Mech 221 Mathematics Lectures 1-4 Notes: Overview and Numerical Methods

Brian Wetton, www.math.ubc.ca/∼wetton

September 21, 2008

1 Overview of Mathematics Material in Mech 221

Last year, most of you learned the basics of calculus in Math 100 (differential calculus) and Math 101 (integral calculus). You also had an introduction to linear algebra (vectors, matrices, linear systems, eigen-analysis) in Math 152, where you were also introduced to the scientific software package, MATLAB.

This term, we will carry on with two main ideas from last year:

Numerical Approximation: Last year, you saw how integrals could be approximated by left and right Riemann sums, and that more accurate approximations could be found using the Trapezoid or Simpsons Rules. We begin the course with a review of this subject, adding more rigour to the theoretical expressions for the error made. More practical experience with these methods will also be gained, using MATLAB to do the computations. Numerical methods for approximating function values (interpolation) and derivatives will then be considered.

Differential Equations: Last year in Math 101, you saw how exponential growth and decay, draining tanks, and damped spring mass systems could be described by differential equations. You learned some analytic techniques to solve simple (linear, constant coefficient) second order problems, and (in Math 152) first order systems. In this course, we will review these previous topics, starting with first order differential equations, then moving to second order equations and ending with

first order differential equation systems. The main discussion is on linear, constant coefficient equations with simple forms for forcing terms. However, we extend the discussion to more complicated equations when possible. In addition, we will learn how to approximate solutions to differential equations using numerical methods. Some theory of these methods, practical matters, and implementation in MATLAB will all be considered.

Two new important new ideas are introduced in this class: linearization and non-dimensionalization (scaling).

The overall goal for the Mathematics part of Mech 221 is to make students better understand the three part process of mathematical modelling, specifically when these models involve differential equations:

1. The conversion of an engineering problem to a mathematical one (with appropriate simplifications). This process is done also in all the other subjects of Mech 221.

2. The solution of the mathematical problem. Often this step can only be done approximately with numerical methods because of the size or complexity of the problem. It is important to determine that the numerical approximation used is accurate enough. The main subject of Mathematics in Mech 221 is analytic and numerical solution techniques to differential equations, which arise as models in many applications.

3. Relate the mathematical solution to the original Engineering problem. Since the mathematical model may not correspond exactly to the physics, some check of the results against experimental tests should be done here. The model may have to be revised to include more effects or have parameters determined more accurately. This would set the process back to the first step above.

Solving an engineering problem with pencil and paper (and computations) in this way is much cheaper and faster than extensive physical design cycles. In some cases, experimental work is not at all possible due to cost or safety concerns. Mathematical modelling is particularly effective in problems where elements with well defined properties are put together in a complex way.

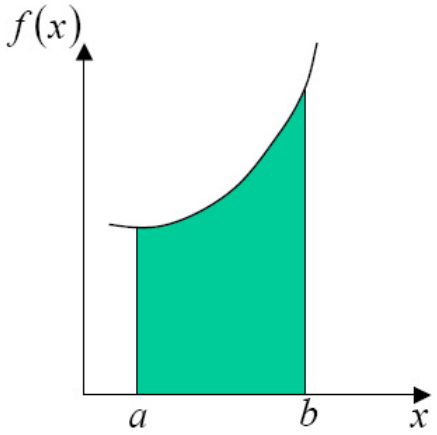

Figure 1: The area is given by $\int_a^b f(x)dx$

2 Numerical Integration

Recall the interpretation of an integral $\int_a^b f(x)dx$ as the area shown in Figure 1. In Math 101, you also saw many applications of integration: average values, volumes, work, arc length, centre of mass, the solution of separable differential equations (such as those coming from the application of Torricelli's law to draining tanks).

From last year, you should know what an indefinite integral (anti-derivative) is and how it relates to finding definite integrals (areas). You should be able to do elementary integrals (integrands that come from simple derivatives) and to use some basic integration rules (chain rule and product rules) and techniques (trigonometric substitution and partial fractions).

Exercise 1 *You should be able to evaluate the following integrals:*

$$\int_0^1 \sin x dx$$
$$\int e^{-x} dx$$
$$\int x e^{x^2} dx$$
$$\int x \cos x dx$$

Consider the indefinite integral

$$\int e^{-x^2} dx. \tag{1}$$

The integrand appears simple. You can find its derivative easily. This integral (actually a scaled version of it) is important in the study of probability (it represents cumulative probabilities of normal distributions) and other fields. However, it is impossible to find an analytic form for this indefinite integral in the "class of functions you know". A scaled version of this integral defines a new well-studied function known as the error function. However, this function is not implemented on your mech2 approved calculator.

Exercise 2 *Define as precisely as you can the class of functions you can find derivatives of. This class corresponds to functions that you can evaluate on your mech2 calculator. Why are these functions on your calculator and not others (like say the error function)?*

2.1 Riemann Sums

Definite integrals corresponding to difficult situations like (1) can occur in applications. In these cases, numerical methods can be used to approximate the definite integrals. Let us consider a very basic idea to approximate integrals, Left Riemann Sums. Consider

$$I = \int_a^b f(x)dx.$$

Divide the interval $[a, b]$ into N equal subintervals having length

$$h = \frac{b-a}{N}.$$

The Left Riemann Sum

$$L_N = hf(a) + hf(a+h) + hf(a+2h) + \cdots + hf(b-h)$$

can be used to approximate I. This is equivalent to approximating the area I by the sum of the areas of N rectangular boxes as shown in Figure 2. As $N \to \infty$ we know that $L_N \to I$ (this was the definition of the integral). For a given integrand f, interval $[a, b]$ and N, Left Riemann sum approximation of integrals are routine computations that can be done easily in MATLAB, as discussed below. However, some important discussion is appropriate at this point:

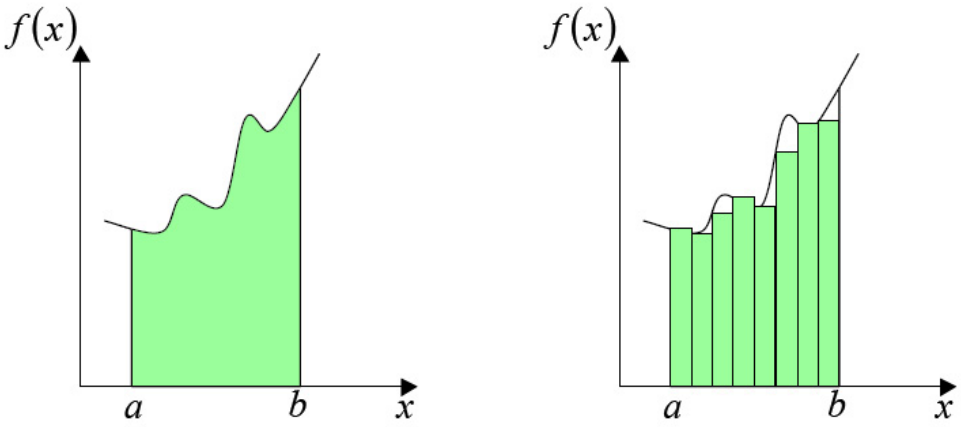

Figure 2: Using Left Riemann Sums to approximate definite integrals.

- The main question when using numerical integration is how large does N have to be in order that the numerical approximation is "accurate enough". Accuracy requirements can vary significantly depending on the application. Consider the extreme cases of a computation as part of a structural design that has a built in safety factor of 50% compared to a computation of the trajectory of a Mars mission.

- All your study last year learning how to evaluate integrals analytically is still useful. Having an analytic answer takes out any concern over the accuracy of an approximation. In addition, parameters can be included in an analytic integration: you can see exactly how the solution will depend on a parameter, rather than having to do many integral approximations and try and piece together the parameter dependence.

- In many cases, the function f in question is only known experimentally at discrete points. In this case, N is not something you can vary in order to get more accurate approximations of the integral. Thus, numerical integration is used both to evaluate integrals approximately for which no analytic forms are known *and* to evaluate approximate integrals where the function is only known at discrete points.

Exercise 3 *Consider* $\int_0^1 f(x)dx$ *approximated by Left Riemann Sums on 2 subintervals (of length* $h = 1/2$*). Sketch the graph of a function* f *for which*

this approximation will be good. Sketch the graph of a different function f for which this approximation will be terrible.

There is a rigorous estimate for the error that will help you determine how many subintervals N you need to get accurate enough answers in approximate integration using Left Riemann Sums:

$$I - L_N = +\frac{1}{2} f'(\xi)(b-a)h \tag{2}$$

for some point ξ in the interval $[a, b]$. It is never necessary to find the point ξ at which this applies, only to know that such a point exists. Note that the point ξ will be different for every different N that you use. The expression (2) makes sense:

- As $N \to \infty$, $h = (b-a)/N \to 0$ and the approximation is more accurate.
- For functions that have large derivatives, $f'(\xi)$ can be large and the error can be large. See Figure 2.
- If $f'(x) > 0$ for all x in $[a, b]$ then $f'(\xi) > 0$ and by (2) L_N will always underestimate the integral I. This makes sense graphically.

In order for (2) to be useful in determining N needed for a given accuracy, a bound on how large $|f'(x)|$ can be on the interval $[a, b]$ is needed. The quantity

$$K_1 = \max_{x \in [a,b]} |f'(x)|$$

could be used, but it is sufficient to find a number M such that

$$|f'(x)| \leq M \quad \text{for all } x \text{ in } [a, b].$$

Of course $K_1 \leq M$. With this information and (2) it is easy to see that

$$|I - L_N| \leq \frac{K_1}{2}(b-a)h = \frac{K_1}{2}\frac{(b-a)^2}{N} \leq \frac{M}{2}\frac{(b-a)^2}{N} \tag{3}$$

These are expressions that can be used to determine how many subintervals (or equivalently what spacing h) is needed to guarantee a given accuracy.

Example 1 *Consider*

$$\int_1^3 f(x)dx.$$

It is known that for all x in the interval [1,3], $|f'(x)| < 5$. How many subintervals N should be used in a Left Riemann Sum approximation of the integral in order to guarantee two decimal place accuracy?

- *We use the first and fourth terms in (3) above*

  $$|error| \leq \frac{M}{2}\frac{(b-a)^2}{N}$$

  where $M = 5$ is the bound on the size of the absolute value of the derivative of the function on the interval given to us by the question and $b - a = 3 - 1 = 2$ is the length of the interval.

- *We now have*

  $$|error| \leq 10/N$$

  To ensure the solution is accurate to 2 decimal places, the error must be smaller than 0.005, which is guaranteed if

  $$10/N \leq 0.005$$

  or, rewriting

  $$N \geq 10/0.005 = 2000.$$

- *To conclude, if we use Left Riemann Sums with $N = 2000$ (or more) subintervals, the approximation is guaranteed to be of the required accuracy.*

2.2 Trapezoidal and Simpsons Rules

You know from Math 101 that you can get more accurate approximations to integrals using other numerical methods than Riemann sums. That is, more accuracy for the same N, or a smaller N to obtain the same accuracy.

One of the better methods is the Trapezoidal Rule:

$$\int_a^b f(x)dx \approx T_N = \frac{h}{2}f(a) + hf(a+h) + hf(a+2h) + \cdots + hf(b-h) + \frac{h}{2}f(b).$$

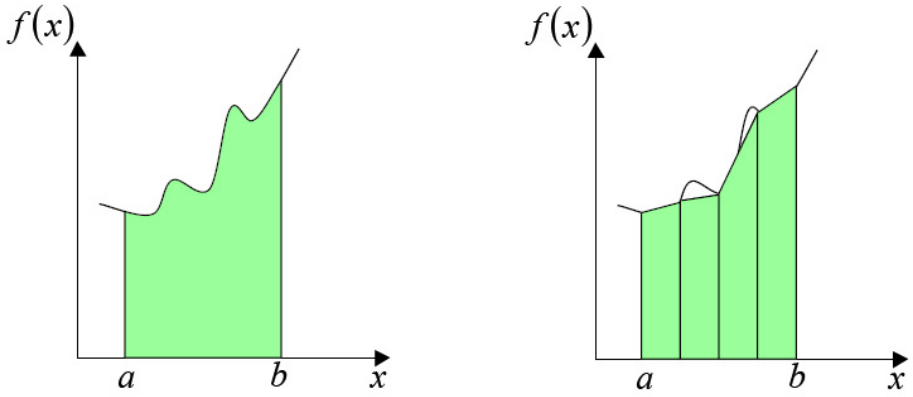

Figure 3: Using the Trapezoidal Rule to approximate definite integrals.

Note that this is only a slight change from the Left Riemann Sum rule but it gives approximations to integrals that are (generally) much more accurate. The graphical interpretation of the trapezoidal rule is shown in Figure 3. The area of the integral is approximated by thin (width h) trapezoidal regions.

Exercise 4 *Suppose we have a cart that runs on a straight track with a speedometer attached to it. The speedometer reports its speed every eighth of a second and sends this data to a computer. In a certain experiment, the cart returns the following sets of data, where t is the time (s) and v is the velocity (m/s).*

```
t = 0 0.1250 0.2500 0.3750 0.5000 0.6250 0.7500 0.8750 1.0000
v = 0 0.0183 0.1250 0.3201 0.5000 0.5335 0.3750 0.1281 0.0000
```

Suppose the experimenter cannot see the cart. Use the Trapezoidal Rule to estimate how far the cart has moved in this one second.

As for Left Riemann Sums, there is a rigorous estimate for the error that will help you determine how many subintervals N you need to get accurate enough answers in approximate integration using the Trapezoidal Rule:

$$I - T_N = -\frac{1}{12} f''(\xi)(b-a)h^2 \tag{4}$$

for some point ξ in $[a, b]$. The expression (4) makes sense:

- As $N \to \infty$, $h = (b-a)/N \to 0$ and the approximation is more accurate. Note that the rate at which the error tends to zero is faster than for Left Riemann Sums ($h^2 \to 0$ faster than $h \to 0$).
- For functions that have large second derivatives, $f''(\xi)$ can be large and the error can be large. See Figure 3.
- If $f''(x) > 0$ for all x in $[a,b]$ (f is concave up on the interval) then $f''(\xi) > 0$ and by (4) T_N will always overestimate the integral I. This makes sense graphically.

An even more accurate method is Simpsons Rule, which can only be applied when N is even. Here,

$$\begin{aligned}\int_a^b f(x)dx \approx S_N &= \frac{h}{3}f(a) + \frac{4h}{3}f(a+h) + \frac{2h}{3}f(a+2h) + \frac{4h}{3}f(a+3h) + \\ &\quad \frac{2h}{3}f(a+4h) + \cdots + \frac{4h}{3}f(b-h) + \frac{h}{3}f(b).\end{aligned}$$

Note the pattern $14242\cdots4241$ in the coefficients.

Exercise 5 *Repeat Exercise 4 using Simpsons Rule.*

The error expression for Simpsons Rule is

$$I - S_N = -\frac{1}{180}f^{(4)}(\xi)(b-a)h^4 \tag{5}$$

where $f^{(4)}$ is the fourth derivative of f. Note that as you use more and more subintervals in the approximation ($N \to \infty$ so $h \to 0$) then Simpsons Rule is expected to be more accurate than the other methods ($h^4 \ll h^2 \ll h$ for h small). We shall see this in the numerical studies in the next section.

Example 2 *Consider approximating $\int_0^1 f(x)dx$. Define*

$$K_j = \max_{x\in[0,1]} f^{(j)}(x).$$

It is known that $K_1 = 1$, $K_2 = 2$ and $K_4 = 1000$. What method would you use to approximate the integral if errors of 0.1 were acceptable? (pick the method that requires the least amount of computational work, measured by the number of function evaluations). Repeat the question if errors smaller than 10^{-8} are required.

- *Here $h = 1/N$ since the interval of integration is of length one.*
- *Using (3) and proceeding as in Example 1 we see that to ensure the desired accuracy*

$$\frac{1}{2N_L} \leq 0.1$$

or $N_L \geq 5$.

- *Starting from (4) we see that what is desired is*

$$|\textit{error in trapezoidal rule}| \leq \frac{1}{12} K_2 (b-a) h^2 \leq 0.1$$

so in this case

$$\frac{2}{12} \frac{1}{N_T^2} \leq 0.1$$

which implies

$$N_T^2 \geq \frac{20}{12} \textit{ so } \; N_T \geq \sqrt{\frac{20}{12}} \geq 1.3.$$

Since N_T has so satisfy the above inequality and be an integer, $N_T \geq 2$ subintervals are required to achieve the target accuracy with trapezoidal rule.

- *Starting from (5) we follow the same pattern as above*

$$|\textit{error in simpsons rule}| \leq \frac{1}{180} K_4 (b-a) h^4 \leq 0.1$$

so in this case

$$\frac{1000}{180} \frac{1}{N_S^4} \leq 0.1$$

which implies

$$N_S^4 \geq \frac{10000}{180} \textit{ so } \; N_S \geq \sqrt[4]{\frac{1000}{18}} \geq 2.8.$$

Since N_S has so satisfy the above inequality and be an even integer, $N_S \geq 4$ subintervals are required to achieve the target accuracy with simpsons rule.

- *Trapezoidal Rule is the most efficient for this modest accuracy.*

- *Following the same algebra as above, for an accuracy of* 10^{-8} *the number of subintervals needed is*

$$\begin{aligned} N_L &\geq 5 \times 10^7 \\ N_T &\geq \sqrt{\frac{2 \times 10^8}{12}} \geq 5774 \\ N_S &\geq \sqrt[4]{\frac{10^8}{180}} \geq 28. \end{aligned}$$

  Simpson's Rule is much more efficient than the others for this case where high accuracy is required.

- *The take home message from this example is that higher order methods are much more efficient at computing accurate approximations. As a rule of thumb, use second order approximations for standard engineering applications, and fourth order methods in situations where higher accuracy is needed.*

2.3 Numerical Integration Example

Let's take as an example

$$I = \int_0^1 \sin x dx = 1 - \cos(1) \approx 0.4597$$

Using numerical methods on this integral is just a theoretical exercise since the exact value is easy to determine. This is convenient for our example since then we can see how the errors to the exact value behave.

Lets work out L_4, T_4 and S_4 for this example by hand (well, with our mech2 calculator). The 4 means $N = 4$, four subintervals. The length of the subintervals is $h = 1/4$ and they are $[0, 1/4]$, $[1/4, 1/2]$, $[1/2, 3/4]$ and $[3/4, 1]$. We can compute

$$L_4 = \frac{1}{4}(\sin 0 + \sin 1/4 + \sin 1/2 + \sin 3/4) \approx 0.3521$$

where the factor of $1/4$ is the common factor of h in the formula for Left Riemann Sums. When evaluating this on your calculator, *do not forget* to switch to the mode in which trigonometric functions take arguments in radians. Similarly we can compute

$$T_4 = \frac{1}{4}(\frac{1}{2}\sin 0 + \sin 1/4 + \sin 1/2 + \sin 3/4 + \frac{1}{2}\sin 1) \approx 0.4573$$

N	$I - L_N$	Bound
2	0.2200	0.2500
4	0.1076	0.1250
8	0.0532	0.0625
16	0.0264	0.0313
32	0.0132	0.0156

Table 1: Errors and theoretical error bounds for Left Riemann Sums applied to the integral $\int_0^1 \sin x dx$.

N	$I - L_N$	$I - T_N$	$I - S_N$
2	0.2200	0.0096	1.6×10^{-4}
4	0.1076	0.0024	1.0×10^{-5}
8	0.0532	0.00060	6.2×10^{-7}
16	0.0264	0.00015	
32	0.0132	0.00004	

Table 2: Approximate integration methods applied to the integral $\int_0^1 \sin x dx$.

and

$$S_4 = \frac{1}{12}(\sin 0 + 4\sin 1/4 + 2\sin 1/2 + 4\sin 3/4 + \sin 1) \approx 0.4597$$

where the factor of $1/12$ is the common factor $h/3$ in the formula for the rule. Note that the Simpsons Rule approximation is the most accurate (all four decimal places shown are correct), Trapezoidal Rule is the next best and the Left Riemann Sums are the least accurate.

Using MATLAB as discussed below, we fill in Tables 1 and 2 for this example. The lines for $N = 4$ can be filled out with the calculations we did by hand above. In Table 1 we compare the errors made with Left Riemann Sums to the error bounds

$$|I - L_N| \leq \frac{K_1}{2N}.$$

In this case

$$\frac{d}{dx}\sin x = \cos x \text{ and } |\cos x| \leq 1 \text{ on } [0,1]$$

which gives us $K_1 = 1$. In Table 1 notice that the errors made are always smaller than the theoretical bound on the error (this is guaranteed by the

theory). If you look at Table 2 carefully, you see that there is a real pattern to the errors. For Left Riemann Sums, every time N is doubled, the error goes down (approximately) by a factor of 2. For Trapezoidal Rule and Simpsons Rule the error goes down by factors of 4 and 16 respectively when N is doubled. These suggest that the errors look approximately like

$$\begin{aligned} I - L_N &\approx C_1 h \\ I - T_N &\approx C_2 h^2 \\ I - S_N &\approx C_3 h^4 \end{aligned}$$

for constants C_1, C_2 and C_3. Written in another way,

$$\begin{aligned} \lim_{h\to 0} \frac{I - L_N}{h} &= C_1 \\ \lim_{h\to 0} \frac{I - T_N}{h^2} &= C_2 \\ \lim_{h\to 0} \frac{I - S_N}{h^4} &= C_3. \end{aligned}$$

Note that this does not contradict (2, 4, 5). In fact it is a stronger statement. Using the Left Riemann Sums as an example, it implies that the value of

$$f'(\xi)$$

in the error expression (2) becomes independent of N for N large. This is generally true, you can see why this is true for Left Riemann Sums by working through Exercise 10. If a numerical method with a parameter h satisfies

$$\lim_{h\to 0} \frac{\text{error}}{h^p} = C$$

where C is a nonzero constant and p is a positive constant, the method is said to be *convergent* (as $h \to 0$ the error goes to zero) with *order* p. Left Riemann Sum is a first order method $p = 1$, Trapezoidal Rule is a second order method $p = 2$ and Simpsons Rule is a fourth order method $p = 4$. From the examples and exercises above, it is clear that higher order methods (large p) save computational work (can give accurate answers even when h is not that small), especially when highly accurate approximations are needed. In a later section, we will see how to construct higher order methods from lower order ones using Richardson extrapolation.

2.4 Using MATLAB to do numerical approximation of integrals

In this section, we will see how to use MATLAB commands to do the numerical integration calculations when f is too complicated or N is too large to do the computations by hand. However, we will just use the simple integral of $\sin x$ from 0 to 1 as in the last section as an example. To get a vector x of values at the ends of 8 equally spaced subintervals between 0 and 1 and a vector y of corresponding $\sin x$ values type

```
x = linspace(0,1,9);
y = sin(x);
```

Note that for 8 subintervals there are 9 end points. Remember that to get more information on MATLAB commands, for example the `linspace` command, you would type

```
help linspace
```

The spacing is

```
h = 1/8;
```

The Left Riemann Sum approximation of

$$\int_0^1 \sin x dx \tag{6}$$

is then given by the MATLAB command

```
h*sum(y(1:8))
```

Note that for the Left Riemann Sum, the last point (number 9) in y is not included in the sum.

There is no need to write separate code for the Trapezoidal Rule, it is already implemented as a MATLAB command

```
trapz(x,y)
```

Note that this command allows the grid points x to be unequally spaced.

For Simpsons Rule, we need to construct the vector of coefficients 142424241. This is done below for general even N:

```
N = 8;
simpc = zeros(1,N+1)+2;
simpc(1) = 1;
simpc(N+1) = 1;
simpc(2:2:N) = 4;
```

The Simpsons Rule approximation of (6) is then given by

```
h/3*sum(simp.*y)
```

Remember that the `.*` is used for pointwise multiplication of vectors of the same length.

Consider now

$$\int_0^1 \arctan x dx. \tag{7}$$

This integral can be done analytically (see Exercise 6 below). However let's pretend we can't, and also that it is too difficult to find bounds on the higher derivatives of the integrand to use the theoretical expressions for the error. We can easily compute trapezoidal rule approximations of the integral

```
N = 2;
x = linspace(0,1,N+1);
y = atan(x);
trapz(x,y)
```

This can be put into a .m file so that the same code can be used for different N (or use the up arrow key to repeat lines you have typed in). Running this gives the approximation 0.4282. With just this one number we can't be sure at all how accurate it is. However, repeat for $N = 4, 8, 16, 32$ to get approximations 0.4362, 0.4382, 0.4387, 0.4388. We know that T_N converges to the exact value of the integral as $N \to \infty$. The T_N values we are computing are getting closer and closer to something that is approximately 0.4388. As a rule of thumb: *when using a second order or higher method, digits that do not change as N is doubled (h is halved) are correct.* It is true that the exact value of (7) is approximately 0.439, obeying this rule.

Numerical approximation of integrals to a specific accuracy are the topic of your first computer lab.

Exercise 6 *Evaluate (7) exactly.*

2.5 A Final Exercise

Exercise 7 *Consider doing numerical integration on*

$$\int_0^1 f(x)dx$$

where $f(x)$ has the following forms:

$$\begin{aligned} f(x) &= e^{x^2} \textit{(exact integral not known)} \\ f(x) &= \begin{cases} \sin(2x^2) \textit{ for } x \leq 1/\sqrt{2} \\ \sin\left(\frac{1}{2x^2}\right) \textit{ for } x > 1/\sqrt{2} \end{cases} \\ f(x) &= \frac{\cos x}{\sqrt{x}} \textit{(singular integrand)} \end{aligned}$$

Discuss how you would handle these cases and how you would ensure that the approximations you obtained were accurate enough for your allowed tolerance.

3 Taylor Polynomial Approximation

3.1 Linear and Quadratic Approximation

For x near a we have the linear (tangent line) approximation:

$$f(x) \approx f(a) + f'(a)(x - a)$$

As an example, take $a = 0$ and $f(x) = e^x$ to get the linear approximation

$$e^x \approx 1 + x \tag{8}$$

as shown in Figure 4. The approximation is called *linear* because the original function is approximated by a straight line. It is also called *first order* approximation because the approximating function is a first order polynomial. This is an artificial example. The function e^x is well-known and easy to evaluate on your mech2 calculator. However, imagine a (differentiable) function that is hard to evaluate. It would be great to replace it with a simple linear function that was "accurate enough" for values of x "near" a. We will consider in more detail the errors from such an approximation below.

A better approximation is the quadratic one

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

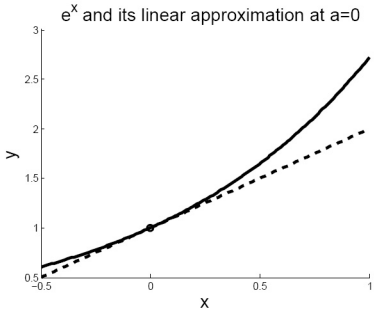

Figure 4: Linear approximation of e^x near $x = 0$

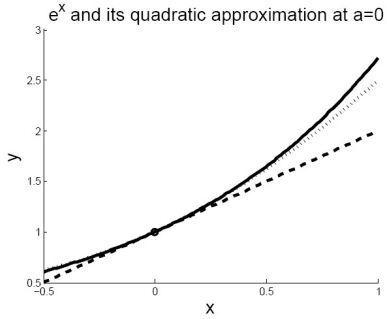

Figure 5: Quadratic approximation of e^x near $x = 0$ (dotted). The linear approximation (dashed) is also shown for comparison.

Note that the values, derivatives and second derivatives of the function and its approximation are the same at $x = a$. This is called a quadratic approximation or a second order Taylor Polynomial approximation of the function at the point $x = a$.

Considering $a = 0$ and $f(x) = e^x$ again we get the quadratic approximation

$$e^x \approx 1 + x + \frac{x^2}{2}$$

shown in Figure 5.

Exercise 8 *Find the quadratic Taylor approximation of e^x based at $a = 1$. Convert it to standard quadratic form, that is write it in the form*

$$c_0 + c_1 x + c_2 x^2$$

with constants c_0, c_1, and c_2.

3.2 Error Expression for Linear Approximation

The error expression for linear approximation is given in the following theorem:

Theorem 1

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

for some ξ between a and x.

Note that the left hand side (LHS) above is the difference between the value of the exact function and the value of the linear approximation at x (the error). This expression makes sense:

- The approximation gets worse (the error gets larger) as $|x-a|$ gets larger, that is as we move away from the point a at which the approximation is based.
- If f'' is large near a then the approximation may not be accurate.
- If $f''(x) > 0$ (f is concave up) for x near a then the linear approximation will underestimate the function values.

The proof of this theorem, together with the proof of Rolle's Theorem below (used as a lemma, as a part of the proof of the theorem above) are given in an appendix to these notes.

Theorem 2 *If f is differentiable, $f(a) = 0$ and $f(b) = 0$ then there is a point ξ in (a, b) at which $f'(\xi) = 0$.*

Exercise 9 *If $f(2) = 1$ and $f'(2) = 5$ write down the linear approximation to f for values of x near 2. If it is known that $|f''(x)| < 3$ for all x, determine the largest interval of x values around $a = 2$ for which you can guarantee that the linear approximation found above is accurate to at least three significant figures.*

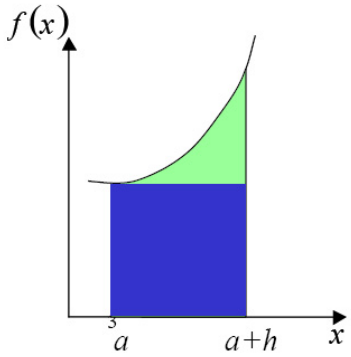

Figure 6: Left Riemann Sum approximation in one subinterval.

3.3 Application to Left Riemann Sums

The error estimate in Theorem 1 can be used to prove the error estimate for Left Riemann Sums in (2). Consider

$$F(x) = \int_a^x f(s)ds.$$

Standard application of the error equation for linear approximation gives

$$F(x) = F(a) + F'(a)(x-a) + \frac{1}{2}F''(\xi)(x-a)^2$$

where ξ is between x and a and we have moved the linear approximation to the RHS. Note that $F(a) = 0$ and also that $F'(a) = f(a)$ and $F''(\xi) = f'(\xi)$ by the Fundamental Theorem of Calculus. Now

$$\int_a^{a+h} f(x)dx = F(a+h) = f(a)h + \frac{1}{2}f'(\xi)h^2.$$

where ξ is between a and $a+h$. Apply this to one subinterval of Left Riemann Sums as shown in Figure 6. Now sum over all subintervals (N subintervals length $h = (b-a)/N$):

$$\begin{aligned}\int_a^b f(x)dx - L_N &= \frac{1}{2}(f'(\xi_1) + \cdots f'(\xi_j) + \cdots f'(\xi_N))h^2 \\ &= \frac{1}{2}(b-a)(f'(\xi_1) + \cdots f'(\xi_j) + \cdots f'(\xi_N))/Nh\end{aligned}$$

where $h = (b-a)/N$ was used to get the second line and each ξ_j is in the j'th subinterval $[a+(j-1)h, a+jh]$. Note that the RHS of the second line above contains an average of N values of a continuous function on an interval. Such an average is between the largest and smallest values on the interval. By the Intermediate Value Theorem, this average value is attained at a point ξ in the interval:

$$\int_a^b f(x)dx - L_N = \frac{1}{2}f'(\xi)(b-a)h$$

Exercise 10 *Consider the argument above carefully. Show that as $N \to \infty$ the values $f'(\xi)$ (remember the ξ can be different for each N) tend to the average value of $f'(\xi)$, that is*

$$f'(\xi) \to \frac{1}{b-a}\int_a^b f'(x)dx.$$

This property leads to the regular behaviour of the error as $N \to \infty$ we saw in the example of section 2.3.

3.4 Taylor Polynomials and Series

Higher order (n) polynomial approximation ($P_n(x)$) can also be used:

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

Error formula

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

where ξ is a point between a and x (that is not known).

In some cases, $P_n(x) \to f(x)$ as $n \to \infty$ for x in an interval around a. The "infinite" order polynomial is called a Taylor Series. Some common series are given below (based at $a = 0$, they can also be called McLaurin Series):

$$\begin{aligned}
\sin x &= x - \frac{x^3}{6} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots \\
\cos x &= 1 - \frac{x^2}{2} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots \\
e^x &= 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{4!} + \cdots \\
\frac{1}{1-x} &= 1 + x + x^2 + x^3 + \cdots \quad \text{for } |x| < 1
\end{aligned}$$

4 Interpolation

Suppose you have limited information about a function $f(x)$. The function (and possibly its derivatives) might be determined experimentally at certain points. Perhaps f is an analytic function but difficult to compute values of. Some values might be precomputed and stored in a look-up table. In either case, you might want to know (as accurately as possible) the value of f at values of x that are not where measurements were done, or in your look-up table.

We will learn in this section how to approximate functions using *polynomials that agree with the data you have*. The approximating polynomials will be be accurate in a certain limited interval. You might wonder why polynomials are always used. They are convenient to work with and easy to evaluate. From the last section on Taylor polynomials, it is clear that polynomials can approximate differentiable functions as accurately as desired.

If you have n pieces of data about the function (function or derivative values at certain points) you can fit an $n-1$ degree polynomial through the data

$$c_0 + c_1 x + c_2 x^2 + \cdots c_{n-1} x^{n-1}. \tag{9}$$

In general, the coefficients $c_0, c_1, \cdots c_{n-1}$ are determined from the given data by solving a linear system of equations. However, it some important cases, finding the coefficients is more straight forward.

We have already seen a kind of interpolation in the last section on Taylor Polynomials. If $f(a)$ and $f'(a)$ are known ($n = 2$ pieces of data) we can fit a first order ($n - 1 = 1$) polynomial to the data

$$f(x) \approx f(a) + f'(a)(x - a).$$

If $f(a)$, $f'(a)$ and $f''(a)$ are known ($n = 3$ pieces of data) we can fit a second order ($n - 1 = 2$) polynomial to the data

$$f(x) \approx f(a) + f'(a)(x - a) + \frac{f''(a)}{2}(x - a)^2.$$

No linear solve is needed in this case. If it is important to get the approximating polynomial in the simpler form (9) we can expand the powers of $x - a$ to reach this form, as we did in Exercise 8.

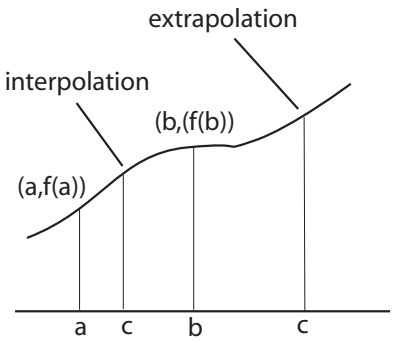

Figure 7: Interpolation and Extrapolation

4.1 Linear Interpolation

Suppose you know the value of $f(a)$ and $f(b)$ ($n = 2$ data) and wanted to estimate the value of $f(c)$. If c is in $[a, b]$ then this is called *interpolation*; If c is outside $[a, b]$ then this is called *extrapolation*. This is illustrated in Figure 7. Since we have $n = 2$ data we can fit a linear polynomial to the data. Graphically, this must be the line segment $S(x)$ between $(a, f(a))$ and $(b, f(b))$ as shown in Figure 8. Note that this is *not* the same idea as least squares line fitting. Doing the algebra we get

$$\begin{aligned} f(x) &\approx S(x) = f(a) + \frac{f(b) - f(a)}{b - a}(x - a) \\ &= \frac{b - x}{b - a} f(a) + \frac{x - a}{b - a} f(b). \end{aligned}$$

Note that when $a < x < b$ (interpolation), the linear approximation of $f(x)$ is a weighted average of the known values $f(a)$ and $f(b)$.

Exercise 11 *Vapour saturation pressure P_{sat} depends on temperature, $P_{sat}(T)$ where P_{sat} is in bar and T is in $^\circ C$. Experimental values of P_{sat} are found in "steam tables":*

$$\begin{aligned} P_{sat}(25) &= 0.03168 \\ P_{sat}(30) &= 0.04241 \end{aligned}$$

What is a good estimate of $P_{sat}(27)$? Use linear interpolation.

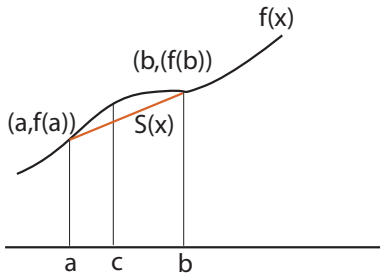

Figure 8: Linear interpolation using $S(x)$.

An error estimate is known for linear interpolation:

$$|f(x) - S(x)| \leq \frac{h^2}{8} \max_{s \in [a,a+h]} |f''(s)|$$

where $h = b - a$, the distance between the data points. Extrapolation is much less accurate. That the error can be larger as h gets larger or for functions with large values of $|f''|$ makes sense. The proof is an assignment question. The use of the estimate is shown in the Example below.

Example 3 *Say you knew from tables that*

$$\begin{aligned} \sin(0.9) &= 0.7833 \\ \sin(1) &= 0.8415 \end{aligned}$$

And you wanted to estimate $\sin(0.95)$. *Use linear interpolation:*

$$\sin(0.95) \approx \frac{1}{2}(\sin(0.9) + \sin(1)) = 0.8124$$

Exact $\sin(0.95) = 0.8134$, *error* ≈ 0.0010. *The error bound above is*

$$\frac{0.1^2}{8} \sin(1) = 0.00105$$

4.2 Fancy Interpolation

If we knew the values of $f(a-h)$, $f(a)$, $f(a+h)$ ($n=3$ data) we could get use a quadratic interpolation in $[a-h, a+h]$.

$$Q(x) = c_0 + c_1 x + c_2 x^2$$

Remember, the idea is that the coefficients are chosen so that $Q(x)$ matches the data.

$$\begin{aligned} Q(a-h) &= c_0 + c_1(a-h) + c_2(a-h)^2 = f(a-h) \\ Q(a) &= c_0 + c_1 a + c_2 a^2 = f(a) \\ Q(a+h) &= c_0 + c_1(a+h) + c_2(a+h)^2 = f(a+h) \end{aligned}$$

This is a linear system for c_0, c_1 and c_2! You can solve for the coefficients, then use $Q(x)$ to approximate $f(x)$.

Exercise 12 *Find the quadratic function $Q(x)$ that matches the data $f(-1) = 1$, $f(0) = 0$, $f(1) = 2$. Use $Q(x)$ to estimate $f(1/2)$.*

You can mix and match data of function values and derivatives at different points as shown in the exercise below.

Exercise 13 *Experimental measurements determine that a function $f(x)$ satisfies $f(0) = 1$, $f'(0) = 1$, and $f(1) = 3$. Estimate $f(1/2)$ using*

(a) *tangent line approximation.*

(b) *linear interpolation.*

(c) *a quadratic interpolation using all the information.*

There is a way to avoid the linear system for the polynomial coefficients (like is possible for Taylor Polynomials) using something called *Lagrange Interpolating Polynomials*. You can learn about these in a more advanced class on numerical methods, like Math 405. Another advanced technique of interpolation is *cubic splines*, which is implemented in MATLAB.

4.3 Richardson Extrapolation

Here is an interesting application of extrapolation. Consider a numerical method to compute approximate solutions to a problem. Suppose the number A is the exact answer of the problem. Suppose the approximations depend on an interval size h. Label by $f(h)$ the approximation computed with step size h. For the numerical integration methods, h can't actually be any number, it must be $(b-a)/N$ where N is an integer (an even integer for Simpsons Rule), but the idea below still works.

Suppose you know that the numerical method you are using is a convergent method of order p, that is

$$f(h) \approx A + Ch^p \tag{10}$$

for h small and some constant C we do not know. Remember our idea to compute $f(h)$ and $f(h/2)$ to see about how accurate our approximation is. The idea of Richardson extrapolation is to use these two approximate values and the known error behaviour (10) to get a more accurate approximation. The algebra begins with

$$\begin{aligned} f(h) &\approx A + Ch^p \\ f(h/2) &\approx A + Ch^p/2^p. \end{aligned}$$

We want the value of A without the errors associated with C. To eliminate C, multiply the second equation below by 2^p and subtract the second equation from the first to obtain

$$f(h) - 2^p f(h/2) \approx (1-2^p)A$$

or rewriting

$$A \approx \frac{2^p f(h/2) - f(h)}{2^p - 1}. \tag{11}$$

Equation (11) specifies Richardson Extrapolation of a numerical scheme of order p. The resulting scheme is convergent and of order higher than p (so converges to accurate answers with less work).

Exercise 14 *Show that the Richardson Extrapolation of the Trapezoidal Rule is Simpson's Rule.*

Exercise 15 *Write the Richardson Extrapolation of Left Riemann Sums. Identify the resulting scheme.*

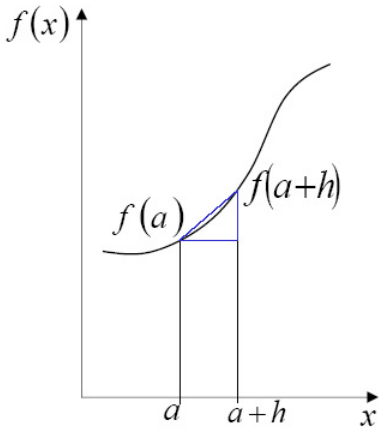

Figure 9: Graphical interpretation of the derivative as a tangent line slope.

5 Differentiation

The derivative of a function is given as

$$f'(a) = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h} \tag{12}$$

The graphical interpretation of the derivative is the tangent line slope as shown in Figure 9. There are many applications of derivatives some of which you saw last year: approximation, related rates, optimization (maximum and minimum), and differential equations.

5.1 Euler Difference Formulas

Suppose you had some experimental values for a function f and wanted to estimate the derivative of f. You might have an analytic function f in front of you that was so complicated that finding its derivative analytically was too much work. In either case, you would want to approximate the derivative numerically. Considering the definition of the derivative in (12), a natural idea is to use this expression with an h that is "small". We want to think of h as always positive and so have two cases:

Forward Euler Difference Approximation:

$$f'(a) \approx \frac{f(a+h) - f(a)}{h} \tag{13}$$

h	FE	FE error	BE	BE error
1/10	0.4974	0.0429	0.5814	-0.0411
1/20	0.5190	0.0213	0.5611	-0.0208
1/40	0.5297	0.0106	0.5508	-0.0105
1/80	0.5350	0.0053	0.5455	-0.0052
1/160	0.5377	0.0026	0.5429	-0.0026

Table 3: Errors in FE and BE approximation of $d/dx \sin x$ at $x = 1$

Backward Euler Difference Approximation:

$$f'(a) \approx \frac{f(a) - f(a-h)}{h}$$

Exercise 16 *Approximate*

$$\frac{d}{dx} \sin x$$

at $x = 1$ *using* $h = 0.1$ *using FE and BE.*

Consider again the approximation of

$$\frac{d}{dx} \sin x$$

at $x = 1$. (exact value $\cos 1 = 0.5403$). Numerical results are shown in Table 3. The results show clearly that both FE and BE are first order accurate methods (they converge to the correct answer as $h \to 0$ and the error is approximately Ch). Notice also that the errors for BE are almost equal in magnitude but opposite in sign to those of FE (this is generally true, not just for this example).

We would have a much more accurate approximation of the derivative if we averaged FE and BE (the error would approximately cancel).

$$f'(a) \approx \frac{1}{2}\frac{f(a+h) - f(a)}{h} + \frac{1}{2}\frac{f(a) - f(a-h)}{h} = \frac{f(a+h) - f(a-h)}{2h}$$

this is known as the centred difference formula.

The error estimates for difference formulae are easy to work out using Taylor Polynomials:

$$f'(a) - \frac{f(a+h) - f(a)}{h} \quad = \quad -\frac{1}{2}hf''(\xi)$$

h	FE error
10^{-4}	4.2e-5
10^{-6}	6.9e-7
10^{-8}	2.3e-6
10^{-10}	0.030

Table 4: Errors in FE approximation of $d/dx \sin x$ at $x = 1$ for h very small, showing the effect of round off errors in floating point computations.

$$\begin{aligned} f'(a) - \frac{f(a) - f(a-h)}{h} &= \frac{1}{2} h f''(\xi) \\ f'(a) - \frac{f(a+h) - f(a-h)}{2h} &= -\frac{1}{6} h^2 f^{(3)}(\xi) \end{aligned}$$

Exercise 17 *Show the error expression for Forward Euler approximation, the first expression in the list above.*

Exercise 18 *Show the error expression for centred differencing, the last expression in the list above.*

Exercise 19 *Knowing that centred differencing above is second order accurate, use Richardson Extrapolation to make a higher order scheme for approximating the first derivative.*

As a practical note, you can determine when difference approximations have converged to a desired accuracy by successively halving h and looking carefully at the resulting sequence of values.

5.2 Roundoff Errors and Noise

Consider using FE to approximate

$$\frac{d}{dx} \sin x = \cos x$$

at $x = 1$ (exact value $\cos 1 = 0.5403$) as before, but now take h to be *very* small. The results are shown in Table 4. Note that for very small h, the values are not at all accurate. This is not a problem with the computational method (which can be proved to converge as $h \to 0$) but with the use of finite precision arithmetic in my mech2 calculator. The loss of precision

comes when two numbers of almost the same size are subtracted, as in the computation of derivative approximations using FE, as can be seen from the formula (13).

Numerical differentiation is also very sensitive to noise in experimental data values. As a rule of thumb, interpolation and numerical integration are not sensitive to floating point errors and less affected by noise.

5.3 Deriving Difference Formulas

Note that the formulas for FE, BE and centred differencing are all linear combinations of the given function values, with coefficients that depend on h. When you take these combinations, and use Taylor polynomial approximations for the function values, the biggest value that shows up is the derivative and the next term (multiplied by h or h^2) is the error. To derive a difference formula, you take a linear combination of the known function values, expand the values in Taylor polynomials, and match the coefficients by solving a linear system. This is illustrated in an example below:

Example 4 *Find an approximation formula for $f''(a)$ when $f(a)$, $f(a-h)$ and $f(a+h)$ are known.*

- *Following the recipe above, we should look for an approximation of the form*
$$f''(a) \approx c_1 f(a) + c_2 f(a+h) + c_3 f(a-h) \tag{14}$$
with c_1, c_2 and c_3 to be determined (they will depend on h but not values of f).

- *Expand $f(a-h)$ and $f(a+h)$ in Taylor Polynomials around the base point a:*
$$\begin{aligned} f(a+h) &\approx f(a) + f'(a)h + \frac{1}{2}f''(a)h^2 \\ f(a-h) &\approx f(a) - f'(a)h + \frac{1}{2}f''(a)h^2. \end{aligned}$$

- *Put these in the form (14):*
$$\begin{aligned} f''(a) \approx\ & c_1 f(a) + c_2(f(a) + f'(a)h + \frac{1}{2}f''(a)h^2) + \\ & c_3(f(a) - f'(a)h + \frac{1}{2}f''(a)h^2) \end{aligned}$$

- *Reorganizing the LHS above we obtain*

$$f''(a) \approx (c_1 + c_2 + c_3)f(a) + h(c_2 - c_3)f'(a) + \frac{h^2}{2}(c_1 + c_2)f''(a).$$

For this to be any good as an approximation, the LHS should have no $f(a)$ terms, no $f'(a)$ terms and exactly one $f''(a)$ term:

$$\begin{aligned} c_1 + c_2 + c_3 &= 0 \\ c_2 - c_3 &= 0 \\ c_2 + c_3 &= \frac{2}{h^2} \end{aligned}$$

- *Recognize the system of equations above as a linear system. It has solution $c_1 = -2/h^2$, $c_2 = c_3 = 1/h^2$. Going back to (14) we see that*

$$f''(a) \approx \frac{f(a-h) - 2f(a) + f(a+h)}{h^2}.$$

This is known as the centred difference approximation of the second derivative.

Exercise 20 *Prove the centred difference approximation to the second derivative derived above is second order accurate.*

Exercise 21 *Find the second order formula for $f'(a)$ when $f(a)$, $f(a-h)$ and $f(a-2h)$ are known. This is known as second order, one-sided (backward) differencing.*

6 Summary

In these notes (that correspond to the first four lectures of Mathematics in Mech 221) you learned how to

- Numerically approximate function values, integrals and derivatives using only finite information about the function.
- Identify whether a method converges, and what order of accuracy it has.
- Use Richardson extrapolation to get more accurate approximations.

Appendix: Proof of Rolle's Theorem and the Error Expression for Linear Approximation

See the handwritten pages that follow.

Mech 221 Math lecture 2 Notes
Brian Wetton (wetton@math.ubc.ca)

I. Rolle's Theorem: If $f(x)$ is differentiable and $f(a)=0$ and $f(b)=0$ then there is a point ξ in (a,b) at which $f'(\xi)=0$.

Proof: there are three cases:

(i) f has some positive values in $[a,b]$.

(ii) f has " negative " " "

(iii) f is identically zero in $[a,b]$.

Case (i) The maximum of f on $[a,b]$ must then be positive and so can't be attained at the end points. Therefore, at a point ξ where the maximum is attained, $f'(\xi)=0$.

Case (ii) similar argument on minimum.

Case (iii) $f'(x)=0$ at all points x in $[a,b]$.

II. Use Rolle's Theorem to prove the remainder expression for linear approximation (Taylor approximation of order $n=1$).

Pick the base point a and a particular value of x - call it b - where the approximation is made.

We want to show that

$$f(b) - [f(a) + f'(a)(b-a)] = \frac{f''(\xi)}{2}(b-a)^2 \quad (\star)$$

for some ξ in $[a,b]$. Consider the somewhat tricky function

$$q(x) = f(x) - [f(a) + f'(a)(x-a)] - (x-a)^2 \underbrace{\frac{1}{(b-a)^2}\{f(b) - [f(a) + f'(a)(b-a)]\}}_{\text{does not depend on } x,\ \text{a big constant } K.}$$

$q(a) = 0$ and $q(b) = 0$ so (Rolle)

$q'(\xi_1) = 0$ for some ξ_1 in $[a,b]$.

$q'(x) = f'(x) - f'(a) - 2(x-a)K.$

So $q'(a) = 0.$

Rolle again, $q''(\xi) = 0$ for ξ in $[a, \xi_1]$

(we can say ξ in $[a,b]$).

$q''(x) = f''(x) - 2K.$

writing out $q''(\xi) = 0$ gives $(\star)$.