Approximating Functions

We have seen how computers can represent numbers approximately using floating-point (and other) representations.

But how does the computer do arithmetic operations, and compute functions such as log and sin?

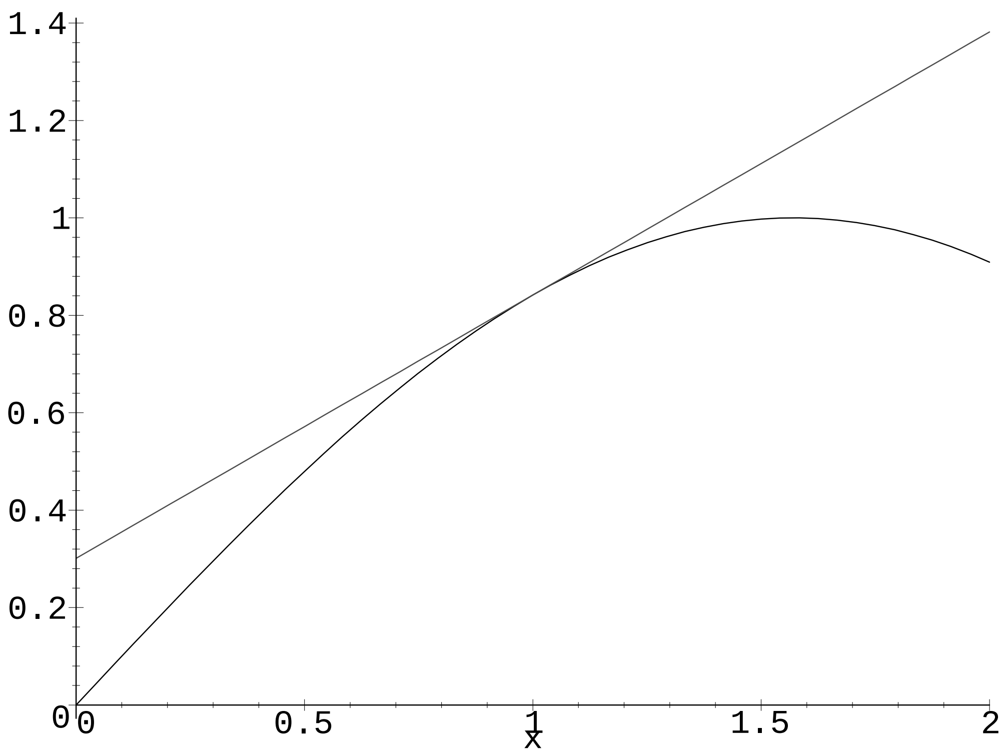
Basic arithmetic ($+$, $-$, $\times$, $/$) is often done in about the same way people do it by hand, though more sophisticated ways are used in the fastest machines.

Functions such as log and sin can be computed in several ways. We will look at one way — by using a *polynomial approximation*, found by the *Taylor series*.

Linear approximation

A crude way to approximate a function, $f(x)$, is to use a line that is tangent to the function at $x = a$: $f(x) \approx f(a) + f'(a)(x - a)$.

Example: we can approximate $\sin(x)$ by a line tanget at $x = 1$, given by $\sin(1) + \cos(1)(x - 1)$



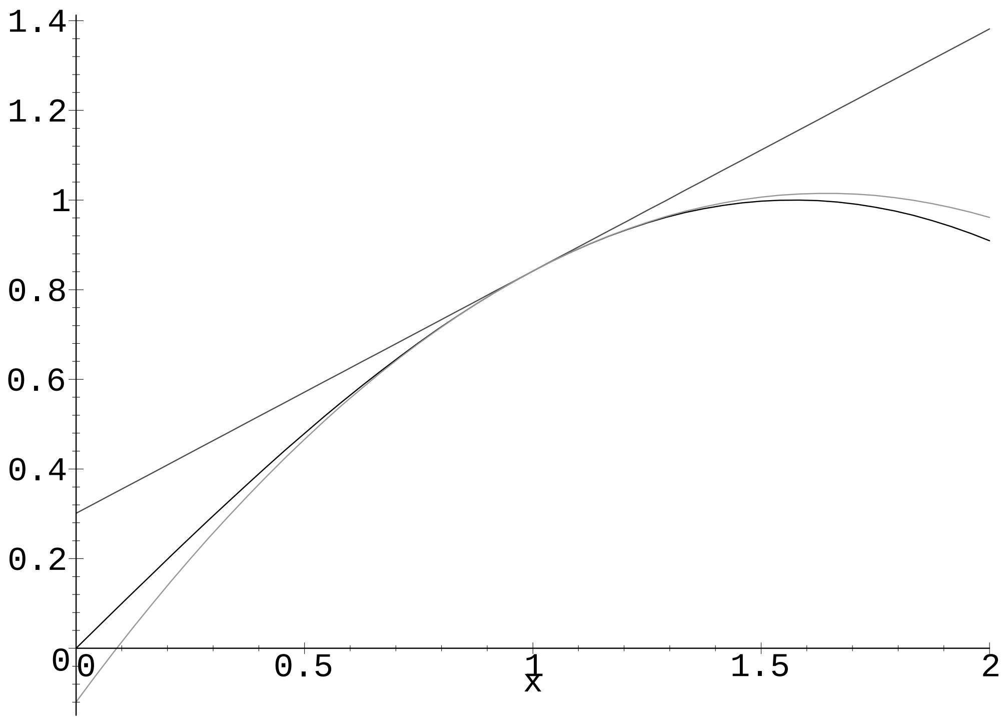
This approximation works OK very near $x = 1$, but not very well farther away.

An aside: Is it cheating to use the values of sin(1) and cos(1) in the approximation?

Quadratic Approximation

We can do better than linear approximation by going to a quadratic (second-order) function.

Here we approximate $\sin(x)$ near $x = 1$ by both linear and quadratic functions:



This quadratic approximation is given by

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

Why should this be the right formula?

Approximation by Taylor Polynomials

In general, we might decide to approximate $f(x)$ by a function that matches its derivatives at $x = a$ up to order n, and whose derivatives of higher order are zero.

This approximation is the *Taylor polynomial* of degree n about $x = a$:

$$f(a) \; + \; f'(a)(x-a) \; + \; \frac{f''(a)}{2}(x-a)^2 \; + \; \cdots \; + \; \frac{f^{(n)}(a)}{n!}(x-a)^n$$

For $f(x) = \sin(x)$, the second degree Taylor polynomial about $x = 1$ is the quadratic approximation we saw before:

$$\sin(1) \; + \; \cos(1)(x-1) \; - \; \frac{\sin(1)}{2}(x-1)^2$$

Taylor's Theorem

How accurate are the Taylor Polynomials as approximations?

Taylor's Theorem tells us this. Assuming all the needed derivatives are defined:

$$f(x) = f(a) + f'(a)(x-a) + \cdots + \frac{f^{(n)}(a)}{n!}(x-a)^n + \frac{f^{(n+1)}(c)}{(n+1)!}(x-a)^{n+1}$$

for some value of c between a and x. *Note that c may depend on x.*

The first terms above are the nth degree Taylor polynomial. The last term is the "remainder". If we can put a limit on how big the remainder might be, we will have put a limit on the error in our approximation.

Does the Infinite Taylor Series Converge to the Right Answer?

Can we always approximate $f(x)$ well by a Taylor polynomial about $x = a$ of high enough order?

In other words, does the infinite *Taylor series*:

$$f(x) \;=\; f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots$$

always converge?

No, it doesn't always converge. Sometimes it converges in some neighborhood of a, but not for all x. For $\log(x)$ about $x = 1$, Maple tells us

```
> taylor(log(x),x=1);
```

$$(x-1) \;-\; \tfrac{1}{2}(x-1)^2 \;+\; \tfrac{1}{3}(x-1)^3 \;-\; \tfrac{1}{4}(x-1)^4 \;+\; \tfrac{1}{5}(x-1)^5 \;+\; O((x-1)^6)$$

This doesn't converge for $x > 2$. If $f(x)$ is *analytic*, the Taylor series will converge for x close enough to a.

Piecewise Approximations

How can we approximate $f(x)$ if the Taylor series doesn't converge for all x?

One possibility is to use a Taylor series about different points in different regions — giving a *piecewise* approximation.

We might do this even if the Taylor series converges everywhere, because it might converge slowly far from a.

A similar idea is to transform the problem using properties of the function. For example:

$$\begin{aligned} \sin(x) &= \sin(x + 2\pi) \\ \log(x) &= \log(ex) - 1 \end{aligned}$$

Example: Taylor Series for Sin(x)

Let's find the Taylor series for $\sin(x)$ about $x = 0$:

$$\begin{aligned} & \sin(0) + \sin'(0)x + \frac{\sin''(0)}{2!}x^2 + \frac{\sin'''(0)}{3!}x^3 + \cdots \\ & = \sin(0) + \cos(0)x - \frac{\sin(0)}{2!}x^2 - \frac{\cos(0)}{3!}x^3 + \cdots \\ & = x - \frac{1}{3!}x^3 + \frac{1}{5!}x^5 - \frac{1}{7!}x^7 + \cdots \end{aligned}$$

This uses $\sin'(x) = \cos(x)$ and $\cos'(x) = -\sin(x)$.

Does this series converge for all x?

How fast does it converge?

What would be a good strategy for using this series to compute $\sin(x)$ for any x?

Using Taylor Series in Practice

To use Taylor series to compute functions, we need to solve two problems:

- How can we tell when we've added enough terms (ie, how high a degree Taylor polynomial do we need)?

- How much effect does rounding error have on the result? Can this round-off error be reduced?

The solutions to these problems can be related. We will look at three possible approaches. There are many more sophisticated methods that we won't look at!

Stopping for Alternating Series

Suppose the terms in the series have alternating signs (eg, sin(x) about $x = 0$), and that the terms are decreasing in size.

We can then put a limit on how much error their may be in our answer — it can't be more than the next term in the series.

Consider a series with $t_1 > t_2 > t_3 > \cdots$

$$t_1 - t_2 + t_3 - t_4 + t_5 - t_6 + \cdots$$

Suppose we are interested in the error when we use only the first term, t_1. We can see that t_1 is too high by rearranging the sum:

$$t_1 + (-t_2 + t_3) + (-t_4 + t_5) + \cdots$$

All the bracketed terms are negative, so t_1 is above the infinite sum. On the other hand, we see that $t_1 - t_2$ is too small:

$$t_1 - t_2 + (t_3 - t_4) + (t_5 - t_6) + \cdots$$

We can use this to decide when to stop.

Stopping using Taylor's Theorem

We might instead decide to stop when the remainder term in Taylor's Theorem says that our error is small enough.

For the series for $\sin(x)$ about $x = 0$, the remainder term is

$$\frac{\sin^{(n+1)}(c)}{(n+1)!}\,x^{n+1}$$

The $(n+1)$st derivative of sin will be either $\pm\sin(x)$ or $\pm\cos(x)$. We don't know c, other than that it's between 0 and x, but even so, we know the remainer will be no bigger than

$$\pm\frac{x^{n+1}}{(n+1)!}$$

For any x, this will eventually become smaller than our desired error bound.

Summing to Saturation

A simple approach is to keep adding in terms of the series until adding more terms doesn't change the sum, due to saturation.

This is very easy to implement. But does it work?

Consider two series:

$$\begin{aligned} \frac{\pi^2}{6} &= 1 + \frac{1}{4} + \frac{1}{9} + \frac{1}{16} + \cdots \\ 2 &= 1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \cdots \end{aligned}$$

What happens when we sum both to saturation with `Digits` set to 5?

Results of Summing to Saturation

Let's try it:

```
> Digits:=5:

> s:=0:
> for i from 1 while evalf(s+1/i^2)>s do s := evalf(s+1/i^2): od:
> s;
                              1.6390
> evalf(Pi^2/6);
                              1.6450

> s:=0:
> for i from 0 while evalf(s+1/2^i)>s do s := evalf(s+1/2^i): od:
> s;
                              2.0000
```

Why the difference?

What would happen if we added up the terms in the opposite order? Why might that be difficult?

Horner's Rule

Suppose we somehow know beforehand how many terms of a power series to use. What's the best way to compute the sum?

The *easiest* way is probably to sum in order. But it may be *faster* to use "Horner's Rule", illustrated by

$$\begin{aligned} & a_0 + a_1 x + a_2 x^2 + a_3 x^3 \\ & \quad = \; a_0 + x(a_1 + x(a_2 + x(a_3))) \end{aligned}$$

The left side requires 5 multiplies and 3 additions.

The right side requires 3 multiplies and 3 additions

Which method will be more *accurate*?