Richardson Extrapolation Examples

Type 1 Error

In this case, assume that we have a quantity Q being approximated by an algorithm that depends on h, call it $F(h)$, and the error is of the form:

$$Q = F(h) + K_1 h + K_2 h^2 + K_3 h^3 + \dots$$

The idea is to cut the h in half and recombine the numbers $F(h)$ and $F(h/2)$ to get an even better approximation, one that is $O(h^2)$. Define $F_1(h) \doteq F(h)$.

$$\begin{array}{rlll} Q & = F(h)+ & K_1 h+ & K_2 h^2 + K_3 h^3 + \dots \\ Q & = F(h/2)+ & K_1 \tfrac{h}{2}+ & K_2 \tfrac{h^2}{4} + K_3 \tfrac{h^3}{8} + \dots \end{array}$$

so that multiplying the lower equation by 2 and subtracting the upper equation:

$$\begin{array}{rlll} -Q & = -F(h)- & K_1 h- & K_2 h^2 - K_3 h^3 + \dots \\ 2Q & = 2F(h/2)+ & K_1 h+ & K_2 \tfrac{h^2}{2} + K_3 \tfrac{h^3}{4} + \dots \\ \hline Q & = 2F(h/2) - F(h) & & -\tfrac{h^2}{2} K_2 - \tfrac{3}{4} K_3 h^3 + \dots \end{array}$$

Let $F_2(h) = 2F(h/2) - F(h)$. Note that $Q = F_2(h) + O(h^2)$. We can extrapolate again:

$$\begin{array}{rlll} Q & = F_2(h)- & \tfrac{h^2}{2} K_2- & \tfrac{3h^3}{4} K_3 + \dots \\ Q & = F_2(h/2)- & \tfrac{h^2}{8} K_2- & \tfrac{3h^3}{32} K_3 + \dots \end{array}$$

Multiply the bottom equation by 4 and subtract the top:

$$\begin{array}{rlll} -Q & = -F_2(h)+ & \tfrac{h^2}{2} K_2+ & \tfrac{3h^3}{4} K_3 + \dots \\ 4Q & = 4F_2(h/2)- & \tfrac{h^2}{2} K_2- & \tfrac{3h^3}{8} K_3 + \dots \\ \hline 3Q & = 4F_2(h/2) - F_2(h) & & +O(h^3) \end{array}$$

So now $F_3(h) = \dfrac{4F_2(h/2) - F_2(h)}{3}$ is an $O(h^3)$ approximation to Q. We could continue to get the pattern:

$$F_{n+1}(h) = \frac{2^n F_n(h/2) - F_n(h)}{2^n - 1}$$

and this is an order $O(h^{n+1})$ approximation to Q.

For example, if $F(h) = \dfrac{f(x+h)-f(x)}{h}$, then the error fits this type (and $Q = f'(x)$). We can now create a table to perform the extrapolation:

$$\begin{array}{llll}
F_1(h) & & & \\
F_1\left(\frac{h}{2}\right) & F_2(h) = \frac{2F_1(h/2)-F_1(h)}{1} & & \\
F_1\left(\frac{h}{4}\right) & F_2\left(\frac{h}{2}\right) = \frac{2F_1(h/4)-F_1(h/2)}{1} & F_3(h) = \frac{2^2F_2(h)-F_2(h/2)}{2^2-1} & \\
F_1\left(\frac{h}{8}\right) & F_2\left(\frac{h}{4}\right) = \frac{2F_1(h/8)-F_1(h/4)}{1} & F_3\left(\frac{h}{2}\right) = \frac{2^2F_2(h/4)-F_2(h/2)}{2^2-1} & F_4(h) = \frac{2^3F_3(h/2)-F_3(h)}{2^3-1}
\end{array}$$

In Matlab, let F be a column vector containing $F_1(h), \ldots, F_1(h/2^j)$. Then we can build the table of interpolated values:

```
n=length(F);
for col=2:n
    for row=col:n
       Q(row,col)=(2^(col-1)*Q(row,col-1)-Q(row-1,col-1))/(2^(col-1)-1);
    end
end
```

Here’s a full script file using $f(x) = xe^x$ at $x = 1$. Run it and see how good the extrapolation can get!

```
f=inline(’x.*exp(x)’);
h=0.1;
x=1;
for j=1:9
    H=h/2^(j-1);
    F(j)=(f(x+H)-f(x))/H;
end

n=length(F);
Q=zeros(n,n);
Q(:,1)=F(:);

for col=2:n
    for row=col:n
       Q(row,col)=(2^(col-1)*Q(row,col-1)-Q(row-1,col-1))/(2^(col-1)-1);
    end
end

%The solution is 2e, so subtract that from everything to see the errors:
Error=diag(abs(Q-2*exp(1)));
```

Error of Type 2

In this case, assume that we have a quantity Q being approximated by an algorithm that depends on h, call it $F(h)$, and the error is of the form:

$$Q = F(h) + K_1 h^2 + K_2 h^4 + K_3 h^6 + \dots$$

We can perform a similar analysis to what we did previously and get a better approximation to Q. Again, define $F_1(h) = F(h)$.

$$\begin{array}{lllll} Q & = F_1(h) + & K_1 h^2 + & K_2 h^4 + K_3 h^6 + \dots \\ Q & = F_1(h/2) + & K_1 \frac{h^2}{4} + & K_2 \frac{h^4}{2^4} + K_3 \frac{h^6}{2^6} + \dots \end{array}$$

so that multiplying the lower equation by 2^2 and subtracting the upper equation:

$$\begin{array}{rlll} -Q & = -F_1(h) - & K_1 h^2 - & K_2 h^4 - K_3 h^6 + \dots \\ 2^2 Q & = 2^2 F_1(h/2) + & K_1 h^2 + & K_2 \frac{h^4}{2^2} + K_3 \frac{h^3}{2^4} + \dots \\ \hline (2^2 - 1) Q & = 2^2 F_1(h/2) - F_1(h) & & -\frac{3h^4}{4} K_2 - \frac{15}{16} K_3 h^6 + \dots \end{array}$$

Define $F_2(h) = \frac{2^2 F_1(h/2) - F_1(h)}{2^2 - 1}$. Then $F_2(h)$ is an $O(h^4)$ approximation to Q.
If we continue, we would find the pattern:

$$F_{n+1}(h) = \frac{4^n F_n(h/2) - F_n(h)}{4^n - 1}$$

and this gives an $O(h^{2(n+1)})$ approximation to Q.

EXAMPLE: The derivative formula:

$$F(h) = \frac{f(x+h) - f(x-h)}{2h}$$

has an error of the type discussed here. To be more specific, let $f(x) = xe^x$, and we'll approximate the derivative at $x = 2$ starting with $h = 0.2$.
Here is a Matlab script that will perform the extrapolation described here. Check out how fast the error decreases!

```
f=inline('x.*exp(x)');
h=0.2;
x=2;
for j=1:4
    H=h/2^(j-1);
    F(j)=(f(x+H)-f(x-H))/(2*H);
end

n=length(F);
Q=zeros(n,n);
Q(:,1)=F(:);

for col=2:n
    for row=col:n
       Q(row,col)=(4^(col-1)*Q(row,col-1)-Q(row-1,col-1))/(4^(col-1)-1);
    end
end
%The solution is 3e^2, so subtract that from everything to see the errors:
Error=diag(abs(Q-3*exp(2)));
```