J. Symbolic Computation (1988) **5**, 377–384

An Algebraic Treatment of Quantum Vibrations Using REDUCE

MARTIN L. SAGE

Department of Chemistry, Syracuse University, Syracuse, New York 13244–1200, *U.S.A.*

(*Received 29 December* 1986)

Many eigenvalue problems that arise in mathematical physics and chemistry can be solved using the algebra of non-commuting operators. REDUCE can readily be used to perform the many time consuming algebraic manipulations necessary to find the solutions to these problems. Applications of REDUCE are given to the quantum mechanical one and two dimensional nearly harmonic oscillator using an algebraic approach.

Introduction

In a recent issue of the *Journal of Symbolic Computation* Fitch (1985) discussed the solution of a variety of algebraic problems using REDUCE. He also gave a table of various systems which support REDUCE. In this paper we use REDUCE to solve equations based on non-commuting operators. This ability was not discussed by Fitch and has wide applicability in problems that occur in physics and chemistry.

To illustrate the method we take an example from the quantum theory of vibrations to demonstrate the power of REDUCE in facilitating the lengthy but fundamentally simple algebraic manipulations that must be carried out in solving eigenvalue problems. The harmonic oscillator and a perturbation technique, the van Vleck transformation (see for example Kemble, 1937) are translated into algebraic form which can readily be implemented in REDUCE. Applications are made to the one-dimensional and to the two-dimension isotropic harmonic oscillator with quartic perturbation.

All programs presented in this paper used REDUCE 3.2 run on a DEC/VAX 8650 virtual memory computer with VMS 4.5 operating system in batch mode. A typical run used 1 M-byte of core memory and 3 M-bytes of additional virtual memory.

The potential function used in this paper was chosen to simplify the presentation and does not accurately describe molecular vibrations. The same procedures can be used for more realistic potentials. Application to actual molecular systems will be presented elsewhere.

The Harmonic Oscillator in One-dimension

The Schrödinger equation for the one dimensional harmonic oscillator, in appropriate units is

$$-\frac{d^2\Psi}{dx^2}+x^2\Psi=E\Psi$$

0747-7171/88/030377+08 $03.00/0

where the wave function Ψ vanishes at $\pm\infty$. Eigenvalues and eigenfunctions of this equation can be found by the polynomial method (Murphy, 1960). An entirely equivalent procedure is to replace the operator $-d^2/dx^2+x^2$ on the left hand side of this equation by the abstract operator

$$H = p^2+x^2$$

and find solutions to the equation

$$H|\Psi\rangle = E|\Psi\rangle$$

using only the commutation relations of the self-adjoint operators p and x

$$[p, x] = -i.$$

We introduce two operators

$$a = (x+ip)/\sqrt{2}$$

and

$$a^+ = (x-ip)/\sqrt{2}$$

where a^+ is the adjoint of a (see for example Merzbacher, 1961). The hamiltonian H can be written in terms of a non-negative, self-adjoint operator N,

$$H = 2N+1$$

where the operators a, a^+, and N obey the following commutation relations:

$$[a^+, a] = -1$$

$$[N, a] = -a$$

$$[N, a^+] = a^+.$$

Using these relations one can readily show that if $|\Psi\rangle$ is an eigenfunction of H with eigenvalue E then

$$Ha|\Psi\rangle = (E-1)a|\Psi\rangle$$

and

$$Ha^+|\Psi\rangle = (E+1)a^+|\Psi\rangle.$$

The operators a^+ and a are called raising and lowering operators, respectively, since when applied to an eigenfunction of H each raises or lowers the eigenvalue by 1. H is a positive operator since N is non-negative and therefore has a least eigenvalue with eigenfunction $|0\rangle$. The lowering operator applied to this eigenfunction must vanish identically. Since $N = a^+a$ then $N|0\rangle = 0$ and $E_0 = 1$. Thus the eigenvalues of H are $2n+1$ for all non-negative integers n with normalised eigenfunctions

$$|n\rangle \equiv (n!)^{-\frac{1}{2}}a^{+n}|0\rangle.$$

This method is related to the factorisation procedure used by Schrödinger (1941) in solving the harmonic oscillator problem in wave mechanics. He gave explicit expressions for a and a^+ in terms of x and d/dx. Infeld & Hull (1951) have applied the factorisation method to a variety of second order differential equations.

The One-dimensional Anharmonic Oscillator

We can now use these functions to find approximate eigenfunctions and eigenvalues for the one dimensional anharmonic oscillator with potential

$$V(x) = x^2 + \lambda x^4$$

where λ is a small parameter.† The Schrödinger equation for this potential can be written

$$(H_0 + \lambda H_1)|\Psi\rangle = E|\Psi\rangle$$

where H_0 is the harmonic oscillator hamiltonian given above and

$$H_1 = (a^+ + a)^4/4.$$

We now look for a unitary transformation of H which eliminates terms that are first order in λ and connect states with differing values of n. For a transformation of an operator O of the type

$$O' = U^+ O U$$

where

$$U = \mathrm{e}^{i\lambda S}$$

with self-adjoint operator S, the transformed operator can be written as a sum of repeated commutators (Merzbacher, 1961) as follows:

$$O' = \sum_{n=0}^{\infty} \frac{\lambda^n i^n}{n!} [O, S]_n$$

where

$$[O, S]_0 \equiv O$$

and

$$[O, S]_n \equiv [[O, S]_{n-1}, S] \quad (n \geqslant 1).$$

Carrying out this transformation on H and collecting terms with the same power of λ yields

$$H' = H_0 + \sum_{n=1}^{\infty} \frac{\lambda^n i^n}{n!} ([H_0, S_1]_n - in[H_1, S_1]_{n-1})$$

where we have written S_1 to indicate that this transformation is first order in λ. For convenience we write the coefficient of λ^n on the right hand side of this equation as H'_n. The operator S_1 is chosen so that the first-order terms arising from H_0 cancel the first-order off-diagonal terms from H_1. This transformation, called a van Vleck transformation, is particularly useful when applied to a group of degenerate or nearly degenerate states.

An explicit expression for S_1 will be given in the next section. The transformed hamiltonian contains terms to all orders in λ. The lowest order off-diagonal terms are second order in λ. For the correct S_1 we can write

$$H'_n = \frac{n-1}{n!} i^{n+1} [H_1, S_1]_{n-1} \quad (n \geqslant 2).$$

Evaluating the repeated commutators of H_1 and S_1 can easily be done using REDUCE.

† Since the operator x^4 is unbounded the perturbation expansion will be, at best an asymptotic series.

Eigenvalues are correct to second-order if we drop all off-diagonal terms in H'. To find eigenvalues to fourth-order we simply transform H' using a unitary operator

$$U_2 = e^{i\lambda^2 S_2}$$

which eliminates all the second-order off-diagonal terms. We can continue eliminating third-order off-diagonal terms in the newly transformed hamiltonian to get eigenvalues to sixth-order and so forth. The algebra gets more involved but is not substantially different.

One-dimensional Anharmonic Oscillator Using REDUCE

In order to implement the van Vleck perturbation procedure described above we must first define all the needed operators. The first line of the program shown in Fig. 1 names all the operators to be used while the second line indicates that multiplication of any of these operators need not be commutative. The commutation relations of the operators a, a^+, and n are given in the next several lines where these operators are called $a(\ )$, $ad(\ )$, and $n(\ )$ respectively. All operators must be followed by closed parentheses. Various labels separated by commas may occur within the parentheses. Note that according to the rules all factors of a^+ will be moved to the left of any operator expression while all factors of n will be moved to the right. Any occurrence of a^+a will be replaced by n.

All terms in the hamiltonian are denoted by $h(m, l)$. The original harmonic oscillator hamiltonian H_0 is called $h(0, 0)$ while the perturbation H_1 is the sum of all terms $h(1, k)$. The index k indicates the change in eigenvalue of n produced by $h(1, k)$. Likewise the first transformation operator, S_1, is the sum of terms written $s(1, k)$ and is generated by a simple "for ... until ... do" statement. Due to the form of H_0 and H_1 a solution to the commutator equation for S_1 can be written down explicitly. Each term $s(1, k)$ depends only on $h(1, k)$. All the terms arising from multiple commutators of H_1 with S_1 are evaluated with "for ... until ... do" statements. Results are given as $h(m, k)$ where m indicates the power of λ and k has the same meaning as above. The second order terms took 2 seconds to evaluate while the third and fourth order took 14. To find the eigenvalue correct to order λ^4 requires finding S_2 which is no more difficult than finding S_1 once we have found H_2. The fourth order correction to the energy required an additional 2·5 seconds. The entire REDUCE program took under 22 seconds of CPU time to execute.

The energy contributions through fourth order are printed at the end of the output log in Fig. 1 and are called $hd(k)$ for $k \leqslant 4$. The energy to fourth order is the sum

$$HD(0)+\lambda HD(1)+\lambda^2 HD(2)+\lambda^3 HD(3)+\lambda^4 HD(4).$$

With the exception of the first term the series alternates sign as has been shown by Loeffel *et al.* (1969).

A REDUCE program to translate the terms into FORTRAN-readable form is given in Fig. 2. This program assumes the REDUCE operators are stored in a file *HD*.RED and the FORTRAN variables will be written into a file *HD*.FOR.

Using REDUCE we have found an expression for the energy in terms of the number operator n which has simple eigenvalues. This expression is an effective hamiltonian for the anharmonic oscillator which would have been difficult to obtain by performing the algebraic manipulations by hand.

Two-dimensional Anharmonic Oscillator Using REDUCE

In this section we give a similar program to find the eigenvalues of the two-dimensional isotropic harmonic oscillator with a quartic perturbation $\lambda x^2 y^2$. This problem has

```
operator a, ad, h, hd, n, s$
noncom a, ad, h, hd, n, s$
a( )*ad( ) := ad( )*a( )+1$
ad( )*a( ) := n( )$
n( )*a( ) := a( )*n( )-a( )$
n( )*ad( ) := ad( )*n( )+ad( )$
j^2 := -1$%defines j to be the square root of -1 which prints as j
h(0, 0) := 2*n( )+1$
h(1, 4) := ad( )^4/4$
h(1, 2) := 1/4* for k := 0 : 3 sum ad( )^k*a( )*ad( )^(3-k)$
h(1, -2) := 1/4* for k := 0 : 3 sum a( )^k*ad( )*a( )^(3-k)$
h(1, -4) := a( )^4/4$
on time$%Next "TIME" is cumulative CPU time. Others are for one step.
TIME: 390 MS
h(1, 0) := (ad( )+a( ))^4/4 - for k := -4 step 2 until 4 sum k*h(1, k)/k$
TIME: 220 MS
for k := -4 step 2 until 4 do s(1, k) := j*k*h(1, k)/2/k^2$
TIME: 70 MS
comment. This step yields s(1, k) which satisfies
[h(0, 0), s(1, k)] = j*h(1, k)
  for k unequal to 0. s(1, 0) equals 0$
for k := -6 step 2 until 6
  do h(2, k) := j/2* for l := max(k-4, -4) step 2 until min(k+4,4)
  sum h(1, k-l)*s(1, l)-s(1, l)*h(1, k-l)$
TIME: 2300 MS
for n := 2 : 3 do for k := (-4-2*n) step 2 until (4+2*n)
  do h(n+1, k) := j*n/(n+1)/(n-1)*
  for l := max(k-4, -4) step 2 until min(k+4, 4)
  sum h(n, k-l)*s(1, l)-s(1, l)*h(n, k-l)$
TIME: 14200 MS
for k := -6 step 2 until 6 do s(2, k) := j*k*h(2, k)/2/k^2$
TIME: 110 MS
for n := 0 : 3 do hd(n) := h(n, 0)$
TIME: 30 MS
hd(4) := h(4, 0)
  +j/2* for k := -6 step 2 until 6 sum h(2, k)*s(2, -k)-s(2, -k)*h(2, k)$
TIME: 2530 MS
for n := 0 : 4 do write hd(n) := hd(n);
HD(0) := 2*N( )+1
HD(1) := (3*(2*N( )^2+2*N( )+1))/4
HD(2) := (-34*N( )^3-51*N( )^2-59*N( )-21)/16
HD(3) := (145*N( )^4+290*N( )^3+554*N( )^2+409*N( )+132)/32
HD(4) := (-12882*N( )^5-32205*N( )^4-89564*N( )^3-102141*N( )^2
         -73942*N( )-21021)/1024
TIME: 170 MS

Accounting information:
Buffered I/O count:          92    Peak working set size:      2048
Direct I/O count:            74    Peak page file size:        6644
Page faults:              11125    Mounted volumes:               0
Charged CPU time:  0 00 : 00 : 21.71  Elapsed time:  0 00 : 01 : 27.27
```

Fig. 1. Log for REDUCE batch job for the one-dimensional anharmonic oscillator.

```
in"hd.red"$
n( ) := n$
on fort$
out"hd.for"$
for k := 0 : 4 do write hd(k) := hd(k);
shut"hd.for"$
```

Fig. 2. A REDUCE program to generate FORTRAN output.

```
operator a, ad, b, bd, h, hd, na, nb, s, x, y$
noncom a, ad, b, bd, h, hd, na, nb, s, x, y$
comment ad( ) produces exp(i phi) factor, bd( ) produces exp(-i phi)$
a( )*ad( ) := ad( )*a( )+1$b( )*bd( ) := bd( )*b( )+1$
ad( )*a( ) := na( )$bd( )*b( ) := nb( )$
na( )*a( ) := a( )*na( )-a( )$nb( )*b( ) := b( )*nb( )-b( )$
na( )*ad( ) := ad( )*na( )+ad( )$nb( )*bd( ) := bd( )*nb( )+bd( )$
b( )*a( ) := a( )*b( )$b( )*ad( ) := ad( )*b( )$b( )*na( ) := na( )*b( )$
bd( )*a( ) := a( )*bd( )$bd( )*ad( ) := ad( )*bd( )$bd( )*na( ) := na( )*bd( )$
nb( )*a( ) := a( )*nb( )$nb( )*ad( ) := ad( )*nb( )$nb( )*na( ) := na( )*nb( )$
hd(0, 0) := h(0, 0) := 2*na( )+2*nb( )+2$
j^2 := -1$% defines j to be the square root of -1 which prints as j
x(1, 1) := ad( )/2$x(-1, 1) := b( )/2$x(1, -1) := bd( )/2$x(-1, -1) := a( )/2/$
y(1, 1) := ad( )/2/j$y(-1, 1) := b( )/2/j$y(1, -1) := -bd( )/2/j$y(-1, -1) := -a( )/2/j$
on time$%The next two steps generates x^2 and y^2
TIME: 530 MS
for n := -2 step 2 until 2 do for l := -2 step 2 until 2
  do x(n, l) := for np := -1 step 2 until 1 sum for lp := -1 step 2 until 1
  sum if abs(n-np) = 1 and abs(l-lp) = 1 then x(np, lp)*x(n-np, l-lp)$
TIME: 540 MS
for n := -2 step 2 until 2 do for l := -2 step 2 until 2
  do y(n, l) := for np := -1 step 2 until 1 sum for lp := -1 step 2 until 1
  sum if abs(n-np) = 1 and abs(l-lp) = 1 then y(np, lp)*y(n-np, l-lp)$
TIME: 490 MS
for n := -4 step 2 until 4 do for l := -4 step 4 until 4 do
h(1, n, l) := for np := -2 step 2 until 2 sum for lp := -2 step 2 until 2
  sum if abs(n-np) <= 2 and abs(l-lp) <= 2 then x(np, lp)*y(n-np, l-lp)$
TIME: 2620 MS
for n := -4 step 2 until 4 do for l := -4 step 4 until 4
  do s(1, n, l) := j*n*h(1, n, l)/2/n^2$
TIME: 290 MS
comment. This step yields s(1, n, l) which satisfies
  [h(0, 0), s(1, n, l)] = j*v(1, n, l) where v(1, n, l) equals h(1, n, l) for
  n unequal to 0 and if 0 otherwise. s(1, 0, l) equals 0$
for l := -4 step 4 until 4 do hd(1, l) := h(1, 0, l)$
TIME: 50 MS
for n := -6 step 2 until 6 do for l := -4 step 4 until 4
  do h(2, n, l) := j/2* for np := -4 step 2 until 4
  sum for lp := -4 step 4 until 4
  sum if abs(n-np) <= 4 and abs(l-lp) <= 4 then
  h(1, np, lp)*s(1, n-np, l-lp)-s(1, n-np, l-lp)*h(1, np, lp)$
TIME: 71760 MS
for l := -8 step 4 until 8 do hd(3, l) := j*2/3* for n := -4 step 2 until 4
  sum for lp := -4 step 4 until 4 sum if abs(l-lp) <= 4 then
  h(2, n, lp)*s(1, -n, l-lp)-s(1, -n, l-lp)*h(2, n, lp)$
TIME: 52450 MS
hd(0, 0) := hd(0, 0);
HD(0, 0) := 2*(NA( )+NB( )+1)
TIME: 30 MS
for l := -4 step 4 until 4 do write hd(1, l) := hd(1, l);
HD(1, (-4)) := (-3*(A( )^2*BD( )^2))/8
HD(1, 0) := (NA( )^2+4*NA( )*NB( )+3*NA( )+NB( )^2+3*NB( )+2)/8
HD(1, 4) := (-3*(AD( )^2*B( )^2))/8
TIME: 120 MS
for l := -4 step 4 until 4 do write hd(2, l) := h(2, 0, l);
HD(2,(-4)) := (17*(A( )^2*BD( )^2*NB( )+A( )^2*BD( )^2+A( )^2*NA( )*BD( )^2)
              /128
HD(2, 0) := (-NA( )^3-33*NA( )^2*NB( )-18*NA( )^2-35*NA( )*NB( )^2
            -64*NA( )*NB( )-29*NA( )-NB( )^3-18*NB( )^2-29*NB( )
            -12)/128
HD(2, 4) := (17*AD( )^2*B( )^2*NB( )+AD( )^2*B( )^2+AD( )^2*NA( )*B( )^2))/128
TIME: 210 MS
for l := -8 step 4 until 8 do write hd(3, l) := hd(3, l);
HD(3,(-8)) := (145*(A( )^4*BD( )^4))/6144
```

Fig. 3.

```
HD(3,(-4)) := (-313*A( )^2*BD( )^2*NB( )^3-281*A( )^2*BD( )^2*NB( )
              +66*A( )^2*BD( )^3-294*A( )^2*NA( )^2*BD( )^2
              -1176*A( )^2*NA( )*BD( )^2*NB( )-1458*A( )^2*NA( )*BD( )^2)/6144
HD(3, 0) := (NA( )^4+648*NA( )^3*NB( )+322*NA( )^3+1995*NA( )^2*NB( )^2
            +2821*NA( )^2*NB( )+1175*NA( )^2+666*NA( )*NB( )^3
            +2893*NA( )*NB( )^2+3417*NA( )*NB( )+1286*NA( )+2*NB( )^4
            +328*NB( )^3+1186*NB( )^2+1292*NB( )+432)/6144
HD(3, 4) := (-297*AD( )^2*B( )^2*NB( )^2-1455*AD( )^2*B( )^2*NB( )
            +102*AD( )^2*B( )^2-290*AD( )^2*NA( )^2*B( )^2
            -1202*AD( )^2*NA( )*B( )^2*NB( )-234*AD( )^2*NA( )*B( )^2)/6144
HD(3, 8) := (145*(AD( )^4*B( )^4))/6144
TIME: 410 MS
```

Accounting information			
Buffered I/O count:	95	Peak working set size:	2048
Direct I/O count:	90	Peak page file size:	6644
Page faults:	101028	Mounted volumes:	0
Charged CPU time:	0 00 : 02 : 11.34	Elapsed time:	0 00 : 02 : 29.39

Fig. 3. Log for REDUCE batch job for the two-dimensional anharmonic oscillator.

recently been treated by Killingbeck & Jones (1986). We choose to use polar coordinates although cartesians could also be used. We must define two raising and lowering operators; a^+, b^+, a, and b. The pair a^+, a raise or lower the number operator N_a while the pair b^+, b raise or lower N_b. The operators a^+ and b also multiply the wave function by $e^{i\phi}$ while a and b^+ multiply it by $e^{-i\phi}$. The raising, lowering, and number operators for a and b obey the same commutation relations as in the one-dimensional harmonic oscillator problem. Each a-operator commutes with each b-operator. As before, the first steps in the program, shown in Fig. 3, define the commutation relations of the operators. According to the rules all a-operators are moved to the left of all the b-operators.

Energy levels of the isotropic harmonic oscillator depend on the sum of the two quantum numbers n_a and n_b and not on n_a and n_b separately. The state vector $|n_a, n_b\rangle$ is an eigenfunction of the total number operator

$$N \equiv N_a + N_b$$

with $(n+1)$-fold degenerate eigenvalue $n = n_a + n_b$ and also of the angular momentum operator

$$L \equiv N_a - N_b$$

with eigenvalue $l = n_a - n_b$.

The operators written as $x(n, l)$ and $y(n, l)$ in the REDUCE program for $n = \pm 1$ and $l = \pm 1$ are the parts of x and y which change the total number operator by n and the angular momentum by l. Squares of x and y are written in a similar fashion with $n = \pm 2, 0$ and $l = \pm 2, 0$. The major difference between the programs is that these operators h and s contain three labels. The first refers to order, the second to changes in number operator, and the third to changes in angular momentum. Due to the four-fold symmetry of the perturbation all h and s connect states with changes in angular momentum equal to 0 mod 4. Each term $s(1, n, l)$ depends on $h(1, n, l)$. Calculations proceed as before except require considerably longer times; the full H_2 took 72 seconds and all the block-diagonal terms in H_3 corresponding to one value of total quantum number took an additional 52 seconds. The entire program to generate the effective hamiltonian through third order used 131 seconds of CPU time.

The effective hamiltonian for the two-dimensional problem is for states with total quantum number n is an $(n+1)$-dimensional matrix involving the number operators n_a and n_b and other operators which connect states which differ in angular momentum quantum number l by ± 4 or ± 8. For odd values of n all eigenvalues of the effective hamiltonian are doubly degenerate while for even values they are all non-degenerate. If we want to further simplify the results we could use REDUCE to generate different effective hamiltonians for states transforming like the irreducible representations of the symmetry group of this problem, D_4.

The examples we have given show that REDUCE is a simple and effective tool to use in performing involved operator manipulations. This ability makes it useful in solving many eigenvalue and other problems.

The author would like to thank the Department of Theoretical Chemistry of the University of Oxford and Dr. Mark S. Child for kind hospitality during the 1985–86 academic year.

References

Fitch, J. (1985). Solving algebraic problems with REDUCE. *J. Symbolic Computation* **1**, 211–227.
Kemble, E. C. (1937). *The Fundamental Principles of Quantum Mechanics*. New York: McGraw Hill.
Killingbeck, J., Jones, M. N. (1986). The perturbed two-dimensional oscillator. *J. Phys. A.* **19**, 705–710.
Infeld, L., Hull, T. E. (1951). The factorization method. *Rev. Mod. Phys.* **23**, 21–51.
Loeffel, J. L., Martin, A., Simon, B., Wightman, A. S. (1969). Pade approximants and the anharmonic oscillator. *Phys. Lett.* **30B**, 656–658.
Merzbacher, E. (1961). *Quantum Mechanics*. New York: John Wiley.
Murphy, G. M. (1960). *Ordinary Differential Equations and Their Solutions*. Princeton: Van Nostrand.
Schrödinger, E. (1941). Further studies on solving eigenvalue problems by factorization. *Proc. Roy. Irish Acad.* **A46**, 183–206.