Lecture 5

Numerical Techniques: Numerical Differentiation and Integration

Variable Force in One Dimension, Simple Pendulum

IKC-MH.55 *Scientific Computing with Python* at November 10, 2023

Dr. Cem Özdoğan
Engineering Sciences Department
İzmir Kâtip Çelebi University

Contents

Variable force in one dimension I

- Consider the motion of a particle of mass m moving along the x-axis under the action of a force F.

Write Newton's second law ($F = ma$) in terms of velocity for a most general force **F(x,v,t)**

$$\begin{aligned} \frac{dv}{dt} &= \frac{F(x,v,t)}{m} \\ \frac{dx}{dt} &= v \end{aligned} \tag{1}$$

- In principle, the functions $v = v(t)$ and $x = x(t)$ can be found by solving Equation 1 for every $F(x, v, t)$ function (i.e., constant acceleration for F = constant, or harmonic oscillating motion for F= -kx).
- However, for more complex forces $F(x, v, t)$ the analytical solution may not always be available.
- In this case, we will consider the numerical solution.

Variable force in one dimension II

- We want to find the solutions of the Equation 1 at the equally spaced times $t_1, t_2, \ldots, t_N$ and x_i and v_i.
- Write the velocity and position derivatives in the form of *forward-difference* derivative approximation. Take $dt = h$ as step length, then:

$$\frac{v_{i+1} - v_i}{h} = \frac{F(x, v, t)}{m} \longrightarrow \boxed{v_{i+1} = v_i + \frac{F_i}{m} h}$$

$$\frac{x_{i+1} - x_i}{h} = v_i \longrightarrow \boxed{x_{i+1} = x_i + v_i h}$$

- By given <u>initial</u> velocity v_1, <u>initial</u> position x_1 at the time $t_1 = 0$ and the values for $F_i = F(x_i, v_i, t_i)$; the values for v_i, x_i can be calculated for $i = 2.3, \ldots$ in a loop.

Numerical Differentiation and Integration I

- When the function is explicitly known, we can emulate the methods of calculus.
- If we are working with experimental data that are displayed in a table of $[x, f(x)]$ pairs emulation of calculus is **impossible**.
  - We must ***approximate*** the function behind the data in some way.
- **Differentiation with a Computer:**
  - Employs the interpolating polynomials to derive formulas for getting derivatives.
  - These can be applied to functions known explicitly as well as those whose values are found in a table.
- **Numerical Integration-The Trapezoidal Rule:**
  - Approximates, the integrand function with a *linear* interpolating polynomial to derive a very simple but important formula for numerically integrating functions between given limits.

Numerical Differentiation and Integration II

- We cannot often find the true answer numerically because the analytical value is the limit of the sum of an infinite number of terms.
- We must be satisfied with approximations for both derivatives and integrals but, for most applications, the **numerical answer is adequate**.
- The derivative of a function, $f(x)$ at $x = a$, is defined as

$$\frac{df}{dx}\Big|_{x=a} = lim_{\Delta x \to 0} \frac{f(a + \Delta x) - f(a)}{\Delta x}$$

- This is called a ***forward-difference*** approximation.
- The limit could be approached from the opposite direction, giving a ***backward-difference*** approximation.

Differentiation with a Computer I

- **Forward-difference** approximation. A computer can calculate an approximation to the derivative, *if a very small value is used for* Δx.

$$\boxed{\left.\frac{df}{dx}\right|_{x=a} = \frac{f(a+\Delta x) - f(a)}{\Delta x}}$$

  - Recalculating with smaller and smaller values of x starting from an initial value.
  - What happens if the value is not small enough?
  - We should expect to find an ***optimal value*** for x.
  - Because round-off errors in the numerator will become great as x approaches zero.

- When we try this for

$$f(x) = e^x sin(x)$$

at $x = 1.9$. **The analytical answer is 4.1653826.**

Differentiation with a Computer II

Apply Forward-difference approximation to $f(x) = e^x sin(x)$.
(**Example py-file:** myforwardderivative.py)

Step	Delta x	Numerical Derivative	Error	Error Ratio
0	0.05/ 1	4.0501022943559306	-0.1152803056440694	-
1	0.05/ 2	4.1095606458263845	-0.0558219541736156	2.0651427803033999
2	0.05/ 4	4.1379199156034474	-0.0274626843965526	2.0326474050228986
3	0.05/ 8	4.1517625462243757	-0.0136200537756244	2.0163418477614425
4	0.05/ 16	4.1586002903972030	-0.0067823096027970	2.0081734060042704
5	0.05/ 32	4.1619983544313754	-0.0033842455686246	2.0040831746005470
6	0.05/ 64	4.1636921950123451	-0.0016904049876549	2.0020324084108938
7	0.05/ 128	4.1645378187649840	-0.0008447812350161	2.0009973204752387
8	0.05/ 256	4.1649603066707641	-0.0004222933292359	2.0004607615861758
9	0.05/ 512	4.1651714696399722	-0.0002111303600278	2.0001544504553519
10	0.05/ 1024	4.1652770308974141	-0.0001055691025860	1.9999256871196425
11	0.05/ 2048	4.1653298064920818	-0.0000527935079182	1.9996606921732443
12	0.05/ 4096	4.1653561930434080	-0.0000264069565921	1.9992272768787172
13	0.05/ 8192	4.1653693860280327	-0.0000132139719673	1.9984117309630689
14	0.05/ 16384	4.1653759824112058	-0.0000066175887943	1.9967955667965809
15	0.05/ 32768	4.1653792810393497	-0.0000033189606503	1.9938738332606181
16	0.05/ 65536	4.1653809300623834	-0.0000016699376166	1.9874758297878456
17	0.05/ 131072	4.1653817542828619	-0.0000008457171381	1.9745817382726043
18	0.05/ 262144	4.1653821710497141	-0.0000004289502860	1.9715970959427951
19	0.05/ 524288	4.1653823852539062	-0.0000002147460938	1.9974765471860429
20	0.05/ 1048576	4.1653824970126152	-0.0000001029873848	2.0851689177747628
21	0.05/ 2097152	4.1653824970126152	-0.0000001029873848	1.0000000000000000
22	0.05/ 4194304	4.1653826087713242	0.0000000087713241	-11.7413726289747711
23	0.05/ 8388608	4.1653826832771301	0.0000000832771301	0.1053269259276438
24	0.05/ 16777216	4.1653829813003540	0.0000003813003540	0.2184029708243117
25	0.05/ 33554432	4.1653829813003540	0.0000003813003540	1.0000000000000000

Table: Forward-difference approximation for $f(x) = e^x sin(x)$.

Differentiation with a Computer III

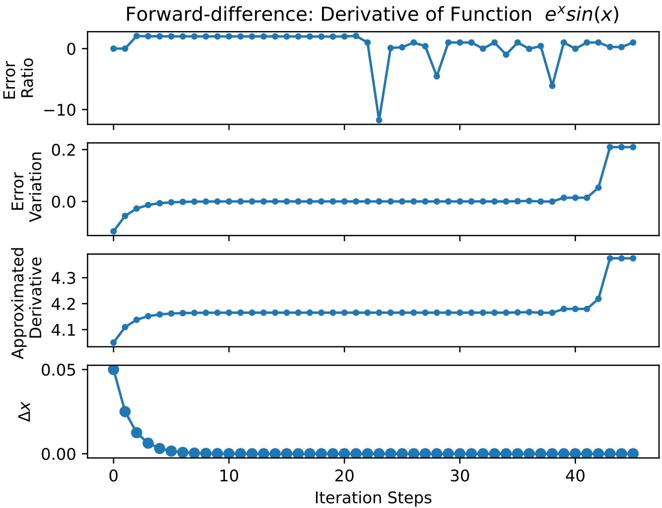


Figure: Forward-difference approximation for $f(x) = e^x sin(x)$.

Differentiation with a Computer IV

- Starting with $\Delta x = 0.05$ and halving Δx each time. Table gives the results.
- We find that the errors of the approximation decrease as Δx is reduced until about $\Delta x = 0.05/2097152$.
- Notice that each successive error is about 1/2 of the previous error as Δx is halved until Δx gets quite small, **at which time round off affects the ratio**.
- At values for Δx smaller than $0.05/2097152$, the error of the approximation increases due to round off.
- In effect, the best value for Δx is **when the effects of round-off and truncation errors are balanced**.
- If a backward-difference approximation is used; similar results are obtained.
- **Backward-difference** approximation.

$$\boxed{\left.\frac{df}{dx}\right|_{x=a} = \frac{f(a) - f(a - \Delta x)}{\Delta x}}$$

Differentiation with a Computer V

Apply Backward-difference approximation to $f(x) = e^x sin(x)$.
(**Example py-file:** mybackwardderivative.py)

Step	Delta x	Numerical Derivative	Error	Error Ratio
0	0.05/ 1	4.2665138904463618	0.1011312904463617	-
1	0.05/ 2	4.2176675936831032	0.0522849936831031	1.9342316661509737
2	0.05/ 4	4.1919610325982859	0.0265784325982859	1.9671962780256349
3	0.05/ 8	4.1787815600844169	0.0133989600844169	1.9836190593027312
4	0.05/ 16	4.1721096042468275	0.0067270042468275	1.9918168017711522
5	0.05/ 32	4.1687529872206142	0.0033703872206141	1.9959143583513135
6	0.05/ 64	4.1670695083894316	0.0016869083894315	1.9979669564331812
7	0.05/ 128	4.1662264750743816	0.0008438750743816	1.9990025071753801
8	0.05/ 256	4.1658046347765776	0.0004220347765775	1.9995391878008091
9	0.05/ 512	4.1655936336883315	0.0002110336883314	1.9998455218901035
10	0.05/ 1024	4.1654881129034038	0.0001055129034038	2.0000746972514616
11	0.05/ 2048	4.1654353474586969	0.0000527474586969	2.0003409834418382
12	0.05/ 4096	4.1654089634539559	0.0000263634539559	2.0007795179306953
13	0.05/ 8192	4.1653957709786482	0.0000131709786482	2.0016321231806500
14	0.05/ 16384	4.1653891745954752	0.0000065745954752	2.0033139221949763
15	0.05/ 32768	4.1653858771314844	0.0000032771314844	2.0062043608692623
16	0.05/ 65536	4.1653842269442976	0.0000016269442975	2.0142862232140093
17	0.05/ 131072	4.1653834027238190	0.0000008027238190	2.0267796458202478
18	0.05/ 262144	4.1653829859569669	0.0000003859569668	2.0798272552576837
19	0.05/ 524288	4.1653827857226133	0.0000001857226133	2.0781366361207314
20	0.05/ 1048576	4.1653826646506786	0.0000000646506786	2.8727094184239901
21	0.05/ 2097152	4.1653825715184212	-0.0000000284815789	-2.2699120324883144
22	0.05/ 4194304	4.1653825342655182	-0.0000000657344819	0.4332821689348803
23	0.05/ 8388608	4.1653825342655182	-0.0000000657344819	1.0000000000000000
24	0.05/ 16777216	4.1653823852539062	-0.0000002147460938	0.3061032715227527
25	0.05/ 33554432	4.1653823852539062	-0.0000002147460938	1.0000000000000000

Table: Backward-difference approximation for $f(x) = e^x sin(x)$.

Differentiation with a Computer VI

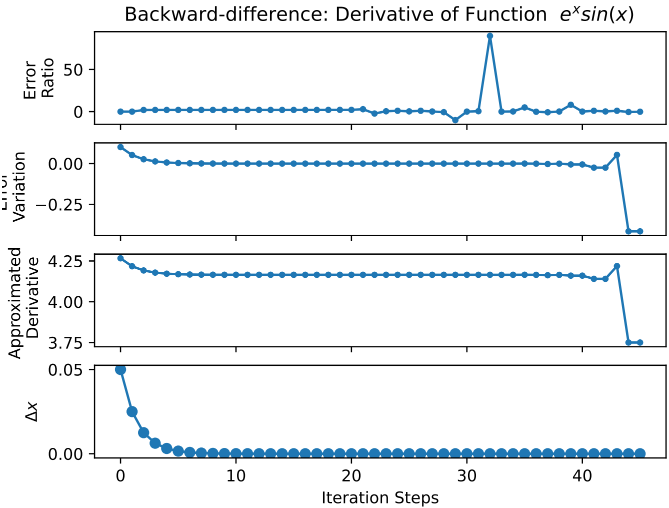


Figure: Backward-difference approximation for $f(x) = e^x sin(x)$.

Differentiation with a Computer VII

- It is not by chance that the errors are about **halved each time** for both forward- and backward-difference approximations.
- Look at this Taylor series where we have used h for Δx:

$$f(x+h) = f(x) + f'(x) * h + f''(\xi) * h^2/2$$

- Where the last term is the error term. The value of ξ is at some point between x and $x+h$.
- If we solve this equation for $f'(x)$, we get

$$f'(x) = \frac{f(x+h) - f(x)}{h} - f''(\xi) * \frac{h}{2} \tag{2}$$

- If we repeat this but begin with the Taylor series for $f(x-h)$, it turns out that

$$f'(x) = \frac{f(x) - f(x-h)}{h} + f''(\zeta) * \frac{h}{2} \tag{3}$$

- Where ζ is between x and $x-h$.
- The two error terms of Eqs. 2 and 3 are not identical though both are $O(h)$.

Differentiation with a Computer VIII

- If we add Eqs. 2 and 3, then divide by 2, we get the ***central-difference*** approximation to the derivative:

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} - f'''(\xi) * \frac{h^2}{6}$$

- We had to extend the two Taylor series by an additional term to get the error **because the $f''(x)$ terms cancel**.
- This shows that using a central-difference approximation is a much *preferred way* to estimate the derivative.
- Even though we use the same number of computations of the function at each step,
- We approach the answer **much more rapidly**.

$$\boxed{\left.\frac{df}{dx}\right|_{x=a} = \frac{f(x+h) - f(x-h)}{2h}}$$

Differentiation with a Computer IX

Apply Central-difference approximation to $f(x) = e^x sin(x)$.
(**Example py-file:** mycentralderivative.py)

Step	Delta x		Numerical Derivative	Error	Error Ratio
0	0.05/	1	4.1583080924011462	-0.0070745075988539	-
1	0.05/	2	4.1636141197547438	-0.0017684802452562	4.0003317073122826
2	0.05/	4	4.1649404741008667	-0.0004421258991334	3.9999471840998186
3	0.05/	8	4.1652720531543963	-0.0001105468456037	3.9994438259975551
4	0.05/	16	4.1653549473220153	-0.0000276526779848	3.9976904104779085
5	0.05/	32	4.1653756708259948	-0.0000069291740052	3.9907610869409487
6	0.05/	64	4.1653808517008883	-0.0000017482991117	3.9633801555226653
7	0.05/	128	4.1653821469196828	-0.0000004530803173	3.8586957876697707
8	0.05/	256	4.1653824707236708	-0.0000001292763292	3.5047430575889518
9	0.05/	512	4.1653825516641518	-0.0000000483358482	2.6745435112230909
10	0.05/	1024	4.1653825719004089	-0.0000000280995911	1.7201619782859658
11	0.05/	2048	4.1653825769753894	-0.0000000230246107	1.2204154723952758
12	0.05/	4096	4.1653825782486820	-0.0000000217513181	1.0585386401779320
13	0.05/	8192	4.1653825785033405	-0.0000000214966596	1.0118464227311168
14	0.05/	16384	4.1653825785033405	-0.0000000214966596	1.0000000000000000
15	0.05/	32768	4.1653825790854171	-0.0000000209145830	1.0278311363130717
16	0.05/	65536	4.1653825785033405	-0.0000000214966596	0.9729224623288757
17	0.05/	131072	4.1653825785033405	-0.0000000214966596	1.0000000000000000
18	0.05/	262144	4.1653825785033405	-0.0000000214966596	1.0000000000000000
19	0.05/	524288	4.1653825854882598	-0.0000000145117403	1.4813288542519762
20	0.05/	1048576	4.1653825808316469	-0.0000000191683531	0.7570676603134135
21	0.05/	2097152	4.1653825342655182	-0.0000000657344819	0.2916027111686003
22	0.05/	4194304	4.1653825715184212	-0.0000000284815789	2.3079648129953259
23	0.05/	8388608	4.1653826087713242	0.0000000087713241	-3.2471242096582569
24	0.05/	16777216	4.1653826832771301	0.0000000832771301	0.1053269259276438
25	0.05/	33554432	4.1653826832771301	0.0000000832771301	1.0000000000000000

Table: Central-difference approximation for $f(x) = e^x sin(x)$.

Differentiation with a Computer X

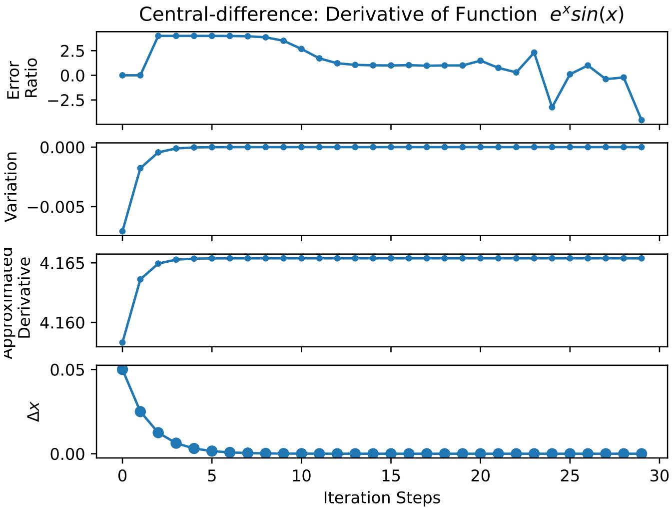


Figure: Central-difference approximation for $f(x) = e^x sin(x)$.

Variable force in one dimension III

Apply Forward-difference approximation to Simple Harmonic Motion. (**Example py-file:** shm.py)

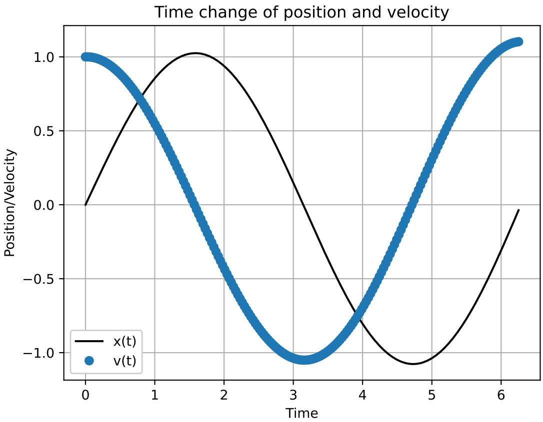


Figure: Time change of position and velocity in motion under the force F=-kx.

Simple Pendulum I

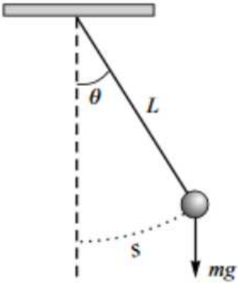


Figure: Simple pendulum.

- The motion of a simple pendulum, which consists of a point mass *m* suspended on the end of a rope of length *L*.
- Newton's second law for motion in the tangential direction:

$$F = ma \longrightarrow -mgSin\theta = m\frac{d^2s}{dt^2}$$

- Here s is the arc length and θ is the angle the rope makes with the vertical (see Figure).
- Since $s = L\theta$,

$$\frac{d^2\theta}{dt^2} = -\frac{g}{L}sin\theta$$

- This differential equation has no analytical solution.

Simple Pendulum II

- However, for small angle oscillations $sin\theta \approx \theta$ is approximated:

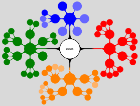
$$\frac{d^2\theta}{dt^2} = -\frac{g}{L}\theta \tag{4}$$

- This equation has a solution in terms of sinusoidal functions:

$$\theta(t) = \theta_0 cos\frac{2\pi t}{T}$$

$$\boxed{T = 2\pi\sqrt{\frac{L}{g}}}$$

- Here T is the period of the oscillation and θ_0 is the angular amplitude.
- This formula is small angle ($\theta_0 \leq 15^\circ$) amplitudes gives approximately good results.
- We want to find the exact solution of the pendulum problem **for each amplitude θ_0 numerically**.
- For this purpose, we will transform the problem into a numerical integral.



Simple Pendulum III

- Let's multiply both sides of the equation 4 by $d\theta/dt$ and rearrange:

$$\frac{d\theta}{dt}\frac{d^2\theta}{dt^2} = -\tfrac{g}{L}sin\theta\tfrac{d\theta}{dt}$$
$$\frac{1}{2}\frac{d}{dt}\left[\frac{d\theta}{dt}\right]^2 = -\tfrac{g}{L}\tfrac{d}{dt}cos\theta$$

- If the indefinite integral is taken side by side,

$$\frac{1}{2}\left(\frac{d\theta}{dt}\right)^2 = \frac{g}{L}cos\theta + C$$

- The initial velocity and angle values are used to determine the integration constant C.
  - If the pendulum is initially released from the maximum angle, its velocity will be zero.
  - So, if $d\theta/dt = 0$ and $\theta = \theta_0$ are substituted into the equation at time $t = 0$, $C = -(g/L)cos\theta_0$ is found.

Simple Pendulum IV

- If the constant C is put in place and the arrangement is made,

$$\begin{aligned} \frac{d\theta}{dt} &= \sqrt{\tfrac{2g}{L}(cos\theta - cos\theta_0)} \\ dt &= \sqrt{\tfrac{L}{2g}}\tfrac{d\theta}{\sqrt{cos\theta - cos\theta_0}} \end{aligned}$$

- Integrating the right side of this expression from $\theta = 0$ to the angle $\theta = \theta_0$, the left side will be one quarter of a period.

$$\frac{T}{4} = \sqrt{\frac{L}{2g}} \int_0^{\theta_0} \frac{d\theta}{\sqrt{cos\theta - cos\theta_0}}$$

Thus, we have written the period formula as an integral.

- For each given amplitude θ_0 we can calculate this integral numerically.
- However, since the value of the integrand diverges at the upper bound (for $\theta = \theta_0$), it is necessary to do a **variable replacement** first.

Simple Pendulum V

- For this purpose, first, the identities $cos\theta = 1 - 2sin^2(\theta/2)$ and $cos\theta_0 = 1 - 2sin^2(\theta_0/2)$ are inserted in the denominator,
- and then with the variable change $k = sin(\theta_0/2) = \frac{sin(\theta/2)}{sin\phi}$, the integral becomes:

$$\boxed{T = 4\sqrt{\frac{L}{g}}\int_0^{\pi/2}\frac{d\phi}{\sqrt{1 - sin^2(\theta_0/2)sin^2\phi}}}$$

- This integral whose denominator is never zero is known as an *elliptical integral of the first kind*.
- **There is no analytical solution.**

Numerical Integration - The Trapezoidal Rule I

- Given the function, $f(x)$, the **antiderivative** is a function $F(x)$ such that $F'(x) = f(x)$.
- The definite integral

$$\int_a^b f(x)dx = F(b) - F(a)$$

can be evaluated from the antiderivative.

- Still, there are functions that do not have an antiderivative expressible in terms of ordinary functions. Such as the function: $f(x) = e^x/log(x)$

```python
from sympy import *
x = symbols('x')
f = exp(x)/log(x)
df = integrate(f, x) # Function derivative in symbolic form
print(df)
# Integral(exp(x)/log(x), x)
```

Numerical Integration - The Trapezoidal Rule II

- Is there any way that the definite integral can be found when the antiderivative is unknown?
- We can do it numerically by using the **composite trapezoidal rule**
- The definite integral is the area between the curve of $f(x)$ and the x-axis.
- That is the principle behind all numerical integration;

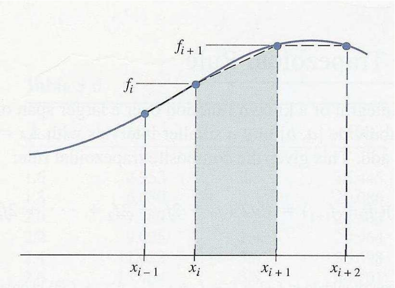


Figure: The trapezoidal rule.

- We divide the distance from $x = a$ to $x = b$ into **vertical strips** and add the areas of these strips.
- The strips are often made equal in widths but that is not always required.
- Approximate the curve with a sequence of straight lines.
- In effect, we slope the top of the strips to match with the curve as best we can.

- The gives us the ***trapezoidal rule***. Figure illustrates this.

The Trapezoidal Rule II

- It is clear that the area of the strip from x_i to x_{i+1} gives an approximation to the area under the curve:

$$\int_{x_i}^{x_{i+1}} f(x)dx \approx \frac{f_i + f_{i+1}}{2}(x_{i+1} - x_i)$$

- We will usually write $h = (x_{i+1} - x_i)$ for the width of the interval.
- Error term for the trapezoidal integration is

$$Error = -(1/12)h^3 f''(\xi) = O(h^3)$$

- What happens, if we are getting the integral of a known function over a larger span of x-values, say, from $x = a$ to $x = b$?
- We subdivide [a,b] into n smaller intervals with $\Delta x = h$, apply the rule to each subinterval, and add.

The Composite Trapezoidal Rule I

- This gives the **composite trapezoidal rule**;

$$\boxed{\int_a^b \approx \sum_{i=0}^{n-1} \frac{h}{2}(f_i + f_{i+1}) = \frac{h}{2}(f_0 + 2f_1 + 2f_2 + \ldots + 2f_{n-1} + f_n)}$$

- The error is not the local error $O(h^3)$ but the global error, the sum of n local errors;

$$Global\ error = \frac{-(b-a)}{12} h^2 f''(\xi) = O(h^2)$$

  where $nh = (b-a)$

- Use the trapezoidal rule to estimate the integral from $x = 1.8$ to $x = 3.4$ for $f(x) = e^x$.
- Applying the trapezoidal rule:

$$\begin{aligned}\int_{1.8}^{3.4} f(x)dx \quad & \approx \tfrac{0.2}{2}[f(1.8) + 2f(2.0) + 2f(2.2) + 2f(2.4) \\ & +2f(2.6) + 2f(2.8) + 2f(3.0) + 2f(3.2) \\ & +f(3.4)] = 23.9944\end{aligned}$$

The Composite Trapezoidal Rule II

Apply The trapezoidal rule to $f(x) = e^x$ in interval of 1.8 and 3.4. (**Example py-file:** mytrapezoid.py)

```
Mytrapezoid: Integration is       23.9941143322614288 for the function $e^x$ in interval of 1.800000 and 3.400000.
SciPy trapezoid: Integration is    23.9146518697568169 for the function $e^x$ in interval of 1.800000 and 3.400000.
```

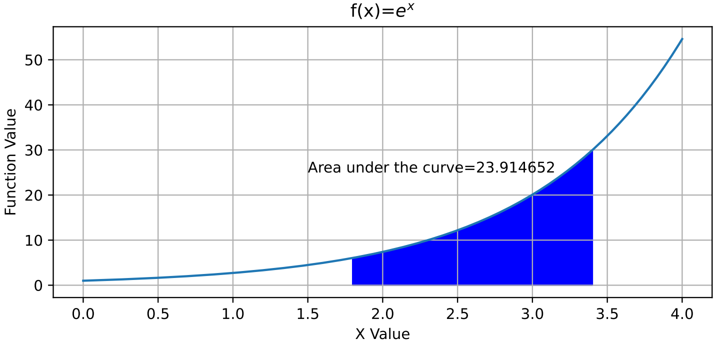


Figure: Integration for $f(x) = e^x$ by the trapezoidal rule.

Simple Pendulum VI

Apply The trapezoidal rule to Simple Pendulum to find the period. (**Example py-file:** simplependulum.py)

Degree	Calculated Period MyTrapezeoid	Calculated Period SciPyTrapezeoid	Small Angle Approximation
0.00	2.0070946166038111	2.0070946166038111	2.0070899231544930
5.00	2.0080503420735032	2.0080503420734983	2.0070899231544930
10.00	2.0109225326358318	2.0109225326358349	2.0070899231544930
15.00	2.0157263062261905	2.0157263062261901	2.0070899231544930
20.00	2.0224871133944586	2.0224871133944538	2.0070899231544930
25.00	2.0312411268457220	2.0312411268457256	2.0070899231544930
30.00	2.0420358043998514	2.0420358043998550	2.0070899231544930
35.00	2.0549306441689779	2.0549306441689810	2.0070899231544930
40.00	2.0699981579512587	2.0699981579512610	2.0070899231544930
45.00	2.0873250976325295	2.0873250976325277	2.0070899231544930
50.00	2.1070139804141048	2.1070139804141186	2.0070899231544930
55.00	2.1291849728107688	2.1291849728107644	2.0070899231544930
60.00	2.1539782117886461	2.1539782117886550	2.0070899231544930
65.00	2.1815566658380487	2.1815566658380630	2.0070899231544930
70.00	2.2121096716366528	2.2121096716366564	2.0070899231544930
75.00	2.2458573268222062	2.2458573268222080	2.0070899231544930
80.00	2.2830559815494613	2.2830559815494604	2.0070899231544930
85.00	2.3240051589550292	2.3240051589550310	2.0070899231544930
90.00	2.3690563597150089	2.3690563597150192	2.0070899231544930

Table: Integration for $\frac{1.0}{\sqrt{1.0-(sin(\theta_0/2)sin(\phi))^2}}$ by the trapezoidal rule.