

Adaptive Simpson Integration (Quadrature)

This *Numerit* program (demoint1) demonstrates how to numerically integrate a user-defined analytic function of the form $y = f(x)$ using an adaptive Simpson solution method. The numerical method used in this function (asimpson) is based on the algorithm described in Chapter 2 and Part V of *Numerical Computing*, L. F. Shampine and R. C. Allen, Jr., (1973), W. B. Saunders Company.

This function numerically integrates a user-defined analytic function of the form

$$S(x) = \int_A^B f(x) dx \quad (1)$$

over the lower and upper limits A and B specified by the user.

The syntax and parameter list of this *Numerit* function are as follows:

```
function asimpson (a, b, acc, ssum, esterr, iflag)
` adaptive simpson quadrature routine
` input
`  a   = lower integration limit
`  b   = upper integration limit
`  acc = accuracy
` output
`  ssum = integral from a to b
`  esterr = relative error
`  iflag = error flag
`      1 => no error
`      2 => more than 30 levels
`      3 => subinterval too small
`      4 => more than 2000 function evaluations
```

The analytic function must be coded by the user in a function with the following syntax:

```
function usrfunc(x, y)
```

In the parameter list, x is the function argument and y is the function value at x . The actual user-defined function can have any *Numerit* compatible name. The main driver program can *redirect* Numerit to use the user's function with a statement similar to `usrfunc -> func1`.

This demonstration program numerically integrates the following function

$$y = f(x) = e^{-x^2} \quad (2)$$

Numerical Analysis with Numerit

over a lower and upper integration limit specified by the user.
For this example the user-defined *Numerit* source code is

```
function func1(x, y)
  y = exp(-x * x)
```

The following is a typical draft output created with this demonstration program.

```
program demoint1
< adaptive Simpson quadrature method >

integral value = 0.746824132731378
error estimate = 1.09250394445245e-10
error flag     = 1
```

For this example the lower limit was 0 and the upper limit was 1. The convergence criterion was 1.0e-8.