Lesson 29: Fourier Series and Recurrence Relations

```
> restart;
```

▼ Convergence of Fourier series.

We considered the following function on the interval $[0, 2\pi)$

```
> f:= t -> t^2;
```

$$f := t \rightarrow t^2 \tag{1.1}$$

We extended it to be periodic using the following "saw" function

```
> saw:= x -> x - 2*Pi*floor(x/(2*Pi));
```

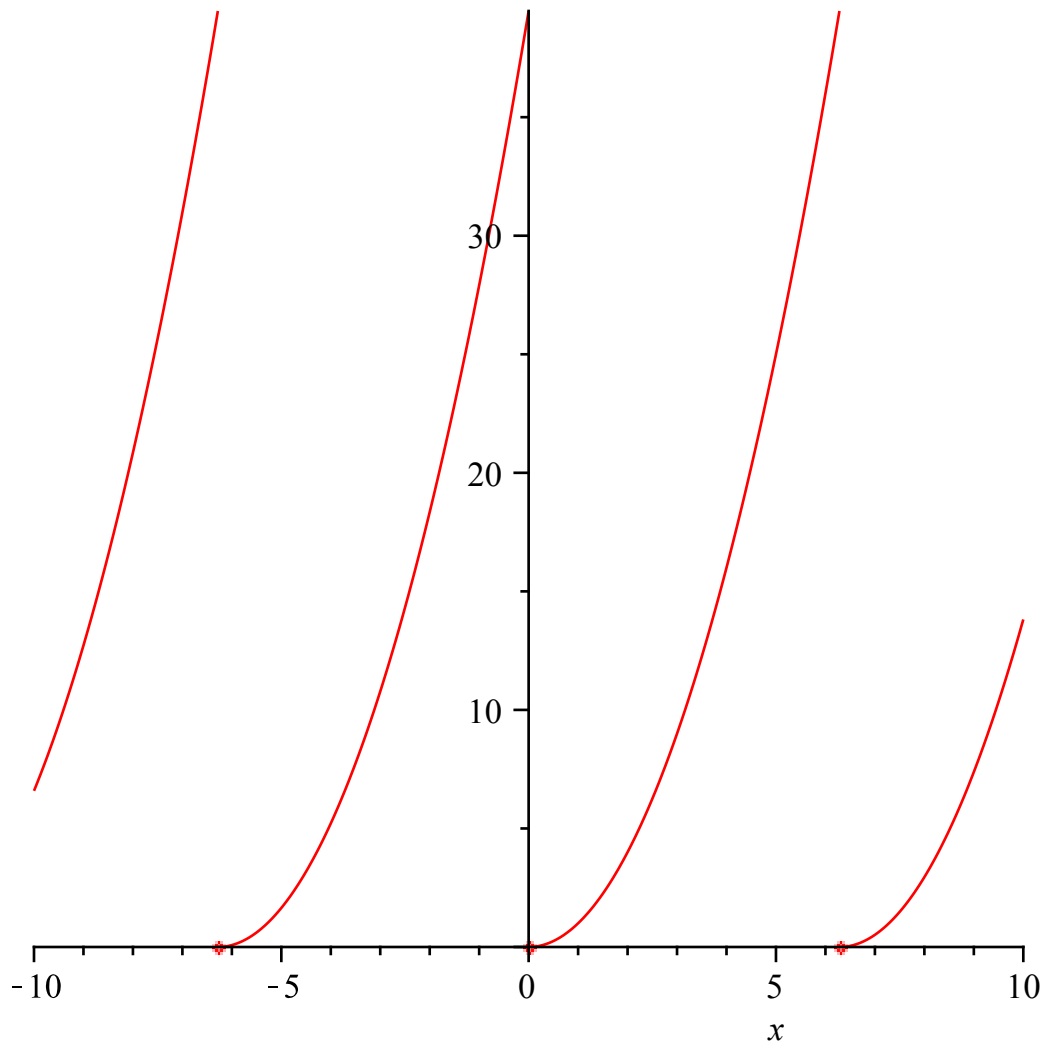$$saw := x \rightarrow x - 2\,\pi\,\mathrm{floor}\!\left(\frac{1}{2}\,\frac{x}{\pi}\right) \tag{1.2}$$

```
> fper:= f @ saw;
```

$$fper := f@saw \tag{1.3}$$

```
> plot(fper(x), x=-10 .. 10, discont=true);
```

The Fourier series of f is $\dfrac{a_0}{2} + \displaystyle\sum_{k=1}^{\infty}\left(a_k \cos(n\,x) + b_k \sin(n\,x)\right)$ where

$a_k = \dfrac{1}{\pi} \displaystyle\int_0^{2\pi} f(x)\,\cos(k\,x)\;\mathrm{d}x$ and $b_k = \dfrac{1}{\pi} \displaystyle\int_0^{2\pi} f(x)\,\sin(k\,x)\;\mathrm{d}x$.

```
> a := unapply( 1/Pi * int(f(t)*cos(k*t),t=0..2*Pi),k) assuming
  k::integer;
  a(0):=  1/Pi * int(f(t)*cos(0*t),t=0..2*Pi);
  b:= unapply( 1/Pi * int(f(t)*sin(k*t),t=0..2*Pi),k) assuming
  k::integer;
```

$$a := k \rightarrow \frac{4}{k^2}$$

$$a(0) := \frac{8}{3}\,\pi^2$$

$$b := k \rightarrow -\frac{4\,\pi}{k} \qquad \text{(1.4)}$$

The theoretical result is that this Fourier series converges to the periodic version $fper(x)$ for every x

where *fper* is continuous,while at points where *fper* has a jump discontinuity with limits from the left and right, it converges to the average of those limits. Here is a function that will produce partial sums.
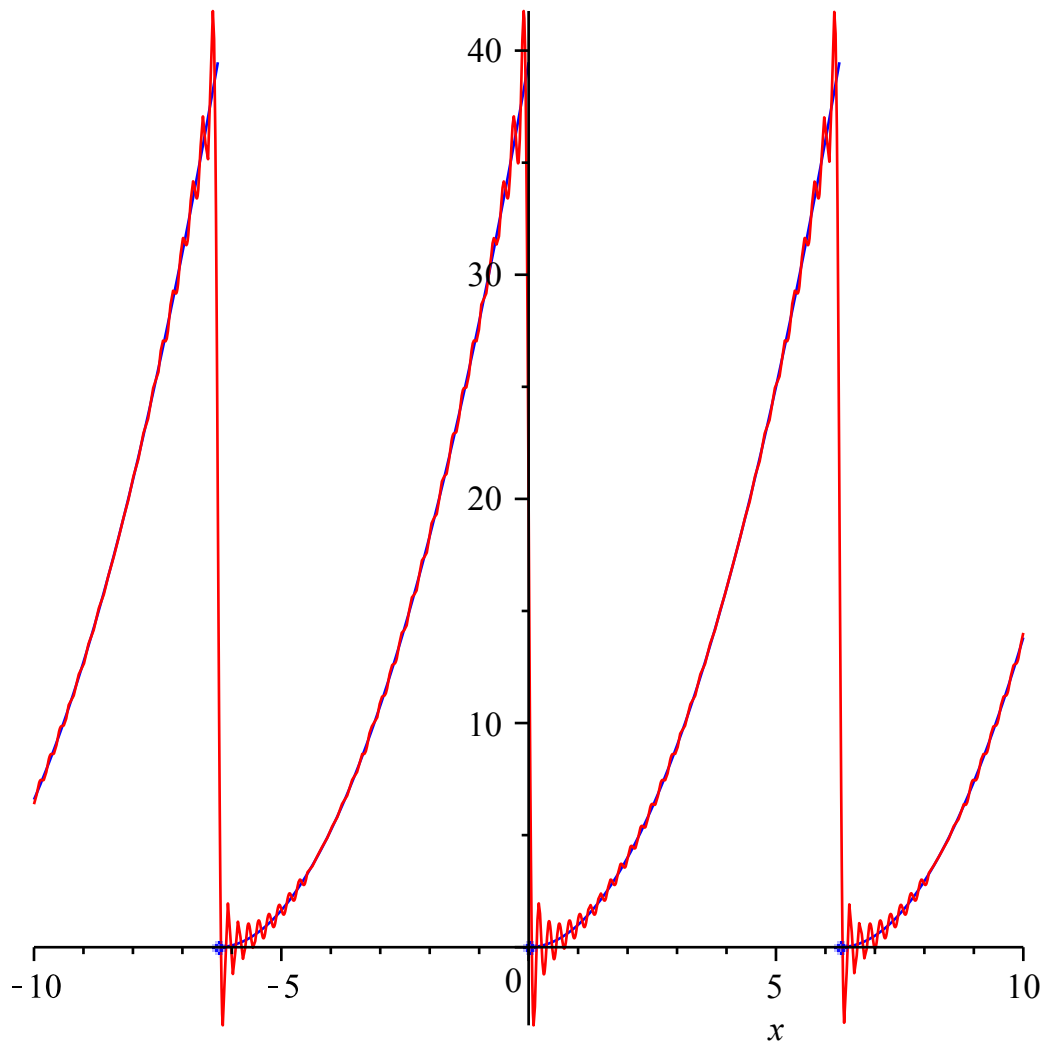
```
> Psum:= N -> a(0)/2 + add(a(k)*cos(k*x)+b(k)*sin(k*x),k=1..N);
```

$$Psum := N \rightarrow \frac{1}{2} a(0) + add(a(k) \cos(k x) + b(k) \sin(k x), k = 1 .. N)$$

(1.5)

```
> plot([fper(x),Psum(3)], x=-10 .. 10, discont=true, colour=
  [blue, red]);
```
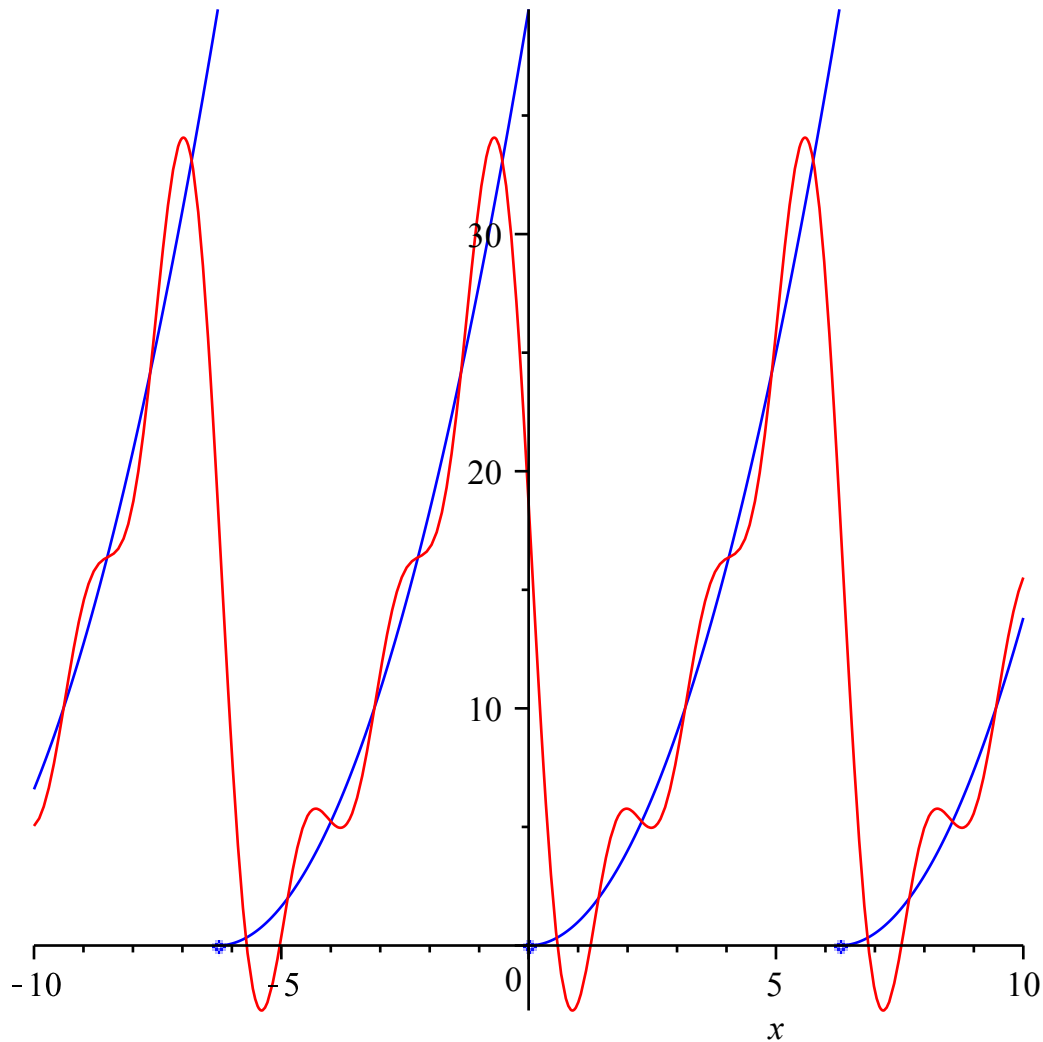


```
> plot([fper(x),Psum(30)], x=-10 .. 10, discont=true, colour=
  [blue, red]);
```

The jumps in *fper* take place at the multiples of 2 π, where the partial sum is not terribly far from the average of the two limits.

Away from the jumps, the partial sum seems pretty close to the function. But something strange happens near the jumps.

```
> eval(Psum(30),x=0); evalf(%); evalf(limit(fper(x),x=0,left));
```

$$\frac{4}{3} \pi^2 + \frac{87453633414459603339103 69}{135616454788597 3785960000}$$

$$19.60807301$$

$$39.47841762 \qquad \textbf{(1.6)}$$

It's easier to see by plotting the difference between the function and the partial sum.

```
> with(plots):
  display([seq(plot(fper(x)-Psum(N), x=-10 .. 10,y=-1..1,
  discont=true), N=1..30)], insequence=true);
```
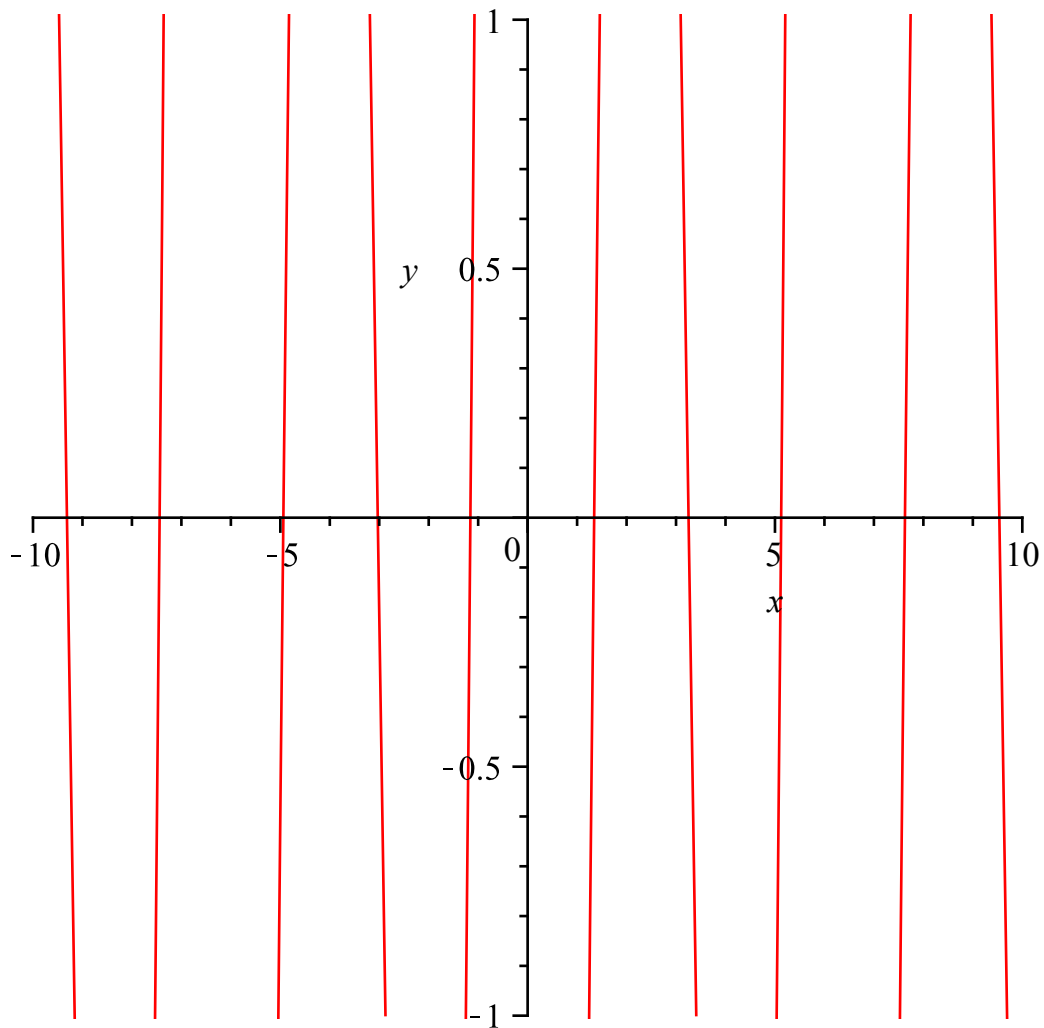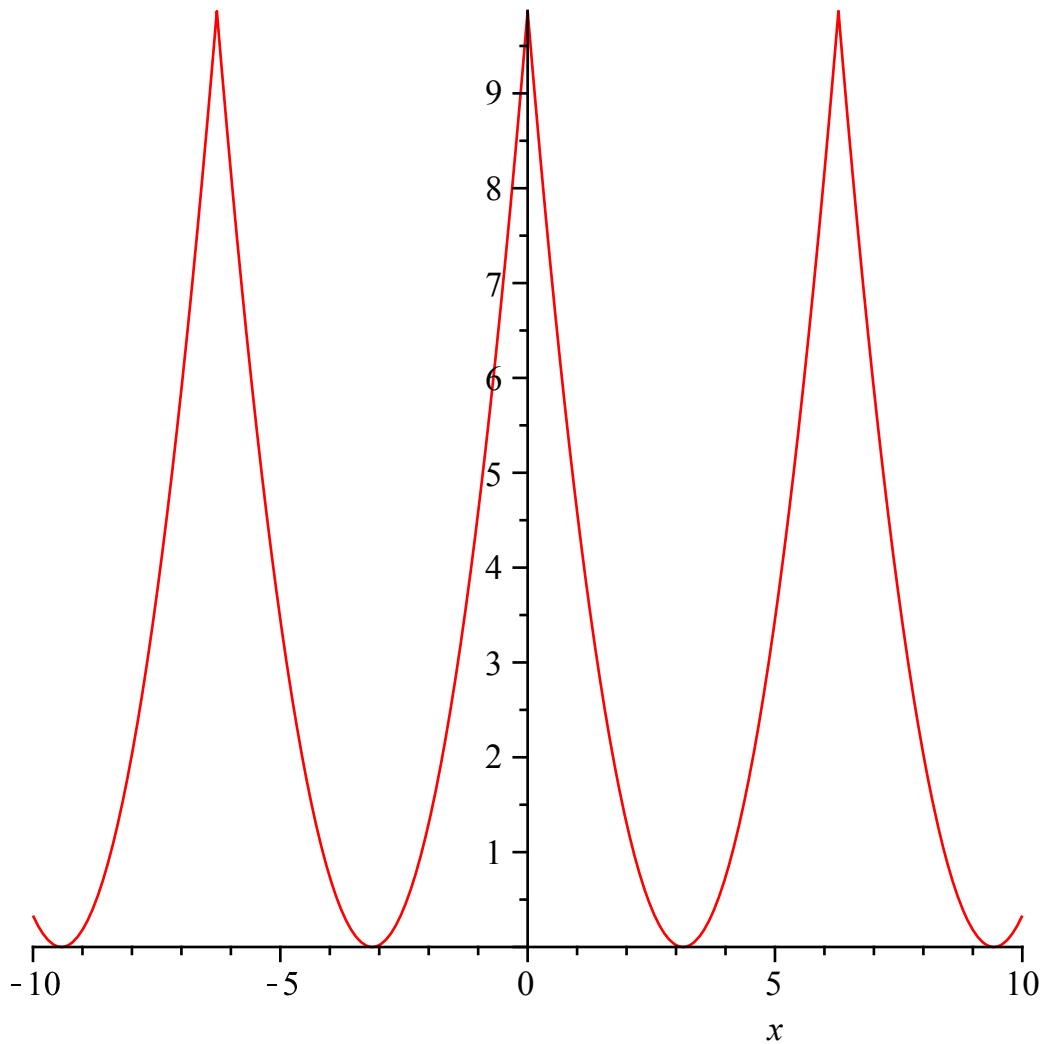
The fact that the "wiggles" don't go to 0 in magnitude is called the Gibbs phenomenon.
It's typical of a function with jumps that some of the coefficients a_k and b_k go to 0 slowly, only
$O\left(\dfrac{1}{k}\right)$, and the convergence of the partial sums to the function is not very good. With a nice
continuous function it should be better.

```
> f:= x -> (x - Pi)^2;
```

$$f := x \rightarrow (x - \pi)^2 \tag{1.7}$$

```
> plot(fper(x),x=-10..10);
```

```
> a := unapply( 1/Pi * int(f(t)*cos(k*t),t=0..2*Pi),k) assuming
  k::integer;
  a(0):=  1/Pi * int(f(t)*cos(0*t),t=0..2*Pi);
  b:= unapply( 1/Pi * int(f(t)*sin(k*t),t=0..2*Pi),k) assuming
  k::integer;
```
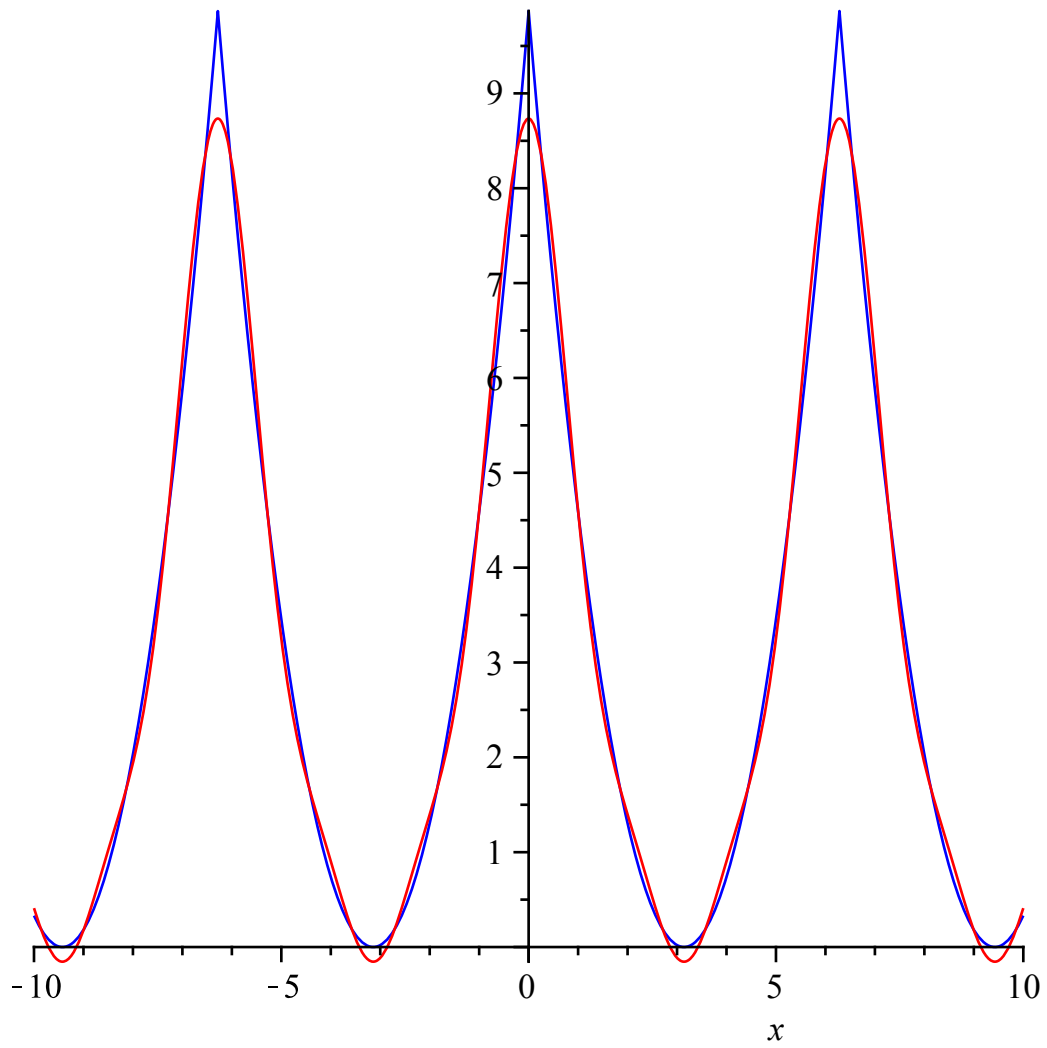
$$a := k \rightarrow \frac{4}{k^2}$$

$$a(0) := \frac{2}{3}\,\pi^2$$

$$b := k \rightarrow 0 \tag{1.8}$$
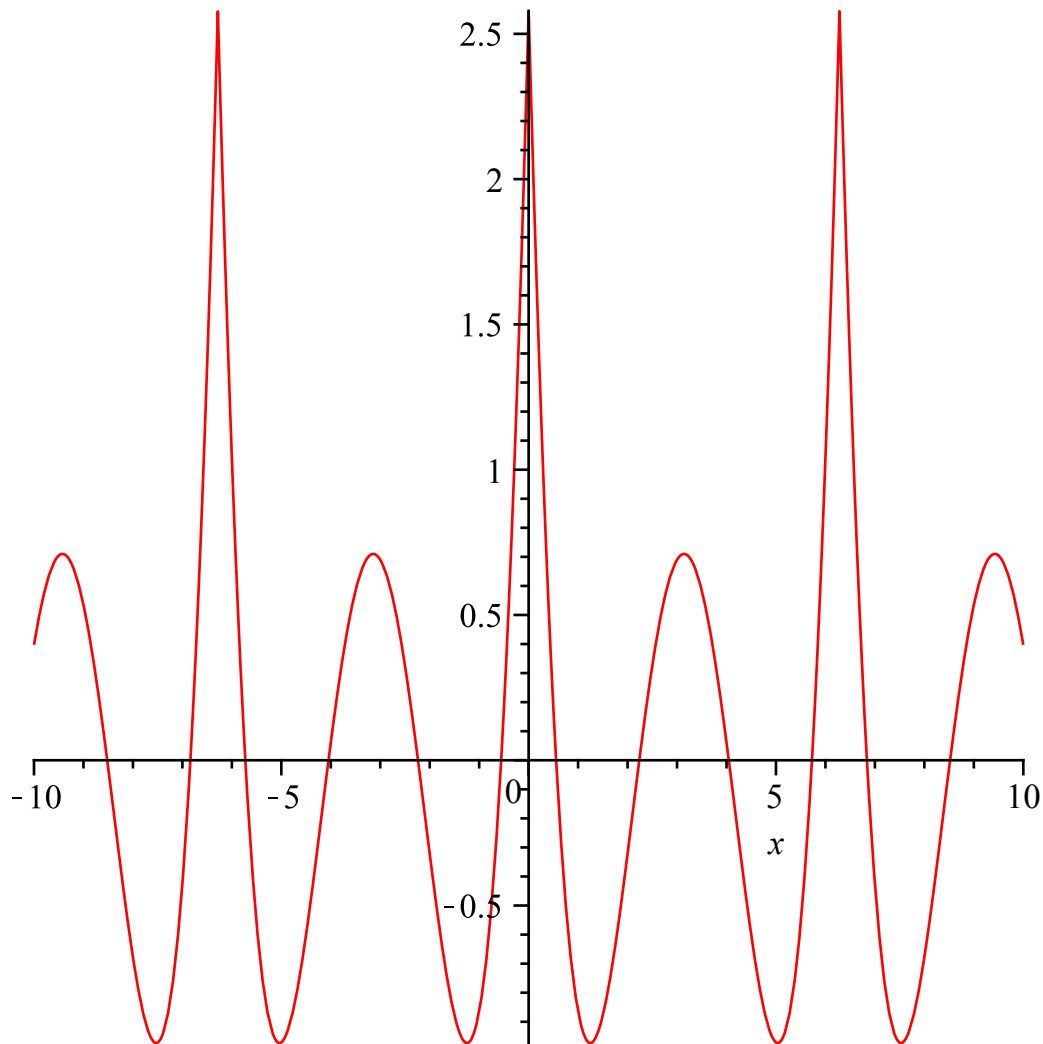
Note that in this case $a_k = O\left(\dfrac{1}{k^2}\right)$ and $b_k = 0$ (by symmetry).

```
> plot([fper(x),Psum(3)], x=-10 .. 10, colour=[blue, red]);
```

```
> display([seq(plot(fper(x)-Psum(N), x=-10 .. 10), N=1..30)],
  insequence=true);
```

The worst places for convergence are still near the multiples of 2π: there's no jump there, but there is a sharp corner: the function is not differentiable there. Basically the more differentiable the function is the faster the coefficients should go to 0, and the faster the partial sums will converge to the function. The most differentiable functions are analytic functions.

```
> f:= x -> 1/(2+cos(x));
```

$$f := x \rightarrow \frac{1}{2 + \cos(x)} \tag{1.9}$$

```
> a := unapply( 1/Pi * int(f(t)*cos(k*t),t=0..2*Pi),k) assuming
  k::integer;
  b := unapply( 1/Pi * int(f(t)*sin(k*t),t=0..2*Pi),k) assuming
  k::integer;
```

$$a := k \rightarrow \frac{\displaystyle\int_{0}^{2\pi} \frac{\cos(k\,t)}{2 + \cos(t)}\, dt}{\pi}$$

$$\tag{1.10}$$

$$b := k \rightarrow \frac{\displaystyle\int_0^{2\pi} \frac{\sin(k\,t)}{2+\cos(t)}\,dt}{\pi} \tag{1.10}$$

```
> seq(a(k),k=0..5);
```

$$\frac{2}{3}\sqrt{3}, \frac{2\pi - \frac{4}{3}\pi\sqrt{3}}{\pi}, \frac{-8\pi + \frac{14}{3}\pi\sqrt{3}}{\pi}, \frac{30\pi - \frac{52}{3}\pi\sqrt{3}}{\pi}, \tag{1.11}$$

$$\frac{-112\pi + \frac{194}{3}\pi\sqrt{3}}{\pi}, \frac{418\pi - \frac{724}{3}\pi\sqrt{3}}{\pi}$$

```
> seq(b(k),k=0..5);
```

$$0, 0, 0, 0, 0, 0 \tag{1.12}$$

In this case I don't know if there's a nice formula for the a_k. But according to theory they should be $O(\exp(-r\cdot k))$ for some $r > 0$.

```
> Digits:= 20: A:=[seq(evalf(Int(cos(k*t)/(2+cos(t)), t = 0 ..
  2*Pi)/Pi),k=1..30)];
```

$$A := \big[\, -0.30940107675850305803, 0.082903768654760703128, \tag{1.13}$$

$$-0.022213997860539754476, 0.0059522227873983147748,$$

$$-0.0015948932890535046237, 0.00042735036881570372001,$$

$$-0.00011450818620931025631, 0.000030682376021537305205,$$

$$-0.0000082213178768389645195, 0.0000022028954858185528703,$$

$$-5.9026406643524696723 \times 10^{-7}, 1.5816077992243498765 \times 10^{-7},$$

$$-4.2379053254492980451 \times 10^{-8}, 1.1355433095536934979 \times 10^{-8},$$

$$-3.0426791276547709876 \times 10^{-9}, 8.1528341508213271605 \times 10^{-10},$$

$$-2.1845453267374481481 \times 10^{-10}, 5.8534715612846913579 \times 10^{-11},$$

$$-1.5684329777650987654 \times 10^{-11}, 4.2026034977617283949 \times 10^{-12},$$

$$-1.1260842133255555555 \times 10^{-12}, 3.0173335557469135802 \times 10^{-13},$$

$$-8.0849208963148148146 \times 10^{-14}, 2.1663480252530864197 \times 10^{-14},$$

$$-5.8047120409259259258 \times 10^{-15}, 1.5553679011728395062 \times 10^{-15},$$

$$-4.1675957432098765431 \times 10^{-16}, 1.1167040679012345679 \times 10^{-16},$$

$$-2.9921981419753086419 \times 10^{-17}, 8.0175479012345679010 \times 10^{-18}\,\big]$$

```
> pointplot([seq([n,ln(abs(A[n]))],n=1..30)]);
```
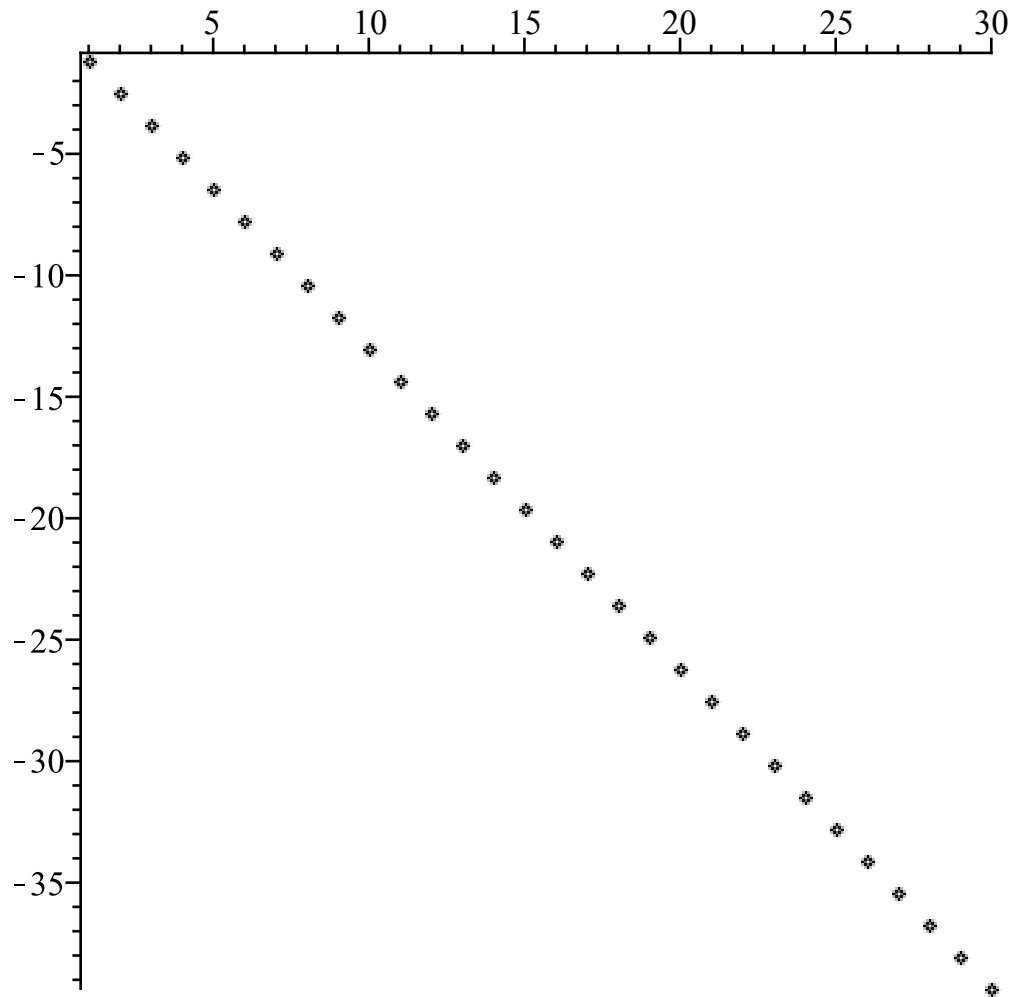
Recurrence Relations

One very useful way to define a sequence a_n is by a recurrence relation, which is an equation expressing a_n in terms of the a_k for $k < n$. For example,

$a_n = 2\, a_{n-1}$

This alone doesn't determine the a_n. You also need an initial condition: for example, a value for a_0.

```
> a[0]:= 1;
  for nn from 1 to 6 do
    a[nn] := 2*a[nn-1]
  end do;
```

$$a_0 := 1$$
$$a_1 := 2$$
$$a_2 := 4$$
$$a_3 := 8$$
$$a_4 := 16$$

$$a_5 := 32$$
$$a_6 := 64$$

Obviously in this case we'll have $a_n = 2^n$ for all n.

Here's a slightly more complicated relation:
$$F_n = F_{n-1} + F_{n-2}$$
$$F_0 = 0, F_1 = 1$$
Note that here we need two initial values to determine the sequence.

```
> F[0]:= 0;
  F[1]:= 1;
  for count from 2 to 10 do
     F[count]:= F[count-1] + F[count-2]
  end do;
```
$$F_0 := 0$$
$$F_1 := 1$$
$$F_2 := 1$$
$$F_3 := 2$$
$$F_4 := 3$$
$$F_5 := 5$$
$$F_6 := 8$$
$$F_7 := 13$$
$$F_8 := 21$$
$$F_9 := 34$$
$$F_{10} := 55$$

This is a famous sequence, known as the Fibonacci numbers (recently featured in *The DaVinci Code*)

Here's a combinatorial problem where the Fibonacci numbers come up.
Suppose there are n Math 210 classes in the term.
You might skip some of the classes, but you don't want to ever miss two consecutive classes. How many different attendance patterns can you have? e.g. with $n = 3$, here are the 5 possible patterns (0 = skip, 1 = attend):
010, 011, 101, 110, 111.

You can get a recurrence relation this way. Let $A(n)$ be the number of patterns for n classes. If you attend the first class, you have $A(n-1)$ possible patterns for the remaining $n-1$ classes. On the other hand, if you skip the first class, you must attend the second; after that you have $A(n-2)$ possible patterns for the remaining $n-2$ classes.
So $A(n) = A(n-1) + A(n-2)$.
The initial conditions are $A(1) = 2$ and $A(0) = 1$ (or if you don't think $A(0)$ should be defined, $A(2) = 3$). Note that $A(n) = F_{n+2}$ for $n = 0, 1, 2$. By mathematical induction, it's easy to prove

$A(n) = F_{n+2}$ for all positive integers n.

▼ Calculating Fibonacci

How can we calculate the Fibonacci numbers? The most straightforward way is with a **for** loop as above, that grinds through them one by one.

Here's another way that looks neater. Instead of explicitly calculating F_k for k from 1 to n, get Maple to do it automatically when needed.

```
> fib1:= n -> fib1(n-1) + fib1(n-2);
  fib1(0):= 0; fib1(1):= 1;
```

$$fib1 := n \rightarrow fib1(n-1) + fib1(n-2)$$
$$fib1(0) := 0$$
$$fib1(1) := 1$$

This is taking advantage of the remember table to supply the values of **fib1(0)** and **fib1(1)** rather than using the recurrence for them. Don't forget to define those values or you'll get an infinite recursion. Also don't try **fib1(n)** where **n** is not a positive integer.

```
> fib1(6);
```
$$8$$

```
> fib1(-1);
```
Error, (in fib1) too many levels of recursion

But this is a really terrible way to calculate the Fibonacci numbers, unless n is really small. It keeps doing the same calculations many times.

Let $T(n)$ be the number of steps **fib1(n)** takes to do the calculation, where each step is either an addition (according to the recursive formula) or remembering the value of **fib1(0)** or **fib1(1)**. This satisfies the recurrence $T(n) = 1 + T(n-1) + T(n-2)$, with $T(0) = 1$ and $T(1) = 1$. It looks rather similar to the Fibonacci recurrence, and the solution is related. We can write the recurrence relation as

$$T(n) + 1 = (T(n-1) + 1) + (T(n-2) + 1)$$

which looks more like Fibonacci. Thus if $g(n) = T(n) + 1$, $g(n)$ satisfies the same recurrence relation as the Fibonacci numbers. The initial conditions, though, are different: $g(0) = 2$ and $g(1) = 2$. If you notice $F_1 = 1$ and $F_2 = 1$, you can see that $g(n) = 2 F_{n+1}$, so $T(n) = 2 F_{n+1} - 1$. These numbers grow rather rapidly.

A better way would be to have Maple remember the values of each Fibonacci number it calculates. A Maple procedure will do this if you specify **option remember** in its definition. You can't use **->** to define such a procedure: instead you use **proc**:

```
> fib2:= proc(n)
    option remember;
    fib2(n-1)+fib2(n-2);
  end proc;
```
$$fib2 := \mathbf{proc}(n) \ \mathbf{option} \ remember; \ fib2(n-1) + fib2(n-2) \ \mathbf{end \ proc}$$

We still need to put the values for 0 and 1 in the remember table.

```
> fib2(0):= 0: fib2(1):= 1:
```

```
> fib2(300);
```
$$222232244629420445529739893461909967206666939096499764990979600$$

To calculate **fib2(n)**, Maple first needs to calculate **fib2(n-1)**, and in the process of doing so it calculates and remembers **fib2(n-2)**. Once it has **fib2(n-1)** it needs to add **fib2(n-2)** but it just remembers this instead of calculating it again. So the number of calculation steps is something like $2\,n$.

▼ Fibonacci numbers using linear algebra

Our next method expresses the recurrence relation in terms of vectors and matrices.

Consider the column vector $X(n) = \begin{bmatrix} F_{n-1} \\ F_n \end{bmatrix}$. So also

```
> X(n) = <F[n-1],F[n-2]+F[n-1]>;
```

$$X(n) = \begin{bmatrix} F_{n-1} \\ F_{n-2}+F_{n-1} \end{bmatrix}$$

This can be written as $X(n) = M.X(n-1)$ for a certain matrix M:

```
> M := <<0,1>|<1,1>>;
```

$$M := \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$$

```
> M . <F[n-2],F[n-1]>;
```

$$\begin{bmatrix} F_{n-1} \\ F_{n-2}+F_{n-1} \end{bmatrix}$$

Starting with

```
> X(1) = <0,1>;
```

$$X(1) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

we'll have

```
> X(2) = M.<0,1>;
```

$$X(2) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

and then $X(n) = M^{n-1}.X(1)$. F_n is the second entry of $X(n)$, which is the [2,2] matrix element of M^{n-1}.

```
> fib3:= proc(n) option remember;
     (M^(n-1))[2,2]
   end proc;
```

$$fib3 := \mathbf{proc}(n)\ \mathbf{option}\ remember;\ (M^{\wedge}(n-1))[2,2]\ \mathbf{end\ proc}$$

```
> fib3(300);
   fib3(300)-fib2(300);
```

$$222232244629420445529739893461909967206666939096499764990979600$$

▼ **Maple objects introduced in this lesson:**

option remember