Adaptive Simpson

We now want to find an approximation $Q(a, b)$ to the integral $I(a, b) = \int_a^b f(x)dx$ so that the error is below a given tolerance ϵ, thus

$$|I(a, b) - Q(a, b)| \lesssim \epsilon,$$

To do this, we will need a strategy for

- estimating the error $I(a, b) - Q(a, b)$.
- Divivde the interval $[a, b]$ interval into subintervals $a = X_0 < X_1 < \cdots < X_M = b$ so that
$$|I(X_k, X_{k+1}) - Q(X_k, X_{k+1})| \lesssim \frac{X_{k+1} - X_k}{b - a}\,\epsilon.$$

And we will use Simpsons method as an example here, but the strategy is applicable for other methods as well.

Error estimate

Simpsons formula is given by

$$S(a, b) = \frac{b - a}{6}\Big(f(a) + 4f(c) + f(b)\Big), \qquad c = \frac{a + b}{2}.$$

and the error is given by

$$I(a, b) - S(a, b) = -\frac{(b - a)^5}{2880} f^{(4)}(\xi), \qquad \xi \in (a, b).$$

Assume now that the interval $[a, b]$ is small, so that $f^{(4)}$ is almost constant over this interval. In that case, the formula above becomes

$$I(a, b) - S(a, b) \approx C(b - a)^5, \qquad C \approx -\frac{f^{(4)}}{2880} \tag{1}$$

Next, let us split the interval in two equal parts: $[a, c]$ and $[c, b]$ and integrate each subinterval by Simpsons method:

$$S_2 = S(a, c) + S(c, b) = \frac{(b - a)}{2 \cdot 6}\Big(f(a) + 4f(d) + 2f(c) + 4f(e) + f(b)\Big), \qquad d = \frac{a + c}{2},\ e = \frac{c + b}{2}$$

and the error is

$$I(a, b) - S_2 = -\frac{(b - a)^5}{2^5 \cdot 2880} f^{(4)}(\xi_1) - \frac{(b - a)^5}{2^5 \cdot 2880} f^{(4)}(\xi_2) \approx C \cdot \frac{1}{2^4}(b - a)^5 \tag{2}$$

For notational convenience, use $S_1 = S(a, b)$. Multiplying (2) by 16, and subtract (1) from it gives

$$15 \cdot I(a, b) - 16S_2 + S_1 \approx 0 \qquad \Rightarrow \qquad I(a, b) \approx S_2 + \frac{S_2 - S_1}{15}. \tag{3}$$

So, we can of course use this as a better approximation to the integral $I(a,b)$, but we can also use it for error estimations of S_1 and S_2, since clearly

$$I(a,b) - S_1 \approx E_1 = \frac{16}{15}(S_2 - S_1) \tag{4}$$

$$I(a,b) - S_2 \approx E_2 = \frac{1}{15}(S_2 - S_1) \tag{5}$$

In a code, we will use S_2, since this is the best of the two approximations and we do have an error estimate for it. But since this error estimate is known, we can add it to S_2 and get an even better approximation to I(a,b) for free, that is, we use the approximation (3). In that case the error is usually overestimated, we get a better solution than we asked for. Which is hardly any reason to complain.

Example 0.1. *We want to approximate*

$$I(0,\pi/2) = \int_0^{\pi/2} \sin(x)dx = 1.$$

We get

$$S_1 = S(0,\pi/2) = 1.00228088$$
$$S_2 = S(0,\pi/4) + S(\pi/4,\pi/2) = 1.00013458.$$

The error esimates are

$$E_1 = -2.28 \cdot 10^{-3}, \qquad E_2 = -1.43 \cdot 10^{-4}$$

The improved approximation is given by:

$$Q = S2 + E2 = 0.9999916.$$

So, even in this simple example where $f^{(4)}$ is far from constant over the interval, the error estimates are quite accurate.

Splitting of the interval

The algorithm is quite trivial: First, calculate S_1, S_2 and E_2 over the whole $[a, b]$. If $|E_2| \geq \epsilon$ then divide the interval in two equal parts, and repeat the procedure on each subinterval with half of the tolerance on each subinterval. Continue until the error is below the tolerance for the given subinterval. The strategy may be best illustrated by an example:

Example 0.2. *We want to approximate* $\int_0^{\pi/2} \sin x dx$ *with an error tolerance* $\epsilon = 10^{-6}$. *The results are given in the following table:*

<table>
<tr><td>level</td><td colspan="3">$a = 0,\ b = \pi/2,\ \epsilon = 10^{-5}$</td></tr>
<tr><td>1</td><td colspan="3">$S_1 = 1.00228088, S_2 = 1.00013458, |E_2| = 1.43 \cdot 10^{-4}$</td></tr>
<tr><td>2</td><td>$a = 0,\ b = \pi/4,\ \epsilon = 5 \cdot 10^{-6}$</td><td colspan="2">$a = \pi/4,\ b = \pi/2,\ \epsilon = 5 \cdot 10^{-6}$</td></tr>
<tr><td></td><td>$S_1 = 0.29293264,\ S_2 = 0.29289564$
$|E_2| = 2.47 \cdot 10^{-6}, Q(0, \pi/4) = 0.29289318$</td><td colspan="2">$S_1 = 0.70720195,\ S_2 = 0.70711265$
$|E_2| = 5.96 \cdot 10^{-6}$</td></tr>
<tr><td>3</td><td></td><td>$a = \pi/4,\ b = 3\pi/8$
$\epsilon = 2.5 \cdot 10^{-6}$</td><td>$a = 3\pi/8,\ b = \pi/2$
$\epsilon = 2.5 \cdot 10^{-6}$</td></tr>
<tr><td></td><td></td><td>$S_1 = 0.32442604$
$S_2 = 0.32442352$
$|E_2| = 1.68 \cdot 10^{-7}$
$Q(\pi/2, 3\pi/4) = 0.32442335$</td><td>$S_1 = 0.38268661$
$S_2 = 0.38268363$
$|E_2| = 1.98 \cdot 10^{-7}$
$Q(3\pi/4, \pi/2) = 0.38268343$</td></tr>
</table>

and the numerical approximation is

$$Q(0, \pi/4) + Q(\pi/4, 3\pi/8) + Q(3\pi/4, \pi/2) = 0.9999996.$$

The error is about $4 \cdot 10^{-8}$ *which is well below the tolerance.*

The implementation of the code is given in Figure 1.

```
function simpson_result = simpson(f,a,b,tol,level,level_max)
%
% simpson_result = simpson(f,a,b,tol,0,level_max)
%
% Compute the integral of a function f from a to be within a tolerance tol,
% using an adaptiv Simpson method.
%
% level_max=10 is a suitable value.
%
level = level+1;
h = b-a;
c = (a+b)/2;
S1 = h*(f(a)+4*f(c)+f(b))/6;
d = (a+c)/2;
e = (c+b)/2;
S2 = h*(f(a)+4*f(d)+2*feval(f,c)+4*f(e) + f(b))/12;
if level>=level_max
    simpson_result = S2;
    warning('Maximum␣level␣reached');
else
    err = (S2-S1)/15;
    if abs(err)<tol
        simpson_result = S2+err;
    else
        left_simpson = simpson(f,a,c,tol/2,level,level_max);
        right_simpson = simpson(f,c,b,tol/2,level,level_max);
        simpson_result = left_simpson+right_simpson;
    end
end
```

Figure 1: Matlab code for adaptive Simpson