$$\begin{pmatrix} \text{forward } 0 \\ \text{forward } 1 \\ \text{forward } 1 \\ \text{forward } 1 \\ \text{backward } 1 \\ \text{backward } 0 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} df(x_0)/dx \\ df(x_1)/dx \\ df(x_2)/dx \\ df(x_3)/dx \\ df(x_4)/dx \\ df(x_5)/dx \end{pmatrix}$$

$$\begin{pmatrix} -1.833 & 3.000 & -1.500 & 0.333 & 0 & 0 \\ 0.333 & -0.500 & 1.000 & -0.167 & 0 & 0 \\ 0 & -0.333 & -0.500 & 1.000 & -0.167 & 0 \\ 0 & 0 & -0.333 & -0.500 & 1.000 & -0.166 \\ 0 & 0 & 0.167 & -1.000 & 0.500 & 0.333 \\ 0 & 0 & -0.333 & 1.500 & -3.000 & 1.833 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} df(x_0)/dx \\ df(x_1)/dx \\ df(x_2)/dx \\ df(x_3)/dx \\ df(x_4)/dx \\ df(x_5)/dx \end{pmatrix}$$

In general, our goal is to seek the appropriate linear combination of functions $f(x_i + j\Delta x)$ that approximate the derivative of the desired order with the required local truncation error at a point x_i. In particular, each grid index j furnishes a label for each unique weight C_j needed for the cancellation of order terms up to the local truncation error seeked.

We obtain a starting point by considering the Taylor expanstion of a function at the $(i+j)$th grid point, $f(x_i + j\Delta x)$ where Δx is the grid spacing.

$$f(x_i + j\Delta x) = \sum_{n=0}^{d-1} j^n \frac{(\Delta x)^n}{n!} \frac{d^n f(x_i)}{dx^n} + j^d \frac{(\Delta x)^d}{d!} \frac{d^d f(x_i)}{dx^d} + \sum_{m=d+1}^{N-1} j^m \frac{(\Delta x)^m}{m!} \frac{d^m f(x_i)}{dx^m} + O(\Delta x^N)$$

There exists a unique linear combination of $j \in [j_{min}, j_{max}]$ such expansions that cause all order terms to vanish up to order $N-1$ which leaves only the desired derivative and a remainder term of order $O(\Delta x^N)$. That is, weighting the left-hand side of the above by coefficients $\{C_j\}$ and summing over a sufficient size stencil $\{j\}$, the above equation amounts to

$$\begin{aligned} \sum_{j=j_{min}}^{j_{max}} C_j f(x_i + j\Delta x) &= 0 + \sum_{j=j_{min}}^{j_{max}} C_j j^d \frac{(\Delta x)^d}{d!} \frac{d^d f(x_i)}{dx^d} + 0 + O(\Delta x^N) \\ \left(\sum_{j=j_{min}}^{j_{max}} C_j j^d \right) \frac{(\Delta x)^d}{d!} \frac{d^d f(x_i)}{dx^d} &= \sum_{j=j_{min}}^{j_{max}} C_j f(x_i + j\Delta x) + O(\Delta x^N) \\ \frac{(\Delta x)^d}{d!} \frac{d^d f(x_i)}{dx^d} &= \frac{\sum_{j=j_{min}}^{j_{max}} C_j f(x_i + j\Delta x)}{\left(\sum_{j=j_{min}}^{j_{max}} C_j j^d \right)} + \frac{1}{\left(\sum_{j=j_{min}}^{j_{max}} C_j j^d \right)} O(\Delta x^N) \end{aligned}$$

or,

$$\frac{(\Delta x)^d}{d!} \frac{d^d f(x_i)}{dx^d} = \sum_{j=j_{min}}^{j_{max}} c_j f(x_i + j\Delta x) + O(\Delta x^N)$$

where $c_j = C_j / (\sum_j C_j j^d)$ are unknown constants at this juncture. In order to see the order achievable, it is transparent to represent the truncation term above as having an order $N = p + d$, where p and d correspond to the effective order of the LTE for the desired derivative of order d. Thus, the above is equivalent to

$$\begin{aligned}\frac{d^d f(x_i)}{dx^d} &= \frac{d!}{(\Delta x)^d}\sum_{i=i_{min}}^{i_{max}} c_j f(x_i + j\Delta x) + \frac{d!}{(\Delta x)^d} O(\Delta x^{p+d}) \\ &= \frac{1}{(\Delta x)^d}\sum_{j=j_{min}}^{i_{max}} w_j f(x_i + j\Delta x) + O(\Delta x^p)\end{aligned}$$

In the final step we have chosen to represent the product $w_j \equiv c_j d!$, as it is the set $\{w_j\}$ that are commonly reported in literature and known as ¡i¿difference coefficients¡/i¿. In order to cause the order terms to cancel up to and including order $N-1 = p+d-1$, we expand the Taylor series up to this order to obtain constraining equations for the weights.

$$\begin{aligned}\frac{d^d f(x_i)}{dx^d} &= \frac{1}{(\Delta x)^d}\sum_{j=j_{min}}^{i_{max}} w_j f(x_i + j\Delta x) + O(\Delta x^p) \\ &= \frac{1}{(\Delta x)^d}\sum_{j=j_{min}}^{i_{max}} w_j \sum_{m=0}^{p+d-1} j^m \frac{(\Delta x)^m}{m!} + O(\Delta x^p) \\ \frac{d^d f(x_i)}{dx^d} &= \frac{1}{(\Delta x)^d}\sum_{m=0}^{p+d-1}\left(\sum_{j=j_{min}}^{i_{max}} w_j j^m\right)\frac{(\Delta x)^m}{m!} + O(\Delta x^p)\end{aligned}$$

Thus, what is required for all $p+d$ order terms to vanish is that the prefactor

$$\boxed{\text{order conditions}} : \sum_{j=j_{min}}^{j_{max}} w_j j^m = \begin{cases} 1 & m = d \\ 0 & 0 \le m \le p+d-1, m \ne d \end{cases}, \quad m = 0, 1, \ldots, p+d-1$$

The formulation presents $j_{max} - j_{min} + 1$ unknown weights w_j and $p+d$ equations. In general, there is no unique solution, but a family of solutions where free parameters may be chosen or otherwise optimized. If we choose to concentrate only on ¡i¿explicit¡/i¿ differencing schemes that have unique solutions, the we require

$$\boxed{\text{Stencil size}} : j_{max} - j_{min} + 1 = p+d, \qquad \text{uniqueness condition for explicit schemes}$$

As an example, we can prove the above scheme from the specific example just above ($p = 2$, $d = 1$, i.e. stencil size $= p+d = 3$, $0 \le m < p+d$) is consistent with this general form by choosing $j_{min} = 0, j_{max} = 2$ and weights $w_0 = -3/2, w_1 = 4/2, w_2 = -1/2$, by writing the order constraints above, where the right-hand side is nonzero only for $m = d = 1$:

$$m = 0 : \sum_{j=0}^{2} w_j j^0 = w_0(0^0) + w_1(1^0) + w_2(2^0) = -\tfrac{3}{2}(1) + \tfrac{4}{2}(1) - \tfrac{1}{2}(1) = 0$$

$$m = 1 : \sum_{j=0}^{2} w_j j^1 = w_0(0^1) + w_1(1^1) + w_2(2^1) = -\tfrac{3}{2}(0) + \tfrac{4}{2}(1) - \tfrac{1}{2}(2) = 1$$

$$m = 2 : \sum_{j=0}^{2} w_j j^2 = w_0(0^2) + w_1(1^2) + w_2(2^2) = -\tfrac{3}{2}(0) + \tfrac{4}{2}(1) - \tfrac{1}{2}(4) = 0$$

The constraints can be cast conveniently in matrix form:

$$\begin{pmatrix} 0^0 & 1^0 & 2^0 \\ 0^1 & 1^1 & 2^1 \\ 0^2 & 1^2 & 2^2 \end{pmatrix} \begin{pmatrix} w_0 \\ w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

or, in general

$$\underbrace{\begin{pmatrix} j_{min}^0 & (j_{min}+1)^0 & \cdots & \cdots & (j_{max}-1)^0 & j_{max}^0 \\ j_{min}^1 & (j_{min}+1)^1 & \cdots & \cdots & (j_{max}-1)^1 & j_{max}^1 \\ j_{min}^2 & (j_{min}+1)^2 & \cdots & \cdots & (j_{max}-1)^2 & j_{max}^2 \\ \vdots & \vdots & & & \vdots & \vdots \\ \vdots & \vdots & & & \vdots & \vdots \\ \vdots & \vdots & & & \vdots & \vdots \\ \vdots & \vdots & & & \vdots & \vdots \\ j_{min}^{p+d-2} & (j_{min}+1)^{p+d-2} & \cdots & \cdots & (j_{max}-1)^{p+d-2} & j_{max}^{p+d-2} \\ j_{min}^{p+d-1} & (j_{min}+1)^{p+d-1} & \cdots & \cdots & (j_{max}-1)^{p+d-1} & j_{max}^{p+d-1} \end{pmatrix}}_{\underline{\underline{J}}_{N\times N}} \underbrace{\begin{pmatrix} w_{j_{min}} \\ w_{j_{min}+1} \\ w_{j_{min}+2} \\ \vdots \\ \vdots \\ \vdots \\ w_{j_{max}-2} \\ w_{j_{max}-1} \\ w_{j_{max}} \end{pmatrix}}_{\underline{w}_{N\times 1}} = \underbrace{\begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \\ 0 \\ 0 \end{pmatrix}}_{\underline{\delta}_{N\times 1,md}}$$

so the following can be written:

$$\boxed{\underline{\underline{J}}_{N\times N}\underline{w}_{N\times 1} = \underline{\delta}_{N\times 1}^{md}}, \qquad \text{where } N = p+d$$

It is noted that the matrix $\underline{\underline{J}}$ is a Vandermonde matrix involving grid indices. While perhaps not anticipated, it is nontheless not surprising to see result. The paradigmatic context for a Vandermonde matrix arises for least squares fitting or Lagrange interpolating polynomials. It is well known that differentiating interpolants of a function produces finite difference schemes of varying degrees of accuracy. What is unclear in that formulation is how the stencil size (how many points are interpolated exactly) affects the order of the local truncation error. The development above answers this question, the larger the stencil size (the more points that are interpolated exactly), the higher order the truncation error will be. What is more, it informs a stencil of size $p+d$ produces a truncation error of $O(\Delta x^p)$. Even if this knowledge were to be gleaned a priori it remains impractical and labor intensive to assemble interpolating polynomials with the required stencil size for each case and differentiate as many times as needed. The above prescription permitted us to clearly understand the size of the stencil needed for a given derivative order d for a desired truncation order p, and allows for efficient calculation of the resulting difference coefficients w_i, an implementation that can be automated, so we have obtained a result with clear advantage and in doing so have discovered more telling information than would be seen by toying with Lagrange interpolating polynomials. Though admittedly, trends would have been observable, satisfying information would not be obtainable.

The delta-function vector on the right-hand side is defined as:

$$\underline{\delta}_{N\times 1}^{md} = (\underbrace{0}_{m=0}, \underbrace{0}_{m=1}, \underbrace{0}_{m=2}, \ldots, \underbrace{1}_{m=d}, \ldots, \underbrace{0}_{m=N-3}, \underbrace{0}_{m=N-2}, \underbrace{0}_{m=N-1})^T$$

The solution to this matrix equation gives the weights $\{w_j\}$, $j = j_{min}, j_{min}+1, \ldots, j_{max}$ for an explicit finite difference scheme estimating a derivative of order d with a local truncation error p. Its stencil size is $p+d$, so that higher accuracy is seento come at the cost of expanding the stencil sign. For rapidly varying functions, it is understood that convergence cannot be expected for sufficiently coarse grids (however, if the grid is refined too significantly, then we must also be wary of the competition of the error drop off on the LTE as compared to the numerical error buildup. When the grid becomes too fine the estimation using a given scheme can become far worse due to the accumulation of an error term that is on the order of the machine error but which scales with an inverse power coresponding to the derivative itself).

Note, in implementing this routine, it is of course discouraged to actually pursue finding the inverse of the $\underline{\underline{J}}$ matrix, but rather to solve through an efficient matrix routine (e.g. factorization methods). In Python, the NumPy package numpy.linalg.solve can be used to efficiently solve the problem.

We calculate the order as above:

$$\text{order}_i = \log_2 \frac{||E_{\Delta x}||_2}{||E_{\Delta x/2}}$$

where $E_{\Delta x} = f_{exact} - f_{\Delta x}$

The truncation error is bounded by constants $C_{\Delta x}, C_{\Delta x/2}, M_{\Delta x}, M_{\Delta x/2}$, which depend on the mesh spacing, where C limits the size on the LTE and M is associated with the aforementioned numerical source of error (and thus contains terms on the order of machine error $\sim 10^{-16}$. Writing the order of the LTE as p, and the derivative order of interest as n (also the negative order with which the machine error scales with). Thus, the above is written as

$$\text{order}_i = \log_2 \frac{C_{\Delta x}(\Delta x)^p + M_{\Delta x}(\Delta x)^{-n}}{C_{\Delta x/2}(\Delta x/2)^p + M_{\Delta x/2}(\Delta x/2)^{-n}}$$

Formally, we can choose to retain the constants as distinct. The result does not change if we choose the constants $C_{\Delta x} = C_{\Delta x/2} \sim 1$ and $M_{\Delta x} = M_{\Delta x/2} \sim \varepsilon$ equal to unity, this also permits a cleaner equation work which is easier to follow and gets to the point more expediently.

$$\begin{aligned}
\text{order}_i &= \log_2 \frac{(\Delta x)^p + \varepsilon(\Delta x)^{-n}}{(\Delta x/2)^p + \varepsilon(\Delta x/2)^{-n}} \\
&= \log_2 \left\{ 2^n \frac{\varepsilon^{-1}(\Delta x)^{n+p} + 1}{\varepsilon^{-1}(2\Delta x)^{n+p} + 1} \right\} \\
&= \log_2 2^n + \log_2 \left\{ \frac{\varepsilon^{-1}(\Delta x)^{n+p} + 1}{(\varepsilon^{-1}2\Delta x)^{n+p} + 1} \right\} \\
\text{order}_i &= n + \log_2 \left\{ \frac{\varepsilon^{-1}(\Delta x)^{n+p} + 1}{\varepsilon^{-1}(2\Delta x)^{n+p} + 1} \right\}
\end{aligned}$$

We consider the case for progressively smaller Δx, and consider a regime where the numerical error accumulation, $\varepsilon(\Delta x)^{-n}$ is sufficiently larger than the LTE term $(\Delta x)^p$, i.e $\varepsilon(\Delta x)^{-n} \gg (\Delta x)^p \Rightarrow \varepsilon^{-1}(\Delta x)^{n+p} \ll 1$. Thus, the term in the denominator $\varepsilon^{-1}(2\Delta x)^{n+p} \ll 1$ permits an expansion in this small parameter.

$$\begin{aligned}
\text{order}_i &= n + \log_2 \left\{ (1 + \varepsilon^{-1}(\Delta x)^{n+p})(1 - \varepsilon^{-1}(2\Delta x)^{n+p} + \varepsilon^{-2}(2\Delta x)^{2(n+p)} + \ldots) \right\} \\
&= n + \log_2(1 + \varepsilon^{-1}(\Delta x)^{n+p}) + \log_2(1 - \varepsilon^{-1}(2\Delta x)^{n+p} + \varepsilon^{-2}(2\Delta x)^{2(n+p)} + \ldots) \\
\text{order}_i &= n + \log_2(1 + \varepsilon^{-1}(\Delta x)^{n+p}) + \log_2(1 - \varepsilon^{-1}(2\Delta x)^{n+p}), \qquad \text{since } \varepsilon^{-1}(2\Delta x)^{2(n+p)} \ll \varepsilon^{-1}(2\Delta x)^{n+p}
\end{aligned}$$

Expanding the logarithms

$$\text{order}_i = n + (\varepsilon^{-1}(\Delta x)^{n+p}) - \varepsilon^{-2}(\Delta x)^{2(n+p)}) + \ldots) + (-\varepsilon^{-1}(2\Delta x)^{n+p} - \varepsilon^{-2}(2\Delta x)^{2(n+p)} + \ldots)$$

Hence, in the limit of $\Delta x \to 0$, we recover a *positive* slope which corresponds to the order of the derivative n

$$\boxed{\lim_{\Delta x \to 0} \text{order}_i = n}, \quad \text{when } \varepsilon(\Delta x)^{-n} \gg (\Delta x)^p$$

$$\frac{\delta^d f}{\delta x^d}(x) = \frac{d!}{\Delta x^d} \sum_{i=i_{min}}^{i_{max}} c_i f(x + i\Delta x) + \frac{d!}{\Delta x^d} O(\Delta x^N)$$