Assignment #6
Numerical Computing (COMP 350)

Guillaume Labranche (260585371)

due on 30 November 2015

See `script.m` for the script aggregating the results of all questions.

1. (a) erf(3) was computed to be 0.99997550396668711 in 17 function evaluations ($2^4 = 16$ subintervals).
   See functions `erf_rtr.m` and `rtr.m` for the MATLAB implementation.

   (b) erf(3) was computed to be 0.99997747287883731 in 41 function evaluations. I was able to save function evaluations by passing $f(a)$, $f(b)$ and $f(c)$ to the next recursion level.
   See functions `erf_asm.m` and `asm.m` for the MATLAB implementation.

2. (a) First, we must map the x-values from $x \in [a, b]$ to $u \in [-1, 1]$:

$$u = \left(x - \frac{a+b}{2}\right)\frac{1-(-1)}{b-a}$$
$$\frac{b-a}{2}u + \frac{a+b}{2} = x$$
$$\frac{b-a}{2}du = dx$$

   We use $-\frac{a+b}{2}$ to center around 0 and multiply by $\frac{2}{b-a}$ to scale the range.
   We can now use our Gaussian two-point quadrature rule:

$$\begin{aligned}\int_a^b f(x)dx &= \frac{b-a}{2} \cdot \int_{-1}^{1} f(\tfrac{b-a}{2}u + \tfrac{a+b}{2})du \\ &\approx \frac{b-a}{2} \cdot \Big(f(\tfrac{b-a}{2} \cdot \tfrac{-\sqrt{3}}{3} + \tfrac{a+b}{2}) + f(\tfrac{b-a}{2} \cdot \tfrac{\sqrt{3}}{3} + \tfrac{a+b}{2})\Big)\end{aligned}$$

   Then we divide the interval $[a, b]$ into n equal subintervals $[x_i, x_{i+1}]$, $i = 0, 1, \ldots, n-1$. Let $h = \frac{b-a}{n}$:

$$\begin{aligned}\int_a^b f(x)dx &= \sum_{i=0}^{n-1} \int_{h\cdot i+a}^{h\cdot(i+1)+a} f(x)dx \\ &\approx \frac{h}{2}\sum_{i=0}^{n-1}\Big(f\big(\tfrac{h}{2}\cdot\tfrac{-\sqrt{3}}{3} + \tfrac{h(2i+1)}{2} + a\big) + f\big(\tfrac{h}{2}\cdot\tfrac{\sqrt{3}}{3} + \tfrac{h(2i+1)}{2} + a\big)\Big) \\ &= \frac{h}{2}\sum_{i=0}^{n-1}\Big(f\big(h\cdot i + \tfrac{h}{2} - \tfrac{h}{2\sqrt{3}} + a\big) + f\big(h\cdot i + \tfrac{h}{2} + \tfrac{h}{2\sqrt{3}} + a\big)\Big)\end{aligned}$$

   (b) Because in question 1.(a) $m = 16$, $n = \left\lfloor \frac{17}{2} \right\rfloor = 8$. There are $2n = 16$ function evaluations. The value computed is 0.99997801381341089.
   See functions `erf_cgtpq.m` and `cgtpq.m` for the MATLAB implementation.
   When comparing the percent error of both results with respect to the correct value 0.99997790950300136 (provided by the MATLAB built-in function `erf(3)`), we find that the composite Gaussian two-point quadrature rule yields a result 23 times closer to the correct value than the recursive trapezoid rule while using 1 less function evaluation. A significant improvement to say the least!