Utilizing the Symbolic Capabilities Of a Computer Algebra System

Melvin J. Maron, Patricia A. Ralston, and Larry D. Tyler
Department of Engineering Mathematics And Computer Science
University of Louisville, KY 40292

Abstract

This paper identifies several ways to exploit the "symbol crunching" capabilities of a Computer Algebra System to enhance learning in an interactive setting, namely: as a source of general formulas; to help recognize patterns and deduce general formulas; to demonstrate mathematical rules via rapid repetition; and to present straightforward, easily-understood symbolic derivations. Illustrative examples are given using Maple V, but can be adapted to any Computer Algebra System.

Introduction

Computer Algebra Systems (CAS) have been used imaginatively and extensively to enhance the presentation of analytic concepts [3], and undergraduate texts have been written for use with a CAS [e.g., 1, 2]. These texts effectively exploit the graphical and numerical capabilities of a CAS to help explain mathematical concepts. However, the symbolic capabilities cannot be used effectively to clarify concepts because hard-copy symbolic displays can just as well be typeset as created by a CAS. Instead, the symbolic capabilities are used primarily to perform the "busy work" steps of problem solutions quickly, by showing CAS commands and working with the returned results. On the other hand, interactive environments such as classroom demonstrations, CAS-based tutorials, or electronic books provide excellent opportunities to use a CAS's ability to perform rapid symbolic computations to enhance the understanding of mathematical concepts and operations.

Four examples are given, some accompanied by possible related tutorial or homework exercises, to illustrate how the symbolic capabilities of a CAS can be used in an interactive setting. The illustrative examples use Maple; however, they can be adapted to any CAS. Readers who are unfamiliar with Maple need to know four things to follow the examples: First, Maple's returned response to a command is displayed if the command ends with a semicolon (;) but not if it ends with a colon (:). Second, Maple stores the most recent response as " (quote symbol). Third, a variable, say **f**, is "cleared" by using the unassign command **f:='f':**, which stores its name (the one-character string *f*) over anything that might have been stored. Finally, Maple obtains and displays all derivatives as partial derivatives.

Four illustrative examples

Generating general formulas When a CAS performs operations on unassigned variables, it returns the general formulas that it uses to carry out these operations. Students can be taught to use this feature to help them recall general formulas for derivatives, integrals, transforms, etc. This symbolic capability also allows the CAS to serve as a vehicle for rapid illustration and pattern recognition, as shown for two differentiation formulas in Examples 1 and 2.

EXAMPLE 1 The Product Rule for $\frac{d}{dx} f(x)g(x)$

The following Maple input group stores the Product Rule formula as *PR*:

```
> f := 'f': g := 'g':Diff((f*g)(x),x):
> PR := "=value(");
```

$$PR := \frac{\partial}{\partial x} \mathrm{f}(x)\,\mathrm{g}(x) = \left(\frac{\partial}{\partial x} \mathrm{f}(x)\right)\mathrm{g}(x) + \mathrm{f}(x)\left(\frac{\partial}{\partial x} \mathrm{g}(x)\right)$$

Rapid illustration Maple automatically updates *PR* when the values assigned to f and/or g are changed. So the product rule can be illustrated rapidly by simply varing the functions stored as f and g and then executing an input group that displays the current value of *PR* along with other relevant current values.

```
> f := x -> exp(3*x):  g := sin:        # vary f, g here
> 'f(x)'=f(x), 'g(x)'=g(x);             # display f(x), g(x)
> Diff('f(x)',x)=diff(f(x),x), Diff('g(x)',x)=diff(g(x),x);
> PR;
```

$$f(x) = e^{(3x)},\ g(x) = \sin(x)$$

$$\frac{\partial}{\partial x} f(x) = 3\, e^{(3x)},\ \frac{\partial}{\partial x} g(x) = \cos(x)$$

$$\frac{\partial}{\partial x} e^{(3x)} \sin(x) = 3\, e^{(3x)} \sin(x) + e^{(3x)} \cos(x)$$

Routine problem Store as *QR* the differentiation rule for $f(x)/g(x)$. Differentiate $\sqrt{x}/\tan(x)$ by hand, then use the stored *QR* to confirm your answer.

Pattern Recognition The following input group illustrates how a CAS can help students recognize the pattern for differentiating a product of *n* functions.

```
> f:='f': g:='g': Diff((f*g*h)(x),x):
> PR3 := "=value(");
```

$$PR3 := \frac{\partial}{\partial x} f(x) g(x) h(x) = \left(\frac{\partial}{\partial x} f(x)\right) g(x) h(x) + f(x)\left(\frac{\partial}{\partial x} g(x)\right) h(x) + f(x) g(x)\left(\frac{\partial}{\partial x} h(x)\right)$$

Discussion problem Based on PR and PR3 (and PR4, obtained for yourself if desired), state the number of terms, and describe the *i*th term, in the general formula for $\frac{d}{dx}\ f_1(x) f_2(x) \cdots f_n(x)$.

Differentiating integrals with variable limits lends itself nicely to this approach.

EXAMPLE 2 Leibnitz' Rule for $\frac{d}{dx}\int_{u(x)}^{v(x)} f(t)\ dt$

First, create the general formula and store it as *LR*.

```
> f:='f': Diff(Int(f(t),t=u(x)..v(x)),x):
> LR := "=value(");
```

$$LR := \frac{\partial}{\partial x}\int_{u(x)}^{v(x)} f(t)\ dt = f(v(x))\left(\frac{\partial}{\partial x} v(x)\right) - f(u(x))\left(\frac{\partial}{\partial x} u(x)\right)$$

As in Example 1, one can rapidly generate illustrative examples by simply varying the function f and the expressions for the variable limits u(*x*) and v(*x*).

```
> f := x -> sqrt(1+x^2): u(x) := x^2: v(x) := sin(x):
> LR;
```

$$\frac{\partial}{\partial x}\int_{x^2}^{\sin(x)} \sqrt{1+t^2}\ dt = \sqrt{1+\sin(x)^2}\cos(x) - 2\sqrt{1+x^4}\ x$$

One can also emphasize that *t* is a "dummy" variable by changing *t* to any symbol (other than *x*).

Multivariable calculus problem Replace the integrand f(*t*) with f(*x*,*t*) in *LR* to obtain the general Leibnitz' Rule formula. Then find *d/dx* of the integral of $\sqrt{x^2+t^2}$ from x^2 to sin(*x*).

Pattern recognition Our third example shows how a general formula can be deduced from a symbolic pattern.

EXAMPLE 3 Deduce a general formula for the Laplace transform of $t^n f(t)$ by looking for a pattern based on the result for *n* = 1, 2, 3.

```
> with(inttrans,laplace): f:='f':
> for n from 1 to 3 do
      L(t^n*f(t)) = laplace(t^n*f(t), t, s)
   od;
```

$$L(t\, f(t)) = -\left(\frac{\partial}{\partial s} \text{laplace}(f(t), t, s)\right)$$

$$L\left(t^2 f(t)\right) = \frac{\partial^2}{\partial s^2} \text{laplace}(f(t), t, s)$$

$$L\left(t^3 f(t)\right) = -\left(\frac{\partial^3}{\partial s^3} \text{laplace}(f(t), t, s)\right)$$

It appears that $L(t^n f(t)) = (-1)^n \frac{d^n}{ds^n} L\big(f(t)\big)$.

Formula Derivation After investing the energy to determine a pattern *deductively*, students may be more interested in seeing a formal derivation. The following Calculus III example illustrates how the interactive symbolic capabilities of a CAS can be exploited to present straightforward, easily-understood formula derivations.

EXAMPLE 4 Derive the general formula for the distance from the point *S*(*xS*, *yS*, *zS*) to the plane defined by the equation *A x* + *B y* + *C z* = *E*.

Approach If *Q* is any point on the plane, **N** is a normal vector, and **QS** is the vector from *Q* to *S*, then

desired distance = | scalar component of **QS** on N |.

Solution First load the needed vector commands and store the equation of the plane and the normal vector **N**.

```
> with(linalg, vector,dotprod):
> eqPl := A*x + B*y + C*z = E;
> N := vector( [A, B, C] );
```

$$eqPl := A\,x + B\,y + C\,z = E$$
$$N := [A \quad B \quad C]$$

We need the vector **QS** from a point Q on the plane to the given point **S**. Assuming $C \neq 0$, a convenient Q is the z-intercept, Q(0, 0, zQ), with zQ obtained by substituting $x = y = 0$ in *eqPl* and then solving for z.

```
> xQ := 0: yQ := 0:
> zQ := solve( subs(x=xQ, y=yQ, eqPl), z );
> QS := vector( [xS,yS,zS] - [xQ,yQ,zQ] );
```

$$zQ := \frac{E}{C}$$

$$QS := \left[xS \quad yS \quad zS - \frac{E}{C} \right]$$

The desired distance is the absolute value of the scalar component of the vector **QS** on the normal vector **N**.

```
> abs( dotprod(N,QS) /sqrt(dotprod(N,N)) ):
> distStoPl := simplify("):
> `Distance from S to the plane` = distStoPl;
```

$$distance\ from\ S\ to\ plane = \left| \frac{A\,xS + B\,yS + zS\,C - E}{\sqrt{A^2 + B^2 + C^2}} \right|$$

If one wanted to see the details of the last step, one could simply cut **simplify** from the last command and re-execute the last input group.

The preceding derivation arguably could just as well have been carried out by hand. However, by following the CAS solution with the following problem, one can demonstrate quickly and persuasively, in a way not realistically possible by hand, that *the derivation approach works regardless of the point used for Q*.

Discussion problem Get a *general* point Q on the plane by replacing **xQ := 0: yQ := 0:** by the *symbolic* assignments **xQ := 'xQ': yQ := 'yQ'**, then re-execute all input groups. What can you conclude from the fact that although the vector **QS** depends on the coordinates of both Q and S, the coordinates of Q do not appear in the (simplified) formula for the distance from S to the plane?

Once the general distance formula has been stored, its use can be illustrated rapidly by assigning values to S and Pl in an input group such as

```
> S := (2,-1,3); Pl := (4,2,5,0);      # store S and Pl here
> xS := S[1]: yS := S[2]: zS := S[3]:
> A := Pl[1]: B := Pl[2]: C := Pl[3]: E := Pl[4]:
> `For S`=S,`and (A,B,C,E)`=Pl,` distance`=distStoPl;
```

and then executing the input group to display the current value of *distStoPl* as in Examples 1 and 2.

Discussion and summary

We have described and illustrated how the symbolic capabilities of a CAS can be used in an interactive setting to enhance conceptual understanding. Students can be taught to use a CAS as a source of general formulas. The use of these formulas can be illustrated repeatedly and rapidly using specific cases chosen by either the student or the instructor. When appropriate, formulas can be displayed in ways that reveal general symbolic patterns.

Finally, a CAS's ability to perform cumbersome symbolic manipulations quickly and accurately makes it possible to give straightforward derivations that are persuasive, informative, and easy to follow because the only displayed intermediate results are those that illuminate the derivation. Intermediate calculations that are initially suppressed can easily be examined selectively after the general idea is understood. Exposure to such derivations should encourage students to obtain and work work with analytic results, rather than be inclined to perform numerical verifications, as they would in a calculator-based environment.

References

1. Holmes M., Ecker J., Boyce W., and Siegmann W., *Exploring Calculus with Maple*, Addison-Wesley, Reading MA, 1992.

2. Harris K. and Lopez R. J., *Discovering Calculus with Maple*, Wiley and Sons, New York, NY, 1995.

3. Karian Z. A., *Symbolic Computation in Undergraduate Mathematics Instruction*, MAA Notes #24, 1991