

In engineering, we often wish to estimate the derivative of a function based on a knowledge of its value at discrete points. For instance, we may expect the height of a surface  $y$  to be a function of its position  $x$ . For a variety of reasons, e.g. limited resources, technological limitations, we cannot get a continuous distribution  $y(x)$ , but we still may need to estimate the slope  $dy/dx$ . Let us say then that we are considering a domain  $x \in [a, b]$  and have  $N + 1$  uniformly distributed points  $x_i$ , where we can take samples. Our sampling points are thus

$$x_i = a + \frac{b-a}{N}(i-1), \quad i = 1, \dots, N+1.$$

At each point  $x = x_i$ , we take a sample of  $y$  and thus define these numbers as

$$y_i = y(x_i), \quad i = 1, \dots, N+1.$$

We can also say that the width of each increment is

$$\Delta x = \frac{b-a}{N}.$$

Now in general we will not know the actual functional form  $y(x)$ ; to assess our method of estimating derivatives, we will in fact take  $y(x)$  and use it to see how well our estimate achieves its goal of capturing the derivative.

Let us choose to build our estimate of the slope  $dy/dx = y'$  around the finite difference approximation

$$y'_i \sim \frac{y_{i+1} - y_{i-1}}{2\Delta x}, \quad i = 2, \dots, N.$$

This is called a *central difference*. We are estimating the derivative at  $x = x_i$  and using values of  $y$  to the left and to the right to do so. The width of this domain is  $2\Delta x$ . (In contrast, a *forward difference* would use the estimate  $y'_i \sim (y_{i+1} - y_i)/\Delta x$ . Notice that our estimate of  $y'_i$  is not valid for the end points,  $y'_1$  and  $y'_{N+1}$  because we have no samples for  $y_0$  or  $y_{N+2}$ . We need special one-sided formulæ for the end points, which can be shown to be

$$\begin{aligned} y'_1 &= \frac{-3y_1 + 4y_2 - y_3}{2\Delta x}, \\ y'_{N+1} &= \frac{y_{N-1} - 4y_N + 3y_{N+1}}{2\Delta x}. \end{aligned}$$

Then, our estimate for all of  $y'_i$  can be written in matrix form. To illustrate the form, we select  $N = 6$  and get

$$\underbrace{\begin{pmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \\ y'_5 \\ y'_6 \\ y'_7 \end{pmatrix}}_{\mathbf{y}'} = \underbrace{\begin{pmatrix} -\frac{3}{2\Delta x} & \frac{4}{2\Delta x} & -\frac{1}{2\Delta x} & 0 & 0 & 0 & 0 \\ -\frac{1}{2\Delta x} & 0 & \frac{1}{2\Delta x} & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{2\Delta x} & 0 & \frac{1}{2\Delta x} & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2\Delta x} & 0 & \frac{1}{2\Delta x} & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{2\Delta x} & 0 & \frac{1}{2\Delta x} & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{2\Delta x} & 0 & \frac{1}{2\Delta x} \\ 0 & 0 & 0 & 0 & \frac{1}{2\Delta x} & -\frac{4}{2\Delta x} & \frac{3}{2\Delta x} \end{pmatrix}}_{\mathbf{D}} \cdot \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{pmatrix}}_{\mathbf{y}}.$$

This is of the form

$$\mathbf{y}' = \mathbf{D} \cdot \mathbf{y}.$$

In index notation, we would say

$$y'_i = \sum_{j=1}^{N+1} D_{ij} y_j.$$

Remarkably, the derivatives are estimated by a matrix-vector multiplication. Even more remarkably, it can be shown that higher order derivatives can be estimated by repeated application of the so-called derivative matrix operator  $\mathbf{D}$ . For instance, the second derivative is estimated by

$$\mathbf{y}'' = \mathbf{D} \cdot \mathbf{D} \cdot \mathbf{y}.$$

In index notation, this is

$$y_i'' = \sum_{j=1}^{N+1} D_{ij} \sum_{k=1}^{N+1} D_{jk} y_k = \sum_{j=1}^{N+1} \sum_{k=1}^{N+1} D_{ij} D_{jk} y_k.$$

Note that  $\mathbf{D}$  can be reduced slightly to

$$\mathbf{D} = \frac{1}{2\Delta x} \begin{pmatrix} -3 & 4 & -1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & -4 & 3 \end{pmatrix}, \quad (\text{if } N = 6),$$

Now consider the function

$$y(x) = e^{-x^2}, \quad x \in [0, 1].$$

1. (70) Write, compile, and execute a **Fortran** program which
  - (a) samples  $y(x)$ ,  $x \in [0, 1]$  for user-defined  $N + 1$  uniformly spaced points  $x_i$ ,
  - (b) allows the user to specify  $N$  at run time, thus requiring use of the **allocate** command,
  - (c) builds the derivative matrix  $\mathbf{D}$  of dimension  $(N + 1) \times (N + 1)$ ,
  - (d) uses matrix-vector multiplication with nested do loops to estimate the derivatives  $y_i'$  and  $y_i''$  at each  $x_i$ ,
  - (e) uses matrix-vector multiplication via the **Fortran** intrinsic **matmul** to obtain the same estimates.
  - (f) for  $N = 6$  gives a continuous plot of the exact solution for  $y'(x)$  super-posed with a discrete plot of the estimate  $y_i'(x_i)$ .
  - (g) for  $N = 6$  gives a continuous plot of the exact solution for  $y''(x)$  super-posed with a discrete plot of the estimate  $y_i''(x_i)$ .
2. (30) Also consider the error in the estimates for  $y'$  at  $x_i = 1/2$ . Note that  $y'(1/2) = -e^{-1/4}$ . Give a log-log plot of how the magnitude of this error varies as a function of  $\Delta x$ . Try to span many orders of magnitude of  $\Delta x$  as possible, and see how low you can get the error. You may consider higher precisions than single precision.

Prepare your document with the  $\text{\LaTeX}$  text formatter and submit it as a **.pdf** file. Include at least one equation. Make all efforts to be *concise*: *For this homework, there is a three page maximum*. You only need to *briefly* summarize the problem statement. You should embed your **Fortran** source code within  $\text{\LaTeX}$ 's **verbatim** mode, e.g.

```
\begin{verbatim}
Fortran code here.
\end{verbatim}
```

As always, pay particular attention to neat, readable, elegant computer-generated plots.