`intlib` — tools for integration

Table of contents

1 Introduction

This libary contains functions for manipulating and solving integrals. Currently there are only described interfaces for the well-known integration methods change of variables and integration by parts. In future versions more interfaces will be added.

2 First steps

Integration is the opposite process to differentiation. Any function `F` in the variable `x` with `diff(F,x)` = `f` is an integral of `f`.

```
>> f:=cos(x)*exp(sin(x))

                     cos(x) exp(sin(x))

>> F:=int(f,x)

                         exp(sin(x))

>> diff(F,x)

                     cos(x) exp(sin(x))
```

A constant is not added to the integral. With MuPAD it is possible to determine integrals of elementary functions, of many special functions and restricted of algebraic functions.

```
>> int(sin(x)^4*cos(x),x)

                               5
                         sin(x)
                         -------
                            5

>> int(1/(2+cos(x)),x)

                 1/2       /      sin(x)       \
              2 3    arctan| ----------------- |
        1/2                |           1/2     |
     x 3                   \ cos(x) + 3    + 2 /
     ------ - ----------------------------------
       3                       3

>> int(exp(-a*x^2),x)

                    1/2      1/2
                  PI    erf(a    x)
                  -----------------
                            1/2
                         2 a
```

```
>> int(x^2/sqrt(1-5*x^3),x)

                                 3
                              2 x
                              ---- - 2/15
                               3
                             -------------
                                     3 1/2
                             (1 - 5 x )

>> normal(simplify(diff(%,x)))

                                   2
                                  x
                             -------------
                                     3 1/2
                             (1 - 5 x )
```

It is also possible to compute definite and multiple integrals:

```
>> int(exp(-x^2)*ln(x)^2,x=0..infinity);

                  5/2     1/2                      2
                PI      PI    (- EULER - 2 ln(2))
                ----- + --------------------------
                 16                 8

>> int(sin(x)*dirac(x+2)-heaviside(x+3)/x,x=1..4)

                                -ln(4)

>> int(int(int(1, z=0..c*(1-x/a-y/b)), y=0..b*(1-x/a)), x=0..a)

                                 a b c
                                 -----
                                   6
```

3 Integration by parts and by change of variables

Typical applications for the rule of integration by parts

$$\int u'(x)v(x)\mathrm{d}x = u(x)v(x) - \int u(x)v'(x)\mathrm{d}x$$

are integrals of the form $\int p(x) * \cos(x)\mathrm{d}x$ where $p(x)$ is polynomial. Thereby one has to use the rule in the way that the polynomial is differentiated. Thus one has to choose $u'(x) = cos(x)$.

```
>> intlib::byparts(hold(int)((x-1)*cos(x),x),cos(x))

                 sin(x) (x - 1) - int(sin(x), x)
```

In particular with the ansatz $u'(x) = 1$ it is possible to compute a lot of the well-known standard integrals, like e.g. $\int \arcsin x \, \mathrm{d}x$.

```
>> intlib::byparts(hold(int)(arcsin(x),x),1)

                                 /      x         \
               x arcsin(x) - int| -----------, x |
                                 |       2 1/2    |
                                 \ (1 - x )       /
```

In order to determine the remaining integral one may use the method change of variable

$$\int f(g(x)) * g'(x)\mathrm{d}x = F(g(x)) + c$$

with $g(x) = 1 - x^2$.

```
>> F:=intlib::changevar(hold(int)(x/sqrt(1-x^2),x), t=1-x^2)

                          /      1       \
                       int| - ------, t |
                          |      1/2     |
                          \   2 t        /
```

Via backsubstition into the solved integral `F` one gets the requested result.

```
>> hold(int)(arcsin(x),x) = x*arcsin(x)-subs(eval(F),t=1-x^2)

                                                         2 1/2
          int(arcsin(x), x) = x arcsin(x) + (1 - x )
```

Applying change of variable with the integrator is problematic, since it may occur that the integrator will never terminate. For that reason this rule is used within the integrator only on certain secure places. On the other hand this may also lead to the fact, that some integrals cannot be solved directly.

```
>> f:= sin(x)*sqrt(1+sin(x)):
   int(f,x)

                                             1/2
                 int(sin(x) (sin(x) + 1)    , x)

>> subs(eval(intlib::changevar(hold(int)(f,x),t=sin(x))),t=sin(x))

                                     1/2 / 2 sin(x)       \
         (sin(x) - 1) (sin(x) + 1)      | -------- + 4/3 |
                                         \    3           /
         -----------------------------------------------
                                      2 1/2
                        (1 - sin(x) )
```

`intlib::byparts` – **performs integration by parts**

`intlib::byparts(integral, du)` performs on `integral` the integration by parts, where `du` is the part to be integrated.

Call(s):

- `intlib::byparts(integral, du)`

Parameters:

`integral`	—	integral: an expression of type `"int"` of the form `int(du*v, x)`
`du`	—	the part to be integrated: an arithmetical expression

Return Value: an arithmetical expression containing the type `"int"` or the unevaluated function call.

Related Functions: `subs`, `intlib::changevar`

Details:

- Mathematically, the rule of integration by parts is formally defined for indefinite integrals as

$$\int u'(x)v(x)\,\mathrm{d}x = u(x)v(x) - \int u(x)v'(x)\,\mathrm{d}x$$

  and for definite integrals as

$$\int_a^b u'(x)v(x)\,\mathrm{d}x = \big[u(x)v(x)\big]_a^b - \int_a^b u(x)v'(x)\,\mathrm{d}x$$

- `intlib::byparts(integral, du)` performs in `integral` the integration by parts where `du` is the part to be integrated and returns an expression containing the unevaluated partial integral.
- `intlib::byparts` works for indefinite as well as for definite integrals.
- If MuPAD cannot solve the integral for `du` in case of definite integration, the function call is returned unevaluated.
- The first argument must be an expression of type `"int"`. This can be obtained with `hold` or `freeze` (cf. example 1).
- The second argument `du` should typically be a partial expression of the integrand in `integral`.

Example 1. As a first example we apply the rule of integration by parts to the integral $\int_a^b x \exp(x)\,dx$. By using the `hold` function we secure that the first argument is of type `"int"`:

```
>> intlib::byparts(hold(int)(x*exp(x), x = a..b), exp(x))

           b exp(b) - a exp(a) - int(exp(x), x = a..b)
```

In this case the ansatz is choosen as $u'(x) = exp(x)$ and thus $v(x) = x$.

Example 2. In the following we give a more advanced example using the method of integration by parts for solving the integral $\int \exp(ax)\sin(bx)\,dx$. For this we have to prevent that the integrator already evaluates the integrals. Thus we first inactivate the requested integral with the function `freeze`

```
>> F := freeze(int)(exp(a*x)*sin(b*x), x)

                    int(sin(b x) exp(a x), x)
```

and apply afterwards partial integration with $u'(x) = exp(ax)$:

```
>> F1 := intlib::byparts(F, exp(a*x))

        sin(b x) exp(a x)      / b cos(b x) exp(a x)    \
        ----------------- - int| -------------------, x |
                a              \          a             /
```

To this result again we can apply integration by parts. But to avoid evaluating that integral we have to be very carefully. In order to get it we must use the function `level`:

```
>> F2 := -op(level(F1, 1), 2)

                         / b cos(b x) exp(a x)    \
                      int| -------------------, x |
                         \          a             /
```

With that we can now calculate the requested integral:

```
>> F3 := expand(simplify(op(F1, 1) -
                intlib::byparts(level(F2, 1), exp(a*x))))

 sin(b x) exp(a x)   b cos(b x) exp(a x)
 ----------------- - ------------------- -
         a                     2
                              a

     2
    b  int(sin(b x) exp(a x), x)
    ----------------------------
                  2
                 a
```

As we can see the both integration by parts steps lead to same integral but with a different factor. Therefore we can solve it for the requested integral and we finally get:

```
>> F = normal(1/(1 + b^2/a^2)*
              _plus(op(level(F3, 1), [1..2])))

 int(sin(b x) exp(a x), x) =

    a sin(b x) exp(a x) - b cos(b x) exp(a x)
    -----------------------------------------
                      2    2
                     a  + b
```

Example 3. Here we demonstrate the difference between indefinite and definite integration by parts. If in the indefinite case the partial part cannot be solved, simply the unevaluated integral is plugged into the integration rule:

```
>> intlib::byparts(hold(int)(x*f(x), x),f(x))

              x int(f(x), x) - int(int(f(x), x), x)
```

This is no longer true for the definite case:

```
>> intlib::byparts(hold(int)(x*f(x), x=a..b),f(x))

 Warning: found no closed form for int(f(x), x) [intlib::bypart\
 s]

          intlib::byparts(int(x f(x), x = a..b), f(x))
```

Changes:

- `intlib::byparts` is a new function.

`intlib::changevar` – change of variable

`intlib::changevar(integral, eq, ..)` performs a change of variable for indefinite and definite integrals.

Call(s):

- `intlib::changevar(integral, eq <, var>)`

Parameters:

`integral`	—	integral: an expression of type `"int"`
`eq`	—	equation defining the new integration variable in terms of the old one: an equation
`var`	—	new integration variable: an identifier

Return Value: an expression of type `"int"`.

Related Functions: `subs`, `intlib::byparts`

Details:

- Mathematically, the substitution rule is formally defined for indefinite integrals as

$$\int f(g(x))g'(x)\,\mathrm{d}x = \int f(t)\,\mathrm{d}t, \quad [t = g(x)]$$

  and for definite integrals as

$$\int_a^b f(g(x))g'(x)\,\mathrm{d}x = \int_{g(a)}^{g(b)} f(t)\,\mathrm{d}t, \quad [t = g(x)]$$

- `intlib::changevar(integral, eq <, var>)` performs in `integral` the change of variable defined by `eq` and returns the unevaluated new integral.
- `intlib::changevar` works for indefinite as well as for definite integrals.
- The first argument must be an expression of type `"int"`. This can be obtained with `hold` or `freeze` (cf. example 1).
- If more than two variables occur in `eq`, the new variable must be given as third argument.
- If MuPAD cannot solve the given equation `eq` an error will occur.

Example 1. As a first example we perform a change of variable for the integral $\int_a^b f(x+c)\,\mathrm{d}x$. By using the `hold` function we secure that the first argument is of type `"int"`:

```
>> intlib::changevar(hold(int)(f(x + c), x = a..b),
                     t = x + c, t)

                  int(f(t), t = a + c..b + c)
```

Note, that in this case the substitution equation has among x two further variables. Thus it is necessary to specify the new integration variable as third argument.

Example 2. In the following we give a more advanced example using the change of variable method for solving the integral $\int \sqrt{\tan(x)}\,\mathrm{d}x$. First we perform the transformation $t = \tan(x)$:

```
>> f1:=intlib::changevar(hold(int)(sqrt(tan(x)), x),
                          t = tan(x), t)

                          /    1/2      \
                          |   t         |
                       int| ------, t |
                          |   2         |
                          \ t   + 1     /
```

We apply the further substitution $t = u^2$ to that result. In order to keep this transformation invertible we have to restrict the domain of u:

```
>> assume(u > 0): f2:=intlib::changevar(f1, t = u^2, u)

                          /      2      \
                          |   2 u       |
                       int| ------, u |
                          |   4         |
                          \ u   + 1     /
```

The result of the last transformation is a rational function integral which we can now solve with MuPAD's integrator. Finally we only have to perform the two back substitutions to get the requested integral.

```
>> F:=simplify(subs(subs(eval(f2), u = sqrt(t)),
                     t = tan(x)))

        / /               1/2 \2        \
  1/2   | |        1/2    2    |         |
 2    ln| | tan(x)    - ---- |  + 1/2 |
        \ \               2   /         /
 -------------------------------------- -
                   4

        / /               1/2 \2        \
  1/2   | |        1/2    2    |         |
 2    ln| | tan(x)    + ---- |  + 1/2 |
        \ \               2   /         /
 -------------------------------------- +
                   4

             /      /             1/2 \ \
  1/2        |  1/2 |       1/2   2    | |
 2    arctan| 2    | tan(x)    - ---- | |
             \      \               2   / /
```

```
   --------------------------------------- +
                     2

              /      /              1/2 \ \
     1/2      |  1/2 |       1/2   2    | |
    2    arctan| 2    | tan(x)    + ---- | |
              \      \              2   / /
   ---------------------------------------
                     2
```

Verifying solutions of integrals is almost always a hard task. In this case we may do it with the following function sequence:

```
>> factor(normal(expand(diff(F, x))))

                          / sin(x) \1/2
                          | ------ |
                          \ cos(x) /
```

Changes:

- `intlib::changevar` used to be `changevar`.