Approximations & the Derivation of the Taylor Series

Dominic Vicharelli

Abstract

Often times in mathematics, seemingly complex formulas seem to arise from thin air with no solid explanation as to where they come from. For example, the Maclaurin series representation of e^x is one of the most well known facts in higher mathematics with applications in many different fields. For those unaware, we can represent e^x as an infinite sum of polynomials, namely:

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots \tag{1}$$

This paper takes a very in-depth look at how we achieve this seemingly abstract and unintuitive equation by deriving several 'taken for granted' topics in math and slowly building our foundation upwards.

Graphing Lines & Slope

We start with the equation of a line. Most math students recall the slope-intercept equation of a line which is given by $y = mx + b$. This equation is nice because it gives a visual representation of the slope (m) and the y-intercept (b) of any given line. What's important to note is that we can determine the equation of any distinct line with only these 2 parameters: its slope and its y-intercept. But in reality, we don't really care about the y-intercept of a line. In fact, we can replace it with something else that is more valuable to us. Forgetting about equations, think about lines in an intuitive sense. Exactly what 2 parameters are needed to make a unique line? Obviously the slope must be one, but realize that we can actually use any point on this line to uniquely determine its equation (given that we know its slope of course). The y-intercept is just one example of a point on the line, which is why this equation works. Why is knowing a single point useful? Well any line going through the origin has the form $y = mx$, so if we know a point on the line, we can simply shift the entire line to the origin by subtracting the x-value of our known point from the x-value of our function, and the same for the y value. For example, if we know a point on the line is given by (x_0, y_0), then using $x - x_0$ instead of just x, and $y - y_0$ instead of y, we now get a line through the origin. So given a point (x_0, y_0) and a slope m of a line, the equation $y = mx$ now becomes:

$$(y - y_0) = m(x - x_0) \tag{2}$$

This equation actually has a name; it's called the point-slope formula of a line. Obviously it looks different than the slope-intercept form we all know and love, but they are essentially synonymous. For example, let's say we have some line that has a slope of 5 and goes through the point (-2,-8). Then to find an equation for this line, we simply shift it to the origin by subtracting -2 from any x-value given and subtracting -8 from the corresponding output y. So our equation $(y - y_0) = m(x - x_0)$ becomes $(y - (-8)) = 5(x - (-2)) \iff (y + 8) = 5(x + 2) \iff y = (5x + 10) - 8 \iff y = 5x + 2$. Now we see that this is just a line with slope of 5 (note the slope is preserved as wanted) that crosses the y-intercept at 2. So we see that we can go back and forth between these two equations, so why exactly would we rather have our line in point-slope form? For the sake of understanding math in a general sense, this formula should be much more intuitive and useful than the more commonly known slope-intercept form. For the sake of this paper, this equation will come in handy later. The main takeaway from this section is that exactly 2 parameters are required to graph a line, its slope and a point on the line.

Derivative Calculus

Here we'll provide a much more brief review of calculus. There are actually only two concepts that we require from calculus for the sake of this paper: the derivative and the power rule. What exactly is the derivative? In the simplest sense, the derivative is something that allows us to talk about the slope of a function.

For example, if we have some line, the derivative isn't of much use to us because we already know what the line's slope is. But more importantly, the slope of a line never changes. A line's slope is its slope; no matter what x-value you give it, it will always have the same rise over run ratio. The derivative really only becomes useful when we talk about curves instead of lines, because the slope of a curve changes as the x-value changes. Let's remove the abstraction of this thing called the derivative and "define" it now as a function that outputs very important information: Given any function $f(x)$, we can take its derivative, denoted as $f'(x)$ (or equivalently $\frac{df}{dx}$), **which is a function whose output (given an input x) is equal to the slope of $f(x)$ at x.** You can think about $f(x)$ and $f'(x)$ almost like having a parent-child relationship. $f'(x)$ gives us a for-sure formula for the slope of its parent function at any point in its domain. For example, let $f(x) = x^2 + 5x$. Then $f'(x) = 2x + 5$. If we input 10 to f's derivative, we get $f'(10) = 2(10) + 5 = 25$ which is the slope of f at the point $x = 10$. In this example, we used the power rule as the method of differentiating. For the sake of this paper, we assume that you have not only seen the power rule, but also know how to use it. We'll spare the details as to other rules for differentiating functions and as to why the derivative does what it does (this would result in a much longer paper). The main takeaway here is that a function's derivative tells us important information (specifically, information related to slope) about the function itself.

Best Linear Approximation

We now combine the ideas used in the previous two sections to explain the concept of best linear approximation. Maybe you've heard the term "best linear approximation", but what exactly is it? Let's say we have a function $f(x)$, and we want to approximate the behavior of this function at some point with another function. Furthermore, we also want this approximation function to be linear, which simply means that it's a line. Before going straight into the math, what sort of characteristics would we want this approximation to have? Surely, if it was a good approximation, then it would have the same slope as $f(x)$ at the point we're interested in. But also, we would want this approximation to have the same output value as $f(x)$ at our point of interest. These two properties should seem intuitive if we want this approximation to actually be a decent approximation, so we can naively start creating some function with these properties. In math terms, given a function $f(x)$, we want to create a function $h(x)$ to approximate f at a point a (it should be obvious that we require a to be in the domain of f) such that:

(i) the slope of $h(x)$ = the slope of $f(x)$ at point a

(ii) $h(a) = f(a)$

So how does knowing these properties help us create $h(x)$? Well we know that $h(x)$ is linear, and recall from earlier that all we need to represent a line is a point on the line and the slope of the line. And that is exactly the information that we have here! We know the point on our line is given by $(a, f(a))$, and the slope of our line is given by the slope of $f(x)$ at a. In order to represent this slope in strictly math terms, we also know that $f'(a)$ is the more formal way of saying "the slope of $f(x)$ at a" from the section on derivative calculus. Thus we create $h(x)$ using the formula we derived in the first section where we replace y with $h(x)$, and also the point (x_0, y_0) with $(a, f(a))$: $(y-y_0) = m(x-x_0) \iff (h(x)-f(a)) = f'(a)(x-a) \iff h(x) = f(a)+f'(a)(x-a)$ Thus we now have a 'naive' approximation to $f(x)$ at a which is given by $h(x) = f(a)+f'(a)(x-a)$. But as it turns out, this is the best approximation that we could possibly get. Why? When creating an equation of a line, there are only 2 parameters which we can control, the slope and the point on the line. Since we've constructed this equation using $f(a)$ as the point of interest and the $f'(a)$ as the slope, there is no possible way we can make this more accurate. In other words, we have already fine tuned all possible things to fine tune when it comes to our line.

Here, we specifically defined the function $h(x)$ as the line that approximates $f(x)$ at a point a, but we can instead say that f is roughly equal to h at a. More formally:

$$f(x) \approx f(a) + f'(a)(x-a) \tag{3}$$

If you're still not convinced that this is indeed the best linear approximation of a function, let's take a look at a quick example. Let's try to find the best linear approximation of a line and see what we get. Let $f(x) = 5x + 7$, and let's just pick a random point on the line, say when x = 2. Then using our formula: $f(x) \approx f(2) + f'(2)(x-2) = (5(2)+7) + (5)(x-2) = 10 + 7 + 5x - 10 = 5x + 7 = f(x)$. In other words, we just said that our linear function is approximately equal to itself near $x = 2$! We actually could have used any point on this line, not just 2, and we would've reached the same conclusion (I leave this as an exercise for the reader :)). Thus we now have a formula for the best linear approximation. The main takeaway from this section is that the linear approximation of some function f at a point a is simply determined by f evaluated at a and f's derivative evaluated at a (because a line is determined by only two parameters).

BLA and Polynomials

In the previous section, we showed how to derive the best linear approximation (which we'll now denote as BLA) of a function near some point, and hopefully convinced you that it is indeed the best linear approximation we can find. Now that we understand where it comes from and why it works, let's look at it from a different perspective. Specifically, as a polynomial.

For those who need a quick reminder, a polynomial is an equation of the form $f(x) = c_0 + c_1x + c_2x^2 + \cdots + c_nx^n$ (this isn't necessarily a rigorous definition of a polynomial but it's pretty close). Furthermore, we can shift this polynomial by an amount of a units on the x-axis by subtracting a from each x-value like so: $f(x) = c_0 + c_1(x-a) + c_2(x-a)^2 + \cdots + c_n(x-a)^n$. This should seem straightforward since it's the exact same concept we looked at in the first section of this paper on graphing lines. The a value is simply moving the "center" of the graph to a instead of the center being at the origin. Because of this, a here is called the center of the polynomial.

So why are we bringing up polynomials? When we first derived the BLA equation, it seemed very intuitive and straightforward; so why the need for more complication? The goal of deriving the BLA from the point-slope equation was to gain an understanding as to where it comes from. But moving onward, there are many reasons why viewing the BLA as a polynomial is in our interest. In particular, taking derivatives of polynomials is extremely fast and simple. As stated earlier, anyone who was taken a calculus I class should remember the power rule and the ease of its implementation. But this just begs another question: why is the ease of taking derivatives important, and what do they have to do with approximations? As we'll see shortly, the information that derivatives encompass is key to understanding approximations.

Let's go back and look at the approximation function we derived in the previous section:
$f(x) \approx f(a) + f'(a)(x-a)$ near a point a in f's domain. Let's now compare that to a polynomial of first degree centered at a (which is just a fancy way of saying a line): $f(x) = c_0 + c_1(x-a)$. Let's denote the approximation function as $A(x)$ to avoid having confusion with two $f(x)$ functions. So we have:

Equation of line in polynomial form: $f(x) = \ c_0 \ + \ \ c_1(x-a)$
BLA for the polynomial above: $A(x) = f(a) + f'(a)(x-a)$
See how similar these equations are when we look at our line equation in polynomial form? They seem to match up perfectly with $f(a)$ matching to the constant term c_0 and $f'(a)$ matching to the x term c_1. In fact, this matching of terms is no coincidence. If we plug in a to $f(x)$, we get: $f(a) = c_0 + c_1(a-a) = c_0$ as expected. But why does $f'(a)$ match with c_1? Note that if we take the derivative of f, all the constant terms disappear and we're just left with c_1! Ah-ha, so we're getting a hint as to why derivatives play a role when we look at these functions from a polynomial viewpoint.

This still doesn't answer exactly why looking at approximations as polynomials is helpful. Sure, they match up nicely with BLA from a visual standpoint, but so what? So far we've only talked about the best **linear** approximation, but what if we want to do better? Realize that the BLA is a really good approximation if we want to approximate a function **at just a single point.** Surely we can find the BLA of a function at (almost) every point in the function's domain in an iterative way. Then we could compare every single one of these BLA's to each other to see how the function's slope changes at different x-values. But approximating an entire function this way creates more work than wanted since a computation must be performed for every single point in the domain. What if we instead wanted to create a function that approximates not just a single point, but a range of points? How would we go about this?

Firstly, it should be obvious that we can't expect to accurately approximate curves with just a line. Instead, we need to approximate curves with other curves. There are many different families of functions that generate curves when graphed, so which do we choose? We know $sin(x), e^x, \frac{1}{x}$ all generate curves, but there is one family of curves that we're most interested in. Polynomials! Polynomials are great for this purpose for multiple reasons. For one, we can get more and more complex curves simply by adding one to the degree of the polynomial. From an intuitive sense, "fitting" a parabola to a curve should give a more accurate approximation than a line, since we can literally match the curve of the parabola to the curve of the function we are approximating (for a small range of x-values). Furthermore, a cubic polynomial gives yet another term we can manipulate (and also another curve) so that we can get an even better approximation. This can be done iteratively to find the best n-th degree polynomial, where each iteration theoretically becomes more and more accurate than the previous one. Besides this reason, there is another important one that makes us choose polynomials as our functions of choice. As aforementioned, we can easily take multiple derivatives of polynomials which encode information about the function. In the case of the BLA, we saw a slight hint of how derivatives are useful in this manner.

An Allude to Taylor Series

The entire previous section was just so that we can set the scene of using polynomial functions as approximation functions. So we know that our goal is to create a polynomial that accurately approximates some curve (which may be or not be a polynomial) within some range. We also want our polynomial approximation to be more accurate as we make it more complex by adding more terms to it (i.e. making it a higher degree). This is still a pretty challenging task from an intuitive perspective, but there seems to be a connection between polynomials and derivatives (if I haven't stated this enough yet). Let's explore this concept now.

When we were dealing with BLA, we saw that if we found the BLA of a line, it's actually just the equation of the line itself (which should match our idea of wanting our approximation to be the best that it can be). Let's see if this idea extends to polynomials of higher degree. In other words, we'll try "approximating" an n-th degree polynomial with another n-th degree polynomial (that don't necessarily look the same) with the hope that our approximation actually turns out to be equal to the function we are approximating. The reason we want to try this is because, in the simplest case, the curve we want to approximate is just another polynomial. Before we try approximating curves that contain $sin(x)$, $cos(x)$, e^x, etc. it will probably be useful to start with the simple case and see if we can derive a general formula from there. We'll consider a cubic function for the example below (i.e. polynomial of degree 3).

So let's look at what these two equations are going to look like, just like we did above for a line in polynomial form and its BLA. The equations below each represent a cubic curve centered at a (note that BCA stands for best cubic approximation):
Cubic in polynomial form: $f(x) = c_0 + c_1(x-a) + c_2(x-a)^2 + c_3(x-a)^3$
BCA to cubic above: $A(x) = \_ + \_(x-a) + \_(x-a)^2 + \_(x-a)^3$
Just like in the case of the line, we are going to want the terms to match up here in order to get the same equation. So we want the first blank space to equal c_0, the second blank space to equal c_1, etc. The first naive thought might be something like cleverly rearrange $f(x)$ to isolate the c_0, c_1, c_2, c_3 terms and then plug in each into $A(x)$. But there is actually a much easier way to solve this problem; through derivatives! If we take the first derivative of each equation, we see that the constant terms go away, the x terms become the constant terms, the x^2 terms become the new x terms, and the x^3 terms become the new x^2 terms. In other words, every term gets "demoted" a power (and the exponent comes down as a coefficient via the power rule), and the constants disappear. Since the equations are cubic, we can also take a second and third derivative, repeating this process. For a visual representation, let's denote the blank spaces in $A(x)$ with a d variable:
Iteration 0:
$f(x) = c_0 + c_1(x-a) + c_2(x-a)^2 + c_3(x-a)^3$
$A(x) = d_0 + d_1(x-a) + d_2(x-a)^2 + d_3(x-a)^3$

Iteration 1:
$f'(x) = c_1 + 2c_2(x-a) + 3c_3(x-a)^2$
$A'(x) = d_1 + 2d_2(x-a) + 3d_3(x-a)^2$

Iteration 2:
$f''(x) = 2c_2 + 6c_3(x-a)$
$A''(x) = 2d_2 + 6d_3(x-a)$

Iteration 3:
$f'''(x) = 6c_3$
$A'''(x) = 6d_3$

Okay cool, so what? Note the following pattern: at each iteration we isolate the next constant term. What we mean by isolate is that we get rid of any x term attached to it. For example, at iteration 0, the c_0 term is already isolated. At the first iteration, the c_1 term is isolated. At the 2nd and 3rd iterations, the c_2 and c_3 terms are isolated except for constants in front (which are fine as long as there are no x terms). Now, we can cancel out all of the other terms (leaving just the c terms we're interested in) by plugging in a to the functions at each iteration:
Iteration 0:
$f(a) = c_0 + c_1(a-a) + c_2(a-a)^2 + c_3(a-a)^3 = c_0$
$A(a) = d_0 + d_1(a-a) + d_2(a-a)^2 + d_3(a-a)^3 = d_0$

Iteration 1:
$f'(a) = c_1 + 2c_2(a-a) + 3c_3(a-a)^2 = c_1$
$A'(a) = d_1 + 2d_2(a-a) + 3d_3(a-a)^2 = d_1$

Iteration 2:
$f''(a) = 2c_2 + 6c_3(a-a) = 2c_2$
$A''(a) = 2d_2 + 6d_3(a-a) = 2d_2$

Iteration 3:
$f'''(a) = 6c_3$
$A'''(a) = 6d_3$
And now we have exactly what we wanted! By using derivatives we very easily isolated each of the constant terms (c_0, c_1, c_2, c_3). Namely:
$c_0 = f(a)$
$c_1 = f'(a)$
$c_2 = \frac{1}{2}f''(a)$
$c_3 = \frac{1}{6}f'''(a)$
Now all we need to do is plug these values into the original $A(x)$ to get the functions to match:

$A(x) = \underline{\hspace{1cm}} + \underline{\hspace{1cm}}(x-a) + \underline{\hspace{1.3cm}}(x-a)^2 + \underline{\hspace{1.3cm}}(x-a)^3$ now becomes...
$A(x) = f(a) + f'(a)(x-a) + \frac{1}{2}f''(a)(x-a)^2 + \frac{1}{6}f'''(a)(x-a)^3$.
And thus we've arrived at the best cubic approximation! In practice, we should pick a random cubic equation and compute $A(x)$ to see if it actually is the same as the function itself, but you can trust us here that this will work (in order to save space on an already very long paper).

So we get the idea that this notion of "approximating" polynomials of higher degree works the way we wanted it to. Now how do we generalize the process we just performed for a polynomial of degree n? If it wasn't for the $\frac{1}{2}$ and $\frac{1}{6}$ constants, generalizing this process would be very straightforward. We would just say that the i-th coefficient of $A(x)$ is simply given by the i-th derivative of $f(x)$. But these fractions suggest we are missing an extra term. These fractions are simply a consequence of taking derivatives of a polynomial iteratively. For a quick example, if we let $g(x) = x^7$, then $g'(x) = 7x^6$, $g''(x) = 7 \cdot 6x^5$, $g'''(x) = 7 \cdot 6 \cdot 5x^4$, and so on. For those familiar, this looks very similar to the factorial operation (denoted by n!). For those not familiar, $n!$ is simply equal to $(n-1)(n-2)\cdots(n-(n-1))$ (e.g. $3! = 3{\cdot}2{\cdot}1 = 6$).

In the process above, in order to isolate the c_i term, note that we had to perform i amount of derivatives to $f(x)$. Since the power rule always brings down the exponent as a coefficient, it should seem natural that factorials show up during this procedure. So our missing term in our function $A(x)$ is $\frac{1}{i!}$ where i denotes the i-th coefficient. If we wanted to be pedantic, the more accurate way to write our function $A(x)$ would have been:
$A(x) = \frac{1}{0!}f(a)(x-a)^0 + \frac{1}{1!}f'(a)(x-a)^1 + \frac{1}{2!}f''(a)(x-a)^2 + \frac{1}{3!}f'''(a)(x-a)^3$
This looks like a very messy and complicated equation, but realize that is came directly from trying to match each of the d coefficients in $A(x)$ to the c coefficients in $f(x)$.

It should now be obvious how this extends to the general case of a polynomial of degree n. In a more formal sense: The best "approximation" of an n-th degree polynomial f centered at a is given by the following polynomial:

$$A(x) = f(a) + f'(a)(x-a) + \frac{1}{2}f''(a)(x-a)^2 + \cdots + \frac{1}{n!}f^n(a)(x-a)^n = \sum_{i=0}^{n} \frac{1}{i!}f^i(a)(x-a)^i \quad (4)$$

Note that in the case of approximating polynomials, the choice of a doesn't matter (as long as a is in f's domain) since our approximation function always turns out to be the exact same function we're trying to approximate.

So it seems that approximating polynomials with other polynomials is actually pretty easy! All you need to compute is n amount of derivatives which are relatively simple since we can use the power rule on polynomials. But how does this extend to approximating more complex curves? We've brought up e^x a couple of times now, so how would this process work for a non-polynomial function like e^x?

The Grand Finale (A.K.A. Taylor Series)

We now know that approximating polynomials with this method we just performed very easily generalizes to any n-th degree polynomial. But now we make an important distinction. The math that we used to derived formula 4 above came directly from applying derivative techniques to the polynomial we wanted to approximate, but the general idea of the process we were applying didn't even rely on the fact that we were approximating a polynomial.

Let me explain. We were specifically tasked with breaking down each of the functions (using derivatives) to gain information on each individual coefficient. But now that we know what the coefficients are, if we take a step back and look back at the same iterations from a more global perspective, what exactly were we doing at each iteration? Looking at the functions as a whole now (instead of just the coefficients), we were trying to match $A(x)$ to $f(x)$ at each step. If this doesn't make sense, go back and look at the iterations. There is a reason that the two functions are formatted in a specific way; namely so that you can visually see the "matching" we are trying to achieve. In other words, for the approximation to work, the two functions at each "derivative-iteration" must match up. In a more formal mathematical sense, we spent two and a half pages just so that we could achieve the following equalities:
$A(a) = f(a)$
$A'(a) = f'(a)$
$A''(a) = f''(a)$
$\vdots$
$A^n(a) = f^n(a)$

On the first go around of doing these iterations, this might not have been so obvious since it was happening sort of under the hood. But this was actually the ultimate goal of what we were trying to achieve. Using derivatives was just one step of the overall process of matching $A^n(a) = f^n(a)$ since we had to know what the coefficients would be. Furthermore, we saw that by assuming $f(x)$ could be written as a polynomial was a very straightforward way to deduce this formula. This is because, in terms of the process we were applying, the only assumption was that the derivatives of $f(x)$ were defined. So we really just used approximating polynomials as a jumping off point that lead to what needs to be done in the general case. Because of this, **we can now approximate a function with an n-th degree polynomial, as long as this function has at least n derivatives, with the same formula as above:**

$$T_n(x) = \sum_{i=0}^{n} \frac{1}{i!} f^i(a)(x-a)^i \tag{5}$$

This function $T_n(x)$ defined here is called the Taylor Series approximation of degree n of $f(x)$ centered at a (a mouthful, yes). Maybe looking at this formula makes the ideas we just presented seem complicated and intricate, but they really are pretty simple. We showed from an algebraic standpoint why matching these derivatives up works, but you can even think about it intuitively now. As stated previously, a function's derivatives contain information about the function itself. So wouldn't it make sense that if we wanted to approximate a function as accurately as possible, we would want the derivatives of the function and our approximation function to match up? And that is really all that the formula above is saying. It looks complicated since it's a nasty sum and factorials are thrown in there and what not, but these are all just side effects from taking multiple derivatives. The core idea remains the same though, approximating functions with polynomials is extremely powerful because of how malleable polynomials are.

We've just about reached the peak of this very large mountain, since we have almost all the tools to finally tackle e^x. But there is one thing we seem to be overlooking. What range of inputs does this approximation work for? When we did the example of polynomials, we noted that the choice of a didn't matter because the approximation ends up being the same function as the one we were approximating. Why is this so? A polynomial of n-th degree only has n derivatives, so our degree n Taylor Series approximation was able to contain exactly all of the information that $f(x)$ did. But what about functions that have an infinite amount of derivatives?

For example, $sin(x)$ has cyclic derivatives between $\pm sin(x), \pm cos(x)$, meaning we can take an infinite amount of derivatives. How would we know what value of n to stop at for our Taylor Series, and more importantly, what range of values would this accurately approximate depending on the choice of n? Well actually, there is an obvious answer to the first part of that question: why stop at a value of n at all? If we can take an infinite amount of derivatives, let's just take an infinite Taylor Series! Theoretically, nothing is stopping us from performing an infinite amount of iterations as we did above, as long as we know there are an infinite amount of derivatives of the function. And so alas, we reach one step closer and can now define the term "Taylor Series" in its fullest: **The general Taylor Series approximation to a function $f(x)$ centered at a, is denoted by $T(x)$ and is given by:**

$$T(x) = \sum_{i=0}^{\infty} \frac{1}{i!} f^i(a)(x-a)^i \tag{6}$$

Yet we still haven't answered the question of what range of values does this work? Does it depend on the function? Does it depend on the choice of a? Does the range grow linearly as more terms are added until the functions are equal? This whole paper was about showing why e^x is **equal to** its Taylor Series, not why e^x's Taylor Series is a really good approximation. Let's consider e^x and the case when $a = 0$. That is, we want to approximate e^x around the origin, and see how well its Taylor Series expansion does its job (i.e. what range of x-values it accurately approximates). Plugging these things in to the Taylor Series formula, we get:

$$f(x) = e^x \approx \sum_{i=0}^{\infty} \left(f^i(0) \cdot \frac{1}{i!} \cdot (x-0)^i \right) = \sum_{i=0}^{\infty} \left(e^0 \cdot \frac{1}{i!} \cdot (x)^i \right) = \sum_{i=0}^{\infty} \left(\frac{x^i}{i!} \right) \tag{7}$$

We've now reached part of what makes this equation so beautiful; e^x is its own derivative! And since we're approximating at the origin (i.e. $a = 0$), we always get the derivative part of the sum to cancel because e^0 always $= 1$. And now we're so close to proving what the original claim was, but we still have an approximation sign instead of an equal sign. We know that $e^x \approx \sum_{i=0}^{\infty} \frac{x^i}{i!}$ near the origin, but how well does it approximate?

As it turns out, this is actually a very hard problem to reason about. We've deduced from earlier that this approximation is about as good as we can get, but there's no good reason to suspect why these two equations are actually equal to each other. Let's start with some basics. Surely we don't expect each n-degree Taylor Series approximation to be an absolutely perfect approximation, so we will have an error term at each iteration. Ideally, this error term will get smaller and smaller as we add more terms to our n degree approximation. Furthermore, since the Taylor Series in its fullest has an infinite amount of terms, can we get this error term to get negligibly small? For those familiar, the idea that we're getting at is the concept of the limit of this error term to equal 0 as n approaches infinity. We can represent this error term we speak about in terms of a function that changes at each n-th iteration. Previously, we found that $e^x \approx T(x)$, but now we get $e^x = T_n(x) + R_n(x)$ where $R_n(x)$ represents the amount of error between e^x and its Taylor Series at any value of n. If we can show that $R_n(x)$ approaches 0 as n tends to infinity, then we'll have proved exactly what we wanted: $e^x = T(x)$.

Before moving on, I need to give a brief aside. My goal of this paper is to give the reader an intuitive understanding for all of the concepts presented here, building up to a climax where all the ideas are put together to form a beautiful equation. An important part of this aspect (in my opinion) is by using more words and explanations than mathematical rigor, since it becomes very easy to lose sight of what is actually going on when thrown lots of equations and numbers. There is a very common (and relatively straightforward) way to solve the problem we have at hand here, but it uses a formula that is hard to grasp intuition for without using lots of calculus. Because of this, we will leave out the rigor as to how this equation is derived and assume it works for free (sad, I know) to keep the paper from gaining another 5+ pages.

This equation is called the Taylor Inequality Theorem and is key to working with Taylor Series. The problem we have at hand is that we need to show that this error function $R_n(x) = 0$ as n approaches infinity. The Taylor's Inequality gives us the following: if the $n+1$ derivative of e^x can be bounded above by a number, let's say M, then we know that we can also bound the error function $R_n(x)$ by $\frac{M \cdot |x|^{n+1}}{(n+1)!}$. Note that this isn't Taylor's Inequality in its fullest, we've tailored it to fit our needs with approximating e^x around the origin. In a more rigorous sense we have the following:

$$|e^{n+1}(x)| \leq M \implies |R_n(x)| \leq \frac{M \cdot x^{n+1}}{(n+1)!} \tag{8}$$

There are multiple proofs of this theorem in the general case, but they require a lot more calculus than I want to put in this paper, so unfortunately we'll cheat by assuming it for free (and trust the very clever people who discovered it).

So from this theorem, if we can show that the $\frac{M\cdot|x|^{n+1}}{(n+1)!}$ term approaches 0 as $n \to \infty$, then we get exactly what we want. If we just look at the fraction $\frac{|x|^{n+1}}{(n+1)!}$, we can see that this definitely goes to 0 as n gets sufficiently large for any value of x. Why? It helps to pick a choice of x and write out a few terms to see what is going on. Let's let $x = 3$ and consider the following iterations for $n = 0, 1, 2, 3$: $\frac{3}{1}$, $\frac{3\cdot 3}{2\cdot 1}$, $\frac{3\cdot 3\cdot 3}{3\cdot 2\cdot 1}$, $\frac{3\cdot 3\cdot 3\cdot 3}{4\cdot 3\cdot 2\cdot 1}$. Now we can imagine as soon as n gets past the point of x (which is 3 in this example), that $n!$ will start growing much much faster since it's adding terms that are bigger than 3 at each next step. So for very very large values of n, the denominator of this fraction will be much larger than the numerator, which in turn makes the value of the fraction get smaller and smaller. Although not a rigorous proof, we can expect for this fraction to go towards 0 since $n!$ will grow at a faster and faster rate than x^n.

Great, so now what about this M term? In the general case, the choice of M is left to us to choose, as long as it satisfies the inequalities above. So normally, we would probably have to choose different values of M depending on which derivative we are taking of the function we are approximating in order to satisfy $|f^{n+1}(x)| \leq M$. Well we have a particularly special case of the Taylor's Inequality since the function in question is e^x. Recall that e^x is its own derivative, so we can actually choose M once and know that if it works in one case, it will work in all cases! In fact, since e^x is always positive and always increasing, we know that its max value will always be at the x value it's given for any derivative. For example, $e(5) \leq e^{n+1}(5)$ no matter how large we choose n. This is important, because now we don't have to worry about the M term in the second inequality. This is because we can choose it to be any constant and the $\frac{M\cdot|x|^{n+1}}{(n+1)!}$ term will still always approach infinity. So for any x, we can choose $M = e^x$ to give us what we want.

This is an important statement, because we're saying that for any x we choose in e^x's domain (which is just all real numbers) I can satisfy the inequality by choosing $M = e^x$. And thus we're done! We've shown that the error term is 0 as n approaches infinity (by using the Taylor's Inequality Theorem), i.e. $e^x = T(x) + 0$ proving our original claim once and for all.

Before ending, we want to make the important distinction here between rigor and intuition. As we neared the end of this paper, a lot of the rigor was excluded to try and keep the intuition as high as possible. Because of this, this paper should not (and can not) be used as a rigorous proof for why the exponential function is equal to its Taylor Series as I omitted multiple parts of the actual proof. Instead, you should walk away with a much better understanding of not only where Taylor Series come from and why they work, but all of the concepts that lead up to the punchline.

Postnote

One of the major downsides to using Taylor Series approximation vs other approximating techniques is that we require a certain number of derivatives of the function we want to approximate. This works great for functions that can be differentiated infinitely, e.g. e^x, $sin(x)$, $\frac{1}{x}$, etc. Under the hood, there are other requirements of using Taylor Series. For example, for some functions, they only work for a small range of values simply because of how the function behavior relates to its derivatives. Although Taylor Series are amazing in their own right, they might fall short for other function families. For example, Fourier Series are another method of approximating different types of functions, maybe covered in another paper :)