1994 February version

SUPERSTRUCTURE — paper II

incomplete draft

by W Eissner, M Jones, P Storey
Department of Physics and Astronomy, University College London,
Gower Street, London WC1E 6BT, England

and

H Nussbaumer,
Eidgenössische Technische Hochschule,
Gruppe für Atom- und Astrophysik, Huttenstrasse 34,
CH-8006 Zürich, Switzerland.

Program Summary

Title of program: SUPERSTRUCTURE
Catalogue number:
Computer for which program is designed and upon which it is operable
Computer: CRAY-ymp8, RISC, VAX etc.

Previously the program has also been run on the following types of machines:
IBM360/65 at UCL
CDC 6600 and 6800, at University College London and
at the University of Colorado, Boulder, Colorado.
UNIVAC 1108, at the University of Miami, Coral Gables Florida;
CRAY-1 at Daresbury Laboratory.

Operating system or monitor under which program is executed: (UNI)COS
Programming language used: FORTRAN 77
High speed store required: > 100 kB. Number of bits in a word: 64.
Is program overlaid? yes, it was before paging became link standard
Number of magnetic tapes required: None.
What other peripherals are used? line printer.
Number of cards in combined program and test deck:
code: ABCDIC (029).

Keywords descriptive of problem and method of solution:
Atomic, atomic structure, relativistic corrections, term-coupling coefficients, radiative data, cascade coefficients.

Contents

1 Nature of the physical problem

The program can be used to carry out multiconfiguration atomic structure calculations for neutral atoms and positive ions. Non-relativistic term energies, gf-values and transition probabilities can be calculated. The program can also calculate various intermediate coupling data, in particular energy levels which include relativistic corrections. Term- coupling coefficients, which are required in the calculation of intermediate-couplng collision strengths from LS-coupling R-matrices (Saraph [14]). These coefficients can be punched out according to a format compatible with the input requirements of Saraph's program JAJOM [14]. Radiation data for electric multipole and magnetic dipole transitions are calculated in intermediate

coupling. Finally, using the electric dipole transition probabilities, cascade coefficients for the ground configuration may be calculated.

The standard SUPERSTRUCTURE code SSS can be modified or extended on linking the sources ZUERMODS or M1MODS, separately or together. When used with ZUERMODS — yielding ZSS — the SSS routines MINIM, RADIAL and DIAGON are replaced so as to incorporate the essential optimization feature of the Zuerich version of SUPERSTRUCTURE (Nussbaumer and Storey [12]) while preserving compatibility with ordinary SSS type input. M1MODS provides for corrections of Breit-Pauli order to radiative operators — e. g. see [8].

2 Method of solution

The algebraic parts of all matrix elements are evaluated by means of systematic algebraic techniques based upon expansions of the wavefunctions in Slater determinants. A common set of one-electron radial wavefunctions is usually employed for all terms; these radial functions may be of statistical model central field type (calculated within the program), or alternatively, may be functions chosen by the user and read in as data. Relativistic corrections are evaluated in the low-Z Breit-Pauli approximation.

3 Restrictions on the complexity of the program

The code is suitable for small work stations and, with restrictions, even on PC's. The amount of fast memory available on the computer used will limit the size of the arrays which store, for example, vector-coupling coefficients, Slater states and energy submatrices. If the allocated storage is exhausted, self-explanatory error messages suggesting remedial action will be printed.

4 Description of the routines

Its name is underlined if the subroutine communicates entirely through the argument list — as opposed to other routines involving common blocks, which may even be subject to preprocessing.

FORTRAN conventions apply when naming real and integer numbers or denoting arrays as implicit loops. In this paper we set FORTRAN variables in typewriter type, otherwise adhering to the italic type modern convention for variables so as to distinguish them from values (e. g. L=S). Thus obvious conflicts are avoided. Upper case roman type indicates quantities relevant to the linking step.

$MAIN calls ZERO, ALGEB, and MINIM.

ALGEB controls the algebraic branch, reading a set of configurations and calling the routines ALGEBi. It also contains the RESTART code if activated by STRPRO.

ALGEB1 organizes computation of Slater states and vector coupling coefficients associated with the arising symmetries SL.

ALGEB2 organizes the expansion of the hamiltonian in LS coupling, using FLGL as the work horse, and it works out the radiative algebra;

ALGEB3 does the same for the hamiltonian in intermediate coupling, calling SPOR and RES for each pair J^π in a configuration representation. DIPOLE, if activated and called after ALGEB3, then forms part of the same logical step.

CALCFX has been written to the requirements of VA04A: it constructs the variational functional `F` calling RADIAL and DIAGON.

CASC computes cascade coefficients in intermediate coupling.

DIAGFS evaluates the full hamilton matrix in intermediate coupling and radiative data for E1, E2 and M1 transitions between spectroscopic levels J^π.

DIAGON evaluates the full hamilton matrix in a spectroscopic term representation and computes radiative data; a variational functional $\mathcal{F}$ or $\mathcal{G}$ is also computed if required (when DIAGON is called via CALCFX and VA04A rather than directly from MINIM).

DIFF computes the first derivative of a function tabulated in array `A` on the standard mesh described for WEDDLE, writing the result into `B`.

DIPOLE is a dummy routine in the SSS source, for ready-made linking of M1MODS, where DIPOLE computes the algebra of terms in Breit-Pauli order for radiative operators. ALGEB calls it after ALGEB3 but *before* closing RESTART cycle 3. It may therefore be used for any new routine ALGEBi that complies with the obvious implications. For routines using additional arrays, however, an entry ALGEB4 after closing of cycle 3 could be introduced.

FLGL is the work horse for ALGEB2, computing the angular coefficients in the expansion of one H-matrix element in a configuration representation. Its salient feature is the tabulation array `VCA`, temporarily holding ...

GAMMA, a gamma function code for real argument, is always left in `SINGLE` PRECISION because only called by WHITEX for a rough-and-ready estimate of a suitable asymptotic normalization factor.

JACORD diagonalizes a real symmetric matrix using Jacobi's method. The code is robust but slow in large cases — hence optional resort to the NAG library when applied to the

ALGEB branch. `JACORD` prints 1-line messages — harmless unless VCU flags serious trouble and aborts the case — if the attainable accuracy is limited. This occurs frequently when using short word length, and the printed real number should be several orders of magnitude smaller than typical eigenvalues.

MINIM controls the analytic branch, reading input that specifies a member of the sequence defined in ALGEB and calling subroutines.

NUMERO, called from RADWAV, propagates solutions of (2) across `N` radial grid points using a Numerov formula.

RADIAL controls the computation of radial functions, a set of scaling parameters being the essential input; while iterating, only those wave functions that need be recalculated will be, along with associated integrals (two-body integrals and effective spin-orbit parameters are computed elsewhere, in DIAGON and SPOR respectively). One or several calls to RADWAV are usually preceeded by one to TFDAPO — large scale factors lead to contracted orbitals usable for correlation configurations.

Unlike SSS the ZSS version of RADIAL enables more efficient correlation orbitals to be computed in a potential

$$\mathcal{Z}_{\mathrm{eff}}^{\mathtt{PJS}}(r) = Z\left[\mathrm{e}^{-Zr/2} + \lambda_{nl}(1 - \mathrm{e}^{-Zr/2})\right] \tag{1}$$

(Nussbaumer and Storey [12] on specifying 'scale factors' $-(100n + \lambda_{nl})$. This is a scaled Coulomb potential marginally modified near $r = 0$ to preserve cusp-free orbitals (3). In both versions a value $\lambda = -1$ yields hydrogenlike orbitals and scaled ones as long as $-100 < \lambda < 0$.

RADWAV computes target orbital functions, integrating the wave equation

$$\left[\frac{\mathrm{d}^2}{\mathrm{d}r^2} - \frac{l(l+1)}{r^2} + \frac{2\mathcal{Z}_{\mathrm{eff}}(r)}{r} + \epsilon_{nl}\right] P_{nl}(r) = 0 \tag{2}$$

on a radial grid `X(I)`$=r_{\mathtt{I}}$ in a potential `POT(I)`$=\mathcal{Z}(r_{\mathtt{I}})/r_{\mathtt{I}}$ for given nl with bound state boundary conditions

$$\lim_{r\to 0} P_j(r) = \bar{P}_j(0) r^{l_j+1}\left\{1 - \frac{\mathcal{Z}(0)r}{l+1} + \mathcal{O}(r^2)\right\} \tag{3}$$

and

$$\lim_{r\to\infty} P_{nl} \sim r^{\nu}\cdot \mathrm{e}^{-zr/\nu} \qquad \text{where } \epsilon = -\frac{z^2}{\nu^2} \quad \text{and } z \equiv \mathcal{Z}(\infty) = \mathcal{Z}(r_{\max}) \tag{4}$$

and normalization

$$\int_0^\infty \mathrm{d}r\ P_{nl}^2(r) = 1 \tag{5}$$

It returns `E`$=\epsilon_{nl}$, writing $P_{nl}(r)$ and

$$Q_{nl}(r) \equiv \left[-\frac{\mathrm{d}^2}{\mathrm{d}r^2} + \frac{l(l+1)}{r^2} - \frac{2Z}{r}\right] P_{nl}(r), \tag{6}$$

into arrays `P` and `Q`; array positions (0) contain $\bar{P}_{nl}(0)$ and the similarly defined quantity

$$\bar{Q}_{nl}(0) \equiv \lim_{r\to 0}\frac{Q_{nl}(r)}{r^{l+1}} = \bar{P}_{nl}(0)\left\{2\frac{\mathrm{d}\mathcal{Z}(r)}{\mathrm{d}r}\Big|_{r=0} + \epsilon_{nl}\right\}. \tag{7}$$

The nuclear electric charge number Z is supplied in `POT(0)`, so that $\mathcal{Z}(0)$=`POT(0)` *except* in the scaled Coulomb case $\mathcal{Z}(r) = Z^*$ when special action is taken.

The routine follows standard procedure: outward integration from a power series expansion for the first few tabulation points, inward integration from two point evaluated by WHITEX as Coulomb functions, and matching close to the last point of inflection. Iterations terminate when $|\Delta\epsilon/\epsilon| <$ `TOL` for the correction $\Delta\epsilon$ to the previous ϵ. Normally 3–4 iterations suffice; but if the input value `TOL=X(0)` has been chosen too small for the machine word length the sign of $\Delta\epsilon$ will start oscillating — hence the limit `MAXIT=16` set in `PARAMETER`, accompanied by a message. Good input σ for the starting value $\epsilon^0 = (Z-\sigma)^2/n^2$ can cut on iterations, and the routine may work badly without if the orbital has many nodes, e. g. for 6s.

As in TFDAPO mere warnings flag terminated iterations, and they disappear when specifying a more realistic value for `ITOL` on machines with a short mantissa.

Unlike the standard version its ZUERMODS variant has code for a radial function output channel `|MQ|` — item (xi) on the Z-card.

RADWIN reads user-supplied orbital functions. It was designed to read untruncated input in COLALG/IMPACT format (Crees *et al.* [6]). The header record (`KEY=-9`) may however be used to signal other types of input, in particular STO parameters, when

$$P_{nl}(r) = \sum_{\mathtt{J}=1}^{\mathtt{M}} \mathtt{C(J)} \cdot r^{\mathtt{IRAD(J)}} \cdot \mathrm{e}^{-\mathtt{ZE(J)}\cdot r} \tag{8}$$

will be computed on the normal radial grid. Clementi-Roetti [5] type input is recognized on testing the normalization integral (5) and converted to STO form:

$$\mathtt{C(J)} = \mathtt{C}^{\mathrm{C-R}}\mathtt{(J)} * \sqrt{2 * \mathtt{ZE(J)} \prod_{\mathtt{K}=1}^{\mathtt{2*IRAD(J)}} \frac{2 * \mathtt{ZE(J)}}{\mathtt{K}}}. \tag{9}$$

RES is the work horse for ALGEB3, computing the two-body finestructure algebra for one pair of levels in a configuration representation. The use of tabulation arrays `VCA` and `VCB` is modelled on the salient feature of FLGL. While *computing* the angular coefficients in `DOUBLE` precision arithmetic if PRECISION has so been specified, it would be meaningless to *store* them at more than short word length, as they contribute only in Breit-Pauli order. Moreover they form the largest transfer block between the two principal branches in full scale calculations.

RK1ST, auxiliary to TFDAPO, propagates a potential equation over one Runge-Kutta step.

ROTSYM is the work horse of JACORD.

SJS is a function routine computing the value of a six-j-symbol — see also under VCC.

SPOR is another work horse for ALGEB3, computing the one-body finestructure algebra, i. e. the angular coefficients to the ordinary spin-orbit parameter in one H-matrix element in a configuration representation.

TFDAPO computes a Thomas-Fermi-Dirac-Amaldi potential $\mathcal{Z}_{\rm eff}(r_{\rm I})/r_{\rm I}$ =`POT(I)` for N=`N` electrons in the field of a nucleus with electric charge number Z=`POT(0)` and residual charge $\mathcal{Z}(r_{\max}) = Z - (N-1)$, subject to the scale factor λ = `ADJUST`: the potential is invariant on the length scale r/λ! The radial points $r_{\rm I}$, computed on the grid described for WEDDLE, are returned in `X(I)`, whereas `TOL=X(0)` is input to the routine — a message is printed when asking for less uncertainty in `POT(I),I=1,NPTS` than compatible with MACHINE word length: it will disappear when specifying a more realist value `ITOL` on the Z-card.

VA04A is a modified version of a procedure written by Powell [13] which determines `N` variational parameters on optimizing a functional `F` computed by a subroutine CALCFX.

VCC is a function routine computing the value of a Clebsch-Gordan coefficients. Like SJS it is an improved version of the Copenhagen package routine with the same name, in particular exploiting the original idea of a factorial array `FAC(I)=(I-1)!/C**(N-1)` but instead of 10 choosing for `C` a suitable power of 2 such as 16, which reduces rounding errors in computed angular coefficients by one order of magnitude — this makes all the difference in calculations with short mantissae! Moreover the addressing algorithm has been made free of divisions, on storing `FAC(I)=(I/2-1)!/16**(I/2-1),I=2,MAXF,2` at even values `I` — and as a bonus alternating phase factors +1 and –1 are available in positions with odd index values.

VCE, auxiliary to VCU, calls routines for diagonalizing symmetric real matrices (which tend to be highly degenerate): depending upon the directive parameter DIAGONALIZATION STRPRO switches between `JACORD` and `NAGLIB` (NAG library routines).

VCU computes Slater states and term coupling coefficients for a given configuration C_k.

WEDDLE, written by Belling [1], uses Weddle's rule to obtain the integral `B` over a function tabulated in an array `A`; the step length doubles from one to the next of `JH` intervals — number of steps and step length are given in arrays `HN` and `NH`, and the value of the integrand at argument 0 in `A(0)`.

WHITEX, a function routine auxiliary to RADWAV, uses a Whittaker expansion to compute the value of a Coulomb function — normalized if so told (a feature immaterial for SSS).

YLAMK is a modified and extended version of YLAM written by Belling [1] to compute, on the grid described for WEDDLE, multipole potential functions

$$y_\lambda(nl, n'l'; r) = \frac{1}{r^{\lambda+1}} \int_0^r \mathrm{d}r' P_{nl}(r') r'^\lambda P_{n'l'}(r') + r^\lambda \int_r^\infty \mathrm{d}r' P_{nl}(r') \frac{1}{r'^{\lambda+1}} P_{n'l'}(r') \qquad (10)$$

if `MODE=0`. This parameter has been added to YLAM to compute kernel functions associated with the magnetic two-body integrals: $N^\lambda(ab, cd)$ when set > 0, and $V^\lambda(ab, cd)$ when set

< 0, while `MODE=0` leads to ordinary Slater integrals

$$R^\lambda(ab,cd) = \int_0^\infty \mathrm{d}r P_a(r) y_\lambda(b,d;r) P_c(r)\,. \tag{11}$$

ZERO initializes the code.

5 General outline for coding orbitals

Orbitals (nl) are internally referred to by a single positive integer $\gamma(nl)$: 1s, 2s,... 4d, 4f, 5s ... by 1, 2,... 9, 10, 11... Thus the array element `QCG(I,K)`, in common /DBD2/, gives γ for electron `I` of configuration `K`. On the records specifying configuration input, however, orbitals nl will be referred to by `1, 2, ...9, A, B...`, this is by one alphameric character.

orbital	nl	1s	2s	2p	...	4d	4f	5s	...	5g	6s	...
'internal'	$\gamma(nl)$	1	2	3		9	10	11		15	16	
'input'	$\alpha(\gamma(nl))$	1	2	3		9	A	B		F	G	

There was no distinction between the two reference levels 'internal' and 'input' when the program was developed on an IBM 360 computer: then format `Z1` was used to read γ, and the number of distinct orbitals was restricted to 15. But format `Z1` has been changed to A1 since.

The above (nl)-table is subject to size restrictions by the STRING variable x`MAXGR-MAXo`(γ). Loop `DO 33` in SR.ALGEB1 assigns $n_\gamma =$`QN`(γ) and $l_\gamma =$`QL`(γ); this default assignment, which should serve most purposes, can be overwritten on the C-terminator card (see later) for the first 15 values of γ; e. g. we may redefine $\gamma(6s) = 15$. Line 'input' of the table is related to 'internal' by array LIT as quoted in DATA in SR.ALGEB1. If `MAXGR`> 16 is required then `LIT` will have to be adjusted in `DIMENSION`, and `DATA` extended in the code; as for `LIT` of `DATA` one could continue with the alphabet, or switch to some other convenient notation. More extensive recoding, both in SR.ALGEB1 and in SR.ALGEB, is necessary if one wants to redefine indices $\gamma > 17$. The default assignment can be written in closed form:

$$\gamma(nl) = \frac{(n-1)\cdot n}{2} + l + 1$$

Tabular printout of SUPERSTRUCTURE quotes orbitals (nl) in the 'internal' notation; thus five integers suffice to represent the quantum numbers that define a Slater integral $R_\lambda(n^a l^a n^b l^b, n^c l^c n^d l^d)$. The alternative level 'input' applies only to data input; only one column has been reserved on cards to specify a value (nl). Having pointed out the distinction between γ and $\alpha(\gamma)$, we will mostly refer to (nl) on both levels just by γ (a practice similar to using Pauli's σ_i or Dirac's γ^ι even where one deals with a matrix representation D(σ_i) or D(γ^ι).

6 Data requirements

The data input may be grouped into two main parts corresponding to the two principal branches, namely the algebraic and analytic branches.

6.1 The algebraic branch: data read by subroutine ALGEB

Essentially a list of configurations is read on one record or a set of records. Input to this branch is terminated by a record which is blank in columns 10 and 11. This terminator may consist simply of a blank record or it may contain additional (but optional) input. A detailed description of these records follows.

α) Card type C.
`Format:  I2, I2, 2I2, 1X, 21(I2,A1), A8`
The data that should appear according to this format is
`T1,I2` which specifies the printing level of MPRINT in the algebraic branch.
If `MPRINT=2` then Slater states and vector coupling coefficients associated with the SL terms of each input configuration C_k are printed, in addition to the printout produced on level `MPRINT=1`. If `MPRINT=1` then the expansion coefficients A and B of equation(EJN·6) together with the expansion coefficients of reduced electric multi-pole elements (see (EJN·107–108)) are printed.

If `MPRINT`≤ 0 then the output described under levels 1 and 2 is suppressed. A blank or zero should be the normal choice. In this case the program prints only a short section on each input configuration, also telling how much array storage has been used up. `MPRINT=-1` reduces such printout to a bare minimum (recommend unless size problems could be expected). There are two more levels of `MPRINT`, both of which have the effect of skipping parts of the code:
`MPRINT=-2` is the choice if one does not want radiative transition data.
`MPRINT<-2` is the 'size-checking only' mode. All time-consuming sections of the code are bypassed, but array size checks are still performed and messages printed where appropriate. It is good practice to begin a big case at level `MPRINT<-2`, and after recompilation with adjusted sizes to change over to `MPRINT>-2` only when the printout looks clean. `MPRINT<-3` has the effect of STOPping the run if the case caused error messages, while in all levels `MPRINT` ≥ -2 the program moves on to the next case, skipping input records that are not processable for lack of storage.
`T3,I2` gives the level MOD of compiling Slater states and VCC's.
Storage requirements depend upon the choice of `MOD`. They reduce with increasing `MOD`, and they are larger for negative values of `MOD` than for the corresponding positive values. Table 3 indicates the problems which can be solved with the different choices of `MOD`: in this table

'SL' represents the non-relativistic SL-coupling problem, and 'SLJ' represents the relativistic problem. 'Structure' and 'radiative' refer to calculations of energies only and of energies and radiative data respectively.

Table 3

Case Mod	*SL* struct.	*SL* rad.	*SLJ* struct.	*SLJ* rad.	azimuthal quantum numbers
–2	yes	yes	no	no	$\begin{cases} \text{all} M_L \geq 0 \\ M_S \geq \min(\lvert M_S \rvert) \end{cases}$
–1	yes	yes	yes	yes	$M_J \geq \min(\lvert M_J \rvert)$
0^+	yes	yes	yes	yes	all M_L, M_S, M_J
1	yes	yes	yes	no	$M_J = \min(\lvert M_J \rvert)$
2	yes	no	no	no	$\begin{cases} M_L = 0 \\ M_S = \min(\lvert M_S \rvert) \end{cases}$
3^*	yes	no	no	no	$\begin{cases} M_L = 0 \\ M_S = \min(\lvert M_S \rvert) \end{cases}$

struct. = structure, rad. = radiative

+: and `KCUT=0` (see later under 'C-terminator'); otherwise restrictions apply, useful in collisional work, and SUPERSTRUCTURE automatically resets `MOD=0` to –1.

*: a single type of term SL is retained, whose value (SL) must be defined on the C-terminator record.

Level `MOD` can be used to turn off the code involved in intermediate coupling (problem 'SLJ'); this is good practice as long as one is searching for the best target representation. Unlike the option `MPRINT=-1` level `MOD` is not suitable to go around the code computing radiative transitions, because in certain cases this data can be obtained with a more restrictive level than indicated by table 3; the program always attempts to do so. It should also be mentioned that the program will perform all structure computations if array storage is lacking for radiative data. Always choose the negative rather than the positive level MOD wherever storage allows, because this saves computer time, and quite substantially so in FS-calculations involving magnetic two-body terms between many levels.

T5,2I2 specifies the core C_0.

A set of closed shells $(nl)^{2(2l+1)}$ may be specified as a core C_0 attached to all configurations, $C_k = C_0 c_k$; e. g. ␣1␣1 = $1s^2$, ␣1␣2 = $1s^2 2s^2$, ␣1␣3 = $1s^2 2s^2 2p^6$.

According to these examples all orbitals nl whose 'internal' labels $\gamma(nl)$ are within the specified range $\gamma_i - \gamma_f$ will be included in C_0. Note that 'internal' rather than 'input' labelling must be used for specifying C_0 on the first C-record. If one redefines the γ associated with an orbital nl, which will be described later, C_0 changes accordingly; suppose we had redefined $\gamma(2s) = 1$, $\gamma(2p) = 2$, and $\gamma(1s) = 3$; then the three examples of input above would result in $C_0 = 2s^2$, $2s^2 2p^6$, $1s^2 2s^2 2p^6$ respectively.

`T10,21I2,A1)` specifies configurations c_k.
We define

$$C_k = C_0 c_k = C_0 \prod_i^m \gamma_i^{q_i} \equiv C_0 \gamma_1^{q_1} \gamma_2^{q_2} \cdots \gamma_m^{q_m}.$$

`T10,21(I2,A1` holds up to 21 sets $(\bar{q}, \alpha)$ per C-card. Here $\alpha = \alpha(\gamma(nl))$ is the alphameric 'input' label for orbital nl, and $q \cong \bar{q}$ modulo 50 is the number of orbitals of type γ; q_i is incremented by 50 to indicate the second and subsequent terms (i. e. $i \geq 2$) in the "product" $\prod$. Examples are ␣`23` = $2p^2$; ␣`23514` = $2p^2 3s$.
The following example specifies more than one configuration:
␣`22523`␣`12523516` = $2s^2 2p^2$, $2s 2p^2 3d$.
According to this coding a new configuration ($k := k + 1$) is separated from the preceeding one when $q \leq 50$ (which means for most systems the first figure will be a cipher (blank)).

`T73,A8` gives heading information
This information will be printed at the beginning of the first line for identification purposes.

More cards C
Configurations can be continued on additional records, using format `9X,21(I2,A1)`. If one wishes so one can specify each configuration on a separate record. However, in large cases one risks without need to exceed the buffer for the (q, α)'s, because each C-card takes up 21 locations. A blank card terminates the C-input.

Systems consisting of closed shells only
In this case the configuration information cannnot be transmitted via the 6^{th} and 8^{th} columns alone, because the code assumes at least one electron outside the core C_0; this is reflected in the data card input: blank columns 10 and 11 on the first C-card of a case make this card a C-terminator, which STOPs the run. Hence at least one of the closed shells must be treated as a valence shell, denoted by c_k. Thus a configuration $C_k = C_0 c_k = 1s^2 2s^2$ may be specified by
␣␣␣␣ ␣`1`␣`1` ␣ ␣`22` on a first C-card.

The C-terminator card
Usually the terminator card is blank. However, it may be necessary to supply additional information on this card.

(i) The RESTART parameter `MSTART`, to be specified in `(T1,I2)`, controls the restart facility, which is fully described in section 7. If in a previous run algebraic results up to those computed in branch ALGEBi have been saved on a file RESTART, work may be resumed on specifying `MSTART=`$i+1$. Examples are `MSTART=1`, which is the creation run mode, and `MSTART=4`, which is used when the whole algebra of an isoelectronic case is already on file RESTART. All the RESTART statements will be bypassed if `MSTART` is left blank or set to 0.

(ii) Columns `(T3,I2)`, for specifying a parameter `IFREE`, have been reserved for controlling calls to SR.DISKDC, which selects Slater state sets and VCC'S from a library (???). This facility, however, is not part of the published standard version of SUPERSTRUCTURE. The control statements have been inactivated by the prefix 'CDISK'.

(iii) In case `MOD=3`, the value (SL) to be selected must be specified according to $\mathcal{C}$`(T5,I2,I2)` = $(2S, 2L)$.

(iv) `KCUT`, a parameter which is equal to the number of configuration which are to be treated as spectroscopic configurations can be read according to format `(T13,I2)`. The remaining configurations $C_0 c_k$, which have $k >$`KCUT`, will be treated as correlation configurations, that is terms not contained in the first `KCUT` configurations, are discarded. All terms are retained if `KCUT` is left blank (or =0).

(v) The γ coding of the orbitals nl may be redefined. The `15(I2,A1)`'s which occupy columns 16 through 60 should contain the nl values of the orbitals coded as $\gamma = 1, 2, 3 \ldots$ For example, if we do not wish to assign γ=3 to the 2p orbital, but rather to denote 3d by γ=3, then the third (i. e. γ^{th}) `(I2,A1)`, which begins in column 22, should contain ␣`32`, i. e. n=3, l=2. This facility can help to reduce storage requirements in cases in which orbitals with high nl are required (e. g. 6s) but in which not all of the lower orbaitals are needed (e. g. although 6s may be required, 5f and 5g may not).

(vi) `KUTSS`, specified in `(T70,I2)`, controls the calls of SR.RES, which computes FS interactions that cannot be absorbed into the spin-orbit parameters $\zeta(\gamma, \gamma')$. Computing the angular coefficients of such fine structure interactions between pairs of valence electrons is time-consuming. Leaving `KUTSS` blank has the same effect as specifying `KUTSS=1`: spin-spin, and residual mutual spin-orbit and spin-other-orbit contributions from other than the ground configuration C_1 will be ignored in the level energies. Specify `KUTSS`> 1 to include contributions from configurations C_k up to k=`KUTSS`; elements $\langle C\beta SLJ|g|C'\beta'S'L'J\rangle$ off-diagonal in the configuration, i. e. $C' \neq C$ (see EJN.46,50–51), will be ignored if `KUTSS` is positive — a choice appropriate in intermediate coupling. `KUTSS=-1` has a special meaning (magnetic two-body terms g will be entirely ignored); so has a value –9: no cut whatsoever of two-body terms!

(vii) In case `MSTART=1` — see (i) — `(T73,2A4)` is written onto file RESTART as a heading label, and in subsequent calls with `MSTART` > 1 this label appears in the log file to facilitate identification.

6.2 The analytic branch: data read in by subroutine MINIM

Several cases can be run using the same algebra but different values Z or other parameters.

Card type Z: `NZION, INCLUD, MEXTRE, NGRP, IMAXIT, JPRINT, ITOL, MPNCH, KUTCA, MRED; MQ, INTRAN, LSLIM1,LSLIM2, LFREE`

`Format(10I5, 4I5,A2)`

Parameters after the semicolon are defined for ZSS but ignored by SSS; moreover ZSS will process numerical data in the first two decimal places of `MEXTRE`. The meaning of the FORTRAN names is as follows:

(i) Z=`NZION`= atomic number (electric nuclear charge number)

(ii) `INCLUD` = number of term energies to be included in the variational sum of equation (27). The lowest `INCLUD` terms in increasing order of energy are inserted into the variational procedure.
If `INCLUD=0` then no variational procedure will be carried out, but a set of data calculated using either supplied λ_l's (see under (δ), card type D) or default values of λ_l. If user-supplied radial functions are processed (see under η) the program always runs in mode `INCLUD=0`.
`INCLUD<0` minimizes the energy functional $\mathcal{F}$ composed of `|INCLUD|` *selected* terms t with weights g_t that must be specified for these t on subsequent records (see under γ).

`NZION` and `INCLUD` are the two mandatory pieces of data. Remaining data on this 'Z-record' is optional. If the rest of the record is left blank and if `INCLUD`$\geq$0 then default values will be taken for the λ_l and σ_{nl}, and no additional records (other than a new card type Z, or a terminator card) will be required. The `INCLUD` terms will be minimised, new values of the λ_l being obtained in the process. If `INCLUD=0` one complete calculation is carried out using default λ_l and σ_{nl}, without minimisation.

(iii) `MEXTRE` specifies the number `NEXTRE` of independent variational parameters d_i. We distinguish between variational parameters d_i and scaling factors λ_l. According to section 2 to each different orbital angular momentum l there will correspond a scaling factor λ_l. However, one might wish to set a number of λ parameters equal to one another. In this case the number of true variational parameters will be less than the number of scaling factors, e. g. consider a system containing s, p and d electrons and suppose we set $\lambda_{\rm d} = \lambda_{\rm p}$. Then there will be two variational parameters d_1 and d_2, and we could assign $\lambda_{\rm s} = d_1$, $\lambda_{\rm p} = \lambda_{\rm d} = d_2$.
Thus in general `NEXTRE` is smaller than or equal to the number of different l's implied by the list of input configurations C_k; `MEXTRE` is used to supply the value of `NEXTRE` and in addition to tell the program if it is to expect data assigning variational parameters d_i to the scaling factors λ. `NEXTRE` and `MEXTRE` are related as follows:

$$\texttt{NEXTRE} \cong \texttt{MEXTRE} \text{ modulo } 100 \ .$$

If MEXTRE=0 the program assumes that the number of variational parameters d_i is equal to the number of scaling factors λ. Moreover the program expects no additional records of type D (see later under δ) which supply initial values of the variational parameters. The starting values of these parameters will be set equal to the d_i obtained in the previous case of the run; in the event of there being no previous run the program will set $\lambda_l = 1.1$ for all l.

If MEXTRE<100, NEXTRE values of the d_i $(i = 1, 2, \ldots,$ NEXTRE) will be read in on a subsequent record type D, which will therefore be expected by the computer. If NEXTRE is less than the number of λ_l's, the λ_l for $l >$ (NEXTRE $- 1$) $= j$ will be set equal to λ_j.

If MEXTRE>100 an assignment other than that valid for MEXTRE<100 is expected. In this case a card type S (see item ζ) will be required to specify this reassignment. The use of MEXTRE>100 is best illustrated by an example. Suppose that we have a case involving s, p and f orbitals, and that we wish to vary two parameters d_1 and d_2 only, defining $\lambda_{\rm s} = d_1$, $\lambda_{\rm p} = \lambda_{\rm f} = d_2$. Then we set MEXTRE=102 (which gives NEXTRE=2). The record type 2 specifies which variational parameters are assigned to $\lambda_{\rm s}$, $\lambda_{\rm p}, \lambda_{\rm d}$ $\ldots \lambda_l$, and in this case the first integer on the record of this type is set equal to 1 (representing d_1) and the second and fourth are set equal to 2 (representing d_2). If we wished to vary $\lambda_{\rm p} (= \lambda_{\rm f})$ first, then the first integer on record type 5 is set equal to 2 and the second and fourth integers equal to 1. The variation of the set $\{d_i\}$ is carried out in the order d_1, $d_2 \ldots$, $d_{\texttt{NEXTRE}}$.

If NEXTRE<-100 no input data of variational parameters is required. This option could be used with the option MEXTRE>100 in the following case: suppose that, in the example of $1s^2 2s^2 3d$, we have already determined $\lambda_{\rm s}$ in a previous run but that $\lambda_{\rm d}$ is still unknown. We therefore wish to vary $\lambda_{\rm d}$ only. It is necessary to read in the known value of $\lambda_{\rm s}$ and an estimate of $\lambda_{\rm d}$, after which one must supply the instruction that $\lambda_{\rm d}$ only is to be varied. The following sequence of records is used:

1. record type 2 with INCLUD=0 and MEXTRE=102;
2. record type 3 with the $d_i := d_1$ = estimate of $\lambda_{\rm d}$, d_2 = known value of $\lambda_{\rm s}$.
3. record type 5: ␣␣2␣␣3␣␣1;
4. another record type 2 with INCLUD=0 and MEXTRE=-101;
5. another record type 5 identical to the previous record of this type.

•• ZSS reads and processes numerical input in the first two decimal places, in addition to positions 3 (for $\ldots$) and 4–5 which as before specifies the number of input values d_i^0. A non-zero value N12 indicates that λ_{nl} rather than λ_l specifies the potential. If INLUD$\neq$0 then the first |N12| variational parameters are optimized; while positions 4–5 along with the overall sign specify how many initial values to read from input, the position 3 initiates rearrangements.

(iv) NGRP specifies the number of input screening parameters. A value 0 suffices; otherwise NGRP values σ^0_{nl} are read and used in a Rydberg formula as starting value for ϵ_{nl} in eq. 2. If initial values σ^0 are not supplied, then the default adopted by the program will be

$$\sigma^0_{nl} = \min(n \cdot (2l+1), N_{\rm el} - 1),$$

where $N_{\rm el}$ is the number of electrons.

(v) IMAXIT controls the number of iterations. A positive value specifies the number of iterative cycles (during an iterative cycle each of the variational coefficients d_i is iterated); the program limits the number of iterative steps (a step means a single variation of one d_i) to ICOUNT= 4*NEXTRE*IMAXIT + IMAXIT + 2. Blank or zero IMAXIT defaults to NEXTRE, and to no constraint on ICOUNT. This should be the usual choice, unless one deals with special problems such as (i) knowing good starting values $d_i{}^0$, so that IMAXIT<NEXTRE may be appropriate; (ii) varying correlation orbitals, which often give a very shallow minimum of the energy functional $\mathcal{F}$, or even the formal answer $d_i = \infty$, making $\delta\mathcal{F} = 0$ a rather useless criterion. To allow for more flexibility in the latter example the option IMAXIT<0 has been introduced: in this case the program limits ICOUNT=|IMAXIT|, IMAXIT defaulting to NEXTRE.

(vi) JPRINT controls the printing in the analytic branch and is normally left blank. Like its counterpart of the algebraic branch, MPRINT, it can assume negative values, which affect the operation mode and are only loosely associated with printing in that these options are run on a low level of printout. First we describe options JPRINT>0; values 1 and 2 cover peculiar modes of printing and operating, while values 3–5 specify increasing amounts of tabulation in each variational step.

JPRINT=1 extends the table on radiative transitions in intermediate coupling: transitions involving correlation levels are included, and their contributions are also passed on to SR.CASC, which computes cascade coefficients. Such transitions are quite meaningless, and may distort cascade results, unless the initial and the final state happen to represent true spectroscopic levels. Therefore these transitions are in general omitted. In the radiative table in SL coupling, however, transitions involving a correlation term are always printed but marked 'COR'.

JPRINT=2 gives detailed information, in both the radiative tables in SL coupling and in intermediate coupling, on the constituents from each configuration to a transition. Such information is useful when sensitive cancellation is suspected; it allows to assess its influence. Preceeding the ordinary line printed for a transition, one line is printed for each constituent: the first two integers denote the pair of parent terms t ($= C\beta SL$ and t', using the number label of the algebraic term table, while the summed value of the line strength amplitude, including the present constituent, is given on the right hand side; in the table in SL coupling both the value in the length and in the velocity

formulation are printed.

`JPRINT=3` is the lowest printing level, even lower than level 0: none of the standard tables in SL coupling, including those on radial functions P_{nl}, Slater integrals I and R_λ, and multipole integrals s_k, will then ever be printed out. This choice makes sense when one is interested strictly in FS results in intermediate coupling, since in big cases the radiative SL tables in particular can be excessively long.

`JPRINT=4` gives high level printout: a full term list is printed after each step of variational iteration. Not recommended except to explore unexplained events while iterating.

`JPRINT=5` gives every possible table, with all extensions, after each iterative step — suitable to produce really bulky printout: when chosen with `INCLUD=0` and variational starting values way off, say ${d_i}^0 = 1.0$, even a moderately complex problem can produce anything between 10^5 and 10^6 lines of printout.

Negative values of `JPRINT` control printing and using variational functionals other than $\mathcal{F}$. `JPRINT=-1` provides a useful piece of information in addition to that of level 0: in each iterative step the functional $\mathcal{G}$ is also computed, and printed together with $\mathcal{F}$. The functional $\mathcal{G}$ is a measure for the difference between radiative data in the length and the velocity formulation (averaged over all spectroscopic transitions only, in SL coupling). However, processing time will increase by some ten percent; therefore this level should not be chosen indiscriminately, certainly not if one has to start iterating in complete ignorance, i. e. with d_i^0. `JPRINT=-2` makes $\mathcal{G}$ rather than $\mathcal{F}$ the subject of the variational procedure. In only one instance might this choice make sense: when prior spot checks have established that $\mathcal{F}$ has a very shallow minimum while $\mathcal{G}$ has a fairly steep one in the same range of a variational parameter. `JPRINT=-3` happens to combine the features of levels –2 and +3. No radiative quadrupole results in LS coupling will be given if `JPRINT`<0; this makes level –1 a convenient choice in intermediate coupling calculations.

(vii) `ITOL` = tolerance of eigenvalues ϵ_{nl} in the solution of the radial equation (-27). The tolerance is defined by

$$\left|\frac{\Delta\epsilon}{\epsilon}\right| < 10^{\texttt{ITOL}},$$

where $\Delta\epsilon$ is the change between subsequent iterations when solving the equations for $P_{nl}(r)$. The default value, obtained by leaving the position blank, is 7. However, `ITOL` defaults to 5 in a module of SUPERSTRUCTURE compiled for `SINGLE` PRECISION unless MACHINE = `CDC`. At best (and only when carefully selecting the mesh of radial integration points) a floating point mantissa of 24 bits will yield a precision corresponding to `ITOL=6`. If one specifies a value that exceeds the precision limitations of the machine one of two things can happen: (i) The S.M. routine TFDAPO, which also uses `ITOL`, breaks down with divide check trouble, which is a rare event — and the course of

action obvious. (ii) After MAXIT=12 iterations SR.RADWAV returns a result based on $|\Delta\epsilon| \equiv |\epsilon(\texttt{MAXIT}) - \epsilon(\texttt{MAXIT}-1)| > |\epsilon| \cdot 10^{-\texttt{ITOL}}$, having printed a warning that shows the return values of $(\epsilon, \Delta\epsilon)$. The user can decide whether the answers derived from such a result are satisfactory. Before augmenting the constant MAXIT, which is declared in DATA of SR.RADWAV, he should look into the causes by rerunning the problem, specifying for σ^0_{nl} the newly obtained σ's. If he had started the previous run with fairly good input values σ^0, or if in the new run ϵ does not improve, the code based on the present machine precision will never yield a better answer; in consecutive iterations ϵ may just be oscillating between two adjacent values.

(viii) MPNCH controls printing and punching of term coupling coefficients (.140) and of cascade coefficients (.135). These coefficients will neither be printed nor punched if MPNCH=0 or if the position is left blank. Positive values ≤ 3 lead to printing only and the corresponding negative values to both printing and punching; a value +1 gives term coupling coeffice, +2 cascade coefficients, and +3 both.
Term coupling coefficients are punched according to the requirements of program JAJOM [14]: for each value J there is one heading card, in format F1 of reference [14], followed by records in format F2 on which associated term coupling coefficients and related labels are punched. The corresponding printout appears as shifted card images behind short strings of dots, conveniently placed in the table that gives details on the H matrix in I.C. coupling — and also on configuration mixing. The header card F1, in FORMAT(7X,I3,I5), specifies $2J$ and the number of term coupling coefficients $f^J_{i'i}$ associated with this value J. It is followed by the appropriate number of records F2, in FORMAT(5(2I3,F9.6)). We note that, for a given value J, the index i of its parent term is a suitable label for level Δ_i; program JAJOM labels terms sequentially, usually in energy ordering, according to card input. The second I3 specifies the level by i, and the first I3 indicates, by i', which level to parent term $\Gamma_{i'} S_{i'} L_{i'}$ contributes with the amplitude f given in F9.6.
As for cascade coefficients see the following paragraph.

(ix) KUTCA controls the calculation of cascade coefficients $C_{j,i}$ (–4.2), where i, j are level indices in energy ordering. In the case that the value of KUTCA is less than the number of levels in the lowest configuration, NG, or if the position is left blank, KUTCA defaults to NG. The index j of the lower level runs from 1 to KUTCA. The index i runs through all levels (i greater than NG) for which electric dipole transitions i to j exist. Output (i, j, C), which is controlled by MPNCH, will be printed in FORMAT(7(2X,2I3,2X,F8.6)) and punched in FORMAT(5(2I3,2X,F8.6).

(x) MRED is usually blank and ignored as long as only program-supplied S.M. functions are used (all input values < 999.0 on the card of type S). However, when processing user-supplied radial functions for "valence" electrons one may wish to employ S.M. "core"

functions in a potential that is equivalent to a higher stage of ionization: these functions will be computed in a field corresponding to a residual charge $z = Z - (N_{\rm el} - 1)$+MRED rather than the usual $z = Z - (N_{\rm el} - 1)$. Suppose that, in a Mg I case, radial functions are supplied for "valence" orbitals 3s and 3p; MRED=2 will then give Mg III type S.M. functions for the "core" $1s^2 2s^2 2p^6$.

Another 5 parameters are read in ZSS mode:

(xi) MQ $\neq 0$ specifies radial orbital function output in COLALG/IMPACT format on channel |MQ|. A positive value of MQ leads to output suitable for STG1 of the R-matrix collision code: no mutual orthogonalization, and truncation as close to the onset of the residual potential as possible. A negative value leads to more bulky untruncated und orthogonalized output as required by IMPACT — tabulating each wave function until it is down to 10^{-5} of its maximum. No trivial tails with padding zeros are ever appended.

(xii) INTRAN should be left at 0 — it can be used to manipulate the tabulation step length by powers of 2, but this may lead to uncomfortably large accumulative numerical errors.

(xiii–xiv) LSLIM1-LSLIM2, if specified $\neq 0$, suppresses radiative output in LS coupling outside the range of terms given by the two numbers.

(xv) LFREE, unless blank in both positions, leads to reading the variational parameters d_i in free format — a *very* useful feature.

None of the following records will be required if positions (ii)–(iv) of the Z-card are blank.

γ) IF INCLUD ≥ 0 GO TO δ

Records type W: (j, g_j); FORMAT(6(I3,F9.0)).

If INCLUD<0 INCLUD weight factors g_j (and indices j relating them to the algebraic term table of SR.ALGEB2) will be read, for use in the variational functional $\mathcal{F}$. If there is a preceeding Z-case and a set (j, g_j) has been supplied there one need not read in an identical set: the previous data can be used again by specifying INCLUD < -MAXTM — typically –9999; the functional $\mathcal{F}$ will then be computed according to the previous value of INCLUD (including INCLUD=0 terms if one has accidentally specified INCLUD< -MAXTM for the first Z-case). In the functional $\mathcal{G}$ transitions involving terms with weight $g_j = 0.0$ will be ignored.

δ) IF MEXTRE ≥ 0 GO TO ϵ

Records type D: (d_i); FORMAT(5F6.3).

Starting values for NEXTRE = MOD(|MEXTRE|,100) variational parameters d_i are read.

Provided that the indices are not redefined (i. e. if MEXTRE< 100) the d_i will be assigned to the scaling factors λ_l of the S.M. potential according to the scheme

d_1	d_2	d_3	...	$d_{\texttt{NEXTRE}}$	$d_{\texttt{NEXTRE}}$	...	$d_{\texttt{NEXTRE}}$
$\lambda_{\rm s}$	$\lambda_{\rm p}$	$\lambda_{\rm d}$	...	$\lambda_{\texttt{NEXTRE}-1}$	$\lambda_{\texttt{NEXTRE}}$		$\lambda_{l_{\max}}$

Here $l_{\max}$ is the highest angular momentum that occurs in the orbitals belonging to the C_k's.

Scaling factors λ_l of statistical model potentials for spectroscopic orbitals are close to 1, whereas values between around 2 and 5 are typical for potentials that produce reasonable correlation functions. Negative starting values d constitute a different option altogether: they cause the program to compute the associated orbitals l in a scaled Coulomb potential $2Z^*/r$, where $|\lambda_l|$ relates the effective electric charge number Z^* to the atomic number Z through $Z^* = |\lambda_l| \cdot Z$. This is a useful feature to obtain results in the hydrogenic approximation, and in particular to compute the leading coefficients when expanding energies or oscillator strengths in powers of Z — having specified ${d_i}^0 = -1, 0$, a suitable multiple of 10 makes a convenient choice for Z. A byproduct of this facility, of little use though, is its combination with a parameter INCLUD=0. Then the automatic variational procedure is invoked as usual, now optimizing the effective Z^*.

ϵ) IF |MEXTRE| < 100 GO TO ζ

Records type U: (IEQUAL(J),J=1,MXVAR); FORMAT(24I3).
The size parameter MXVAR being the maximum number of d_i's, $l_{\max}+1$ integers have to be specified for redefining the preceeding scheme that relates orbitals l to indices i. The element IEQUAL(l+1) says which of the NEXTRE variational parameters d_i to use for the statistical model potential of radial functions with angular l. According to the preceeding scheme IEQUAL defaults to IEQUAL(J) = MIN(J,NEXTRE).

ζ) IF NGRP = 0 GO TO β

Records type S: (SCREEN(I),I=1,NGRP); FORMAT(8F9.1).
Screening parameters σ^0_γ are read for orbitals NGRP, where γ is the 'internal' notation of nl. While values typically in the order of 1 serve as a screening constant in a rydberg formula and provide a starting value ϵ^0_{nl} for the radial function eigenvalue problem, a value $\sigma^0_\gamma \geq 999.0$ indicates to subsequently read user-supplied radial functions for this orbital γ. In the case of $\sigma^0_\gamma = 999.0$ the program first computes P_γ in a S.M. potential — which has the potential side effect of extending the range of radial tabulation and may therefore not be a desirable choice.

η) IF all specified $\sigma_\gamma < 999.0$ GO TO β,

otherwise SR.RADWIN will read user-supplied radial functions[1]

[1] of the type expected by program IMPACT (and produced by COLALG) if "–9" is punched in the

1. Structure of P/Q input set:

   (η_0) Record type `KEY=-9:  KEY,MPMX,MC.ZC; FORMAT(I5,20X,2A4).`
   Records type `KEY=-8:  (I,R(I),K=1,MPMX); FORMAT(5X,2(I4,E14.7,14X)).`
   This set specifies the radial tabulation points of the input, associating indices `I` with radial points `R(I)`.

   (η_1) Card type `KEY=-7; KEY,NP,N,L,EPS,IPMX,ORIGIN;`
   `FORMAT(9I5,I5,I5,I3,7X,F12.6,I6,29X,A8).`
   `IF IPMX = 0 GO TO` β — end of input (this terminator record is usually blank; however, specify `NP`[2] if you require tabulation of the first `NP` radial functions P_γ and the associated functions Q_γ);

2. The pattern for processing the P/Q input is as follows:
   The first set P/Q of a given orbital value l_0 will be assigned to the lowest value γ_{n,l_0} that has been marked for replacement by specifying $\sigma^0_\gamma = 999.$ on the screening parameter record. Superfluous radial functions in the input stream will be discarded; a message is printed for each such function (`EPS` and `ORIGIN` may serve for identification — which is their only role). Consider the example of an isoelectronic sequence involving orbitals 1s, 2s, 2p, and 3s, and an input stream containing two s-functions and two p-functions, which for convenience we may label '2's, '3's, '2'p, '3'p; suppose we specified $\sigma^0_{nl} = 999$ for 2s, 2p and 3s (such a specification for 3p would be ignored, because we did not assume this orbital to contribute to any configuration); then 'n'l will be identified with nl while '3'p will be ignored.

3. Messages are printed wherever irregularities occur. They are largely self-explanatory. The most likely one concerns limited buffer space when reading (η_i): it is tied to the size parameter `MAXB1` (twice that many locations `I` are available if the precision-keywork `DOUBLE` had been chosen). A warning will be printed for each affected function, and the user may judge the consequences — say by analyzing the subsequent ordinary radial function table, or even by making use of the `NP`-option on the P/Q-terminator as described under point (η_1). Another message concerns limited space for internally tabulating the associated radial functions Q_γ; this can only happen if, in order to economize on fast storage space, one has chosen `MAXB2<MAXB1` — again the `NP`-option can be helpful. We mention that one may specify `MAXB2=1` if the module is not supposed to process radial function input; when trying to do so with such a module the case will be skipped.

`KEY`-position of the heading record.

[2] or any number `NP`=$\gamma_{\max}$ for including all the $\gamma_{\max}$ orbitals involved in the case; respecify the orbital indices γ if otherwise unoccupied values γ would be embedded.

7 Restarting with previous algebraic results

It is good practice, except in fairly small cases, to save the data computed by the three algebraic SR's ALGEBi on a temporary data set file, which will be called RESTART. The control parameter `MSTART` must be specified in `(T1,I2)` on the C-terminator record. A value +1 activates the restart facility in a creation run. If this run has completed all three algebraic steps i one may restart an isoelectronic case in a later run by specifying `MSTART=4`; then all the previous algebraic results will be copied from RESTART, computations starting only in the second, the analytic branch of SUPERSTRUCTURE. The intermediate RESTART levels, which is `MSTART =` 2 and 3, apply when not all three algebraic steps have been completed yet — scan through the printout of the creation run: having stored data on RESTART at the end of a step i the program prints the message

`SR.ALGEB CREATES RESTART FILE 'xxxxxxxx', MSTART=`i,

giving the eight comment columns of the C-terminator between the inverted commata as an identifier. One restarts with `MSTART` $= i + 1$.

It is vital to restart with the correct `MSTART`; this concerns the non-trivial values 2, 3 and ≥ 4 (> 4 defaults to 4). Not only would `MSTART=` 4 make no sense before step $i = 3$ is completed; a value `MSTART=3` will be equally disastrous, without any warning, if level $i = 3$ has already been signalled as complete in a previous creation run. Parameters that are relevant to already completed steps will be given their previous value, regardless of the data input when restarting. Thus the parameter `MOD`, which affects all the algebraic steps, maintains the value assigned to it at `MSTART=1`; but `KUTSS`, a parameter affecting only ALGEB3, will be picked from the data input present at the start of step $i = 3$. The parameter `MPRINT` can be changed when restarting, provided one has chosen `MSTART`≥ 0 at level $i = 1$; if this choice was < -1, which gives the options for mere size-checking, no restart data will be saved.

In a run consisting of more than a single C-case, say the beryllium as well as the carbon sequence, only one of them can use RESTART. This channel is declared by `MR/10/` in DATA of routines MAIN and ALGEB. In general unit MR need not be specified in JCL as long as `MSTART`≤ 0; an exception to this rule is the case MACHINE = 'UNIVAC', where some rewind-statement in MAIN that cannot be controlled by MSTART overcomes a technical problem (we note that the UNIVAC facility NTRAN, rather than unformatted WRITE/READ and the associated REWIND, had to be implemented). In order to create a library of files RESTART one will have to refer to different sets RESTART by different file names in JCL. In a creation run the RESTART case need not be the first C-case; previous data on the restart-file will be overwritten.

In the STRING deck of SUPERSTRUCTURE switch parameters (I, J and K) allow to insert or delete the restart-code and associated common area declarations on the FORTRAN compile file. These switches are controlled by the restart-keywords RESTART and NORESTART

respectively.

The restart facility may not work properly if one restarts a case using recompiled code whose algebraic array sizes have changed. However, arrays that have not been involved in the concluded steps i can be changed; suppose the job has aborted in level $i = 3$ for lack of space: before restarting with `MSTART=3` one may safely recompile with adjuted FS-arrays. Parameters affecting merely the analytic branch can however be changed freely, as one starts such a run with `MSTART=4`.

8 Pre-processing

The program SUPERSTRUCTURE had been written in FORTRAN IV;[3] there are modest adjustments towards FORTRAN 77. Like some other programs of the UCL package the raw card image file STRUCT needs pre-processing before it can be handled by a FORTRAN compiler. In SUPERSTRUCTURE array size parameters that may depend upon the complexity of the physical problem are left in symbolic form, and suitable numerical values must be supplied to the preprocessor STRPRO for insertion into the COMPILE file cards. Apart from substituting character strings in declaration statements by numbers — a task that can be achieved by a basic STRING processor[4] — STRPRO can also select marked statements and conditionally include them in COMPILE, some obsolete; for instance CDC OVERLAY statements and the related `PROGRAM` statements had to be deleted before compiling STRUCT on installations other than a CDC machine; similarly, a suitable statement must be selected where one formulation applies to `DOUBLE` and another to `SINGLE` PRECISION.

8.1 Concept

The preprocessor STRPRO[5] is written in Standard FORTRAN and consists of three parts: (i) a MAIN program specific to SUPERSTRUCTURE, (ii) SUBROUTINE PROCS, which is based on the procedure STRING [7] and will be the same in future preprocessors except for minor modifications, and (iii) BLOCK DATA — they contain both general and specific information. It is convenient to keep STRPRO in compiled form.

[3] A few Extended FORTRAN features such as PRINT and the corresponding READ — though not PUNCH — have been used. Array indices may be of composed form `I+K`, except in inplied DO loops in the list of READ or WRITE. In FORMAT statements T format can be substituted by X format code if the compiler cannot handle T: the preprocessor will replace such records if the related Key Word is XFORMAT rather than TFORMAT; however, no heroic attempts have been made to always match the tidyness of the printout which T-formate code achieves effortlessly.

[4] it forms the nucleus of all MACRO processors.

[5] preprocessors to forthcoming programs will be given names such as IMPPRO and COLPRO

Before processing the raw file STRUCT the pre-processor STRPRO expects 5 data records, for example

```
   10   11    0         *                                                  SPDATA01
DEFAULT,DOUBLE,TFORMAT,RESTART,JACORD
    60    21    46    13    60    18   450   450   148 20500  1900 45000
  8000350000 40000  3390150000 11000   120 -3600   100  2000     3   250
    90   -45   325   325    21   350  2116   450                       SPDATA05
    32    10    30    11    22    18    60    60   113   552  1900  4560 smaller
  1520  6000   954  3390  2500   500   120 -3600   100  2000     3    55 case...
    24   -45   325   320    10   350  2116   450                       SPDATA05
```

Record SPDATA01, in `FORMAT(315)`, specifies the input file STRUCT by a peripheral channel number in the usual way (10 would change to 5 if STRUCT were submitted as part of the same file; the 11 specifies the channel for the output file COMPILE, which will be passed on to a FORTRAN compiler; the third integer is reserved for listing control and is usually left blank. The star ... Other options ...

On record SPDATA02 five Key Words must be specified. We give the options in tabular form:

	Key Word	options	remarks
1	MACHINE	`IBM, CDC, CRAY, UNIVAC, DEFAULT`	`DEFAULT` deletes all MACHINE specific code
2	PRECISION	`SINGLE, DOUBLE`	controls the word length for real arithmetic
3	FORMAT	`TFORMAT, XFORMAT`	see footnote...
4	RESTART	`OFFRESTART, RESTART, NTRAN`	see section ...
5	DIAGONALIZATION	`JACORD, NAGLIB`	see section ...

Records SPDATA03–05, in `FORMAT(12I6)`, assign numbers to 33 Primary Variables, which are explained in subsection 3 and in the Prologue to STRUCT. In the example, according to the first integer, storage for `MAXCF=25` configurations will be allocated in the compiled program. If positions on cards SPDATA03–05 are left blank STRPRO will substitute default values (they are assigned to array `N0` of BLOCK DATA); these values give a compiled version of SUPERSTRUCTURE that is fairly small[6] yet big enough to deal with the test case O III of 3 spectroscopic and 2 correlation configurations ($C_4 = C_0 2s2p^2 3d$, $C_5 = C_0 2s^2 2p3d$).

As already happened in the example some of the Primary Variables can be given negative values: then at least storage according to their absolute value will be provided, but STRPRO

[6] requiring core memory of 15kWords of (MACHINE,PRECISION) = (CDC,SINGLE), and 92 kBytes for `(IBM,DOUBLE)` H-level compiled; this compares with 220 kBytes in the example of *the original* SPDATA02-05.

augments it if this optimizes common storage in a sense explained later when describing individual Primary Variables — and the two major cases, involving the branches ALGEB1 and DIAGFS, will be outlined further down in this section. Having processed cards SPDATA01–05 STRPRO copes with file STRUCT as follows. Each card image is first checked in column 1, and passed immediately on to the COMPILE file unless one of the three MACRO control characters, +, * or &[7] has been encountered. A "+" in the first column labels this card as a MACRO comment, which will be suppressed. Records beginning with a star — * — pass directives, in addition to the basic data SPDATA01–05, on to STRPRO. On file STRUCT this information of STRING type forms part of the Prologue and can be condensed on the following few records:

```
+     ZUERICH MODS AS STARTED WITH CLAUDE MEUDON'84NOV -'85JL14. UPDATED'93FEV27
+     PRIMARY VARIABLES (33)                          AFTER '80AUG17TH   SPDATA06
*     MAXCF,MAXGR,MAXCL,MXEL0,MAXSL,MAXJG,MAXTM,MAXCT,MAXLV,MAXAD,MXADJ SPDATA07
*     MXEST,MXST0,MAXDC,MAXUC,MAXJU,MAXRK,MAXRL,MAXMI,MXRSS,MXSOI,MXSOC SPDATA08
*     MAXLL,MAXDF,MAXDI,MAXDK,MAXB1,MAXB2,MXVAR,MXNOR,MAXTR,MAXCA,LPNCH SPDATA09
+     SECONDARY VARIABLES
*     MXD01,MXD02,MXD03,MXD04,MXD05,MXD06,MXD07,MXD08,MXD09,MXD10,MXD11 SPDATA11
*     MXD12,MXD13,MXD14,MXD15,MXD16,MXD17,MXD18,MXD19,MXD20,MXD21,MXD22 SPDATA12
*     MXD23,MXD24,MXD25,MXD26,MXD27,MXD28,MXD29,MXD30,MXD31,MXD32,MXD33 SPDATA13
+                          **********************
```

Records SPDATA07–09 associate names with the values specified on SPDATA03–05. The code in MAIN of STRPRO is based on the present sequence; one should therefore stick to the given order.

There is an application for the whole set: records SPDATA01–13 must precede a group of individual subroutines one wants to process as a hard deck. For instance one may prefer to recompile SR.ALGEB alone, rather than the entire program, if `MAXDC` or `MXEST` are too small; or SR.DIAGFS because `MAXJU, MAXTR`, or `LPNCH` of a compiled module have proven unsuitable. The most obvious example is testing of a modified routine. As delimiter at the end of such a deck STRPRO accepts any comment card with ampersands & in columns 1 and 2.

Records begining with the third control character, &, are the main subject of preprocessing. If column 2 is blank columns 7 through 72 are searched for "&names". Names matching a Primary Variable are replaced by the specified values, in a few cases subject to upward

[7] The master deck in EBCDIC code uses punch 12, which appears as & on 029 punch key boards. A suitable alternative for BCD is 5·8, which some 026 key boards give as $\leq$. We use 11·7·8 — logical $\neg$ on a 029 key punch — for transmission through a remote CDC link that cannot recognize 5·8. In principle any character that does not form part of FORTRAN statements will do. Also change `KAMPER` in BLOCK DATA!

adjustment — for instance for alignment purposes. The Secondary Variables, as named on SPDATA11–13, are computed in the driver program MAIN to the STRING replacement routine PROCS, and depend upon both the Primary Variables and the Key Words. Finally the case that a & in column 1 is not followed by a blank `b`. It must then be one of the 24 used switch characters `A-Y` (0 has been excluded). They are explained in the Prologue. STRPRO sets the switches according to the Key Words and the Primary Variables. A statement will not be passed on to COMPILE if the switch is OFF, which is internally indicated by a numerical value –1. If the switch is ON, this means for a value +1, columns 7 through 72 are also scanned for &names before STRPRO inserts the processed card image onto COMPILE. We illustrate this with the code related to switch &W in SR.DIAGFS, which boils down to the following statements:

```
&L    DOUBLE PRECISION DPNL,DU,DV,DUY,X, ARR
&     PARAMETER(LEN0=(&MAXB1+1)*(&MAXGR+1)+&MAXGR*&MAXGR)
&     PARAMETER(LEN3=(&MAXB1+1)*(&MAXGR+1)+1, LEN2=&MAXB1+2)
      DIMENSION
&    * ARR(LEN0), X(0:&MAXB1),DPNL(0:&MAXB1,&MAXGR),DUY(&MAXGR,&MAXGR)
      COMMON
&    C /RADF/ ARR(LEN0),ENR(&MAXTM)
C    C /RADF/ X(0:&MAXB1),DPNL(0:&MAXB1,&MAXGR),DUY(&MAXGR,&MAXGR),ENR:
      EQUIVALENCE
     C  (ARR(1),X(0),DU(1,1)),(ARR(LEN2),DPNL(0,1)),(ARR(LEN3),DUY(1,1))
&W   C, (ARR(&MXD25),DV(1,1))
```

SR.DIAGFS evaluates level energies and radiative data in Breit-Pauli approximation. By the time the square arrays `DU` and `DV` are required for diagonalizing the reduced portions of the H-matrix the entire space of /RADF/ is available for scratch purposes; this common block had supplied the radial tabulation points $r_{\mathtt{I}}$ = `X(I)`, radial functions `PK(I) = DPNL(I,K)` to orbitals $n_{\mathtt{K}}l_{\mathtt{K}}$, and related integrals `DUY`. STRPRO secures enough common space for `DU`, augmenting `MAXB1` if necessary. Switch &W and with it the second half of the EQUIVALENCE statement, however, will only be turned ON if enough space is left in /RADF/; obviously `MXD25=MAXDK*MAXDK+1`. `MAXDK` is among the Primary Variables that can be specified with a minus sign. If it is, as on record SPDATA05, STRPRO augments it to optimal usage of /RADF/. This can be of mixed blessing if &W is OFF, because then array `DV` is local and the size of SR.DIAGFS might increase considerably — an undesirable side effect if this routine lies in the longest overlay branch and space is at a premium. What to choose is obvious when &W is OFF, but even when it is ON one might prefer not to increase `MAXDK` because there are still a couple of one-dimensional local arrays that go with `MAXDK`. Therefore STRPRO does not adjust `MAXDK` unless so told.

It is beyond the scope of STRPRO to assess such a situation, since the relative length of

the overlay branches or of pages varies from machine to machine, and even a change of compiler can matter. The user must judge for himself. Once familiar with the program he might, before recompiling a sensitive module, choose to run STRPRO just with a deck of tentatively modified cards SPDATA plus delimiter &&...SPDATA15. Doing so with the quoted set SPDATA02–13 results in the following preprocessor printout (the unix script printing the first line):

```
recreating modules for STRUCT run:

MACHINE CODE= 4      PRECISION CODE= 2      FORMAT CODE= 1      RESTART CODE= 2      DIAGONALISATION CODE= 1


SWITCH CHARACTERS IN USE, AND THEIR VALUES ARE AS FOLLOWS-                    -1 = OFF,   +1 = ON
   A   B   C   D   E   F   G   H   I   J   K   L   M   N   O   P   Q   R   S   T   U   V   W   X   Y
  -1   1   1  -1   1   1  -1   1   1   1  -1   1  -1   1  -1  -1  -1   1  -1   1  -1  -1   1   1  -1

          6777     RECORDS READ FROM UNIT   10
          5110     RECORDS WRITTEN TO UNIT  11
                             IN LRECF='*'
                   NORMAL END


NAMES AND THEIR ASSIGNED VALUES ARE AS FOLLOWS         -- A LETTER C, O, OR D BETWEEN NAME AND VALUE
                                                           INDICATES A COMPULSARY CHANGE, AN OPTIONAL
                                                           CHANGE, OR A DEFAULT ASSIGNMENT

     NAME    VALUE                NAME    VALUE                NAME    VALUE                NAME    VALUE

    MAXCF       60               MAXGR       21               MAXCL       46               MXEL0       13
    MAXSL       60               MAXJG       18               MAXTM      450               MAXCT      450
    MAXLV      148               MAXAD    20500               MXADJ     1900               MXEST    45000
    MXST0     8000               MAXDC   350000               MAXUC    40000               MAXJU     3390
    MAXRK   150000               MAXRL    11000               MAXMI      120               MXRSS  O 61247
    MXSOI      100               MXSOC     2000               MAXLL        3               MAXDF      250
    MAXDI       90               MAXDK  O    61               MAXB1      325               MAXB2      325
    MXVAR       21               MXNOR  C  7612               MAXTR     2116               MAXCA      450
    LPNCH  D     7               MXD01   505001               MXD02   505001               MXD03     8101
    MXD04      463               MXD05      139               MXD06      822               MXD07    22093
    MXD08   124998               MXD09       42               MXD10    63751               MXD11     1922
    MXD12     4101               MXD13      138               MXD14       61               MXD15   700000
    MXD16    20501               MXD17     1265               MXD18   761122               MXD19   -45288
    MXD20   -27252               MXD21       10               MXD22    15224               MXD23    64233
    MXD24     1323               MXD25     3722               MXD26      692               MXD27      650
    MXD28    11441               MXD29      504               MXD30       64               MXD31        4
    MXD32       16               MXD33      231
```

It contains the answer to several questions put forward in this section. Full processing will only alter lines 5 and 6. Returning to the size problem of /RADF/, let us re-examine the course of action taken by STRPRO. To taylor `MAXB1` clearly isn't the only solution, but the alternative of increasing `MAXGR` has more drastic side effects: the latter parameter enters both the ALGEB and the MINIM branches — the two large primary branches of SUPERSTRUCTURE —, whereas `MAXB1` affects only MINIM. One can therefore safely reprocess the

MINIM section alone[8], having modified MAXDK and perhaps some more local parameters; in particular a RESTART file containing the ALGEB results is still usable. Changing the value of MAXGR excludes this option. Furthermore the subbranch RADIAL of MINIM increases substantially with MAXGR if array DQNL(0:&MAXB2,&MAXGR) has been specified with a MAXB2 in the order of the minimum requirements for MAXB1, a necessity if radial function input will be processed. But there are instances where these points matter little or not at all. If one is going to compile a fresh module one might have better use for one that allows for more distinct orbitals, while nothing can be gained from storage for additional radial tabulation points. In such a case MAXGR on SPDATA02 should be specified with a minus sign. Or strike a subtler balance than STRPRO with its all-or-nothing options can do.

A last point concerning /RADF/ arises from its final use in SR.DIAGFS and has been ignored so far. After arrays DU and DV have played their role /RADF/ provides storage of length 2*MXNOR-1 for conveying radiative data to SR.CASC, which computes cascade coefficients. As no other storage is affected by MXNOR the preprocessor will always augment this primary variable to a value that makes full use of the available space; STRPRO even accounts for the fact that MXNOR defines storage of ordinary word length, whereas the arrays considered in routine DIAGFS can be of DOUBLE length: preface the coding example by "&L DOUBLE PRECISION DPNL,DUY,DX,DU,DV "!

We note that in the converse case of MXNOR dominating the storage requirements it will most likely be MAXB1 one prefers for adjusting /RADF/, and so for a very simple reason: lack of storage in this instance usually shows up not before a problem passes through SR.DIAGFS.

The most demanding task when compiling SUPERSTRUCTURE for very big cases often is allocation of working space for diagonalizing the matrices S^2 and L^2 in the Slater state representation. As a helpful tool an elaborate procedure has been incorporated in STRPRO, involving three Primary Variables: MAXDF (the maximal number of terms arising from one configuration), MXSOC (the maximum number of algebraic spin-orbit coefficients), and MXRSS (the maximum number of magnetic two-body coefficients). The storage problem is as follows: three real, and maybe DOUBLE PRECISION, square arrays of size MAXDF*MAXDF share common storage in blank COMMON // and in /COEFF/ — provided they fit into these common blocks. This storage is available when required at the ALGEB1 stage (in a later step ALGEB3 will use it for transfer of algebraic finestructure data on to the analytic main branch), but it may be too small. However, if one specifies the Primary Variable MXRSS with a minus sign STRPRO will augment it so that // can hold the first of the three arrays; the third array, which is quadratic or linear in MAXDF depending upon the choice of JACORD or NAGLIB for Key Word DIAGONALIZATION, will also be allocated space in //, but only if this can be done without incrementing MXRSS. Switches &T and &U indicate whether or not the first and the third array share common storage in //. Similarly space for the

[8] one might even keep the two halves of STRUCT on separate disk or tape files.

second square array can be secured in /COEFF/ by specifying `MXSOC` as a negative number; this involves switch &V. Conversely there might be common storage space for larger square arrays: specify `MAXDF` by its negative value, and STRPRO will augment it to the biggest value that keeps both the first and second square array in common storage; with regard to the third array, once in common storage it must stay there unless it is on-dimensional (case NAGLIB). In other words: a negative `MAXDF` never extends local storage in ALGEB1 at more than a linear rate if &U happens to be `ON` or NAGLIB is chosen. Apart from the third array SR.ALGEB1 always contains a few arrays of length `MAXDF`.

8.2 Directives

The first 2 input records direct the pre-processor.

1. `(2I5, T20,A1, T25,A1)` are used to specify the input and output channel (for source and FORTRAN file), to reset the substitution control character (& is default), and to choose FORTRANfile output other than economical record format V.

2. A literal string for choice of positioned parameters (see Table), separated by commata without blanks.

MACHINE: `DEFAULT` will be the normal choice unless the computer hardware provides for genuine `INTEGER*2` space allocation, when `IBM` should be chosen: only half the memory will then be needed for the large quantum number arrays, in particular those associated with Slater states. Other MACHINE options activate specific code and are now largely obsolete.

PRECISION means what it says: thus `SINGLE` leads to 64-bit real arithmetic on a CRAY type computer. As long as no variational iterations are involved a mantissa of 24 bits — let alone 28 as on a VAX — yields satisfactory results. However optimization in such '`SINGLE`' arithmetic is discouraged unless cumulative rounding errors are cut by judiciously reducing the number of grid points; the numerical methods though can cope with this case!

FORMAT was introduced at a time when some compilers did not know about the tabulation specifier `T`!

`RESTART` activates code for a facility described in section 7, using ordinary unformatted `READ/WRITE`. `NTRAN` had to be chosen on UNIVACs where ordinary `READ/WRITE` could not handle very long arrays.

DIAGONALIZATION selects the code activated in ALGEB1 for simultaneously diagonalizing the highly degenerate matrices S^2 and L^2. It matters in big cases that the QL or QR

algorithm is much faster than Jacobi's method. However release Mark 14 of the NAG library returns nonsense in such a simple case as p^2d^4 — with an error code number claiming success (JACORD has no difficulty and still gives a tolerable answer even with a 24-bit mantissa). Of course the consistency checks in the ALGEB1 branch throw out such results, printing details. Minor inaccuracies in `JACORD` for bigger cases had once lead to introducing alternative code on specifying `NAGLIB`.

8.3 Parameters

Size parameters are supplied on 3 records in format `(8I9)`, following the 2 records with directives. Tentative values for medium cases are given in brackets, after the STRPRO defaults.

(1) `MAXCF` [=6, 30] specifies the number of configurations.

(2) `MXORB` [=4, 15] currently still called `MAXGR`, specifies the number of radial orbitals.

(3) `MAXCL` [=10, 46] specifies the number of closed shell electrons.

(4) `MXELO` [=4, 13] specifies the number of valence electrons.

(5) `MAXSL` [=9, 20] specifies the number of distinct values SL, hence $\leq$ `MAXTM`; n b: notation SL implies parity π;

(6) `MAXJG` [=30, 18] specifies the corresponding J^π in intermediate coupling.

(7) `MAXTM` [=23, 100] specifies the number of terms SL.

(8) `MAXCT = MAXTM`, kept separate for previous convenience — algebraic and analytic branch could thus be recompiled independently.

(9) `MAXLV` [=45, 200] specifies the number of levels J^π.

(10) `MAXAD` [100, 1999] is used to declare an address array the size of the hamiltonian LS in the configuration representation, in triangular form without the empty submatrices;

(11) `MXADJ` [=220, 3999] the same but for the J^π hamiltonian.

(12) `MXEST` [=1600, 40000] specifies the total number of Slater state positions ...

(13) `MXSTO` [=600, 8000] specifies the number of Slater states — there must be enough room for a full set while evaluating a configuration before reducing it according to table 3.

(14) |MAXDC| [=1171, 10000] allocates array space for term coupling coefficients relating terms to Slater states. In the case of MACHINE= IBM the pre-processor checks the sign and confines INTEGER*2 to arrays storing quantum numbers if MAXDC carries a minus sign — and also if |MAXDC| and other parameters are larger than 32767.

(15) MAXUC [=77, 8000] specifies the number of distinct transformation coefficients needed when diagonalizing the LS-hamiltonian;

(16) MAXJU [=400, 18000] the same for the J^π-hamiltonian.

(17) MAXRK [=430, 2000] gives the cumulative number of angular coefficients in the expansion of the hamiltonian in LS coupling (held in tridiagonal form due to symmetry);

(18) MAXRL [=50, 600] specifies the number of distinct radial integrals in such an expansion.

(19) MAXMI [=20, 120] is the number of magnetic two-body integrals N_λ and V_λ,

(20) |MXRSS| [=180, 3000] is the number of associated angular coefficients.

(21) MXSOI [=6, 100] is the number of spin-orbit integrals — in the case of BP coreections to the radiative operators augmented by the number of one-body integrals arising there.

(22) |MXSOC| [=120, 1000] is the number ofspin-orbit parameters.

(23) MAXLL [=2, 3] reserves space of size $(\texttt{MAXLL}+1)^5$ for arrays holding suitable products of Clebsch-Gordan coefficients that dramatically reduce computer time; in fact the Slater state approach would hardly be viable with out them, especially for the two-body magnetic terms. These arrays share storage previously held by the diagonalization arrays, and MAXLL is upwardly adjusted. But because of the high power with which the size goes it may not always be economical to provide for the highest l's associated with a case; then the required coefficients are computed calling VCC etc.

(24) MAXDF [=24, 250] is used in the diagonalization code of the matrices S^2 and L^2 in a Slater state representation and must match the largest number of Slater states contributing to any such matrix.

(25) MAXDI [=5, 90] is used in the diagonalization code of the hamiltonian in LS coupling and must match the maximum number of symmetries associated with the same value SL;

(26) |MAXDK| [=10, 90] is the corresponding quantity in intermediate coupling and must match the maximum number of symmetries with the same value J^π.

(27) MAXB1 [=220, 320] is the maximum number of radial grid points.

(28) MAXB2 =1 suffices in SSS as long as RADWIN is not called. ZSS however always needs working space declared by MAXB2≤ MAXB1; the job aborts if MAXB2≤IEND (the current range of a non-Coulombic potential), whereas a mere warning is printed if space might not suffice (and a larger value is suggested).

(29) MXVAR, the number of variational parameters, rarely exceeds 5 in SSS. ZSS however requires MXVAR = MXORB — otherwise jobs abort in the analytic branch.

(30) MXNOR [=100,] locations are available to store non-vanishing electric dipole transition probabilities in intermediate coupling in DIAGFS for computing cascade coefficients in CASC.

(31) MAXTR [=150, 2116] allocates buffer storage for printing term coupling coefficients as used in JAJOM [14]

(32) MAXCA [=190, 450] specifies the number of cascade coefficients.

(33) LPNCH, not exactly a size parameter, specifies the output channel for for the quantities controlled by MPNCH — position (viii) on the Z-record. It defaults to 7 but can be overridden by LPNCH.

9 Short description of tabular output

10 Test cases

References

[1] J. A. Belling — private communication 1967.

[2] K. A. Berrington, P. G. Burke, K. Butler, M. J. Seaton, P. J. Storey, K. T. Taylor and Yu Yan, *J. Phys. B: At. Mol. Phys.* **20** (1987) 6379–97.

[3] K. A. Berrington, W. B. Eissner, H. E. Saraph, M. J. Seaton, and P. J. Storey, *Comput. Phys. Commun.* **44** (1987) 105–19

[4] M. Blume and R. E. Watson, *Proc. R. Soc.* **270 A** (1962) 127–43.

[5] E. Clementi and R. Roetti, *At. Data Nucl. Data Tables* **14** (1974) 177–478.

[6] M. A. Crees, M. J. Seaton, and P. M. H. Wilson, *Comput. Phys. Commun.* **15** (1978) 23–83.

[7] C. Day, *Comput. Phys. Commun.* **13** (1977) 101–4.

[8] W. Eissner, *J. Physique IV* **1** (1991) C3–13.

[9] W. Eissner, M. Jones and H. Nussbaumer, *Comput. Phys. Commun.* **8** (1974) 270–306.

[10] A. Hibbert, *Comput. Phys. Commun.* **9** (1975) 141–72.

[11] M. Jones, *Radiative corrections in Atomic Structure Calculations,* Ph D thesis London 1971.

[12] H. Nussbaumer and P. J. Storey, *Astron. Astrophys.* **64** (1978) 139–44.

[13] M. J. D. Powell, *Comp. J.* **7** (1965) 303–7.

[14] H. E. Saraph, *Comput. Phys. Commun.* **3** (1972) 256–68, and **15** (1978) 247–58.