SCIENTIFIC COMPUTATION COMP. REVIEW

Topic: Stencils (Function values, Differentiation, Integration, and all that.)

Nick Battista

Date Created: 5/21/2014
Date Modified: 9/2/2014

1 BACKGROUND

As children we first learn of stencils as those amazingly simplistic things in art class or laying around the house that help us become *better* artists. Sometimes they allow us to take a complicated structure (animal, flower, cloud, ceiling in the Sistine Chapel) and break it down into basic pieces so we can more masterfully draw it with smooth, confident lines (eh, line segments more precisely).

In computational mathematics, we have similar tools that help us more accurately approximate desired quantities (function values, derivatives, integrals, etc). Obviously it is not a tangible device, but rather a geometric collection of nodal points that relate to a point of interest (sometimes called the *reference point*). The overarching idea is that we can use nodal values surrounding the reference point to approximate a function value, derivative value, or integral over an integration bound containing the reference point and stencil's nodal values.

1.1 BUILDING UP THE IDEA OF A STENCIL

Let's build up the idea some more. Suppose we're given the following data $(-1, f(-1))$ and $(1, f(1))$ and we wish to approximate a derivative at $x = 0$. Without getting into the nitty-gritty details (we'll save that fun for later!), we can best approximate $f'(0)$ with that data as,

$$f'(0) \approx \frac{f(1) - f(-1)}{2}.$$

However, it doesn't take a research mathematician (or a beginning Calculus student) for that matter to know that this approximation doesn't seem quite kosher. Suppose instead we had the following data, $(-\frac{1}{4}, f(-\frac{1}{4}))$ and $(\frac{1}{4}, f(\frac{1}{4}))$, and still want to approximate $f'(0)$. The best approximation we can muster with that data is now,

$$f'(0) \approx \frac{f(\frac{1}{4}) - f(-\frac{1}{4})}{2}.$$

You might be able to guess where we're going now (...no, not to the computer lab just yet). Although the above approximation is better than the first approximation we made, you might be wrangling the idea that if we have some parameter $h > 0$ and data $(-h, f(-h))$ and $(h, f(h))$, and have the approximation,

$$f'(0) \approx \frac{f(h) - f(-h)}{2}.$$

Now if h is *small* (in the correct mathematically hand-wavy ideal), we expect to get desirable accuracy for the derivative at 0. Don't worry, we'll discuss how to intelligently talk about errors associated with these stencils later. And oh, it will be fun since we have multiple ways to compute errors. (Or are they one and the same?).

1.2 General Ideas

For now we can slightly generalize the idea of a stencil. Suppose we want to approximate a function value, derivative, higher order derivative, or integral around a point x_r.

$$\left.\begin{array}{c} f(x_r), \\ f'(x_r), \\ f''(x_r), \\ \vdots \\ f^{(k)}(x_r), \text{ or} \\ \displaystyle\int_{x_r-h}^{x_r+h} f(x)dx \end{array}\right\} = \sum_{m=-N}^{N} c_m f(x_r + mh).$$

Our task (as it usually is in computation) is to find the coefficients, $\{c_m\}$, to complete the stencil in such a way that it minimizes truncation error. To make our lives easier, we're going to mandate that the stencil nodal points are on a uniform grid, i.e., have a uniform spacing h between points. Moreover, we can break any interval $[a, b]$ into subintervals of uniform length.

Suppose we're given the interval $[-h, h]$ and we wish to integrate a function $f(x)$ from $[-h, h]$ using a 2−pt stencil, i.e.,

$$\int_{-h}^{h} f(x)dx \approx c_0 f(-h) + c_1 f(h).$$

We can find the values of those coefficients to get an approximation scheme. However, what if we divided the integral into two subintervals,

$$\int_{-h}^{h} f(x)dx = \int_{-h}^{0} f(x)dx + \int_{0}^{h} f(x)dx,$$

could use the same stencil for each subinterval integral approximation? Yes, a million times yes. We would then have for each sub-integral component,

$$\begin{aligned}
&\int_{-h}^{0} f(x)dx \approx c_0 f(-h) + c_1 f(0),\\
&\int_{-h}^{0} f(x)dx \approx c_0 f(0) + c_1 f(h), \quad \text{and hence}\\
&\int_{-h}^{0} f(x)dx = \int_{-h}^{0} f(x)dx + \int_{0}^{h} f(x)dx \approx c_0 f(-h) + c_1 f(0) + c_0 f(0) + c_1 f(h).
\end{aligned}$$

To be clear the values of c_0 and c_1 are the same in the computation above as they were before. Furthermore, you probably recognize this idea already from Calculus - those ~~pesky~~ fun Riemann sums that needed to be computed by hand.

Let's boogey!

1.3 Function Value Stencils

As you might recall from interpolation theory, there is a unique interpolating polynomial that goes through a given set of data. Hence for these stencils, let's use monomial interpolation to find the necessary coefficients.

Recall for monomial interpolation, we're given data, $\{(x_i, y_i)\}_{i=0}^{N}$, and find its interpolating polynomial, $p(x) = \sum\limits_{j=0}^{N} a_j x^j$, by solving the following linear system,

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^N \\ 1 & x_1 & x_1^2 & \cdots & x_1^N \\ 1 & x_2 & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ 1 & x_N & x_N^2 & \cdots & x_N^2 \end{bmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{N-1} \\ a_N \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \\ y_N \end{pmatrix}.$$

Now we will derive how to use stencils to approximate function values. It will feel like we are doing the same thing as interpolation; however, we will just be using interpolation ideas to derive a way to approximate function values using other *known* function values. This might seem a bit convoluted. Let's talk about it.

Say we've sampled some function, $f(x)$, between $[0,1.0]$, with $x_0 = 0.0$, $x_1 = 0.1$, $x_2 = 0.2,\ldots,$ and $x_{10} = 1.0$, so we can call the grid spacing, h, and $h = 0.1$ in this example. Rather than compute an 11^{th} order interpolating polynomial, say we only want to use less than the 11 points to approximate values we don't know, i.e., those that aren't $\{x_k\}_{k=0}^{11}$ above. There are a few reasons for this and we will discuss them later.

For now, let's say we wish to find the function value associated with some point, $x_R = 0.15$. Well in the manner discussed previously, we could approximate the value of $f(x_R)$ using the surrounding function values, i.e., if we arbitrarily choose to use the two closest function values on each side of x_R,

$$\tilde{f}(x_R) = c_0 f(x_3) + c_1 f(x_4) + c_2 f(x_5) + c_3 f(x_6).$$

Why would we only use 2 function values on either side? Well, we'll chat about this more once we begin our error analysis here. For now, let's just use 4 total points, for what we will call a *4-pt. symmetric stencil.*

Right now just think, we want to be lazy and use a few points that will lead to a smaller linear system. *Spoiler:* depending on the problem at hand, the resulting linear system obtained for stencil coefficients *can* get ill-conditioned pretty quickly as the number of stencil points increases.

We will now derive what are called the *moment equations*, using a method that should feel pretty comfortable, namely *the method of undetermined coefficients*. In a nut shell, if we have n stencil points, we will derive a system of n equations, in which we can solve for the stencil coefficients, $\{c_j\}_{j=0}^{n-1}$. For our example, we have $n = 4$, and hence will now derive a system of 4 equations to solve for the 4 unknowns, $\{c_0., c_1, c_2, c_3\}$.

The main mantra we will follow for implementing the *method of undetermined coefficients*, is that we wish to choose the stencil coefficients, which are also called *weights*, to give the exact value of function for the first n (in our case, $n = 4$) polynomials exactly. With this mantra, we obtain the following equations,

$$\begin{aligned}
c_0 \cdot 1 + c_1 \cdot 1 + c_2 \cdot 1 + c_3 \cdot 1 &= 1 \\
c_0 \cdot x_0 + c_1 \cdot x_1 + c_2 \cdot x_2 + c_3 \cdot x_3 &= (x - x_R)\Big|_{x=x_R} \\
c_0 \cdot x_0^2 + c_1 \cdot x_1^2 + c_2 \cdot x_2^2 + c_3 \cdot x_3^2 &= (x - x_R)^2\Big|_{x=x_R} \\
c_0 \cdot x_0^3 + c_1 \cdot x_1^3 + c_2 \cdot x_2^3 + c_3 \cdot x_3^3 &= (x - x_R)^3\Big|_{x=x_R}
\end{aligned}$$

We could then solve the above equations for the coefficients. Note that we will get a nice, linear system.

However, we will now do a change of variables to make the system look a little easier to comprehend. This will become apparent once we start creating stencils for differentiation and integration. To do this we begin by making the transformation that we want $\tilde{x}_R = 0$, that is, we want to map $x_R \to 0$, and hence use $\tilde{x_j} = x_j - x_R$ for all $\{x_j\}_{j=0}^{4}$. Hence we now have

$$\begin{aligned}
\tilde{x}_0 &= -\frac{3h}{2} \quad (x_0 = 0.0)\\
\tilde{x}_1 &= -\frac{h}{2} \quad (x_0 = 0.1)\\
\tilde{x}_2 &= \frac{h}{2} \quad (x_0 = 0.2)\\
\tilde{x}_3 &= \frac{3h}{2} \quad (x_0 = 0.3)
\end{aligned}$$

and obtain the system,

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ -\frac{3h}{2} & -\frac{h}{2} & \frac{h}{2} & \frac{3h}{2} \\ (-\frac{3h}{2})^2 & (-\frac{h}{2})^2 & (\frac{h}{2})^2 & (\frac{3h}{2})^2 \\ (-\frac{3h}{2})^3 & (-\frac{h}{2})^3 & (\frac{h}{2})^3 & (\frac{3h}{2})^3 \end{bmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Solving the above system, will give coefficients $\{c_0, c_1, c_2, c_3\}$ and hence an approximation for $f(x_R)$. Now note for any $x_R + jh \in [0.1, 0.9]$, where $j \in \{0,1,2,3,4,5,6,7\}$, we can use the 4-pt stencil above to approximate the function value, i.e.,

$$\tilde{f}(x_R + jh) = \tilde{f}(0.15 + jh) = c_{0+j} f(x_{0+j}) + c_{1+j} f(x_{1+j}) + c_{2+j} f(x_{2+j}) + c_{3+j} f(x_{3+j}).$$

Note that we *cannot* approximate $f(0.05)$ or $f(0.95)$ using this stencil because we can longer use a symmetric stencil if we wish to use a 4-pt stencil.

If you are wondering, the values of the coefficients are

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} -\frac{1}{16} \\ \frac{9}{16} \\ \frac{9}{16} \\ -\frac{1}{16} \end{pmatrix}.$$

1.3.1 Error Analysis: Function Value Stencils

We will now discuss how to perform error analysis for this type of stencil. Recall we have done the following approximation,

$$\begin{aligned}
\tilde{f}(0.15) &= -\frac{1}{16}f(0) + \frac{9}{16}f(0.1) + \frac{9}{16}f(0.2) - \frac{1}{16}f(0.3) \\
&= -\frac{1}{16}f\Big(0.15 - \frac{3h}{2}\Big) + \frac{9}{16}f\Big(0.15 - \frac{h}{2}\Big) + \frac{9}{16}f\Big(0.15 + \frac{h}{2}\Big) - \frac{1}{16}f\Big(0.15 + \frac{3h}{2}\Big) \\
&= -\frac{1}{16}f\Big(x_R - \frac{3h}{2}\Big) + \frac{9}{16}f\Big(x_R - \frac{h}{2}\Big) + \frac{9}{16}f\Big(x_R + \frac{h}{2}\Big) - \frac{1}{16}f\Big(x_R + \frac{3h}{2}\Big)
\end{aligned}$$

You can probably see where this is going...it's about to become a Taylor Series party! We will now expand the RHS of the above equation in Taylor Series centered at $x_R = 0.15$.

$$\begin{aligned}
\tilde{f}(x_R) = &-\frac{1}{16}\Big[f(x_R) - \frac{3h}{2}f'(x_R) + \frac{1}{2!}\Big(-\frac{3h}{2}\Big)^2 f''(x_R) + \frac{1}{3!}\Big(-\frac{3h}{2}\Big)^3 f'''(x_R) + \frac{1}{4!}\Big(-\frac{3h}{2}\Big)^4 f^{(iv)}(x_R) + \ldots\Big] \\
&+ \frac{9}{16}\Big[f(x_R) - \frac{h}{2}f'(x_R) + \frac{1}{2!}\Big(-\frac{h}{2}\Big)^2 f''(x_R) + \frac{1}{3!}\Big(-\frac{h}{2}\Big)^3 f'''(x_R) + \frac{1}{4!}\Big(-\frac{h}{2}\Big)^4 f^{(iv)}(x_R) + \ldots\Big] \\
&+ \frac{9}{16}\Big[f(x_R) + \frac{h}{2}f'(x_R) + \frac{1}{2!}\Big(\frac{h}{2}\Big)^2 f''(x_R) + \frac{1}{3!}\Big(\frac{h}{2}\Big)^3 f'''(x_R) + \frac{1}{4!}\Big(\frac{h}{2}\Big)^4 f^{(iv)}(x_R) + \ldots\Big] \\
&- \frac{1}{16}\Big[f(x_R) + \frac{3h}{2}f'(x_R) + \frac{1}{2!}\Big(\frac{3h}{2}\Big)^2 f''(x_R) + \frac{1}{3!}\Big(\frac{3h}{2}\Big)^3 f'''(x_R) + \frac{1}{4!}\Big(\frac{3h}{2}\Big)^4 f^{(iv)}(x_R) + \ldots\Big],
\end{aligned}$$

and hence

$$\tilde{f}(x_R) = f(x_R) - \Big[\frac{3}{128}\, f^{(iv)}(x_R)\Big]\, h^4 + O(h^5)$$

It is clear in the above Taylor Series expansions, we didn't have to make use of what the actual value of h was. If we were able to sample the true function, $f(x)$, at a finer grid such that h was smaller, we'd expect to get a more accurate approximation to $f(x_R)$. This is true. The caviot is that we have to make sure we sample the points in a similar fashion such that we can use the same symmetric 4pt stencil to approximate values between two successive sampled points. Next we see that the error will decrease because we now have a smaller h. Note, if $h \to \frac{h}{4}$, the error decreases by a factor of $h^4 \to \frac{h^4}{256}$.

1.3.2 Non-symmetric function value stencils

In the example above, we approximated the value of $f(0.15)$ using a 4-pt symmetric stencil. It was briefly mentioned that if we wished to approximate $f(0.05)$, we would not be able to use the same 4-pt symmetric stencil, this is, of course, because we do not have the value of $f(-0.1)$ to use in our stencil by the points we were given to use, i.e.,

$$\tilde{f}(0.05) = -\frac{1}{16}\, f(-0.1) + \frac{9}{16}\, f(0.0) + \frac{9}{16}\, f(0.1) - \frac{1}{16}\, f(0.2).$$

Rather we can use a non-symmetric stencil to approximate its value, i.e.,

$$\tilde{f}(0.05) = \tilde{c}_0\, f(0) + \tilde{c}_1\, f(0.1) + \tilde{c}_2\, f(0.2) + \tilde{c}_3\, f(0.3)$$

and hence we wish to construct the 4-pt non-symmetric stencil using the transformation $\tilde{x} = x - x_R$,

$$\Rightarrow \tilde{f}\Big(\frac{h}{2}\Big) = \tilde{c}_0\, f\Big(-\frac{h}{2}\Big) + \tilde{c}_1\, f\Big(\frac{h}{2}\Big) + \tilde{c}_2\, f\Big(\frac{3h}{2}\Big) + \tilde{c}_3\, f\Big(\frac{5h}{2}\Big).$$

The method will not be any different than the one previously done above as you will see. We begin with the *moment equations*,

$$\begin{aligned}
\tilde{c}_0 \cdot 1 + \tilde{c}_1 \cdot 1 + \tilde{c}_2 \cdot 1 + \tilde{c}_3 \cdot 1 &= 1\\
\tilde{c}_0 \cdot \Big(-\frac{h}{2}\Big) + \tilde{c}_1 \cdot \Big(\frac{h}{2}\Big) + \tilde{c}_2 \cdot \Big(\frac{3h}{2}\Big) + \tilde{c}_3 \cdot \Big(\frac{5h}{2}\Big) &= 0\\
\tilde{c}_0 \cdot \Big(-\frac{h}{2}\Big)^2 + \tilde{c}_1 \cdot \Big(\frac{h}{2}\Big)^2 + \tilde{c}_2 \cdot \Big(\frac{3h}{2}\Big)^2 + \tilde{c}_3 \cdot \Big(\frac{5h}{2}\Big)^2 &= 0\\
\tilde{c}_0 \cdot \Big(-\frac{h}{2}\Big)^3 + \tilde{c}_1 \cdot \Big(\frac{h}{2}\Big)^3 + \tilde{c}_2 \cdot \Big(\frac{3h}{2}\Big)^3 + \tilde{c}_3 \cdot \Big(\frac{5h}{2}\Big)^3 &= 0
\end{aligned}$$

and obtain the system,

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ -\frac{h}{2} & \frac{h}{2} & \frac{3h}{2} & \frac{5h}{2} \\ (-\frac{h}{2})^2 & (\frac{h}{2})^2 & (\frac{3h}{2})^2 & (\frac{5h}{2})^2 \\ (-\frac{h}{2})^3 & (\frac{h}{2})^3 & (\frac{3h}{2})^3 & (\frac{5h}{2})^3 \end{bmatrix} \begin{pmatrix} \tilde{c}_0 \\ \tilde{c}_1 \\ \tilde{c}_2 \\ \tilde{c}_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

And hence we find,

$$\begin{pmatrix} \tilde{c}_0 \\ \tilde{c}_1 \\ \tilde{c}_2 \\ \tilde{c}_3 \end{pmatrix} = \begin{pmatrix} \frac{5}{16} \\ \frac{15}{16} \\ \frac{-5}{16} \\ -\frac{1}{16} \end{pmatrix}.$$

So we have the following stencil,

$$\tilde{f}(0) = \frac{5}{16}\, f\Big(-\frac{h}{2}\Big) + \frac{15}{16}\, f\Big(\frac{h}{2}\Big) - \frac{5}{16}\, f\Big(\frac{3h}{2}\Big) + \frac{1}{16}\, f\Big(\frac{5h}{2}\Big).$$

We can do a similar error analysis as well using Taylor Series, if we do that we find

$$\begin{aligned}\tilde{f}(0) = \frac{5}{16}\Big[f(0) - \frac{h}{2}f'(0) + \frac{1}{2!}\Big(-\frac{h}{2}\Big)^2 f''(0) + \frac{1}{3!}\Big(-\frac{h}{2}\Big)^3 f'''(0) + \frac{1}{4!}\Big(-\frac{h}{2}\Big)^4 f^{(iv)}(0) + \dots\Big] \\ + \frac{15}{16}\Big[f(0) + \frac{h}{2}f'(0) + \frac{1}{2!}\Big(\frac{h}{2}\Big)^2 f''(0) + \frac{1}{3!}\Big(\frac{h}{2}\Big)^3 f'''(0) + \frac{1}{4!}\Big(\frac{h}{2}\Big)^4 f^{(iv)}(0) + \dots\Big] \\ - \frac{5}{16}\Big[f(0) + \frac{3h}{2}f'(0) + \frac{1}{2!}\Big(\frac{3h}{2}\Big)^2 f''(0) + \frac{1}{3!}\Big(\frac{3h}{2}\Big)^3 f'''(0) + \frac{1}{4!}\Big(\frac{3h}{2}\Big)^4 f^{(iv)}(0) + \dots\Big] \\ + \frac{1}{16}\Big[f(0) + \frac{5h}{2}f'(0) + \frac{1}{2!}\Big(\frac{5h}{2}\Big)^2 f''(0) + \frac{1}{3!}\Big(\frac{5h}{2}\Big)^3 f'''(0) + \frac{1}{4!}\Big(\frac{5h}{2}\Big)^4 f^{(iv)}(0) + \dots\Big],\end{aligned}$$

and hence we get the following error

$$\tilde{f}(\tilde{x}=0) = f(\tilde{x}=0) + \Big[\frac{115}{3072}\, f^{(iv)}(\tilde{x}=0)\Big]\, h^4 + O(h^5).$$

Let’s now move onto Differentiation stencils using the machinery we have built up here.

Well, there is a long and a short answer to this. Much of the machinery for stencils we have already built up in previous sections. Now it is time to harness it. Let's explore the short answer, via an example. But first, let's define some of the players in the game, i.e., the tools we can use, i.e., the variables we're given.

Assume we're given a uniform grid of $N+1$ points, $\{x_0 + jh\}_{j=0}^{N}$, where we define, $x_j = x_0 + jh$, as well as their associated function values, $u_j = u(x_j) = u(x_0 + jh)$ for $j = 0, 1, \ldots N$.

1.4.1 SYMMETRIC 3-PT 2^{nd} DERIVATIVE STENCIL

Without loss of generality, let's suppose we wish the find the second derivative of our true nice and smooth function, $u(x)$, at one of the grid points, say x_m, where $m = 1, 2, \ldots, N-1$, i.e, $\tilde{u}''_m = \tilde{u}''(x_m)\ \forall x_m \in \{x_1, x_2, \ldots, x_{N-1}\}$ using a *symmetric* 3-pt stencil. Again, if we wanted to find an approximate value for the second derivative at the end-points, x_0 or x_N, we would need to find a different stencil that is non-symmetric, since points x_{-1} and x_{N+1} do not exist for our considerations.

Recall since we want to find a *symmetric* 3-pt second derivative stencil, we have the following skeletal form for our stencil,

$$\tilde{u}''_m = c_0 u_{m-1} + c_1 u_m + c_2 u_{m+1},$$

or equivalently, using $\tilde{x} = x - x_m$,

$$\tilde{u}''_0 = c_0 u_{-h} + c_1 u_0 + c_2 u_h,$$

Note that even though we are approximating derivatives, we can still only use information about the function values to do our approximations.

Just like in the previous section, we will construct the *moment equations* using the *method of undetermined coefficients* to ensure that our 3-pt 2^{nd} derivative stencil exact for the first 3 polynomials basis functions.

Note: If we had an n-pt symmetric 2^{nd} derivative stencil, we'd expect the first n monomials to be differentiated exactly.

Setting up the *moment equations*, we get

$$
\begin{aligned}
c_0 \cdot 1 + c_1 \cdot 1 + c_2 \cdot 1 &= \frac{d^2}{d\tilde{x}^2}\Big(1\Big)\Big|_{\tilde{x}=0} \\
c_0 \cdot 1 + c_1 \cdot 1 + c_2 \cdot 1 &= \frac{d^2}{d\tilde{x}^2}\Big(\tilde{x}\Big)\Big|_{\tilde{x}=0} \\
c_0 \cdot 1 + c_1 \cdot 1 + c_2 \cdot 1 &= \frac{d^2}{d\tilde{x}^2}\Big(\tilde{x}^2\Big)\Big|_{\tilde{x}=0}
\end{aligned}
$$

and obtain the following linear system,

$$
\begin{bmatrix} 1 & 1 & 1 \\ -h & 0 & h \\ (-h)^2 & 0 & h^2 \end{bmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}.
$$

Solving the above system yields,

$$
\begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{h^2} \\ -\frac{2}{h^2} \\ \frac{1}{h^2} \end{pmatrix},
$$

and hence a stencil of

$$
\tilde{u}_0'' = \frac{u_h - 2u_0 + u_{-h}}{h^2}.
$$

There are a few things to note. The first thing is that for a 2^{nd} derivative stencil, we need *at least* 3-pts in our stencil in order for ensure that the we would match exact values of the second derivatives for certain polynomial orders in the *moment equations*.

We will now prove that they match exactly. Let

$$
u(\tilde{x}) = a\tilde{x}^2 + b\tilde{x} + c,
$$

and hence

$$
u''(\tilde{x}) = 2a.
$$

Now we subtract our stencil from the true value of the second derivative of $u(\tilde{x})$ evaluated at the point $\tilde{x} = 0$.

$$
u''(\tilde{x})\Big|_{\tilde{x}=0} - \frac{u_h - 2u_0 + u_{-h}}{h^2} = 2a - \frac{(ah^2 + bh + c) - 2(c) + (a(-h)^2 + b(-h) + c)}{h^2} = 2a - \frac{2ah^2}{h^2} = 0,
$$

hence proving our stencil gives exact values of the second derivatives for polynomials up to order 2. *Note* that this stencil is actual exact for polynomials up to order 3 by virtue of the symmetry and parity of monomial basis functions. It is now time to perform the error analysis on this differentiation stencil.

1.4.2 Error Analysis of the Symmetric 3-pt 2^{nd} Derivative Stencil

Not to beat a dead horse, (I apologize for such harsh imagery), but again we will call on our best friend, Taylor Series, to do the work necessary to find error estimates. Completely in the veins of the methodology for error analysis in approximating function values, we start with the stencil and expand all the functions about our point of interest, namely x_m (or $\tilde{x} = 0$).

$$\tilde{u}_0'' = \frac{u_h - 2u_0 + u_{-h}}{h^2}$$

$$= \frac{1}{h^2}\Big(u(0+h) - 2u(0) + u(0-h)\Big)$$

$$= \frac{1}{h^2}\Bigg(\Big(u(0) + hu'(0) + \frac{1}{2!}h^2u''(0) + \frac{1}{3!}h^2u'''(0) + \frac{1}{4!}h^4u^{(iv)}(0) + O(h^5)\Big) - 2u(0) + \ldots$$
$$+ \ldots \Big(u(0) - hu'(0) + \frac{1}{2!}h^2u''(0) - \frac{1}{3!}h^2u'''(0) + \frac{1}{4!}h^4u^{(iv)}(0) - O(h^5)\Big)\Bigg)$$

$$= u''(0) + \left[\frac{1}{288}u^{(iv)}(0)\right] h^2 + O(h^3)$$

Thus our symmetric 3-pt second derivative stencil is *second-order accurate.*

1.5 Differentiation Stencils Generalized

You may have already noticed that the above procedure for calculating a 2^{nd} derivative stencil can be generalized for, say, the n^{th} derivative as long as we have $N+1$ points in our stencil. This is because we need *at least* $N+1$ points in the stencil to get to the appropriate number of equations in our *moment equation system* to ensure we can match values between our stencil and N polynomials with appropriate derivative values.

Without loss of generality, say we want to find an $2n+1$-pt. stencil for approximate the $2n^{th}$ derivative of a function, $u(x)$. Note that $2n+1$ is inherently odd for symmetry. For an even point stencil, you would do a completely analogous procedure, but the stencil would simply no longer be symmetric. We are looking for a stencil of the form,

$$\tilde{u}_0'' = \sum_{j=-n}^{n} c_j u_j.$$

We then have the following system,

$$\begin{bmatrix} 1 & 1 & \dots & 1 & 1 & 1 & \dots & 1 \\ (-nh) & (-n+1)h & \dots & -h & 0 & h & \dots & (nh) \\ (-nh)^2 & \Big((-n+1)h\Big)^2 & \dots & (-h)^2 & 0 & h^2 & \dots & (nh)^2 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ (-nh)^{2n} & \Big((-n+1)h\Big)^{2n} & \dots & (-h)^{2n} & 0 & h^{2n} & \dots & (nh)^{2n} \end{bmatrix} \begin{pmatrix} c_{-n} \\ c_{-n+1} \\ \vdots \\ c_0 \\ \vdots \\ c_{n-1} \\ c_n \end{pmatrix} = \begin{pmatrix} \frac{d^{2n}}{d\tilde{x}^{2n}}\Big(1\Big)\Big|_{\tilde{x}=0} \\ \frac{d^{2n}}{d\tilde{x}^{2n}}\Big(\tilde{x}\Big)\Big|_{\tilde{x}=0} \\ \frac{d^{2n}}{d\tilde{x}^{2n}}\Big(\tilde{x}^2\Big)\Big|_{\tilde{x}=0} \\ \vdots \\ \frac{d^{2n}}{d\tilde{x}^{2n}}\Big(\tilde{x}^{2n-1}\Big)\Big|_{\tilde{x}=0} \\ \frac{d^{2n}}{d\tilde{x}^{2n}}\Big(\tilde{x}^{2n}\Big)\Big|_{\tilde{x}=0} \end{pmatrix}$$

and hence

$$\begin{bmatrix} 1 & 1 & \dots & 1 & 1 & 1 & \dots & 1 \\ (-nh) & (-n+1)h & \dots & -h & 0 & h & \dots & (nh) \\ (-nh)^2 & \Big((-n+1)h\Big)^2 & \dots & (-h)^2 & 0 & h^2 & \dots & (nh)^2 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ \\ \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ (-nh)^{2n} & \Big((-n+1)h\Big)^{2n} & \dots & (-h)^{2n} & 0 & h^{2n} & \dots & (nh)^{2n} \end{bmatrix} \begin{pmatrix} c_{-n} \\ c_{-n+1} \\ \vdots \\ c_0 \\ \vdots \\ c_{n-1} \\ c_n \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \\ (2n)! \end{pmatrix}.$$

We now make note that error analysis can be completed in a completely analogous way using Taylor Series. It is the author's hopes that if you're using the above generalized differentiation stencil, you never have to do it by hand for, say $n > 3$. ☺

Finally, let's take a glance, or two, at creating stencils for integration.

2 Integration Stencils

In our venture so far, we've always assumed the stencil points were given and only had to solve for the weight coefficients. If we had n points in our stencil, this translated into n degrees of freedom, for choosing the coefficients, $\{c_j\}_{j=0}n-1$, in a way to minimize the error, i.e., cancel terms in the Taylor series analysis.

However we now can choose whether we wish to exploit the extra degrees of freedom (*Gaussian quadrature*) or not (*Newton-Cotes quadrature*), to find an approximate solution to the following problem,

$$I = \int_a^b f(x)dx.$$

Basically we have two avenues to go to now,

1. *Newton-Cotes Quadrature*: Use n degrees of freedom coming from *choosing* ONLY the weight coefficients, $\{c_j\}_{j=0}^{n-1}$ since the quadrature points $\{x_j\}_{j=0}^{n-1} \in [a,b]$ will be uniformly spaced. The weight coefficients are again chosen to minimize error.

   Hence the stencil looks like

   $$I = \int_a^b f(x)dx \approx \sum_{j=0}^{n-1} c_j f(x_j),$$

   but where ONLY $\{c_j\}$ are unknown.

2. *Gaussian Quadrature*: Use $2n$ degrees of freedom coming from *choosing* BOTH the quadrature points $\{x_j\}_{j=0}^{n-1} \in [a,b]$ and the weight coefficients, $\{c_j\}_{j=0}^{n-1}$ to minimize error.

   Note in this case that the quadrature points, $\{x_j\}$, do not necessarily have to uniformly spaced. *Spoiler* - they won't be. Hence our stencil is,

$$I = \int_a^b f(x)dx \approx \sum_{j=0}^{n-1} c_j f(x_j),$$

where BOTH $\{c_j\}$ and $\{x_j\}$ are unknown.

We will now begin with a discussion about Newton-Cotes quadrature.

2.1 Newton-Cotes Quadrature

Without loss of generality, let's look at the following problem,

$$I = \int_0^h f(x)dx.$$

Remember we have been traditionally calling h the step-size, or grid-size; however, now we will call it the size of the integration interval. If we want $N+1$ quadrature points in our stencil, we choose them uniformly, using $\delta x = \frac{h}{N}$, i.e.,

$$\{x_j\}_{j=0}^{N} = \{j\delta x\}_{j=0}^{N}.$$

Now that we have our quadrature points, it is time to find our coefficients. To illustrate the process (a process that will feel very, *very* similar to how we've constructed stencils before but with one tweak), we will look at an **example** with 3 quadrature points, i.e., $\{0, \frac{h}{2}, h\}$. Our stencil will take the form,

$$I_h = c_0 f(0) + c_1 f\left(\frac{h}{2}\right) + c_2 f(h).$$

Once again, we will set up a formulation using the *method of undetermined coefficients*, via constructing *moment equations*. In this case, since we have 3 quadrature points, we want to ensure that the first 3 monomial basis functions integrate exactly. The equations we get are:

$$
\begin{aligned}
c_0 \cdot 1 + c_1 \cdot 1 + c_2 \cdot 1 &= \int_0^h 1 dx \\
c_0 \cdot 1 + c_1 \cdot 1 + c_2 \cdot 1 &= \int_0^h x dx \\
c_0 \cdot 1 + c_1 \cdot 1 + c_2 \cdot 1 &= \int_0^h x^2 dx
\end{aligned}
$$

and obtain the following linear system,

$$
\begin{bmatrix} 1 & 1 & 1 \\ 0 & \frac{h}{2} & h \\ 0 & \left(\frac{h}{2}\right)^2 & h^2 \end{bmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} h \\ \frac{h^2}{2} \\ \frac{h^3}{3} \end{pmatrix}.
$$

Solving the above system yields,

$$
\begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} \frac{h}{6} \\ \\ \frac{2h}{3} \\ \\ \frac{h}{6} \end{pmatrix},
$$

Hence our 3-pt Newton-Cotes integration stencil is

$$
I_h = \frac{h}{6}\left[f(0) + 4f\left(\frac{h}{2}\right) + f(h) \right].
$$

Every Newton-Cotes scheme can be constructed the same way, just adding in appropriate rows and columns in the above matrix accordingly as well as the number of integrations on monomial polynomials.

The caviot is we still have to do some integrations, i.e., the RHS vector of the above linear system, in order to find the coefficients to do our *numerical integration.* This may seem silly, but it basically just safeguards us from doing mindless integrations using numerical methods. Or, it just makes sure we have a basic understanding of integration before we start wielding mathematical weapons of mass fun.

There are two things left for us to do. We can prove that the above integration is exact for polynomials up to 2^{nd} order as well as perform error analysis.

1. *Exact for* 2^{nd} *order polynomials*: Let $f(x) = ax^2 + bx + c$. Now let's look at the difference between $\int_0^h f(x)dx$ and our stencil, i.e.,

$$\int_0^h ax^2 + bx + cdx - \frac{h}{6}\left[f(0) + 4f\left(\frac{h}{2}\right) + f(h)\right]$$

$$= \frac{ah^3}{3} + \frac{bh^2}{2} + ch - \frac{h}{6}\left[c + 4\left(\frac{ah^2}{4} + \frac{bh}{2} + c\right) + (ah^2 + bh + c)\right]$$

$$= \frac{ah^3}{3} + \frac{bh^2}{2} + ch - \frac{h}{6}\left[2ah^2 + 3bh + 6c\right] = 0.$$

2. *Error Analysis*: Welp, just to toss a slight curve into our error analysis, we can proceed with the usual Taylor series analysis for our stencil; however, what do we do with the integral? The answer is staring at us right in the face - just use Taylor series there as well. For our analysis we will center the Taylor's series about 0. Let's check it out.

$$Error = I - I_h$$

$$= \int_0^h f(x)dx - \frac{h}{6}\left[f(0) + 4f\left(\frac{h}{2}\right) + f(h)\right]$$

$$= \int_0^h f(0+x)dx - \frac{h}{6}\left[f(0) + 4f\left(0+\frac{h}{2}\right) + f(0+h)\right]$$

$$\begin{aligned}
= \int_0^h \left(f(0) + xf'(0) + \frac{x^2}{2!}f''(0) + \frac{x^3}{3!}f'''(0) + \dots\right)dx - \frac{h}{6}\Bigg[f(0) + \dots \\
+ 4\left(f(0) + \frac{h}{2}f'(0) + \frac{h^2}{4\cdot 2!}f''(0) + \frac{h^3}{8\cdot 3!}f'''(0) + \dots\right) + \dots \\
+ \left(f(0) + hf'(0) + \frac{h^2}{2!}f''(0) + \frac{h^3}{3!}f'''(0) + \dots\right)\Bigg]
\end{aligned}$$

$$\begin{aligned}
= \left[f(0)x + \frac{1}{2!}f'(0)x^2 + \frac{1}{3!}f''(0)x^3 + \frac{1}{4!}f'''(0)x^4 + \frac{1}{5!}f^{(iv)}(0)x^5 + \dots\right]\Bigg|_{x=0}^{x=h} \\
- \frac{h}{6}\left[6f(0) + 3hf'(0) + h^2 f''(0) + \frac{3}{2\cdot 3!}\, h^3 f'''(0) + \dots\right]
\end{aligned}$$

$$\begin{aligned}
= \left[f(0)h + \frac{1}{2!}f'(0)h^2 + \frac{1}{3!}f''(0)h^3 + \frac{1}{4!}f'''(0)h^4 + \frac{1}{5!}f^{(iv)}(0)h^5 + \frac{1}{6!}f^{(v)}(0)h^6 + \dots\right] \\
- \left[f(0) + \frac{1}{2}\, h^2 f'(0) + \frac{1}{6}\, h^3 f''(0) + \frac{1}{4\cdot 3!}\, h^4 f'''(0) + \frac{5}{24\cdot 4!}h^5 f^{(iv)} + \dots\right]
\end{aligned}$$

$$= \left[\frac{-1}{2880}f^{(iv)}(0)\right]\, h^5 + O(h^6).$$

Hence we see that this 3-pt integration is 5^{th} order accurate.

A few questions that may be lingering in the back of your mind are, "Newton-Cotes seems convenient, why ever use something else?" or "Why is Newton's name attached to this as well, is there anything he didn't do?" Well, to answer the former - there are definite positives and negatives associated with a Newton-Cotes scheme.

Some of the positives is that constructing a stencil is very straight-forward, the quadrature points are easy to compute, and the error estimate is simple to find and approximate. Furthermore, we can apply an extrapolation scheme to these type of error estimates to bump up the order of the quadrature, i.e., *Romberg Integration*.

Some of the negatives are that since we have uniform quadrature points, there is going to be a numerical upper limit on the number of points we can use before the linear system becomes ill-conditioned, we are not taking advantage of all degrees of freedom that would be available to us to get an even more accurate method for the same number of quadrature points (not the same points, though), and that there is no way for us to take into account singular integrands.

We will now switch gears and talk about *Gaussian quadrature*.

2.2 Gaussian Quadrature

We just saw that using a 3-pt Newton-Cotes integration stencil yields a 3^{rd} order accurate integration scheme. What if I told you, if we spent *a tad* little more time to find better quadrature points than just uniformly partitioned points, we could actually achieve an integration scheme that is 6^{th} order accurate? To good to be true? Nah. What's the catch?

The catch is that to find those 3 elusive quadrature points is going to take a little extra theory to develop. Fortunately, this is a beautiful subject involving inner products over infinite-dimensional vector spaces with orthogonal basis vectors, i.e., invoking some of the extraordinary beauty within Sturm-Liouville theory. It's almost time to call some of our friends, various orthogonal polynomial basis functions, to join the party!
As an example we will find the 2-pt Gaussian quadrature for the integral,

$$I = \int_0^h f(x)dx.$$

This example was chosen for two reasons. The first because we will show that a 2-pt Gaussian quadrature is more accurate than a 3-pt Newton-Cotes method, and the second because this author doesn't believe doing higher order stencils will add any extra insight to Gaussian elimination (of course besides illustrating the author's terrible ability of meticulous arithmetic).

Hence the 2-pt Gaussian quadrature stencil we're looking for will take the form,

$$I_h = c_0 f(x_0) + c_1 f(x_1),$$

where $\{x_0, x_1\} \in [0,1]$. We start where we always start with these stencils - writing out the *moment equations* associated with the *method of undetermined coefficients.* If we do this, we get the modest looking equations,

$$
\begin{aligned}
c_0 \cdot 1 + c_1 \cdot 1 &= \int_0^h 1 dx \\
c_1 \cdot x_0 + c_1 \cdot x_1 &= \int_0^h x dx. \\
c_1 \cdot x_0^2 + c_1 \cdot x_1^2 &= \int_0^h x^2 dx. \\
c_1 \cdot x_0^3 + c_1 \cdot x_1^3 &= \int_0^h x^3 dx.
\end{aligned}
$$

Recall that in the above equations c_0, c_1, x_0, and x_1 are all unknowns. It is clear the above equation is a system of 4 non-linear coupled equations. We could simply attempt to solve this system, or we can build up a stronger formalism for essentially solving first for the quadrature points and then using those to find the weight coefficients.

The first step is to definite the correct *inner product.* In this case, we do not have any *weight function,* $w(x)$, in the integrand so we can define the inner product as

$$
< f, g >= \int_0^h g(x) \cdot \bar{f}(x) dx = \int_0^h g(x) \cdot f(x) dx =
$$

Now, because we will only be dealing with *real* functions, or functions with the mapping, $f : \mathbb{R} \to \mathbb{R}$, we do not need the conjugation of f in the inner product above. (But keeping it won't hurt anything, afterall).

It is now time to put this inner product to use. It was stated briefly above we would need to use some orthogonal polynomials. For an n-pt Gaussian quadrature scheme, we will need to set the n^{th} order orthogonal polynomial equal to zero. The roots that we obtain will be the quadrature points we are after.

It is now time to build up a basis of orthogonal polynomials using this inner product. Note, this will feel very similar to the method for orthogonalizing vectors using a standard Gram-Schmidt process.

Essentially we will iteratively subtract projections of previously computed orthogonal polynomials from the current polynomial we are trying to build up. We will now illustrate by example for our stencil with 2 quadrature points. We start with a the lowest order polynomial, namely a constant.

$$p_0(x) = 1.$$

Now to find $p_1(x)$, we the analogous step to Gram-Schmidt,

$$p_1(x) = xp_0(x) - \frac{< xp_0, p_0 >}{< p_0, p_0 >} p_0(x).$$

To find $p_2(x)$, we will again do the natural thing for Gram-Schmidt,

$$p_2(x) = xp_1(x) - \frac{< xp_1, p_1 >}{< p_1, p_1 >} p_1(x) - \frac{< xp_1, p_0 >}{< p_0, p_0 >} p_0(x).$$

Now that we have the form of the 2^{nd} order orthogonal polynomial in this inner product ,$p_2(x)$ (and all the lower order polynomials), we can perform all the necessary inner products to compute explicitly find the polynomials.

$$< 1, 1 > = \int_0^h 1 dx = h$$

$$< x, 1 > = \int_0^h x dx = \frac{h^2}{2}$$

$$< x, 2 > = \int_0^h x^2 dx = \frac{h^3}{4}$$

$$< x, 3 > = \int_0^h x^3 dx = \frac{h^4}{4}$$

and hence we find

$$\begin{aligned} p_0(x) &= 1 \\ p_1(x) &= x - \frac{1}{2} \\ p_2(x) &= x^2 - hx + \frac{1}{6} h^2. \end{aligned}$$

Therefore our 2 quadrature points are roots of $p_2(x)$. Finding the roots of $p_2(x)$, we get

$$x_0 = \frac{h}{6}(3-\sqrt{3}) \quad \text{and} \quad x_1 = \frac{h}{6}(3+\sqrt{3}).$$

Lastly we need to find the weight coefficients. Yuuuup, you've guessed it! We can derive the *moment equations* again using the *method of undetermined coefficients*, i.e.,

$$c_0 \cdot 1 + c_1 \cdot 1 =< 1, 1 >$$

$$c_0 \cdot x_0 + c_1 \cdot x_2 =< x, 1 >,$$

and obtain the system

$$\begin{bmatrix} 1 & 1 \\ x_0 & x_1 \end{bmatrix} \begin{pmatrix} c_0 \\ c_1 \end{pmatrix} = \begin{pmatrix} h \\ \frac{h^2}{2} \end{pmatrix}.$$

Solving this system, we find

$$\begin{pmatrix} c_0 \\ \\ c_1 \end{pmatrix} = \begin{pmatrix} \frac{h}{2} \\ \\ \frac{h}{2} \end{pmatrix}.$$

Hence we find that our 2-pt Gaussan quadrature stencil is

$$I_h = \frac{h}{2} f\left(\frac{h}{6}(3-\sqrt{3})\right) + \frac{h}{2} f\left(\frac{h}{6}(3+\sqrt{3})\right).$$

Now let's do what we do best (well maybe second best to setting up the *moment equations*) and prove that this stencil is exact for polynomials up the order 3 and perform error analysis.

- *Exact for* 3^{rd} *order polynomials*: Let $f(x) = bx^3 + cx^2 + dx + e$. Now let's look at the difference between $\int_0^h f(x)dx$ and our stencil, i.e.,

$$\int_0^h bx^3 + cx^2 + dx + e\,dx - \frac{h}{2}\left[f\left(\frac{h}{2}\left(1-\frac{1}{\sqrt{3}}\right)\right) + f\left(\frac{h}{2}\left(1+\frac{1}{\sqrt{3}}\right)\right)\right]$$

$$= \frac{bh^4}{4} + \frac{ch^3}{3} + \frac{dh^2}{2} + eh - \frac{h}{2}\left[\frac{bh^3}{2} + \frac{2ch^2}{3} + dh + 2e\right]$$

$$= \frac{bh^4}{4} + \frac{ch^3}{3} + \frac{dh^2}{2} + eh - \left[\frac{bh^4}{4} + \frac{ch^3}{3} + \frac{dh^2}{2} + eh\right] = 0.$$

- emphError analysis for 2-pt Gaussian Quadrature Stencil: We will now call upon our friend, Taylor Series, and perform error analysis completely analogously to how we did it for Newton-Cotes quadrature. The only slight hiccup is that the quadrature points aren't so simple, but that's okay. It should add to the fun.

$Error = I - I_h$

$$= \int_0^h f(x)dx - \frac{h}{2}\left[f\left(\frac{h}{2}\left(1-\frac{1}{\sqrt{3}}\right)\right) + f\left(\frac{h}{2}\left(1+\frac{1}{\sqrt{3}}\right)\right)\right]$$

$$= \int_0^h f(0+x)dx - \frac{h}{2}\left[f\left(0+\frac{h}{2}\left(1-\frac{1}{\sqrt{3}}\right)\right) + f\left(0+\frac{h}{2}\left(1+\frac{1}{\sqrt{3}}\right)\right)\right]$$

$$\begin{aligned} &= \int_0^h \left(f(0) + xf'(0) + \frac{x^2}{2!}f''(0) + \frac{x^3}{3!}f'''(0) + \ldots\right)dx \\ &\qquad - \frac{h}{2}\left[\left(f(0) + \frac{h}{2}\left(1-\frac{1}{\sqrt{3}}\right)f'(0) + \frac{1}{2!}\left(\frac{h}{2}\left(1-\frac{1}{\sqrt{3}}\right)^2\right)f''(0) + \frac{1}{3!}\left(\frac{h}{2}\left(1-\frac{1}{\sqrt{3}}\right)^3\right)f'''(0) + \ldots\right) + \ldots\right. \\ &\qquad\qquad \left. + \left(f(0) + \frac{h}{2}\left(1+\frac{1}{\sqrt{3}}\right)f'(0) + \frac{1}{2!}\left(\frac{h}{2}\left(1+\frac{1}{\sqrt{3}}\right)^2\right)f''(0) + \frac{1}{3!}\left(\frac{h}{2}\left(1+\frac{1}{\sqrt{3}}\right)^3\right)f'''(0) + \ldots\right)\right] \end{aligned}$$

$$\begin{aligned} &= \left[f(0)x + \frac{1}{2!}f'(0)x^2 + \frac{1}{3!}f''(0)x^3 + \frac{1}{4!}f'''(0)x^4 + \frac{1}{5!}f^{(iv)}(0)x^5 + \ldots\right]\Bigg|_{x=0}^{x=h} \\ &\qquad - \frac{h}{2}\left[2f(0) + hf'(0) + \frac{h^2}{3}f''(0) + \frac{h^3}{12}f'''(0) + \frac{7}{432}\,h^4 f^{(iv)}(0) + O(h^5)\right] \end{aligned}$$

$$\begin{aligned} &= \left[f(0)h + \frac{1}{2!}f'(0)h^2 + \frac{1}{3!}f''(0)h^3 + \frac{1}{4!}f'''(0)h^4 + \frac{1}{5!}f^{(iv)}(0)h^5 + \ldots\right] \\ &\qquad - \left[hf(0) + \frac{1}{2}\,h^2f'(0) + \frac{1}{6}\,h^3f''(0) + \frac{1}{24}\,h^4f'''(0) + \frac{7}{864}\,h^5f^{(iv)}(0) + O(h^6)\ldots\right] \end{aligned}$$

$$= \left[\frac{1}{4320}f^{(iv)}(0)\right]\,h^5 + O(h^6).$$

Hence we find our 2-pt Gaussian quadrature stencil is 5^{th} order accurate *locally*. At this point we can also brag that this 2-pt integration stencil is more accurate than the 3-pt Newton-Cotes scheme we derived earlier. There are a few things we wish to note for Gaussian quadrature stencils.

1. For an n-pt Gaussian quadrature stencil, the stencil will yield the exact result for polynomials of degree $2n-1$.
2. For an n-pt Gaussian quadrature stencil, we expect it to be $(2n+1)^{th}$ order accurate locally.

The last thing we need to discuss with Gaussian quadrature is it's ability to accurately integrate functions with integrable singularities.

2.2.1 Singular Integrals and Gaussian Quadrature

We will now discuss how to integrate functions of the following form,

$$I = \int_0^h w(x) f(x) dx,$$

where $f(x) \in C^\infty$ and $w(x)$ is singular, but integrable at 0. Some examples of what $w(x)$ may be are $w(x) = \sqrt{x}$ or $w(x) = \frac{1}{\sqrt{x}}$.

Let's illustrate this with an example. Say we wish to approximate the solution to the following singular integral using a 1-pt Gaussian quadrature stencil,

$$I = \int_0^h \frac{f(x)}{\sqrt{x}} dx,$$

even though we're using a 1-pt Gaussian quadrature, we'd expect more convergence than using any order Newton-Cotes method since the Newton-Cotes method will evaulate $\frac{f(x)}{\sqrt{x}}$ at the end-point, which is the singular point.

To begin constructing our Gaussian quadrature stencil, the first thing we need to do is define a new inner product that takes into account the singular function, $w(x)$. This is relatively painless, simply we define

$$< f, g > = \int_0^h \frac{f(x)g(x)}{sqrtx} dx.$$

Now that we have defined our inner product, we can begin construct a basis of orthogonal polynomials in the space defined by this inner product.

$$p_0(x) = 1$$

$$p_1(x) x p_0(x) - \frac{< x p_0, p_0 >}{< p_0, p_0 >} p_0(x) = x - \frac{1}{3} h,$$

since $< p_0, p_0 >=< 1, 1 >= 2\sqrt{h}$ and $< xp_0, p_0 >=< x, 1 >= \frac{2}{3}h^{3/2}$. Hence since we are only interested in a 1-pt stencil, we can set $p_1(x) = 0$ to find our quadrature point, i.e.,

$$p_1(x) = x - \frac{h}{3} = 0 \ \Rightarrow \ x = \frac{h}{3}.$$

Therefore our quadrature point is $x_0 = \frac{h}{3}$ and we can write out the *moment equation* to find its associated weight coefficient. This is straight-forward for our 1-pt stencil and is completely analogous to how we did it for the standard Gaussian quadrature methodology. We just need to make sure we use the appropriate inner product. Check it!

$$[1](c_0) =< 1, 1 >= \int_0^h \frac{1}{\sqrt{x}} dx$$

and hence we find that

$$c_0 = 2\sqrt{h}$$

so our 1-pt Gaussian quadrature for the singular integral above is

$$I_h = 2\sqrt{h} f\Big(\frac{h}{3}\Big).$$

We note that this stencil should be exact for polynomials of order 1 (since we expect an n-pt Gaussian quadrature stencil to be exact for polynomials of order $(2n-1)$ and below, i.e., for $f(x) = ax + b$,

$$\int_0^h \frac{ax+b}{\sqrt{x}} dx - 2\sqrt{h} f\Big(\frac{h}{3}\Big) = \frac{2}{3}ah^{3/2} + 2b\sqrt{h} - 2\sqrt{h}\Big(a\Big(\frac{h}{3}\Big) + b\Big) = 0.$$

Moreover we can perform the error analysis in a completely analogous way as before,

$$\text{Error} = I - I_h = \int_0^h \frac{f(x)}{\sqrt{x}}\,dx - 2\sqrt{h}\,f\left(\frac{h}{3}\right)$$

$$= \int_0^h \frac{1}{\sqrt{x}}\, f(0+x)\,dx - 2\sqrt{h}\,f\left(0+\frac{h}{3}\right)$$

$$= \int_0^h \frac{1}{\sqrt{x}}\left(f(0) + xf'(0) + \frac{x^2}{2!}f''(0) + \frac{x^3}{3!}f'''(0) + \dots\right)$$
$$- 2\sqrt{h}\left(f(0) + \frac{h}{3}f'(0) + \frac{h^2}{9\cdot 2!}f''(0) + \frac{h^3}{27\cdot 3!}f'''(0) + \dots\right)$$

$$= \left[2f(0)x^{1/2} + \frac{2}{3}f'(0)x^{3/2} + \frac{5}{2\cdot 2!}f''(0)x^{5/2} + \frac{7}{2\cdot 3!}f'''(0)x^{7/2}\right]\Bigg|_0^h$$
$$- \left(2\sqrt{h}f(0) + \frac{2h^{3/2}}{3}f'(0) + \frac{2h^{5/2}}{9\cdot 2!}f''(0) + \frac{2h^{7/2}}{27\cdot 3!}f'''(0) + \dots\right)$$

$$= \left[2f(0)h^{1/2} + \frac{2}{3}f'(0)h^{3/2} + \frac{5}{2\cdot 2!}f''(0)h^{5/2} + \frac{7}{2\cdot 3!}f'''(0)h^{7/2}\right]\Bigg|_0^h$$
$$- \left(2h^{1/2}f(0) + \frac{2h^{3/2}}{3}f'(0) + \frac{2h^{5/2}}{9\cdot 2!}f''(0) + \frac{2h^{7/2}}{27\cdot 3!}f'''(0) + \dots\right)$$

$$= \left[\frac{41}{36}f''(0)\right]h^{5/2} + O\left(h^{7/2}\right).$$

Hence our 1-pt Gaussian quadrature stencil is order $\frac{5}{2}$ accurate. Note, that in a standard (non-singular) Gaussian quadrature case, for a 1-pt stencil we would expect to be 3^{rd} order accurate.

2.2.2 Steps to performing Gaussian Quadrature

We will now list the general procedure for constructing a stencil for Gaussian quadrature. Consider the integral

$$I = \int_0^h w(x)f(x)\,dx,$$

where we consider $w(x)$ to be an integrable function or simply 1, and $f(x)$ is, of course, a smooth function (or *smooth enough* - meaning $f(x)$ has enough continuous derivatives for the error analysis in the stencil we desire.)

- *Steps for n-pt Gaussian Quadrature:* $I_h = \sum_{j=0}^{n-1} c_j f(x_j)$

  1. Define an inner product,
  $$< f, g > = \int_0^h w(x) f(x) g(x) dx.$$
  2. Use that inner product and construct a basis of orthogonal polynomials up to order n using the Gram-Schmidt analog, i.e.,
  $$p_n = x p_{n-1} - \sum_{j=0}^{n-1} \left[\frac{< x p_{n-1}(x), p_j >}{< p_j, p_j >} \right] p_j(x).$$
  3. Set $p_n(x)$ equal to zero and the roots, which will be unique, are the quadrature points.
  4. Construct the *moment equations* using the *method of undetermined coefficients* formulation and solve the resulting linear system to find the weight coefficients, $\{c_j\}$.

2.3 Composite Integration

So far in our story of numerical integration we considered the integral,

$$I = \int_0^h f(x) dx$$

where we approximated its value using an n-pt stencil, whether it is constructed as a Newton-Cotes scheme or Gaussian quadrature. What if instead, we partition the interval $[0, h]$ into N-subintervals, such as

$$[0, h] = \bigcup_{j=0}^{N-1} \left[\frac{jh}{N}, \frac{(j+1)h}{N} \right],$$

and then break the original integral up into a summation of successive integrals over these partitions,

$$I=\int_0^h f(x)dx=\sum_{j=0}^{N-1}\int_{jh/N}^{(j+1)h/N} f(x)dx.$$

Now consider the integral over one partition,

$$I_j=\int_{jh/N}^{(j+1)h/N} f(x)dx.$$

We can use our n-pt Newton-Cotes integration stencil to approximate the value of the integral over that partition using our known n-equally spaced quadrature points, using the same weights. We will illustrate this by example now.

2.3.1 Example: Partitioning integration interval into subintervals

Consider the integral,

$$\int_0^1 f(x)dx$$

, where $f(x)$ is smooth, spice, and everything nice. Consider a 3-pt Newton-Cotes scheme over $[0,1]$ to approximate the integral,

$$\int_0^h f(x)dx\approx I_{[0,h]}=\frac{1}{6}f(0)+\frac{2}{3}f(0.5h)+\frac{1}{6}f(h).$$

Now since know how to perform error analysis (and we performed the analysis above), we know that this Newton-Cotes method has the following error term,

$$|\text{error}|=\left[\frac{1}{2880}|f^{(iv)}(0)|\right]h^5+O(h^6).$$

However, we would expect if we integrated over a smaller region, other than $[0,h]$ using this stencil, we'd expect higher accuracy. Hence let's partition the interval $[0,h]$ into 10 subintervals,

$$[0,h] = \bigcup_{j=0}^{9} [j\Delta x, (j+1)\Delta x],$$

where $\Delta x = \frac{h}{10}$ and is the size of each partition. Hence our integration task now takes the form,

$$I_{[0,h]} = \int_0^h f(x)dx = \sum_{j=0}^{9} \int_{j\Delta x}^{(j+1)\Delta x} f(x)dx = \sum_{j=0}^{9} I_j,$$

where

$$I_j = \int_{j\Delta x}^{(j+1)\Delta x} f(x)dx.$$

What is convenient for us at this junction is that even though we are considering a summation of integrals over smaller bounds, we can still use the integration stencil we rigged up earlier!

If we wish to perform one of the integrals, say I_j, we can use that *same* 3-pt Newton-Cotes method we found earlier but just have to evaluate the function, $f(x)$, at the correct quadrature pts in the interval $[j\Delta x, (j+1)\Delta x]$, i.e.,

$$I_j = \int_{j\Delta x}^{(j+1)\Delta x} f(x)dx \;\approx\; \tilde{I}_j = \frac{1}{6}\, f\big(j\Delta x\big) + \frac{2}{3}\, f\Big(\Big(j+\frac{1}{2}\Big)\Delta x\Big) + \frac{1}{6}\, f\big((j+1)\Delta x\big).$$

Using the same method as before, we find that the error for $\tilde{I}_j$ is,

$$\text{error}_j = I_j - \tilde{I}_j = \left[\frac{1}{2880}|f^{(iv)}\big(j\Delta x\big)|\right] \Delta x^5 + O(\Delta x^6).$$

We will call the above error, the *local error* since it is the error over one sub-interval of our whole composite integration, or integral over the whole domain we're interested in. Now let's attempt to find the total error.

$$
\begin{aligned}
\text{Error} &= \sum_{j=0}^{9} \text{error}_j \\
&= \sum_{j=0}^{9} \Big[I_j - \tilde{I}_j \Big] \\
&= \sum_{j=0}^{9} \bigg[\frac{1}{2880} |f^{(iv)}\big(j\Delta x\big)| \bigg] \Delta x^5 + O(\Delta x^6) \\
&= C \sum_{j=0}^{9} \bigg[\Delta x^5 + O(\Delta x^6), \quad \text{where} \quad C = \max_{[0,h]} |f^{(iv)}(x)| \\
&= CN\Big(\Delta x^5 + O(\Delta x^6)\Big) \\
&= C\Delta x^4 + O(\Delta x^5),
\end{aligned}
$$

since $N = \frac{1}{\Delta x}$, as N is the number of sub-intervals, i.e., partitions. We now note that is your *local stencil* is n^{th} order accurate, if you use that stencil for a composite integration it will be $(n-1)^{st}$ accurate.

However, now we note the comparison between our integral on $[0,1]$ using simply the 3-pt stencil versus approximating the integral using the composite method. Recall, we have approximated $I_{[0,1]}$ using the following two methods,

1. *“Single-Stencil" Integration:*

$$
\int_0^h f(x)dx \approx \frac{1}{6} f(0) + \frac{2}{3} f(0.5) + \frac{1}{6} f(1.0),
$$

with error,

$$
\text{error}_s = \bigg[\frac{1}{2880} |f^{(iv)}(0)| \bigg] h^5 + O(h^6).
$$

2. *Composite Integration:*

$$
\int_0^h f(x)dx \approx \sum_{j=0}^{9} \int_{j\Delta x}^{(j+1)\Delta x} f(x)dx,
$$

with total composite error of,

$$
\text{error}_c = C\Delta x^4 + O(\Delta x^5).
$$

If we compare the two errors, and put the *composite integration error* into terms of the other, i.e., $\Delta x = \frac{h}{10}$, we find that

$$
\text{error}_c = C\Delta x^4 + O(\Delta x^5) = C\bigg(\frac{h}{10}\bigg)^5 + O(h^5),
$$

and hence our error has decreased by a factor of $\frac{1}{10^5}$! Well, *almost.* We still have that pesky constant, C, we so hand-wavingly used in the above error derivation for the *composite error.*

The three things that are important to note for composite integration are,

1. If your *local* integration stencil is n^{th} order accurate, if you use that stencil for a composite integration it will be $(n-1)^{st}$ accurate.
2. Performing composite integration over a whole interval, i.e., partitioning the integration bound into smaller intervals and integrating over them, will decrease the error by a larger factor.
3. We've only done examples using Newton-Cotes methods. This is because subdividing intervals and finding new quadrature points in the interval is easier than doing the same thing with Gaussian Quadrature. It can be done, but takes a little extra work, but the big picture idea of composite integration are identical.

We will now shift gears and talk about how to wring out every last bit we can from a Newton-Cotes methods, that is, bumping up the error by combining an idea of composite integration and *Richardson Extrapolation* to bump up the order of accuracy of a method. This is called *Romberg Integration.*

2.4 Romberg Integration

Romberg Integration, i.e., using an easy-to-derive-not-so-high-order-Newon-Cotes-integration-stencil-and-mathematically-manipulating-in-a-clever-way-to-boost-accuracy, is an example of a so-called *boot-strapping* scheme in mathematics.

The long (and short of it) is that we will take a modest Newton-Cotes stencil from $[0,h]$ and use a composite method over $[0,h/2]$ and $[h/2,h]$ and combine them in such a way that it eliminates the leading order error term to boost the method so it's granted VIP status to hang out with the more exquisite stencils in higher order land.

Again let's consider the integral,

$$I = \int_0^h f(x)dx,$$

and define our Newton-Cotes method on the entire integral as $\tilde{I}(h)$, and the composite as $\tilde{I}\left(\frac{h}{2}\right)$, i.e.,

$$\tilde{I}(h) = \sum_{j=0}^{n-1} w_j f(j\Delta x_1), \quad \text{with} \;\; \Delta x_1 = \frac{h}{N}$$

$$\tilde{I}\left(\frac{h}{2}\right) = \sum_{j=0}^{n-1} w_j f(j\Delta x_2) + \sum_{k=0}^{n-1} w_k f\left(\frac{h}{2} + k\Delta x_2\right), \quad \text{with} \;\; \Delta x_2 = \frac{(h/2)}{N},$$

and hence we have the following two error formulae,

$$e_h = I - \tilde{I}(h) = Ch^{n+1} + O\left(h^{n+2}\right)$$

$$e_{h/2} = I - \tilde{I}\left(\frac{h}{2}\right) = C\left(\frac{h}{2}\right)^{n+1} + O\left(h^{n+2}\right).$$

If we find the difference between the above two equations, we find

$$\tilde{(}h) - \tilde{I}\left(\frac{h}{2}\right) = Ch^{n+1}\left(1 - \frac{1}{2^{n+1}}\right),$$

and hence

$$C = \frac{1}{h^{n+1}}\left(\frac{\tilde{I}(h) - \tilde{I}\left(\frac{h}{2}\right)}{1 - \frac{1}{2^{n+1}}}\right) + O(h).$$

Therefore substituting C into the formulae for $\tilde{I}(h)$, we get

$$\tilde{I}(h) = \int_0^h f(x)dx + \left[\frac{1}{h^{n+1}}\left(\frac{\tilde{I}(h) - \tilde{I}\left(\frac{h}{2}\right)}{1 - \frac{1}{2^{n+1}}}\right) + O(h)\right] h^{n+1} + O\left(h^{n+2}\right),$$

and it is now clear that we can make a new method,

$$\hat{\tilde{I}}(h) = \tilde{I}(h) - \left(\frac{\tilde{I}(h) - \tilde{I}\left(\frac{h}{2}\right)}{1 - \frac{1}{2^{n+1}}} \right) = \int_0^h f(x)\,dx + O\big(h^{n+2}\big).$$

We find that the new method we defined by $\hat{\tilde{I}}(h)$ is $O\big(h^{n+2}\big)$, even though the stencils used were of order $O\big(h^{n+1}\big)$!

Furthermore we can *boot-strap* this method. Since $\hat{\tilde{I}}(h)$ is order $O\big(h^{n+2}\big)$, we can do the same procedure to define another method $\hat{\hat{\tilde{I}}}(h)$ that is $O\big(h^{n+3}\big)$, i.e.,

$$\hat{\hat{\tilde{I}}}(h) = \hat{\tilde{I}}(h) - \left(\frac{\hat{\tilde{I}}(h) - \hat{\tilde{I}}\left(\frac{h}{2}\right)}{1 - \frac{1}{2^{n+1}}} \right) = \int_0^h f(x)\,dx + O\big(h^{n+2}\big).$$

Moreover, we could continue boot-strapping this method above if we so desired. And...checkmate!

We will now briefly describe *Richardson Extrapolation* in general, of which *Romberg Integration* is one example.

2.4.1 Richardson Extrapolation

Consider some stencil that approximates a function value, derivative value, or integral, for a step-size h. Let's call that stencil approximation $\tilde{F}(h)$ for approximating what we really care about, call it F_{exact}.

Since $\tilde{F}(h)$ is an approximation, it has a leading order error, i.e.,

$$\tilde{F}(h) = F_{\text{exact}} + ch^k + O\big(h^r\big),$$

where $r > k$. Note that we assume that proper error analysis was performed on $\tilde{F}(h)$ so that both k and r are known. Now consider that we have used the stencil and computed $\tilde{F}$ for two step-sizes, say $\tilde{F}(h)$ and $\tilde{F}\big(\frac{h}{q}\big)$, for $q \in \mathbb{Z}^+$. We then obtain,

$$\tilde{F}(h) = F_{\text{exact}} + ch^k + O\big(h^r\big)$$

$$\tilde{F}\left(\frac{h}{q}\right) = F_{\text{exact}} + ch^k + O\big(h^r\big),$$

which can be viewed as a system of 2 equations and 2 unknowns, c and F_{exact}. This system should have a very similar feel to the one we encountered in *Romberg Integration*. Solving this we find that

$$F_{\text{exact}} = \tilde{F}(h) + \frac{\tilde{F}(h) - \tilde{F}(h/q)}{q^{-p} - 1} + O\big(h^r\big),$$

and hence our new approximation value, $\hat{\tilde{F}}(h)$ is,

$$\hat{\tilde{F}}(h) = \tilde{F}(h) + \frac{\tilde{F}(h) - \tilde{F}(h/q)}{q^{-p} - 1},$$

which is higher order than our original approximation, $\tilde{F}(h)$. We also brag that in most circumstances, as in the case for *Romberg Integration*, Richardson Extrapolation schemes can be *boot-strapped* to achieve higher accuracy.

2.4.2 Romberg Integration

Let's proceed in explicitly showing how boost accuracy for integration schemes using *Romberg Integration*. We are going to assume we are using a Newton-Cotes method, for simplicity in choosing quadrature points, without any other fuss.
We define the following integration stencil,

$$I_h: \text{ composite method on } [A,B] \text{ with step, h} = \int_A^B f(x)dx = Ch^k + \mathcal{O}(h^{k+1}).$$

Hence the largest error term is $\mathcal{O}(h^k)$. Next we consider, $I_{h/2}$, that is, the same stencil but done in a composite scheme to break the interval into two segments with length $h/2$ each, e.g.,

$$I_{h/2} = \int_A^B f(x)dx + C\left(\frac{h}{2}\right)^k + \mathcal{O}(h^{k+1}).$$

Now we substrate $I_{h/2}$ from I_h,

$$I_h - I_{h/2} = Ch^k - C\left(\frac{h}{2}\right)^k + \mathcal{O}(h^{k+1})$$

$$= Ch^k\left(1 - \frac{1}{2^k}\right) + \mathcal{O}(h^{k+1}).$$

Now solving for C, we find that

$$C = \frac{1}{h^k}\frac{I_h - I_{h/2}}{1-\frac{1}{2^k}} + \frac{\mathcal{O}(h^{k+1})}{h^k},$$

and hence

$$C = \frac{1}{h^k}\frac{I_h - I_{h/2}}{1-\frac{1}{2^k}} + \mathcal{O}(h),$$

and see that C is 1^{st}-order accurate. Now we will approximate the error,

$$I_h - Ch^k = I_h - \frac{I_h - I_{h/2}}{1-\frac{1}{2^k}} = \int_A^B f(x)dx + Dh^{k+1} + \mathcal{O}(h^{k+2}).$$

Therefore we see that the leading error term is $\mathcal{O}(h^{k+1})$, even though I_h itself is only $\mathcal{O}(h^k)$. We have successfully boosted the accuracy by one order! For completion, let's define a new integration method, call it $\tilde{I}_h$ and define is as,

$$\tilde{I}_h = I_h - \frac{I_h - I_{h/2}}{1-\frac{1}{2^k}} = \int_A^B f(x)dx + \mathcal{O}(h^{k+1}).$$

This is our new integration method. However, the fun does not need to stop there! We can define another integration method, call it $\tilde{\tilde{I}}_h$,

$$\tilde{\tilde{I}}_h = \tilde{I}_h - \frac{\tilde{I}_h - \tilde{I}_{h/2}}{1-\frac{1}{2^k}} = \int_A^B f(x)dx + \mathcal{O}(h^{k+2}),$$

that has higher-order accuracy! In principle, we could continue doing this for as long as we'd like, but will stop here to save some paper.