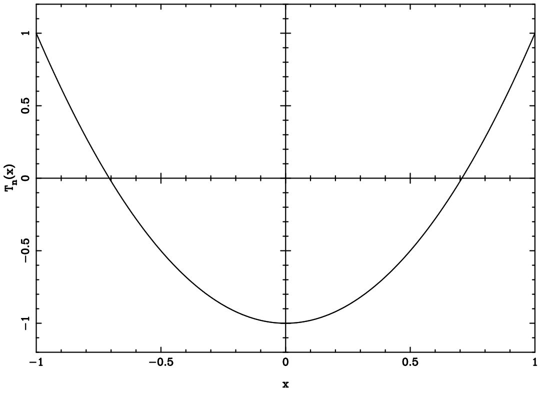
$T_0(x)$



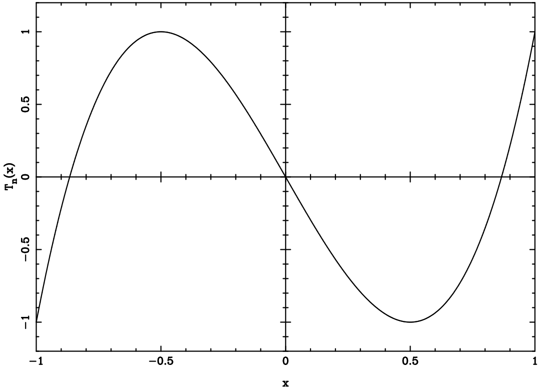
$T_1(x)$



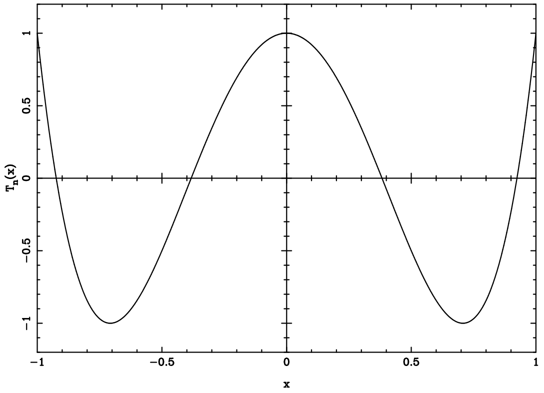
$T_2(x)$



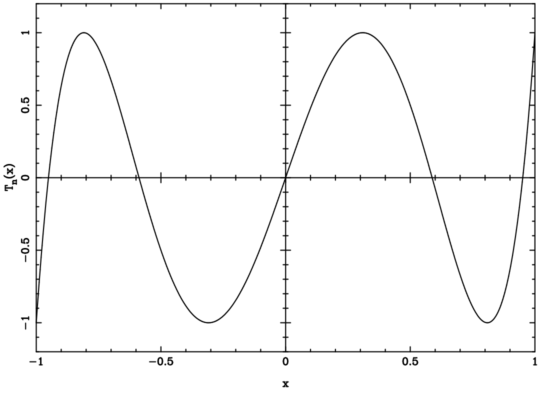
$T_3(x)$



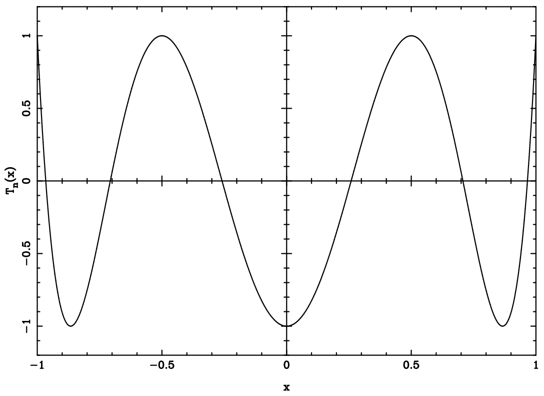
$T_4(x)$



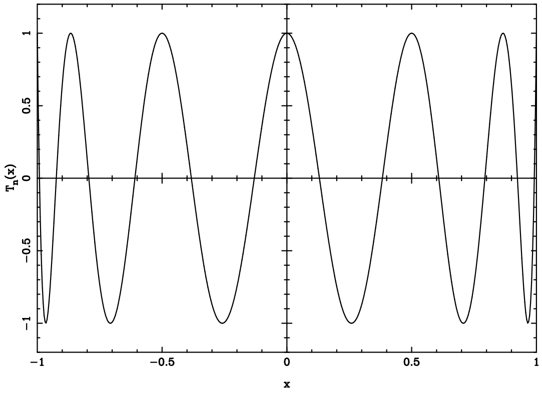
$T_5(x)$



$T_6(x)$



$T_{12}(x)$



Why are these useful?

It can be shown

$$\int_{-1}^{1} \frac{T_n(x)T_m(x)}{\sqrt{1-x^2}}\,dx = \begin{cases} 0 & : n \neq m \\ \pi & : n = m = 0 \\ \pi/2 & : n = m \neq 0 \end{cases}$$

The polynomials are **orthogonal**. (See Linear Algebra next year.)

If we have a function $f(x)$ defined on $[-1,1]$ we can approximate it as

$$f(x) = \sum_{n=0}^{N} a_n T_n(x)$$

For $n > 0$

$$a_n = \frac{2}{\pi} \int_{-1}^{1} \frac{T_n(x) f(x)}{\sqrt{1-x^2}}\, dx$$

The factor $2/\pi$ is replaced by $1/\pi$ for $n = 0$.

The value of a_n is independent of N.

We will come back to these later.

Simultaneous Equations

We have seen one way to approach simultaneous equations.

Given

$$\begin{aligned} x + y + z &= 1, \\ 2x + 4y + 3z &= 3, \\ 2x + 6y + (3 + a^2)z &= -4a, \end{aligned}$$

we write them in matrix form

$$A\mathbf{x} = \mathbf{b}$$

and use Excel, for example, to find

$$\mathbf{x} = A^{-1}\mathbf{b}$$

If we try and solve

$$\begin{aligned} x+y+z &= 1, \\ 2x+4y+3z &= 3, \\ 2x+6y+(3+a^2)z &= -4a, \end{aligned}$$

for the three cases $a=-1$, 0 and 1 we would (or should) fail to get solutions in two of the cases with Excel.

Let us try Derive...

If we try and solve

$$\begin{aligned} x + y + z &= 1, \\ 2x + 4y + 3z &= 3, \\ 2x + 6y + (3 + a^2)z &= -4a, \end{aligned}$$

for the three cases $a = -1$, 0 and 1 we would (or should) fail to get solutions in two of the cases with Excel.

Let us try Derive...

Using symbolic algebra packages can have their advantages.

Computer Algebra

The leading packages for symbolic/computer algebra are

- Maple
- Mathematica

Other, less feature rich, packages also exist

- Derive (now discontinued)
- Macsyma, now Maxima (Free)
- MuPAD, was free, now Symbolic Math Toolbox in Matlab
- Sage (Free)
- and many more...

What can they do?

Some are quite specialised, but the likes of Maple and Mathematica can easily solve problems like

- Find the general solution to

$$y'' + 3y' + 4y = x^2 + e^{3x}$$

- Find the general solution to

$$A_{n+1} + 4A_n = n$$

- ...

But not Derive!

Solving Differential Equations

Some differential equations can be solved using symbolic/computer algebra packages, but not all differential equations or systems of equations can be solved in that way.

Many methods or varying degrees of accuracy and complexity exist for finding solutions to ordinary differential equations of the form

$$y' = f(y, x)$$

or systems of equations

$$\mathbf{y}' = \mathbf{f}(\mathbf{y}, x)$$

Higher order equations can be written as systems of equations, for example

$$y'' = f(y', y, x)$$

Define

$$z = y'$$

then

$$y' = z$$

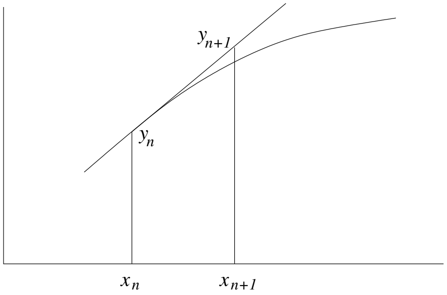
$$z' = f(z, y, x)$$

and we have a system of two first order equations.

The simplest way to solve numerically

$$y' = f(y, x)$$

is the explicit Euler method. Use a regular grid of spacing h (so $x_{n+1} - x_n = h$)



$$y'(x_n) = f(y_n, x_n) \approx \frac{y_{n+1} - y_n}{h} \qquad \text{or} \qquad y_{n+1} = y_n + hf(y_n, x_n)$$

Another approach uses

$$\int_{x_n}^{x_{n+1}} y'(x)\,dx = y(x_{n+1}) - y(x_n)$$

or

$$y_{n+1} = y_n + \int_{x_n}^{x_{n+1}} f(y,x)\,dx$$

$$\approx y_n + h\frac{f(y_{n+1},x_{n+1}) + f(y_n,x_n)}{2}$$

using the trapezium rule.

$$y_{n+1} = y_n + h\frac{f(y_{n+1}, x_{n+1}) + f(y_n, x_n)}{2}$$

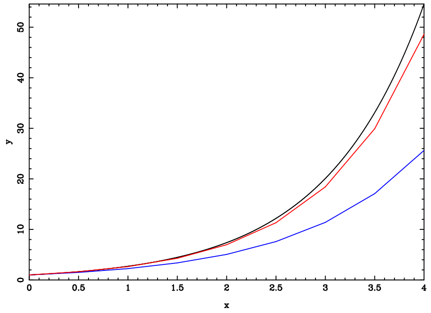
Clearly, we don't know y_{n+1} on the right hand side, but we could use the Euler method to give us a guess.

This gives rise to Heun's method/modified Euler's method/explicit trapezoidal rule:

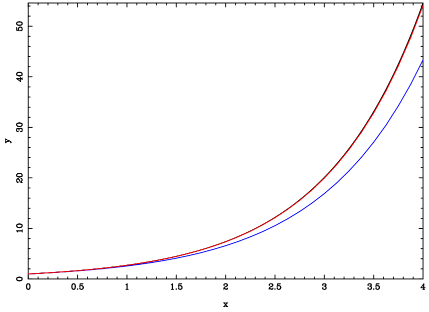
$$\tilde{y}_{n+1} = y_n + hf(x_n, y_n)$$

$$y_{n+1} = y_n + h\frac{f(\tilde{y}_{n+1}, x_{n+1}) + f(y_n, x_n)}{2}.$$

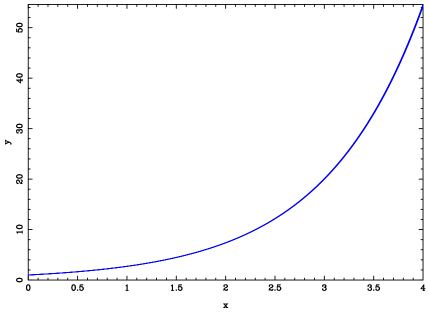
Solutions to $y' = y$, $y(0) = 1$ with step length $h = 1/2$



Solutions to $y' = y$, $y(0) = 1$ with step length $h = 1/8$



Solutions to $y' = y$, $y(0) = 1$ with step length $h = 1/256$



Although Heun's method is more involved, you need to do far less computation for a given amount of accuracy.

There are many more methods out there with better accuracy, better stability, better robustness, ...

Solving Partial Differential Equations

This is another area where computational mathematics plays an important rôle. Many partial differential equations are difficult or impossible to solve except with the use of a computer.

There are different types of partial differential that must be solved in very different ways. All beyond the scope of this course.

But one example that that shows how different aspects of computational mathematics can be inter-related...

Solving the Heat Equation

The 1-dimensional heat equation for the temperature $T(x,t)$ in a rod is

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2}$$

Suppose we approximate T

$$T(x,t) \approx \sum_{n=0}^{N} a_n(t) T_n(x)$$

then substituting into the heat equation gives

$$\sum_{n=0}^{N} a_n'(t) T_n(x) = \kappa \sum_{n=0}^{N} a_n(t) T_n''(x)$$

Since $T_n(x)$ is a polynomial of order n, $T_n''(x)$ is a polynomial of order $n-2$, and can be expressed in terms of a sum of lower order Chebyshev polynomials.

After some algebra, and some difficulties to do with the end conditions, you can end up with a system of equations

$$\begin{pmatrix} a'_n(t) \\ a'_{n-1}(t) \\ a'_{n-2}(t) \\ \vdots \\ a'_1(t) \\ a'_0(t) \end{pmatrix} = M \begin{pmatrix} a_n(t) \\ a_{n-1}(t) \\ a_{n-2}(t) \\ \vdots \\ a_1(t) \\ a_0(t) \end{pmatrix}$$

for some matrix M.

This is now a system of ordinary differential equations which you can solve using your favourite technique.

Does any of this mean you can get away with not knowing some maths, and rely on the computer to solve your problems?

Does any of this mean you can get away with not knowing some maths, and rely on the computer to solve your problems?

No!!!

If you don't understand the task the computer is undertaking you cannot guarantee you will do it right, and if you don't have sufficient understanding you will not spot it when things are wrong.

Computers are tools to help you, but you still have to know your mathematics first!