A Case History in Interactive Problem-Solving

Richard J. Fateman
Department of Mathematics
Massachusetts Institute of Technology
Cambridge, Mass.

ABSTRACT

MACSYMA [1], a computer program for algebraic manipulation, is used to solve, symbolically and exactly, a problem involving integration and the solution of an infinite set of linear equations. This is a tutorial in the use of MACSYMA, and illustrates some of the choices available to the interactive user which aid in solving a mathematical problem.

I - The problem

In [2], H. P. Greenspan develops a model of the generation of large scale magnetic fields from small scale motions of a conducting fluid. We take as a starting point, a set of equations (whose significance will not be discussed here) to be solved for coefficients B[2N+1]. The equations are

$$\sum_{N=0}^{INF} (A_{N,K}\, B_{2N+1}) + \frac{2\,(K+1)\,(2K+1)\,B_{2K+1}}{4K+3} = C_K \qquad K = 0, 1, \ldots \qquad (1)$$

$$A_{N,K} = -\int_0^{\frac{\%PI}{2}} ((P^{(1)}_{2K+1}(COS(TH))\, SIN(TH)^{2N+3} \left(\int_{TH}^{\frac{\%PI}{2}} (P^{(1)}_{2N+1}(COS(Z))\, SIN(Z)^{-2N-3})\, dZ\right))/COS(TH))\, dTH \qquad (2)$$

and

$$C_K = -\frac{GAMMA(\frac{3}{2})\, GAMMA(2K+3)}{2\, GAMMA(K+3)\, GAMMA(\frac{3}{2}-K)\, GAMMA(2K+1)} \qquad (3)$$

where P is an associated Legendre function. The above displays were generated by MACSYMA, and includes a summation sign and two integrals (stand back a bit if you can't see them).

II - The Solution

Equation (1) is an infinite set of linear equations. The technique for solving for the B[i] is to generate and solve the first equation for B[1], then generate the next equation, and solve these two equations together to get B[1] and B[3], etc. One hopes that the sequence of values for B[1] derived in this way converges, and similarly for B[3], etc. It is also expected, in the context of this problem, that B[i] tends to 0 for large i.

We will assume that the reader has a slight acquaintance with MACSYMA [1] or a similar system for algebraic manipulation.

Let us do the easy parts first. Equation (3) defines a set of values for C. We could define a function C(K) which when given the value for K returns the value for C(K), but this would repeatedly compute C. We might as well save them in an array, especially since in MACSYMA it is just as simple. In this case, when C[1] is requested the first time, it gets computed using the formula, and is then stored in memory so that subsequent references to C[1] are just read out of storage. The definition (and display) of C as given to MACSYMA is:

```
(C1) C[K]:= -GAMMA(3/2)*GAMMA(2*K+3)/
     (2*GAMMA(2*K+1)*GAMMA(3/2-K)*GAMMA(3+K));
```

$$\text{(D1)} \quad C_K := -\frac{SQRT(\%PI)\; GAMMA(2K+3)}{4\, GAMMA(K+3)\, GAMMA(\frac{3}{2}-K)\, GAMMA(2K+1)}$$

Note that MACSYMA has simplified the GAMMA(3/2). Let us compute a list of the first 4 values.

```
(C2) [C[0],C[1],C[2],C[3]];
```

$$\text{(D2)} \quad [-\frac{1}{2}, -\frac{1}{2}, \frac{5}{32}, -\frac{7}{80}]$$

Next, let us compute the associated Legendre functions by any convenient method. Rodrigues' formula is handy:

$$P^{(N)}_M(X) = \frac{(-1)^N\,(1-X^2)^{N/2}\,\left(\frac{D^{N+M}}{DX^{N+M}}(X^2-1)^M\right)}{2^M\, M!}$$

But since all we need is the N=1 case evaluated at cos(y), we can replace (1-X↑2)↑(N/2) by sin(y), and after taking derivatives with respect to x, substitute cos(y) for x. We can set up an array of functions, one for each associated Legendre function at a cosine point by

```
(C3) ALF[M](Y):=-SIN(Y)/(2↑M*M!)*SUBST(COS(Y),X,
     DIFF((X↑2-1)↑M,X,M+1));
```

$$\text{(D3)} \quad ALF_M(Y) := -\frac{SIN(Y)\; SUBST(COS(Y),\, X,\, \frac{D^{M+1}}{DX^{M+1}}(X^2-1)^M)}{2^M\, M!}$$

```
(C4) ALF[5](X);
```

$$\text{(D4)} \quad -((7200\,(COS^2(X)-1)^2 + 57600\, COS^4(X) + 86400\, COS^2(X)\,(COS^2(X)-1))\, SIN(X))/3840$$

This might be sufficient, but it is somewhat neater to convert these results into a simpler form involving just sines or just cosines. Since formula (2) already involves sines, we choose sines. We can do this by defining yet another array, which has the same values as ALF, but with (1-SIN(Y)↑2) substituted for COS(Y)↑2 everywhere. Since we also want to convert COS(Y)↑4 (etc) to SIN's, we use RATSUBST:

```
(C5) ALF1[M](Y):= RATSUBST(1-SIN(Y)↑2,COS(Y)↑2,ALF[M](Y))$
```

It seems preferable to divide the denominator through, so we type:

```
(C6) RATEXPAND:T$

(C7) ALF1[5](X);
                       5            3
                315 SIN (X)   105 SIN (X)
(D7)          - ----------- + ----------- - 15 SIN(X)
                     8             2
```

The next problem is to evaluate the integrals of equation (2). We may have no a priori knowledge of whether the integral is singular, or if it can be done exactly by conventional methods, or if it involves special functions. We can let MACSYMA take care of these problems if we just define:

```
(C8) A[N,K]:= INTEGRATE(-SIN(TH)↑(2*N+3)/COS(TH)*
ALF1[2*K+1](TH)*
      INTEGRATE(ALF1[2*N+1](Z)/SIN(Z)↑(2*N+3),Z,TH,%PI/2),
            TH,0,%PI/2)$
```

This is a time-consuming calculation since definite integration in MACSYMA is done with great concern for a function's behavior, and can use contour integration algorithms, and various heuristic methods. Another fault with this formulation is that if we are going to compute many of these A[N,K], we will be recomputing the inner integral, which depends only on N, repeatedly. Although this formulation can be used, it is helpful to analyze the problem further.

Some experimentation shows that the inner integral is always 0 at %PI/2, and that all we have to do is compute:

```
(C9) INNERINT[N]:= -INTEGRATE(ALF1[2*N+1](TH)/SIN(TH)↑(2*N+3),TH)$
```

An improvement in running time can be made by first cancelling the SIN's in the denominator and expanding the integrand. After this expansion, the integration program produces an integral entirely in terms of 1/TAN(TH). Without this transformation, the integration program returns a much more unwieldy answer. The interactive nature of MACSYMA aided in this discovery, because both of the indicated programs were tried out.

```
(C10) INNERINT[N]:= -INTEGRATE(RATEXPAND(ALF1[2*N+1](TH)/
    SIN(TH)↑(2*N+3)),TH)$

(C11) INNERINT[2];

                      15          15           3
(D11)           - --------- + ---------- - --------
                  8 TAN(TH)        3           5
                              2 TAN (TH)   TAN (TH)
```

This is fairly neat. What we would really prefer, is to remove the TAN's and put in SIN's, or at least SIN's and COS's. This can be done quite simply using RATSUBST.

```
(C12) I1[N]:=RATSUBST(1-SIN(TH)↑2,COS(TH)↑2,
    RATSUBST(SIN(TH)/COS(TH),TAN(TH),INNERINT[N]))$

(C13) I1[2];
                  99 COS(TH)   27 COS(TH)   3 COS(TH)
(D13)           - ---------- + ---------- - ---------
                  8 SIN(TH)         3            5
                               2 SIN (TH)   SIN (TH)
```

Now we are at the point where we can generate the integrand for the outer integral, and examine what it is we really are after. The integrands have a suprisingly simple structure.

```
(C14) R(N,K):=RATEXPAND(-SIN(TH)↑(2*N+3)/
   COS(TH)*ALF1[2*K+1](TH)*I1[N])$

(C15) R(0,1);
                          5
                   15 SIN (TH)         3
(D15)              ----------- - 6 SIN (TH)
                        2

(C16) R(1,0);
                         5
                   7 SIN (TH)         3
(D16)              ---------- - 2 SIN (TH)
                        2

(C17) R(1,1);
                        7
                 105 SIN (TH)          5             3
(D17)          - ------------ + 36 SIN (TH) - 12 SIN (TH)
                      4
```

It can be determined that in fact all integrands are sums of powers of SIN(TH), and only odd powers, at that. Integration could be done by using the INTEGRATE command, but this computation would proceed by doing the indefinite integral, and then return the difference of evaluating at the limits. This is fine for one or two cases, but we can save considerable time by computing definite integrals of powers of SIN, and storing them in an array. We can also illustrate some more facilites of MACSYMA along the way.

There is a little formula available for computing the integral from 0 to %PI/2 of an odd power of SIN. It is:

```
(C18) ISIN[N]:=BLOCK([M], M:(N-1)/2,
 RETURN(2↑(2*M)*(M!)↑2/N!))$
```

Now we are ready to produce the A[N,K]:

```
(C19) A[N,K]:= BLOCK([EXP,ANS,COEF,H],
ANS:0,
EXP:R(N,K),
H:HIPOW(EXP,SIN(TH)),
FOR I:H STEP -2 THRU 1 DO
[COEF:BOTHCOEF(EXP,SIN(TH)↑I),
ANS:ANS+ISIN[I]*INPART(COEF,1),
EXP:INPART(COEF,2)],
RETURN(ANS))$
```

Let us describe the steps in the program above:

(line 1) the variables EXP, ANS, COEF, and H are declared local to the BLOCK. That is, values of EXP (etc.) within this program bear no relation to values previously assigned to EXP (etc.).
(line 2) Set ANS to 0. ANS will have the answer when we conclude execution of this program.
(line 3) Set EXP to the integrand computed by function R.
(line 4) Set H to the highest power of SIN(TH) in EXP.
(line 5) Set up a loop on I for all (odd) powers of SIN(TH) in EXP.
(line 6) Set COEF to a list of (1) the coefficient of SIN(TH)↑I in EXP, and (2), the rest of EXP with the SIN(TH)↑I term removed.
(line 7) Add to ANS the integral of SIN(TH)↑I times the appropriate coefficient. (INPART(COEF,1) picks out the first part of the answer from BOTHCOEF.)
(line 8) Set EXP to the part which remains after removing the SIN(TH)↑I term from the integrand. (Also, return to line 6 for the next value of I unless I=1)
(line 9) Return ANS as the value of the integral, and set A[N,K] to that value.

We can now set up the equations to be solved. We can convert equation (2) to the form

```
       INF
      \----
       \
        >    (BET(N, K) B        ) = C
       /                2 N + 1       K
      /----
      N = 0
```

where BET(N,K) is A[N,K] if N is not equal to K, otherwise to A[N,K] + an additional term. It looks like this:

```
(C20) BET(N,K):= IF (N#K) THEN A[N,K] ELSE
   A[N,K]+(2*N+1)*(2*N+2)/(4*N+3)$
```

SETUP sets up N equations, EQ[0] , ..., EQ[N-1].

```
(C21) SETUP(N):= FOR L:0 THRU N-1 DO
   EQ[L]:SUM(BET(I,L)*B[2*I+1],I,1,N)=C[L]$
```

Finally we define a program which sets up lists of equations and variables, and calls the built-in command, SOLVE. CONS is a program which adds an element to the beginning of a list. An empty list is [].

```
(C22) ISOLVE(N):= BLOCK ([EQS,VARS],
EQS:[], VARS:[],
SETUP(N),
FOR L: 1 THRU N DO
[EQS:CONS(EQ[L-1],EQS),
VARS:CONS(B[2*L+1],VARS)],
RETURN(SOLVE(EQS,VARS)))$
```

Let's try it out.

```
(C23) ISOLVE(1);
(D23)                              [E23]
```

This is a list of the solutions from SOLVE. Evaluate it:

```
(C24) EV(%);
                                          15
(D24)                            [B  =  - --]
                                  3       16
```

If we set TIME to TRUE, we can find out how long computations take.

```
(C25) TIME:TRUE$
TIME= 4 MSEC.

(C26) ISOLVE(2);
TIME= 11853 MSEC.
(D27)                          [E26, E27]

(C28) EV(%);
TIME= 11 MSEC.
                            35              35
(D28)                   [B  = ---, B  =  - --]
                          5   144   3       48

(C29) ISOLVE(3);
TIME= 51849 MSEC.
(D31)                     [E29, E30, E31]

(C32) EV(%);
TIME= 14 MSEC.
                          63          77              35
(D32)            [B  =  - ---, B  = ---, B  =  - --]
                   7      512   5   576   3       48
```

In fact, we can run this example out much further, and we note that we have an exact solution from ISOLVE(N) for the terms up to 2N-1. That is, the infinite set of equations has an exact solution. This surprising situation is a result of the fact that A[N,K] is 0 for K > N, which is obvious if we look at the values of the A[N,K]. We can display them in a matrix.

First we have to offset the values of A by 1, since matrices in MACSYMA (as in most other places) begin their indexing at 1, not 0. We can do it as follows:

```
(C33) AA[N,K]:=A[N-1,K-1]$
TIME= 20 MSEC.

(C34) GENMATRIX(AA,3,3);
TIME= 3055 MSEC.
                          [   2              ]
                          [ - -     0     0  ]
                          [   3              ]
                          [                  ]
                          [ 8        4       ]
(D34)                     [ --     - -    0  ]
                          [ 15       5       ]
                          [                  ]
                          [   16    24     6 ]
                          [ - --    --   - - ]
                          [   35    35     7 ]
```

The surprising pattern in the values of the double integral of equation (2) led to a recursion relation (note that AA[1,2]=AA[1,1]*AA[2,2], etc.) and a formula to evaluate it analytically for all values of N and K. Further examination shows that the matrix "BET" leads to a system where each row can be cancelled, except for the diagonal term, by a multiple of the next. This diagonal system was explicitly solved to yield a closed form result for the B[2N+1]. For some details of this, consult the paper by Greenspan [2].

III - Additional Comments

As is fairly typical in mathematics, the problem as originally stated (and which was, in fact, also solved) was of no interest because the equations were erroneous. In fact, that use of MACSYMA was somewhat more elegant, because an approximation to a non-rational portion of the integrals was computed using power series. However, the values of B[2N+1] which were produced by that formulation did not converge, but oscillated and grew. This unsatisfactory result suggested that MACSYMA was being asked the wrong questions, as indeed was the case. A corrected version of the equations leads, as we have seen, to a decreasing and convergent series.

IV - Conclusions

We would like to briefly point out a few of the facilities in MACSYMA which make it useful to a working applied mathematician.

(1) Incorrect and/or useless formulations can be discarded more rapidly by following such formulations through to symbolic results (or numerical results) in sufficient quantity to answer basic questions concerning the behavior of a solution.

(2) Partial results can be examined immediately to determine procedures to be followed in the next steps.

(3) Large numbers of techniques can be drawn upon, without explicit user definition. For example, the presence of GAMMA function evaluation and simplification routines, the INTEGRATE command, the SOLVE command.

(4) The ability to set up hierarchies of programs which duplicate the sequence of typed in commands, once a sequence is, in fact, well-defined, makes it easy to initiate complicated calculations in a simple fashion. Furthermore, the extension to more terms or higher accuracy is usually extremely simple.

Acknowledgments

Work reported herein was supported in part by Project MAC, an M.I.T. interdepartmental laboratory sponsored by the Advanced Research Projects Agency, Department of Defense, under Office of Naval Research Contract N0014-70-A-0362-001, and in part by the Department of Mathematics, M.I.T. under National Science Foundation Grant GP 22796.

I would like to thank H. P. Greenspan for introducing me to this problem, and M. Israeli for his suggestions concerning various modifications of this problem and its solution.

References

[1] Bogen, R., et al, The MACSYMA Manual, Project MAC, M.I.T.

[2] Greenspan, H. P., "On Alpha-dynamos", J. of Applied Math. (to appear).