1 Numerical Differentiation and Integration with a Computer

- 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.
- We continue to exploit the useful properties of polynomials to develop methods for a computer to do **integrations** and to find **derivatives**.
- When the function is explicitly known, we can emulate the methods of calculus.
- But doing so in getting derivatives requires the *subtraction* of quantities that are nearly equal and that runs into **round-off** error.
- However, integration involves only *addition*, so round-off is not problem.
- 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**.

1.1 Differentiation with a Computer

- The derivative of a function, $f(x)$ at $x = a$, is defined as

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

- This is called a ***forward-difference*** approximation.
- The limit could be approached from the opposite direction, giving a ***backward-difference*** approximation.
- **Forward-difference** approximation. A computer can calculate an approximation to the derivative, *if a very small value is used for* Δx.

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

- 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.
- Starting with $\Delta x = 0.05$ and halving Δx each time. Table 1 gives the results.
- We find that the errors of the approximation decrease as Δx is reduced until about $\Delta x = 0.05/128$.

- 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 0.05/128, 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.

Δx	Approximation	Error	Ratio of errors
0.05	4.05010	−0.11528	
0.05/2	4.10955	−0.05583	2.06
0.05/4	4.13795	−0.02743	2.04
0.05/8	4.15176	−0.01362	2.01
0.05/16	4.15863	−0.00675	2.02
0.05/32	4.16199	−0.00389	1.99
0.05/64	4.16382	−0.00156	2.18
0.05/128	**4.16504**	**−0.00034**	**4.67***
0.05/256	4.16504	−0.00034	
0.05/512	4.16504	−0.00034	
0.05/1024	4.16992	0.00454	
0.05/2048	4.17969	0.01430	

Table 1: Forward-difference approximations for $f(x) = e^x sin(x)$.

- **Backward-difference** approximation.

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

With MATLAB. **Analytical answer** to the function of Table 1.

```
format long;
syms x;
f='exp(x)*sin(x)';
df=diff(f,x)
exactvalue=subs(df,1.9,'x')
```

With MATLAB. **Numerical answer** to the function of Table 1.

- It is not by chance that the errors are about **halved each time**.
- 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

$$f'(x) = \frac{f(x + h) - f(x)}{h} - f''(\xi) * \frac{h}{2} \qquad (1)$$

```
%%%Forward-Difference%%%%%
disp('Step       Del        Numerical         Error       Error')
disp('                      Derivative                    Ratio')
disp('----- -------------- ------------ ------------ ----------')
x=1.9;
delini=1;
error(1)=1;
for i=1:30
 del=delini/2;
 xplus=x+del;
 f=exp(x).*sin(x);
 fplus=exp(xplus).*sin(xplus);
 num=fplus-f;
 deriv=num/del;
 error(i+1)=deriv-exactvalue;
 [D]=sprintf('%2d %1.15f %12.10f %12.10f %f ',i,del,deriv,error(i),
                                          error(i)/error(i+1));
 disp(D);
 delini=del;
end
```

- Which shows that the errors should be about proportional to h, precisely what Table 1 shows.
- If we repeat this but begin with the Taylor series for $f(x-h)$, it turns out that

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

- Where ζ is between x and $x - h$.
- The two error terms of Eqs. 1 and 2 are not identical though both are $O(h)$.
- If we add Eqs. 1 and 2, then divide by 2, we get the ***central-difference*** approximation to the derivative:

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} - f'''(\xi) * \frac{h^2}{6} \tag{3}$$

- 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**.

```
%%%Central-Difference%%%
disp('Step       Del       Numerical        Error      Error')
disp('                     Derivative                  Ratio')
disp('----- -------------- ------------ ------------ ----------')
x=1.9;
delini=0.1;
error(1)=1;
for i=1:20
 del=delini/2;
 xplus=x+del;
 xminus=x-del;
 fplus=exp(xplus).*sin(xplus);
 fminus=exp(xminus).*sin(xminus);
 num=fplus-fminus;
 deriv=num/(2*del);
 error(i+1)=deriv-exactvalue;
 [D]=sprintf('%2d %1.15f %12.10f %12.10f %f ',i,del,deriv,error(i),
                                              error(i)/error(i+1));
 disp(D);
 delini=del;
end
```

With MATLAB,

Table 2 illustrates this, showing that errors decrease about four fold when Δx is halved (as Eq. 3 predicts) and that a more accurate value is obtained.

Δx	Approximation	Error	Ratio of errors
0.05	4.15831	−0.00708	
0.05/2	4.16361	−0.00177	4.00
0.05/4	4.16496	−0.00042	4.21
0.05/8	4.16527	−0.00011	3.80
0.05/16	**4.16534**	**−0.00004**	**2.75**
0.05/32	4.16534	−0.00004	
0.05/64	4.16565	−0.00027	

Table 2: Central-difference approximations for $f(x) = e^x sin(x)$.

1.2 Numerical Integration - The Trapezoidal Rule

- 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.

```
>> syms x
>> int(exp(x)/log(x))
Warning: Explicit integral could not be found.
> In sym.int at 58
ans =  int(exp(x)/log(x),x)
```

- 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**

```
>> fx(i)=exp(x(i))/log(x(i))
>> x=linspace(2,3,10);
>> for i=1:10
fx(i)=exp(x(i))/log(x(i));
end
>> result=fx(1)+fx(10);
>> for i=2:9
result=result+2*fx(i);
end
>> result=(((3-2)/(10-1))/2)*result
%%%result=(0.1111/2)*result
result =   13.6904
```

- The definite integral is the area between the curve of $f(x)$ and the x-axis.

- That is the principle behind all numerical integration;

- 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.

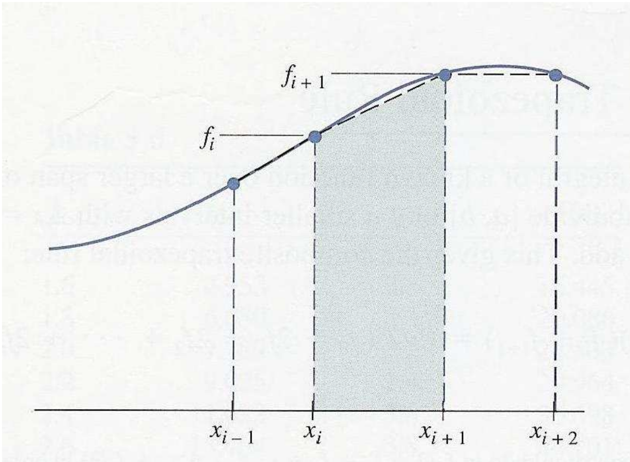

Figure 1: The trapezoidal rule.

1.2.1 The Trapezoidal Rule

- 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.
- We are approximating the curve with interpolating polynomials of degree-1.
- The gives us the ***trapezoidal rule***. Figure 1 illustrates this.
- 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 $x = b$?

1.2.2 The Composite Trapezoidal Rule

- We subdivide [a,b] into n smaller intervals with $\Delta x = h$, apply the rule to each subinterval, and add.
- This gives the **composite trapezoidal rule**;

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

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

$$Global\ error = (-1/12)h^3[f''(\xi_1) + f''(\xi_2) + \ldots + f''(\xi_n)]$$

- In this equation, each of the ξ_i is somewhere within each subinterval.
- If $f''(x)$ is continuous in [a, b], there is some point within [a,b] at which the sum of the $f''(\xi_i)$ is equal to $nf''(\xi)$, where ξ in [a, b].
- We then see that, because $nh = (b - a)$,

$$Global\ error = (-1/12)h^3nf''(\xi) = \frac{-(b-a)}{12}h^2f''(\xi) = O(h^2)$$

- **Example:** Given the values for x and $f(x)$ in Table3.

x	$f(x)$	x	$f(x)$
1.6	4.953	2.8	16.445
1.8	6.050	3.0	20.086
2.0	7.389	3.2	24.533
2.2	9.025	3.4	29.964
2.4	11.023	3.6	36.598
2.6	13.464	3.8	44.701

Table 3: Example for the trapezoidal rule.

- Use the trapezoidal rule to estimate the integral from $x = 1.8$ to $x = 3.4$.

- Applying the trapezoidal rule:

$$\begin{aligned}\int_{1.8}^{3.4} f(x)dx \quad &\approx \tfrac{0.2}{2}[6.050 + 2(7.389) + 2(9.025) + 2(11.023) \\ &+2(13.464) + 2(16.445) + 2(20.086) + 2(24.533) \\ &+29.964] = 23.9944\end{aligned}$$

- The data in Table 3 are for $f(x) = e^x$ and the true value is $e^{3.4} - e^{1.8} = 23.9144$.

- The trapezoidal rule value is off by 0.08; there are *three digits of accuracy.*

- How does this compare to the estimated error?

$$\begin{aligned}Error \quad &= -\tfrac{1}{12}h^3 n f''(\xi),\ 1.8 \le \xi \le 3.4 \\ &= -\tfrac{1}{12}(0.2)^3(8) * \left\{ \begin{array}{ll} e^{1.8} & (max) \\ e^{3.4} & (min) \end{array} \right\} = \left\{ \begin{array}{ll} -0.0323 & (max) \\ -0.1598 & (min) \end{array} \right\}\end{aligned}$$

Alternatively,

$$Error \quad = -\tfrac{1}{12}(0.2)^2(3.4 - 1.8) * \left\{ \begin{array}{ll} e^{1.8} & (max) \\ e^{3.4} & (min) \end{array} \right\} = \left\{ \begin{array}{ll} -0.0323 & (max) \\ -0.1598 & (min) \end{array} \right\}$$

- The actual error was -0.080. It is reasonable since the value is in the error bounds.

Thanks for attending and listening.