Symbolic Integration and Asymptotic Expansions

Classroom notes presents ideas for *Mathematica*-based classroom assignments and suggestions for ways in which *Mathematica* can contribute to the classroom experience. The *Journal* encourages instructors to submit articles to the editor. This column presents a calculus enrichment project on symbolic integration and a note on the contrast between convergent and divergent series for the exponential integral.

Edited by Jack K. Cohen

Elementary Symbolic Integration

Jack K. Cohen

In the new calculus curriculum, finding antiderivatives by exotic substitutions and other tricks is replaced by investigating classes of integrals that succumb to systematic approaches. It is easy to design *Mathematica* exercises based on the built-in function `Integrate`. However, an additional lecture and assignment explaining how packages such as *Mathematica* handle symbolic integration is useful in dispelling the impression that these tools are magic "black boxes." It isn't difficult to build a small prototype symbolic integration package. Indeed, the discussion in section 2.3.13 of the *Mathematica* Book provides a good start and is the basis for this article. On the other hand, it should be emphasized that the function developed here is, indeed, only a prototype and is not part of the built-in *Mathematica* function `Integrate`.

I present this lecture at about the 3/4 point of the Freshman year. It uses a level of coding higher than than that presented elsewhere in the course. Thus, I am careful to explain the coding carefully, but then present (mostly) problems that require only "tweaking" of this code. This lecture also represents my attempt to alert students to the possibility of more sophisticated use of *Mathematica*. Elsewhere in the course, *Mathematica* usage is restricted to a small set of functions and the occasional loop structure. In this lecture, for example, I finally explain the underscore notation that has been used repeatedly throughout the course in defining functions: The symbol `_` means an arbitrary pattern, while the typical combination `x_` simply means an arbitrary pattern named `x`. The latter usage allows us to refer to the variable `x_` on the right side of formulas, e.g., `f[x_] := Sin[Exp[x]]`.

We will call our integrator `OurIntegrate`. Initially it is undefined and just returns our input without doing anything:

```
In[1]:=  fn[x_] := 5 Sin[x] + a x^2 + 3;
```

Jack K. Cohen is a professor of mathematics at the Colorado School of Mines and a member of the faculty at the Center for Wave Phenomena. He received his Ph.D. in mathematics from New York University in 1966.

```
In[2]:=  OurIntegrate[fn[x], x]

Out[2]=  OurIntegrate[3 + a x  + 5 Sin[x], x]
```

Notice that the integration variable `x` is distinguished by appearing as a second argument. We have seen this trick for marking distinguished variables in many built-in *Mathematica* commands such as `Plot`, `D`, and `Integrate`.

Define the sum rule by

```
In[3]:=  OurIntegrate[f_ + g_, x_] :=
            OurIntegrate[f, x] + OurIntegrate[g, x]
```

The patterns `f_` and `g_` represent two expressions to be integrated and `x_` is a pattern for the integration variable. Try out the sum rule:

```
In[4]:=  OurIntegrate[fn[x], x]

Out[4]=  OurIntegrate[3, x] + OurIntegrate[a x , x] +
            OurIntegrate[5 Sin[x], x]
```

It didn't do the integrals, but it did apply the sum rule! What's more, the sum rule was applied twice, even though our rule only indicated one summation. This is a basic feature of *Mathematica*: it applies simplification rules repeatedly without our having to ask for that explicitly.

Let's tell `OurIntegrate` about taking constants out of the integral. The syntax at the end of the following rule reads: on the condition that (`/;`) `c` does not depend on `x` (`FreeQ[c, x]`). We wouldn't want to pull an `x` out of the integral!

```
In[5]:=  OurIntegrate[c_ f_, x_] :=
            c OurIntegrate[f, x] /; FreeQ[c, x]
```

Try it out:

```
In[6]:=  OurIntegrate[fn[x], x]

Out[6]=  OurIntegrate[3, x] + a OurIntegrate[x , x] +
            5 OurIntegrate[Sin[x], x]
```

Not bad: both the 5 and the constant a were pulled out. We still haven't integrated anything, but we're getting a bit closer!

Here's the rule for integrating constants:

```
In[7]:=   OurIntegrate[c_, x_] := c x /; FreeQ[c, x]
```

```
In[8]:=   OurIntegrate[fn[x], x]
```

```
Out[8]=   3 x + a OurIntegrate[x , x] + 5 OurIntegrate[Sin[x], x]
                                  2
```

We actually integrated the 3 correctly!

Here's a rule for integrating powers. Notice that we don't allow $1/x$, which we know has to be handled by a separate rule giving $\ln(x)$:

```
In[9]:=   OurIntegrate[x_^n_, x_] := x^(n+1)/(n+1) /;
              FreeQ[n, x] && n != -1
```

```
In[10]:=  OurIntegrate[fn[x], x]
```

```
                    3
                 a x
Out[10]= 3 x + ---- + 5 OurIntegrate[Sin[x], x]
                  3
```

So far, so good: the x^2 term integrated correctly. Try some other examples:

```
In[11]:=  OurIntegrate[u^8, u]
```

```
           9
          u
Out[11]=  --
          9
```

```
In[12]:=  OurIntegrate[1/x^2, x]
```

```
             1
Out[12]=  -(-)
             x
```

```
In[13]:=  OurIntegrate[x, x]
```

```
Out[13]=  OurIntegrate[x, x]
```

The problem here is that x does *not* fit the pattern x^n! And writing x as x^1 won't help because *Mathematica* simplifies this back to x right away:

```
In[14]:=  OurIntegrate[x^1, x]
```

```
Out[14]=  OurIntegrate[x, x]
```

We could write a special rule for this case. A more elegant solution is to specify that the "default" exponent is 1 when none actually appears. The syntax for a pattern with a default value of 1 is n_:1. However, *Mathematica* "knows" the defaults for the elementary operations. The simpler notation n_. tells *Mathematica* to use its built-in defaults, which are 1 for exponents, 1 for multiplicative factors, and 0 for additive terms. Thus:

```
In[15]:=  OurIntegrate[x_^n_., x_] := x^(n+1)/(n+1) /;
              FreeQ[n, x] && n != -1
```

```
In[16]:=  OurIntegrate[x, x]
```

```
           2
          x
Out[16]=  --
          2
```

Now we define rules for some common functions. For a little extra generality, we replace the argument x by the pattern a_. x_ + b_., which matches an arbitrary linear function of x. (The coefficient a defaults to 1 and the constant b defaults to 0.)

```
In[17]:=  OurIntegrate[1/(a_. x_ + b_.), x_] := Log[a x + b]/a /;
              FreeQ[{a, b}, x]
```

```
In[18]:=  OurIntegrate[Exp[a_. x_ + b_.], x_] := Exp[a x + b]/a /;
              FreeQ[{a, b}, x]
```

```
In[19]:=  OurIntegrate[Sin[a_. x_ + b_.], x_] := -Cos[a x + b]/a /;
              FreeQ[{a, b}, x]
```

```
In[20]:=  OurIntegrate[Cos[a_. x_ + b_.], x_] := Sin[a x + b]/a /;
              FreeQ[{a, b}, x]
```

Now we can integrate our original function.

```
In[21]:=  OurIntegrate[fn[x], x]
```

```
                    3
                 a x
Out[21]= 3 x + ---- - 5 Cos[x]
                  3
```

Here are two more examples.

```
In[22]:=  OurIntegrate[3x^2 + E^(2x) + 3/x + 4, x]
```

```
           2 x
          E              3
Out[22]=  ---- + 4 x + x  + 3 Log[x]
           2
```

```
In[23]:=  OurIntegrate[Sin[2x + q], x]
```

```
          -Cos[q + 2 x]
Out[23]=  -------------
                2
```

Exercise: Fix up the rules so that OurIntegrate can do this integration:

```
In[24]:=  OurIntegrate[(x + 2)^3, x]
```

```
Out[24]=  OurIntegrate[(2 + x)^3, x]
```

Compare the result of your routine with the built-in Integrate.

Exercise: Instead of defining linear arguments for every function, would a general rule like the following work?

```
OurIntegrate[f_[a_. x_ + b_.], x_] :=
    OurIntegrate[f[u], u]/a /. u -> a x + b
```

To bring our little project to a close, we develop the rules for integrating powers of the sine and cosine functions. To do this, we use recursion formulas that appear in all serious tables of integrals:

```
In[25]:=  OurIntegrate[Cos[x_]^m_., x_] :=
              Cos[x]^(m-1) Sin[x]/m +
                  (m-1) OurIntegrate[Cos[x]^(m-2), x]/m /;
              FreeQ[m, x] && IntegerQ[m] && (1 < m)
```

```
In[26]:=  OurIntegrate[Sin[x_]^n_., x_] :=
              -Sin[x]^(n-1) Cos[x]/n +
                  (n-1) OurIntegrate[Sin[x]^(n-2), x]/n /;
              FreeQ[n, x] && IntegerQ[n] && (1 < n)
```

```
In[27]:=  OurIntegrate[Cos[x_]^m_. Sin[x_]^n_., x_] :=
              Cos[x]^(m-1) Sin[x]^(n+1)/(m+n) +
              (m-1) OurIntegrate[Cos[x]^(m-2) Sin[x]^n, x]/(m+n) /;
              FreeQ[{m, n}, x] && IntegerQ[m] && IntegerQ[n] &&
                  (0 < = n < = m)
```

```
In[28]:=  OurIntegrate[Cos[x_]^m_. Sin[x_]^n_., x_] :=
              -Sin[x]^(n-1) Cos[x]^(m+1)/(m+n) +
              (n-1) OurIntegrate[Cos[x]^m Sin[x]^(n-2), x]/(m+n) /;
              FreeQ[{m, n}, x] && IntegerQ[m] && IntegerQ[n] &&
                  (0 < = m < n)
```

Exercise: Since the first two formulas are special cases of the last two, why do we need them? Can you find a way around this redundancy?

Exercise: What is the exact series of reductions used to do the following integral?

```
OurIntegrate[Cos[x]^4 Sin[x]^5, x]
```

If you try this example, you'll see that the answer is returned in a clunky form. A good package would clean it up; we will settle for using Expand to clean up after the fact:

```
In[29]:=  OurIntegrate[Cos[x]^4 Sin[x]^5, x] // Expand
```

$$
\text{Out[29]}= \; \frac{-8\,\text{Cos}[x]}{315} + \frac{8\,\text{Cos}[x]\,\text{Sin}[x]^2}{315} - \frac{4\,\text{Cos}[x]^3\,\text{Sin}[x]^2}{105} +
$$
$$
\frac{4\,\text{Cos}[x]^3\,\text{Sin}[x]^4}{63} - \frac{\text{Cos}[x]^5\,\text{Sin}[x]^4}{9}
$$

Exercise: Check the results of the following integrations:

```
OurIntegrate[Cos[x]^2, x]
```
```
OurIntegrate[Cos[x]^3, x]
```
```
OurIntegrate[Cos[x]^4, x] // Expand
```
```
OurIntegrate[Cos[x]^4 Sin[x]^4, x] // Expand
```

Exercise: Extend our integration function to handle functions of the form x^n E^(ax), x^n Cos[ax]^m, and x^n Sin[ax]^m.

Finally, when doing experiments, don't forget to use Remove[OurIntegrate] to get a "clean slate" before re-building the rules.

Convergent vs. Asymptotic Representations

John W. Stockwell, Jr.
Colorado School of Mines
john@dix.mines.edu

In mathematical physics, important integrals that defy analytical solution are often encountered. Asymptotic expansions provide a useful means for approximating such integrals.

Pedagogically speaking, the subject of asymptotic series presents a counter-intuitive dilemma to many students. Based on experience with convergent series expansions gleaned from lower-level math courses, students often unconsciously live by two "rules" when dealing with series expansions: First, a series must converge to be a valid representation. Second, the more terms in a partial sum, the better the representation.

Asymptotic series generally violate these two "rules." Such series cannot be guaranteed to converge. Indeed, divergence is the common case. This means that "more terms" will almost never be "better."

Remarkably, a partial sum of just the first few terms of such a divergent series can provide a much better approximation than many terms of an convergent series representation of the same integral. However, asymptotic series do have the fundamental limitation that they can provide only a certain degree of accuracy and no more. On the other hand, a convergent series can theoretically achieve unlimited accuracy. But this is, indeed, only a theoretical result. In practice, for a fixed precision, round-off error puts a limit on the accuracy of even a convergent series. In bad cases, the limitation of round-off error can preclude getting even a single significant figure. To illustrate the limitation of round-off error, the computations below deliberately do not use *Mathematica*'s capability for extended precision.

Chapter 1 of *Asymptotic Expansions of Integrals*, by N. Bleistein and R.A. Handlesman (Dover Publications, 1986) is the source of the examples that follow. All equation numbers refer to this text.

The integral studied here is a form of the "exponential integral" (equation 1.1.1):

$$
I(x) = x e^x \int_x^\infty \frac{e^{-t}}{t}\, dt
$$

The *Mathematica* implementation is

```
In[1]:=  exact[x_] := x Exp[x] ExpIntegralE[1, x]
```

The convergent series approximation to this integral is based on equation 1.1.5,

$$
I(x) = x e^x \left[-\log x - \gamma + \sum_{n=1}^\infty \frac{(-1)^{n+1} x^n}{n \cdot n!} \right]
$$

with implementation

```
In[2]:=  convergent[x_, m_] :=
             x Exp[x] (-Log[x] - EulerGamma +
                 Sum[ (-1)^(n+1) x^n / (n n!), {n, 1, m}])
```

The (divergent) asymptotic approximation is based on equation 1.1.6,

$$I(x) = \sum_{n=0}^{N-1} \frac{(-1)^n n!}{x^n} + (-1)^N N! \, x e^x \int_x^\infty \frac{e^{-t}}{t^{N+1}} \, dt$$

with implementation

In[3]:= `asymptotic[x_, m_] := Sum[n! (-1)^n/x^n, {n, 0, m}]`

For small x, the convergent expansion shines. For example, with $x = 0.1$, we obtain six significant figures with only three terms of the series:

In[4]:= `exact[0.1] // N`

Out[4]= 0.201464

In[5]:= `TableForm[`
`    Table[{m, convergent[0.1, m] // N}, {m, 5}],`
`    TableHeadings -> {None, {"m", "Conv. Series"}},`
`    TableAlignments -> {Right, Right} ]`

Out[5]//TableForm=

m	Conv. Series
1	0.201735
2	0.201458
3	0.201464
4	0.201464
5	0.201464

Even for $x = 1$, the convergent series is adequate; it now takes eight terms to achieve similar accuracy:

In[6]:= `exact[1] // N`

Out[6]= 0.596347

In[7]:= `TableForm[`
`    Table[{m, convergent[1, m] // N}, {m, 10}],`
`    TableHeadings -> {None, {"m", "Conv. Series"}},`
`    TableAlignments -> {Right, Right} ]`

Out[7]//TableForm=

m	Conv. Series
1	1.14925
2	0.469677
3	0.620692
4	0.592377
5	0.596907
6	0.596278
7	0.596355
8	0.596347
9	0.596347
10	0.596347

But consider $x = 10$:

In[8]:= `exact[10] // N`

Out[8]= 0.915633

It takes roughly 50 terms of the convergent series to reproduce these six significant digits:

In[9]:= `TableForm[`
`    Table[{m, convergent[10, m] // N}, {m, 1, 70, 10}],`
`    TableHeadings -> {None, {"m", "Conv. Series"}},`
`    TableAlignments -> {Right, Right} ]`

Out[9]//TableForm=

m	Conv. Series
1	$1.56833 \, 10^6$
11	$2.21337 \, 10^7$
21	62688.
31	2.9344
41	0.915636
51	0.915633
61	0.915633

We can obtain three-place accuracy with only six terms of the asymptotic expansion, but we can achieve no further accuracy with more terms. In fact, a partial sum of 14 terms of the asymptotic expansion begins to exhibit divergent oscillations.

In[10]:= `TableForm[`
`    Table[{m, asymptotic[10, m] // N}, {m, 1, 20}],`
`    TableHeadings -> {None, {"m", "Asymp. Series"}},`
`    TableAlignments -> {Right, Right} ]`

Out[10]//TableForm=

m	Asymp. Series
1	0.9
2	0.92
3	0.914
4	0.9164
5	0.9152
6	0.91592
7	0.915416
8	0.915819
9	0.915456
10	0.915819
11	0.91542
12	0.915899
13	0.915276
14	0.916148
15	0.91484
16	0.916933
17	0.913376
18	0.919778
19	0.907614
20	0.931943

The approximations of the exponential integral for $x = 100$ dramatically show the advantage of using an asymptotic expansion. The exact value we are shooting for is

```
In[11]:= exact[100] // N
```

```
Out[11]= 0.990194
```

Here we show selected partial sums of up to 201 terms of the convergent series approximation. Though the series representation is convergent, the numerical implementation of this approximation need not be, owing to the effects of roundoff error. Indeed, this is what is happening here, with normal precision being assumed. An implementation of this representation in a language such as C or FORTRAN would provide similar results.

```
In[12]:= TableForm[
        Table[{m, convergent[100, m] // N}, {m, 1, 201, 50}],
          TableHeadings -> {None, {"m", "Conv. Series"}},
          TableAlignments -> {Right, Right} ]
```

```
Out[12]//TableForm=
```

m	Conv. Series
1	$2.54881 \ 10^{47}$
51	$2.23022 \ 10^{79}$
101	$1.39425 \ 10^{85}$
151	$8.1686 \ 10^{79}$
201	$2.78734 \ 10^{67}$

However, the desired degree of accuracy can be obtained in only three terms with the asymptotic approximation.

```
In[13]:= TableForm[
        Table[{m, asymptotic[100, m] // N}, {m, 1, 5}],
          TableHeadings -> {None, {"m", "Asymp. Series"}},
          TableAlignments -> {Right, Right} ]
```

```
Out[13]//TableForm=
```

m	Asymp. Series
1	0.99
2	0.9902
3	0.990194
4	0.990194
5	0.990194

For additional insight into the problem of asymptotic expansions, students could be asked to vary the precision in the numerical evaluation of the series expansions above. Series expansions of other integrals can be easily studied in a similar manner.

Call for Submissions

Teachers, we need your ideas for this column! You may submit either *Mathematica* notebooks or text-processed materials. Good concepts for spicing up lectures or for making good lab assignments are more important than sophisticated *Mathematica* coding. Moreover, we are glad to make suggestions for improving *Mathematica* usage in the editing process.

Jack K. Cohen
Colorado School of Mines, Golden, CO 80401
  jkc@dix.mines.edu

The electronic supplement contains the notebook OurIntegrate.