620123 - LECTURE SUMMARY 4
APPROXIMATION OF INTEGRALS AND ODE'S
Marty Ross
August 15, 2006

This is a summary of my lectures 9 and 10 (roughly equal to lecture 8 and 9 in the subject guide). See the indicated sections of Thomas [T] for further discussion and worked examples.

1 Approximation of Integrals ([T, 8.7])

We have various techniques for computing integrals, but the reality is that many integrals we run into, \( e^x \) for example, cannot be computed exactly. In such cases, we may need to resort to numerical techniques to approximate the integral. The definition of the integral leads easily to approximations by \textbf{Riemann sums} (see [1.1]), it is also a good introduction to the issues involved. However, of much more practical value are the \textbf{Trapezoid Rule} ([1.2]) and \textbf{Simpson’s Rule} ([1.3]).

1.1 Riemann Sums ([T, 5.1-5.3, 325-352])

(This section is really just for motivation and to establish notation: your focus should be upon the subsequent sections).

Suppose \( f(x) \) is a bounded function on \([a, b]\). Then, we define the integral of \( f \) over \([a, b]\) in terms of \textbf{Riemann sums}. We use a \textbf{partition} \( \{a = x_0, x_1, \ldots, x_{n-1}, x_n = b\} \) to split \([a, b]\) into \textbf{subintervals}, and then place a rectangle of height \( f(x_k) \) over the \( k \)th subinterval \([x_{k-1}, x_k]\), where \( x_{k-1} \leq c_k \leq x_k \).

The corresponding \textbf{Riemann Sum} is then

\[
\mathcal{R} = \sum_{k=1}^{n} f(c_k) \Delta x_k
\]

where

\[
\Delta x_k = x_k - x_{k-1}.
\]

We say \( f \) is \textbf{Riemann integrable} on \([a, b]\) if there is a number \( I \) such that

\[
\lim_{n \to \infty} \mathcal{R} = I \text{ as } \max(\Delta x_k) \to 0.
\]

(The limit is taken over all possible partitions). We then say that \( I \) is the \textbf{integral} of \( f \) over \([a, b]\), and we write

\[
I = \int_{a}^{b} f(x) \, dx.
\]

(The key theorem in this context is that continuous functions are integrable. [T, p345])

Thus, sums of rectangles are used to define integrals. But we can then turn this around, using (\(\bullet\)) to approximate integrals. In doing so, we make things easy by using a \textbf{regular partition}, where all the subintervals are of equal length \( h \). Since there are \( n \) subintervals, we have

\[
\Delta x_k = h = \frac{b-a}{n}.
\]

There are various ways to choose the height \( f(c_k) \) of the \( k \)th rectangle. Most common is to either use the \textbf{left endpoint}, giving \( c_k = x_{k-1} \), or the \textbf{right endpoint}, giving \( c_k = x_k \). This gives the two Riemann sum approximations,

\[
\begin{align*}
L &= h \cdot \left( f(x_0) + f(x_1) + \cdots + f(x_{n-1}) \right) = h \cdot \sum_{k=1}^{n} f(x_{k-1}), \\
R &= h \cdot \left( f(x_1) + \cdots + f(x_{n-1}) + f(x_n) \right) = h \cdot \sum_{k=1}^{n} f(x_k).
\end{align*}
\]
### 1.1.1 Example

Consider the integral

\[ I = \int_1^2 x^2 \, dx. \]

Of course we can trivially compute

\[ I = \frac{7}{3}, \]

so there is no real point in approximating this integral. But it is a good example, since it will permit us to look at how good our approximations really are.

If we use four subintervals, then \( n = 4 \), and from (\( \bullet \)), \( h = \frac{1}{4} \). Given \( f(x) = x^2 \), we then easily compute

\[
\begin{align*}
L &= \frac{1}{4} \left( 1^2 + \left( \frac{3}{4} \right)^2 + \left( \frac{1}{2} \right)^2 + \left( \frac{1}{4} \right)^2 \right) = \frac{63}{32} \\
R &= \frac{1}{4} \left( \left( \frac{5}{4} \right)^2 + \left( \frac{3}{2} \right)^2 + \left( \frac{7}{4} \right)^2 + 2^2 \right) = \frac{87}{32}.
\end{align*}
\]

Note that in this case we know \( I \), so we know precisely the \textit{error} in these approximations:

\[
\begin{align*}
E_L = L - I &= -\frac{3}{96} \\
E_R = R - I &= \frac{37}{96}
\end{align*}
\]

As we could have predicted from the graph, \( L \) is an underestimate of \( I \) (giving a negative error), and \( R \) is an overestimate.

### 1.1.2 Error Bounds for Riemann Approximations

We emphasise that, usually we do not know the exact integral \( I \) (that’s the whole point of approximating the integral!), and thus we \textit{cannot calculate the error} \( E \) precisely. But we still need to know \textit{something} about the error: otherwise, the “approximation” is meaningless.\(^1\) To this end, what we try to get in general is an \textit{error bound}, a \textit{guarantee} that the magnitude of the error is no larger than some specified amount. This bound may itself be quite inaccurate, but we don’t mind as long as the inaccuracy is in the right direction: the magnitude of the actual error must be less than the error bound.\(^2\)

Consider the left endpoint approximation, and let’s consider the error on the \( k \)'th subinterval: in effect, we’re approximating \( f(x) \) by the constant function \( f(x_k) \). At the left endpoint \( x_k-1 \), there is no error. But as \( x \) leaves \( x_k-1 \) and \( f(x) \) changes, there will be some error.

We can see this area graphically as the shaded region between the graph of \( f(x) \) and the rectangle. But suppose now that that the \textit{slope} of the graph \( f(x) \) is no greater than some constant \( M \):

\[ \left| f'(x) \right| \leq M_1 \text{ on } [a,b]. \]

Then the shaded region will be completely inside the pictured triangle, with base \( h \) and altitude \( Mh \). We thus conclude that the error on the \( k \)'th subinterval satisfies

\[ |E_k| \leq \frac{1}{2} Mh^2. \]

\(^1\)As a non-mathematical illustration: “George Bush found approximately 10000 WMD’s in Iraq.” This is true, give or take 10000 WMD’s. That is, \( \theta = 10000 \pm 10000 \). Unless we know something about the error, “approximation” is just a mushy nothing-verse.

\(^2\)To continue our political example, suppose we know that there were 10000 WMD’s in Iraq, give or take 7000. The error bound 7000 is a huge fraction of the total claimed, and may be much larger than the actual error, even so, together the WMD estimate and bound would allow us to conclude there are at least 3000 WMD’s in Iraq, which would be a very (politically) significant conclusion.
Then, if we add up the bounds over the $n$ subintervals and use (4) to eliminate $h$, we find

$$|E_L| = |L - I| \leq \frac{M_1(b - a)^2}{2n} \text{ where } |f'(x)| \leq M_1 \text{ on } [a,b]$$

Of course, the right endpoint error can be bounded in exactly the same manner:

$$|E_R| = |R - I| \leq \frac{M_1(b - a)^2}{2n} \text{ where } |f'(x)| \leq M_1 \text{ on } [a,b]$$

Note that the $n$ in the denominator of (4) means that doubling the number of subintervals halves the error bound.

Let’s apply these bounds to the previous example. In this case, $f' = 2x$, and so the maximum of $|f'|$ on $[1,2]$ is 4. So, with $n = 4$, (4) promises us that

$$|E_L|, |E_R| \leq \frac{4 \cdot 1^2}{2 \cdot 4} = \frac{1}{2}$$

This is consistent with the actual errors of $\frac{1}{24}$ and $\frac{7}{24}$.

1.1.3 Alternative Calculation of the Riemann Error Bound

Rather than relying upon a picture-proof (where we have in fact cheated slightly!), we’ll give a completely analytical justification for (1). Note that the error on the $k$th subinterval is exactly

$$E_{k}\text{th} = h \cdot f(x_k) - \int_{x_{k-1}}^{x_k} f(x) \, dx$$

To make the notation less cluttered, let’s assume that $n = 0$ and that we’re looking at the first subinterval (these assumptions will make no difference to the estimate we obtain). Then the error we’re interested in is

$$E_{1}\text{st} = h \cdot f(0) - \int_0^h f(x) \, dx$$

We can get to this expression in a sneaky way, by considering the integral

$$\int_0^h (x - h) f'(x) \, dx.$$

Integrating by parts, we see

$$\int_0^h (x - h) f'(x) \, dx = [(x - h)f(x)]_0^h - \int_0^h f(x) \, dx = h \cdot f(0) - \int_0^h f(x) \, dx = E_{1}\text{st}.$$

So, the only question is, the size of the integral (+). If $|f'(x)| \leq M_1$, as we are assuming, then we can easily calculate

$$\int_0^h (x - h) f'(x) \, dx \leq \int_0^h |x - h| f'(x) \, dx \leq \int_0^h (h - x) M_1 \, dx = M_1 \int_0^h (h - x) \, dx = \frac{M_1 h^2}{2}.$$

Done!

1.2 The Trapezoid Rule [T, 603-608]

To try account for the slope of $f$, it is natural to try to use trapezia to approximate $\int f$. As the diagram makes clear, the trapezium on the $k$th subinterval will exactly be the average of the left endpoint and right endpoint rectangles.

Thus, by (7), the total trapezoidal approximation will be

$$T = \frac{L + R}{2} \implies T = h \cdot \left( \frac{1}{2} f(x_0) + f(x_1) + f(x_2) + \cdots + f(x_{n-1}) + \frac{1}{2} f(x_n) \right)$$
1.2.1 Example
Consider the integral from § 1
\[ I = \int_1^2 x^2 \, dx. \]

Again using \( n = 4 \) subintervals, we find
\[ T = \frac{3}{4} \left( \left( \frac{3}{4} \right)^2 + \left( \frac{3}{4} \right)^2 + \left( \frac{3}{4} \right)^2 + \frac{1}{4} \cdot 2^2 \right) = \frac{75}{32}. \]

(Of course, we could have used our computations of \( L \) and \( R \) and just averaged them, but this is unnatural: normally, you would calculate \( T \) directly using (\ref{eq:trap}), rather than going via Riemann sums.)

Recalling that \( I = \frac{2}{3} \), we see that the error in this approximation is
\[ E_T = T - I = \frac{1}{96}. \]

This is a strikingly better approximation, for a miniscule amount of extra work.

1.2.2 Error Bounds for The Trapezoid Rule
The error in the Trapezoid Rule clearly comes from the failure of the function \( f(x) \) to be linear: so, it is not surprising that an error bound for \( T \) involves (the maximum of) the second derivative of \( f \). In fact, if we approximate \( \int_a^b f \) with \( n \) subintervals, then we obtain the error bound
\[ |E_T| = |T - I| \leq \frac{M_2(b-a)^3}{12n^2} \quad \text{where} \quad |f''(x)| \leq M_2 \quad \text{on} \quad [a,b]. \]

In the previous example, we have \( f'' = 2 \), and thus we can take \( M_2 = 2 \). We therefore conclude
\[ |E_T| \leq \frac{2 \cdot 1^3}{12} = \frac{1}{6}. \]

Here the error bound is exactly precise, indicating the quadratic function is in some sense the worst for the Trapezoid Rule. We emphasise that (\ref{eq:trap_err}) will not normally give the exact error, only a bound for that error.

Notice the \( n^2 \) in the denominator of (\ref{eq:trap_err}), implying that doubling the number \( n \) of subintervals will reduce the error bound by a factor of 4. This summarizes why the Trapezoid Rule is so much more accurate than Riemann sums.

To prove (\ref{eq:trap_err}), we perform a calculation similar to that in § 1.1.3. As there, we shall assume that \( a = 0 \), and we’ll get an estimate on the error in the first subinterval. To do this, we pull a strange integral out of our pocket and compute:
\[ \int_0^b x(h - x)f''(x) \, dx = \int_0^b [x(h - x)f''(x)]_0^b - \int_0^b (h - 2x)f'(x) \, dx \]
\[ = \int_0^b -(h - 2x)f'(x) \, dx - \int_0^b 2f(x) \, dx \]
\[ = h \cdot (f(0) + f(h)) - \frac{2}{3} \int_0^b f(x) \, dx \]

Dividing by 2, we have
\[ \frac{h}{2} (f(0) + f(h)) - \frac{1}{2} \int_0^b f(x) \, dx = \frac{h}{2} \int_0^b x(h - x) f''(x) \, dx \]

The LHS is exactly the trapezoidal error on the first subinterval. So, summing over \( n \) subintervals, and using \( |f''(x)| \leq M_2 \) (and using (\ref{eq:trap_err}) to eliminate \( h \)), we have
\[ |E_T| \leq \frac{nM_2}{2} \int_0^b x(x-h) \, dx = \frac{nM_2}{2} \frac{b^3}{6} - \frac{M_2 h}{12n^2} \]

Done!

1.3 Simpson’s Rule (\([T, 608-613]\))
It is clear that the Trapezoid Rule will produce errors to the extent that \( f(x) \) fails to be linear. Thus, the next step is to consider the approximation of \( f(x) \) on the subintervals by parabolas. Though one might imagine that this would be painful, the resulting formula, Simpson’s Rule, is surprisingly simple (and the associated error bound is impressive!).

A parabola is determined by 3 points. So, we demand that
\[ n \quad \text{is even for Simpson’s Rule}, \]
and we fit parabolas to the graph of \( f \) using two subintervals at a time. Then the formula for Simpson’s Rule (which we justify below) is
\[ \frac{3}{3} \int_0^b (f(x_0) + f(x_1) + 2f(x_2) + 4f(x_3) + \cdots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_0)) \]

\[ \frac{3}{3} \int_0^b (f(x_0) + f(x_1) + 2f(x_2) + 4f(x_3) + \cdots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_0)) \]

\[ \frac{3}{3} \int_0^b (f(x_0) + f(x_1) + 2f(x_2) + 4f(x_3) + \cdots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_0)) \]

\[ \frac{3}{3} \int_0^b (f(x_0) + f(x_1) + 2f(x_2) + 4f(x_3) + \cdots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_0)) \]

\[ \frac{3}{3} \int_0^b (f(x_0) + f(x_1) + 2f(x_2) + 4f(x_3) + \cdots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_0)) \]
We can also give the error bound for Simpson’s Rule (which we don’t justify below)

\[ |E_R| = |S - I| \leq \frac{M_4(b-a)^5}{180h^4} \text{ where } |f^{(4)}(x)| \leq M_4 \text{ on } [a, b]. \]

Note that doubling the number of subintervals reduces the error bound by a factor of 16: impressive!

### 1.3.1 Example

Returning to the example of §1.3, we obtain the approximation to \( \int_{-h}^{h} x^2 \, dx \) of

\[ S = \frac{1}{3} \left[ 1^2 + 4 \left( \frac{3}{2} \right)^2 + 2 \left( \frac{1}{2} \right)^2 + 4 \left( \frac{3}{2} \right)^2 + 2 \right] = \frac{7}{3}. \]

So, in this case, Simpson’s Rule gives exactly the right answer: this is to be expected, given we were approximating a parabola by parabolas!

### 1.3.2 Justification of Simpson’s Rule

To derive the formula \( \Phi \), we first consider the special case where \( x_0 = 0 \), and then \( x_{k+1} = \pm h. \) Let’s refer to the values of \( f(x) \) at these points as \( y_0, y_1 \) and \( y_2 \) (see the diagram). We are then interested in the parabola passing through the three points \((-h, y_0), (0, y_1), \) and \((h, y_2)\). We write the parabola fitting these three points as

\[ y = Ax^2 + Bx + C. \]

Then \( A, B \) and \( C \) are determined by the equations

\[
\begin{align*}
  y_0 &= Ah^2 + Bh + C \\
  y_1 &= C \\
  y_2 &= Ah^2 + Bh + C
\end{align*}
\]

We are not actually interested in the coefficients per se: what we want to know is the integral

\[ \int_{-h}^{h} y \, dx = \frac{h}{3} \left( y_0 + 4y_1 + y_2 \right). \]

(This is exactly the Simpson estimate for \( f \) on the subintervals in question).

Of course \( C = y_1 \) we know directly. And if we add the 1st and 3rd equations above, we find

\[ y_0 + y_2 = 2Ah^2 + 2C = 2Ah^2 + 2y_1. \]

Substituting in, we easily calculate that

\[ \int_{-h}^{h} y \, dx = \frac{h}{3} \left( y_0 + 4y_1 + y_2 \right). \]

If we now consider summing over all \( n \) subintervals, we’ll obtain \( \frac{n}{2} \) expressions of this form. The \( y \) terms with even index, except those corresponding to \( a = x_0 \) and \( b = x_n \), will appear in two such expressions: thus the even-index \( y \) terms (except \( y_0 \) and \( y_n \)) will appear with a factor of 2. This observation then readily leads to the formula \( \Phi \).
2 Approximation of Solutions to IVP’s

Consider an IVP,

\[
\begin{aligned}
  \frac{dy}{dx} &= F(x, y) \\
  y(x_0) &= y_0
\end{aligned}
\]

(IVP)

If \( F \) and \( F_y \) are continuous in a region around \((x_0, y_0)\) then the Existence and Uniqueness Theorem (Handout 3, §3) promises that (IVP) will have a solution, at least on some small interval \([x_0, x_0 + \delta]\). On the other hand, we may not be able to explicitly solve (IVP) (e.g. Handout 3, §3.1). In such a case, we may have to resort to the numerical approximation of the solution \(y(x)\). (This is a huge topic: we briefly describe two approximation methods here, but it amounts to the tiniest introduction to a subtle and important field of mathematics).

To permit the possibility of a numerical scheme, we consider \(x\) as changing in discrete steps of some size \(h\). So \(x_1 = x_0 + h\), \(x_2 = x_1 + h\) and so on. In general,

\[
x_{n+1} = x_n + h
\]

Then, we consider methods to obtain an approximation \(y_n\) to \(y\) at \(x_n\). In this regard, it is important to keep the meaning of these terms clear:

\[
\begin{aligned}
  y(x_n) &\quad \text{the value of the solution to (IVP) at } x = x_n \\
  y_n &\quad \text{the approximation to } y(x_n) \text{ obtained by some numerical scheme.}
\end{aligned}
\]

2.1 Euler’s Method

Starting with \(y(x_0) = y_0\), how can we then approximate \(y(x_1)\), the value of \(y\) at \(x_1\)?: One thing we do know, directly from the ODE, is the derivative of \(y\) at \(x_0\):

\[
y'_{(x_0)} = F(x_0, y_0).
\]

This means we can consider a linear approximation to \(y\) at \(x_0\):

\[
y_1 = y_0 + h \cdot y'_{(x_0)} = y_0 + h \cdot F(x_0, y_0).
\]

We can then take the RHS of (1) to define \(y_1\), our approximation to \(y(x_1)\):

\[
y_1 = y_0 + h \cdot F(x_0, y_0).
\]

Now, we’re at \((x_1, y_1)\). To determine \(y_2\), we apply the same process: a solution to the ODE passing through \((x_1, y_1)\) would have derivative \(F(x_1, y_1)\). So, a second linear approximation leads to us choosing

\[
y_2 = y_1 + h \cdot F(x_1, y_1).
\]

Continuing this way, we obtain Euler’s Method for approximating the solution to (IVP):

\[
\begin{aligned}
x_{n+1} &= x_n + h \quad &y_{n+1} &= y_n + h \cdot F(x_n, y_n)
\end{aligned}
\]

We make a few comments on Euler’s Method below, but we first give a simple example.

2.1.1 Example

Consider the IVP from Handout 3, §3.3.

\[
\begin{aligned}
  \frac{dy}{dx} &= x^2 + y^2 \\
  y(0) &= 1
\end{aligned}
\]

Here \(x_0 = 0\), and \(y_0 = y(x_0) = 1\).

Suppose we take stepsize \(h = 0.1\). Then \(x_1 = 0.1\), and we can calculate

\[
y_1 = y_0 + h \cdot F(x_0, y_0) = 1 + 0.1 ((0)^2 + 1^2) = 1.1.
\]

(In brief, the slope of \(y\) at \(x = 0\) is 1, and so the change in \(y\) is approximately \(0.1 \cdot 1 = 0.1\)).

At the next step, we have \(x_2 = 0.2\), and

\[
y_2 = y_1 + h \cdot F(x_1, y_1) = 1.1 + 0.1 ((0.1)^2 + (1.1)^2) = 1.222.
\]

It is obviously easy to continue the process: of course, the calculations will get messy, but (\(\Phi\)) is really something to program into a computer (which is trivial to do).
2.1.2 Some Comment on Euler’s Method

(a) Euler’s method is the simplest and most intuitive method for approximating IVP’s, but it is not obvious how good a scheme it is. In effect, we need an estimate on the error in (1) (which we’ll obtain in Week 11). Notice that, whatever the error in (1), this error is compounded as we iterate the Euler scheme: new errors are building upon previous errors. This smells (accurately) as if the Euler method may not be all that great, at least if the number of iterations is large. See (d) below.

(b) Intuitively, if we reduce the stepsize \( h \), the approximation should improve, which is indeed the case: see below. However, increasing the number of steps obviously increases the amount of computation (to get to a specified value of \( x \)). As well, it can make the effect of rounding error more important.

(c) Notice that Euler’s Method can’t be too great an approximation for the IVP above, since we know the actual solution explodes at least by \( x = 1 \) (see Handout 3, §3.4).

(d) After \( n \) steps, the error in Euler’s Method is

\[ E_n = y_n - y(x_n) \]

It is not at all obvious (and you don’t need to know it), but we can prove (in suitable circumstances) an error bound for \( E_n \) of the form

\[ |E_n| \leq e^{\alpha h} M h \]

Here, \( M \) and \( L \) are constants which depend upon the function \( F(x, y) \) (similar in spirit to the constants in the bounds for the Trapezoidal Rule and Simpson’s Rule). We have an optimistic and a pessimistic way of interpreting (•):

(i) (optimism) Doubling \( n \) and halving \( h \) (so, getting to the same \( x \) in twice the number of steps), then the error bound is halved.

(ii) (pessimism) For fixed stepsize \( h \), the error bound grows exponentially with \( n \).

In regard to (ii), we might hope to prove a better error bound; alas, (•) reflects the underlying reality of Euler’s Method.

2.2 The New and Improved, Lemon-Scented Euler’s Method

Either because of the previous discussion, or out of general optimism, we might hope to improve upon Euler’s method. There is in fact such an improvement which, with great imagination, is called the Improved (or Modified) Euler’s Method:

\[
\begin{align*}
x_{n+1} &= x_n + h \\
y_{n+1} &= y_n + \frac{h}{2} [F(x_n, y_n) + F(x_{n+1}, y_{n+1} + hF(x_n, y_n))]
\end{align*}
\]

It is not at all obvious from where (•) comes, and we discuss that below. We first consider a simple example.

2.2.1 Example

Considering the example from §2.1.1. For the improved Euler method, we have

\[
y_1 = y_0 + \frac{h}{2} [F(x_0, y_0) + F(x_1, y_0 + hF(x_0, y_0))]
\]

\[
= y_0 + \frac{h}{2} [F(0, 1) + F(0.1, 1 + 0.1F(0, 1))]
\]

\[
= 1 + \frac{0.1}{2} (1 + 0.1)^2 = 1.111.
\]

We can then calculate

\[
y_2 = y_1 + \frac{h}{2} [F(x_1, y_1) + F(x_2, y_1 + hF(x_1, y_1))]
\]

and so on. □

2.2.2 Motivation for the Improved Euler Method

To explain where (•) comes from, note that a solution to (IVP) satisfies

\[
y(x_1) - y(x_0) = \int_{x_0}^{x_1} \frac{dy}{dx} dx \implies y(x_1) = y_0 + \int_{x_0}^{x_1} F(x, y(x)) dx.
\]

Written as such, the question becomes how to approximate \( \int F(x, y(x)) \). If we use the left endpoint approximation (one subdivision), this gives exactly Euler’s method. On the other hand, if we approximate the integral with a trapezoid, we get

\[
y(x_1) \approx y_0 + \frac{h}{2} [F(x_0, y_0) + F(x_1, y_1)]
\]

Now, we can’t (easily) use this to define \( y_1 \), since we don’t know \( y(x_1) \) on the RHS. However, if we use a linear approximation for \( y(x_1) \) on the RHS, we have

\[
y(x_1) \approx y_0 + \frac{h}{2} [F(x_0, y_0) + F(x_1, y_0 + hF(x_0, y_0))].
\]

Then defining \( y_1 \) to be this approximation, we get exactly the Improved Euler Method. Finally, we remark that the Improved Euler Method satisfies an error bound of the form \( |E_n| \leq e^{\alpha h} M h^2 \). So, the error still grows exponentially with \( n \), but the convergence with respect to \( h \) is quadratic: a strong improvement over the linear convergence of the Euler Method.