Hermite Interpolation Between 2 Points

Problem Setting

In general, the term ”Hermite interpolation” refers to interpolation by means of a polynomial that passes through a given number of sample points (x_i, y_i) and also satisfies constraints on some number of derivatives $y'_i, y''_i, \ldots$ at these sample points. Here, we consider the problem of finding a polynomial that goes through the two points $(x_0 = 0, y_0)$ and $(x_1 = 1, y_1)$. In addition to prescribe the function values y_0, y_1, we also prescribe values for some number of derivatives $y'_0, y'_1; y''_0, y''_1; etc.$. Our particular choice of the x coordinates has been made to keep the formulas simple. However, if we want to have arbitrary x-coordinates for the endpoints, say x_{min}, x_{max}, we may simply transform the input value for the polynomial by $\tilde{x} = (x - x_{min})/(x_{max} - x_{min})$. Our new variable $\tilde{x}$ will then pass through the range $0, \ldots, 1$ when the original x passes through $x_{min}, \ldots, x_{max}$. The number of derivatives that we want to control dictates the order of the polynomial that we have to use. In order to be able to prescribe values for M derivatives, we need a polynomial of order $N = 2M + 1$.

Derivation for the $7th$ Order Case

To illustrate the procedure to compute the polynomial coefficients, we consider - as example - the case where we control $M = 3$ derivatives. This calls for a $7th$ order polynomial. In the following derivation, the framed equations are those that we actually need for the implementation. Our interpolating polynomial and its first 3 derivatives have the general form:

$$\begin{aligned}
y &= a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0 \\
y' &= 7a_7x^6 + 6a_6x^5 + 5a_5x^4 + 4a_4x^3 + 3a_3x^2 + 2a_1x + a_1 \\
y'' &= 42a_7x^5 + 30a_6x^4 + 20a_5x^3 + 12a_4x^2 + 6a_3x + 2a_1 \\
y''' &= 210a_7x^4 + 120a_6x^3 + 60a_5x^2 + 24a_4x + 6a_3
\end{aligned} \tag{1}$$

To satisfy our constraints at the left endpoint $x_0 = 0$, we put in $x = 0$ on the right hand sides and y_0, y'_0, y''_0, y'''_0 on the left hand sides, and we immediately obtain a_0, a_1, a_2, a_3:

$$\boxed{y_0 = a_0, \quad y'_0 = a_1, \quad y''_0 = 2a_2, \quad y'''_0 = 6a_3} \tag{2}$$

...for the actual implementation, you need to solve them for the $a-$coefficients (this is left for the reader as exercise ;-). To satisfy our constraints at the right endpoint $x_1 = 1$, we put in $x = 1$ on the right hand sides and y_1, y'_1, y''_1, y'''_1 on the left hand sides - we obtain 4 equations for the remaining 4 unknowns a_4, a_5, a_6, a_7:

$$\begin{aligned}
y_1 &= a_7 + a_6 + a_5 + a_4 + a_3 + a_2 + a_1 + a_0 \\
y'_1 &= 7a_7 + 6a_6 + 5a_5 + 4a_4 + 3a_3 + 2a_2 + a_1 \\
y''_1 &= 42a_7 + 30a_6 + 20a_5 + 12a_4 + 6a_3 + 2a_2 \\
y'''_1 &= 210a_7 + 120a_6 + 60a_5 + 24a_4 + 6a_3
\end{aligned} \tag{3}$$

bringing the already known a_0, a_1, a_2, a_3 to the left side:

$$\begin{aligned}
y_1 - a_3 - a_2 - a_1 - a_0 &= a_7 + a_6 + a_5 + a_4 \\
y'_1 - 3a_3 - 2a_2 - a_1 &= 7a_7 + 6a_6 + 5a_5 + 4a_4 \\
y''_1 - 6a_3 - 2a_2 &= 42a_7 + 30a_6 + 20a_5 + 12a_4 \\
y'''_1 - 6a_3 &= 210a_7 + 120a_6 + 60a_5 + 24a_4
\end{aligned} \tag{4}$$

for convenience, we define constants k_0, k_1, k_2, k_3 for the 4 left hand sides of the equations:

$$\boxed{\begin{aligned} k_0 &= y_1 - a_3 - a_2 - a_1 - a_0 \\ k_1 &= y_1' - 3a_3 - y_0'' - a_1 \\ k_2 &= y_1'' - y_0''' - y_0'' \\ k_3 &= y_1''' - y_0''' \end{aligned}} \tag{5}$$

where we have also used that $6a_3 = y_0'''$ and $2a_2 = y_0''$. Our system of equations now becomes:

$$\begin{aligned} k_0 &= a_7 + a_6 + a_5 + a_4 \\ k_1 &= 7a_7 + 6a_6 + 5a_5 + 4a_4 \\ k_2 &= 42a_7 + 30a_6 + 20a_5 + 12a_4 \\ k_3 &= 210a_7 + 120a_6 + 60a_5 + 24a_4 \end{aligned} \tag{6}$$

finally, solving this system for the remaining 4 unknowns a_4, a_5, a_6, a_7 gives:

$$\boxed{\begin{aligned} a_4 &= \frac{-k_3 + 15k_2 - 90k_1 + 210k_0}{6} \\ a_5 &= -\frac{-k_3 + 14k_2 - 78k_1 + 168k_0}{2} \\ a_6 &= \frac{-k_3 + 13k_2 - 68k_1 + 140k_0}{2} \\ a_7 &= -\frac{-k_3 + 12k_2 - 60k_1 + 120k_0}{6} \end{aligned}} \tag{7}$$

Results for Some Other Cases

Having seen the derivation for the $7th$ order case, it shall suffice for other cases to just give the results. Here we go:

1st order case

$$a_0 = y_0, \quad a_1 = y_1 - y_0 \tag{8}$$

3rd Order Case

$$a_0 = y_0, \quad a_1 = y_0' \tag{9}$$

$$k_0 = y_1 - a_1 - a_0, \quad k_1 = y_1' - a_1 \tag{10}$$

$$a_2 = 3k_0 - k_1, \quad a_3 = k_1 - 2k_0 \tag{11}$$

5th Order Case

$$a_0 = y_0, \quad a_1 = y_0', \quad a_2 = \frac{y_0''}{2} \tag{12}$$

$$k_0 = y_1 - a_2 - a_1 - a_0, \quad k_1 = y_1' - y_0'' - a_1, \quad k_2 = y_1'' - y_0'' \tag{13}$$

$$a_3 = \frac{k_2 - 8k_1 + 20k_0}{2}, \quad a_4 = -k_2 + 7k_1 - 15k_0, \quad a_5 = \frac{k_2 - 6k_1 + 12k_0}{2} \tag{14}$$

The General Case

For the general case, where we control M derivatives by using a polynomial of order $N = 2M+1$, a general pattern emerges. The polynomial coefficients a_n for powers up to M can be computed straightforwardly via:

$$a_n = \frac{y_0^{(n)}}{n!}, \quad n = 0, \ldots, M \tag{15}$$

where $y^{(n)}$ denotes the $n-th$ derivative of y, the $0-th$ derivative is the function itself. Now, we establish a vector $\mathbf{k} = (k_0, \ldots, k_M)$ of $M+1$ k-values, whose element k_n is given by:

$$k_n = y_1^{(n)} - \sum_{i=n}^{M} \alpha_{n,i} a_i \quad n = 0, \ldots, M \tag{16}$$

where

$$\alpha_{n,i} = \prod_{m=i-n+1}^{i} m \tag{17}$$

Note that for this product to work in general, we must make use the definition of the empty product: $\prod_{i=n}^{N} a_i = 1, \quad \text{for } N < n$, i.e. when the end-index is lower than the start-index. We also establish a $(M+1) \times (M+1)$ matrix $\mathbf{A}$, whose element $A_{i,j}$ is given by:

$$A_{i,j} = \prod_{m=M+j-i+2}^{M+j} m \tag{18}$$

Now, we collect our remaining unknowns $a_{M+1}, \ldots, a_N$ into the vector $\mathbf{a}$, such that: $\mathbf{a} = (a_{M+1}, \ldots, a_M)$. The system of equations for the remaining unknowns may now be expressed as the matrix equation:

$$\mathbf{k} = \mathbf{A}\mathbf{a} \tag{19}$$

Numerically solving this equation for $\mathbf{a}$ (for example by Gaussian elimination) yields the remaining polynomial coefficients $a_{M+1}, \ldots, a_N$. [Question to self: can a simpler solution be derived that avoids the need for the general linear system solver?]