This diverges if $|x| > 1$ and converges to $a/(1-x)$ if $|x| < 1$.

There are a number of tests used to see if a given series is convergent. We look at one useful test.

2.5.5 The Ratio Test

Consider a series $\sum_{n=0}^{\infty} a_n$, with each $a_n \neq 0$, such that

$$r = \lim_{n\to\infty} \left| \frac{a_{n+1}}{a_n} \right|$$

either exists or is infinite.
Then

$$\begin{array}{lll} \text{if} & r < 1 & \text{the series converges,} \\ \text{if} & r > 1 & \text{the series diverges,} \\ \text{if} & r = 1 & \text{the ratio test is inconclusive.} \end{array}$$

Example 2.19. Prove that $\sum_{n=0}^{\infty} \frac{(-1)^n n}{2^n}$ converges.

Example 2.20. Find if $\sum_{n=1}^{\infty} \frac{2^n}{n^2}$ converges or diverges.

Further examples and exercises:

- (Stewart 2012, pg. 758-63)
- (Morris & Stark 2015, pg. 261-2, Ex. 9.4, Q18-23)

2.6 Taylor series

2.6.1 Approximating functions by polynomials

Consider the logistic function we used in Figure 2.3 to model the probability of surviving the Titanic disaster as a function of passenger age and fare. This is a highly nonlinear model, using special functions. How does the computer know how to evaluate this function so that it can be plotted? At its core, the computer is a fairly dumb machine, being only able to do simple operations like addition and multiplication (and therefore subtraction and division). It has no *a priori* concept of an exponential, much less a logistic, function. In order to evaluate these and other functions, they must be approximated by polynomials. One way to do this is using *Taylor polynomials*, which we now build up as a sequence of approximations.

We can approximate a function at a point by its tangent line. In this section we shall discuss how to achieve better approximations by using, instead of a linear function like a line, quadratic, cubic and higher order polynomials. We will find we can do even better — getting exact infinite *series* for many functions — enabling evaluation for arbitrary accuracy.

We have seen that near a point $(a, f(a))$ we can approximate a differentiable function $f(x)$ by its tangent line whose equation is

$$y = f(a) + f'(a)(x - a)$$

We will call this polynomial the first order Taylor polynomial for f at a. It is denoted by $P_1(x)$ so we have

$$P_1(x) = f(a) + f'(a) \cdot (x - a)$$

We can obtain the same result by finding a degree 1 polynomial P_1 such that the value of P_1 and its first derivative agree with those of f at the point a.

Example 2.21. Find the first order Taylor polynomial for $f(x) = \log(x)$ at $a = 1$ and use it to approximate $\log(1.1)$.

Clearly we might hope to get a better approximation if we used, say, a quadratic polynomial to approximate $f(x)$ near $x = a$. We look for a quadratic which goes through $(a, f(a))$ and has the same value and first and second derivatives at $x = a$.

We get

$$P_2(x) = f(a) + f'(a)(x - a) + \frac{f''(a)}{2}(x - a)^2$$

as the quadratic approximation to f at a.
Notice that in order to determine the three constants in the quadratic $P_2(x)$, there must be three conditions imposed on it; namely the value, the first *and second* derivative at $x = a$.

Example 2.22. Approximate $\log 1.1$ using the quadratic $P_2(x)$.

The nth degree polynomial approximation can be written:

$$P_n(x) = c_0 + c_1(x - a) + c_2(x - a)^2 + \cdots + c_n(x - a)^n$$

and we can use the same procedure as above.

We obtain the formula

$$P_n(x) = f(a)+f'(a)(x-a)+\frac{f''(a)}{2!}(x-a)^2+\frac{f'''(a)}{3!}(x-a)^3+\cdots+\frac{f^{(n)}(a)}{n!}(x-a)^n$$

and this is called the *Taylor Polynomial of degree* n for $f(x)$ at $x = a$.

The special case when $a = 0$ is called the *Maclaurin Polynomial of degree* n for $f(x)$:

$$P_n(x) = f(0) + f'(0)x + \frac{f''(0)}{2!}x^2 + \frac{f'''(0)}{3!}x^3 + \cdots + \frac{f^{(n)}(0)}{n!}x^n .$$

Example 2.23. Find the Taylor polynomial of degree n for $\log x$ with $a = 1$. Use $P_4(x)$ to estimate $\log 1.1$.

Example 2.24. Find the Maclaurin polynomial of degree n for e^x.

It is natural to ask how accurate Taylor approximations are. In general we expect better accuracy as n increases and if $(x - a)$ is small. This is summarised by Taylor's theorem.

Theorem 2.25 (Taylor's Theorem)**.** *Suppose the function* f *has* $(n + 1)$ *derivatives on some interval containing* a *and* x*. Then if*

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

is the n*th Taylor polyomial of* f *at* a*,* $f(x) = P_n(x) + R_n(x)$ *where the remainder*

$$R_n(x) = \frac{f^{(n+1)}(z)}{(n+1)!}(x - a)^{n+1}$$

for some number z *between* a *and* x*.*

Proof: (Stewart 2012, pg. 787-9).

Observe that the remainder (error) is essentially the next $(n + 1$ degree) term in which $f^{(n+1)}$ is evaluated at a point z between a and x. We will use Taylor's Theorem to estimate the error in the approximation of $f(x)$ by $P_n(x)$. Notice that we have

$$\begin{aligned} |f(x) - P_n(x)| &= |R_n(x)| \\ &= \left| \frac{f^{(n+1)}(z)}{(n+1)!}(x - a)^{n+1} \right|. \end{aligned}$$

Assume we can find a constant C such that

$$|f^{(n+1)}(z)| \leqslant C$$

for all z between a and x then we have

$$|f(x) - P_n(x)| \leqslant \frac{C}{(n+1)!}|x - a|^{n+1}$$

and this enables us to estimate the error.

Example 2.26. Determine the accuracy of the use of $P_4(x)$ to estimate $\log(1.1)$.

A natural question to ask is: How many terms are needed in a Taylor polynomial to get a specified accuracy?

Example 2.27. If we use the Maclaurin polynomial $P_n(x)$ for e^x, find the smallest value of n which gives $e = e^1$ to within the accuracy of 0.000005 (i.e 5-figure accuracy)?

Example 2.28. Use a Maclaurin polynomial of degree 3 for $(1+x)^{1/2}$ to approximate $\sqrt{5}$. Estimate the error.

Example 2.29. Find the Maclaurin polynomial of degree $n = 2k$ for $f(x) = \cos x$.

And now, just as we did for finite summations, it's natural to ask: what happens if we keep adding terms, i.e., compute an infinite series of polynomial terms? This is a Taylor series, and just as for the infinite series we've seen already, we should be concerned about the convergence of these series.

Further examples and exercises:

- Taylor polynomial worked examples (Stewart 2012, Sec. 11.11, Examples 1 & 2, pp. 793-5)
- Exercises (Stewart 2012, Sec. 11.11, Q1-26, 30, pp. 798-9.)

2.6.2 Taylor, Maclaurin, and power series

If the function f has derivatives of all orders, we can write $f(x)$ in the form given by Taylor's Theorem for all values of n. Further, if we can show the remainder $R_n(x) \to 0$ as $n \to \infty$ then

$$f(x) = \lim_{n\to\infty} P_n(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!}(x-a)^k$$

The expression on the right hand side is called the *Taylor series* of f at $x = a$. In the special case $a = 0$ we have the *Maclaurin series* of f.

For example, Maclaurin series for e^x, $\cos x, \sin x, \frac{1}{1-x}, (1+x)^k$ are

$$\begin{aligned}
e^x :&\quad 1 + x + \tfrac{x^2}{2!} + \tfrac{x^3}{3!} + \cdots &&= \sum_{n=0}^{\infty} \frac{x^n}{n!} \\
\sin x :&\quad x - \tfrac{x^3}{3!} + \tfrac{x^5}{5!} - \tfrac{x^7}{7!} + \cdots &&= \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{(2n+1)!} \\
\cos x :&\quad 1 - \tfrac{x^2}{2!} + \tfrac{x^4}{4!} - \tfrac{x^6}{6!} + \cdots &&= \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n}}{(2n)!} \\
\frac{1}{1-x} :&\quad 1 + x + x^2 + x^3 + \cdots &&= \sum_{n=0}^{\infty} x^n \\
(1+x)^k &&&= 1 + \sum_{n=1}^{\infty} \binom{k}{n} x^n
\end{aligned}$$

These are examples of infinite series of the form $\sum\limits_{n=0}^{\infty} a_n x^n$. Such series are called **power series**. We start our investigation of power series with (again) **the geometric series** and consider the question of **convergence**.

For the geometric series we have

$$\sum_{n=0}^{\infty} x^n = \frac{1}{1-x} \quad \text{if} \quad |x| < 1, \qquad \text{but} \ \sum_{n=0}^{\infty} x^n \quad \text{diverges} \quad \text{if} \quad |x| \geqslant 1.$$

Theorem 2.30 (Convergence of Power Series)**.** *Consider the power series*

$$\sum_{n=0}^{\infty} a_n x^n.$$

Then either

(1) *the series converges for all values of* x*, or*
(2) *the series converges only for* $x = 0$*, or*
(3) *there exists a number* $R > 0$ *such that* $\sum_{n=0}^{\infty} a_n x^n$ *converges for all* x *wi*
diverges for all x *with* $|x| > R$*.*

The number R *is called the radius of convergence. In case (1) we often write "*$R = \infty$*" and in case (2),* $R = 0$*.*

Example 2.31. The Maclaurin series for $\cos x$ is

$$1 - \frac{x^2}{2} + \frac{x^4}{4} - \cdots = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n}}{(2n)!}.$$

Find its interval of convergence.

The ratio test tells us that this power series converges - but not what it converges to. We would hope it converges to $\cos x$. To see this we consider the remainder formula $f(x) = P_n(x) + R_n(x)$ as $n \to \infty$.

Theorem 2.32 (Taylor Series Representation)**.** *If* $f(x)$ *has derivatives of all orders on some interval containing* a *and if* $\lim_{n\to\infty} R_n(x) = 0$ *for each* x *in the interval, then the Taylor Series for* $f(x)$ *converges to* $f(x)$ *at each* x *in the interval. That is,*

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n.$$

In the case that $a = 0$ *(Maclaurin Series)*

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(0)}{n!} x^n.$$

Example 2.33. Show that the Taylor series of $f(x) = e^x$ converges to $f(x) = e^x$ for all x.

Note: Similarly $R_n(x) \to 0$ for $\sin x, \cos x$ so their Maclaurin series converge to $\sin x, \cos x$ respectively. We have

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots$$

for all x and

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \cdots$$

for all x.

Further examples and exercises:

- (Stewart 2012, Sec. 11.10, all questions, pp. 789-90)

2.6.3 Gradient descent and Taylor series

Why should we are about Taylor series in data science? Yes, they may be fundamental in how computers and calculators compute special functions, but that is not essential to know the details of in the modern age. However, in modern machine learning, Taylor series approximation forms the backbone of one of the major algorithms for attacking problems: *gradient descent.* What is gradient descent? Let's start with an example.

The Wisconsin Breast Cancer Dataset (Wohlberg & Mangasarian, 1990) is a famous early example of pattern recognition (i.e., machine learning) techniques applied to a medical problem – diagnosing tumours as being benign or malignant based on a set of features (in ML language) or predictors (in statistics language). The dataset consists of 569 labelled instances of benign/malignant tumours, and has 32 recorded features of these tumours. Figure 2.4 shows a scatter plot of two of these features, stratified by tumour status.

We're going to build a model based on just these features – more sophisticated approaches would use all 32.

How might we build a model to distinguish between benign and malignant tumours based on this dataset? One very naive idea might be to try and find a line through the origin which best separates between the two classes of tumour. By looking at the data it's clear that such a model cannot do this perfectly (see Figure 2.5 which shows one such model), but it has the advantage of being simple to visualise, and parameterised by one parameter, the slope of the line m.

Naive, but inspired by fairly sophisticated methods like so-called *support vector machines...*

How to choose this value m? As our model is imperfect there will always be some *misclassifications*, which we should try to avoid. One way to choose m might be to minimise the number of misclassifications, or to try and minimise the total distance of all

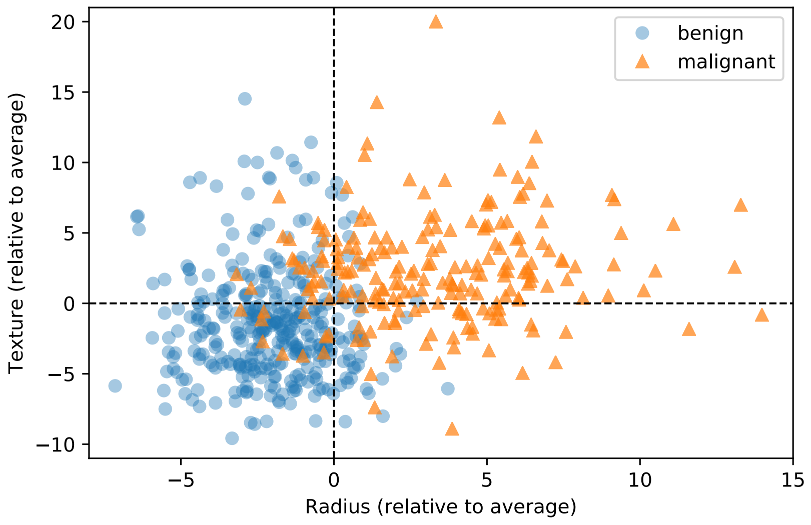

Figure 2.4: *Scatter plot of differences from the mean tumour radius* (x) *versus differences from the mean tumour texture* (y) *for the Wisconsin breast cancer dataset. Blue circles shown benign tumours, orange crosses show malignant tumours.*

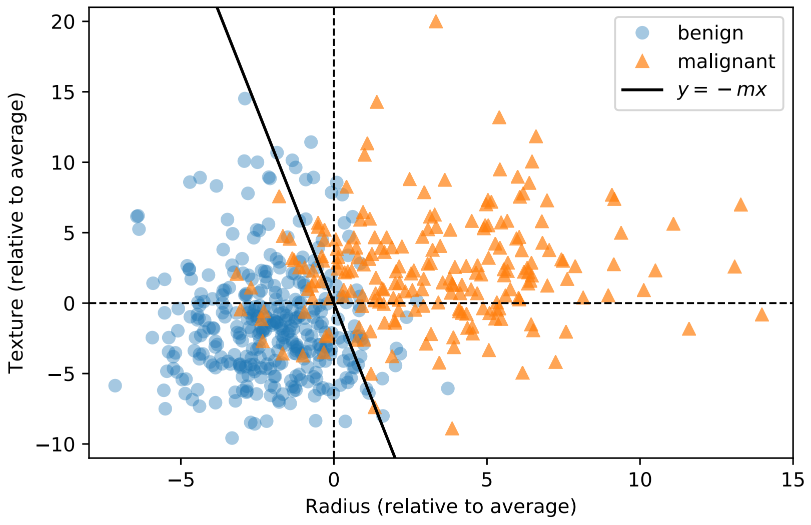

Figure 2.5: *Scatter plot of tumour features, with an example classification model superimposed. In this model points to the right of/above the line would be classified as malignant, while points to the left of/below the line would be classified as benign.*

these misclassifications from our separating line. Inspired by this we can define a *loss function* $l(m)$, which penalises the distances of these misclassifications from the separating line:

$$\begin{aligned} f(m) &= \sum_{i \in \{\text{misclassifications}\}} d\left((x_i, y_i) \text{ from line } y = -mx\right) \\ &= \sum_{i \in \{\text{misclassifications}\}} \frac{|mx_i + y_i|}{\sqrt{m^2+1}}. \end{aligned}$$

There are lots of potential loss functions – part of the challenge of data science in ML is thinking about an appropriate choice.

You don't have to remember the formula for distance of a point from a line – I looked it up on Wikipedia!

Here I'm using i to sum over the misclassifications – the information being used from each misclassified point are the coordinates of that point (x_i, y_i).

The details of this loss function aren't important, except to say that this function is highly unpleasant to deal with. In particular, minimising this function by differentiating and finding critical points (i.e., in the way taught at high school) is impractical, as our function is very messy, involves a different number of terms depending on the number of misclassifications, and lots of arbitrary data points.

So how do we (approximately) find the minimum of $f(m)$ in practice? One key algorithm is *gradient descent*, which essentially goes like this:

1. Guess a solution m.
2. Change m by some amount h, $m \to m + h$, such that
$$f(m+h) < f(m).$$
3. If h is very small, STOP. Otherwise GOTO 2.

That is, we basically head "downwards" on the function $f(m)$ until our steps get sufficiently small that we can stop. This is a simple idea, but now the problem becomes: how do we choose the *step size* h? Taylor series are essentially the answer to this. First, let $x = m + h$, and $a = m$, in the $(n = 1)$ Taylor polynomial formula. This gives us the following approximation for $f(m + h)$:

$$f(m+h) \approx P_1(m+h) = f(m) + f'(m)h.$$

ML people call η the *learning rate.*

If we choose $h = -\eta f'(m)$, for some $\eta > 0$, then we have

$$f(m+h) = f(m) - \eta\left(f'(m)\right)^2 < f(m),$$

because $\left(f'(m)\right)^2 > 0$.

This works well at finding the minimum in a number of cases. Taking the example (from Wikipedia) of finding the minimum of $f(x) = x^4 - 3x^3 + 2$ starting from $x = 6$ gives a reasonable

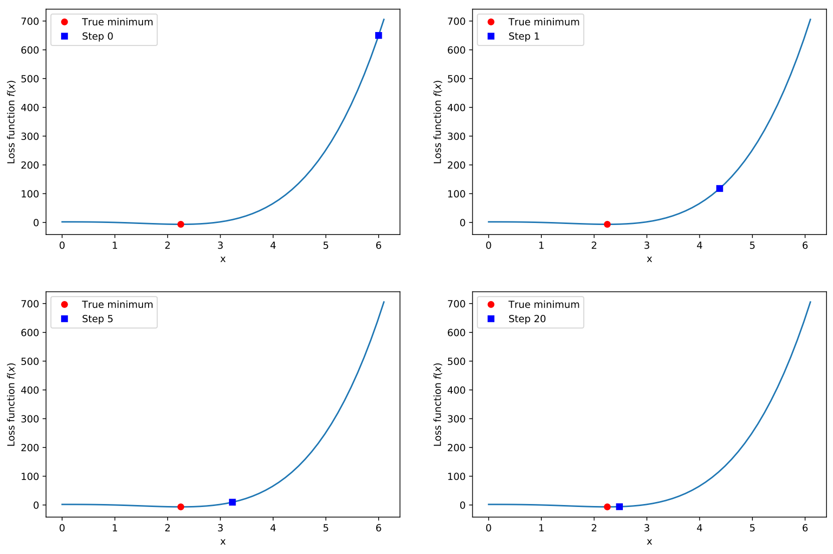

Figure 2.6: Gradient descent to find local minimum of $f(x) = x^4 - 3x^3 + 2$*, using a learning rate of* $\eta = 0.003$*. We show the approximation after 0, 1, 5, and 20, steps.*

approximation to the true solution after less than 20 steps (see Figure 2.6).

There are of course a number of issues here, including:

- We've only discussed gradient descent for a single-valued function. Real ML applications consider high-dimensional functions, necessitating Taylor series (and calculus) of many variables. But the idea is essentially the same.
- Choosing the learning rate η now becomes the major choice – too small and the algorithm runs too slow; too fast and the algorithm overshoots the minimum and doesn't converge. This can be fiddly!
- This will only find *local* minima, not necessarily the *global* minimum we probably want for our ML model (it will be the "best-fitting" one). There is no definite solution to this, but it can be attacked by tuning the learning rate, or through stochastic approaches.
- We still need to know $f'(x)$! This one is relatively easy, as the derivative can be approximated using a finite-difference approach:
$$f'(x) \approx \frac{f(x + \Delta x) - f(x)}{\Delta x}.$$

Note that we've only talked about a first-order Taylor polynomial approximation here – you can build more sophisticated methods using higher-order Taylor approximations. These can potentially converge faster, at the expense of greater complexity. This has been

If you liked this, you might also enjoy APP MATH 3014 Optimisation III.

a contrived example, but it is nonetheless relevant in real machine learning – the “model” we presented in Figure 2.5 has analogies with methods used in real data science, e.g., Linear Discriminant Analysis (LDA) and Support Vector Machines (SVMs).