4 Integration of Functions

Functions:

trapzd: integrate a function using the trapezoidal rule
qtrap: use the trapezoidal rule to desired accuracy
qsimp: use Simpson's rule to desired accuracy
qromb: use Romberg adaptive quadrature to desired accuracy
midpnt: the extended midpoint rule
qromo: use open Romberg to desired accuracy
midinf: function on a semi infinite interval
midsql: function with a square root singularity
midsqu: inverse square root singularity
midexp: increased exponentially
qgaus: Gaussian quadratures
gauleg: compute Gauss-Legrendre weights and abscissas
quadf: integrate a function of n variables returning a function of all but the last variable
quad2d: two dimensional integrals
quad3d, quad4d: 3 and 4 dimensional integrals

---

trapzd

Description:

This routine computes the nth stage of refinement of an extended trapezoidal rule. Func is input as the name of a function to be integrated between a and b, input in the form of a list $[a, b]$. When called with $n = 1$ the function returns the crudest estimate of the integral. Subsequent calls will improve the accuracy by adding at the nth stage 2^{n-1} additional points.

trapzd(func, int, n) $\longrightarrow$ **s**

Where:

func is the name of a function of one variable to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration,
n is a positive integer with initial value 1,
s a floating point number representing an approximation to the integral.

Example:

In this example the integral

$$\int_0^{\frac{\pi}{2}} x^2(x^2-2)sin(x)dx$$

is computed using 10 calls to `trapzd` and the result compared with the true integral.

```
-> f(x) := x^2*(x^2-2)*sin(x);
-> g(x) := 4*x*(x^2-7)*sin(x) - (x^4-14*x^2+28)*cos(x);
-> ab:[0,%pi/2];
-> for n from 1 by 1 to 10 do print(trapzd(f, ab, n));
0.9057729897325448d0
-0.020944909148688295d0
-0.3614612746083103d0
-0.44958372807768d0
-0.47175628988830026d0
-0.4773076434726939d0
-0.4786959850722389d0
-0.479043101764932d0
-0.47912988289145086d0
-0.4791515782951273d0
-> eval(g(float(%pi/2)) - g(0d0))?
-0.47915881010719374d0
```

qtrap

Description:

This function returns the integral of a function over an interval using the trapezoidal rule.

qtrap(func, int) $\longrightarrow$ **s**

Where:

func is the name of a function of one variable to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration.
s a floating point number representing an approximation to the integral.

Pre-variables:

qtrap_eps [1.0d-6] a real number representing the desired fractional accuracy,
qtrap_jmax [20] is a whole number such that 2^{jmax-1} is the maximum allowed number of steps.

Example:

In this example successive calls are made internally to the trapezoidal rule to compute the integral

$$\int_0^{\frac{\pi}{2}} x^2(x^2-2)sin(x)dx$$

to an accuracy specified by `qtrap_eps`.

```
-> f(x) := x^2*(x^2-2)*sin(x);
-> ab:[0,%pi/2];
-> qtrap(f, ab) ?
-0.4791586971100573d0
```

qsimp

Description:

This function returns the integral of a function over an interval using Simpson's Rule.

qsimp(func, int) $\longrightarrow$ s

Where:

func is the name of a function of one variable to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration,
s a floating point number representing an approximation to the integral.

Pre-variable:

qsimp_eps [1.0d-6] a real number representing the desired fractional accuracy.

Example:

In this example Simpson's Rule is used to compute the integral

$$\int_0^{\frac{\pi}{2}} x^2(x^2-2)sin(x)dx.$$

```
-> f(x) := x^2*(x^2-2)*sin(x);
-> ab:[0,%pi/2];
-> qsimp(f, ab) ?
-0.479158807329163d0
```

qromb

Description:

This function returns the integral of a function over an interval using Romberg's method of order twice the value of the pre-variable **qromb_k**, where order 4 corresponds to Simpson's Rule.

$$\textbf{qromb(func, int)} \longrightarrow \textbf{ss}$$

Where:

func is the name of a function of one variable to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration,
ss a floating point number representing an approximation to the integral.

Pre-variables:

qromb_eps [1.0d-6] a real number being the fractional accuracy required,
qromb_jmax [20] an integer upper bound for the total number of steps,
qromb_jmaxp [21] an integer being **1+qromb_jmax**,
qromb_k [5] the number of points used in the extrapolation,
qromb_km [4] an integer **qromb_k-1**.

Example:

In this example the function $f(x) = x^2(x^2 - 2)sin(x)$ is integrated over the interval $[0, \frac{\pi}{2}]$ using `qromb`. The function $k(x) = x^4 ln(x + sqrt(x^2 + 1))$ is then integrated over [0,2] using `qromb`, `qtrap`, `qsimp` and `trapzd` so that a time comparison might be made of the alternative integration methods.

```
-> f(x) := x^2*(x^2-2)*sin(x);
-> qromb(f, ab) ?
-0.479158823970037950
-> k(x) := x^4*ln(x+sqrt(x^2+1));
-> qromb(k, [0,2])?
8.153364369647917d0
-> time?
[66.69922, 0.0]
-> trapzd(k, [0,2],1)?
23.098167602860965d0
-> time?
[10.0, 0.0]
-> qsimp(k, [0,2]) ?
8.153364159059803d0
-> time?
[118.30078, 0.0]
-> qtrap(k, [0,2])?
8.153365179815244d0
-> time?
[3261.7031, 0.0]
```

midpnt

Description:

This function returns the integral of a function over an interval using the extended mid-point rule. Each successive call to the function improves the accuracy of the previous estimate by adding $(2/3)3^{n-1}$ additional interior points for $n = 2, 3, \ldots$.

midpnt(func, int, n) $\longrightarrow$ **s**

Where:

func is the name of a function of one variable to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration,
n is a positive integer being the order of the call with $n = 1$ for the first,
s a floating point number representing an approximation to the integral.

Example:

Int this example one call is made to `midpnt` to compute an approximation to

$$\int_0^1 \frac{1}{\sqrt{x}} dx.$$

```
-> f(x):= 1/sqrt(x);
-> midpnt(f, [0,1],1)?
1.414213562373095d0
```

qromo

Description:

Romberg integration on an open interval using a specified integrating function whose name is the value of the pre_variable **qromo_chose**.

qromo(func, int) $\longrightarrow$ **s**

Where:

func is the name of a function of one variable to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration,
s a floating point number representing an approximation to the integral.

Pre-variables:

qromo_chose [`"midpnt"`] a string specifying an integrating function being one of the list `"midpnt"`, `"midinf"`, `"midsql"`, or `"midsqu"`,
qromo_eps [1.0d-6] a real number being the fractional accuracy required,
qromo_jmax [14] an upper bound for the total number of steps,
qromo_jmaxp [15] **1+qromo_jmax**,
qromo_k [5] the number of points used in the extrapolation,
qromo_km [4] **qromo_k-1**.

Example:

The function $f(x) = \frac{\sqrt{x}}{sin(x)}$ is integrated on $[0, \frac{\pi}{2}]$ using the integrating function `midsql`.

```
-> f(x):= sqrt(x)/sin(x);
-> qromo_choose:"midsql";
-> qromo(f, [0, %pi/2])?
2.75314193300339993d0
-> time?
[118.39844, 0.0]
```

midinf

Description:

This function returns the integral of a function over an interval using the extended mid-point rule. Each successive call to the function improves the accuracy of the previous estimate by adding $(2/3)3^{n-1}$ additional interior points for $n = 2, 3, ...$ at evenly spaced points in $1/x$. The endpoints of the iterval of integration must both have the same sign. Either the left endpoint may be arbitrarily large and negative or the right endpoint arbitrarily large and positive.

$$\textbf{midinf(func, int)} \longrightarrow \mathbf{s}$$

Where:

func is the name of a function of one variable to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration,
s a floating point number representing an approximation to the integral.

Example:

The function $f(x) = \frac{sin(x)}{x}$ is integrated over $[\frac{\pi}{2}, \infty)$ using `qromo` and the integrating function `midinf`.

```
-> inf:1.0d20;
-> f(x) := sin(x)/x^2;
-> qromo_choose:"midinf";
-> qromo(f, [%pi/2, inf])?
0.1643603661876054d0
-> qromo(f, [-inf, -%pi/2])?
-0.16436035686134282d0
```

midsql

Description:

This function returns the integral of a Senac function allowing for an inverse sqare root singularity in the integrand at the lower limit of integration.

midsql(func, int) $\longrightarrow$ **s**

Where:

func is the name of a function of one variable to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration,
s a floating point number representing an approximation to the integral.

Example:

The function $f(x) = \frac{\sqrt{x}}{sin(x)}$ is integrated over $[\frac{\pi}{2}, \infty)$ using `qromo` and the integrating function `midsql`.

```
-> f(x):= sqrt(x)/sin(x);
-> qromo_choose:"midsql";
-> qromo(f, [0, %pi/2])?
2.7531419330033993d0
-> time?
[118.39844, 0.0]
```

midsqu

Description:

This function returns the integral of a Senac function allowing for an inverse square root singularity in the integrand at the upper limit of integration.

midsqu(func, int) $\longrightarrow$ s

Where:

func is the name of a function of one variable to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration,
s a floating point number representing an approximation to the integral.

Example:

The function $f(x) = \frac{\sqrt{x}}{sin(x)}$ is integrated over $[\frac{\pi}{2}, \pi)$ using `qromo` and the integrating function `midsqu`.

```
-> f(x):= sqrt(float(%pi)-x)/sin(x);
-> qromo_choose:"midsqu";
-> qromo(f, [%pi/2, %pi])?
2.753141933003379d0
-> time?
[126.703125, 0.0]
```

midexp

Description:

Returns the integral of a function allowing for a negative exponential decay and upper limit positive infinity in the integrand.

midexp(func, int) $\longrightarrow$ **s**

Where:

func is the name of a function of one variable to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration,
s a floating point number representing an approximation to the integral.

Example:

The function $f(x) = \frac{e^{-x}}{\sqrt{x}}$ is integrated over $[1, \infty)$ using `qromo` and the integrating function `midexp`.

```
-> inf:1.0d20;
-> f(x) := exp(-x)/sqrt(x);
-> qromo_choose:"midexp";
-> b:qromo(f, [1,inf])?
0.2788294694583553d0
-> time?
[118.30469, 0.0]
```

qgaus

Description:

Returns the integral of a function over an interval by a ten-point Gauss-Legendre integration: the function is evaluated exactly ten times at interior points of the interval.

qgaus(func, int) $\longrightarrow$ **ss**

Where:

func is the name of a function of one variable to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration,
ss a floating point number representing an approximation to the integral.

Example:

The function $g(x) = xe^{-x}$ is integrated over a set of intervals using `qgaus` and the result compared with the true value.

```
-> g(x) := x*exp(-x);
-> h(x) := -(1+x)*exp(-x);
-> for i from 1 by 1 to 5 do
        print([qgaus(g, [0,i]), h(float(i))-h(0d0)]);
[0.264240900783703d0, 0.264241117657115 33d0]
[0.5939937341701486d0, 0.5939941502901619d0]
[0.8008512007116191d0, 0.8008517265285442d0]
[0.9084211778513934d0, 0.9084218055563291d0]
[0.9595715585181745d0, 0.9595723180054871d0]
```

gauleg

Description:

Given an interval of integration and positive integer n this function returns a list of the abscissas and weights of the Guass-Legendre n-point quadrature formula.

gauleg(int, n) $\longrightarrow$ [x, w]

Where:

int is a real interval of the form $[x1, x2]$ being the interval of integration,
n is a positive integer being the order of the rule,
x is a floating point list representing the abscissas,
w is a floating point list representing the weights.

Example:

Lists of abscissas and weights for the interval [0,1] and order 5 are computed.

```
-> float_format(4,"f");
->  gauleg([0,1], 5)?
[[0.0469, 0.2308, 0.5000, 0.7692, 0.9531], [0.1185, 0.2393, 0.2844, 0.2393,
0.1185]]
```

quad2d

Description:

This function returns the integral of a function over a two dimensional region bounded by an interval on the first axis and by two functions for the second in an iterated integral.

quad2d(func, int, fafb) $\longrightarrow$ **ss**

Where:

func is the name of a function of two variables to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration,
fafb is a list of two functions of one variable of the form $[fa, fb]$, such that, for each x, the limits of integration are $[fa(x), fb(x)]$,
ss a floating point number representing an approximation to the integral.

Example:

The function $f(x, y) = x^2 + y^2$ is integrated over a set of circlar discs of increasing radius.

```
-> f(x,y):= x^2+y^2;
-> for xmax from 0.6d0 by 0.1d0 to 1.0d0 do
  program([],
     defev(fa, x, -sqrt(xmax^2-x^2)),
     defev(fb, x,  sqrt(xmax^2-x^2)),
     print(quad2d(f, [-xmax, xmax], [fa, fb])));
0.20376438158167923d0
0.37749867297655 22d0
0.64399607018407 25d0
1.03155718175725 06d0
1.57225603072283 28d0
```

quad3d

Description:

This function returns the integral of a function over a three dimensional region bounded by an interval on the first axis, by two functions of one variable for the second, and by two functions of two variables for the third in an interated integral.

quad3d(func, int, fafb, gagb) $\longrightarrow$ **ss**

Where:

func is the name of a function of two variables to be integrated,
int is a real interval of the form $[a, b]$ being the interval of integration,
fafb is a list of two functions of one variable of the form $[fa, fb]$, such that, for each x, the limits of integration are $[fa(x), fb(x)]$,
gagb is a list of two functions of two variables of the form $[ga, gb]$, such that, for each x,y , the limits of integration are $[ga(x, y), gb(x, y)]$,
ss a floating point number representing an approximation to the integral.

Example:

The function $f(x, y, z) = x^2 + y^2 + z^2$ is integrated over a set of 3-dimensional balls of increasing radius.

```
-> f(x,y,z):= x^2+y^2+z^2;
-> for xmax from 0.6d0 by 0.1d0 to 1.0d0 do
  program([],
     defev(fa, x, -sqrt(xmax^2-x^2)),
     defev(fb, x,  sqrt(xmax^2-x^2)),
     defev(ga, [x,y], -sqrt(xmax^2-x^2-y^2)),
     defev(gb, [x,y],  sqrt(xmax^2-x^2-y^2)),
     print(quad3d(f, [-xmax, xmax], [fa, fb], [ga, gb])));
0.19558280178723578d0
0.42273150072506066d0
0.8241843169964173d0
1.4852069010718212d0
2.515210928333793d0
```

quad4d

Description:

This function returns the integral of a function over a four dimensional region bounded by an interval on the first axis, by two functions of one variable for the second, by two functions of two variables for the third, and by two functions of three variables for the fourth in an interated integral.

quad4d(func, int, fafb, gagb, hahb) $\longrightarrow$ **ss**

Where:

func is the name of a function of two variables to be integrated.
int is a real interval of the form $[a, b]$ being the interval of integration,
fafb is a list of two functions of one variable of the form $[fa, fb]$, such that, for each x, the limits of integration are $[fa(x), fb(x)]$,
gagb is a list of two functions of two variables of the form $[ga, gb]$, such that, for each x, y , the limits of integration are $[ga(x, y), gb(x, y)]$,
hagh is a list of two functions of two variables of the form $[ha, hb]$, such that, for each x, y, z, the limits of integration are $[ha(x, y, z), hb(x, y, z)]$,
ss a floating point number representing an approximation to the integral.

Example:

The function $f(x, y, z, w) = x^2 + y^2 + z^2 + w^2$ is integrated over a 4-dimensional ball.

```
-> f(x,y,z,w):= x^2+y^2+z^2+w^2;
-> xmax:1d0;
->   program([],
     defev(fa, x, -sqrt(xmax^2-x^2)),
     defev(fb, x,  sqrt(xmax^2-x^2)),
     defev(ga, [x,y], -sqrt(xmax^2-x^2-y^2)),
     defev(gb, [x,y],  sqrt(xmax^2-x^2-y^2)),
     defev(ha, [x,y,z], -sqrt(xmax^2-x^2-y^2-z^2)),
     defev(hb, [x,y,z],  sqrt(xmax^2-x^2-y^2-z^2)),
     return(quad4d(f, [-xmax, xmax], [fa, fb], [ga, gb], [ha, hb])));
3.292049122509002d0
```