DARC manual

Patrick Norrington

January 24, 2002

DARC has been developed mainly by Patrick Norrington in collaboration with Prof Ian Grant FRS. Any inquiries or comments on the package can be directed to Dr Norrington at

Department of Applied Mathematics and Theoretical Physics
The Queen's University of Belfast
Belfast
Northern Ireland
BT7 1NN
United Kingdom

telephone : +44 (0) 28 9027 3190
fax : +44 (0) 28 9023 9182

email : p.norrington@qub.ac.uk
homepage : http://www.am.qub.ac.uk/users/p.norrington

DISCLAIMER

This code has been developed in good faith. PN believes that it mostly produces reliable results. However the user is warned that:

1. the code is outdated, the technology is old-fashioned and the code runs slowly, using lots of disc space and is plain inefficient;
2. the code contains bugs (hard to find, hairy and horrible);
3. nonsense in ⟶ nonsense out **ALWAYS**
   sense in ⟶ nonsense out **SOMETIMES**

Use of this code can cause sleepless nights. You have been **WARNED**.

Contents

Chapter 1

Theory

1.1 Brief history

The R-matrix method originated in nuclear physics and the non-relativistic derivation is due to Wigner and Eisenbud (1947 [65]). Shortly afterwards it was generalised by Goertzel (1948 [35]) to the Dirac equation.

The method was pioneered in atomic physics by Burke and his collaborators at Belfast. The non-relativistic method was formulated by Burke et al. (1971 [19]) and Burke and Taylor (1975 [20]), coded as RMATRX by Berrington et al. (1974 [9]) and used successfully. The non-relativistic programs were modified and republished by Berrington et al. (1978 [10]). The non-relativistic version uses the Schrödinger Hamiltonian and includes much of the technology from the CIV3 atomic structure package of Hibbert. It allows continuum problems, such as electron excitation and photo-ionisation, to be handled for complex light ions and atoms.

One approach to extending the method to heavier systems is to use the Pauli Hamiltonian. This was described by Scott and Burke (1980 [53]) and the first Breit-Pauli code was published by Scott and Taylor (1982 [56]). The Opacity Project, led by Seaton (1987 [59]), developed further related codes for the calculation of atomic data needed in stellar opacity models. See also Berrington et al. (1987 [11]) and the Opacity Project Team's book (1995 [49]). RMATRX is currently being used in the follow-on IRON Project [51], for example Hummer et al. (1993 [41]). The latest Breit-Pauli code revision, called RMATRX1, was described by Berrington et al. (1995 [12]).

An alternative and more satisfactory approach to treating heavier systems is to use the Dirac Hamiltonian. The atomic theory was given by Chang (1975 [27], 1977 [28]). It was decided to base the program on the Dirac atomic structure code now known as GRASP. GRASP is an acronym for the General-purpose Relativistic Atomic Structure Package developed by Ian Grant and collaborators at Oxford. Three versions of the package have been published (1980 [39, 45], 1989 [34], 1996 [50]).

Young (1992 [66]) tackles the solution of the asymptotic equations. Szmytkowski and Hinze (1996 [62]) have re-examined the Dirac theory and shown that earlier derivations have been in error. Their work has put the theory on a more consistent basis. Fortunately their correction to the R-matrix does not affect any calculations based on the theory of Chang.

1.2 Units

Most equations are written here in Hartree atomic units (a.u.) where the mass of the electron m, the charge of the proton e and the reduced Planck constant $\hbar = h/2\pi$ become unity. The speed of light equals the reciprocal of the fine structure constant, $c = 1/\alpha = 137.03599$. The unit of length is the Bohr radius $a_0 = 5.291772 \times 10^{-11}$ m, and the unit of energy is the Hartree energy $E_h = 4.35975 \times 10^{-18}$ J.

Other common units of energy are Rydbergs, 1 a.u. = 2 Ryd, and electron-volts, 1 a.u. = 27.2124 eV. Cross sections have the units of area. Photo-ionisation cross sections are normally given in Megabarns where 1 Mbarn = 10^{-22} m^2. Data is from the recommended values published in CODATA Bulletin (1986 [29]).

1.3 Dirac R-matrix method

The atomic system consists of an N-electron atomic target plus one electron which can be in the continuum. For electron excitation this electron is free both initially and finally. For photo-ionisation the electron is initially bound and finally free.

The R-matrix scheme divides configuration space into two regions by a spherical boundary at $r = a$, known as the R-matrix boundary.

In the external region, $r > a$, electron exchange with the atom is negligible and the scattering equations are considerably simplified though still non-trivial to solve in general.

In the internal region, $r < a$, the free electron is indistinguishable from the other N target electrons. Essentially the problem reduces to an atomic structure calculation for an $N + 1$ electron system. $N+1$ electron wavefunctions are constructed from a complete orthonormal set of bound and continuum one-electron functions. The bound functions have zero amplitude at the R-matrix boundary while the continuum functions satisfy a specific R-matrix boundary condition there. The $(N + 1)$-electron wavefunctions are of two types: continuum and capture. The capture wavefunctions are formed only from the bound one-electron functions. They allow the possibility of the free electron being temporarily captured by the N-electron target. This gives rise to the important effect of resonance which dominates much of low-energy scattering. The continuum states use the continuum one-electron functions to describe the extra electron.

This set of $(N + 1)$-electron wavefunctions is then diagonalised over the Hamiltonian to get the R-matrix basis which is energy independent. For each energy the R-matrix can be calculated quickly. It describes the surface boundary conditions for matching to asymptotic solutions found in the external region. Matching obtains the K-matrix and the required physical quantities.

The R-matrix version of close-coupling allows rapid calculation of the resonance structure of processes in the low energy range covered by the target states included in the problem. Results at higher energies may also be obtained if continuum states are suitably treated i.e. a large enough continuum basis and/or pseudo-states are used.

1.4 Equation of motion

The target can have a large nuclear potential so that relativistic effects need to be considered. Relativistic effects, like the spin-orbit interaction, can be important for low Z as well when non-relativistic solutions produce cancellation in physical quantities. Solutions are required to the time independent equation

$$H^{N+1}\Psi = E\Psi \tag{1.1}$$

The Dirac Hamiltonian for the $(N+1)$-electron system can be written in atomic units as

$$H^{N+1} = \sum_{i=1}^{N+1} -\mathrm{i}c\boldsymbol{\alpha}\cdot\boldsymbol{\nabla}_i + (\beta-1)c^2 - \frac{Z}{r_i} + \sum_{j=i+1}^{N+1} \frac{1}{|\boldsymbol{r}_j - \boldsymbol{r}_i|} \tag{1.2}$$

where i and j index the individual electrons, Z is the charge of an infinitely heavy point nucleus, the electron rest mass has been subtracted, and $\boldsymbol{\alpha}$ and β are the usual Dirac matrices constructed from Pauli spin and unit matrices

$$\begin{gathered}\boldsymbol{\alpha} = \begin{pmatrix} 0 & \boldsymbol{\sigma} \\ \boldsymbol{\sigma} & 0 \end{pmatrix} \qquad \beta = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \\ \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad \sigma_y = \begin{pmatrix} 0 & \mathrm{i} \\ -\mathrm{i} & 0 \end{pmatrix} \qquad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\end{gathered} \tag{1.3}$$

The first three one-electron terms in the Hamiltonian are a momentum term, mass term and the electron-nucleus Coulomb attraction. The final two-electron term is the Coulomb electron-electron repulsion. The Hamiltonian matrix element can be split so that angular integrals are calculated using Racah algebra and radial integrals by numerical quadrature.

Further relativistic corrections, for example the Breit interaction (a virtual photon exchanged between electrons) and other quantum electrodynamic (QED) corrections, self-energy and vacuum polarisation, are not included in the Hamiltonian. However the eigenvalues of the Hamiltonian may be adjusted to approximate these effects, and providing the changes are small this appears to be a reasonable approach.

1.5 Model potentials

With large complex systems the computational problem can be drastically reduced by using a model potential to describe the inner closed shell core of electrons. Integrals then only need to be calculated for valence electrons, which form excited states, and continuum electrons. The Dirac Hamiltonian becomes

$$H^{N+1} = \sum_{i=1}^{M+1} -\mathrm{i}c\boldsymbol{\alpha}\cdot\boldsymbol{\nabla}_i + (\beta-1)c^2 + V(r_i) + \sum_{j=i+1}^{M+1} \frac{1}{|\boldsymbol{r}_j - \boldsymbol{r}_i|} \tag{1.4}$$

where there are M valence electrons in the target and $V(r)$ is the model potential satisfying the boundary conditions

$$V(r) \underset{r\to 0}{\sim} -\frac{Z}{r} \qquad V(r) \underset{r\to\infty}{\sim} -\frac{Z-N}{r} \tag{1.5}$$

Although the basis that will be constructed from one-electron orbitals is reduced in size, the continuum functions have to be orthogonalised with the full set of bound orbitals.

1.6 The R-matrix scheme

1. Configuration space is partitioned so that electron exchange and correlation of a continuum electron with the atom is negligible in the external region ($r > a$). Normally a is chosen so that the absolute value of the large radial component of all bound orbitals is less than a small value, typically 10^{-3}. The internal region is treated similarly to a simple Dirac-Fock atomic structure calculation. In the external region the electron moves in the long-range multipole potential of the target.

2. In the internal region N-electron target states Φ and $(N+1)$-electron capture states θ are constructed from the configuration interaction (CI) expansion of a set of orthonormal bound orbitals. The continuum orbitals satisfy specified boundary conditions at $r = a$. For different values of κ, sets of continuum electron functions are created, orthonormal to the bound orbitals. The target states may include pseudo-continuum-orbitals as well as Dirac-Fock orbitals.

3. An $(N + 1)$-electron basis Θ_k is constructed from target states coupled to continuum electron states, plus the capture states (also known as correlation functions.) The Hamiltonian matrix $\langle\Theta_k|H^{N+1}|\Theta_{k'}\rangle$ is diagonalised to get an orthonormal set of energy-independent wavefunctions Ψ_k.

4. At each energy the R-matrix is evaluated, specifying the surface boundary conditions. The channel wavefunctions in the external region are calculated using various asymptotic packages. The solutions are matched at $r = a$ to get the K-matrix and ultimately the required cross sections. When the residual atomic charge is not high and the electron is moving in the external region with a velocity much less than c, non-relativistic asymptotic solutions are adequate.

1.7 Channels, symmetries and selection rules

There are constraints in combining a scattering or bound electron with the target/core. And further constraints come from a dipole transition of the system absorbing a photon.

1.7.1 Conservation of angular momentum and parity

Target N-electron states are labelled by total angular momentum J^{t} and parity π^{t}. These states are constructed from linear combinations of configuration state functions (CSFs) which have the same angular momentum and parity. Each CSF defines the principal and angular momentum quantum numbers of each individual electron in the configuration. Thus a target state, or level, can be distinguished by the CSF mixing coefficients and it has a particular energy.

The $(N + 1)$-electron, or target+electron, system is also characterised by total angular momentum J and overall parity π. J is formed by coupling vectorially J^{t} and j, the total angular momentum of the additional electron. The possible ways of doing this are governed by the triangular relation $|J^{\mathrm{t}} - j| \leq J \leq J^{\mathrm{t}} + j$, where the minimum difference between each J is 1. Then each pair $J\pi$ are referred to as a "symmetry".

The $(N+1)$th electron is labelled by quantum number κ which is equivalent to specifying both j and the orbital angular momentum l. ($2j + 1 = 2|\kappa|$, if $\kappa < 0$ then $l = -\kappa - 1$ else $\kappa > 0$ and $l = \kappa$.) Thus a symmetry of the $(N+1)$-system is composed of the possible "channels" which are labelled by ($J^{\mathrm{t}}\pi^{\mathrm{t}}$,$\kappa$) pairs.

In a scattering process the symmetry will be conserved. The channel will not change for elastic scattering but will for inelastic scattering. It may take the sum of many symmetries for a calculation to converge as the scattering energy increases and with it the number of energetically accessible channels. This corresponds to the familiar partial wave expansion in low energy potential scattering.

In photo-ionisation the symmetry is not conserved. The electric dipole transition process involves absorption of a photon with angular momentum 1 and a change of parity. Therefore photo-ionisation from a particular bound state restricts the number of symmetries in the calculation to two or three.

1.7.2 Conservation of energy

If E^{N+1} it the total energy of the electronic system and E_i^N the energy of the target state coupled to the ith channel then the channel energy of a scattered electron is $\epsilon_i = E^{N+1} - E_i^N$. An open channel has $\epsilon_i > 0$, a closed channel has $\epsilon_i < 0$ and at the channel threshold $\epsilon_i = 0$.

For photo-ionisation by a photon with incident energy ω of an atom in an initial bound state with energy E_0^{N+1} the channel energy is similarly $\epsilon_i = \omega + E_0^{N+1} - E_i^N$. Of course all channels are closed in a bound state.

1.8 Target states

The $(N+1)$-electron wavefunction is constructed from N-electron target states (possibly including pseudo-states) and single particle wavefunctions for the continuum/valence electron. The target states are eigenfunctions of the equation

$$H^N \Phi_i = E_i \Phi_i \tag{1.6}$$

where eigenfunctions, indexed by i, have a particular total angular momentum and overall parity. They are written as a configuration-interaction (CI) expansion of a basis set of N-electron functions.

$$\Phi_i = \sum_k b_{ik} \psi_k \tag{1.7}$$

Thus the configuration mixing coefficients b_{ik} are found by diagonalising the target Hamiltonian matrix. The ψ_k configuration state functions (CSFs) are obtained by the antisymmetric coupling together of N orthonormal single particle orbitals. These orbitals are usually of self-consistent-field type, calculated for example by the atomic structure code GRASP (Dyall et al. 1989 [34]). Pseudo-orbitals may be included to discretely approximate continuum states that contribute to correlation.

The single particle functions have the form

$$\phi(\boldsymbol{r}) = \frac{1}{r}\begin{pmatrix} \mathcal{P}(r)\chi_{\kappa m} \\ \mathrm{i}\mathcal{Q}(r)\chi_{\bar{\kappa} m} \end{pmatrix} \tag{1.8}$$

Here ϕ refers to continuum functions and $\underline{\phi}$ to bound orbitals. The $\chi_{\kappa m}$ are two-component spinor made up of spherical harmonics and Clebsch-Gordan coefficients. The radial parts of the bound orbitals are input to the R-matrix code as numerical values on a suitable mesh of points.

1.9 Continuum orbital boundary conditions

Only the continuum orbitals are non-zero on the surface of the internal region. The single particle continuum function basis are solutions to the differential equations

$$\begin{aligned} \frac{d\mathcal{P}_i}{dr} &= -\frac{\kappa}{r}\mathcal{P}_i + \left(2c + \frac{\epsilon_i}{c} - \frac{V(r)}{c}\right)\mathcal{Q}_i - \frac{1}{c}\sum_{l=1}^{n}\lambda_{il}\underline{\mathcal{Q}}_l \\ \frac{d\mathcal{Q}_i}{dr} &= \frac{\kappa}{r}\mathcal{Q}_i - \left(\frac{\epsilon_i}{c} - \frac{V(r)}{c}\right)\mathcal{P}_i + \frac{1}{c}\sum_{l=1}^{n}\lambda_{il}\underline{\mathcal{P}}_l \end{aligned} \tag{1.9}$$

where i indexes the continuum functions and l the bound orbitals, and λ_{il} are Lagrange multipliers chosen so that the continuum and bound wavefunctions are orthogonal

$$\int_0^a (\mathcal{P}_i\underline{\mathcal{P}}_j + \mathcal{Q}_i\underline{\mathcal{Q}}_j)\,dr = 0 \tag{1.10}$$

Combining the differential equations and using orthogonality of the continuum functions

$$\begin{aligned} (\mathcal{Q}_j\mathcal{P}_i - \mathcal{Q}_i\mathcal{P}_j)_{r=a} &= \int_0^a \left(\mathcal{Q}_j\frac{d\mathcal{P}_i}{dr} + \mathcal{P}_i\frac{d\mathcal{Q}_j}{dr} + \mathcal{Q}_i\frac{d\mathcal{P}_j}{dr} + \mathcal{P}_j\frac{d\mathcal{Q}_i}{dr}\right)dr \\ &= \frac{\epsilon_i - \epsilon_j}{c}\int_0^a (\mathcal{P}_i\mathcal{P}_j + \mathcal{Q}_i\mathcal{Q}_j)\,dr \end{aligned} \tag{1.11}$$

then the boundary condition at $r = a$ is chosen

$$\frac{\mathcal{Q}_i(a)}{\mathcal{P}_i(a)} = \frac{b + \kappa}{2ac} \tag{1.12}$$

for all i, where b is an arbitrary constant. The boundary condition reduces in the non-relativistic limit to

$$\left.\frac{d\mathcal{P}_i}{dr}\right|_{r=a} = \frac{b}{a}\mathcal{P}_i(a) \tag{1.13}$$

From an asymptotic polynomial expansion the boundary condition at the origin is

$$\begin{aligned}
\mathcal{P}_i &\underset{r\rightarrow 0}{\sim} p_i r^\gamma \\
\mathcal{Q}_i &\underset{r\rightarrow 0}{\sim} \left\{ \begin{array}{ll} \frac{c(\kappa+\gamma)}{Z} p_i r^\gamma & \kappa > 0 \\ \frac{Z}{c(\kappa-\gamma)} p_i r^\gamma & \kappa < 0 \end{array} \right.
\end{aligned} \tag{1.14}$$

where $\gamma = \sqrt{\kappa^2 - (Z/c)^2}$ and p_i is a normalisation constant. The functions can be made orthonormal over the internal region and then the scattered electron wavefunctions expanded as linear combinations of this basis set.

1.10 Internal region

A coupled state of the $(N+1)$-electron system is made from a target state $\Phi = |J^{\mathrm{t}} M^{\mathrm{t}}\rangle$ and a single electron state $\phi = |jm\rangle$

$$\begin{aligned}
\mathcal{A}[\Phi, \phi]^{J,M} &= \sum_{i=1}^{N+1} \frac{(-1)^{N+1-i}}{\sqrt{N+1}} [\Phi(\boldsymbol{r}_1, .., \boldsymbol{r}_{i-1}, \boldsymbol{r}_{i+1}, .., \boldsymbol{r}_{N+1}), \phi(\boldsymbol{r}_i)]^{J,M} \\
&= \mathcal{A} \sum C(J^{\mathrm{t}} jJ; M^{\mathrm{t}} mM) |J^{\mathrm{t}} M^{\mathrm{t}}\rangle |jm\rangle
\end{aligned} \tag{1.15}$$

so $\mathcal{A}$ is an asymmetrisation operator to account for electron exchange. Capture states are defined when all $N+1$ single particle states are bound, $\theta = \mathcal{A}[\Phi, \underline{\phi}]^{J,M}$, and are needed to make the wavefunction complete. All allowed coupled states for a total angular momentum J and parity π form the basis set Θ_k. The J and M superscripts are omitted from later equations.

Diagonalising the $(N+1)$-Hamiltonian matrix $\langle\Theta_k|H^{N+1}|\Theta_{k'}\rangle$ over the internal region an orthonormal set of wavefunctions is obtained that can be written as

$$\Psi_k = \sum_{ij} c_{ijk} \mathcal{A}[\Phi_i, \phi_{ij}] + \sum_m d_{mk} \theta_m \tag{1.16}$$

with eigenvalue E_k^{N+1} and eigenvector (c_{ijk}, d_{mk}) where k is the eigenvector index, j the continuum basis function index for a particular κ value, and i the channel index indicating a pair $(J^{\mathrm{t}}\pi^{\mathrm{t}}, \kappa)$ coupled to get the total angular momentum J and parity π. Note that for the continuum functions the radial integrals are over a finite range of integration even though the bra and ket notation is being used.

1.11 Derivation of the R-matrix

The total wavefunction which satisfies eq.(1.1) for any energy E is expanded in terms of the basis of eq.(1.16) as

$$\Psi_E = \sum_k A_{Ek} \Psi_k \tag{1.17}$$

If we define the surface amplitudes w_{ik} by

$$\frac{1}{r}\left(\begin{array}{c} w_{ik}(r)\chi_{\kappa m} \\ \mathrm{i}v_{ik}(r)\chi_{\bar{\kappa}m} \end{array}\right) = \sum_j c_{ijk}\phi_{ij}(\boldsymbol{r}) \tag{1.18}$$

and the wavefunction of the scattered electron in channel i can be written as an expansion

$$\left(\begin{array}{c} P_i(r) \\ Q_i(r) \end{array}\right) = \sum_k A_{Ek}\left(\begin{array}{c} w_{ik}(r) \\ v_{ik}(r) \end{array}\right) \tag{1.19}$$

then using the Hamiltonian and the boundary conditions eq.(1.12), an expression for the expansion coefficients is eventually found [27]

$$\begin{aligned} A_{Ek} &= \frac{1}{E_k^{N+1}-E}[\langle\Psi_E|H^{N+1}|\Psi_k\rangle - \langle\Psi_k|H^{N+1}|\Psi_E\rangle] \\ &= \frac{1}{E_k^{N+1}-E}\int_0^a \left[\sum_i c\frac{d}{dr}(P_iv_{ik} - Q_iw_{ik})\right] dr \\ &= \frac{1}{2a(E_k^{N+1}-E)}\sum_i w_{ik}(a)[2acQ_i(a) - (b+\kappa_i)P_i(a)] \end{aligned} \tag{1.20}$$

The large radial component of the continuum electron at the boundary can be written in terms defining the R-matrix

$$\begin{aligned} P_i(a) &= \sum_j R_{ij}[2acQ_j(a) - (b+\kappa_j)P_j(a)] \\ R_{ij} &= \frac{1}{2a}\sum_k \frac{w_{ik}(a)w_{jk}(a)}{E_k^{N+1}-E} \end{aligned} \tag{1.21}$$

In practice the R-matrix expansion has to be truncated but the contribution that is lost can be adequately represented by the Buttle correction [24, 27].

Szmytkowski and Hinze (1996 [62]) have shown that the full R-matrix expansion eq.(1.21) is not quite correct. There is a small correction because the Dirac R-matrix expansion of the wavefunction does not converge to the function on the surface boundary. They give a thorough derivation of the proper expressions for the multichannel case. Fortunately when the expansion is truncated and the Buttle correction is used the error is canceled. It would be a be a small error to the R-matrix anyway, for a large atom approximately $-(b+\kappa)/4(ac)^2 \sim \mathcal{O}(10^{-8})$.

1.12 Buttle correction

The truncation of the expansion of the R-matrix eq.(1.21) produces the largest error in the method. The missing terms from the diagonal can be important, especially for elastic scattering, so an approximate correction from the formula discussed by Buttle (1967 [24]) is added. The interchannel coupling is considered as negligible and thus the Hamiltonian matrix automatically diagonal. The radial differential equations for the continuum functions eq.(1.9) are solved without imposing the boundary conditions at $r = a$ so that for an arbitrary channel energy ϵ_i^{c} the wavefunction is labelled $\mathcal{P}_i^{\mathrm{c}}$ and $\mathcal{Q}_i^{\mathrm{c}}$. If the R-matrix is truncated to the first K terms, i.e. continuum functions, then the correction added to the diagonal is given by

$$\begin{aligned} R_{ii}^{\mathrm{c}} &= \frac{1}{2a} \sum_{k=K+1}^{\infty} \frac{\mathcal{P}_{ik}(a)^2}{\epsilon_{ik} - \epsilon_i^{\mathrm{c}}} \\ &= \left(\frac{2ac\mathcal{Q}_i^{\mathrm{c}}(a)}{\mathcal{P}_i^{\mathrm{c}}(a)} - b - \kappa \right)^{-1} - \frac{1}{2a} \sum_{k=1}^{K} \frac{\mathcal{P}_{ik}(a)^2}{\epsilon_{ik} - \epsilon_i^{\mathrm{c}}} \end{aligned} \tag{1.22}$$

When ϵ_i^{c} is between the eigenvalues ϵ_{ik} the function R_{ii}^{c} is smooth and is in practice approximated by quadratic Lagrange interpolation. The correction is only calculated exactly at two points between each pair of adjacent eigenvalues.

1.13 External region

Neglecting the electron exchange the scattered electron wavefunction satisfies the radial differential equations

$$\begin{aligned} \frac{d}{dr}P_i + \frac{\kappa}{r}P_i - \left(2c + \frac{\epsilon_i}{c} + \frac{z}{cr}\right) Q_i &= -\frac{1}{c} \sum_{j=1}^{n} \sum_{\lambda=1}^{\lambda_{\max}} \frac{a_{ij}^{\lambda}}{r^{\lambda+1}} Q_j \\ \frac{d}{dr}Q_i - \frac{\kappa}{r}Q_i + \left(\frac{\epsilon_i}{c} + \frac{z}{cr}\right) P_i &= \frac{1}{c} \sum_{j=1}^{n} \sum_{\lambda=1}^{\lambda_{\max}} \frac{a_{ij}^{\lambda}}{r^{\lambda+1}} P_j \end{aligned} \tag{1.23}$$

where n is the number of channels and the residual charge is $z = Z - N$. Using an expansion for the long-range potential in terms of Legendre polynomials

$$\sum_{k=1}^{N} \frac{1}{|\boldsymbol{r}_k - \boldsymbol{r}_{N+1}|} = \sum_{\lambda=1}^{\infty} \left(\frac{1}{r_{N+1}}\right)^{\lambda+1} \sum_{k=1}^{N} r_k^{\lambda} P_{\lambda}(\hat{\boldsymbol{r}}_k \cdot \hat{\boldsymbol{r}}_{N+1}) \tag{1.24}$$

the asymptotic scattering coefficients are

$$a_{ij}^{\lambda} = \mathrm{Ang}\langle[\Phi_i, \phi_i]| \sum_{k=1}^{N} r_k^{\lambda} P_{\lambda}(\hat{\boldsymbol{r}}_k \cdot \hat{\boldsymbol{r}}_{N+1}) |[\Phi_j, \phi_j]\rangle \tag{1.25}$$

where Ang denotes that the integral of the outer electron radial coordinate $r_{N+1} = r$ is left out. Note that $a_{ij}^0 = N\delta_{ij}$. The multipole expansion of the long-range potential is truncated to $\lambda_{\max}$ terms since the range of λ for which $a_{ij}^{\lambda} \neq 0$ is determined by the angular momenta of the channels i and j.

1.13.1 Asymptotic form of scattered electron

The reactance K-matrix describes the asymptotic form of the whole wavefunction for a scattered electron. The photo-electron asymptotic form can also be expressed in terms of it. The K-matrix is a real symmetric matrix and, as given by Young (1992 [66]), it is defined

$$\begin{pmatrix} P_{ij} \\ Q_{ij} \end{pmatrix} \underset{r\to\infty}{\sim} \begin{cases} S_i\delta_{ij} + C_iK_{ij} & \epsilon_i > 0 \quad i = 1,\ldots,n_\mathrm{o} \\ D_iL_{i-n_\mathrm{o},j} \sim \mathcal{O}(r^{-2}) & \epsilon_i < 0 \quad i = n_\mathrm{o}+1,\ldots,n \end{cases} \tag{1.26}$$

where n_o is the number of open channels, $j = 1,\ldots,n_\mathrm{o}$ indexes the linearly independent solutions, $L_{i-n_\mathrm{o},j}$ is the closed channel K-matrix , and

$$S_i = \sqrt{q_i}\begin{pmatrix} \sin\vartheta_i \\ \frac{\epsilon_i}{k_ic}\cos\vartheta_i \end{pmatrix} \qquad C_i = \sqrt{q_i}\begin{pmatrix} \cos\vartheta_i \\ -\frac{\epsilon_i}{k_ic}\sin\vartheta_i \end{pmatrix}$$

$$k_i = \sqrt{2\epsilon_i + \tfrac{\epsilon_i^2}{2c^2}} \qquad q_i = \tfrac{1}{k_i}\left(1 + \tfrac{\epsilon_i}{2c^2}\right)$$

$$\vartheta_i(r) = k_ir + y_i\ln 2k_ir + \varphi_i - \arg\Gamma(\gamma_i - iy_i) + \tfrac{\pi}{2}(\gamma_i - 1)$$

$$y_i = \tfrac{z}{k_i}\left(1 + \tfrac{\epsilon_i}{c^2}\right) \qquad \gamma_i = \sqrt{\kappa_i^2 - \tfrac{z^2}{c^2}} \qquad \varphi_i = \tfrac{1}{2\mathrm{i}}\ln\left(\tfrac{\mathrm{i}\frac{z}{k_i} - \kappa_i}{\gamma_i - \mathrm{i}y_i}\right) \tag{1.27}$$

$$D_i = \sqrt{\hat{q}_i}\begin{pmatrix} \mathrm{e}^{-\hat{\vartheta}_i} \\ \frac{\epsilon_i}{\hat{k}_ic}\mathrm{e}^{-\hat{\vartheta}_i} \end{pmatrix} \qquad \hat{k}_i = |k_i| \qquad \hat{q}_i = \tfrac{1}{\hat{k}_i}\left(1 + \tfrac{\epsilon_i}{2c^2}\right)$$

$$\hat{\vartheta}_i(r) = \hat{k}_ir - \hat{y}_i\ln 2\hat{k}_ir \qquad \hat{y}_i = \tfrac{z}{\hat{k}_i}\left(1 + \tfrac{\epsilon_i}{c^2}\right)$$

The K-matrix is obtained by solving the differential equations and using a matching process to the R-matrix similar to that given in Burke et al. (1971 [19]) and Chang (1975 [27]).

1.13.2 Asymptotic form of photo-electron

The asymptotic boundary conditions for photo-ionisation are that of an outgoing spherical wave in the final state channel plus ingoing-waves in all open channels as described by Henry and Lipsky (1967 [40]) and Burke (1976 [21]). The photo-electron wavefunction, labelled with a minus sign, can be written in terms of the K-matrix and a scattered electron wavefunction. Real boundary conditions in matrix notation give the relationship

$$\begin{pmatrix} \boldsymbol{P}^- \\ \boldsymbol{Q}^- \end{pmatrix} = \begin{pmatrix} \boldsymbol{P} \\ \boldsymbol{Q} \end{pmatrix}(\boldsymbol{1} + \mathrm{i}\boldsymbol{K})^{-1} \tag{1.28}$$

1.14 Matching of internal and external region solutions

The inner region solutions will match to a linear combination of the solutions returned by the asymptotic package

$$\begin{pmatrix} P_{ij}(a) \\ Q_{ij}(a) \end{pmatrix} = \sum_{l=1}^{n+n_\mathrm{o}} \begin{pmatrix} F_{il}(a) \\ G_{il}(a) \end{pmatrix} X_{lj} \tag{1.29}$$

the asymptotic boundary conditions being

$$\left(\begin{array}{c} F_{il} \\ G_{il} \end{array}\right) \underset{r\to\infty}{\sim} \left\{\begin{array}{cc} S_i\delta_{il} & l = 1,\ldots,n_\mathrm{o} \\ C_i\delta_{i,l-n_\mathrm{o}} & l = n_\mathrm{o}+1,\ldots,2n_\mathrm{o} \\ D_i\delta_{i,l-n_\mathrm{o}} & l = 2n_\mathrm{o}+1,\ldots,n+n_\mathrm{o} \end{array}\right. \tag{1.30}$$

where $i = 1,\ldots,n$ and then comparing with the definition of the K-matrix in eq.(1.26)

$$\begin{array}{ll} X_{lj} = \delta_{lj} & l,j = 1,\ldots,n_\mathrm{o} \\ K_{ij} = X_{i+n_\mathrm{o},j} & i,j = 1,\ldots,n_\mathrm{o} \end{array} \tag{1.31}$$

If the R-matrix boundary conditions eq.(1.12) are put into the linear expansion then

$$\sum_{l=1}^{n+n_\mathrm{o}} F_{il}(a)X_{lj} = \sum_{l=1}^{n+n_\mathrm{o}} \sum_{m=1}^{n} R_{im}[2acG_{ml}(a) - (b+\kappa_m)F_{ml}(a)]X_{lj} \tag{1.32}$$

which can be solved for X_{lj} and so K_{ij} is found.

Alternatively using matrix notation the asymptotic solutions can be split into six submatrices where X_{ij} is defined slightly differently

$$\begin{array}{lclcl} \boldsymbol{P} & = & \left(\begin{array}{c} \boldsymbol{F}_S \\ 0 \end{array}\right) + \left(\begin{array}{cc} \boldsymbol{F}_C & 0 \\ 0 & \boldsymbol{F}_D \end{array}\right)\left(\begin{array}{c} \boldsymbol{K} \\ \boldsymbol{L} \end{array}\right) & = & \boldsymbol{F}_\mathcal{S} + \boldsymbol{F}_\mathcal{C}\boldsymbol{X} \\ \boldsymbol{Q} & = & \left(\begin{array}{c} \boldsymbol{G}_S \\ 0 \end{array}\right) + \left(\begin{array}{cc} \boldsymbol{G}_C & 0 \\ 0 & \boldsymbol{G}_D \end{array}\right)\left(\begin{array}{c} \boldsymbol{K} \\ \boldsymbol{L} \end{array}\right) & = & \boldsymbol{G}_\mathcal{S} + \boldsymbol{G}_\mathcal{C}\boldsymbol{X} \end{array} \tag{1.33}$$

at $r = a$. Then putting in the R-matrix boundary conditions and rearranging

$$\{\boldsymbol{F}_\mathcal{C} - \boldsymbol{R}[2ac\boldsymbol{G}_\mathcal{C} - (b\mathbf{1}+\boldsymbol{\kappa})\boldsymbol{F}_\mathcal{C}]\}\boldsymbol{X} = \boldsymbol{R}[2ac\boldsymbol{G}_\mathcal{S} - (b\mathbf{1}+\boldsymbol{\kappa})\boldsymbol{F}_\mathcal{S}] - \boldsymbol{F}_\mathcal{S} \tag{1.34}$$

then solve for $\boldsymbol{X}$ and obtain $K_{ij} = X_{ij}$ for $i,j = 1,\ldots,n_\mathrm{o}$.

1.15 Non-relativistic approximation in the external region

Except for cases where the target is highly charged an electron that is scattered or emitted can be described quite well non-relativistically. This was verified by the work of Young (1992 [66]), Young and Norrington (1994 [67]). When $c \to \infty$ then $k_i^2 = 2\epsilon_i$ and the definition of the K-matrix becomes

$$P_{ij} \underset{r\to\infty}{\sim} \left\{\begin{array}{ccc} S_i\delta_{ij} + C_iK_{ij} & \epsilon_i > 0 & i = 1,\ldots,n_\mathrm{o} \\ D_iL_{i-n_\mathrm{o},j} \sim \mathcal{O}(r^{-2}) & \epsilon_i < 0 & i = n_\mathrm{o}+1,\ldots,n \end{array}\right. \tag{1.35}$$

where $j = 1, \dots, n_{\mathrm{o}}$ indexes the linearly independent solutions and

$$S_i = \tfrac{1}{\sqrt{k_i}} \sin \vartheta_i \qquad C_i = \tfrac{1}{\sqrt{k_i}} \cos \vartheta_i \qquad D_i = \tfrac{1}{\sqrt{|k_i|}} \mathrm{e}^{-\hat{\vartheta}_i}$$

$$\vartheta_i(r) = k_i r + \tfrac{z}{k_i} \ln 2k_i r - \arg \Gamma(l_i + 1 - i\tfrac{z}{k_i}) - \tfrac{\pi l_i}{2} \tag{1.36}$$

$$\hat{\vartheta}_i(r) = |k_i| r - \tfrac{z}{|k_i|} \ln 2|k_i| r$$

As before the inner region solutions match to a linear combination of the solutions from the asymptotic package at $r = a$

$$\boldsymbol{P} \quad = \quad \begin{pmatrix} \boldsymbol{F}_S \\ 0 \end{pmatrix} + \begin{pmatrix} \boldsymbol{F}_C & 0 \\ 0 & \boldsymbol{F}_D \end{pmatrix} \begin{pmatrix} \boldsymbol{K} \\ \boldsymbol{L} \end{pmatrix} \quad = \quad \boldsymbol{F}_{\mathcal{S}} + \boldsymbol{F}_{\mathcal{C}} \boldsymbol{X} \tag{1.37}$$

The K-matrix can be found by putting in the non-relativistic R-matrix boundary conditions

$$P_i(a) = \sum_j R_{ij} \left(a \frac{dP_j}{dr} - bP_j \right)_{r=a} \tag{1.38}$$

then rearranging

$$\left[\boldsymbol{F}_{\mathcal{C}} - \boldsymbol{R} \left(a \frac{d}{dr} \boldsymbol{F}_C - b\boldsymbol{F}_{\mathcal{C}} \right) \right] \boldsymbol{X} = \boldsymbol{R} \left(a \frac{d}{dr} \boldsymbol{F}_{\mathcal{S}} - b\boldsymbol{F}_{\mathcal{S}} \right) - \boldsymbol{F}_{\mathcal{S}} \tag{1.39}$$

and solving for $\boldsymbol{X}$ where $K_{ij} = X_{ij}$ for $i, j = 1, \dots, n_{\mathrm{o}}$.

1.16 Electron scattering

For completeness the expression for the electron scattering cross section is given here. The usual expressions for the S-matrix and T-matrix as related to the K-matrix are

$$\boldsymbol{S} = (\boldsymbol{1} + \mathrm{i}\boldsymbol{K})(\boldsymbol{1} - \mathrm{i}\boldsymbol{K})^{-1} \qquad \boldsymbol{T} = \boldsymbol{S} - \boldsymbol{1} \tag{1.40}$$

where these matrices have particular total J and π values. The partial collision strength which contributes to the cross section for a transition of the target from state r to state s is a summation over the channels $\alpha_r J^{\mathrm{t}}_r j_r$ to $\alpha_s J^{\mathrm{t}}_s j_s$ that are coupled to them. α represents all the other quantum numbers that uniquely identify a target state. The total collision strength is a sum of the partial collision strength for each symmetry

$$\Omega^{J\pi}_{rs} = \frac{g}{2} \sum_{j_r j_s} |T_{rs}|^2 \qquad \Omega_{rs} = \sum_{J\pi} \Omega^{J\pi}_{rs} \tag{1.41}$$

where $g = 2J+1$ for jj-coupling and Ω_{rs} should be symmetric. Then the total cross section for this transition is given by

$$\sigma_{r \to s} = \frac{\pi a_0^2}{k_r^2 g_r} \Omega_{rs} \tag{1.42}$$

where $g_r = 2J^{\mathrm{t}}_r + 1$ and a_0 is the Bohr radius to convert from atomic units.

1.17 Bound states

This section summarises details of the R-matrix bound-state method given by Burke and Seaton (1984 [22]) and Seaton (1985 [57]) in vector notation. This includes the basic eigenvalue equation that needs to be solved and how to stop R-matrix poles causing inaccuracy and instability.

1.18 Finding bound state eigenvalues

The column vector of functions $\boldsymbol{P}(r)$ in the inner region describing the continuum electron satisfies the non-relativistic R-matrix equation on the boundary at $r = a$

$$\boldsymbol{P} = \boldsymbol{R}(a\boldsymbol{P}' - b\boldsymbol{P}) \tag{1.43}$$

where $\boldsymbol{R}$is given by eq.(1.21) and $\boldsymbol{P}' = d\boldsymbol{P}/dr$. Matching the $\boldsymbol{P}$ functions to the outer region closed channel solutions $\boldsymbol{F}$ uses the linear expansion

$$\boldsymbol{P} = \boldsymbol{F}\boldsymbol{X} \qquad \boldsymbol{P}' = \boldsymbol{F}'\boldsymbol{X} \tag{1.44}$$

where $\boldsymbol{X}$ is a column vector. Putting in the R-matrix equation gives the general form

$$\boldsymbol{B}(E)\boldsymbol{X} = 0 \tag{1.45}$$

where

$$\boldsymbol{B} = \boldsymbol{F} - \boldsymbol{R}(a\boldsymbol{F}' - b\boldsymbol{F}) \tag{1.46}$$

and solutions exist for discrete bound-state eigenvalues at the zeros of the determinant of $\boldsymbol{B}(E)$. These zeros can be found by scanning the energy range in sufficiently small steps and using a standard method like bisection when a root is bracketed by successive steps.

1.19 Elimination of R-matrix poles

The above method for scanning for bound states may not work if the eigenvalue occurs near an R-matrix pole. This happens quite often when bound-state type functions from the CSF expansion make a large contribution to both an R-matrix eigenfunction and a bound-state eigenfunction. Also high in the Rydberg series with an infinite number of bound states close together there is likely to be at least one close to a pole. But by diagonalising the part of the R-matrix without the pole an equivalent expression to eq.(1.45) can be obtained with the pole multiplied out.

Close to the pole e_k the R-matrix can be written

$$\boldsymbol{R}(E) = \frac{\boldsymbol{S}}{e_k - E} + \boldsymbol{T}(E) \tag{1.47}$$

then diagonalise $\boldsymbol{S}$ with an orthogonal matrix $\boldsymbol{U}$,

$$\boldsymbol{S}\boldsymbol{U} = \boldsymbol{U}\bar{\boldsymbol{S}} \tag{1.48}$$

For example take $\boldsymbol{U}$ such that

$$\bar{S}_{ii'} = \delta_{iI}\,\delta_{i'I}\,\Gamma^2 \qquad \text{and} \qquad \Gamma^2 = \sum_i^I \{w_{ik}\}^2 \tag{1.49}$$

where there are I channels included in $\boldsymbol{F}$. Burke and Seaton (1984 [22]) give example expressions for U.

Now defining $\boldsymbol{L}$ and $\boldsymbol{M}$ by

$$\boldsymbol{L} = \boldsymbol{U}^{\mathrm{T}}[\boldsymbol{F} - \boldsymbol{T}(a\boldsymbol{F}' - b\boldsymbol{F}] \qquad \text{and} \qquad \boldsymbol{M} = \boldsymbol{U}^{\mathrm{T}}(a\boldsymbol{F}' - b\boldsymbol{F}) \tag{1.50}$$

and multiplying equation eq.(1.45) by $\boldsymbol{U}^{\mathrm{T}}$ we can obtain another eigenvalue equation.

$$\bar{\boldsymbol{B}}(E)\boldsymbol{X} = 0 \tag{1.51}$$

with

$$\begin{array}{rcll} \bar{B}_{ii'} & = & L_{ii'} & i \neq I \\ \bar{B}_{Ii'} & = & (e_k - E)L_{Ii'} - \Gamma^2 M_{Ii'} & \end{array} \tag{1.52}$$

Only one of the I equations contained a term with the pole and it has been made definite by multiplying by $(e_k - E)$. Again $\bar{\boldsymbol{B}}(E)$ may be evaluated systematically for different energies to find roots. But in the region of each pole the expression for $\bar{\boldsymbol{B}}(E)$ will be different, i.e. the total energy range needs to be divided up appropriately.

1.20 Photo-ionisation

This derivation of photo-ionisation cross sections is related to the general review by Burke (1976 [21]) and the more specific Dirac R-matrix theory from Chang (1977 [28]).

Various books deal with photo-ionisation and relativistic effects e.g. Bethe and Salpeter (1957 [13]) on one and two electron atoms. Jacobs (1972 [42]) developed a general theory of atomic photo-ionisation measurements using a density matrix and including fine structure. Expressions are given for photo-electron angular distribution and spin polarisation parameters in terms of reduced matrix elements. Kabachnik and Sazhina (1976 [43]) is similar to Jacobs but focuses on the resonances of autoionising states. Walker and Waber (1973 [64]) derive the angular distribution from relativistic orbitals and in terms of matrix elements both for general multipole transitions and the asymmetry parameter β for electric dipole transitions. Grant (1974 [38]) describes the evaluation of relativistic multipole matrix elements in a general gauge and shows the reduction of the electric dipole to the velocity and length forms in the non-relativistic limit.

Burke and Taylor (1975 [20]) describe the non-relativistic R-matrix theory of photo-ionisation and apply the method to Ne and Ar. The thesis of Taylor (1975 [63]) is about this R-matrix theory and applies it to various systems. The thesis of Scott (1980 [54]) describes the development of the Breit-Pauli R-matrix code for electron scattering and photo-ionisation.

1.21 Expressions for the differential cross section

The differential cross section for state i to state j is defined

$$\frac{d\sigma_{ij}}{d\Omega} = \frac{1}{f} W_{ij} \tag{1.53}$$

where f is the incident photon flux and W the transition probability per unit time into states at energies in the range $E \pm \eta$ and comes from Fermi's Golden Rule

$$W_{ij} = \frac{2\pi}{\hbar} |T_{ji}|^2 \rho_j \tag{1.54}$$

Note that this is first order perturbation theory and is valid as long as the time t during which the perturbation T_{ji} acts satisfies the conditions $W_{ij} t \ll 1$ and $\eta t \gg 2\pi\hbar$. See Bransden and Joachain [16]. Working here in SI units.

The transition amplitude is

$$T_{ij} = \langle \Psi^{-}_{jE} | H' | \Psi_i \rangle \tag{1.55}$$

and ρ_j is the density of states for the final state wavefunction Ψ^{-}_{jE}. The interaction part of the Hamiltonian can be expressed in terms of the vector potential $\boldsymbol{A}_l$ by

$$H' = e \sum_{l=1}^{N+1} c\boldsymbol{\alpha} \cdot \boldsymbol{A}_l \tag{1.56}$$

where there are $N + 1$ electrons in the system. The vector potential can be written

$$\boldsymbol{A}_l = A_0(\omega) \hat{\boldsymbol{\epsilon}} (\mathrm{e}^{\mathrm{i}(\boldsymbol{k}\cdot\boldsymbol{r}_l - \omega t)} + \mathrm{e}^{-\mathrm{i}(\boldsymbol{k}\cdot\boldsymbol{r}_l - \omega t)}) \tag{1.57}$$

where $\hat{\boldsymbol{\epsilon}}$ is the polarisation vector of the photon beam and the first and second components of the equation correspond to photon absorption and emission respectively. A_0 is the complex amplitude.

The incident flux can be expressed in terms of the energy flow given by the Poynting vector $\boldsymbol{P}$ and photon energy $\hbar\omega$

$$f = \frac{1}{\hbar\omega} P = \frac{2\varepsilon_0 c\omega}{\hbar} |A_0|^2 \tag{1.58}$$

where ε_0 is the permittivity of free space.

Normalise the initial bound state and final continuum states by

$$\begin{array}{rcl} \langle \Psi_i | \Psi'_i \rangle & = & \delta_{ii'} \\ \langle \Psi^{-}_{jE} | \Psi^{-}_{j'E'} \rangle & = & \delta_{jj'} \delta(E - E') \end{array} \tag{1.59}$$

corresponding to $\rho_j = 1$ in eq.(1.54).

Combining the previous equations and replacing the exponential corresponding to absorption in eq.(1.57) by unity for the wavelengths of interest we obtain the dipole approximation velocity form of the differential cross section.

$$\frac{d\sigma_{ij}^{\mathrm{V}}}{d\Omega} = \frac{\pi e^2}{\varepsilon_0 c\omega} |\langle \Psi_{jE}^{-} | \boldsymbol{D}^{\mathrm{V}} \cdot \hat{\boldsymbol{\epsilon}} | \Psi_i \rangle|^2 \tag{1.60}$$

where the dipole velocity operator is

$$\boldsymbol{D}^{\mathrm{V}} = \sum_{l=1}^{N+1} c\boldsymbol{\alpha}_l \tag{1.61}$$

Use the commutation relation

$$[\boldsymbol{r}, H] = \mathrm{i}c\boldsymbol{\alpha} \tag{1.62}$$

to get the relationship

$$\mathrm{i}c\langle \Psi_a | \boldsymbol{\alpha} | \Psi_b \rangle = (E_b - E_a) \langle \Psi_a | \boldsymbol{r} | \Psi_b \rangle = \hbar\omega_{ab} \langle \Psi_a | \boldsymbol{r} | \Psi_b \rangle \tag{1.63}$$

and thus the length form of the differential cross section.

$$\frac{d\sigma_{ij}^{\mathrm{L}}}{d\Omega} = \frac{\pi\hbar^2 e^2 \omega}{\varepsilon_0 c} |\langle \Psi_{jE}^{-} | \boldsymbol{D}^{\mathrm{L}} \cdot \hat{\boldsymbol{\epsilon}} | \Psi_i \rangle|^2 \tag{1.64}$$

where the dipole length operator is

$$\boldsymbol{D}^{\mathrm{L}} = \sum_{l=1}^{N+1} \boldsymbol{r}_l \tag{1.65}$$

Working for the rest of the discussion in atomic units, the differential cross sections are then written

$$\begin{aligned} \frac{d\sigma_{ij}^{\mathrm{V}}}{d\Omega} &= \frac{4\pi^2 \alpha}{\omega} |\langle \Psi_{jE}^{-} | \boldsymbol{D}^{\mathrm{V}} \cdot \hat{\boldsymbol{\epsilon}} | \Psi_i \rangle|^2 \\ \frac{d\sigma_{ij}^{\mathrm{L}}}{d\Omega} &= 4\pi^2 \alpha\omega |\langle \Psi_{jE}^{-} | \boldsymbol{D}^{\mathrm{L}} \cdot \hat{\boldsymbol{\epsilon}} | \Psi_i \rangle|^2 \end{aligned} \tag{1.66}$$

where $\varepsilon_0 = 1/4\pi$ and α is the fine-structure constant. To convert to other units multiply by a_0^2.

1.22 Cross section in terms of reduced matrix elements

The final continuum state $\Psi_{jE}^{-}(\hat{\boldsymbol{k}})$ may be written in a similar fashion to the non-relativistic partial wave expansion of spherical harmonics and Clebsch-Gordan coefficients as described by Rose (1957 [52])

$$\Psi_{jE}^{-}(\hat{\boldsymbol{k}}) = \sum_{lmjm_jJM} \mathrm{i}^l \mathrm{e}^{-\mathrm{i}\sigma_l} Y_{lm}^{*}(\hat{\boldsymbol{k}}) C(lsj; mm_s m_j) C(J^{\mathrm{t}} jJ; M^{\mathrm{t}} m_j M) \Psi_{jE}^{-} \tag{1.67}$$

The minus sign in $\Psi^{-}_{jE}(\hat{\boldsymbol{k}})$ and Ψ^{-}_{jE} means that the wavefunctions have the asymptotic form of a Coulomb-modified plane wave in the final state channel and incoming spherical waves in all open channels. The phase shift of the lth partial wave is then $\sigma_l = \arg\Gamma[l{+}1{-}i(Z{-}N)/k]$, c.f. Bransden (1983 [15]). Also see Walker and Waber (1973 [64]) or Bethe and Salpeter (1957 [13]).

Using the Wigner-Eckart theorem

$$\langle\Psi^{-}_{jE}(\hat{\boldsymbol{k}})|D_{1\mu}|\Psi_i\rangle = (-1)^{J-M}\begin{pmatrix} J & 1 & J_i \\ -M & \mu & M_i \end{pmatrix}\langle\Psi^{-}_{jE}(\hat{\boldsymbol{k}})\|D_1\|\Psi_i\rangle \tag{1.68}$$

$D_{1\mu}$ is either the dipole length operator or the dipole velocity operator combined with the photon polarisation μ. The $\langle\Psi^{-}_{jE}(\hat{\boldsymbol{k}})\|D_1\|\Psi_i\rangle$ are reduced dipole matrix elements. Then substituting eq.(1.67) and eq.(1.68) into the expressions for the differential cross sections eq.(1.66), averaging over the initial magnetic quantum numbers and summing over the final magnetic quantum numbers gives expressions with the standard form for light polarised along the z-axis and circularly polarised light

$$\begin{aligned} \frac{d\sigma_{ij}}{d\hat{\boldsymbol{k}}} &= \frac{\sigma}{4\pi}[1 + \beta \mathrm{P}_2(\cos\theta)] \\ \frac{d\sigma_{ij}}{d\hat{\boldsymbol{k}}} &= \frac{\sigma}{4\pi}[1 - \frac{1}{2}\beta \mathrm{P}_2(\cos\theta)] \end{aligned} \tag{1.69}$$

Where β is a combination of 3-J, 6-J coefficients and reduced matrix elements. Integrating over all ejected-electron angles $\hat{\boldsymbol{k}}$ and averaging over the photon polarisation gives the total cross section

$$\sigma = \frac{8\pi^2\alpha C}{3(2J_i+1)}\sum_{ljJ}|\langle\Psi^{-}_{jE}\|D_1\|\Psi_i\rangle|^2 \tag{1.70}$$

where $C = \omega$ in the length approximation and $C = \omega^{-1}$ in the velocity approximation.

1.23 Dipole matrix elements

The reduced dipole matrix elements between two states a and b can be split into a contribution from the internal region (I) and from the external region (E), where the integrals are performed over the ranges $0 \le r < a$ and $a \le r \le \infty$ respectively

$$\langle\Psi_a\|D_{1\mu}\|\Psi_b\rangle = \langle\Psi_a\|D^{(\mathrm{I})}_{1\mu}\|\Psi_b\rangle + \langle\Psi_a\|D^{(\mathrm{E})}_{1\mu}\|\Psi_b\rangle \tag{1.71}$$

For photo-ionisation from low n initial states, i.e. orbitals closely bound to the target, the contribution to the dipole matrix elements from the external region will be negligible or small because of the rapidly decaying exponential functions. The R-matrix boundary can be increased to check convergence if this contribution is not being calculated.

1.23.1 Internal region contribution

The internal region contribution can be expanded in terms of the R-matrix basis

$$\langle\Psi_a\|D^{(\mathrm{I})}_{1\mu}\|\Psi_b\rangle = \sum_{k,k'} A_{E_ak}\langle\Psi_k\|D^{(\mathrm{I})}_{1\mu}\|\Psi_{k'}\rangle A_{E_bk'} \tag{1.72}$$

In matrix notation this can be written

$$\langle\Psi_a\|D^{(\mathrm{I})}_{1\mu}\|\Psi_b\rangle = \boldsymbol{P}^{\mathrm{T}}_a\boldsymbol{R}^{-1}_a\boldsymbol{w}_a\boldsymbol{G}_a\boldsymbol{V}^{\mathrm{T}}_a\boldsymbol{M}\boldsymbol{V}_b\boldsymbol{G}_b\boldsymbol{w}^{\mathrm{T}}_b\boldsymbol{R}^{-1}_b\boldsymbol{P}_b \tag{1.73}$$

using reduced matrix elements from the $(N+1)$-electron CSF basis

$$M_{ll'} = \langle\Theta_l||D^{(\mathrm{I})}_{1\mu}||\Theta_{l'}\rangle \tag{1.74}$$

with expressions obtained from eq.(1.16), eq.(1.20) and eq.(1.21) for state a

$$\begin{aligned}\Psi_k &= \sum_l \Theta_l V_{lk}\\ A_{E_ak} &= \sum_m G_{km}\sum_j w^{\mathrm{T}}_{mj}\sum_i R^{-1}_{ji}P_i(a)\\ G_{km} &= [2a(E_k-E_a)]^{-1}\delta_{km}\end{aligned} \tag{1.75}$$

where the eigenvalues, eigenvectors and surface amplitudes are from the $J\pi$ symmetry corresponding to state a. Alternatively the expression for the A_{Ek} expansion coefficients eq.(1.20) could be used directly without requiring the inverse of the R-matrix .

1.23.2 External region contribution

The external region contribution ignores exchange and thus asymmetrisation. For example the dipole length operator is treated as two parts, one for the target $\boldsymbol{R} = \sum_{l=1}^{N}\boldsymbol{r}_l$ and the other for the outer electron $\boldsymbol{r} = \boldsymbol{r}_{N+1}$. Using non-relativistic asymptotic solutions, the contribution is expressed as a sum over the channels

$$\begin{aligned}\langle\Psi_a\|D^{(\mathrm{E})}_{1\mu}\|\Psi_b\rangle &= \sum_{i,j}\alpha_{ij}\langle P_{ia}|P_{jb}\rangle + \beta_{ij}\langle P_{ia}|r|P_{jb}\rangle\\ \alpha_{ij} &= \mathrm{Ang}\langle[\Phi_i,\phi_i]\|\sum_{k=1}^{N}\boldsymbol{r}_k\cdot\hat{\boldsymbol{\epsilon}}\|[\Phi_j,\phi_j]\rangle\\ \beta_{ij} &= \mathrm{Ang}\langle[\Phi_i,\phi_i]\|\hat{\boldsymbol{r}}\cdot\hat{\boldsymbol{\epsilon}}\|[\Phi_j,\phi_j]\rangle\end{aligned} \tag{1.76}$$

As before, Ang means that the integral over the outer electron radial coordinate is omitted. The α coefficient is non-zero if there is a dipole transition between the target states of the channels involved and the angular momentum of the outer electron does not change. The β coefficient is non-zero if the channels involved have the same target state because the dipole transition is by the outer electron. Seaton (1986 [58]) describes how the integrals are evaluated for the non-relativistic code. The length and velocity dipole matrix elements can be expressed in terms of the acceleration gauge, which allows the matrix elements to be approximated with surface terms and integrals that converge rapidly with increasing r.

The calculation of free-free transitions cross sections also requires this significant contribution to the matrix elements because of the continuum orbitals involved. This process is described, with an expression for the outer region contribution, by Bell et al. (1977 [7]).

1.24 Other related processes

1.24.1 Oscillator strengths

Dipole transition oscillator strengths $f(b,a)$, and thus transition rates, can be calculated like photo-ionisation cross sections but of course using two R-matrix bound states.

$$f(b,a) = \frac{2C}{3g_a}|\langle\Psi_a\|D_1\|\Psi_b\rangle|^2 \tag{1.77}$$

where $g_a = 2J_a + 1$ is the statistical weight of the initial state a in jj-coupling. $C = \omega$ in the length gauge and $C = \omega^{-1}$ in the velocity gauge.

1.24.2 Free-free transitions

As mentioned previously, free-free transitions, electron scattering combined with photo-absorption, can be calculated in a similar manner to photo-ionisation except using a continuum initial state. The process is important in the solar atmosphere and in the heating of plasmas in general. The outer region contribution to the dipole matrix elements is required for this type of calculation. Bell et al. (1977 [7]) describe the non-relativistic R-matrix method for differential and total cross sections and any photon polarisation.

1.24.3 Frequency dependent polarisability

The atomic dynamic and static polarisability can be also be calculated in a similar fashion to the photo-ionisation. As a first approximation the external region can be ignored. Only the initial bound state, dipole matrix elements and some Clebsch-Gordan coefficients are required, as described by Allison et al. (1972 [1]) using the R-matrix method for oxygen. There are well known relationships between the polarisability and other physical quantities like the refractive index of a gas. Allison et al. derive the correspondence between the imaginary part of the polarisability and the total photo-ionisation cross section, and calculate the polarisability for imaginary frequencies that is needed to find the Van der Waals C_6 coefficient.

Shorer (1980 [61]) gives a slightly different description of the R-matrix method and shows that surface terms are required to get resonances in the correct positions.

1.24.4 Two-photon photo-ionisation

Shorer (1980 [61]) also describes how the method used for polarisability can also be applied to the time independent perturbation theory matrix element for two-photon photo-ionisation. This results in a method that is not much more complicated than for single-photon photo-ionisation. Shorer also suggests how to extend the R-matrix method to three-photon and generalise to N-photon processes.

Chapter 2

General information

2.1 What is DARC?

DARC (Dirac Atomic R-matrix Codes) is a suite of programs that enables the study of continuum processes for a general atomic system. Modules allow the calculation of electron-atom or electron-ion collision cross-sections. In addition the programs include code that allows for bound-state and photoionisation calculations.

DARC is closely related to the well-known atomic structure package GRASP developed by Grant and associates (Dyall et al 1989 [34], Parpia et al 1996 [50]).

A version of GRASP, known as GRASP0, is included as a DARC module. This version is derived from the original MCDF/BENA codes (Grant et al 1980 [39], McKenzie et al 1980 [45]) and is similar to (but not the same as) the Dyall et al (1989 [34]) version. GRASP0 shares much code with the DSTG2 module, mainly concerned with the angular momentum algebra. A version of GRASP0 that only does the angular integrals is known as ANGUL.

The R-matrix method is closely related to the close-coupling method. In fact it can be viewed as a numerical method for solving the close-coupling equations. This approach has been pioneered in atomic and molecular physics by Burke and associates. The monograph by Burke and Berrington (1993 [23]) gives an excellent overview of the method and its applications spanning twenty years.

The non-relativistic theory due to Burke et al (1971 [19]) was programmed by Berrington et al (1974 [9], 1978 [10]) as the package RMATRX. Scott and Taylor (1982 [56]) published a suite of programs that included model potentials and relativistic effects. The theory was based on the Breit-Pauli hamiltonian and used intermediate coupling. A revised version RMATRX1 has been published by Berrington et al (1995 [12]). This package is appropriate for systems in which the spin-orbit interaction can be treated as a perturbation.

DARC uses the Dirac-Coulomb hamiltonian and jj coupling. This gives a better description of target fine-structure. The package can be applied to atomic systems throughout the periodic table. The underlying theory is described in detail by Chang (1975 [27]). The theory was re-examined by Szmytkowski and Hinze (1996 [62]) and they discovered an error in the matching formulae at the R-matrix boundary. However this correction does not affect the results obtained with DARC even though DARC is based on Chang's erroneous formulae. From a computational viewpoint DARC has a structure similar to RMATRX1.

The method is limited to low-energy scattering involving a small number of target states. The limitation is one of computer memory and CPU time. If you include 100 target states then you may well generate up to 400 scattering channels. Each channel may involve up to 20 continuum orbitals and this gives a continuum hamiltonian matrix of dimension 400*20=8000. This matrix must be diagonalised and the procedure repeated for a number of electron+target symmetries (say 20 to 30). Thus the inclusion of target fine-structure through the Dirac equation and jj coupling can involve a substantial computational commitment.

2.2 Outline of Dirac R-matrix theory

Units

The equations that have been programmed are expressed in Hartree atomic units (au). In the output other 'energy' units such as Ryd, eV, cm^{-1} or Ang may be used.

Orbitals

The atomic wavefunctions in DARC are built from one-electron functions that have a prescribed angular momentum. These are sometimes called Dirac spin-orbitals or simply orbitals. The spin-angular part of an orbital is known. In the Dirac theory each orbital has two radial components. One of these, known as the large component, will in the non-relativistic limit reduce to the radial component of a Schrödinger orbital. The other component, known as the small component, tends to zero in the non-relativistic limit.

The angular momentum quantum numbers for an orbital are $sljm$. The spin angular momentum is $s = 1/2$, while $l = 0, 1, 2, \ldots$ is the orbital angular momentum. The total angular momentum j can take two values, $j = l + 1/2$ and $j = l - 1/2$, except when $l = 0$ and only $j = 1/2$ occurs. For a given j there are $2j + 1$ values for the magnetic quantum number m, namely it ranges from $-j$ to j in integral steps. The quantum numbers lj can be replaced by a single quantum number κ. This is a non-zero integer.

1. Negative κ correspond to $j = l + 1/2$ and $\kappa = -l - 1$.
2. Positive κ correspond to $j = l - 1/2$ and $\kappa = l$.

In addition we use the K quantum number in some of the code. The following table illustrates the relationship between these quantum numbers.

K	κ	l	j	label
1	-1	0	1/2	s
2	1	1	1/2	p−
3	-2	1	3/2	p
4	2	2	3/2	d−
5	-3	2	5/2	d
6	3	3	5/2	f−
7	-4	3	7/2	f

Orbitals with differing κ or m are orthogonal by virtue of the spin-angular functions. When the κ-values are the same then orthogonality becomes a condition imposed on the radial components. This orthogonality condition is a requirement of the underlying angular momentum algebra as implemented in this version of DARC.

Orbitals associated with the target are referred to as bound. These may be obtained from a program such as GRASP. Their main characteristic is that they tend to zero at large radii. They are thus appropriate building-blocks for the target wavefunction. They may be Dirac-Fock orbitals, optimised on some state of the atomic system, but this is not necessary. It is common to include one-electron functions, sometimes called pseudo-orbitals, that attempt to account for correlation effects. All of these orbitals are called bound.

In contrast orbitals that are associated with the free electron are referred to as continuum. In the R-matrix method these orbitals satisfy prescribed boundary conditions at the R-matrix boundary. Combining the bound and continuum orbitals gives a set of functions that are used for the expansion of the total wavefunction.

Configurations

R-matrix theory

The scattering problem is divided into two regions by a spherical boundary at $r = a$. The boundary encloses most of the target wavefunction.

In the internal region, $r < a$, the continuum wavefunction is expanded in terms of a set of continuum orbitals that satisfy prescribed boundary conditions at the origin and at $r = a$. These continuum orbitals resemble in many ways the bound orbitals used in atomic structure packages such as GRASP. The treatment of the internal region is similar to a structure calculation and much of the analysis and coding can be taken over from packages such as GRASP.

In the external region, $r > a$, the target wavefunction is small and so exchange between the target and the continuum electron can be neglected. The close-coupling equations are then considerably simplified and the scattering channels are coupled by long-range potentials of the form $1/r^{\lambda+1}$ where $\lambda = 1, 2, 3 \ldots$. A set of linearly independent solutions to the scattering equations in the external region is calculated and the K-matrix is obtained by matching these to the internal solution at the R-matrix boundary.

A significant advantage of the R-matrix method is that the internal region calculation is only done once. It does not depend on the energy of the continuum electron. However the external region calculation must be repeated for each continuum electron energy required. When a large number of energies are required (e.g. , in a resonance region), then it is necessary to have an efficient means for solving the external region problem.

A further approximation in the external region is to replace the Dirac close-coupling equations by their non-relativistic limit. A number of packages for solving the non-relativistic problem are available. This is a good approximation since the channel energies are usually small and the potentials are comparatively weak.

Photoionisation

Abbreviations

The following abbreviations are used.

AL	average level
ASF	atomic state function
CFP	coefficient of fractional parentage
CI	configuration interaction
CSF	configuration state function
EAL	extended average level
EOL	extended optimal level
MCDF	multi-configurational Dirac-Fock
OL	optimal level
QED	quantum electrodynamics
SCF	self-consistent field
STO	Slater type orbitals

2.3 Language, precision and compilation

DARC is written in FORTRAN 77. It is designed to be transportable and has been implemented on a number of computers with comparative ease. The present version can be used with the g77 compiler available with Linux.

The programs use double precision.

LAPACK and BLAS routines are used for the linear algebra. Many machines provide these routines as optimised libraries. If this is not the case then the FORTRAN source has been included as part of DARC. This can be compiled and linked to the relevant modules. Alternatively create your own libraries.

Dimensions are set by means of INCLUDE statements within the code. For many of the modules the file to be included is called darc.inc. You only need to edit this small file in order to change the dimensions before recompilation.

There is some limited preprocessing. This has been kept simple and can be carried out with a text editor such as `sed`.

The machine specific code is mostly confined to the routines CALEN and QUARTZ and involves accessing the time and date at execution of the program and the processor clock for timings. Versions of these routines for various compilers are supplied and you must link the appropriate ones. It is possible to choose dummy ones (`calen.none` and `quartz.none`) if you are unsure of the correct system calls.

Using sed

As an example of using a text editor to preprocess a code, consider the UNIX stream editor, `sed`. If the raw (unprocessed) code is `dstgh.raw` and the `sed` commands are in file `dstgh.sed` then you would issue the following command:

```
sed -f dstgh.sed dstgh.raw > dstgh.f
```

The processed output is piped to file `dstgh.f`. This can be compiled in the normal way.

The format of the `sed` commands is for example

```
s/LAPACK/C/g
```

which is a global (i.e. , the ending `g`) substitution of the text `LAPACK` by `C`. In practice this would comment out lines that begin with the string `LAPACK`.

2.4 Package layout

The layout of the package is shown below.

```
GRASP0 (NJGRAF or NJSYM)             atomic structure package provides
                                     the bound orbitals
  |

DSTG0                                interface between GRASP etc and
                                     DSTG1: creates the file TARGET.INP
  |                                  containing bound orbitals

DSTG1/ORBS                           calculates continuum orbitals,

  |

DSTG1/INTS                           calculates radial integrals

  |

DSTG2 (NJGRAF or NJSYM)              defines channels,
                                     calculates angular integrals,
  |                                  continuum hamiltonians,
  |                                  asymptotic scattering coefficients,
  |                                  dipole matrix elements
  |                                  (photoionisation)

DSTGH (HSLDR or LAPACK)              diagonalises hamiltonians

  |

DSTG3 i.e. DASYM/DASYPCK/DCOUL/DVPM  external region:
or STGFJJ                            asymptotic solutions,
                                     collision strengths
  |

DSTG4                                analyses collision strengths
```

For photoionisation the modules DSTG3 and DSTG4 are replaced by the following. In addition DSTG2 calculates dipole matrix elements.

```
  |
  |
```

`PDSTG3 i.e. PASYPCK/PCOUL`	`external region:`
	`asymptotic solutions,`
	`cross sections`

2.5 Advice on using DARC

You will need a good target description. This can be obtained from a package such as GRASP.

Time spent at this stage is vital. There is a limit to the number of configurations that you can reasonably use with present day resources. Careful consideration of the target will save time later.

Most applications of the code have used targets obtained from EAL (extended average level) GRASP calculations. This type of calculation is comparatively easy to converge and gives a set of bound orbitals which is optimised over all the levels included. It is a compromise set which is appropriate in a scattering situation involving excitation of the lowest states.

You should always run through the modules with a small (say one electron+target symmetry) calculation first. This is just to gauge the size of the problem that you are attempting and to check the input.

Before a large scale calculation:

1. Check that the R-matrix boundary is correct.

2. Check that the radial mesh is adequate. Be conservative and pick a mesh with a small step-size. The program does this automatically but you should still check. You should have at least 15 mesh points between the nodes of each orbital.

3. Check that the target potential is correct. It is a good idea to plot the potential and the bound orbitals.

4. Check that the number of continuum orbitals per angular momentum (variable NRANG2) is adequate. It should be at least 20 for heavy atoms. I would rarely pick less than 20. This is partly determined by the energy range over which you wish to calculate cross-sections. Again this emphasizes the importance of a small preliminary calculation.

5. Check that the DSTG2 target energies are consistent with the target energies originally calculated by GRASP.

6. DSTG2 prints out the possible asymptotic channel couplings. The variable LAMBB is set in DSTG1/INTS and determines which couplings are included. Check that LAMBB allows for all the couplings that you wish to include.

2.6 I/O

Input data is read from stream IREAD which is set to 5. The input file is opened internally and has the name `file.INP` where `file` would be `ORBS`, `INTS`, `DSTG2`, `DSTGH`, `DSTG3`, `PDSTG3` or `STGF`. If the input file does not exist then the program terminates with an error message.

Code has been included to write out the input data at the start of the calculation. This uses a scratch file and stream 9.

DSTG0, DSTG4, DUMP1 and DUMPH are interactive programs and do not require input files.

Formatted output is written to stream IWRITE which is set to 7. This output file is opened internally and has the name `file.OUT` where `file` would be `DSTG0`, `ORBS`, `INTS`, `DSTG2`, `DSTGH`, `DSTG3`, `PDSTG3`, `STGF`, `DSTG4` `DUMP1` or `DUMPH`. If a file with this name already exists then it is overwritten.

There are also some PRINT statements which write to stream 6. If the programs are run interactively then this gives some indication on the screen of the progress of the job.

Beware overwriting of files

Most of the files used in the program are opened with STATUS set to UNKNOWN. This means that if the file already exists, then it will be overwritten. It is important that you rename files that you wish to keep. This is particularly important in DSTG3 where the collision strengths are written to `DSTG3.DAT`. Generally DSTG3 is run a large number of times for a given case as the incident energy and symmetries are varied.

Data for plotting

It is useful to have formatted files containing data that can be easily accessed by a graphics package. In DARC files that are essentially ksh or bash scripts contain this data. By running the script the file explodes into separate files, one for each orbital say. A list of the orbitals and their contents is in file `INDEX`. You can easily import such files into a graphics package such as GNUPLOT.

code	data	file name
DSTG0	target orbitals	`TARGET.SH`
DSTG1/ORBS	orbitals and potentials	`ORBS.SH`
DSTG3	phase-shifts	`PHASE.SH`
DSTG4	cross-sections or collision strengths	name input by user)
PDSTG3	phase-shifts cross-section	`PHASE.SH` `PHOTO.SH`

Unformatted files and transfer between machines

Another difficulty is the transfer of unformatted files between different machines. For this reason the `DSTG3.DAT` files produced by DSTG3 and STGFJJ are now formatted. These files contain collision strengths. DSTG4 can still read the old unformatted form of these files.

Also an option (option 33) allows DSTGH to write either formatted or unformatted dump files. A module called DUMPH can be used to examine and combine DSTGH dump files. This module will also convert between formatted and unformatted and vice-versa as requested. Input to DUMPH is interactive.

2.7 Dimensions

Dimensions are set by the file `darc.inc` which is included in the relevant routines. All dimension variables beginning with MX are set in this file. To change a dimension you need only change `darc.inc`.

A sample `darc.inc` is given below.

```
C-----------------------------------------------------------------------
C   darc.inc
C   This file is INCLUDEd in the DARC modules.
C   It is used to set dimensions.
C   The following table indicates where the dimensions are used.
C
C           dstg0  dstg1  dstg2  dstgh  dstg3  pdstg3  dstg4
C
C   MXCH                    *      *      *      *
C   MXHD                           *      *      *
C   MXI1             *      *
C   MXI2             *      *
C   MXNA                           *      *      *
C   MXNB             *      *      *      *      *
C   MXNC                    *
C   MXNK             *      *      *      *      *
C   MXNL                    *      *      *      *
C   MXNM                    *
C   MXNP      *      *
C   MXNW      *             *
C   MXP1      *      *
C   MXNAST                                               *
C   MXNREC                                               *
C   MXNSYM                                               *
C   MXNENE                                               *
C
C   Note:
C
C   1. dstg3 means dasym,dasypck,dcoul,dvpm
C   2. pdstg3 means pasypck,pcoul
C
C           asycoul   asypck   asyvpm   hsldr
C
```

```
C   MXCH      *         *        *
C   MXHD                                  *
C   MXNA                *        *
C   MXNK      *                  *
C
C-----------------------------------------------------------------------
C   MXCH   channels
C   MXHD   size of Hamiltonian
C   MXI1   radial integrals
C   MXI2   marks position of radial integrals
C   MXNA   lambda-values for asymptotic coefficients
C   MXNB   continuum orbitals per KAPPA value
C   MXNC   relativistic configurations
C   MXNK   KAPPA values
C   MXNL   target levels or correlation functions
C   MXNM   MCP angular coefficients
C   MXNP   points in radial mesh
C   MXNW   relativistic orbitals
C   MXP1   mesh points used to store orbitals
C   MXNAST target states
C   MXNREC records
C   MXNSYM symmetries per energy point
C   MXNENE energy points
C-----------------------------------------------------------------------
      INTEGER MXCH,MXHD,MXI1,MXI2,MXNA,MXNB,MXNC
      INTEGER MXNK,MXNL,MXNM,MXNP,MXNW,MXP1
      INTEGER MXNAST,MXNREC,MXNSYM,MXNENE
      PARAMETER (MXCH = 200)
      PARAMETER (MXHD = 2000)
      PARAMETER (MXI1 = 150000)
      PARAMETER (MXI2 = 300)
      PARAMETER (MXNA = 4)
      PARAMETER (MXNB = 40)
      PARAMETER (MXNC = 600)
      PARAMETER (MXNK = 100)
      PARAMETER (MXNL = 300)
      PARAMETER (MXNM = 130000)
      PARAMETER (MXNP = 1000)
      PARAMETER (MXNW = 150)
      PARAMETER (MXP1 = 100000)
      PARAMETER (MXNAST = 27)
      PARAMETER (MXNREC = 20000)
      PARAMETER (MXNSYM = 60)
      PARAMETER (MXNENE = 2000)
C-----------------------------------------------------------------------
```

The array IDMTST is used throughout the modules to check the dimensions. Below the elements of this array are defined.

- IDMTST(1)

contains the dimension of the arrays containing the number of channels. It is set by MXCH.

- IDMTST(2)
  contains the dimension of the Hamiltonian matrix. It is set by MXHD.

- IDMTST(3)
  contains the dimension of the array RKSTO used to store radial integrals. It is set by MXI1.

- IDMTST(4)
  contains the dimension of the arrays ISTX1 and ISTX2. It is set by MXI2.

- IDMTST(5)
  contains the last dimension of the array CF and corresponds to the number of λ values in the asymptotic coefficients. It is set to MXNA.

- IDMTST(6)
  contains the maximum number of continuum orbitals per angular momentum. It is set by MXNB.

- IDMTST(7)
  contains the maximum number of CSFs. It is set to MXNC.

- IDMTST(8)
  contains the maximum value of KX, the number of bound/continuum κ-values. It is set to MXNK.

- IDMTST(9)
  contains the maximum number of target CSFs or correlation functions. It is set by MXNL.

- IDMTST(10)
  contains the maximum number of MCP angular coefficients. It is set by MXNM.

- IDMTST(11)
  contains the dimension of arrays used to tabulate mesh, orbitals and potentials. It is set to MXNP.

- IDMTST(12)
  contains the maximum number of target orbitals. It is set to MXNW.

- IDMTST(13)
  is the maximum number of points that can be used for storing orbitals. It is set to MXP1.

- IDMTST(14)
  contains the maximum number of target states. It is set to MXNAST. Used in the DSTG4 module.

- IDMTST(15)
  contains the maximum number of records where a record corresponds to a symmetry/energy combination. It is set to MXNREC. The maximum possible value is MXNSYM$*$MXNENE but it is more flexible to allow a smaller dimension. Used in the DSTG4 module.

- IDMTST(16)
  contains the maximum number of symmetries per energy. It is set to MXNSYM. Used in the DSTG4 module.

- IDMTST(17)
  contains the maximum number of energy points. It is set to MXNENE. Used in the DSTG4 module.

- IDMTST(21)
  contains the maximum value of NBT, the number of bound orbitals to which the continuum orbitals can be orthogonalised using Lagrange multipliers. It is set to ND11=12. Used in the DSTG1 module.

- IDMTST(22)
  contains the maximum number of bound orbitals to which a continuum orbital can be Schmidt orthogonalised. It is set to 10. Used in the DSTG1 module.

- IDMTST(23)
  is the maximum pseudo principal quantum number allowed in array IPOS. It is set to ND15=MXNB+MXNK/2. This assumes that MXNK is sufficiently large, say 20. Used in the DSTG1 module.

- IDMTST(24)
  contains the dimension of array IRX and is the maximum number of subintervals allowed in the mesh. It is set to 20.

- IDMTST(25)
  contains the dimension of the arrays defining the number of energy ranges. This is set to variable NDENER. Used in the DSTG3/PDSTG3 modules.

- IDMTST(26)
  contains the maximum number of open shells in a nonrelativistic CSF. At present this must be 5. Used in the DSTG2 module.

2.8 Compilation

The following table shows the dependence of the various modules on the LIB modules.

dir	code	sys	_misc	lapack
dstg0	dstg0.raw	x	x	
dstg1	dstg1int.raw	x	x	
dstg1	dstg1orb.raw	x	x	x
dstg2	dstg2.raw	x	x	x
dstgh	dstgh.raw	x	x	x
dstg3	dcoul.raw	x	x	x
dstg3	dasypck.raw	x	x	x
dstg3	dvpm.raw	x	x	x
dstg3	pcoul.raw	x	x	x
dstg3	pasypck.raw	x	x	x
dstg4	dstg4.raw	x	x	
util	dump1.raw	x	x	
util	dumph.raw	x		
LIB	_asycoul.raw		x	x
LIB	_asypck.raw		x	x
LIB	_asyvpm.raw			x
LIB	_njgraf.raw		x	
LIB	_njsym.raw		x	

dir	code	_njgraf	_njsym	_hsldr	_asycoul	_asypck	_asyvpm
dstg2	dstg2.raw	x	x				
dstgh	dstgh.raw			x			
dstg3	dcoul.raw				x		
dstg3	dasypck.raw					x	
dstg3	dvpm.raw						x
dstg3	pcoul.raw				x		
dstg3	pasypck.raw					x	

Of course it is important to compile the modules in the correct order. The bash script `xcompile-on-g77` is available for compiling using the g77 compiler on a linux system. This is used in the discussion below.

It is necessary to compile LIB/sys, LIB/blas, LIB/lapack and LIB/_misc prior to the compilation of any other modules.

```
>>    xcompile-on-g77 LIB sys    nolink
>>    xcompile-on-g77 LIB blas   nolink
>>    xcompile-on-g77 LIB lapack nolink
>>    xcompile-on-g77 LIB _misc  nolink
```

The following modules can now be compiled.

```
>>    xcompile-on-g77 dstg0 dstg0    LIB
>>    xcompile-on-g77 dstg1 dstg1orb LIB
>>    xcompile-on-g77 dstg1 dstg1int LIB
>>    xcompile-on-g77 dstg4 dstg4    LIB
>>    xcompile-on-g77 util  dump1    LIB
>>    xcompile-on-g77 util  dumph    LIB
```

If the dstg2 module is required, then it is necessary to compile the recoupling packages LIB/_njgraf and LIB/_njsym.

```
>>    xcompile-on-g77 LIB _njgraf nolink
>>    xcompile-on-g77 LIB _njsym  nolink
>>    xcompile-on-g77 dstg2 dstg2 LIB
```

If the diagonalisation package HSLDR is used by module dstgh (selected in dstgh.sed) then you must compile LIB/_hsldr.

```
>>    xcompile-on-g77 LIB   _hsldr nolink
>>    xcompile-on-g77 dstgh dstgh  LIB
```

The asymptotic packages LIB/_asycoul, LIB/_asypck and LIB/_asyvpm should be compiled if the corresponding dstg3 modules are needed i.e.

LIB/_asycoul	for dstg3/dcoul or dstg3/pcoul
LIB/_asypck	for dstg3/dasypck or dstg3/pasypck
LIB/_asyvpm	for dstg3/dvpm

```
>>    xcompile-on-g77 LIB   _asycoul nolink
>>    xcompile-on-g77 dstg3 dcoul    LIB
>>    xcompile-on-g77 dstg3 pcoul    LIB

>>    xcompile-on-g77 LIB   _asypck  nolink
>>    xcompile-on-g77 dstg3 dasypck  LIB
>>    xcompile-on-g77 dstg3 pasypck  LIB

>>    xcompile-on-g77 LIB   _asyvpm  nolink
>>    xcompile-on-g77 dstg3 dvpm     LIB
```

The script `xcompile-darc` is available for compilation of all DARC modules at once.

Also scripts are available for compiling individual modules e.g. `xcompile-dstg0`, etc. .

Scripts are available for compiling with other compilers. See `xcompile-darc` for information.

Chapter 3

DSTG0

3.1 Introduction

$Source: /home/phn/DARC/RCS/dstg0.mak,v
$Author: phn
$Date: 2001/10/31 12:12:08
$Revision: 11.1

DARC, the Dirac Atomic R-matrix Codes.
homepage: http://www.am.qub.ac.uk/DARC/
DSTG0 module

This module will read bound orbitals from GRASP dumps and write the formatted file `TARGET.INP` that is required for module DSTG1.

It is run **interactively**.

Calling tree:

```
DSTG0
    AAMN0
        CALEN
        DMSET0
        MCDF09
            DMCHK0
                DMPRT0
            DMPRT0
        MCDF10
            DMCHK0
                DMPRT0
            DMPRT0
        IN2CH3
        DMPRT0
```

The orbitals would normally be obtained from a package such as GRASP. They are Dirac two-component (large and small) radial wavefunctions. It is possible to input Slater-type orbitals, e.g. from CIV3, into GRASP0. This would allow a nonrelativistic target to be used.

You can create your own `TARGET.INP` and bypass this module.

An option allows the orbitals to be written out to the formatted file `TARGET.SH`. This file takes the form of a ksh or bash script that can be executed to produce an index file (called `INDEX`) and one file for each orbital (called `orbxxx`, where `xxx` is an integer). The index file gives information on the orbital files. Each orbital file has 4 columns, col.1 is the radial coordinate, col.2 is the large component (P), col.3 is the small component (Q), col.4 is the density ($P^2 + Q^2$). A perl script, `draw-wave.phn` is available which will process the index and orbital files and produce a GNUPLOT script. This allows easy viewing of the orbital functions.

3.2 Files

The files are defined internally. Messages are also written to output stream 6 using the PRINT command.

The following files are used:

stream name	description	stream number	file name	FORM
IWRITE	Output.	7	`DSTGO.OUT`	Formatted
JREAD	Input GRASP dump file.	10	Specified by user.	Unformatted
JWRITE	File containing target orbitals.	20	`TARGET.INP`	Formatted
JWRITE	Shell script containing target orbitals.	20	`TARGET.SH`	Formatted

3.3 Input data

Input is interactive.

3.4 Target dump (`TARGET.INP`)

The input records D to G are repeated for J=1,NW.

```
A. NW,N
B. (UCF(J),J=1,NW)
C. (DR(I),I=1,N)

D. NP(J),NAK(J)
E. (CP(I),I=1,N)
F. NP(J),NAK(J)
G. (CQ(I),I=1,N)
```

The variables are:

- NW
  is the number of orbitals.
- N
  is the number of radial mesh points.
- UCF(J) J=1,NW
  is the orbital occupation number for orbital J.
- DR(I) I=1,N
  is the radial mesh.

- NP(J) J=1,NW
  is the principal quantum number for orbital J.
- NAK(J) J=1,NW
  is the κ quantum number for orbital J.
- CP(I) I=1,N
  is an array containing orbital large components.
- CQ(I) I=1,N
  is an array containing orbital small components.

3.5 Routines

Routine DSTG0

This is the PROGRAM routine. The output file is set up: IWRITE (`DSTG0.OUT`). A call is made to routine AAMN0 which controls the computation.

Routine AAMN0

```
(IREAD, IWRITE)
```

This is the main controlling routine for the DSTG0 module.

Routine DMCHK0

```
(I, ID, IWRITE, IDMTST, NDIMAX)
```

This routine is called to test array overflow.

Routine DMPRT0

```
(IWRITE, IDMTST, NDIMAX)
```

This routine writes out information on the dimensions used.

Routine DMSET0

```
(IWRITE, IDMTST, NDIMAX)
```

This routine initialises the arrays (IDMTST, maxima set, and NDIMAX, maxima used) that check on dimensions. Dimensions are set by the file `darc.inc` which is included in the relevant routines. All dimension variables beginning with MX are set in this file. To change a dimension you need only change `darc.inc`.

Routine MCDF09

```
(IWRITE, ITAPE3, N, DR, ZZ, NW, NH, NP, NAK, IMCDF, UCF, CP,
CQ, IDMTST, NDIMAX)
```

This routine reads the GRASP0 MCDF dump.

Routine MCDF10

```
(IWRITE, ITAPE3, N, DR, ZZ, NW, NH, NP, NAK, IMCDF, UCF, CP,
CQ, IDMTST, NDIMAX)
```

This routine reads the GRASP1 MCDF dump.

Chapter 4

DSTG1/ORBS

4.1 Introduction

$Source: /home/phn/DARC/RCS/dstg1orb.mak,v
$Author: phn
$Date: 2001/10/31 12:05:15
$Revision: 11.1

DARC, the Dirac Atomic *R*-matrix Codes.
homepage: http://www.am.qub.ac.uk/DARC/
DSTG1/ORBS module

DSTG1/ORBS evaluates the continuum orbitals.

Calling tree:

```
ORBS
    AAORBS

        QUARTZ  timing routine
        CALEN   access date/time at execution
        DMSET1  set dimensions

        RDORBS  read input data

        DA1     open/access DA files

        STP0    read target data, set up potentials
        WDISC4  writes 1st part of DSTG1/ORBS dump (ORBS.DAT)

        RDISC1  read information from ORBITALS.DAT

        BASORB  generate continuum orbitals, Buttle correction

        DMPRT1  final print of dimensions used
```

In outline:

1. DSTG1/ORBS reads in bound orbitals from `TARGET.INP`.
2. Using the bound orbitals DSTG1/ORBS will generate an *R*-matrix boundary and an appropriate radial mesh. All orbitals (continuum and bound) are tabulated on this radial mesh. You can optionally read in the *R*-matrix boundary and/or the radial mesh.
3. The continuum orbitals are generated using a target potential. This is usually the static potential. DSTG1/ORBS will use the bound orbitals to generate such a potential or it can be optionally read in.

4. Routine BASORB generates the continuum orbitals.

   This produces a set of energies and boundary amplitudes for each angular momentum. These are stored on the output file `ORBS.DAT`. The tabulated wave-function is stored on disc for use later in evaluating radial integrals.

   The numerical solution of the differential equations uses a 4th order predictor-corrector method (see routine DFBSFN).

   The Buttle corrections are also calculated at this time. These compensate for the truncation in the continuum orbital expansion. The Buttle correction is a smoothly varying function of energy. For this reason it is calculated at a small number of energy points and polynomial interpolation can be used for any energy. The way that this is implemented differs from the current versions of the Breit-Pauli R-matrix codes.

4.2 Files

The files are defined internally. Messages are also written to output stream 6 using the PRINT command.

The following files are used:

stream name	description	stream number	file name	FORM
IREAD	Input data.	5	`ORBS.INP`	Formatted
IWRITE	Output.	7	`ORBS.OUT`	Formatted
—	Scratch file for input data.	9	—	Formatted
JREAD	Input bound orbitals.	10	`TARGET.INP`	Formatted
—	DA file for orbitals.	11	`ORBITALS.DAT`	Unformatted
IDISC3	Output (options 36,37,38).	13	`ORBS.SH`	Formatted
IDISC4	DSTG1/ORBS dump.	14	`ORBS.DAT`	Unformatted

DA means direct access.

4.3 Input data (`ORBS.INP`)

The input data uses Namelist format. Each Namelist has a name. If this is say ORBS then the input data starts with the string &ORBS. There must be a space before the & even if it is at the beginning of the line. The end of this Namelist is denoted by &END, with & again preceded by a space. Variables in a Namelist can be given default values. The default value is indicated in brackets after the variable name.

Input record 1

The first input data record is an 80-character title (variable IHED).

Input record 2

The following Namelist is used.

```
 DOUBLE PRECISION BSTO
 DOUBLE PRECISION HINT
 INTEGER IRX(NIRX)
 INTEGER KBMAX
 INTEGER KCMAX
 INTEGER KCMIN
 INTEGER MAXFUL(MXNK)
 INTEGER MAXNLG(MXNK)
 INTEGER MAXNQN(MXNK)
 INTEGER MINNQN(MXNK)
 INTEGER NELC
 INTEGER NIX
 INTEGER NRANG2
 INTEGER NZ
 INTEGER OPT(50)
 DOUBLE PRECISION RA
 NAMELIST / ORBS /BSTO, HINT, IRX, KBMAX, KCMAX, KCMIN,
+ MAXFUL, MAXNLG, MAXNQN, MINNQN, NELC, NIX, NRANG2, NZ, OPT, RA
```

- BSTO [0]
is the constant used in the boundary condition at RA for the continuum orbitals. Results (cross-sections) should be insensitive to BSTO.

- HINT [0]
is the basic stepsize in the regular mesh. If it is positive then the regular mesh is read as input data (i.e. you must specify NIX and array IRX).

- IRX [−1] — array
defines the radial index at the end of the subintervals in the regular mesh. They must be even numbers if Simpson's rule is used and multiples of four if Bode's rule is used. See the description of option 44.

- KBMAX [−1]
is the largest bound angular momentum K-value to be included.

- KCMAX [0]
is the largest continuum angular momentum K-value to be included.

- KCMIN [1]
is the smallest continuum angular momentum K-value to be included.

- MAXFUL [−1] — array
This array marks the bound orbitals that correspond to full subshells in the problem. Default: no full subshells.

- MAXNLG [−1] — array
For the Kth κ-value bound orbitals with principal quantum number lying between MINNQN(K) and MAXNLG(K) are to be orthogonalised to the continuum orbitals using Lagrange multipliers. Any bound orbitals with principal quantum number

greater than MAXNLG(K) are to be Schmidt orthogonalised to the continuum orbitals.
Default: same as MAXNQN.

- MAXNQN [−1] — array
  MAXNQN(K) contains the maximum principal quantum number for the bound orbitals with K-value K.

- MINNQN [−1] — array
  MINNQN(K) contains the minimum principal quantum number for the bound orbitals with K-value K. If l is the orbital angular momentum corresponding to K then we would set MINNQN(K) to $l + 1$ if all orbitals are to be included.
  Default: all orbitals are included.

- NELC [−1]
  is the number of electrons in the target.

- NIX [−1]
  is the number of subintervals in the regular mesh with stepsize doubling in successive intervals.

- NRANG2 [20]
  is the number of continuum orbitals to be evaluated for each continuum κ-value.

- NZ [−1]
  is the atomic number of the target.

- OPT [0] – array
  Input a list of integers giving the options to be set. An option is a switch that is either on or off. To set option J input J on this input record. The options are stored in array ITC. If option J is set then ITC(J)=1; otherwise ITC(J)=0.

  Operational options.

  - 31
    This option allows the stepsize of the automatic mesh to be doubled (more coarse mesh).
  - 32
    This option allows the stepsize of the automatic mesh to be halved (more fine mesh).
  - 36
    When this option is set the bound orbitals are written to file `ORBS.SH`.
  - 37
    When this option is set the static, nuclear and core potentials are written to file `ORBS.SH`.
  - 38
    When this option is set the continuum orbitals are written to file `ORBS.SH`.

  - 40
    When this option is set the static potential is read in from file `TARGET.INP`.
  - 42
    When this option is set the core potential is read in from file `TARGET.INP`.
  - 43
    This option allows you to replace the orbital occupation numbers that have been read in from file `TARGET.INP`.
  - 44
    When this option is set the radial integration weights set in routine SETW correspond to Bode's rule. This requires that the elements of array IRX should be multiples of four. If this is not the case then the weights revert to the default which is Simpson's rule.

  Print options.

  - 1
    Debug print from routine FINDER.
  - 2
    Debug print from routine DFBSFN of the mixing coefficients for solutions from routine DHPCG.
  - 3
    Debug print from routine DFBSFN of the solutions from routine DHPCG. Note option 3 can produce large amounts of output.
  - 4
    Debug print from routine DFBSFN of the final solution (combination of solutions from routine DHPCG). Note option 4 can produce large amounts of output.
  - 5
    Print orbital overlap integrals in routine BASORB.
  - 6
    Print Schmidt coefficients in routine SCHMDT.
  - 8
    Print Buttle corrections in routine BASORB.
  - 15
    Print multipole radial integrals as they are evaluated in routine INTRD.

- RA [0]
  is the R-matrix boundary. If it is zero then a boundary will be determined in routine STP0.

Input record 3

When option 43 is set read in the array UCF. This overwrites the array UCF read from file `TARGET.INP`.

Read the following input records using FORTRAN free-format.

```
A. (UCF(J),J=1,NW)
```

Input data from file `TARGET.INP`

Read in the target orbitals

Read the following input records using FORTRAN free-format. The input records D to G are repeated for J=1,NW.

```
A. NW,N
B. (UCF(J),J=1,NW)
C. (DR(I),I=1,N)
D. NP(J),NAK(J)
E. (CP(I),I=1,N)
F. NP(J),NAK(J)
G. (CQ(I),I=1,N)
```

These variables are described in the section on the DSTG0 module.

The following data, if required, should be added to the file `TARGET.INP` generated by module DSTG0. The potentials are defined at the radial mesh given by array DR. The form of the potentials is described in a later section.

When option 40 is set then read in the static potential.

Read the following input records using FORTRAN free-format.

```
A. N
B. (DR(I),I=1,N)
C. (ZSTAT(I),I=1,N)
```

When option 42 is set then read in the core potential.

Read the following input records using FORTRAN free-format.

```
A. N
B. (DR(I),I=1,N)
C. (ZCORE(I),I=1,N)
```

4.4 Routines

Routine ORBS

This is the PROGRAM routine. The input and output files are set up: IREAD (`ORBS.INP`) and IWRITE (`ORBS.OUT`). A call is made to routine AAORBS which controls the computation.

Routine AAORBS

(IREAD, IWRITE)

This is the main controlling routine for the DSTG1/ORBS module.

Routine BASORB

(BSTO, CL, CP, CQ, DR1, HINT, IDISC3, IDISC4, IDMTST, IPOS, IREC1, IRX,ITC, IWRITE, JAG, KAG, KBMAX, KCMAX, KCMIN, LAG, MAXNLG, MAXNQN, NDIMAX, NELC, NPTS, NRANG2, NUMORB, NZ, RA, RMESH, WEIGHT, ZSTAT)

Calling tree:

```
BASORB
    DA1
    DFBSFN
        DGETRF
        DGETRS
        DHPCG
            FCT
                BZVALU
            OUTP
    DMCHK1
        DMPRT1
    FINDER
        DFBSFN (see above)
        ROOT
    IN2CH3
    INTRD
    MATOUT
    QUARTZ
    SCHMDT
        INTRD
        MATOUT
```

This routine controls the evaluation of the continuum orbitals. There is a loop over the required K-values from KCMIN to KCMAX. Within this loop there is an inner loop over the required number of continuum orbitals, NRANG2.

- The continuum orbitals are obtained from a call to routine FINDER which iterates the eigenvalue until a solution with the required boundary conditions is achieved.
- A call is made to routine SCHMDT in an attempt to Schmidt orthogonalise any bound orbitals that have not been included in the differential equations through Lagrange multipliers.
- The large and small radial functions for each continuum orbital are written to the DA file `ORBITALS.DAT`. See routine SKIPER for the format of this file.
- If option 38 is set then the continuum orbitals are written out to the formatted file `ORBS.SH`. This file takes the form of a ksh or bash script that can be executed to

produce an index file (called INDEX1) and one file for each orbital (called basxxx, where xxx is an integer). The index file gives information on the orbital files. Each orbital file has 4 columns, col.1 is the radial coordinate, col.2 is the large component (P), col.3 is the small component (Q), col.4 is the density ($P^2 + Q^2$). A perl script draw-wave.phn is available which will process the index and orbital files and produce a GNUPLOT script. This allows easy viewing of the orbital functions.

- The eigenvalues (array EIGENS), in au, and amplitudes (array ENDS), large component at $r = \mathrm{RA}$, of the continuum orbitals are written to file ORBS.DAT.

- This routine also calculates Buttle corrections. These are stored in the arrays EBUT (energies in au) and CBUT (corrections), and are written to file ORBS.DAT. If option 8 is set then the Buttle corrections are printed.

  For a given κ value, there are n_{bas} (variable NRANG2) continuum orbitals with amplitudes $P_i(a)$ and eigenvalues E_i, $(i = 1 \ldots n_{bas})$. The Buttle energy is given by E_{but} (array EBUT). The continuum orbital differential equations are solved at this energy, by a call to routine DFBSFN. The resulting solutions are (P, Q). The Buttle correction (array CBUT) at this energy is given by:

$$R^{but}(\kappa, E_{but}) = \left[\frac{1}{\frac{2caQ(a)}{P(a)} - b - \kappa} \right] - \frac{1}{2\,a} \sum_{i=1}^{n_{bas}} \frac{P_i(a)\, P_i(a)}{E_i - E_{but}} \tag{4.1}$$

  where a (variable RA) is the R-matrix boundary and b (variable BSTO) is a constant. All energies are in au.

  We choose the Buttle energies so as to avoid the eigenvalues E_i. If this occurred then the R-matrix boundary condition would be satisfied and the denominator of the first term on the rhs would be zero. Also one of the denominators in the summation of the second term could be zero.

  The Buttle correction is a smoothly varying function of the energy. Two energies in each successive interval between the eigenvalues are chosen for the Buttle energies. The Buttle correction is interpolated for the energy required.

- Overlap integrals are calculated between all orbitals with the same K-value. The orbitals should form an orthonormal set. Any discrepancy larger than variable SMALL (set to 10^{-3}) gives a warning message. If option 5 is set then the overlap integrals (less 1 on the diagonal) are printed.

  Failure to achieve orthonormality is an indication of either

  - a poor choice of radial mesh, or
  - failure of Schmidt orthogonalisation, or
  - poor choice of bound orbitals.

  The routine prints information on the number of radial mesh points used between nodes of the continuum orbitals. This should be checked.

Routine BZVALU

`(X, IX, ID, RES1, RES2, IFLAG, CP, CQ, NORI, NPTS, RMESH, ZSTAT)`

This routine uses cubic Lagrange interpolation to determine values of either the wavefunctions or static potential that are off the supplied mesh. It is called by routine FCT which evaluates the rhs of the differential equations for the continuum orbitals.

Routine DFBSFN

`(NBT, NODE, AKAP, FIRSTP, FIRSTQ, WR, BVALUE, CL, CP, CQ,`
`DR1, HINT, IRX, ITC, IWRITE, NORI, NPTS, ORBP, ORBQ, RA, RMESH,`
`WEIGHT, ZSTAT)`

This routine controls the solution of a pair of radial Dirac equations which satisfy a specified boundary condition. The equations can contain Lagrange multipliers.

$$\begin{aligned}\frac{dP}{dr} &= -\frac{\kappa}{r}P + \frac{1}{c}\left[2c^2 + E + \frac{Z_s}{r}\right]Q - \frac{1}{c}\sum_j \lambda_j\,\bar{Q}_j \qquad (4.2)\\ \frac{dQ}{dr} &= \frac{\kappa}{r}Q - \frac{1}{c}\left[E + \frac{Z_s}{r}\right]P + \frac{1}{c}\sum_j \lambda_j\,\bar{P}_j\end{aligned}$$

where c is the speed of light in au, E is the energy eigenvalue (less the rest energy) in au, κ is the angular momentum quantum number and Z_s is the static potential.

Also $(\bar{P}_j, \bar{Q}_j)$ are bound orbitals. The Lagrange multipliers λ_j are included to ensure orthogonality between (P, Q) and the bound orbitals. This routine will determine the appropriate Lagrange multipliers.

The boundary conditions are

$$P(0) = 0 \qquad \text{and} \qquad \frac{2caQ(a)}{P(a)} = b + \kappa \qquad (4.3)$$

where a (variable RA) is the R-matrix boundary and b (variable BSTO) is a constant.

The energy eigenvalue is held constant in this routine. Of course in general the solution will not satisfy the boundary condition at the R-matrix boundary. The variation of the eigenvalue is carried out in routine FINDER.

The equations are solved by a call to routine DHPCG.

Set option 2 for a print of the mixing coefficients for solutions from routine DHPCG.
Set option 3 for a print of the solutions from routine DHPCG.
Set option 4 for a print of the final solution (combination of solutions from routine DHPCG).
Note options 3 and 4 can produce large amounts of output.

TODO: more detail on how λ are determined

Routine DHPCG

`(DR1, HINT, IRX, RA, Y, NBT, AKAP, CONST1, CONST2, CP, CQ, NORI,`
`NPTS, RCL, RMESH, ZSTAT, UP, UQ)`

This routine solves the set of Dirac first order equations. The Runge-Kutta method provides starting values for Hamming's modified predictor-corrector method. The integration is carried out from the initial point DR1 to the final point RA i.e. there is no inward integration as often occurs in bound state codes.

The rhs of the equations are obtained in routine FCT.

The solution is passed to routine OUTP which returns the arrays UP and UQ, the eventual output from this routine

TODO: formulae

Routine DMCHK1

`(I, ID, IWRITE, IDMTST, NDIMAX)`

This routine is called to test array overflow.

Routine DMPRT1

`(IWRITE, IDMTST, NDIMAX)`

This routine writes out information on the dimensions used.

Routine DMSET1

`(IWRITE, IDMTST, NDIMAX)`

This routine initialises the arrays (IDMTST, maxima set, and NDIMAX, maxima used) that check on dimensions. Dimensions are set by the file `darc.inc` which is included in the relevant routines. All dimension variables beginning with MX are set in this file. To change a dimension you need only change `darc.inc`.

Routine DQSF

`(K, X, Y, ASY, RMESH, DR1, HINT, IRX, NIX, NPTS)`

This routine evaluates the $\frac{1}{r}Y^k(X;r)$ potentials that are used for Slater integrals and for potentials.

$$\frac{1}{r}\,Y^k(X;r) = \int_0^{\mathrm{RA}} X(s)\,U^k(r,s)\,ds \tag{4.4}$$

where

$$U^k(r,s) = \begin{cases} \frac{s^k}{r^{k+1}} & \text{when } s < r \\ \frac{r^k}{s^{k+1}} & \text{when } r < s \\ \frac{1}{r} & \text{when } r = s \end{cases} \tag{4.5}$$

Split the integrand into two parts i.e.

$$\frac{1}{r}\,Y^k(X;r) = \frac{1}{r^{k+1}}\int_0^r X(s)\,s^k\,ds + r^k\int_r^{\mathrm{RA}} X(s)\,\frac{1}{s^{k+1}}\,ds \tag{4.6}$$

These integrals are evaluated using Simpson's rule.

TODO: more details on integration and corrections

Routine FCT

`(X, IX, FUN, DFUN, NBT, AKAP, CONST1, CONST2, CP, CQ, NORI, NPTS, RCL, RMESH, ZSTAT)`

This routine evaluates the rhs of the differential equations for the continuum orbitals.

$$\begin{aligned} \frac{dP_i}{dr} &= -\frac{\kappa}{r}P_i + \frac{1}{c}\left[2c^2 + E + \frac{Z_s}{r}\right]Q_i - \frac{1}{c}\bar{Q}_i \\ \frac{dQ_i}{dr} &= \frac{\kappa}{r}Q_i - \frac{1}{c}\left[E + \frac{Z_s}{r}\right]P_i + \frac{1}{c}\bar{P}_i \end{aligned} \tag{4.7}$$

where c is the speed of light in au, E is the energy eigenvalue (less the rest energy) in au, κ is the angular momentum quantum number and Z_s is the static potential.

Also $(\bar{P}_i, \bar{Q}_i)$ is a bound orbital. This allows the inclusion of Lagrange multipliers in the equations. If there are NBT bound orbitals (or equivalently Lagrange multipliers) then there will be NBT+1 pairs of equations as above. One pair for each of the NBT bound orbitals. The final pair of equations has no bound orbitals on the rhs.

The rhs of the equations may have to be evaluated at a radial point which is not on the regular mesh. In this case routine BZVALU is called to perform interpolation of the orbital functions and potentials.

Routine FINDER

`(NBT, AKAP, FIRSTP, FIRSTQ, BSTX, ETRIAL, NODES, CL, CP, CQ, DR1, HINT, ORBP, ORBQ, RA, RMESH, WEIGHT, ZSTAT, IRX, ITC, IWRITE, NORI, NPTS)`

This routine iterates the energy eigenvalue and attempts to find a solution to the continuum orbital differential equations that satisfies the boundary conditions,

$$P(0) = 0 \qquad \text{and} \qquad \frac{2caQ(a)}{P(a)} = b + \kappa \tag{4.8}$$

where a (variable RA) is the R-matrix boundary and b (variable BSTO) is a constant. The solution must have the required number of nodes.

Next an energy interval is identified that contains the eigenvalue. Routine ROOT is called to converge to the required value.

Option 1 gives a print out of the iterations. The variable printed out is BVALUE−BSTX which should tend to zero.

Routine INTRD

`(N1, K1, N2, K2, KK, RESULT, CP, CQ, WEIGHT, RMESH, IWRITE, ITC, NPTS, IPOS)`

This routine evaluates the following radial integral over the finite interval $[0, \mathrm{RA}]$. See routine SETW for a description of the numerics.

$$I_L^k(1,2) = \int_0^{\mathrm{RA}} \left[P_1P_2 + Q_1Q_2\right]\, r^k\, dr \tag{4.9}$$

where P and Q are the large and small radial components. When $k = 1$ this integral occurs in the length form of the dipole matrix element. Option 15 gives a debug print out.

Routine OUTP

`(UP, UQ, IRAD, Y, NBTP1)`

This routine is called by DHPCG and stores the computed solutions in the arrays UP and UQ.

Routine RDISC1

```
(IDMTST, IWRITE, KBMAX, LAG, MAXNQN, NDIMAX, NPTS,
CP, CQ, IPOS, RMESH, WEIGHT, ZCORE, ZNUCL, ZSTAT)
```

This routine reads the bound orbitals from `ORBITALS.DAT` and stores them in the CP/CQ arrays.

The array IPOS points to the position of the bound orbitals in the CP/CQ array. For example for an orbital with N-value N1 and K-value K1 then define J1=NPTS∗(IPOS(N1,K1)−1). The large component of this orbital is CP(J) for J=J1+1,J1+NPTS and similarly for the small component in the CQ array.

It also reads the arrays RMESH, WEIGHT, ZSTAT, ZNUCL and ZCORE.

Routine RDORBS

```
(IDMTST, IREAD, IWRITE, NDIMAX, RECORD, BSTO, CL,
DR1, HINT, IHED, IRX, ITC, JAG, KAG, KBMAX, KCMAX,
KCMIN, KX, LAB, LAG, MAXFUL, MAXNLG, MAXNQN,
MINNQN, NELC, NIX, NPTS, NRANG2, NZ, RA)
```

This routine reads in the input data from file `ORBS.INP`. The speed of light c (variable CL) is set to 137.0359895D0 in au.

Routine ROOT

`(T, FT, B, C, RELERR, ABSERR, IFLAG)`

This routine computes a root of the nonlinear equation $F(x) = 0$, where $F(x)$ is a continuous real function of a single real variable x. The method of solution is a combination of bisection and the secant rule.

The routine is used by routine FINDER in the iterations that search for the continuum orbital eigenvalue. It is taken from the book by Shampine and Gordon (1975 [60]). See the routine listing for a fuller description.

Routine SCHMDT

```
(IWRITE, KC, CP, CQ, IPOS, ITC, MAXNLG, MAXNQN, NPTS,
NRANG2, RMESH, WEIGHT)
```

This routine attempts to Schmidt orthogonalise the continuum orbitals to the bound orbitals that are not included in the Lagrange orthogonalisation in DFBSFN for the K-value KC.

TODO: more details on theory

Routine SETW

`(DR1, HINT, IRX, ITC, IWRITE, NIX, NPTS, RMESH, WEIGHT)`

This routine:

- sets up the RMESH array which stores the regular mesh
- calculates array WEIGHT that contains the integration weights for either Simpson's rule (option 44 unset, the default) or Bode's rule (option 44 set).

The regular mesh has a basic stepsize HINT. The range is divided into NIX subintervals. Within each subinterval the stepsize is held constant. The stepsize in successive subintervals is doubled. In subinterval 1 the radial mesh has stepsize HINT. The radial index runs from 1 to IRX(1) and the first radial point is HINT i.e. RMESH(1)=HINT, RMESH(2)=2∗HINT, . . . RMESH(IRX(1))=IRX(1)∗HINT. In subinterval 2 the stepsize is doubled to 2∗HINT. The radial index runs from IRX(1)+1 to IRX(2). And so on . . .

The integration weights are used in the evaluation of all radial integrals.

A radial integral is evaluated as follows:

$$\int_0^{\mathrm{RA}} I(r)\, dr \approx \sum_{i=1}^{\mathrm{NPTS}} w_i\, I_i \tag{4.10}$$

The integrand $I(r)$ is tabulated on the regular mesh as I_i. The corresponding weights are w_i. It is assumed that $I(0) = 0$ so that the origin is not included in the summation.

There are the following constraints on the regular mesh.

- There must be more than 4 points in each subinterval. This is required by routine DDET5 which evaluates the derivative of a function tabulated on the regular mesh.
- For Simpson's rule the endpoints of subintervals (i.e. IRX(I)) must be even.
- For Bode's rule the endpoints of subintervals (i.e. IRX(I)) must be multiples of 4. If this is not satisfied then the code will try to use Simpson's rule.

Routine STP0

```
(IDISC3, IREC1, CP, CQ, DR1, HINT, IDMTST, IREAD, IRX,
ITC, IWRITE, JREAD, KAG, KBMAX, LAG, MAXNQN, NDIMAX, NELC, NIX,
NPTS, NUMORB, NZ, RA, RMESH, WEIGHT, ZCORE, ZNUCL, ZSTAT)
```

Calling tree:

```
STP0
    DA1
    DMCHK1
        DMPRT1
    DQSF
    IN2CH3
    SETW
    XFITPL
```

This routine determines the regular mesh, bound orbitals and potentials. In more detail:

- The generalised orbital occupation numbers (array UCF) are read in from the formatted file `TARGET.INP`. These are used to generate a static potential.

- The bound orbitals are read in from `TARGET.INP`. This file can be created by DARC module DSTG0.

  The arrays CP/CQ are used for storage of these orbitals. This is temporary storage just for this routine. The arrays CP/CQ are in the argument list so as to save memory, not to pass the results.

- The nuclear potential is set to point charge. It is intended to allow a finite nuclear model at some future time.

- If option 43 is set then the array UCF is replaced by input data read from file `ORBS.INP`.

- A check is made on the bound orbitals and array UCF. For example the input data in `ORBS.INP` specifies the bound orbitals to be used and there must be sufficient orbitals in `TARGET.INP`. Also the sum of the array UCF must equal the number of electrons in the target (variable NELC).

- The regular mesh is set up.

  The R-matrix boundary and regular mesh can be read in from `ORBS.INP`.

  Otherwise the regular mesh is generated automatically.

  The first step is to determine the R-matrix boundary. This can be specified in the input data in `ORBS.INP`. Otherwise it is chosen so that the magnitude of the large component of all bound orbitals are decreasing and less than variable PMAX, set to 10^{-3}, at the boundary.

  The second step is to generate a sequence of subintervals. The philosophy adopted here is to try to match the logarithmic radial mesh used in GRASP. The basic stepsize is small (HINT=1.953125D−5). This gives a good representation of the orbitals near the nucleus. Generally a greater number of subintervals will be used than in the nonrelativistic R-matrix program RMATRX1.

  If option 31 is set then the generated basic stepsize is doubled giving a coarser mesh with fewer points.

  If option 32 is set then the generated basic stepsize is halved giving a finer mesh with more points.

- The regular mesh and integration weights are obtained from a call to routine SETW. By setting option 44, Bode's weights are used rather than the default Simpson's weights.

- The regular mesh array RMESH and the radial weights array WEIGHT are written to DA file `ORBITALS.DAT`. This is the first information written to the file.

- The nuclear potential is interpolated to the regular mesh.

- The bound orbitals are interpolated to the regular mesh and written to `ORBITALS.DAT`. See routine SKIPER for the format of this file.

  Important information (normalisation, $\langle 1/r\rangle$, $\langle r\rangle$, $\langle r^2\rangle$) on the bound orbitals is calculated and written out. This should be checked for consistency with e.g. GRASP if that program is the origin of the orbitals.

  If option 36 is set then the bound orbitals are written out to the formatted file `ORBS.SH`. This file takes the form of a ksh or bash script that can be executed to produce an index file (called `INDEX1`) and one file for each orbital (called `orbxxx`, where `xxx` is an integer). The index file gives information on the orbital files. Each orbital file has 4 columns, col.1 is the radial coordinate, col.2 is the large component (P), col.3 is the small component (Q), col.4 is the density ($P^2 + Q^2$). A perl script, `draw-wave.phn`, is available which will process the index and orbital files and produce a GNUPLOT script. This allows easy viewing of the orbital functions.

- If option 40 is set then the static potential is read in from `TARGET.INP` and interpolated to the regular mesh.

  Otherwise the default procedure is to generate the static potential $Z_s(r)$ using the supplied bound orbitals and generalised occupation numbers (array UCF).

$$Z_s(r) = Z_n(r) - \sum_j q(j)\, Y^0(j,j;r) \tag{4.11}$$

  where Z_n is the nuclear potential ($Z_n = Z$, the atomic number, for a point nucleus), $q(j)$ is the generalised occupation number of orbital j ($\sum_j q(j) = N$, the number of target electrons) and the Slater potential is

$$\frac{1}{r}Y^0(j,j;r) = \int_0^{\mathrm{RA}} \left[P_j(s)P_j(s) + Q_j(s)Q_j(s)\right] U^0(r,s)\, ds \tag{4.12}$$

  with

$$U^0(r,s) = \begin{cases} \frac{1}{r} & \text{when } s \le r \\ \frac{1}{s} & \text{when } r < s \end{cases} \tag{4.13}$$

  The behaviour of the static potential is

$$Z_s(r) = \begin{cases} Z & \text{when } r \longrightarrow 0 \\ Z - N & \text{when } r \longrightarrow \mathrm{RA} \end{cases} \tag{4.14}$$

  assuming that RA encloses the bound orbitals.

- If option 42 is set then the core potential is read in from `TARGET.INP` and interpolated to the regular mesh.

  Otherwise the default procedure is to set the core potential to zero.

- The static potential (array ZSTAT), the nuclear potential (array ZNUCL) and the core potential (array ZCORE) are written to `ORBITALS.DAT`.

- If option 37 is set then the static potential (array ZSTAT), the nuclear potential (array ZNUCL) and the core potential (array ZCORE) are written to `ORBS.SH`. The index file in called `INDEX2` and the file names are `stapot`, `nucpot` and `corpot` corresponding to static, nuclear and core respectively. Each of these `pot` files has two columns, col.1 is the radial coordinate, col.2 is the potential. A perl script, `draw-pot.phn`, is available which will process the index and potential files and produce a GNUPLOT script. This allows easy viewing of the potential functions.

Routine WDISC4

```
(IDISC4, BSTO, CL, DR1, HINT, IHED, IRX, IWRITE, JAG, KAG,
KBMAX, KCMAX, KCMIN, KX, LAG, MAXFUL, MAXNLG, MAXNQN, MINNQN,
NELC, NIX, NRANG2, NUMORB, NZ, RA, RECORD)
```

This routine writes the first part of the DSTG1/ORBS dump file, `ORBS.DAT`. It consists of the basic data only. The continuum orbital eigenvalues and amplitudes as well as the Buttle corrections are written in routine BASORB.

Routine XFITPL

```
(FA, FB, ICON, RA, RB, MAXA, MAXB, IWRITE)
```

This routine is used to interpolate bound orbitals and potentials from some input radial mesh to another radial mesh. The routine uses cubic Lagrangian interpolation. For example GRASP bound orbitals in the file `TARGET.INP` will be tabulated on a logarithmic radial mesh. However DARC uses a regular radial mesh (see routine SETW).

A parameter ICON can be set to 1 to indicate that the function is a bound orbital that can be completed with zeros for large radii if necessary.

Chapter 5

DSTG1/INTS

5.1 Introduction

$Source: /home/phn/DARC/RCS/dstg1int.mak,v
$Author: phn
$Date: 2001/10/31 12:05:15
$Revision: 11.1

DARC, the Dirac Atomic *R*-matrix Codes.
homepage: http://www.am.qub.ac.uk/DARC/
DSTG1/INTS module

DSTG1/INTS evaluates the radial integrals.

Calling tree:

```
INTS
    AAINTS

        QUARTZ  timing routine
        CALEN   access date/time at execution
        DMSET1  set dimensions

        RDINTS  read input data

        DA1     open/access DA files

        RDISC4  read information from ORBS.DAT,
                write to DSTG1.DAT
        RDISC1  read information from ORBITALS.DAT

        GEN     calculate radial integrals

        DMPRT1  final print of dimensions used
```

Routine GEN controls the calculation of the radial integrals. These have the following types:

- **Multipole**
  These are one-electron integrals which have 3 kinds: bound-bound, bound-continuum and continuum-continuum.

  The bound-bound integrals are needed for the calculation of the long-range scattering potentials. These have the form

  $$\frac{a_{ij}^{\lambda}}{r^{\lambda+1}} \tag{5.1}$$

  where $\lambda = 1, 2, 3 \ldots$ and i, j label the scattering channels. The potentials determine the solution in the external ($r > a$) region.

All three types are required in the calculation of photo-ionisation matrix elements. In this case there is a further subdivision of radial integral into length and velocity form. Only dipole matrix elements are required for photo-ionisation so that λ is one.

- **Continuum-continuum one-electron and Slater two-electron**
  These are used in the evaluation of the continuum-continuum part of the continuum ($(N+1)$-electron) Hamiltonian.
- **Bound-continuum one-electron and Slater two-electron**
  These are used in the evaluation of the bound-continuum part of the continuum ($(N+1)$-electron) Hamiltonian.
- **Bound-bound one-electron and Slater two-electron**
  These are used in the evaluation of the target Hamiltonian and the bound-bound part of the continuum ($(N+1)$-electron) Hamiltonian.

5.2 Files

The files are defined internally. Messages are also written to output stream 6 using the PRINT command.

The following files are used:

stream name	description	stream number	file name	FORM
IREAD	Input data.	5	`INTS.INP`	Formatted
IWRITE	Output.	7	`INTS.OUT`	Formatted
—	Scratch file for input data.	9	—	Formatted
—	DA file for orbitals.	11	`ORBITALS.DAT`	Unformatted
—	DA file for radial integrals.	12	`INTEGRAL.DAT`	Unformatted
IDISC4	DSTG1/ORBS dump.	14	`ORBS.DAT`	Unformatted
IDISC5	DSTG1 dump.	15	`DSTG1.DAT`	Unformatted

DA means direct access.

5.3 Input data (`INTS.INP`)

The input data uses Namelist format. Each Namelist has a name. If this is say INTS then the input data starts with the string &INTS. There must be a space before the & even if it is at the beginning of the line. The end of this Namelist is denoted by &END, with & again preceded by a space. Variables in a Namelist can be given default values. The default value is indicated in brackets after the variable name.

Input record 1

The following Namelist is used.

```
INTEGER JTARG
INTEGER LAMBB
INTEGER LAMBC
INTEGER LAMCC
INTEGER OPT(50)
NAMELIST / INTS / JTARG, LAMBB, LAMBC, LAMCC, OPT
```

- JTARG [-1]
  is the maximum $2J$-value for the target states to be used later in the calculation i.e. in DSTG2. This attempts to restrict the number of radial integrals that are calculated.

- LAMBB [4]
  is the maximum λ value allowed for bound-bound multipole integrals. LAMBB must lie between 1 and 10 inclusive.

- LAMBC [1]
  is the maximum λ value allowed for bound-continuum multipole integrals. For the dipole matrix elements of photo-ionisation the value should be 1.

- LAMCC [1]
  is the maximum λ value allowed for continuum-continuum multipole integrals. For the dipole matrix elements of photo-ionisation the value should be 1.

- OPT [0] – array
  Input a list of integers giving the options to be set. An option is a switch that is either on or off. To set option J input J on this input record. The options are stored in array ITC. If option J is set then ITC(J)=1; otherwise ITC(J)=0.

  Print options.

  - 15
    Print multipole radial integrals as they are evaluated in routines INTRD and INTRDV.
  - 16
    Print one-electron radial integrals as they are evaluated in routine INTRD1.
  - 17
    Print two-electron radial integrals as they are evaluated in routine INTRD2.
  - 27
    This option gives a debug print from routine GENCC, that evaluates continuum-continuum radial integrals.

5.4 DSTG1 dump file (DSTG1.DAT)

This file is unformatted.

Repeat records 9 to 13 for KC = KCMIN,KCMAX

```
 1. IHED,RECORD
 2. KBMAX,KCMIN,KCMAX,KX,NRANG2,NELC,NZ
 3. LAMBB,LAMBC,LAMCC
 4. (MINNQN(L),L=1,KX), (MAXNQN(L),L=1,KX),
    (MAXNLG(L),L=1,KX), (MAXFUL(L),L=1,KBMAX),
    (LAG(L),L=1,KX), (KAG(L),L=1,KX), (JAG(L),L=1,KX)
 5. RA,BSTO,HINT,CL,DR1
 6. NIX
 7. (IRX(I),I=1,NIX)
 8. NUMORB

 9. (EIGENS(KC,N),N=1,NRANG2)
10. (ENDS(KC,N),N=1,NRANG2)
11. MBUT(KC)
12. (EBUT(KC,I),I=1,MBUT(KC))
13. (CBUT(KC,I),I=1,MBUT(KC))
```

where

- IHED
  is the 80-character title for the calculation.

- RECORD
  is a 20-character record of time/date at execution. It is obtained by a call to the system dependent routine CALEN.

- KBMAX
  is the maximum bound K-value.

- KCMIN
  is the minimum continuum K-value.

- KCMAX
  is the maximum continuum K-value.

- KX
  is the larger of KBMAX and KCMAX (usually same as KCMAX).

- NRANG2
  is the number of continuum orbitals for each K-value.

- NELC
  is the number of electrons in the target.

- NZ
  is the atomic number of the target.

- LAMBB
  is the maximum λ value allowed for bound-bound multipole integrals.

- LAMBC
  is the maximum λ value allowed for bound-continuum multipole integrals.

- LAMCC
  is the maximum λ value allowed for continuum-continuum multipole integrals.

- MINNQN(K) K=1,KX
  This array gives the minimum principal quantum numbers for the bound orbitals. For K-value K, MINNQN(K) is the minimum principal quantum number. If MINNQN(K) is larger than $l + 1$ then this indicates that a core potential may be being used. This is only used in the evaluation of radial integrals.

- MAXNQN(K) K=1,KX
  This array gives the maximum principal quantum numbers for the bound orbitals. For K-value K, MAXNQN(K) is the maximum principal quantum number.

- MAXNLG(K) K=1,KX
  This array marks the bound orbitals to be Schmidt orthogonalised. For K-value K, Lagrange multipliers are included for bound orbitals with principal quantum number $N \leq$ MAXNLG(K). Orbitals with principal quantum number greater than MAXNLG(K) up to and including MAXNQN(K) are Schmidt orthogonalised.

- MAXFUL(K) K=1,KBMAX
  This array marks the bound orbitals that correspond to full subshells. For K-value K, MAXFUL(K) is the maximum principal quantum number for subshells that are always full. This means that subshells with principal quantum number $N \leq$ MAXFUL(K) are full in all target configurations. This is used to restrict the number of radial integrals that are calculated in module DSTG1/INTS.

- LAG(K) K=1,KX
  is an array of orbital l-values. K is the K-value.

- KAG(K) K=1,KX
  is an array of orbital κ-values. K is the K-value.

- JAG(K) K=1,KX
  is an array of orbital $2j$-values. K is the K-value.

- RA
  is the R-matrix boundary radius.

- BSTO
  is the constant in the boundary condition at RA for the continuum orbitals (corresponds to the nonrelativistic logarithmic derivative)

  $$\frac{2caQ(a)}{P(a)} = b + \kappa \tag{5.2}$$

  where a (variable RA) is the R-matrix boundary and b (variable BSTO) is a constant.

- HINT
  is the basic stepsize for the regular radial mesh.

- CL
  is the speed of light in au. It is set to 137.0359895D0 in routine RDORBS of module DSTG1/ORBS.

- DR1
  is the initial point (equals HINT) in the regular radial mesh.

- NIX
  is the number of subintervals in the regular radial mesh, each subinterval has a constant stepsize with stepsize doubling in successive subintervals.

- IRX(I) I=1,NIX
  defines the index at the end of the subintervals in the regular radial mesh.

- NUMORB
  is the number of bound orbitals.

- EIGENS(K,N) N=1,NRANG2 K=KCMIN,KCMAX
  is an array of energy eigenvalues for the continuum orbitals (in au), K is the K-value and N is the N-value.

- ENDS(K,N) N=1,NRANG2 K=KCMIN,KCMAX
  is an array of large component amplitudes for the continuum orbitals at the R-matrix boundary RA, $P(a)$, K is the K-value and N is the N-value.

- MBUT(K) K=KCMIN,KCMAX
  is an array giving the number of Buttle corrections stored for each continuum K-value, K.

- EBUT(K,I) I=1,MBUT(K) K=KCMIN,KCMAX
  is an array of Buttle energies in au (MBUT(K) in number) for each continuum K-value K.

- CBUT(K,I) I=1,MBUT(K) K=KCMIN,KCMAX
  is an array of Buttle corrections (MBUT(K) in number) for each continuum K-value K, tabulated as a function of energy (array EBUT).

Write the multipole integrals

```
14. IRK1,IREC
```

IRK1 is the number of radial integrals calculated. If (IRK1 $>$ 0) then write records 15 to 17 and a call is made to DA1 that outputs array RKSTO to file `INTEGRAL.DAT`.

If (LAMBB $>$ 0) then write record 15
Set LAMIND = (LAMBB+1)/2

```
15. (((IBBPOL(I,J,K),I=1,KBMAX),J=1,KBMAX),K=1,LAMIND)
```

If (KCMAX > 0 and LAMBC > 0) then write record 16
Set LAMIND = (LAMBC+1)/2

```
16. (((IBCPOL(I,J,K),I=1,KBMAX),J=1,KCMAX),K=1,LAMIND)
```

If (KCMAX > 0 and LAMCC > 0) then write record 17
Set LAMIND = (LAMCC+1)/2

```
17. (((ICCPOL(I,J,K),I=1,KCMAX),J=1,KCMAX),K=1,LAMIND)
```

Write the continuum-continuum radial integrals

Repeat records 18 and 19 for KC=KCMIN,KCMAX and KCP=KC,KCMAX

```
18. IRK1,KC,KCP,IREC
```

IRK1 is the number of radial integrals calculated. If (IRK1 > 0) then write record 19 and a call is made to DA1 that outputs array RKSTO to file `INTEGRAL.DAT`. IREC is the number of the first record on the direct access file that contains this array.

Set I1=MIN(KBMAX+KBMAX,KC+KCP)/4+1
and I2=MIN(KBMAX+KC,KBMAX+KCP)/4+1

```
19. (((ICTX(I,J,K),I=1,KBMAX),J=1,KBMAX),K=1,I1),
    (((ICTY(I,J,K),I=1,KBMAX),J=1,KBMAX),K=1,I2)
```

Write the bound-continuum radial integrals

Repeat records 20 to 21 for KC=KCMIN,KCMAX

```
20. IRK1,IRK2,KC,IREC
```

IRK1 is the number of radial integrals calculated. If (IRK1 > 0) then write record 21 and a call is made to DA1 that outputs array RKSTO to file `INTEGRAL.DAT`. IREC is the number of the first record on the direct access file that contains this array.

```
21. (((ICTX(I,J,K),I=1,KBMAX),J=1,KBMAX),K=1,KBMAX),
    (ISTX1(I),I=1,IRK2), (ISTX2(I),I=1,IRK2)
```

Write the bound-bound radial integrals

```
22. IRK1,IRK2,IREC
```

IRK1 is the number of radial integrals calculated. If (IRK1 > 0) then write record 23 and a call is made to DA1 that outputs array RKSTO to file `INTEGRAL.DAT`. IREC is the number of the first record on the direct access file that contains this array.

```
23. (ISTX(I),I=1,KBMAX),
    (((ICTX(I,J,K),I=1,KBMAX),J=1,KBMAX),K=1,KBMAX),
    (ISTX1(I),I=1,IRK2), (ISTX2(I),I=1,IRK2)
```

5.5 Orbitals dump file (`ORBITALS.DAT`)

This file contains:

1. radial mesh and integration weights
2. bound orbitals
3. potentials
4. continuum orbitals

Each array may span more than one direct access record. Routine DA1 increments variable IREC so that it is correct for the next call.

The following code indicates the order in which the data is written. The second argument of DA1, 2, corresponds to a write and the fourth argument, 11, is the stream number.

For the orbitals, PA is the large radial component while QA is the small radial component.

```
IREC = 1
CALL DA1('ORBITALS.DAT',2,IREC,11,NPTS,RMESH)
CALL DA1('ORBITALS.DAT',2,IREC,11,NPTS,WEIGHT)

DO KB = 1,KBMAX
  LB = LAG(KB) + 1
  MAXHF = MAXNQN(KB)
  DO NB = LB,MAXHF
    CALL DA1('ORBITALS.DAT',2,IREC,11,NPTS,PA)
    CALL DA1('ORBITALS.DAT',2,IREC,11,NPTS,QA)
  ENDDO
ENDDO

CALL DA1('ORBITALS.DAT',2,IREC,11,NPTS,ZSTAT)
CALL DA1('ORBITALS.DAT',2,IREC,11,NPTS,ZNUCL)
CALL DA1('ORBITALS.DAT',2,IREC,11,NPTS,ZCORE)

DO KC = KCMIN,KCMAX
  DO NC = 1,NRANG2
    CALL DA1('ORBITALS.DAT',2,IREC,11,NPTS,PA)
    CALL DA1('ORBITALS.DAT',2,IREC,11,NPTS,QA)
  ENDDO
ENDDO
```

where

- NPTS
  is the number of points in regular radial mesh (equals IRX(NIX)).

- RMESH(I) I=1,NPTS
  is an array containing the regular radial mesh.
- WEIGHT(I) I=1,NPTS
  is an array containing radial integration weights.

  $$\int_0^a f(r)\,dr \approx \sum_{i=1}^{n_{pts}} w_i\,f_i \tag{5.3}$$

  where n_{pts} is NPTS, w_i is array WEIGHT.
- ZSTAT(I) I=1,NPTS
  is the static potential, tabulated on the regular radial mesh. The behaviour of the static potential Z_s is

  $$Z_s(r) = \left\{ \begin{array}{ll} Z & \text{when } r \longrightarrow 0 \\ Z - N & \text{when } r \longrightarrow \text{RA} \end{array} \right. \tag{5.4}$$

  where Z is the atomic number (variable NZ) and N is the number of target electrons (variable NELC). This assumes that RA encloses the bound orbitals.
- ZNUCL(I) I=1,NPTS
  is the nuclear potential, tabulated on the regular radial mesh. For a point nucleus the nuclear potential $Z_n = Z$ where Z is the atomic number (variable NZ).
- ZCORE(I) I=1,NPTS
  is the core potential, tabulated on the regular radial mesh.

5.6 Routines

See the DSTG1/ORBS section for a description of routines DMCHK1, DMPRT1, DMSET1, DQSF, INTRD and RDISC1.

Routine INTS

This is the PROGRAM routine. The input and output files are set up: IREAD (`INTS.INP`) and IWRITE (`INTS.OUT`). A call is made to routine AAINTS which controls the computation.

Routine AAINTS
`(IREAD, IWRITE)`

This is the main controlling routine for the DSTG1/INTS module.

Routine DDET5
`(F, DF, HINT, IRX, NIX)`

This routine evaluates the first derivative DF of the function F at the regular mesh points.

The following formulae are used:

$$\begin{array}{lll} \text{A:} & Dy_1 = & (-25y_1 + 48y_2 - 36y_3 + 16y_4 - 3y_5)/(12h) \\ \text{B:} & Dy_2 = & (-3y_1 - 10y_2 + 18y_3 - 6y_4 + y_5)/(12h) \\ \text{C:} & Dy_3 = & (y_1 - 8y_2 + 8y_4 - y_5)/(12h) \end{array} \qquad (5.5)$$

Formula A is used for the first and last points in a subinterval.

Formula B is used for the second and second last points in a subinterval.

Formula C is the well-known central difference formula with error of order h^4. It is used at most points on the regular mesh.

Note that there must be more than 4 points in a subinterval.

Routine GEN

```
(CL, CP, CQ, DR1, EIGENS, HINT, IDISC5, IDMTST, IPOS, IREC2,
IRX, ITC, IWRITE, JAG, JTARG, KAG, KBMAX, KCMAX, KCMIN, LAB,
LAG, LAMBB, LAMBC, LAMCC, MAXFUL, MAXNLG, MAXNQN, MINNQN, NDIMAX,
NIX, NPTS, NRANG2, NUMORB, RMESH, WEIGHT, ZCORE, ZNUCL, ZSTAT)
```

Calling tree:

```
GEN
    DA1
    DMCHK1
        DMPRT1
    GENBB
        INTRD1
            DDET5
        INTRD2
            DQSF
    GENBC
        INTRD1 (see above)
        INTRD2 (see above)
    GENCC
        INTRD1 (see above)
        INTRD2 (see above)
    GENMBB
        INTRD
        INTRDV
    GENMBC
        INTRD
        INTRDV
        SKIPER
            DA1
    GENMCC
        INTRD
        INTRDV
        SKIPER (see above)
    QUARTZ
    SKIPER (see above)
```

This routine controls the generation of the radial integrals.

Routine GENBB

```
(IFLAG, CL, CP, CQ, DR1, EIGENS, HINT, IDMTST, IPOS, IRX, ITC,
IWRITE, JAG, KAG, KBMAX, MAXFUL, MAXNLG, MAXNQN, MINNQN, NIX,
NPTS, RMESH, WEIGHT, ZCORE, ZNUCL, ZSTAT, ICTX, IRK1, IRK2, ISTX,
ISTX1, ISTX2, RKSTO)
```

This routine generates the bound-bound integrals, one-electron Hamiltonian (in routine INTRD1) and two-electron Slater (in routine INTRD2).

Routine GENBC

```
(IFLAG, KC, CL, CP, CQ, DR1, EIGENS, HINT, IDMTST, IPOS, IRX,
ITC, IWRITE, JAG, KAG, KBMAX, MAXFUL, MAXNLG, MAXNQN, MINNQN,
NIX, NPTS, NRANG2, RMESH, WEIGHT, ZCORE, ZNUCL, ZSTAT, ICTX,
IRK1, IRK2, ISTX1, ISTX2, RKSTO)
```

This routine generates the bound-continuum integrals, one-electron Hamiltonian (in routine INTRD1) and two-electron Slater (in routine INTRD2), for K-value KC.

Routine GENCC

```
(IFLAG, KC, KCP, CL, CP, CQ, DR1, EIGENS, HINT, IDMTST, IPOS,
IRX, ITC, IWRITE, JAG, JTARG, KAG, KBMAX, LAB, LAG, MAXFUL,
MAXNLG, MAXNQN, MINNQN, NIX, NPTS, NRANG2, RMESH, WEIGHT,
ZCORE, ZNUCL, ZSTAT, ICTX, ICTY, IRK1, RKSTO)
```

This routine generates the continuum-continuum integrals, one-electron Hamiltonian (in routine INTRD1) and two-electron Slater (in routine INTRD2), for K-values KC and KCP. Option 27 gives a debug print out.

Routine GENMBB

```
(IFLAG, CL, CP, CQ, IDMTST, IPOS, ITC, IWRITE, KAG, KBMAX, LAG,
LAMBB, MAXNQN, NPTS, RMESH, WEIGHT, IBBPOL, IRK1, RKSTO)
```

This routine generates the bound-bound multipole integrals, length (in routine INTRD) and velocity (in routine INTRDV).

Routine GENMBC

```
(IFLAG, CL, CP, CQ, IDMTST, IPOS, ITC, IWRITE, KAG, KBMAX,
KCMAX, KCMIN, LAG, LAMBC, MAXNQN, NPTS, NRANG2, NUMORB, RMESH,
WEIGHT, IBCPOL, IRK1, RKSTO)
```

This routine generates the bound-continuum multipole integrals, length (in routine INTRD) and velocity (in routine INTRDV).

Routine GENMCC

```
(IFLAG, CL, CP, CQ, IDMTST, IPOS, ITC, IWRITE, KAG, KBMAX,
KCMAX, KCMIN, LAG, LAMCC, MAXNQN, NPTS, NRANG2, NUMORB, RMESH,
WEIGHT, ICCPOL, IRK1, RKSTO)
```

This routine generates the continuum-continuum multipole integrals, length (in routine INTRD) and velocity (in routine INTRDV).

Routine INTRD1

`(MODE, N1, N2, K1, RESULT, IWRITE, ITC, NIX, IRX, NPTS, KAG, KBMAX, MAXNLG, MAXNQN, IPOS, HINT, RMESH, WEIGHT, CL, EIGENS, ZCORE, ZNUCL, ZSTAT, CP, CQ)`

This routine evaluates the following radial integral, (one-electron Hamiltonian), over the finite interval $[0, \mathrm{RA}]$. See routine SETW for a description of the numerics.

$$I(1,2) = \int_0^{\mathrm{RA}} I(r)\; dr \tag{5.6}$$

with

$$\begin{aligned} I(r) \;\; &= \;\; c\,[Q_1 P_2' - P_1 Q_2'] + \frac{c\kappa}{r}\,[P_1 Q_2 + Q_1 P_2] \\ &\quad -\frac{(Z_n + Z_c)}{r}\,[P_1 P_2 + Q_1 Q_2] - 2c^2 Q_1 Q_2 \end{aligned} \tag{5.7}$$

where P and Q are the large and small radial components, c is the speed of light and κ is the orbital angular momentum quantum number. Z_n is a nuclear potential while Z_c is a core potential. The orbitals must have the same κ values.

The derivatives in the integrand are evaluated in routine DDET5.

For integrals involving continuum orbitals that have not been Schmidt orthogonalised it is possible to use a simpler expression by replacing the derivatives using the differential equation satisfied by the continuum orbital.

$$I(1,2) = \int_0^{\mathrm{RA}} I(r)\; dr + \delta_{n_1 n_2} E_{n_1} \tag{5.8}$$

with

$$I(r) = \frac{(Z_s - Z_n - Z_c)}{r}\,[P_1 P_2 + Q_1 Q_2] \tag{5.9}$$

where Z_s is the static potential. The energy eigenvalue E_{n_1} is added on in the diagonal case.

Option 16 gives a debug print out.

Routine INTRD2

`(N1, K1, N2, K2, N3, K3, N4, K4, K, RESULT, MODE, IWRITE, ITC, WEIGHT, RMESH, DR1, HINT, IRX, NIX, NPTS, IPOS, CP, CQ)`

This routine evaluates the following radial integral, (two-electron Slater), over the finite interval $[0, \mathrm{RA}]$. See routine SETW for a description of the numerics.

$$R^k(1,2,3,4) = \int_0^{\mathrm{RA}} \int_0^{\mathrm{RA}} I(r,s)\; dr\; ds \tag{5.10}$$

with

$$I(r,s) = [P_1(s)P_3(s) + Q_1(s)Q_3(s)]\;\; [P_2(r)P_4(r) + Q_2(r)Q_4(r)]\;\; U^k(r,s) \tag{5.11}$$

where P and Q are the large and small radial components and

$$U^k(r,s) = \begin{cases} \frac{s^k}{r^{k+1}} & \text{when } s < r \\ \frac{r^k}{s^{k+1}} & \text{when } r < s \\ \frac{1}{r} & \text{when } r = s \end{cases} \tag{5.12}$$

Write the integral as

$$R^k(1,2,3,4) = \int_0^{\mathrm{RA}} \frac{1}{r} Y^k(1,3;r)\ \left[P_2(r)P_4(r) + Q_2(r)Q_4(r)\right]\ dr \tag{5.13}$$

where the Slater potential is

$$\frac{1}{r} Y^k(1,3;r) = \int_0^{\mathrm{RA}} \left[P_1(s)P_3(s) + Q_1(s)Q_3(s)\right] U^k(r,s)\ ds \tag{5.14}$$

The Slater potential is evaluated by a call to routine DQSF.

Option 17 gives a debug print out.

Routine INTRDV

```
(N1, K1, N2, K2, RESULT, CP, CQ, WEIGHT, IWRITE, ITC, NPTS,
IPOS, KAG, CL)
```

This routine evaluates the following radial integral over the finite interval $[0, \mathrm{RA}]$. See routine SETW for a description of the numerics.

$$I_V(1,2) = c \int_0^{\mathrm{RA}} \left[(\kappa_2 - \kappa_1 + 1)P_1 Q_2 + (\kappa_2 - \kappa_1 - 1)Q_1 P_2\right]\ dr \tag{5.15}$$

where P and Q are the large and small radial components, c is the speed of light and κ are angular momentum quantum numbers. This integral occurs in the velocity form of the dipole matrix element. Option 15 gives a debug print out.

Routine RDINTS

```
(IREAD, IWRITE, ITC, JTARG, LAMBB, LAMBC, LAMCC)
```

This routine reads in the input data from file `INTS.INP`.

Routine RDISC4

```
(IDISC4, IDISC5, BSTO, CBUT, CL, DR1, EBUT, EIGENS, ENDS,
HINT, IDMTST, IHED, IRX, IWRITE, JAG, KAG, KBMAX, KCMAX,
KCMIN, KX, LAG, LAMBB, LAMBC, LAMCC, MAXFUL, MAXNLG, MAXNQN,
MBUT, MINNQN, NDIMAX, NELC, NIX, NRANG2, NUMORB, NZ, RA, RECORD)
```

This routine reads the DSTG1/ORBS dump file, `ORBS.DAT`, and writes the first part of the DSTG1 dump file, `DSTG1.DAT`. It consists of basic data, eigenvalues, amplitudes and Buttle corrections.

Routine SKIPER

```
(K1, K2, KCMIN, NPTS, NRANG2, KBMAX, LAG, MAXNQN, NUMORB,
IPOS, CP, CQ)
```

This routine reads continuum orbitals from the DA file `ORBITALS.DAT` and stores them in the CP/CQ arrays. Only at most 2 K-values are held in memory at any one time. This is required for the calculation of continuum-continuum radial integrals. The bound orbitals are always available in the CP/CQ arrays (see routine EVALUE). Therefore the dimension of the array CP would be (2∗NRANG2+NUMORB)∗NPTS where NRANG2 is the number of continuum orbitals per K-value, NUMORB is the number of bound orbitals and NPTS is the number of radial mesh points.

The array IPOS points to the position of the continuum orbitals in the CP/CQ array. For example for an orbital with N-value N1 and K-value K1 then define J1=NPTS∗(IPOS(N1,K1)−1). The large component of this orbital is CP(J) for J=J1+1,J1+NPTS and similarly for the small component in the CQ array.

The arrays written to DA file `ORBITALS.DAT` are

```
   1. RMESH            ... regular mesh
   2. WEIGHT           ... integration weights

 Records 3 and 4 are repeated for the NUMORB bound orbitals

   3. P                ... large component
   4. Q                ... small component

   5. ZSTAT            ... static potential
   6. ZNUCL            ... nuclear potential
   7. ZCORE            ... core potential

 Records 8 and 9 are repeated for the continuum orbitals
 i.e. for K=KCMIN,KCMAX with NRANG2 orbitals per K-value

   8. P                ... large component
   9. Q                ... small component
```

Thus there are 5+2∗NUMORB+2∗NRANG2∗(KCMAX−KCMIN+1) records. Each record consists of NPTS values and will correspond to a number of ‘file records’ determined by the record length used in routine DA1.

Chapter 6

DSTG2

6.1 Introduction

$Source: /home/phn/DARC/RCS/dstg2.mak,v
$Author: phn
$Date: 2001/10/31 13:03:20
$Revision: 11.1

DARC, the Dirac Atomic R-matrix Codes.
homepage: http://www.am.qub.ac.uk/DARC/
DSTG2 module

DSTG2 defines the target states and channels. Then it evaluates the angular integrals, asymptotic coefficients and continuum Hamiltonian. The photo-ionisation code evaluates dipole matrix elements.

Calling tree:

```
DSTG2
    AAMN2
        QUARTZ  timing routine
        CALEN   access date/time at execution
        DMSET2  set dimensions
        READ2   reads input data
        DA1     opens DA file INTEGRAL.DAT
        WRITP2  reads DSTG1 dump, writes part of DSTG2 dump
        BMAT    evaluates target Hamiltonian

        READ2C  reads input data for a symmetry
        CONT    generates channels, controls angular integrals
        AIJZ    evaluates asymptotic coefficients
        CMAT    evaluates continuum Hamiltonian
        DMEL    evaluates dipole matrices

        DMPRT2  final print of dimensions used
```

- Routine BMAT uses the input target CSFs to calculate a target Hamiltonian that is diagonalised. This gives the target energies and CSF mixing coefficients that are to be used in the scattering calculation. All angular integrals are calculated in DSTG2 and these are combined with the appropriate radial integrals that have been calculated in DSTG1. When you compare (as you must) these target energies with the energies that were calculated by the structure program (e.g. , GRASP) then there will be discrepancies but hopefully not large ones. These discrepancies arise because the radial integrals in the two calculations are different.
    - The radial integrals are only calculated between $r = 0$ and $r = a$. Although the R-matrix boundary should contain most of the target wavefunction, there may be a small residual effect from the wavefunction outside the boundary.

  - The bound orbitals from GRASP are interpolated from a logarithmic radial mesh (with steadily increasing stepsize) to a regular radial mesh (with constant stepsize in a subinterval). There will be an error associated with this interpolation.

  If there is a large discrepancy in the target energies then STOP and do a debug print of the radial integrals used to evaluate the target Hamiltonian. Compare these with GRASP.

- For each $(N + 1)$-electron symmetry, specified by a total angular momentum J and parity Π, the following steps are carried out.

  - Routine CONT is called to generate the channels. A channel is a target state coupled to a continuum electron. Each target state is built from a linear combination of target CSFs. In a similar way we can define a set of continuum CSFs which consist of a target CSF coupled to the continuum electron. The channels are then built from the continuum CSFs. These continuum CSFs are the basic quantities that are used to evaluate the angular integrals.

    The angular integrals are evaluated by calls to:

    * routine ANGULA for the asymptotic coefficients
    * routine ANGULH for the continuum Hamiltonian matrix
    * routine ANGULD for the dipole matrix elements

    All of the angular integrals are evaluated at this stage and stored in a file. Later they are read back and combined with the appropriate radial integrals. This procedure is adopted so that the code for the angular calculation requires little modification when taken from GRASP.
  - A call to routine AIJZ controls the calculation of the asymptotic coefficients.
  - A call to routine CMAT controls the calculation of the continuum Hamiltonian matrix.
  - A call to routine DMEL controls the calculation of the dipole matrix elements.

In a photo-ionisation calculation the first symmetry is treated as the initial state.

I/O (of both radial and angular integrals) can be a significant part of the computation.

6.2 Recoupling package: NJGRAF or NJSYM

The angular integrals in DSTG2 require the calculation of recoupling coefficients. This is done using either the NJGRAF program of Bar-Shalom and Klapisch (1988 [6]) or the NJSYM program of Burke (Scott and Hibbert 1982 [55]). The current version of NJGRAF is taken from GRASP92 (Parpia et al 1996 [50]).

NJGRAF is much faster than NJSYM in most cases and is recommended. You must choose one of these in preprocessing. Lines beginning with {NJGRAF} refer to the NJGRAF version while lines beginning with {NJSYM} refer to the NJSYM version.

6.3 Files

The files are defined internally. Messages are also written to output stream 6 using the PRINT command. The following files are used:

stream name	description	stream number	file name	FORM
IREAD	input data	5	`DSTG2.INP`	Formatted
IWRITE	printed output	7	`DSTG2.OUT`	Formatted
IDISC1	‘angular’ file	11	scratch	Unformatted
IDISC2	‘radial’ file	12	scratch	Unformatted
IDISC3	AIJ angular coefficients (bound parts)	13	scratch	Unformatted
IDISC4	direct slater coefficients	14	scratch	Unformatted
IDISC5	storage of asymptotic coefficients	15	scratch	Unformatted
ITAPE1	DSTG1 dump file	20	`DSTG1.DAT`	Unformatted
ITAPE2	DSTG2 dump file	21	`DSTG2.DAT`	Unformatted
—	radial integral DA file	22	`INTEGRAL.DAT`	Unformatted

DA means direct access.

6.4 Input data (`DSTG2.INP`)

The input data uses Namelist format. Each Namelist has a name. If this is say DSTG2 then the input data starts with the string &DSTG2. There must be a space before the & even if it is at the beginning of the line. The end of this Namelist is denoted by &END, with & again preceded by a space. Variables in a Namelist can be given default values. The default value is indicated in brackets after the variable name.

Input records 1 and 2 are read by routine READ2. Input records 3, 4 and 5 are read by routine DATNR. Input record 6 is read by routine READ2C.

The following input data is for nonrelativistic CSFs. The input data is read by routine DATNR. Having defined the nonrelativistic CSFs, a set of NCF relativistic CSFs is generated from them (by calls to routine MAN5 that in turn calls COUP). The relativistic CSFs can be printed by a call to CFOUT. A restriction on this type of input data is that there can be no more than five open shells in any nonrelativistic CSF.

Input record 1

Read the 80-character title, variable IHED.

Input record 2

Namelist DSTG2 ...

```
      INTEGER IPOLPH
      INTEGER NAST
      INTEGER NMAN
```

```
INTEGER NWM
INTEGER OPT(50)
NAMELIST / DSTG2 / IPOLPH,NAST,NMAN,NWM,OPT
```

- IPOLPH [1]
  is a parameter for determining the type of calculation:
  = 1 for electron scattering;
  = 2 for photo-ionisation or polarisation calculation

- NAST [−1]
  The first NAST target levels are to be included in the generation of channels. If NAST > NCF or NAST < 1 (the default) then NAST is reset to NCF (i.e. all levels are used). NCF is the number of relativistic CSFs.

- NMAN [−1]
  is the number of nonrelativistic target CSFs (compulsory).

- NWM [−1]
  is the number of nonrelativistic bound orbitals (compulsory).

- OPT [0] – array

  Input a list of integers giving the options to be set. An option is a switch that is either on or off. To set option J input J on this input record. The options are stored in array ITC. If option J is set then ITC(J)=1; otherwise ITC(J)=0.

  Operational options.

  - 1
    When this option is set only direct angular coefficients are calculated for the continuum Hamiltonian. This corresponds to a non-exchange calculation. It is achieved by setting the parameter IEXC in routine ANGULH and will only be used if there are no correlation functions for the particular symmetry being considered.
  - 2
    When this option is set the maximum value for λ in the asymptotic scattering coefficients is reduced to 2. Higher values of λ are not used in the perturbation code of Seaton and this option can save some time for the case of ions.
  - 3
    Skip the computation of asymptotic scattering coefficients. These are sometimes not required when we consider scattering from highly ionised systems. In this case we match to Coulomb solutions in the external region and the asymptotic scattering coefficients are not used.
  - 47
    In evaluating the angular integrals required to form the Hamiltonian matrix it was noticed that there was a large amount of degeneracy in the calculation. This arose from the definition of the continuum CSFs which were originally different

for each channel function. The code has been modified to remove the duplicate calculation. However this option has been retained to allow the program to be run in its original form.

- 48
  When this option is set asymptotic scattering coefficients are calculated for every symmetry. Thus there is no reuse of coefficients for symmetries with the same total J-value.
- 49
  When this option is set there is no attempt to reuse direct Hamiltonian angular coefficients from a previous symmetry with the same total J-value. By default the code will store direct angular coefficients on stream IDISC4 and attempt to use them if the next symmetry has the same total J-value.
- 50
  When this option is set angular asymptotic scattering coefficients are calculated for all pairs of continuum CSFs. There should be symmetry between the pairs of continuum CSFs and this option can be used to check this occurs correctly.

Printing options.

- 7
  Call to CFOUT for target case. This routine writes out the relativistic CSFs in an easily read form. When there are a large number of CSFs the printing can be excessive.
- 8
  Call to CFOUT for continuum case. As above.
- 9
  Call to CFOUT for photo-ionisation case. As above.
- 10
  Print the target energies in eV, cm^{-1} and Ang in routine PROP.
- 11
  Print the target Hamiltonian in routine BMAT.
- 12
  Print the target eigenvectors in routine PROP.
- 13
  Print the target Hamiltonian angular coefficients in routine MCPINB.
- 14
  This option provides a debug printout from routine MCPINB of the arrays NNLDR, NSLDF, ISLDR, JSLDR and XSLDR that are used to store the target Hamiltonian angular coefficients.
- 15
  This option gives a debug print from routine BMATX. Note that this routine is called by BMAT to evaluate the target Hamiltonian and by CMAT to evaluate the bound-bound contribution to the continuum Hamiltonian.

- 20
  Print the dipole and quadrupole coupling in routine AIJZ.
- 21
  Print the asymptotic scattering coefficients in routine AIJZ.
- 22
  Printout from routine FINM. This will give a print of the angular and radial integrals used in routine AIJZ.
- 23
  Use this option to examine the build-up of the asymptotic scattering coefficients with target CI.
- 25
  Printout from routine DMEL, information on photo-ionisation matrix elements.
- 26
  Printout from routine DELIN, gives angular integrals for photo-ionisation.
- 27
  Printout from routines FINMBB, FINMBC and FINMCC, gives radial integrals for photo-ionisation.
- 30
  Print the continuum Hamiltonians.
- 31
  Use this option to examine the build-up of the continuum-continuum contributions to the continuum Hamiltonian with target CI.
- 32
  Use this option to examine the build-up of the bound-continuum contributions to the continuum Hamiltonian with target CI.
- 33
  This option prints out the bound-bound contributions to the continuum Hamiltonian.
- 34
  Print the continuum Hamiltonian angular coefficients in routine MCPINC.
- 35
  This option provides a debug printout from routine MCPINC of the arrays NNLDR, NSLDF, ISLDR, JSLDR and XSLDR that are used to store the Hamiltonian angular coefficients.
- 36
  One-line printout from the LOC... routines. See also option 37.
- 37
  This option gives additional output from the LOC... routines. When it is set the integer arrays that are used to locate the radial integrals in the block are printed.
- 38
  One-line printout from routine FINBB.

- 39
  One-line printout from routine FINBC. See option 41 for extra printing.
- 40
  One-line printout from routine FINCC. See option 41 for extra printing.
- 41
  This option allows the printout of radial integrals from FINCC and FINBC. This applies only if options 40 and 39 have been set for FINCC and FINBC respectively.

Input record 3 — orbital data

Namelist ORB ...

```
INTEGER PRINC
INTEGER KAPPA
INTEGER CSF(MXNC)
NAMELIST / ORB / PRINC,KAPPA,CSF
```

Read NWM input records, one input record for each orbital J, containing the following data:

- PRINC
  is the principal quantum number.
- KAPPA
  is the κ quantum number. κ is a nonzero integer. For nonrelativistic orbitals it takes the values $-1, -2, -3, \ldots$ which correspond to s, p, d, ...
- CSF
  is an array containing orbital occupation numbers for each CSF:
  0 numbers — subshell is full in each CSF
  1 number — each CSF has this occupation
  NMAN numbers — occupation for each CSF

The program cannot handle the case of relativistic orbitals with $j > \frac{7}{2}$ and occupation number $2 < q < 2j - 1$. To handle such a case would require an extension to the tables of fractional parentage coefficients.

Input record 4 - Options for the angular momentum programs

Namelist ANGOPT ...

```
INTEGER OPT(20)
NAMELIST / ANGOPT / OPT
```

- OPT

  If option J is to be set then input the number J. Up to 6 options apply to the angular routines. These are used to set the variables IBUG1 to IBUG6.

  - 1
    write out the angular coefficients calculated
  - 2
    output from RKCO, COR, MUMDAD, VIJOUT (two-electron case)
  - 3
    output from NJGRAF/NJSYM
  - 4
    output from VIJOUT (one-electron case)
  - 5
    output from TMSOUT
  - 6
    output from TNSRJJ (one-electron case)

Input record 5 — nonrelativistic CSF total *J*-values

Namelist JVALUE ...

```
      INTEGER JOPT
      DOUBLE PRECISION JLOW(MXNC)
      DOUBLE PRECISION JHIGH(MXNC)
      NAMELIST / JVALUE / JOPT,JLOW,JHIGH
```

This input record specifies the total J-values of the nonrelativistic CSFs.

- JOPT = −1
  If JLOW(1) is not specified then all possible CSFs are to be generated i.e. all possible J-values. Optionally input JLOW(1) and only CSFs with this J-value are included.

- JOPT = −2
  You must specify NMAN values for JLOW and NMAN values for JHIGH. These specify limits for the allowed J-values for the particular CSF i.e. only J-values satisfying JMIN $\leq J \leq$ JMAX are included.

- JOPT = −3
  You must specify JLOW(1) and JHIGH(1) values. These specify limits for the allowed J-values for all the CSFs i.e. only J-values satisfying JMIN $\leq J \leq$ JMAX are included.

- JOPT = −4
  You must specify NMAN values for JLOW. These are the J-values of the CSFs in the order described above.

Input record 6

Namelist SYM ...

```
INTEGER CORINC(MXNC)
DOUBLE PRECISION JTOT
INTEGER NCFGP
INTEGER NPTY
NAMELIST / SYM / CORINC,JTOT,NCFGP,NPTY
```

Repeat for each $(N+1)$-electron symmetry required. Terminate with JTOT= -1. Also the code will stop if this input record is not present.

- CORINC [1]
  is an array used to mark the nonrelativistic CSFs that are to be included in generating correlation functions. By default CORINC is set to 1, all CSFs are to be included. By setting CORINC(I) to 0 then CSF I will be excluded.
- JTOT [-1.0]
  is the total J-value for the $(N+1)$-electron symmetry. JTOT is a real variable such as 0.5 or 1.5 or 2.0. It is used to form the integer J2P1=2$*$JTOT+1. Set negative to end input.
- NPTY [0]
  is the parity of the $(N+1)$-electron symmetry (-1 for odd or 1 for even)
- NCFGP [-1]
  is the number of correlation functions to be included. Default gives all correlation functions. At present you can have either 0 or all possible correlation functions.

6.5 DSTG2 dump file (`DSTG2.DAT`)

This file is unformatted. The first 13 records of the DSTG1 and DSTG2 dump files are the same.

Write target data

```
14. NAST
15. (JAT(I),I=1,NAST)
16. (ENAT(I),I=1,NAST)
```

where

- NAST
  is the number of target states.

- JAT(I) I=1,NAST
  is an array of $2J+1$ angular momenta multiplied by the parity ($+1$ even, -1 odd) for the target states. The states are in the same order as array ENAT (target energies).
- ENAT(I) I=1,NAST
  is an array of target energies in au, ordered by increasing energy.

Write the continuum data

Records 17 to 26 are repeated for each $(N+1)$-electron symmetry.

Set J2P1X=−J2P1 if IPOLPH=2 otherwise J2P1X=J2P1 where J2P1 is $(2J+1)$ for the symmetry.

```
17. J2P1X,NPTY,NCFGP
18. MNP1,NCHAN
19. (NCONAT(I),I=1,NAST)
20. (K2P(I),I=1,NCHAN)
```

Write the asymptotic coefficients

Write record 22 if IKMAX > 0
Repeat record 22 for LAM = 1,IKMAX

```
21. IKMAX
22. ((ASY(I,J),J=1,NCHAN),I=1,NCHAN)
```

Write the continuum Hamiltonian

Continuum-continuum part: records 23 and 24
Repeat records 23 and 24 for I = 1,NCHAN and J = I,NCHAN
Note that they are not in ascending order

Write record 24a if I = J
Write record 24b if I ≠ J

```
23. I,J

24a. ((HAMIL(L1,L2),L2=L1,NRANG2),L1=1,NRANG2)

24b. ((HAMIL(L1,L2),L2=1,NRANG2),L1=1,NRANG2)
```

Bound-continuum part: record 25
This is only written if NCFGP > 0
Repeat record 25 for I = 1,NCHAN
Note that they are not in ascending order

```
25. I, ((HAMIL(L1,L2),L2=1,NCFGP),L1=1,NRANG2)
```

Bound-bound part : record 26
This is only written if NCFGP > 0
Repeat record 26 for I = 1,NCFGP

```
26. (XMT(I,J),J=I,NCFGP)
```

Write the dipole matrix elements

If IPOLPH=2, then these are written for each symmetry except the first. The following 3 records are written for each dipole matrix block. The end of the blocks is denoted by a single record 27a in which all the variables are set to −1. Array DEL contains the length matrix elements while array DEV contains the velocity matrix elements.

```
27a. I,J,K,L
27b. ((DEL(M,N),M=1,K),N=1,L)
27c. ((DEV(M,N),M=1,K),N=1,L)

28. MAXM1
29. (CGC(M),M=1,MAXM1)
```

The following code indicates the order in which the dipole matrix elements are written.

```
      DO I = 1,NCHAN
C
C  continuum-continuum terms
C  -------------------------
C
        DO J = 1,INCHAN
          WRITE (ITAPE2) I,J,NRANG2,NRANG2
          WRITE (ITAPE2) ((DEL(L1,L2),L1=1,NRANG2),L2=1,NRANG2)
          WRITE (ITAPE2) ((DEV(L1,L2),L1=1,NRANG2),L2=1,NRANG2)
        ENDDO
C
C  continuum-bound terms
C  ---------------------
C
        J = 0
        WRITE (ITAPE2) I,J,NRANG2,ICFCOR
        WRITE (ITAPE2) ((DEL(L1,L2),L1=1,NRANG2),L2=1,ICFCOR)
        WRITE (ITAPE2) ((DEV(L1,L2),L1=1,NRANG2),L2=1,ICFCOR)
C
      ENDDO
C
C  bound-continuum terms
C  ---------------------
C
      DO J = 1,INCHAN
        I = 0
```

```
          WRITE (ITAPE2) I,J,NCFGP,NRANG2
          WRITE (ITAPE2) ((DEL(L1,L2),L1=1,NCFGP),L2=1,NRANG2)
          WRITE (ITAPE2) ((DEV(L1,L2),L1=1,NCFGP),L2=1,NRANG2)
        ENDDO
C
C  bound-bound terms
C  -----------------
C
        I = 0
        J = 0
        WRITE (ITAPE2) I,J,NCFGP,ICFCOR
        WRITE (ITAPE2) ((DEL(L1,L2),L1=1,NCFGP),L2=1,ICFCOR)
        WRITE (ITAPE2) ((DEV(L1,L2),L1=1,NCFGP),L2=1,ICFCOR)
C
C  mark end of this data on file
C  -----------------------------
C
        I = -1
        J = -1
        K = -1
        L = -1
        WRITE (ITAPE2) I,J,K,L
C
C  evaluate 3J symbol for polarisability
C  -------------------------------------
C
        WRITE (ITAPE2) MAXM1
        WRITE (ITAPE2) (CGC(M),M=1,MAXM1)
```

6.6 Routines

Routine DSTG2

This is the PROGRAM routine. The input and output files are set up: IREAD (DSTG2.INP) and IWRITE (DSTG2.OUT). A call is made to routine AAMN2 which controls the computation.

Routine AAMN2
(IREAD, IWRITE)

This is the main controlling routine for the DSTG2 module.

Routine AIJ1
(IDISC3, IBUG1, IBUG2, IBUG3, IBUG4, IBUG6, ICHOP, IQ, ISPAR,
ITC, ITJPO, IWRITE, JCUP, JQS, NAK, NCF, NH, NP, NW, ITAB,
JTAB, NTAB)

This routine is called by routine ANGULA and carries out part 1 of the evaluation of the angular asymptotic coefficients.

Routine AIJ2

`(IDISC1, IDISC3, IBUG1, ITC, ITJPO, IWRITE, J2P1, K2P,`
`NCCK, NCCT, NCFCON, NH, NP)`

This routine is called by routine ANGULA and carries out part 2 of the evaluation of the angular asymptotic coefficients.

Routine AIJZ

`(ASY, EMTR, IBBPOL, IBCPOL, ICCPOL, IDISC1, IDISC5,`
`IPOLPH, IRK1, ITAPE2, ITC, IWRITE, J2P1, JLAST, K1S,`
`K2P, K2S, KBMAX, KCMAX, KMULTN, KMULTR, LAMBB, LAMBC,`
`LAMCC, NAST, NCASES, NCCA, NCCN, NCCT, NCFGP, NCHAN,`
`NPTY, NRANG2, NTARG, RKSTO, LAG, MAXNQN, NAK, NH, NP)`

This routine evaluates the asymptotic coefficients required for the external region. The angular coefficients are combined with the appropriate radial integrals.

Calling tree:

```
AIJZ
    AIJZ1
    AIJZ2
    FINM
    LOCM
        DA1
    MATOUT
```

Routine AIJZ1

`(ASY, IMARK, IWRITE, NAST, NCHAN, NTARG)`

This routine checks for dipole coupling.

Routine AIJZ2

`(ASY, IMARK, IWRITE, NAST, NCHAN, NTARG)`

This routine checks for quadrupole coupling.

Routine ANGLB1

`(IWRITE, ICHOP, IQ, ISPAR, ITJPO, JCUP, JQS, NCF, NW,`
`NICHOP, IQTAR, NISPAR, NITJPO, NJCUP, NJQS, NCFTAR, NWTAR)`

This routine copies the bound state information into the TARGET data.

```
TARGET
------
NWTAR        ... NW
NCFTAR       ... NCF
NICHOP(J,I) ... ICHOP(J,I) , J = 1,NW , I = 1,NCF
IQTAR(J,I)  ... IQ(J,I)    , J = 1,NW , I = 1,NCF
NJQS(K,J,I) ... JQS(K,J,I) , K = 1,3 , J = 1,NW , I = 1,NCF
NITJPO(I)   ... ITJPO(I)   , I = 1,NCF
NISPAR(I)   ... ISPAR(I)   , I = 1,NCF
NJCUP(K,I)  ... JCUP(K,I)  , K = 1,10 , I = 1,NCF
```

Routine ANGLB2

```
(IWRITE, ICHOP, IQ, ISPAR, ITJPO, JCUP, JQS, NCF, NW,
NCFCON, NCFGP, NICHOP, IQTAR, NISPAR, NITJPO, NJCUP, NJQS,
NCFTAR, NWTAR)
```

This routine copies the bound state information from the TARGET data.

Routine ANGLD1

```
(ICFCON, ICFCOR, ICHOP, IICHOP, IIQ, IISPAR, IITJPO, IJ2P1,
IJCUP, IJQS, INAK, INAKK, INCCP, INCF, INCHAN, INH, INTARG,
IQ, ISPAR, ITJPO, IWRITE, J2P1, JCUP, JQS, NAK, NAKK, NCCP, NCF,
NCFCON, NCFGP, NCFTAR, NCHAN, NH, NTARG, NW)
```

This routine, for a photo-ionisation calculation, copies the initial state information into the INITIAL data.

```
   INITIAL
   -------
   INCHAN      ... NCHAN
   IJ2P1       ... J2P1
   ICFCOR      ... NCFGP
   ICFCON      ... NCFCON
   INCF        ... NCF
   INH(J)      ... NH(J)      , J = 1,NW
   INAK(J)     ... NAK(J)     , J = 1,NW
   IICHOP(J,I) ... ICHOP(J,I) , J = 1,NW , I = 1,NCF
   IIQ(J,I)    ... IQ(J,I)    , J = 1,NW , I = 1,NCF
   IJQS(K,J,I) ... JQS(K,J,I) , K = 1,3  , J = 1,NW , I = 1,NCF
   IITJPO(I)   ... ITJPO(I)   , I = 1,NCF
   IISPAR(I)   ... ISPAR(I)   , I = 1,NCF
   IJCUP(K,I)  ... JCUP(K,I)  , K = 1,10 , I = 1,NCF
   INTARG(N)   ... NTARG(N)   , N = 1,NCHAN
   INAKK(N)    ... NAKK(N)    , N = 1,NCHAN
   INCCP(L,N)  ... NCCP(L,N)  , L = 1,NCFTAR , N = 1,NCHAN
```

Routine ANGULA

```
(IDISC1, IDISC3, IBUG1, IBUG2, IBUG3, IBUG4, IBUG6, ICHOP, IQ,
ISPAR, ITAB, ITC, ITJPO, IWRITE, J2P1, JCUP, JLAST, JQS, JTAB,
K2P, NAK, NCASES, NCCK, NCCT, NCF, NCFCON, NH, NP, NTAB, NW)
```

This routine controls the evaluation of angular coefficients for the long range scattering potentials (asymptotic coefficients).

Part 1 (routine AIJ1): calculate the parts of the coefficients which depend only on the target angular momenta. These are independent of the channel angular momenta. The coefficients are written to stream IDISC3.

Part 2 (routine AIJ2): complete the calculation of the asymptotic coefficients. Read the parts which depend on the target and include the factors which depend on the channel angular momenta. Results from this routine are written to stream IDISC1.

Calling tree:

```
ANGULA
    AIJ1
        CRE
            CLRX
        TNSRJJ   ---- see routine
    AIJ2
        CRE
            CLRX
        DRACAH
```

Routine ANGULD

```
(IDISC1, IBUG1, IBUG2, IBUG3, IBUG4, IBUG6, ICFCON, ICFCOR, ICHOP,
IDMTST, IICHOP, IIQ, IISPAR, IITJPO, IJCUP, IJQS, INAK, INCF,
INCHAN, INH, IQ, ISPAR, ITAB, ITC, ITJPO, IWRITE, JCUP, JQS, JTAB,
NAK, NCF, NCFCON, NCFGP, NCHAN, NDIMAX, NH, NP, NTAB, NW, NWTAR)
```

This routine, for a photo-ionisation calculation, evaluates the angular coefficients and writes them to stream IDISC1

Routine ANGULH

```
(IDISC1, IBUG1, IBUG2, IBUG3, IBUG4, IBUG5, IBUG6, ICHOP, IDMTST,
IEXC, IQ, ISPAR, ITAB, ITC, ITJPO, IWRITE, J2P1, JCUP, JLASTX,
JQS, JTAB, K2P, NAK, NCCK, NCCT, NCF, NCFCON, NCFGP, NCHAN,
NDIMAX, NH, NHC, NP, NPTY, NROWS, NTAB, NW)
```

This routine sets up the arrays for the calculation of angular coefficients for the continuum Hamiltonian

Calling tree:

```
ANGULH
    CFOUT
    DMCHK2
        DMPRT2
    SMCP       ---- see routine
```

Routine BLDST2 — block data

This block data sets up tables of quantum numbers of terms which can be formed by CSFs j^n. Symmetry of the table for particle/hole CSFs is used to compress it.

Routine BMAT

```
(IDISC1, ITAPE2, EMTB, EMTR, ENAT, FACTAN, FACTCM, FACTEV, IBUG1,
IBUG2, IBUG3, IBUG4, IBUG5, IBUG6, ICHOP, IDMTST, IEXC, IQ, IRK1,
IRK2, ISPAR, ITC, ITJPO, IWRITE, JCUP, JQS, KBMAX, MAXFUL, MAXNQN,
MINNQN, NAK, NAST, NCF, NDIMAX, NH, NP, NW, NW1, ICTXB, ISTX1B,
ISTX2B, ISTXB, KBBN1, KBBN2, KBBR, K1S, K2S, ITAB, JTAB, NROWS,
NTAB, RKSTO, ISLDR, JSLDR, NMCP, NNLDR, NSLDF, XSLDR)
```

This routine sets up the bound Hamiltonian matrix.

Calling tree:

```
BMAT
    BMATX
        CLRX
        FINBB
            PLACE2
            PLACE3
    CFOUT
    DSYEV
    LOCBB
        DA1
    MATOUT
    MCPINB
        DMCHK2
            DMPRT2
    PROP
        MATOUT
    SMCP        ---- see routine
```

Routine BMATX

`(XMT, IFST, ILST, JST, ICTXB, IQ, IRK1, ISLDR, ISTX1B, ISTX2B, ISTXB, ITC, IWRITE, JSLDR, KBMAX, MAXFUL, MAXNQN, MINNQN, NAK, NH, NMCP, NNLDR, NP, NSLDF, NW, NW1, RKSTO, XSLDR)`

This routine is called by routine BMAT to evaluate the target Hamiltonian and by routine CMAT to evaluate the bound-bound contribution to the continuum Hamiltonian. The result is stored in array XMT. Arguments IFST and ILST mark the first and last CSFs to be used i.e. the size of the Hamiltonian is then ILST−IFST+1. The list of CSFs can consist of the continuum+correlation CSFs. Argument JST marks the (first−1) CSF in the array IQ.

Routine CFOUT

`(IQ, ISPAR, ITJPO, IWRITE, JCUP, JQS, NAK, NCF, NH, NP, NW)`

This routine prints relativistic CSF data in a neat format.

Routine CFP

`(IWRITE, LOCK, NEL, IJD, IVD, IWD, IJP, IVP, IWP, COEFP)`

This routine selects the appropriate table of CFPs (coefficients of fractional parentage) in jj-coupling.

Calling tree:

```
CFP
    CFP3
    CFP5
    CFP7
    CFPD
```

These routines have been adapted from the program of Grant (1972 [36]). The routine will return results for $j = \frac{3}{2}, \frac{5}{2}, \frac{7}{2}$. Higher values of j return a value 1.0 if NEL (the number of equivalent electrons in the shell) is 1 or 2; otherwise 0.0 with an error signal.

Routine CFP3
`(IWRITE, NEL, IJD, IJP, COEFP)`

Table look-up for CFPs of equivalent electrons with $j = \frac{3}{2}$.

Routine CFP5
`(IWRITE, NEL, IJD, IVD, IJP, IVP, COEFP)`

Table look-up for CFPs of equivalent electrons with $j = \frac{5}{2}$.

Routine CFP7
`(IWRITE, NEL, IJD, IVD, IJP, IVP, COEFP)`

Table look-up for CFPs of equivalent electrons with $j = \frac{7}{2}$.

Routine CFPD
`(IWRITE, LOCK, NEL, COEFP)`

This routine returns values for 1 or 2 particle or single hole states, and signals an error otherwise.

Routine CLEGOR
`(XJ1, XJ2, XM1, XM2, XJ3, XM3, CG, IER)`

This routine calculates Clebsch-Gordon coefficients.

Routine CLRX — function
`(KAP1, K, KAP2)`

The value of function CLRX is the $3j$-symbol

$$\begin{pmatrix} j_1 & K & j_2 \\ \frac{1}{2} & 0 & -\frac{1}{2} \end{pmatrix} \tag{6.1}$$

where j_1 and j_2 are half-integral angular momenta, and K is integral. The triangular condition $\Delta(j_1 K j_2)$ must be satisfied.

See Brink and Satchler (1993 [17]) p.138.

Routine CMAT
```
(IDISC1, IDISC2, IDISC4, ITAPE2, EMTB, EMTR, ENAT, FACTEV, ICTXB,
IDMTST, IQ, IRK1, IRK2, ISLDR, ISTX1B, ISTX2B, ISTXB, ITC, IWRITE,
J2P1, JLASTX, JSLDR, K1S, K2P, K2S, KBBN1, KBBN2, KBBR, KBCN1,
KBCN2, KBCR, KBMAX, KCCN, KCCR, KCMAX, KCMIN, KPOS, MAXFUL, MAXNQN,
MAXP, MINNQN, NAK, NAKK, NCCK, NCCP, NCCT, NCF, NCFCON, NCFGP,
NCHAN, NDIMAX, NH, NMCP, NNLDR, NP, NRANG2, NSLDF, NTARG, NW, NW1,
RKSTO, XSLDR)
```

This routine constructs the continuum Hamiltonian matrix.

Calling tree:

```
CMAT
    BMATX
```

```
        CLRX
        FINBB
            PLACE2
            PLACE3
    CMATX
        CLRX
        FINBB
            PLACE2
            PLACE3
        FINCC
            LOCCC
                DA1
            MATOUT
            PLACE
        LOCBB
            DA1
    CMATY
        FINBC
            LOCBC
                DA1
            PLACE1
    DSYEV
    LOCBB
        DA1
    MATOUT
    MCPINC
        DMCHK2
            DMPRT2
```

Routine CMATX

```
(IDISC2, ICH, JCH, NXX, IR, IS, MX, TOTAL, ITC, IWRITE, EMTB, IQ, K2P,
KBMAX, KCMAX, KCMIN, MAXFUL, MAXNQN, MINNQN, NAK, NAKK, NCCK,
NCCT, NH, NP, NRANG2, NW, NW1, KBBN1, KBBN2, KBBR, KCCN, KCCR,
K1S, K2S, KPOS, MAXP, IRK1, IRK2, ICTXB, ISTX1B, ISTX2B, ISTXB,
RKSTO, ISLDR, JSLDR, NMCP, NNLDR, NSLDF, XSLDR)
```

This routine forms blocks of the Hamiltonian for continuum CSFs IR and IS.

Routine CMATY

```
(IDISC2, NXX, I, TOTAL, ITC, IWRITE, KBMAX, KCMAX, KCMIN, MAXFUL,
MAXNQN, MINNQN, NAK, NAKK, NCFCON, NCFGP, NH, NP, NRANG2, NW,
NW1, K1S, K2S, KPOS, MAXP, KBCN1, KBCN2, KBCR, RKSTO, ISLDR, JSLDR,
NNLDR, NSLDF, XSLDR)
```

This routine forms bound-continuum contributions for continuum CSF I.

Routine CONT

```
(IDISC1, IDISC3, MX, EMTR, IBUG1, IBUG2, IBUG3, IBUG4, IBUG5,
IBUG6, ICFCON, ICFCOR, ICHOP, IDMTST, IEXC, IICHOP, IIQ, IISPAR,
IITJPO, IJ2P1, IJCUP, IJQS, INAK, INAKK, INCCP, INCF, INCHAN, INH,
INTARG, IPOLPH, IQ, ISPAR, ITAB, ITC, ITJPO, IWRITE, J2P1, JCUP,
```

JLAST, JLASTX, JQS, JTAB, K2P, KCMAX, KCMIN, NAK, NAKK, NAST, NCASES, NCCA, NCCK, NCCN, NCCP, NCCT, NCF, NCFCON, NCFGP, NCHAN, NDIMAX, NH, NHC, NP, NPTY, NROWS, NTAB, NTARG, NW)

This routine controls the angular integrations.

Calling tree:

```
CONT
    ANGLB1
    ANGLB2
    ANGLD1
    ANGULA
        AIJ1
            CRE
                CLRX
            TNSRJJ    ---- see routine
        AIJ2
            CRE
                CLRX
            DRACAH
    ANGULD
        CFOUT
        CRE
            CLRX
        DMCHK2
            DMPRT2
        TNSRJJ        ---- see routine
    ANGULH
        CFOUT
        DMCHK2
            DMPRT2
        SMCP          ---- see routine
    DMCHK2
        DMPRT2
```

Routine COR

(JA1, JB1, JA2, JB2, IBUG1, IBUG2, IBUG3, IEXC, IME, IWRITE, JA, JB, NAK, NH, NOUTX, NP, NW, NWA, ITAB, JTAB, NTAB, JJC1, JJC2, JJQ1, JJQ2, JLIST, NPEEL, NQ1, NQ2)

This routine computes the MCP/MCBP coefficients. The equations can be found in Grant (1973 [37]).

Calling tree:

```
COR
    CRE
        CLRX
    CXK
        CRE
        CLRX
GENSUM       --- see routine
LTAB
MODJ23
MUMDAD
    CFP      --- see routine
NJGRAF       --- see routine
OCON
SETJ
SKRC
SNRC
SPEAK
```

Routine COUP

`(IWRITE, NL, NQ, JI, JF, JQS, IQ, JCUP, ICASE, MODE, ICASEX, ITAB, JTAB, NROWS, NTAB)`

This routine determines coupling for nonrelativistic CSFs.

Routine CRE — function

`(KAP1, K, KAP2)`

This function computes the relativistic reduced matrix element, in the Fano & Racah convention,

$$(l_1\tfrac{1}{2}j_1 \mid\mid C_K \mid\mid l_2\tfrac{1}{2}j_2) = (-)^{j_1+\frac{1}{2}}\sqrt{(2j_1+1)(2j_2+1)}\begin{pmatrix} j_1 & K & j_2 \\ \frac{1}{2} & 0 & -\frac{1}{2} \end{pmatrix} \tag{6.2}$$

where j_1 and j_2 are half-integral angular momenta, and K is integral. The triangular condition $\Delta(j_1Kj_2)$ must be satisfied. Also we must have l_1+K+l_2 even.

See Brink and Satchler (1993 [17]) p.153.

Routine CXK

`(S, IS, KAPS, NU, K, IBR, IEX, IWRITE)`

This routine evaluates factors used for the Breit angular coefficients.

Routine DATNR

`(NMAN, NWM, IBUG1, IBUG2, IBUG3, IBUG4, IBUG5, IBUG6, ICHOP, IDMTST, IPOS, IQ, IREAD, ISPAR, ITAB, ITJPO, IWRITE, JCHOP, JCUP, JQS, JTAB, NAK, NCF, NDIMAX, NH, NLX, NP, NQX, NROWS, NTAB, NW)`

This routine reads input data defining nonrelativistic CSFs and uses it to generate all possible relativistic CSFs which produce the required total J-value.

Routine DELIN

`(IDISC1, IDMTST, NDIMAX, ITC, IWRITE, ICFCON, ICFCOR, NCFCON, NCFGP, ISLDR, JSLDR, MNLDR, NMCP, XSLDR)`

This routine reads the photo-ionisation angular coefficients.

Routine DMCHK2
`(I, ID, IWRITE, IDMTST, NDIMAX)`

This routine is called to test array overflow.

Routine DMEL

```
(IDISC1, ITAPE2, EMTR, IBBPOL, IBCPOL, ICCPOL, ICFCON, ICFCOR,
IDMTST, IJ2P1, INAKK, INCCP, INCHAN, INTARG, IRK1, ISLDR, ITC,
IWRITE, J2P1, JSLDR, K1S, K2S, KBMAX, KCMAX, KMULTN, KMULTR,
LAG, LAMBB, LAMBC, LAMCC, MAXNQN, MNLDR, NAK, NAKK, NCCP, NCF,
NCFCON, NCFGP, NCHAN, NDIMAX, NH, NMCP, NP, NRANG2, NTARG, RKSTO,
XSLDR)
```

This routine constructs the photo-ionisation matrix elements.

Calling tree:

```
DMEL
    CLEGOR
    DELIN
        DMCHK2
            DMPRT2
    FINMBB
    FINMBC
    FINMCC
    LOCM
        DA1
    MATOUT
```

Routine DMPRT2
`(IWRITE, IDMTST, NDIMAX)`

This routine writes out information on the dimensions used.

Routine DMSET2
`(IWRITE, IDMTST, NDIMAX)`

This routine initialises the arrays (IDMTST, maxima set, and NDIMAX, maxima used) that check on dimensions. Dimensions are set by the file `darc.inc` which is included in the relevant routines. All dimension variables beginning with MX are set in these files. To change a dimension you need only change `darc.inc`.

Routine FINBB — function

```
(ICON, JA, JB, JC, JD, LAM, ICTXB, IRK1, ISTX1B, ISTX2B, ISTXB,
ITC, IWRITE, KBMAX, MAXFUL, MAXNQN, MINNQN, NAK, NH, NP,
RKSTO)
```

This function locates a bound-bound radial integral. The parameter ICON determines the integral type:
=1 for $I(ab)$ integrals,
=2 for $F^k(ab)$ integrals,

=3 for $G^k(ab)$ integrals,
=4 for $R^k(abcd)$ integrals.

Routine FINBC

```
(IDISC2, JJ, WA, ICON, JJA, JJB, JJC, JJD, LAM, TOTAL, ITC, IWRITE,
K1S, K2S, KBCN1, KBCN2, KBCR, KBMAX, KCMAX, KCMIN, KPOS,
MAXFUL, MAXNQN, MAXP, MINNQN, NAK, NAKK, NH, NP, NRANG2, NW, RKSTO)
```

This routine locates bound-continuum radial integrals. The parameter ICON determines the integral type:
=1 for $I(a, b)$ integrals,
=2 for $R^k(abcd)$ integrals.

Routine FINCC

```
(IDISC2, WA, ICON, JA, JB, JC, JD, LAM, TOTAL, ITC, IWRITE, K1S, K2S,
KBMAX, KCCN, KCCR, KCMAX, KCMIN, KPOS, MAXFUL, MAXNQN, MAXP,
MINNQN, NAK, NAKK, NH, NP, NRANG2, NW, RKSTO)
```

This routine locates continuum-continuum radial integrals. The parameter ICON determines the integral type:
=1 for $I(ab)$ integrals,
=2 for $F^k(ab)$ integrals,
=3 for $G^k(ab)$ integrals,
=4 for $R^k(abcd)$ integrals.

Routine FINM — function

```
(NCFR, NCFS, JA, JB, LAM, WA, IBBPOL, IRK1, ITC, IWRITE, LAG,
MAXNQN, NAK, NH, NP, RKSTO)
```

This function locates a bound-bound multipole integral. It is called by routine AIJZ in the evaluation of the asymptotic coefficients.

Routine FINMBB

```
(WA, JA, JB, LAM, FINTL, FINTV, IBBPOL, IRK1, IWRITE, ITC,
LAG, MAXNQN, NAK, NH, NP, RKSTO)
```

This routine locates bound-bound multipole integrals. It is called by routine DMEL in the evaluation of the dipole matrix elements.

Routine FINMBC

```
(WA, WB, K2, JA, LAM, FINTL, FINTV, IBCPOL, IRK1, ITC, IWRITE,
LAG, NAK, NH, NP, NRANG2, RKSTO)
```

This routine locates bound-continuum multipole integrals. It is called by routine DMEL in the evaluation of the dipole matrix elements.

Routine FINMCC

```
(WA, WB, KI, KJ, LAM, FINTL, FINTV, ICCPOL, IRK1, ITC, IWRITE,
NRANG2, RKSTO)
```

This routine locates continuum-continuum multipole integrals. It is called by routine DMEL in the evaluation of the dipole matrix elements.

Routine FIXJ

```
(JA1, JA2, KA, IS, KS, NS, KJ23, J1, J2, J3, MMOM, NMOM,
JBQ2, JJC1, JJC2, JJQ1, JJQ2, JLIST)
```

This routine sets up the arrays J1, J2, J3 required by the recoupling package.

Routine GENCOR

```
(NMAN, NWM, NCFST, JF, NPTY, CORINC, ICHOP, IDMTST, IPOS,
IQ, ISPAR, ITAB, ITJPO, IWRITE, JCHOP, JCUP, JQS, JTAB,
NAK, NCF, NDIMAX, NLX, NQX, NROWS, NTAB, NW)
```

This routine generates correlation functions.

Routine IROW1 — function

```
(NELC, KSI, IWRITE)
```

This function locates the row position of CSF j^n in table NTAB.

Routine LOCBB

```
(K1, K2, K1S, K2S, RKSTO, ICTXB, IRK1, IRK2, ISTX1B,
ISTX2B, ISTXB, ITC, IWRITE, KBBN1, KBBN2, KBBR, KBMAX)
```

This routine locates the bound-bound radial integral blocks.

Routine LOCBC

```
(IDISC2, K1, K2, K1S, K2S, KPOS, MAXP, RKSTO, ICTX, ISTX1,
ISTX2, IRK1, IRK2, ITC, IWRITE, KBCN1, KBCN2, KBCR, KBMAX,
KCMAX, KCMIN)
```

This routine locates the bound-continuum radial integral blocks from the scratch dataset IDISC2.

Routine LOCCC

```
(IDISC2, K1, K2, K1S, K2S, KPOS, MAXP, RKSTO, ICTX, ICTY,
IRK1, ITC, IWRITE, KCCN, KCCR, KBMAX, KCMAX, KCMIN)
```

This routine locates the continuum-continuum radial integral blocks from the scratch dataset IDISC2.

Routine LOCM

```
(K1, K2, K1S, K2S, RKSTO, IBBPOL, IBCPOL, ICCPOL, IRK1,
ITC, IWRITE, KMULTN, KMULTR, KBMAX, KCMAX, LAMBB, LAMBC, LAMCC)
```

This routine locates the multipole radial integral blocks.

Routine LTAB

```
(IS, NQS, KS, IROWS, IWRITE)
```

This routine locates rows of possible parents of active shell states for accessing array NTAB. It is assumed that empty shells have been eliminated from consideration by routine RKCO.

Routine MAN5

```
(IWRITE, KCHOP, IPAR, JMIN, JMAX, MFAIL, JSTORE, ICHOP, IQ,
ISPAR, ITAB, ITJPO, JCUP, JPOS, JQS, JTAB, MLX, MQX, NAK,
NCF, NROWS, NTAB, NW)
```

For a particular CSF this routine evaluates all the possible ways of coupling the open shells defined by MLX (l-values) and MQX (occupation numbers) to give the resultant total J value JF. The results are then used to fill arrays IQ, JQS, JCUP, ICHOP, ITJPO, ISPAR.

Routine MCPINB

```
(IDISC1, NXX, IDMTST, ISLDR, ITC, IWRITE, JSLDR, NDIMAX,
NH, NMCP, NNLDR, NP, NSLDF, NW1, XSLDR)
```

This routine reads any MCP coefficients which are required for the target Hamiltonians.

Routine MCPINC

```
(IDISC1, IDISC4, NXX, IDMTST, ISLDR, ITC, IWRITE, J2P1,
JLASTX, JSLDR, NCCK, NCFCON, NCFGP, NDIMAX, NH, NMCP,
NNLDR, NP, NSLDF, NW, NW1, XSLDR)
```

This routine reads any MCP coefficients which are required for the continuum Hamiltonians.

Routine MODJ23

```
( J2, J3, J2S, J3S, NMOM)
```

This routine restores the arrays J2 and J3 from saved values in the arrays J2S and J3S. This is for the exchange case with elements J3(1,3) and J2(1,1) interchanged.

Routine MUMDAD

```
(IBUG2, IS, IWRITE, JBQ1, JBQ2, JJQ1, JJQ2, JTQ1,
JTQ2, KAPS, NQ1, NQ2, X)
```

This routine evaluates the product of 4 CFPs.

Routine OCON — function

```
(IA1, IB1, IA2, IB2, NQ1, NQ2)
```

This function evaluates the multiplicative statistical factor. It is assumed that states are ordered so that IA1≤IB1 and IA2≤IB2.

Routine PLACE

```
(MODE, N1, K1, N2, K2, IS, MAXFUL, MAXNQN, MINNQN)
```

This routine finds the position of continuum-continuum two-electron integrals within a given block.

Routine PLACE1

```
(N1P, K1, N2P, K2, N3P, K3, IS, MAXFUL, MAXNQN, MINNQN)
```

This routine finds the position of bound-continuum two-electron integrals within a given block.

Routine PLACE2

```
(N1P, K1, N2P, IS, MAXFUL, MINNQN)
```

This routine finds the position of bound-bound one-electron integrals.

Routine PLACE3

```
(N1P, K1, N2P, K2, N3P, K3, N4P, K4, IS, MAXFUL, MAXNQN, MINNQN)
```

This routine finds the position of bound-bound two-electron integrals.

Routine PROP

```
(EIG, XMT, FACTAN, FACTCM, FACTEV, ISPAR, ITC, ITJPO,
IWRITE, NAST, NCF)
```

This routine writes out the target energy levels and eigenvectors.

Routine READ2

```
(IBUG1, IBUG2, IBUG3, IBUG4, IBUG5, IBUG6, ICHOP,
IDMTST, IHED, IPOLPH, IPOS, IQ, IREAD, ISPAR, ITAB, ITC,
ITJPO, IWRITE, JCHOP, JCUP, JQS, JTAB, NAK, NAST, NCF, NDIMAX,
NH, NLX, NMAN, NP, NQX, NROWS, NTAB, NW, NWM)
```

This routine reads the input data.

Calling tree:

```
READ2
    DATNR
        DMCHK2
            DMPRT2
        MAN5
            COUP
        REORDR
    DATNRP
    DMCHK2
        DMPRT2
```

Routine READ2C

```
(MORE, ICHOP, IDMTST, IPOS, IQ, IREAD, ISPAR, ITAB,
ITJPO, IWRITE, J2P1, JCHOP, JCUP, JQS, JTAB, NAK, NCF,
NCFGP, NDIMAX, NLX, NMAN, NPTY, NQX, NROWS, NTAB, NW, NWM)
```

This routine reads the input data defining the $(N + 1)$-electron symmetry. This is input record 6.

Calling tree:

```
READ2C
    DMCHK2
        DMPRT2
    GENCOR
        DMCHK2
            DMPRT2
        MAN5
            COUP
```

Routine REORDR

```
(NST, ICHOP, IQ, ITJPO, JCUP, JQS, NCF, NW)
```

This routine sorts CSFs into the order of descending total J. A linked list sorting method is used.

Routine RKCO

(JA, JB, NH, IBUG1, IBUG2, IBUG3, ICHOP, IEXC, IQ, IME, ISPAR,
ITJPO, IWRITE, JCUP, JQS, NAK, NCF, NOUTX, NP, NW, NWA, ITAB,
JTAB, NTAB)

This routine analyses the tables of quantum numbers to determine all possible sets of interacting orbitals which give a non-vanishing Coulomb matrix element, and initiates the calculation of coefficients.

Routine RKCO1

(IDQ, JA1, JB1, JA2, JB2, JLIST, NPEEL, NQ1, NQ2)

This routine is called by RKCO.

Routine RKCO2

(KWA, JT1, JT2, IBUG1, IBUG2, IBUG3, IEXC, IME, IWRITE,
JA, JB, NAK, NH, NOUTX, NP, NW, NWA, ITAB, JTAB, NTAB, JJC1,
JJC2, JJQ1, JJQ2, JLIST, NPEEL, NQ1, NQ2)

This routine is called by RKCO.

Routine RKCO3

(JT1, JT2, JLIS, JC1S, JC2S, IBUG1, IBUG2, IBUG3, IEXC,
IME, IWRITE, JA, JB, NAK, NH, NOUTX, NP, NW, NWA, ITAB, JTAB, NTAB,
JJC1, JJC2, JJQ1, JJQ2, JLIST, KLIST, NCORE, NPEEL, NQ1, NQ2)

This routine is called by RKCO.

Routine SETJ

(IS, JBQ1, JJC1, JJC2, JJQ1, JJQ2, JLIST, JS, JTQ1,
JTQ2, KS, NS, J1, J2, J2S, J3, J3S, KJ23, MMOM, NMOM)

This routine sets the arrays required by the recoupling coefficient package, for example NJSYM written by Burke (1969 [18]).

The variables MMOM and NMOM and arrays J1, J2 and J3 are set.

- MMOM
  is the total number of angular momentum values in the initial and final states.

- NMOM
  is the number of basic angular momentum values that are coupled.

- J1 array
  (J1(I),I=1,MMOM) the angular momentum values stored as $2J + 1$.

- J2 array
  ((J2(I,J),I=1,(NMOM−1)),J=1,3) gives the position in the J1 array of the initial state triads.

- J3 array
  ((J3(I,J),I=1,(NMOM−1)),J=1,3) gives the position in the J1 array of the final state triads.

These are stored for the first call of NJSYM involving direct integrals. Subsequent exchange calls of NJSYM must be preceded by a call of MODJ23 to restore the arrays `J2` and `J3` to their correct initial state.

Routine SETUP

```
(IBUG2, IWRITE, JA, JB, ICHOP, IQ, JCUP, JQS, NCF, NW,
JJC1, JJC2, JJQ1, JJQ2, JLIST, KLIST, NCORE, NPEEL, NQ1, NQ2)
```

This routine generates the arrays defining the quantum numbers of the states involved in the matrix element linking CSFs labelled by JA, JB.

Routine SKRC

```
(IS, KAPS, KS, KD1, KD2, KE1, KE2)
```

This routine determines the range of the tensor rank K for the two-electron Coulomb integrals.

Routine SMCP

```
(NOUT, ISMCP, IBUG1, IBUG2, IBUG3, IBUG4, IBUG5, IBUG6,
ICHOP, IEXC, IQ, ISPAR, ITJPO, IWRITE, JCUP, JQS, NAK,
NCF, NH, NP, NW, ITAB, JTAB, NROWS, NTAB)
```

This routine evaluates two-electron angular coefficients required for the Dirac Hamiltonian. This version also evaluates one-electron coefficients.

Calling tree:

```
SMCP
    RKCO
        COR       ---- see routine
        RKCO1
        RKCO2
            COR   ---- see routine
        RKCO3
            COR   ---- see routine
        SETUP
        VIJOUT
    SPEAK
    TMSOUT
    TNSRJJ        ---- see routine
```

Routine SNRC

```
(IS, KAPS, KS, ID1, ID2, NE1, NE2, IBRD, IBRE)
```

This routine sets limits for the order of the Breit interaction coefficients.

Routine SPEAK

```
(ITYPE, IA, IB, IC, ID, K, COEF, IBUG1, IME, IWRITE, JA,
JB, NOUTX, NH, NP, NWA)
```

This routine outputs MCP coefficients to the file on stream NOUTX if NOUTX > 0. Also it prints them if IBUG1=1.

ITYPE=7: one-electron integrals
ITYPE=8: direct two-electron integrals
ITYPE=9: exchange two-electron integrals

Routine TMSOUT
`(IWRITE, ITAB, JTAB, NROWS, NTAB)`

This routine prints a table of terms set by BLDST2.

Routine TNSRJJ
`(KA, IOPAR, JA, JB, IA1, IA2, VSHELL, IBUG2, IBUG3, IBUG4, IBUG6, ICHOP, IQ, ISPAR, ITJPO, IWRITE, JCUP, JQS, NAK, NCF, NW, ITAB, JTAB, NTAB)`

This routine evaluates the reduced matrix elements of a one particle operator for CSFs in jj-coupling. Variable KA is the operator rank and variable IOPAR is the parity.

Calling tree:

```
TNSRJJ
    CFP      ---- see routine
    FIXJ
    GENSUM   ---- see routine
    IROW1
    NJGRAF   ---- see routine
    SETUP
    VIJOUT
```

Routine VIJOUT
`(IWRITE, JA, JB, JJC1, JJC2, JJQ1, JJQ2, JLIST, NCORE, NQ1, NQ2, NPEEL)`

This routine prints out tables of CSF quantum numbers defined by routine SETUP for the current matrix element.

Routine WRITP2
`(ITAPE1, ITAPE2, IDISC2, RECORD, IHED, IBBPOL, IBCPOL, ICCPOL, ICTXB, IDMTST, IPOLPH, ISTX1B, ISTX2B, ISTXB, IWRITE, JAG, K1S, K2S, KAG, KBBN1, KBBN2, KBBR, KBCN1, KBCN2, KBCR, KBMAX, KCCN, KCCR, KCMAX, KCMIN, KMULTN, KMULTR, KPOS, KX, LAG, LAMBB, LAMBC, LAMCC, MAXFUL, MAXNQN, MAXP, MINNQN, NDIMAX, NRANG2)`

This routine reads the file, `DSTG1.DAT`, created by module DSTG1.

- The basic data, continuum orbital data (eigenenergies and amplitudes) and Buttle corrections are read from file `DSTG1.DAT`.
- This data is written to file `DSTG2.DAT`, so that the first part of the two files is identical.
- The remainder of the file `DSTG1.DAT` is read. This data contains indexing information for the radial integrals.

- The arrays for the multipole integrals (IBBPOL, IBCPOL and ICCPOL) are held in memory. New variables KMULTN and KMULTR hold IRK1 (number of radial integrals) and IREC (DA record number in `INTEGRAL.DAT`).

- The arrays for the continuum-continuum integrals (ICTX and ICTY) are transferred to scratch file IDISC2. New arrays KCCN and KCCR hold IRK1 (number of radial integrals) and IREC (DA record number in `INTEGRAL.DAT`).

- The arrays for the bound-continuum integrals (ICTX, ISTX1 and ISTX2) are transferred to scratch file IDISC2. New arrays KBCN1, KBCN2 and KBCR hold IRK1 (number of radial integrals), IRK2 and IREC (DA record number in `INTEGRAL.DAT`).

- The arrays for the bound-bound integrals (ICTXB, ISTXB, ISTX1B and ISTX2B) are held in memory. New arrays KBBN1, KBBN2 and KBBR hold IRK1 (number of radial integrals), IRK2 and IREC (DA record number in `INTEGRAL.DAT`).

- Parameters K1S, K2S, MAXP and KPOS are initialised. These are used in the LOC... routines.

Chapter 7

DSTGH

7.1 Introduction

$Source: /home/phn/DARC/RCS/dstgh.mak,v
$Author: phn
$Date: 2001/10/31 13:09:12
$Revision: 11.1

DARC, the Dirac Atomic R-matrix Codes.
homepage: http://www.am.qub.ac.uk/DARC/
DSTGH module

DSTGH diagonalises the continuum Hamiltonians. The code outline is given below.

Calling tree:

```
DSTGH
    AAMNH
        QUARTZ  timing routine
        CALEN   access date/time at execution
        DMSETH  set dimensions

        READH   reads input data
        RDDMPH  reads Hamiltonian
        RSCTH   diagonalises Hamiltonian

        DMPRTH  final print of dimensions used
```

DSTGH reads the DSTG2 dump, `DSTG2.DAT`, for a particular $(N+1)$-electron symmetry and diagonalises the corresponding continuum Hamiltonian. For large cases (i.e. , many channels) this part of the calculation is the most time consuming and can be a computational bottle-neck. In addition there is the handling of a very large matrix.

The code can adjust the target energy levels if required. This procedure should be used with caution. It is appropriate when the energy shifts are small. If they are large then really you should reconsider the target description.

The eigenvalues (the R-matrix poles) and the surface amplitudes are written to the file `DSTGH.DAT`. The surface amplitudes require less space than the eigenvectors. However for bound-state or photo-ionisation calculations you would require the eigenvectors. By setting an option (33) the output file is formatted and is called `DSTGHF.DAT`. This can easily be transferred between different machines.

A module called DUMPH can be used to examine and combine DSTGH dump files. This module will also convert between formatted and unformatted and vice-versa as requested. Input to DUMPH is interactive.

7.2 Matrix diagonalisation: LAPACK or HSLDR

There are two versions of the module which use different amounts of memory.

1. The first version stores the full continuum Hamiltonian matrix and uses the LAPACK library routine DSYEV. An advantage of LAPACK is that there may be optimised libraries available.

2. The second version stores only the upper triangle including diagonal and uses the HSLDR package. When the dimension of the continuum Hamiltonian is very large, memory constraints may require use HSLDR. However this is generally slower in terms of CPU time.

You select between the two versions by preprocessing. Lines beginning with {LAPACK} refer to the LAPACK version while lines beginning with {HSLDR} refer to the HSLDR version.

7.3 Files

The files are defined internally. Messages are also written to output stream 6 using the PRINT command. The following files are used:

stream name	description	stream number	file name	FORM
IREAD	input data	5	`DSTGH.INP`	Formatted
IWRITE	printed output	7	`DSTGH.OUT`	Formatted
IO1	DSTG2 dump file	11	`DSTG2.DAT`	Unformatted
IO2	DSTGH dump file	12	`DSTGH.DAT`	Unformatted
			`DSTGHF.DAT`	Formatted (option 33 set)

7.4 Input data (`DSTGH.INP`)

The input data uses Namelist format. Each Namelist has a name. If this is say DSTGH then the input data starts with the string &DSTGH. There must be a space before the & even if it is at the beginning of the line. The end of this Namelist is denoted by &END, with & again preceded by a space. Variables in a Namelist can be given default values. The default value is indicated in brackets after the variable name.

Input record 1

Read the 80-character title, variable IHED.

Input record 2

Namelist DSTGH ...

```
INTEGER NAST
DOUBLE PRECISION XNAT(MXNL)
INTEGER OPT(50)
NAMELIST /DSTGH/ NAST,XNAT,OPT
```

- NAST [−1]
  is the number of target states. NAST must agree with the value on the DSTG2 dump.

  Only required if either option 24 or 25 is set.

- XNAT [0]
  is an array containing the target energies in au.

  Only required if either option 24 or 25 is set.

  These are used to adjust the diagonal elements of the Hamiltonian (in routine RDHAM) and as excitation thresholds in the collision problem.

  These are corrected energies for the target states. It is assumed that the levels are in the same order as given in DSTG2 (i.e. have the same assignment). Therefore it may happen that the input energies do not increase monotonically indicating that a reordering is needed. The module will attempt to do this unless option 20 is set.

- OPT [0] – array

  Input a list of integers giving the options to be set. An option is a switch that is either on or off. To set option J input J on this input record. The options are stored in array ITC. If option J is set then ITC(J)=1; otherwise ITC(J)=0.

  Operational options.

  - 20
    When energy levels are input to DSTGH (i.e. either option 24 or 25 is set) then the new energies may be incorrectly ordered (i.e. do not increase monotonically). The default action of the program is to reorder the data. This option switches off this ordering.
  - 24
    When this option is set input target energies are used to adjust the diagonal elements of the continuum Hamiltonian.
  - 25
    The diagonal elements of the Hamiltonian are adjusted as in option 24 but keeping the trace of the continuum-continuum Hamiltonian unchanged. Only the excitation energies are required for this adjustment.
  - 33
    The DSTGH dump is formatted and called `DSTGHF.DAT`. Default: the DSTGH dump is unformatted and called `DSTGH.DAT`.

  Printing options.

  - 5
    Set this option to print the Hamiltonian to be diagonalised. This may be different from the Hamiltonian on the DSTG2 dump because of the adjustment of the diagonal elements when either option 24 or 25 is set.
  - 7
    Set this option to print the eigenvalues and eigenvectors of the Hamiltonian.

  - 8
    Set this option to print the surface amplitudes.
  - 9
    Set this option to examine the eigenvectors of the Hamiltonian.
  - 10
    Set this option to print the Hamiltonian eigenvalues in au.
  - 11
    Set this option to print the Hamiltonian eigenvalues relative to the ground-state in au.
  - 12
    Set this option to print the Hamiltonian eigenvalues relative to the ground-state in Ryd.
  - 13
    Set this option to print the Hamiltonian eigenvalues relative to the ground-state in eV.

Input record 3

This input record specifies the $(N + 1)$-electron symmetries to be considered.

Namelist SYM ...

```
DOUBLE PRECISION JTOT
INTEGER NPTY
INTEGER NUMSYM
NAMELIST /SYM/ JTOT,NPTY,NUMSYM
```

- JTOT [-1.0]
  is the J-value (total angular momentum) of the $(N + 1)$-electron symmetry. This is a real number such as .5 or 1. or 1.5 etc. It is used to form the integer variable J2P1=2∗JTOT+1.
- NPTY [0]
  is the Π-value (parity) of the $(N + 1)$-electron symmetry (-1 for odd, 1 for even).
- NUMSYM [-1]
  - > 0 ... number of symmetries to be used. The first NUMSYM symmetries on the DSTG2 dump are used.
  - $= 0$ ... no symmetries are processed
  - $= -1$ ... all symmetries on the DSTG2 dump are used

Input can be ended by setting NUMSYM to 0. If the code cannot find the SYM input record then NUMSYM is set to 0 and the input data is terminated. If JTOT (≥ 0) is specified on the input record then NUMSYM is set to -2.

e.g.

- all symmetries are to be processed (takes the default for NUMSYM)
  &SYM &END
- the first 5 symmetries are to be processed
  &SYM NUMSYM=5 &END
- two specified symmetries are to be processed e.g. $\frac{1}{2}^e$ and $\frac{3}{2}^e$
  &SYM JTOT=.5 NPTY=1 &END
  &SYM JTOT=1.5 NPTY=1 &END
- no symmetries are to be processed, leave out SYM input record or use
  &SYM NUMSYM=0 &END

7.5 DSTGH dump file (DSTGH.DAT or DSTGHF.DAT)

If option 33 is not set then the file is unformatted with name DSTGH.DAT.

If option 33 is set then the file is formatted with name DSTGHF.DAT.

The form of the output dump depends on the parameters IPOL2 and MORE2. If the DSTG2 dump is marked as containing dipole matrix elements then IPOL2=2 otherwise IPOL2=1.

For the case IPOL2=2 the parameter MORE2 is set to 1 for the first symmetry and to 2 for any following. MORE2=1 when IPOL2=1.

Data that is independent of the symmetry

```
 1. IHED,RECORD
 2. NELC,NZ,NRANG2,KCMAX,NAST,RA,BSTO
 3. (ENAT(I),I=1,NAST)
 4. (JAT(I),I=1,NAST)
```

Repeat records 5 to 7 for L=1,KCMAX

```
 5. MBUT(L)
 6. (EBUT(L,I),I=1,MBUT(L))
 7. (CBUT(L,I),I=1,MBUT(L))
```

Data for each symmetry

Repeat records 8 to 13 for each symmetry to be treated.

If IPOL2=2 then set J2P1X=−J2P1 otherwise J2P1X=J2P1

```
 8. J2P1X,NPTY,NCHAN,MNP1,NCFGP,IKMAX
 9. (NCONAT(I),I=1,NAST)
10. (K2P(I),I=1,NCHAN)
```

If IKMAX > 0 then write record 11

```
11. (((CF(I,J,M),I=1,NCHAN),J=1,NCHAN),M=1,IKMAX)
```

If MORE2=2 then call DMECP to transfer the dipole matrix elements from the DSTG2 dump to the DSTGH dump.

If IPOL2>1 then repeat record 12a for NOO=1,MNP1

```
12a. AKS(NOO),(X(I),I=1,MNP1)
```

If IPOL2=1 then write record 12b

```
12b. (AKS(I),I=1,MNP1)

13. ((WMAT(K,I),K=1,NCHAN),I=1,MNP1)
```

Formats for the formatted file

```
 1.  using format 4001
 2.  using format 4002
 3.  using format 4003
 4.  using format 4004
 5.  using format 4004
 6.  using format 4003
 7.  using format 4003
 8.  using format 4004
 9.  using format 4004
10.  using format 4004
11.  using format 4003
12a. using format 4003
12b. using format 4003
13.  using format 4003
```

The formats are:

```
4001 FORMAT(A40/A40/A20)
4002 FORMAT(5I14/1P,2E15.8)
4003 FORMAT(1P,5E15.8)
4004 FORMAT(5I14)
```

Description of data

- IHED
  is the 80-character title for the calculation.

- RECORD
  is a 20-character record of time/date at execution. It is obtained by a call to the system dependent routine CALEN.

- NELC
  is the number of electrons in the target.

- NZ
  is the atomic number of the target.

- NRANG2
  is the number of continuum orbitals for each K-value.

- KCMAX
  is the maximum continuum K-value.

- NAST
  is the number of target states.

- RA
  is the R-matrix boundary radius.

- BSTO
  is the constant in the boundary condition at RA for the continuum orbitals (corresponds to the nonrelativistic logarithmic derivative)

  $$\frac{2caQ(a)}{P(a)} = b + \kappa \tag{7.1}$$

  where a (variable RA) is the R-matrix boundary and b (variable BSTO) is a constant.

- ENAT(I) I=1,NAST
  is an array of target energies in au, ordered by increasing energy.

- JAT(I) I=1,NAST
  is an array of $2J+1$ angular momenta multiplied by the parity ($+1$ even, -1 odd) for the target states. The states are in the same order as array ENAT (target energies).

- MBUT(K) K=KCMIN,KCMAX
  is an array giving the number of Buttle corrections stored for each continuum K-value, K.

- CBUT(K,I) I=1,MBUT(K) K=KCMIN,KCMAX
  is an array of Buttle corrections (MBUT(K) in number) for each continuum K-value K, tabulated as a function of energy (array EBUT).

- EBUT(K,I) I=1,MBUT(K) K=KCMIN,KCMAX
  is an array of Buttle energies in au (MBUT(K) in number) for each continuum K-value K.

- J2P1X
  J2P1 is $2J + 1$ for the $(N + 1)$-electron symmetry where J is the total angular momentum. If IPOL2=2 then set J2P1X=−J2P1 otherwise J2P1X=J2P1.

- NPTY
  is the parity (+1 even, −1 odd) for the $(N + 1)$-electron symmetry.

- NCHAN
  is the number of channels.

- MNP1
  is the dimension of the continuum Hamiltonian. It is NRANG2∗NCHAN+NCFGP.

- NCFGP
  is the number of correlation functions. This is the number of $(N + 1)$-electron states formed from the bound orbitals only.

- IKMAX is the maximum λ for multipoles.

- NCONAT(I) I=1,NAST
  gives the number of channels for target state I.

- K2P(I) I=1,NCHAN
  gives the κ-value for each channel I.

- CF(I,J,M) I=1,NCHAN J=1,NCHAN M=1,IKMAX
  are the coefficients in the multipole potentials.

- AKS(I) I=1,MNP1
  are the continuum Hamiltonian eigenvalues. Units are a.u.. These are in decreasing order.

- X(I) I=1,MNP1
  are the continuum Hamiltonian eigenvectors.

- WMAT(K,I) K=1,NCHAN I=1,MNP1
  are the surface amplitudes.

7.6 Routines

Routine DSTGH

This is the PROGRAM routine. The input and output files are set up: IREAD (`DSTGH.INP`) and IWRITE (`DSTGH.OUT`). A call is made to routine AAMNH which controls the computation.

Routine AAMNH
`(IREAD, IWRITE)`

This is the main controlling routine for the DSTGH module.

Routine DMCHKH
`(I, ID, IWRITE, IDMTST, NDIMAX)`

This routine is called to test array overflow.

Routine DMECP
`(IO1, IO2, IWRITE, ITC1)`

This routine transfers the dipole matrix elements to the output stream IO2.

Routine DMPRTH
`(IWRITE, IDMTST, NDIMAX)`

This routine writes out information on the dimensions used.

Routine DMSETH
`(IWRITE, IDMTST, NDIMAX)`

This routine initialises the arrays (IDMTST, maxima set, and NDIMAX, maxima used) that check on dimensions. Dimensions are set by the file `darc.inc` which is included in the relevant routines. All dimension variables beginning with MX are set in this file. To change a dimension you need only change `darc.inc`.

Routine PRTHAM
`(IWRITE, HAMIL, ND)`

This routine prints a symmetric matrix whose upper triangle is stored in an array.

Routine RDDMPH

```
(IO1, IO2, IWRITE, IDMTST, NDIMAX, HAMIL, RECORD, IHED,
BSTO, CF, CL, ENAT, ENDS, GROUND, RA, XNAT, IKMAX, IORD,
IPOL2, ITC, J2P1, JAT, K2P, LORD, MNP1, MORE2, NAST,
NCASES, NCFGP, NCHAN, NCONAT, NPTY, NRANG2, NUMSYM)
```

This routine reads the data from DSTG2 corresponding to the required values of J2P1 and NPTY.

Routine RDHAM

```
(IO1, IWRITE, NORD, ENAT, GROUND, HAMIL, XNAT, ITC,
MNP1, NAST, NCFGP, NCHAN, NCONAT, NRANG2)
```

This routine reads in the continuum Hamiltonian. It is stored in the array HAMIL. The complete Hamiltonian is stored when LAPACK is used. The upper triangle is stored when HSLDR is used.

Routine READH

```
(IREAD, IWRITE, IHED, RECORD, XNAT, ITC, J2P1,NAST,
NCASES, NPTY, NUMSYM)
```

This routine reads the input data.

Routine RSCTH

```
(IO1, IO2, IWRITE, AKS, CF, ENDS, HAMIL, WMAT, X, XNAT,
IKMAX, IPOL2, ITC, J2P1, K2P, MNP1, MORE2, NAST, NCFGP,
NCHAN, NCONAT, NPTY, NRANG2)
```

This routine sets up and diagonalises the Hamiltonian matrix and determines the surface amplitudes from the eigenvectors.

Chapter 8

DSTG3

8.1 Introduction

$Source: /home/phn/DARC/RCS/dstg3.mak,v
$Author: phn
$Date: 2001/10/31 13:31:23
$Revision: 11.1

DARC, the Dirac Atomic R-matrix Codes.
homepage: http://www.am.qub.ac.uk/DARC/
DSTG3 module

DSTG3 (i.e. , DASYM or DASYPCK or DCOUL or DVPM) is designed for electron scattering only. It solves the external region scattering equations. The DSTGH dump is read and asymptotic solutions, K-matrix and collision strengths are calculated.

Calling tree:

```
DSTG3
    AAMN3


        QUARTZ  timing routine
        CALEN   access date/time at execution
        DMSET3  set dimensions

        READ3   read input data

        RDDUMP  read the DSTGH file for a symmetry
        RSCT    loop over the incident energy

        XSUM    output results summed over symmetry

        DMPRT3  final print of dimensions used
```

DSTG3 can be interfaced to four asymptotic codes at present. The resulting modules require preprocessing to select the relevant asymptotic code.

DASYM module

DASYM is the relativistic program that was developed by Young (Young 1992 [66], Young and Norrington 1994 [67]). It uses DE/STEP (Shampine and Gordon 1975 [60]) to integrate.

DASYPCK module

DASYPCK is a nonrelativistic program. It is based on the ASYPCK2 package developed by Crees (1980 [31], 1981 [32]). The module uses the full version of ASYPCK2.

DCOUL module

DCOUL is a nonrelativistic program for ion targets. It neglects channel coupling and uses Coulomb solutions. Closed channel solutions are obtained from the COUL program of

Bell and Scott (1980 [8]). Open channel solutions use the COULFG program of Barnett (1982 [5]). This version of COULFG has been taken from the CFASYM program of Noble and Nesbet (1984 [46]).

DVPM module

DVPM is a nonrelativistic program for neutral targets. It uses the variable phase method and is an adaption of VPM which was developed by Croskery et al (1982 [33]).

8.2 Files

The files are defined internally. Messages are also written to output stream 6 using the PRINT command.

The following files are used:

stream name	description	stream number	file name	FORM
IREAD	Input data.	5	`DSTG3.INP`	Formatted
IWRITE	Output.	7	`DSTG3.OUT`	Formatted
—	Scratch file for input data.	9	—	Formatted
ITAPE1	DSTGH file.	11	`DSTGH.DAT`	Unformatted
			`DSTGHF.DAT`	Formatted
ITAPE2	Ω-file.	12	`DSTG3.DAT`	Formatted
ITAPE4	Phase shifts, option 4 set.	14	`PHASE.SH`	Formatted
ITAPE5	K-matrices, option 7 set.	15	`KMAT.DAT`	Formatted

8.3 Input data (`DSTG3.INP`)

The input data uses Namelist format. Each Namelist has a name. If this is say DSTG3 then the input data starts with the string &DSTG3. There must be a space before the & even if it is at the beginning of the line. The end of this Namelist is denoted by &END, with & again preceded by a space. Variables in a Namelist can be given default values. The default value is indicated in brackets after the variable name.

Input record 1

Read the 80-character title, variable IHED.

Input record 2

Namelist DSTG3 ...

```
INTEGER NENRGS
INTEGER OPT(50)
NAMELIST / DSTG3/ NENRGS,OPT
```

- NENRGS [0]
  is the number of energy ranges (compulsory). The maximum value is set in the module by parameter NDENER.

- OPT [0] – array

  Input a list of integers giving the options to be set. An option is a switch that is either on or off. To set option J input J on this input record. The options are stored in array ITC. If option J is set then ITC(J)=1; otherwise ITC(J)=0.

  Operational options.

  - 3
    Calculate and print the eigenphases in routine XSECN. The eigenphases lie in the range 0 to π.
  - 4
    When this option is specified the eigenphase sum is calculated in routine XSECN and written to file `PHASE.SH`. The individual eigenphases are printed only when option 3 is set. The eigenphases lie in the range 0 to π.
  - 5
    Set the off-diagonal asymptotic scattering coefficients to zero. This corresponds to the neglect of inter-channel coupling.
  - 6
    Set all the asymptotic scattering coefficients to zero. This corresponds to the Coulomb case.
  - 7
    When this option is specified the K-matrices are written to file `KMAT.DAT`.
  - 8
    The input energies are in eV rather than the default of Ryd.
  - 50
    This option is set internally and indicates whether the DSTGH dump is formatted or unformatted.

  Printing options.

  - 1
    This is a useful debug print that gives the Hamiltonian eigenvalues, total energy, channel energies and momenta, Buttle corrections, R-matrix and K-matrix .
  - 2
    Print the asymptotic scattering coefficients.

There will be no printed results from DSTG3 unless one of the options 14, 15, 16, or 17 is set. The collision strengths are written to the Ω-file `DSTG3.DAT`. If the file is saved then it is possible at a later date to examine them using module DSTG4.

- 14
  Print the cross-section calculated in routine XSECN.
- 15
  Print the collision strength calculated in routine XSECN.
- 16
  At the end of the calculation call routine XSUM and print only the total results. The default is no call to XSUM.
- 17
  At the end of the calculation call routine XSUM and print both the partial and total results. The default is no call to XSUM.
- 18
  In routine XSUM print out cross-sections. The default is collision strengths.
- 49
  Print out the asymptotic solutions from routine CFACE (i.e. DCOUL package), routine DFACE (i.e. DASYM package) or routine AFACE (i.e. DASYPCK package). This option also gives a debug print from routine VFACE (i.e. DVPM package).

The next 8 options are debug options used in DASYM. Options 43, 47 and 48 are most useful. The others can give a lot of output.

- 41
  Debug print from routine CONRAD. This gives the channel convergence radii against the number of terms.
- 42
  Debug print from routine CONRAD. This gives the channel convergence radius.
- 43
  Debug print from routine INTNR. This gives information on the inward radial integration.
- 44
  Debug print from routines BSDEXP and GAIEXP. This gives information on the asymptotic expansions.
- 45
  Debug print from routines CFEVAL and DFEVAL. This gives information on the evaluation of the accelerated series.
- 46
  Debug print from routines GAISOL and BSDSL1/2. This gives information on the asymptotic expansions.
- 47
  Debug print from routines GAISOL and BSDSL. This gives the asymptotic solutions and error estimates.
- 48
  Debug print from routine WRONSK. This prints the wronskian.

Input record 3

Namelist ASYM ...
Input data for the asymptotic packages.

- **DASYM package**

```
{DASYM}     DOUBLE PRECISION DEGENY
{DASYM}     DOUBLE PRECISION EPS
{DASYM}     INTEGER IASY
{DASYM}     INTEGER IGAIL
{DASYM}     INTEGER LAMAX
{DASYM}     INTEGER NONREL
{DASYM}     NAMELIST /ASYM/ DEGENY,EPS,IASY,IGAIL,LAMAX,NONREL
```

  - DEGENY [1D−3]
    is the parameter used to determine degenerate channels
  - EPS [1D−4]
    is the accuracy parameter.
  - IASY [30]
    is the number of terms in the series expansion. You can use between 10 and 60 terms, try 20 to 30.
  - IGAIL [0]
    =0 Burke and Schey expansion
    =1 Gailitis expansion
    =2 no coupling
  - LAMAX [4]
    is the maximum multipole in the coupling (>0)
  - NONREL [0]
    =0 relativistic solutions
    =1 nonrelativistic solutions

- **DASYPCK package**

```
{DASYPCK}     DOUBLE PRECISION DR
{DASYPCK}     DOUBLE PRECISION EROR
{DASYPCK}     INTEGER IAUTO
{DASYPCK}     INTEGER IBUG
{DASYPCK}     INTEGER MFG
{DASYPCK}     NAMELIST /ASYM/ DR,EROR,IAUTO,IBUG,MFG
```

  - DR [.15]
    is the step length for Fox-Goodwin integration
  - EROR [1D−4]
    is the accuracy parameter for asymptotic solutions

- IAUTO [1]
  controls variation in MFG
  = 0 for using specified value of MFG
  = 1 for conditional automatic MFG
  = 2 for unconditional automatic MFG
- IBUG [0]
  is a debug parameter (=0 for no printed output from ASYPCK)
- MFG [200]
  is the number of steps with Fox-Goodwin

- **DCOUL package**

  No parameters are required for this package. But you still need the namelist card in the input data.

- **DVPM package**

```
{DVPM}     DOUBLE PRECISION ABSERR
{DVPM}     DOUBLE PRECISION ENDEG
{DVPM}     INTEGER IBSG
{DVPM}     NAMELIST /ASYM/ ABSERR,ENDEG,IBSG
```

  - ABSERR [1D−4]
    is an accuracy parameter used by routine ASYBSG to determine the convergence radius of the asymptotic expansions.
  - ENDEG [1D−3]
    is a parameter used by routine ASYBSG that assumes that two channels are degenerate if the absolute value of their energy difference is ≤ ENDEG.
  - IBSG [1]
    =0 for the Burke and Schey expansion
    =1 for the Gailitis expansion
    It is better to run VPM with the Gailitis expansion (IBSG=1) rather than with the Burke and Schey expansion (IBSG=0). The Gailitis expansion allows for the angular momentum of the scattered electron so that the radius at which VPM can match to the asymptotic expansion increases only very slowly with angular momentum. With Burke and Schey much larger radii may be needed at higher j-values. The Gailitis expansion works very well, except if the energy in a channel lies between threshold and about 0.001 Ryd above threshold. The Gailitis expansion coefficients are obtained from the Burke and Schey ones and the transformation needs more than a 64 bit word to keep enough accuracy at these very low energies.
    Thus use IBSG=0 close to threshold, and be prepared for some long integrations, and IBSG=1 elsewhere.

Input record 4

Namelist RANGE ...

```
DOUBLE PRECISION EINCR
DOUBLE PRECISION ESTART
INTEGER NUME
NAMELIST /RANGE/ EINCR,ESTART,NUME
```

Read in the incident electron energies in Ryd (option 8 set: in eV) relative to the ground-state of the target.

- EINCR [0]
  is the energy increment in the range.
- ESTART [0]
  is the first energy in the range.
- NUME [0]
  is the number of energy points in energy range. If NUME < 1 (default) then there is only one energy in the range, namely ESTART.

This input record is repeated for each of the NENRGS ranges.

Input record 5

This input record specifies the $(N + 1)$-electron symmetries to be considered.

Namelist SYM ...

```
DOUBLE PRECISION JTOT
INTEGER NPTY
INTEGER NUMSYM
NAMELIST /SYM/ JTOT,NPTY,NUMSYM
```

- JTOT [−1.0]
  is the J-value (total angular momentum) of the $(N + 1)$-electron symmetry. This is a real number such as .5 or 1. or 1.5 etc. It is used to form the integer variable J2P1=2∗JTOT+1.
- NPTY [0]
  is the Π-value (parity) of the $(N + 1)$-electron symmetry (-1 for odd, 1 for even).
- NUMSYM [−1]
  - > 0 ... number of symmetries to be used. The first NUMSYM symmetries on the DSTGH dump are used.
  - $= 0$ ... no symmetries are processed
  - $= -1$ ... all symmetries on the DSTGH dump are used

Input can be ended by setting NUMSYM to 0. If the code cannot find the SYM input record then NUMSYM is set to 0 and the input data is terminated. If JTOT (≥ 0) is specified on the input record then NUMSYM is set to -2.

e.g.

- all symmetries are to be processed (takes the default for NUMSYM)
  &SYM &END
- the first 5 symmetries are to be processed
  &SYM NUMSYM=5 &END
- two specified symmetries are to be processed e.g. $\frac{1}{2}^e$ and $\frac{3}{2}^e$
  &SYM JTOT=.5 NPTY=1 &END
  &SYM JTOT=1.5 NPTY=1 &END
- no symmetries are to be processed, leave out SYM input record or use
  &SYM NUMSYM=0 &END

8.4 Ω-file (DSTG3.DAT)

An Ω-file can be formatted or unformatted. The standard form of the Ω-file is given below.

```
record a. NZ,NELC,NAST,IREC,NRANG2
record b. (JAT(I),I=1,NAST)
record c. (ENAT(I),I=1,NAST)

record d. ISTORE,JSTORE,ESTORE, (OMEGA(I),I=1,IREC)
```

Records a,b,c are written first.

- NZ
  is the atomic number of the target.
- NELC
  is the number of electrons in the target.
- NAST
  is the number of target states.
- IREC
  is the number of collision strengths in array OMEGA.
  = NAST∗(NAST+1)/2 for neutrals
  = NAST∗(NAST−1)/2 for ions
- NRANG2
  is the number of continuum orbitals for each K-value.
- JAT(I) I=1,NAST
  is an array of $(2J + 1)$ angular momenta multiplied by the parity (+1 even, −1 odd) for the target states. The states are in the same order as array ENAT (target energies).

- ENAT(I) I=1,NAST
  is an array of target energies in au, ordered by increasing energy.

Record d is repeated for each symmetry/energy (JSTORE/ESTORE) combination calculated.

- ISTORE
  is a flag indicating the type of calculation. The following values are recognised in the code.
  - = −1, IPERT value taken from STGFJJ, this is set to ISTORE = 1 in routine XYZ of module DSTG4
  - = 0, IPERT value taken from STGFJJ, no perturbation corresponding to Coulomb
  - = 1, IPERT value taken from STGFJJ, with perturbation
  - = 10, DVPM calculation
  - = 30, DASYPCK calculation
  - = 40, DCOUL calculation
  - = 50, DASYM calculation

  You can use other values for ISTORE. The only place where these particular values are checked is in routine ABC0 of module DSTG4 when a summary of the results is printed.
- JSTORE
  is $\Pi(2J + 1)$ for the $(N + 1)$-electron symmetry, where J is the total angular momentum and Π is the parity (+1 even, −1 odd)
- ESTORE
  is the incident energy in Ryd.
- OMEGA
  is an array containing the collision strengths.

When the Ω-file is formatted then

- record a is written as FORMAT (5I14)
- record b is written as FORMAT (5I14)
- record c is written as FORMAT (5E15.8)
- record d is written as FORMAT (2I14,E15.8/(5E15.8))

Format 1P is used so that the real numbers have 9 digit accuracy.

The advantages of the unformatted file is an increase in accuracy (about 14 digits compared to 9) and smaller file size. The advantage of formatted file is that it is portable between machines. The file size disadvantage can be reduced by file compression using e.g. gzip.

8.5 K-matrix file (KMAT.DAT)

A K-matrix file is formatted. The standard form of the K-matrix file is given below.

```
record a. NZ,NELC,NAST
record b. (JAT(I),I=1,NAST)
record c. (ENAT(I),I=1,NAST)

record d. J2P1,NPTY,NCHAN,NOPEN,(NCONAT(I),I=1,NAST)
record e. W1,((AKMAT(I,J),I=1,NOPEN),J=1,NOPEN)
```

Records a,b,c are written first.

- NZ
  is the atomic number of the target.
- NELC
  is the number of electrons in the target.
- NAST
  is the number of target states.
- JAT(I) I=1,NAST
  is an array of $(2J + 1)$ angular momenta multiplied by the parity (+1 even, −1 odd) for the target states. The states are in the same order as array ENAT (target energies).
- ENAT(I) I=1,NAST
  is an array of target energies in au, ordered by increasing energy.

Records d and e are repeated for each symmetry/energy combination calculated.

- J2P1
  is $(2J + 1)$ for the $(N + 1)$-electron symmetry, where J is the total angular momentum
- NPTY
  is Π for the $(N + 1)$-electron symmetry, and Π is the parity (+1 even, −1 odd)
- NCHAN
  is the number of channels.
- NOPEN
  is the number of open channels.
- NCONAT(I) I=1,NAST
  gives the number of channels for target state I.
- W1
  is the incident energy in Ryd.
- AKMAT(I,J) I=1,NOPEN J=1,NOPEN
  is the NOPEN × NOPEN K-matrix .

The following FORMATs are used:

- record a is written as FORMAT (5I14)
- record b is written as FORMAT (5I14)
- record c is written as FORMAT (5E15.8)
- record d is written as FORMAT (5I14)
- record e is written as FORMAT (5E15.8)

Format 1P is used so that the real numbers have 9 digit accuracy.

8.6 Routines

Routine DSTG3

This is the PROGRAM routine. The input and output files are set up: IREAD (`DSTG3.INP`) and IWRITE (`DSTG3.OUT`). A call is made to routine AAMN3 which controls the computation.

Routine AAMN3

`(IREAD, IWRITE)`

This is the main controlling routine for the DSTG3 module.

Routine DMCHK3

`(I, ID, IWRITE, IDMTST, NDIMAX)`

This routine is called to test array overflow.

Routine DMPRT3

`(IWRITE, IDMTST, NDIMAX)`

This routine writes out information on the dimensions used.

Routine DMSET3

`(IWRITE, IDMTST, NDIMAX)`

This routine initialises the arrays (IDMTST, maxima set, and NDIMAX, maxima used) that check on dimensions. Dimensions are set by the file `darc.inc` which is included in the relevant routines. All dimension variables beginning with MX are set in this file. To change a dimension you need only change `darc.inc`.

Routine INTBUT

`(KK, XX, BUTTLE, CBUT, EBUT, MBUT)`

This routine interpolates the Buttle correction at the required energy.

Routine PRMAT

```
(ETOT, AKS, CBUT, CL, EBUT, ENAT, ENG, ET, RA, RCRTN,
RMAT, WMAT, ITC, IWRITE, K2P, MBUT, MNP1, NAST, NCHAN,
NCONAT, NOPEN)
```

This routine calculates

- channel energies (array ENG),

$$\epsilon_i = E - E_i^N \tag{8.1}$$

  where E is the total energy (variable ETOT), E_i^N is the energy of the N-electron state associated with channel i (array ENAT), energies are in au,

- channel momentum squared (array ET),

$$k_i^2 = 2\epsilon_i + \left(\frac{\epsilon_i}{c}\right)^2 \tag{8.2}$$

  where c is the speed of light (variable CL),

- number of open channels n_o (variable NOPEN), a channel is open if $k_i^2 > 0$,

- R-matrix (array RMAT),

$$R_{ij} = \frac{1}{2a} \sum_{k=1}^{n_H} \frac{w_{ik} w_{jk}}{e_k - E} \tag{8.3}$$

  R_{ij} is symmetric with i and j labelling the channels, a is the R-matrix radius (variable RA), the sum is over the R-matrix poles labelled by k, w_{ik} are the surface amplitudes (array WMAT), e_k are the continuum Hamiltonian eigenvalues (array AKS), n_H is the dimension of the Hamiltonian matrix,

- Buttle corrections to the R-matrix , by a call to routine INTBUT,

$$R_{ii} \longrightarrow R_{ii} + R^{but}(\kappa_i, E) \tag{8.4}$$

A debug print is given if option 1 is set.

There is no test in the code for a possible divide by zero when the R-matrix is calculated.

Routine RDDUMP

```
(ITAPE1, ITAPE2, ITAPE5, AKS, BSTO, CBUT, CF, CL, EBUT,
ENAT, RA, WMAT, IDMTST, IKMAX, ION, ITC, IWRITE, J2P1,
JAT, K2P, KCMAX, MBUT, MNP1, NAST, NCASES, NCFGP, NCHAN,
NCONAT, NDIMAX, NELC, NPTY, NRANG2, NREC, NUMSYM, NZ)
```

This routine reads the data from DSTGH corresponding to the required symmetry.

Routine READ3

```
(RECORD, IHED, FENER, RPARM, SENER, IBUG1, IBUG2, IBUG3,
IDMTST, ION, IPARM, IREAD, ITC, IWRITE, J2P1, MXET,
NCASES, NDIMAX, NENER, NENRGS, NPTY, NUMSYM)
```

This routine reads the input data.

Routine RSCT

```
(ITAPE2, ITAPE4, ITAPE5, AKS, BSTO, CBUT, CF, CL, EOM,
EBUT, ENAT, FENER, RA, SENER, WMAT, IKMAX, ITC, IWRITE,
J2P1, JAT, K2P, MBUT, MNP1, MXET, NAST, NCHAN, NCONAT,
NENER, NENRGS, NPTY, NREC, IBUG1, IBUG2, IBUG3, ION,
IPARM, RPARM, NELC, NZ)
```

This routine controls the loop over the incident energies.

Calling tree:

```
    RSCT
        MATOUT
        PRMAT         calculate channel information
            INTBUT
            MATOUT
        QUARTZ
        XSECN         calculate collision strength
            AFACE     interface to asymptotic solver
            DGETRF
            DGETRI
            DSYEV
            IN2CH3
            MATOUT
```

- Variable ISTORE indicates the asymptotic package that is being used: ISTORE=10 for VPM, ISTORE=30 for ASYPCK, ISTORE=40 for COUL, ISTORE=50 for ASYM.
- The channel orbital angular momenta l_i $(i = 1 \dots n_{ch})$ are evaluated (array L2P).
- Variable JSTORE is set to $\Pi(2J+1)$ where Π is the parity of the symmetry (± 1) and J is the total angular momentum.
- The asymptotic coefficient matrix is stored in array CF(I,J,K) with I=1,NCHAN and J=1,NCHAN corresponding to the channels and K=1,IKMAX referring to the λ value. The asymptotic potentials are

$$\frac{a_{ij}^{\lambda}}{r^{\lambda+1}} \qquad \text{and} \qquad a_{ij}^{\lambda} \equiv \text{ CF}(i,j,\lambda) \tag{8.5}$$

  The array CF is modified as follows:

  - If option 6 is set then IKMAX is set to 1 and all elements of array CF are set to zero.
  - If option 5 is set then the off-diagonal elements of array CF are set to zero.
  - For ASYM

$$a_{ij}^{\lambda} \longrightarrow 2\, a_{ij}^{\lambda} \tag{8.6}$$

  - For ASYPCK

$$a_{ij}^{\lambda} \longrightarrow -2\, a_{ij}^{\lambda} - l_i(l_i+1)\; \delta_{ij}\; \delta_{1\lambda} \tag{8.7}$$

  - For VPM

$$a_{ij}^{\lambda} \longrightarrow 2\, a_{ij}^{\lambda} \tag{8.8}$$

  Option 2 prints the asymptotic coefficients.

- Option 1 prints the continuum Hamiltonian eigenvalues e_k ($k = 1 \dots n_H$) (array AKS) in au.

- For each incident energy E_{inc}:

  - Variable ESTORE stores the incident energy in Ryd.
  - The total energy E (variable ETOT) available in the $(N+1)$-electron system is calculated

    $$E = E_1^N + E_{inc} \tag{8.9}$$

    where E_1^N is the energy of the target ground-state and the energies are in au.
  - Routine PRMAT is called. This routine evaluates channel energies, square of channel momenta, number of open channels and the R-matrix including Buttle corrections. Option 1 gives a debug print in routine PRMAT.
  - The current incident energy is skipped if there are no open channels or if any of the channel momenta squared (array ET) is less than 10^{-5} (variable TINY). The channel momentum squared is approximately twice the channel energy i.e. it is the channel energy in Ryd.
  - Routine XSECN is called. This routine controls the solution in the external region and calculates the collision strengths. Variables ISTORE, JSTORE and ESTORE are passed to routine XSECN and written to the Ω-file `DSTG3.DAT`.

Routine XSECN

```
(ITAPE2, ITAPE4, ITAPE5, W1, KOUNT, BSTO, CF, ENG,
ESTORE, ET, RA, RMAT, RPARM, IBUG1, IBUG2, IBUG3, IKMAX,
ION, IPARM, ISTORE, ITC, IWRITE, J2P1, JAT, JSTORE, K2P,
L2P, MXET, NAST, NCHAN, NCONAT, NELC, NOPEN, NPTY, NREC, NZ)
```

This routine evaluates the asymptotic solutions and K-matrix . It calculates the eigenphases and cross-sections.

Routine XSUM

```
(ITAPE2, IWRITE, CL, FENER, SENER, ITC, NENER, NENRGS)
```

This routine sums the partial cross-sections for each energy.

Chapter 9

PDSTG3

9.1 Introduction

$Source: /home/phn/DARC/RCS/pdstg3.mak,v
$Author: phn
$Date: 2001/10/31 13:36:25
$Revision: 11.1

DARC, the Dirac Atomic R-matrix Codes.
homepage: http://www.am.qub.ac.uk/DARC/
PDSTG3 module

PDSTG3 (i.e. , PASYPCK or PCOUL) solves the external region scattering equations for a photo-ionisation calculation.

$$\mathrm{A}(N+1)+\hbar\omega \longrightarrow \mathrm{A}^{+}(N)+\mathrm{e} \tag{9.1}$$

The first $(N+1)$-electron symmetry read from the DSTGH dump is taken to be the initial bound-state. A call is made to routine PITERE and an iterative procedure is used to find the energy of the lowest bound-state for this symmetry. The initial wavefunction is then calculated.

The second $(N+1)$-electron symmetry read from the DSTGH dump is the final state in the process, the photon has been absorbed by the initial bound-state resulting in an electron in the continuum. The dipole matrix elements are read from the dump, combined with the initial wavefunction and the asymptotic solutions for the second symmetry. This gives the cross-section.

Calling tree:

```
PDSTG3
    AAPHOT

        QUARTZ  timing routine
        CALEN   access date/time at execution
        DMSET3  set dimensions

        PREAD3  read input data

        PRDDMP  read DSTGH file for required symmetry

        PITERE  calculate initial bound-state
        PRSCT   calculate cross-sections

        DMPRT3  final print of dimensions used
```

The module can be interfaced to two asymptotic modules at present.

PASYPCK module

PASYPCK is a nonrelativistic program. It is based on the ASYPCK2 package developed by Crees (1980 [31], 1981 [32]).

PCOUL module

PCOUL is a nonrelativistic program for ion targets. It neglects channel coupling and uses Coulomb solutions. Closed channel solutions are obtained from the COUL program of Bell and Scott (1980 [8]). Open channel solutions use the COULFG program of Barnett (1982 [5]). This version of COULFG has been taken from the CFASYM program of Noble and Nesbet (1984 [46]).

9.2 Files

The files are defined internally. Messages are also written to output stream 6 using the PRINT command.

The following files are used:

stream name	description	stream number	file name	FORM
IREAD	Input data.	5	`PDSTG3.INP`	Formatted
IWRITE	Output.	7	`PDSTG3.OUT`	Formatted
—	Scratch file for input data.	9	—	Formatted
ITAPE1	DSTGH file.	11	`DSTGH.DAT`	Unformatted
			`DSTGHF.DAT`	Formatted
ITAPE2	Cross section file.	12	`PHOTO.SH`	Formatted
ITAPE3	—	—	scratch	Unformatted
ITAPE4	Phase shifts, option 4 set.	14	`PHASE.SH`	Formatted

9.3 Input data (`PDSTG3.INP`)

The input data uses Namelist format. Each Namelist has a name. If this is say PDSTG3 then the input data starts with the string &PDSTG3. There must be a space before the & even if it is at the beginning of the line. The end of this Namelist is denoted by &END, with & again preceded by a space. Variables in a Namelist can be given default values. The default value is indicated in brackets after the variable name.

Input record 1

Read the 80-character title, variable IHED.

Input record 2

Namelist PDSTG3 ...

```
DOUBLE PRECISION DIFFE
DOUBLE PRECISION ENSTX
INTEGER IPOLPH
INTEGER NENRGS
INTEGER NOMBX
INTEGER OPT(50)
DOUBLE PRECISION VALMX
NAMELIST /PDSTG3/ DIFFE,ENSTX,IPOLPH,NENRGS,NOMBX,OPT,VALMX
```

- DIFFE [1D−6]
  is the convergence parameter for the bound-state energy in routine PITERE.

- ENSTX [0]
  is an initial guess for the energy increment in routine PITERE. This is used if option 10 is set.

- IPOLPH [2]
  =2 for photo-ionisation
  =3 for polarisability

- NENRGS [0]
  is the number of energy ranges (compulsory). The maximum value is set in the module by parameter NDENER.

- NOMBX [30]
  is the maximum number of iterations in routine PITERE.

- OPT
  Read the options.
  Format : list of integers

  Input a list of integers giving the options to be set. An option is a switch that is either on or off. To set option J input J on this input record. The options are stored in array ITC. If option J is set then ITC(J)=1; otherwise ITC(J)=0.

  Operational options.

  - 3
    Calculate and print the eigenphases in routine PXSECN. The eigenphases lie in the range 0 to π.
  - 4
    When this option is specified the eigenphase sum is calculated in routine PXSECN and written to file `PHASE.SH`. The individual eigenphases are printed only when option 3 is set. The eigenphases lie in the range 0 to π.
  - 5
    Set the off-diagonal asymptotic scattering coefficients to zero. This corresponds to the neglect of inter-channel coupling.

- 6
  Set all the asymptotic scattering coefficients to zero. This corresponds to the Coulomb case.
- 8
  The input energies are in eV rather than the default of Ryd.
- 10
  Use the input values for VALMX and ENSTX in routine PITERE. This can be useful when the bound-state energy fails to converge properly.
- 50
  This option is set internally and indicates whether the DSTGH dump is formatted or unformatted.

Printing options.

- 1
  This is a useful debug print that gives the Hamiltonian eigenvalues, total energy, channel energies and momenta, Buttle corrections, R-matrix and K-matrix .
- 2
  Print the asymptotic scattering coefficients.
- 33
  Print the photo-ionisation results in routine PXSECN.
- 34
  Print the initial state wavefunction expansion coefficients in routine PWFNIN.
- 35
  Print the dipole matrix elements in routine PRSCT.
- 36
  Print the K-matrix in routine PXSECN.
- 37
  Print the dipole matrix elements in routine PDMAT.
- 38
  Debug print from routine PXSECN.
- 39
  Debug print from routine PITERE.
- 49
  Print out the asymptotic solutions from routine CFACE (i.e. DCOUL package), or routine AFACE (i.e. DASYPCK package).

- VALMX [0]
  is an initial guess for the bound-state energy in routine PITERE. This is used if option 10 is set.

Input record 3

Namelist ASYM ...
Input data for the asymptotic packages.

- **DASYPCK package**

  ```
  {DASYPCK}     DOUBLE PRECISION DR
  {DASYPCK}     DOUBLE PRECISION EROR
  {DASYPCK}     INTEGER IAUTO
  {DASYPCK}     INTEGER IBUG
  {DASYPCK}     INTEGER MFG
  {DASYPCK}     NAMELIST /ASYM/ DR,EROR,IAUTO,IBUG,MFG
  ```

  - DR [.15]
    is the step length for Fox-Goodwin integration
  - EROR [1D−4]
    is the accuracy parameter for asymptotic solutions
  - IAUTO [1]
    controls variation in MFG
    = 0 for using specified value of MFG
    = 1 for conditional automatic MFG
    = 2 for unconditional automatic MFG
  - IBUG [0]
    is a debug parameter (=0 for no printed output from ASYPCK)
  - MFG [200]
    is the number of steps with Fox-Goodwin

- **DCOUL package**

  No parameters are required for this package. But the Namelist card is still required.

Input record 4

Namelist RANGE ...

```
DOUBLE PRECISION EINCR
DOUBLE PRECISION ESTART
INTEGER NUME
NAMELIST /RANGE/ EINCR,ESTART,NUME
```

Read in the photo-electron energies in Ryd (option 8 set: in eV) relative to the ground-state of the final-state ion.

- EINCR [0]
  is the energy increment in the range.

- ESTART [0]
  is the first energy in the range.
- NUME [0]
  is the number of energy points in energy range. If NUME < 1 (default) then there is only one energy in the range, namely ESTART.

This input record is repeated for each of the NENRGS ranges.

Input record 5

This input record specifies the $(N+1)$-electron symmetries to be considered.

Namelist SYM ...

```
DOUBLE PRECISION JTOT
INTEGER NPTY
INTEGER NUMSYM
NAMELIST /SYM/ JTOT,NPTY,NUMSYM
```

- JTOT [-1.0]
  is the J-value (total angular momentum) of the $(N+1)$-electron symmetry. This is a real number such as .5 or 1. or 1.5 etc. It is used to form the integer variable J2P1=2∗JTOT+1.
- NPTY [0]
  is the Π-value (parity) of the $(N+1)$-electron symmetry (-1 for odd, 1 for even).
- NUMSYM [-1]
  - > 0 ... number of symmetries to be used. The first NUMSYM symmetries on the DSTGH dump are used.
  - $= 0$ ... no symmetries are processed
  - $= -1$ ... all symmetries on the DSTGH dump are used

Input can be ended by setting NUMSYM to 0. If the code cannot find the SYM input record then NUMSYM is set to 0 and the input data is terminated. If JTOT (≥ 0) is specified on the input record then NUMSYM is set to -2.

e.g.

- all symmetries are to be processed (takes the default for NUMSYM)
  &SYM &END
- the first 5 symmetries are to be processed
  &SYM NUMSYM=5 &END
- two specified symmetries are to be processed e.g. $\frac{1}{2}^e$ and $\frac{3}{2}^e$
  &SYM JTOT=.5 NPTY=1 &END
  &SYM JTOT=1.5 NPTY=1 &END
- no symmetries are to be processed, leave out SYM input record or use
  &SYM NUMSYM=0 &END

9.4 Routines

See the DSTG3 section for a description of routines DMCHK3, DMPRT3, DMSET3, INTBUT and PRMAT.

Routine PDSTG3

This is the PROGRAM routine. The input and output files are set up: IREAD (`PDSTG3.INP`) and IWRITE (`PDSTG3.OUT`). A call is made to routine AAPHOT which controls the computation.

Routine AAPHOT

```
(IREAD, IWRITE)
```

This is the main controlling routine for the PDSTG3 module.

Routine PDMAT

```
(ITAPE1, CGC, DML, DMV, WFIN, ITC, IWRITE, MAXM1, MCFGP,
MCONHP, MNP1, NCFGP, NCONHP, NRANG2)
```

This routine evaluates each vector of dipole matrix elements in both length and velocity forms, each component corresponding to a transition from the initial state to some term of a final state basis.

Routine PITERE

```
(ITAPE1, ITAPE3, AKS, BSTO, CBUT, CF, CL, DIFFE, EOM,
EBUT, ENAT, ENG, ET, RA, RCRTN, RMAT, RPARM, WMAT, WFIN,
IBUG1, IBUG2, IBUG3, IKMAX, ION, IPARM, ITC, IWRITE, J2P1,
J2P1T, K2P, L2P, MBUT, MCFGP, MCONHP, MNP1, NAST, NCFGP,
NCHAN, NCONAT, NCONHP, NOMBX, NOPEN, NPTY, NPTYT, VALMX,
ENSTX)
```

This routine determines the energy of a bound-state, for example the initial state in a photo-ionisation or polarisability calculation.

Calling tree:

```
    PITERE
        AFACE           interface to asymptotic solver
        DGETRF
        DGETRS
        MATOUT
        PRMAT           calculate channel information
            INTBUT
            MATOUT
        PWFNIN
            DGETRF
            DGETRI
        QUARTZ
```

Routine POLZ

```
(W1, AKS, CGC, DKL, DKV, EOM, IWRITE, MAXM1, MNP1)
```

This routine calculates the frequency dependent polarisability from the diagonalised Hamiltonian matrix for a given frequency.

Routine PRDDMP

```
(ITAPE1, BSTO, CBUT, CF, CL, EBUT, ENAT, RA, IDMTST,
IKMAX, ION, ITC, IWRITE, J2P1, JAT, K2P, KCMAX, L2P,
MBUT, MNP1, NAST, NCASES, NCFGP, NCHAN, NCONAT, NCONHP,
NDIMAX, NELC, NPTY,NRANG2, NUMSYM, NZ)
```

This routine reads the data from DSTGH corresponding to the required symmetry.

Routine PREAD3

```
(RECORD, IHED, DIFFE, FENER, RPARM, SENER, IBUG1, IBUG2,
IBUG3, IDMTST, ION, IPARM, IPOLPH, IREAD, ITC, IWRITE,
J2P1, MXET, NCASES, NDIMAX, NENER, NENRGS, NOMBX, NPTY,
NUMSYM, VALMX, ENSTX)
```

This routine reads the input data.

Routine PRSCT

```
(ITAPE1, ITAPE2, ITAPE4, AKS, BSTO, CBUT, CF, CGC, CL, DKL,
DKV, DML, DMV, EOM, EBUT, ENAT, ENG, ET, FENER, RA, RCRTN,
RMAT, RPARM, SENER, WMAT, WFIN, IBUG1, IBUG2, IBUG3, IKMAX,
ION, IPARM, IPOLPH, ITC, IWRITE, J2P1, J2P1T, K2P, L2P, MAXM1,
MBUT, MCFGP, MCONHP, MNP1, MXET, NAST, NCFGP, NCHAN, NCONAT,
NCONHP, NENER, NENRGS, NOPEN, NPTY, NPTYT, NRANG2)
```

This routine controls the loop over the photo-electron energies.

Calling tree:

```
    PRSCT
        PDMAT           read dipole matrix elements
            MATOUT
        POLZ
        PRMAT           calculate channel information
            INTBUT
            MATOUT
        PXSECN
            AFACE       interface to asymptotic solver
            DGETRF
            DGETRI
            DSYEV
            IN2CH3
            MATOUT
        QUARTZ
```

- Routine PDMAT is called to calculate $\bar{D}_i^{(L)}$ and $\bar{D}_i^{(V)}$ for $i = 1 \dots n_H$.
- The continuum Hamiltonian eigenvalues and eigenvectors are read in from the DSTGH dump.

$$\sum_{j=1}^{n_H} H_{ij}\, x_{jk} = x_{ik}\, e_k \tag{9.2}$$

  The eigenvalues e_k (array AKS) are in a.u. and in decreasing order. The eigenvector x_{ik} (array X) for $i = 1 \dots n_H$ corresponds to eigenvalue e_k.

- The code calculates

$$D_k^{(L)} = \sum_{i=1}^{n_H} x_{ik}\, \bar{D}_i^{(L)} \qquad \text{and} \qquad D_k^{(V)} = \sum_{i=1}^{n_H} x_{ik}\, \bar{D}_i^{(V)} \tag{9.3}$$

  These are stored in arrays DKL and DKV respectively.

- The surface amplitudes w_{ik} (array WMAT) for $i = 1 \dots n_{ch}$ and $k = 1 \dots n_H$ are read in from the DSTGH dump.
- Option 1 prints the continuum Hamiltonian eigenvalues.
- For each photo-electron energy E_{inc}:
  - If IPOLPH=3 then routine POLZ is called to evaluate the frequency dependent dipole polarisability, $\alpha_D(E_{inc})$
  - The total energy E (variable ETOT) available in the $(N+1)$-electron system is calculated

$$\begin{aligned} E &= E_1^N + E_{inc} \\ &= E^{N+1} + \hbar\omega \\ &= E_i^N + \epsilon_i \end{aligned} \tag{9.4}$$

    where E_i^N are the energies of the residual N-electron system after photo-ionisation (array ENAT), E^{N+1} is the energy of the initial $(N+1)$-electron system (variable E0M). The energies are in au. E^{N+1} in Ryd is stored as variable W1. The photon energy $\hbar\omega$ in Ryd is stored as variable WPHOT.
  - Routine PRMAT is called. This routine evaluates channel energies, square of channel momenta, number of open channels and the R-matrix including Buttle corrections. Option 1 gives a debug print in routine PRMAT.
  - The current photo-electron energy is skipped if there are no open channels or if any of the channel momenta squared (array ET) is less than 10^{-5} (variable TINY). The channel momentum squared is approximately twice the channel energy i.e. it is the channel energy in Ryd.
  - Routine PXSECN is called. This routine controls the solution in the external region and calculates the cross-sections.

Routine PWFNIN

`(ITAPE3, AAX, AKS, EOM, FOX, RA, RMATX, WMAT, WFIN, ITC,`
`IWRITE, MCFGP, MCONHP, MNP1, NCFGP, NCHAN, NCONHP)`

This routine determines the bound initial state wavefunction used in photo-ionisation or polarisability calculations.

Routine PXSECN

`(ITAPE3, AAX, AKS, EOM, FOX, RA, RMATX, WMAT, WFIN, ITC,`
`IWRITE, MCFGP, MCONHP, MNP1, NCFGP, NCHAN, NCONHP)`

This routine evaluates the asymptotic solutions, K-matrix and cross-sections.

- A call is made to an asymptotic package that calculates solutions in the external region ($r > a$). The package returns:

  - A set of solutions and their derivatives evaluated at the R-matrix radius $r = a$ (variable RA). These are stored in the common block /F0F0D/ as the arrays F0 and F0D.
  - The $n_{ch} \times n_{ch}$ K-matrix K (array AKMAT), where n_{ch} is the number of channels (variable NCHAN). A submatrix is the $n_o \times n_o$ open-open K-matrix $K^{(oo)}$, where n_o is the number of open channels (variable NOPEN). An open channel has energy $\epsilon > 0$.

  The packages available are ASYPCK and COUL. You choose the package to be used when the code is initially preprocessed. Both provide nonrelativistic solutions in the external region. ASYPCK includes channel coupling. COUL neglects channel coupling and provides Coulomb solutions, appropriate for ions only. The K-matrix is calculated in the interface routine AFACE for ASYPCK and CFACE for COUL.

- Option 1 prints $K^{(oo)}$. Option 36 prints K.

- If either option 3 or 4 is set then $K^{(oo)}$ is diagonalised. The eigenvalues are $\tan \delta_i$ for $i = 1 \ldots n_o$ and δ_i is known as the eigenphase. The eigenphases are chosen to lie between 0 and π. The eigenphase sum is

  $$\delta_{sum} = \sum_i \delta_i \tag{9.5}$$

  and lies between 0 and π. The behaviour of the eigenphase sum as a function of energy gives information on the resonance behaviour of the cross-sections.

  Option 3 provides a printed record of the eigenvalues, eigenvectors and eigenphase sum.

  If option 4 is set then the eigenphase sum is written out to the formatted file `PHASE.SH`. This file takes the form of a ksh or bash script that can be executed to produce an index file (called `INDEXP`) and one file for each symmetry (called `phaxxx`, where `xxx` is an integer). The index file gives information on the symmetry files. Each symmetry file has 2 columns, col.1 is the photo-electron energy (E_{inc}) in Ryd, col.2 is the eigenphase sum (δ_{sum}). A perl program, `draw-phase.phn`, can be used to generate a gnuplot script for plotting.

- Calculate the $n_o \times n_o$ matrix

$$\left[1 + \left(K^{(oo)}\right)^2\right]^{-1} \tag{9.6}$$

- Calculate the $n_{ch} \times n_o$ matrix

$$X_{ij} = \frac{1}{\sqrt{\pi}} \left[\sum_{k=1}^{n_o} F^{(2)}_{ik} \, K_{kj} + \sum_{k=n_o+1}^{n_{ch}} F^{(1)}_{ik} \, K_{kj} + F^{(1)}_{ij}\right] \tag{9.7}$$

  Option 38 prints X.

- Calculate the $n_{ch} \times n_o$ matrix

$$T = R^{-1} \, X \, \left[1 + \left(K^{(oo)}\right)^2\right]^{-1} \tag{9.8}$$

  where R is the $n_{ch} \times n_{ch}$ R-matrix (variable RMAT). Option 38 prints T.

- Calculate the following vectors, for $j = 1 \ldots n_o$, where superscript (L) denotes length form and superscript (V) denotes velocity form

$$R^{(L)}_j = \frac{1}{2a} \sum_{k=1}^{n_H} \sum_{i=1}^{n_{ch}} \left(\frac{1}{e_k - E}\right) w_{ik} \, T_{ij} \, D^{(L)}_k \text{ and } S^{(L)}_j = -\sum_{k=1}^{n_o} R^{(L)}_k \, K^{(oo)}_{kj} \tag{9.9}$$

$$R^{(V)}_j = \frac{1}{2a} \sum_{k=1}^{n_H} \sum_{i=1}^{n_{ch}} \left(\frac{1}{e_k - E}\right) w_{ik} \, T_{ij} \, D^{(V)}_k \text{ and } S^{(V)}_j = -\sum_{k=1}^{n_o} R^{(V)}_k \, K^{(oo)}_{kj} \tag{9.10}$$

  where n_H is the dimension of the continuum Hamiltonian matrix (variable MNP1), e_k are the continuum Hamiltonian eigenvalues (array AKS) and w_{ik} are the surface amplitudes (array WMAT). Option 38 prints the vectors.

- The partial cross-sections for $j = 1 \ldots n_o$ are

$$\sigma^{(L)}_j = \frac{C\omega}{(2J_T + 1)} \left[\left(R^{(L)}_j\right)^2 + \left(S^{(L)}_j\right)^2\right] \tag{9.11}$$

$$\sigma^{(V)}_j = \frac{4C}{\omega(2J_T + 1)} \left[\left(R^{(V)}_j\right)^2 + \left(S^{(V)}_j\right)^2\right] \tag{9.12}$$

  where

$$\begin{aligned} C &= \frac{4\pi^2\alpha}{3} \, a_0^2 \qquad (9.13) \\ &= 0.0960293174 \, a_0^2 \\ &= 0.0960293174 \times 28.0028561 \text{ Mb} \\ &= 2.68909516 \text{ Mb} \end{aligned}$$

  and 1 Mb (Mega barn) is 10^{-22} m^2. ω is the photon energy in Ryd and J_T is the total angular momentum of the initial $(N + 1)$-electron state.

- The total cross-sections are

$$\sigma^{(L)} = \sum_{j=1}^{n_o} \sigma_j^{(L)} \qquad \text{and} \qquad \sigma^{(V)} = \sum_{j=1}^{n_o} \sigma_j^{(V)} \tag{9.14}$$

- Option 33 prints partial and total cross-sections.

- The total cross-sections are written out to the formatted file `PHOTO.SH`. This file takes the form of a ksh or bash script that can be executed to produce an index file (called `INDEXC`) and one file for each symmetry (called `croxxx`, where `xxx` is an integer). The index file gives information on the symmetry files. Each symmetry file has 3 columns, col.1 is the photo-electron energy (E_{inc}) in Ryd, col.2 is the total (length form) cross-section ($\sigma^{(L)}$), col.3 is the total (velocity form) cross-section ($\sigma^{(V)}$). The cross-sections are in units of Mb. A perl program, `draw-photo.phn`, can be used to generate a gnuplot script for plotting.

Chapter 10

DSTG4

10.1 Introduction

$Source: /home/phn/DARC/RCS/dstg4.mak,v
$Author: phn
$Date: 2001/10/31 13:27:20
$Revision: 11.1

DARC, the Dirac Atomic *R*-matrix Codes.
homepage: http://www.am.qub.ac.uk/DARC/
DSTG4 module

DSTG4 is a program that processes the Ω-files produced by the DARC modules DSTG3 (DCOUL, DASYPCK, DVPM) and STGFJJ. An Ω-file contains collision strengths as a function of incident energy and $(N+1)$-electron symmetry. An analysis of the contribution of individual symmetries to the total collision strength at a particular energy can be made. This is useful for examining the convergence of the results.

This module is run **interactively**.

Below is a flowchart of the routine calls. It has been arranged in the logical order that the code uses.

```
DSTG4  AAMN4
          QUARTZ                        initialisation
          CALEN
          DMSET4

          ADDUP  DMCHK4 DMPRT4        a.create DAT1.DAT by
                 QUARTZ                 combining omega-files
          XYZ    INDEX1               b.create DAT2.DAT and
                 DMCHK4 DMPRT4          DAT3.DAT
                 QUARTZ

          ABC0   DMCHK4 DMPRT4        a. set up arrays using
                 QUARTZ                  DAT2.DAT and DAT3.DAT
                                      b. optionally print
                                         information

          ABC    ABC1   INDEX2           analyse results at a
                 QUARTZ                  specified energy

          DEF    DEF1   DEF2   IN2CH3    output results
                        INDEX2
                        QUARTZ
                 DMCHK4 DMPRT4
                 QUARTZ

          DUMP4  QUARTZ                  convert DAT1.DAT to
```

	`formatted file DAT1.FMT`
`DMPRT4`	`final dimension print`

Collision strengths are calculated in DSTG3 for a wide range of energies and symmetries. These are usually obtained in more than one run of the program. The program DSTG4 was developed to facilitate bringing together the results from various calculations.

In a DSTG3 calculation the collision strengths are written to the Ω-file, called `DSTG3.DAT`. This is a sequential, formatted file. It is opened with STATUS set to UNKNOWN. Another DSTG3 calculation will over-write this file. Therefore if you want to keep the previous results you must rename the file.

Given a number of Ω-files, DSTG4 will combine them into a single file that is called `DAT1.DAT`. It is possible by mistake to combine Ω-files corresponding to different cases. You should carefully check the output to ensure that this does not happen. The program does some checking for consistency but it is not foolproof.

For easy handling of the collision strengths, `DAT1.DAT` is used to create a random access file `DAT3.DAT` and a smaller sequential file `DAT2.DAT`. Each record of the random access file contains the collision strengths for a particular incident energy, $(N+1)$-electron symmetry (total angular momentum J, parity Π) and DSTG3 calculation type.

Calculation type refers to the asymptotic package used. At present there are 5 packages, namely STGFJJ (ISTORE=0 or 1), DVPM (ISTORE=10), DASYPCK (ISTORE=30), DCOUL (ISTORE=40) and DASYM (ISTORE=50). Of course this classification is very restricted. For example DASYM can be used in a variety of ways by setting a range of options. The ISTORE parameter does not attempt to distinguish these. You can mix Ω-files that have different calculation type. But you should be careful doing this.

DSTG4 indexes the records according to energy and symmetry. In particular they are ordered by increasing symmetry (i.e. , J) with even and odd parity together.

The program can do the following:

- combine Ω-files (routine ADDUP)
- summarise the Ω-file (routine ABC0)
- analyse the collision strengths at a particular energy (routine ABC)
- output the collision strengths in a standard form (routine DEF)
- convert the Ω-file from unformatted to formatted (routine DUMP4)

10.2 Files

The files are defined internally. Messages are also written to output stream 6 using the PRINT command.

The following files are used:

stream name	description	stream number	file name	FORM
IWRITE	Output.	7	`DSTG4.OUT`	Formatted
IPUNCH	Output, records the input data.	8	`DSTG4.PUN`	Formatted
IO1	Old Ω-files.	20 ...	specified by user	specified by user
	New Ω-file.	1	`DAT1.DAT`	Unformatted
	Sequential file.	2	`DAT2.DAT`	Unformatted
	Random access file.	3	`DAT3.DAT`	Unformatted
	Output, results as shell script.	4	specified by user	Formatted.
	New Ω-file.	11	`DAT1.FMT`	Formatted

10.3 Routines

Routine DSTG4

The main routine DSTG4:

- sets the I/O streams IREAD (set to 5), IWRITE (set to 7) and IPUNCH (set to 8)
- opens the output files `DSTG4.OUT` (stream IWRITE) and `DSTG4.PUN` (stream IPUNCH)
- prints version information
- calls the driving routine AAMN4

The input data (stream IREAD) is read interactively. Alternatively if it is in a file `DSTG4.INP` then it can be piped to the program e.g.

```
dstg4.exe < DSTG4.INP
```

The file `DSTG4.PUN` contains a record of the interactive input data. This file, possibly edited, could be piped to the program in a later run.

Stream 6 is used for PRINT statements throughout the program. These are mainly requesting input data. In a non-interactive run you could pipe these to a file e.g.

```
dstg4.exe < DSTG4.INP > DSTG4.LIS
```

Routine AAMN4
`(IPUNCH, IREAD, IWRITE)`

This is the main controlling routine for the AAMN4 module.

Initialisation involves calls to the utility routines QUARTZ (timing), CALEN (date/time) and DMSET4 (setting maximum dimensions).

Routine ABC

```
(ESTO, OMEGA, XTMP, IE, ION, IREC, ISTO, IWRITE, JMARK,
JPOS, JTMP, KSTO, MARK, NAST, IPUNCH, IREAD)
```

This routine reads the input for the option of analysing the data at a particular energy. This is done by a call to routine ABC1.

Routine ABC0

```
(IPRT1, IPRT2, ENAT, ESTO, ETMP, OMEGA, IDMTST, IE, ION, IREC,
ISTO, IWRITE, JAT, KSTO, MARK, NAST, NDIMAX, NREC)
```

This routine opens the files `DAT2.DAT` and `DAT3.DAT`. It can also print information about the data.

The default summary lists the incident energies which are available in the Ω-file. For each energy the number of symmetries available is listed. The calculation type is also given. A typical example of the output would be

```
 1.5E02     4  0  0  0  0  0  6
```

This indicates that there are 4+6=10 symmetries for the incident energy of 150 Ryd. The columns correspond to

```
     1      2       3       4      5       6
 STGFJJ  DVPM  unused  DASYPCK  DCOUL  DASYM
```

In the above example there are 4 STGFJJ results and 6 DASYM results.

The figure for STGFJJ does not distinguish between ISTORE=0 and 1. You could check this by a complete listing of the results in the Ω-file. A difficulty with STGFJJ is that when you specify IPERT=1 (i.e., do a perturbation calculation) the program can fail and then set IPERT=0 (i.e., neglect perturbation being treated. The program prints out a warning message when this occurs and the result is not written to the Ω-file. You can supplement the missing results by doing a DASYM or DASYPCK calculation. This is an example of the advantage of mixing Ω-files which have different calculation type.

Routine ABC1

```
(JNUM, EINC, ITEST, ESTO, OMEGA, XTMP, IE, ION, IREC, ISTO,
IWRITE, JMARK, JPOS, JTMP, KSTO, MARK, NAST)
```

This routine analyses the results for a particular energy.

Routine ADDUP

```
(NIO, FILEX, FORM, ENAT, OMEGA, IDMTST, IWRITE, JAT, NDIMAX)
```

This routine combines Ω-files.

In many cases there will be a number of Ω-files. The Ω-files are combined into a single, unformatted omega file called `DAT1.DAT`. This is achieved in routine ADDUP. Then routine XYZ creates the two unformatted files `DAT2.DAT` and `DAT3.DAT` using `DAT1.DAT` as input.

```
              ADDUP              XYZ
omega files ------> DAT1.DAT ------> DAT2.DAT
(formatted or       (unformatted)    DAT3.DAT
 unformatted)                        (unformatted)
```

The dimension of array FILEX (in routines AAMN4 and ADDUP) restricts the number of Ω-files to a maximum of 50. This is easily changed.

The input Ω-files must be all either formatted or unformatted. A mixture of formats can be handled by running the code more than once i.e. combine the unformatted files into a `DAT1.DAT` (rename as `DAT1.A`), combine the formatted files into a `DAT1.DAT` (rename as `DAT1.B`), finally combine `DAT1.A` and `DAT1.B` into a `DAT1.DAT`. Remember that `DAT1.DAT` is unformatted. An option allows you to create a formatted file `DAT1.FMT`.

`DAT3.DAT` is a random access file containing all the record d from `DAT1.DAT`. This allows rapid access when looking for a particular symmetry/energy result.

`DAT2.DAT` is a sequential file that contains records a,b,c from `DAT1.DAT`. Also it contains indexing information for the variable ESTORE (energy) in the records of `DAT3.DAT`. A call to routine INDEX1 produces a sort based on the variable ESTORE.

This section can be skipped if the required files `DAT2.DAT` and `DAT3.DAT` already exist.

Routine DEF

```
(ENAT, ESTO, OMEGA, XTMP, IDMTST, IE, ION, IPUNCH, IREAD,
IREC, ISTO, IWRITE, JAT, JMARK, JPOS, JTMP, KSTO, MARK, NAST,
NDIMAX, NREC)
```

This routine reads the input data for the option of outputting results. It calls routine DEF1.

Routine DEF1

```
(ITEST, JNUM, J2P1, NPTY, KOUNT, FILE1, IOPT1, IOPT2, IOPT3,
ENAT, ESTO, OMEGA, XPLOT, XTMP, IE, ION, IREC, ISTO, IWRITE,
JAT, JMARK, JPOS, JTMP, KSTO, MARK, NAST, NREC)
```

This routine calculates the required results and outputs them by a call to routine DEF2.

Routine DEF2

```
(IPLOT, JPLOT, KOUNT, IOPT1, IOPT2, XPLOT, YPLOT, IWRITE, NPLOT)
```

This routine writes the results to the formatted file.

Routine DMCHK4

```
(I, ID, IWRITE, IDMTST, NDIMAX)
```

This routine is called to test array overflow.

Routine DMPRT4

```
(IWRITE, IDMTST, NDIMAX)
```

This routine writes out information on the dimensions used.

Routine DMSET4

```
(IWRITE, IDMTST, NDIMAX)
```

This routine initialises the arrays (IDMTST, maxima set, and NDIMAX, maxima used) that check on dimensions. Dimensions are set by the file `darc.inc` which is included in the relevant routines. All dimension variables beginning with MX are set in this file. To change a dimension you need only change `darc.inc`.

Routine DUMP4

`(ENAT, OMEGA, IDMTST, IWRITE, JAT, NDIMAX)`

This routine writes the Ω-file `DAT1.DAT` as a formatted file `DAT1.FMT`.

Routine INDEX1

`(n, arr, indx)`

This routine sorts an array of real numbers. It returns an index array that gives the ascending order.

Routine INDEX2

`(n, iarr, indx)`

This routine sorts an array of integer numbers. It returns an index array that gives the ascending order.

Routine XYZ

`(ENAT, ETMP, OMEGA, IDMTST, IWRITE, JAT, MARK, NDIMAX)`

This routine reorders the Ω-file. It uses a random access file for this. The random access file and information on the ordering are written to disk.

Chapter 11

DUMP1 – utility

11.1 Introduction

$Source: /home/phn/DARC/RCS/dump1.mak,v
$Author: phn
$Date: 2001/10/31 13:19:52
$Revision: 11.1

DARC, the Dirac Atomic *R*-matrix Codes.
homepage: http://www.am.qub.ac.uk/DARC/
DUMP1 module

DUMP1 gives information about the DSTG1 files: `DSTG1.DAT`, `ORBITALS.DAT` and `INTEGRAL.DAT`. Since the first 13 records of `ORBS.DAT` and `DSTG2.DAT` are the same as `DSTG1.DAT`, information can be given for these files.

Input is **interactive**.

Calling tree:

```
DUMP1
    AADMP1
        CALEN   access date/time at execution
        DMSET1  set dimensions
        CHECK1
            DA1
            DMCHK1
                DMPRT1
        DA1
        DMPRT1  final print of dimensions used
```

11.2 Files

The files are defined internally. Messages are also written to output stream 6 using the PRINT command.

The following files are used:

stream name	description	stream number	file name	FORM
IWRITE	Output.	7	`DUMP1.OUT`	Formatted
—	DA file for orbitals.	11	`ORBITALS.DAT`	Unformatted
—	DA file for radial integrals.	12	`INTEGRAL.DAT`	Unformatted
IDISC4	DSTG1 dump.	14	`DSTG1.DAT`	Unformatted
	DSTG1/ORBS dump.		`ORBS.DAT`	Unformatted
	DSTG2 dump.		`DSTG2.DAT`	Unformatted

11.3 Routines

See the DSTG1/ORBS section for a description of routines DMCHK1, DMPRT1 and DMSET1.

Routine DUMP1

This is the PROGRAM routine. The output file is set up: IWRITE (`DUMP1.OUT`). A call is made to routine AADMP1 which controls the computation.

Routine AADMP1
`(IREAD, IWRITE)`

This is the main controlling routine for the DUMP1 module.

The first input record sets variable IOPT0 which indicates the file that is available:

IOPT0 = 1 ... `DSTG1.DAT` is opened
IOPT0 = 2 ... `ORBS.DAT` is opened
IOPT0 = 3 ... `DSTG2.DAT` is opened

The second input record sets variable IOPT1:

IOPT1 = 1 ... print the radial integrals (needs `DSTG1.DAT` and `INTEGRAL.DAT`)
IOPT1 = 0 ... no print

The third input record sets variable IOPT2:

IOPT2 = 1 ... print the radial mesh, potentials and bound orbitals (needs `ORBITALS.DAT`)
IOPT2 = 0 ... no print

The fourth input record sets variable IOPT3:

IOPT3 = 1 ... print the continuum orbitals (needs `ORBITALS.DAT`)
IOPT3 = 0 ... no print

Routine CHECK1
`(IDISC4, IDMTST, IOPT0, IOPT1, IOPT2, IOPT3, IWRITE, NDIMAX)`

This routine will read the files created by DSTG1. The information output depends on the input options set.

Chapter 12

DUMPH – utility

12.1 Introduction

$Source: /home/phn/DARC/RCS/dumph.mak,v
$Author: phn
$Date: 2001/10/31 13:23:56
$Revision: 11.1

DARC, the Dirac Atomic *R*-matrix Codes.
homepage: http://www.am.qub.ac.uk/DARC/
DUMPH module

DUMPH allows examination, combination and change of file FORMAT for DSTGH files.

Input is **interactive**.

Calling tree:

```
DUMPH
    AADMPH
        CALEN   access date/time at execution
        DMPSET  set dimensions
        DUMPH1
            DMPCHK
                DMPPRT
        DUMPH2
            DMPCHK (see above)
        DUMPH3
            DMPCHK (see above)
        DMPPRT  final print of dimensions used
```

12.2 Files

The files are defined internally. Messages are also written to output stream 6 using the PRINT command.

The following files are used:

stream name	description	stream number	file name	FORM
IWRITE	Output.	7	`DUMPH.OUT`	Formatted
IO1	Old DSTGH dumps.	20 ...	specified by user	specified by user
IO2	New DSTGH dump.	2	`DSTGH.NEW`	Unformatted
			`DSTGHF.NEW`	Formatted

12.3 Routines

<u>Routine DUMPH</u>

This is the PROGRAM routine. The output file is set up: IWRITE (`DUMPH.OUT`). A call is made to routine AADMPH which controls the computation.

Routine AADMPH
`(IREAD, IWRITE)`

This is the main controlling routine for the DUMPH module.

Routine DMPCHK
`(I, ID, IWRITE, IDMTST, NDIMAX)`

This routine is called to test array overflow.

Routine DMPPRT
`(IWRITE, IDMTST, NDIMAX)`

The following code writes out information on the dimensions used in the code.

Routine DMPSET
`(IWRITE, IDMTST, NDIMAX)`

This routine initialises the arrays (IDMTST, maxima set, and NDIMAX, maxima used) that check on dimensions. Dimensions are set by the file `darc.inc` which is included in the relevant routines. All dimension variables beginning with MX are set in this file. To change a dimension you need only change `darc.inc`.

Routine DUMPH1
`(ICASE, NIO, INPU, OUTU, IO1, IO2, IWRITE, IDMTST, NDIMAX)`

This routine transfers the data from one file to another. The data is independent of symmetry. The input and output files can be formatted or unformatted as chosen by the user through the input.

Routine DUMPH2
`(ICASE, NIO, INPU, OUTU, IO1, IO2, IWRITE, IDMTST, NDIMAX)`

This routine transfers the data from one file to another. This data depends on the symmetry. The input and output files can be formatted or unformatted as chosen by the user through the input.

Routine DUMPH3
`(INPU, IO1, IWRITE, IDMTST, NDIMAX)`

This routine writes information about the DSTGH files.

Chapter 13

Library

13.1 LIB — system calls

Routine CALEN

This routine returns in variable RECORD (CHARACTER$*$20) the time and date at execution of program. The routine also writes out RECORD. It uses system calls that are machine specific. This routine is not essential and can be replaced by a dummy routine if necessary.

(g77, pgf77) version uses routines TIME and DATE.
(none) version uses no routines.
(watcom) version uses routines GETTIM and GETDAT.
(sun) version uses routine date_and_time.

Routine QUARTZ

This routine calculates the CPU time which has been used. It can be entered with three values for argument MODE:

- MODE= -1 — used to initialise the time stored in variable TIME1
- MODE= 0 — returns in variable TIME2 the time used in seconds since the last call and writes it out
- MODE= 2 — returns in variable TIME2 the time used in seconds since the last call

It uses system calls that are machine specific. This routine is not essential and can be replaced by a dummy routine if necessary.

(g77, pgf77) version uses routine SECNDS.
(none) version uses no routines.
(watcom) version uses routine GETTIM.
(sun) version uses routine date_and_time.

13.2 LIB — lapack/blas

This lapack/blas library is used for 3 purposes.

- Invert a matrix (using DGETRF and DGETRI).
- Solve a system of linear equations (using DGETRF and DGETRS).
- Find the eigenvalues and eigenvectors of a real, symmetric matrix (using DSYEV).

This is the double precision version of the library.

Routine DGETRF

– LAPACK routine (version 2.0) –
Univ. of Tennessee, Univ. of California Berkeley, NAG Ltd.,
Courant Institute, Argonne National Lab, and Rice University
March 31, 1993

This routine computes an LU factorisation of a general m-by-n matrix A using partial pivoting with row interchanges.

The factorisation has the form $A = P * L * U$ where P is a permutation matrix, L is lower triangular with unit diagonal elements (lower trapezoidal if $m > n$), and U is upper triangular (upper trapezoidal if $m < n$).

This is the right-looking Level 3 BLAS version of the algorithm.

```
======================================================================
      SUBROUTINE DGETRF( M, N, A, LDA, IPIV, INFO )
======================================================================

Arguments
=========

M       (input) INTEGER
        The number of rows of the matrix A.  M >= 0.

N       (input) INTEGER
        The number of columns of the matrix A.  N >= 0.

A       (input/output) DOUBLE PRECISION array, dimension (LDA,N)
        On entry, the M-by-N matrix to be factored.
        On exit, the factors L and U from the factorisation
        A = P*L*U; the unit diagonal elements of L are not stored.

LDA     (input) INTEGER
        The leading dimension of the array A.  LDA >= max(1,M).

IPIV    (output) INTEGER array, dimension (min(M,N))
```

```
        The pivot indices; for 1 <= i <= min(M,N), row i of the
        matrix was interchanged with row IPIV(i).

INFO    (output) INTEGER
        = 0:  successful exit
        < 0:  if INFO = -i, the i-th argument had an illegal value
        > 0:  if INFO = i, U(i,i) is exactly zero. The factorisation
              has been completed, but the factor U is exactly
              singular, and division by zero will occur if it is used
              to solve a system of equations.

=====================================================================
```

Routine DGETRI

– LAPACK routine (version 2.0) –
Univ. of Tennessee, Univ. of California Berkeley, NAG Ltd.,
Courant Institute, Argonne National Lab, and Rice University
September 30, 1994

This routine computes the inverse of a matrix using the LU factorisation computed by DGETRF.

This method inverts U and then computes $\mathrm{inv}(A)$ by solving the system $\mathrm{inv}(A) * L = \mathrm{inv}(U)$ for $\mathrm{inv}(A)$.

```
=====================================================================
      SUBROUTINE DGETRI( N, A, LDA, IPIV, WORK, LWORK, INFO )
=====================================================================

Arguments
=========

N       (input) INTEGER
        The order of the matrix A.  N >= 0.

A       (input/output) DOUBLE PRECISION array, dimension (LDA,N)
        On entry, the factors L and U from the factorisation
        A = P*L*U as computed by DGETRF.
        On exit, if INFO = 0, the inverse of the original matrix A.

LDA     (input) INTEGER
        The leading dimension of the array A.  LDA >= max(1,N).

IPIV    (input) INTEGER array, dimension (N)
        The pivot indices from DGETRF; for 1<=i<=N, row i of the
        matrix was interchanged with row IPIV(i).

WORK    (workspace/output) DOUBLE PRECISION array, dimension (LWORK)
        On exit, if INFO=0, then WORK(1) returns the optimal LWORK.
```

```
LWORK   (input) INTEGER
        The dimension of the array WORK.  LWORK >= max(1,N).
        For optimal performance LWORK >= N*NB, where NB is
        the optimal blocksize returned by ILAENV.

INFO    (output) INTEGER
        = 0:  successful exit
        < 0:  if INFO = -i, the i-th argument had an illegal value
        > 0:  if INFO = i, U(i,i) is exactly zero; the matrix is
              singular and its inverse could not be computed.

=====================================================================
```

Routine DGETRS

– LAPACK routine (version 2.0) –
Univ. of Tennessee, Univ. of California Berkeley, NAG Ltd.,
Courant Institute, Argonne National Lab, and Rice University
March 31, 1993

This routine solves a system of linear equations $A * X = B$ or $A' * X = B$ with a general n-by-n matrix A using the LU factorisation computed by DGETRF.

```
=====================================================================
      SUBROUTINE DGETRS( TRANS, N, NRHS, A, LDA, IPIV, B, LDB, INFO )
=====================================================================

Arguments
=========

TRANS   (input) CHARACTER*1
        Specifies the form of the system of equations:
        = 'N':  A * X = B  (No transpose)
        = 'T':  A'* X = B  (Transpose)
        = 'C':  A'* X = B  (Conjugate transpose = Transpose)

N       (input) INTEGER
        The order of the matrix A.  N >= 0.

NRHS    (input) INTEGER
        The number of right hand sides, i.e., the number of columns
        of the matrix B.  NRHS >= 0.

A       (input) DOUBLE PRECISION array, dimension (LDA,N)
        The factors L and U from the factorisation A = P*L*U
        as computed by DGETRF.

LDA     (input) INTEGER
```

```
        The leading dimension of the array A.  LDA >= max(1,N).

IPIV    (input) INTEGER array, dimension (N)
        The pivot indices from DGETRF; for 1<=i<=N, row i of the
        matrix was interchanged with row IPIV(i).

B       (input/output) DOUBLE PRECISION array, dimension (LDB,NRHS)
        On entry, the right hand side matrix B.
        On exit, the solution matrix X.

LDB     (input) INTEGER
        The leading dimension of the array B.  LDB >= max(1,N).

INFO    (output) INTEGER
        = 0:  successful exit
        < 0:  if INFO = -i, the i-th argument had an illegal value

======================================================================
```

Routine DSYEV

– LAPACK driver routine (version 2.0) –
Univ. of Tennessee, Univ. of California Berkeley, NAG Ltd.,
Courant Institute, Argonne National Lab, and Rice University
September 30, 1994

This routine computes all eigenvalues and, optionally, eigenvectors of a real symmetric matrix A.

```
======================================================================
   SUBROUTINE DSYEV( JOBZ, UPLO, N, A, LDA, W, WORK, LWORK, INFO )
======================================================================

Arguments
=========

JOBZ    (input) CHARACTER*1
        = 'N':  Compute eigenvalues only;
        = 'V':  Compute eigenvalues and eigenvectors.

UPLO    (input) CHARACTER*1
        = 'U':  Upper triangle of A is stored;
        = 'L':  Lower triangle of A is stored.

N       (input) INTEGER
        The order of the matrix A.  N >= 0.

A       (input/output) DOUBLE PRECISION array, dimension (LDA, N)
        On entry, the symmetric matrix A.  If UPLO = 'U', the
        leading N-by-N upper triangular part of A contains the
        upper triangular part of the matrix A.  If UPLO = 'L',
        the leading N-by-N lower triangular part of A contains
        the lower triangular part of the matrix A.
        On exit, if JOBZ = 'V', then if INFO = 0, A contains the
        orthonormal eigenvectors of the matrix A.
        If JOBZ = 'N', then on exit the lower triangle (if UPLO='L')
        or the upper triangle (if UPLO='U') of A, including the
        diagonal, is destroyed.

LDA     (input) INTEGER
        The leading dimension of the array A.  LDA >= max(1,N).

W       (output) DOUBLE PRECISION array, dimension (N)
        If INFO = 0, the eigenvalues in ascending order.

WORK    (workspace/output) DOUBLE PRECISION array, dimension (LWORK)
        On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK   (input) INTEGER
        The length of the array WORK.  LWORK >= max(1,3*N-1).
        For optimal efficiency, LWORK >= (NB+2)*N,
        where NB is the blocksize for DSYTRD returned by ILAENV.

INFO    (output) INTEGER
        = 0:  successful exit
        < 0:  if INFO = -i, the i-th argument had an illegal value
        > 0:  if INFO = i, the algorithm failed to converge; i
              off-diagonal elements of an intermediate tridiagonal
              form did not converge to zero.

=====================================================================
```

13.3 LIB — miscellaneous routines

These routines are used in a number of modules.

Routine DA1

This routine controls access to a direct access (DA) file. The record length is 8∗LREC bytes (LREC is set to 512). Parameters allow you to open the file, write to the file or read from the file.

In the following examples the file name is 'file.dat' and file stream number is IO.

```
IREC = -1
CALL DA1('file.dat',0,IREC,IO,0,ARRAY)   opens a scratch file
                                          filename is not used

IREC = 0
CALL DA1('file.dat',1,IREC,IO,0,ARRAY)   opens file 'file.dat' with
                                          status OLD (will check file
                                          exists) for reading

IREC = 0
CALL DA1('file.dat',2,IREC,IO,0,ARRAY)   opens file 'file.dat' with
                                          status UNKNOWN for writing

CALL DA1('file.dat',1,IREC,IO,L,ARRAY)   reads ARRAY(1:L)
     on entry IREC is the first DA record for array,
     on exit IREC is the next available DA record

CALL DA1('file.dat',2,IREC,IO,L,ARRAY)   writes ARRAY(1:L)
     on entry IREC is the first DA record for array,
     on exit IREC is the next available DA record
```

Routine DRACAH

This routine evaluates Racah coefficients.

MXNY is the dimension used for the array ISTO in routine DRACAH. This array is INTEGER*2 so as to save memory.

The following table shows results from calling DRACAH for all values of the arguments from 0 to jmax.

numc — number of coefficients calculated

numr — number of coefficients read from table

numsto — number of non-zero coefficients stored (only non-zero coefficients are stored in array XSTO)

If you choose MXNY to be one of these jmax values then numsto would be the dimension required for array XSTO.

jmax	numc	numr	numsto
0	1	0	1
1	19	45	4
2	144	585	17
3	676	3420	48
4	2350	13275	128
5	6651	40005	288
6	16219	101430	613
7	35344	226800	1176
8	70551	460890	2171
9	131275	868725	3757
10	230626	1540935	6273

Routine IN2CH3

This routine converts an integer into a 3-character variable. The integer lies between 0 and 999 inclusive. E.g. if the integer is 29 then the variable is 029.

Routine MATOUT

This is a routine for printing matrices. A maximum of 5 columns is printed using a line length of 75. The input parameter MODE determines the type of matrix: =1 symmetric; =2 unsymmetric; =3 matrix of eigenvectors; =4 unsymmetric with no heading; =5 symmetric with no heading.

13.4 LIB — HSLDR

This package diagonalises a real, symmetric matrix.

```
                          HSLDR
                            |
               ----------------------------------------
               |            |           |             |
            HOUSE        EIGEN        VECTOR       EIGVEC
                                        |             |
                                   -----------        |
                                   |         |       NORM
                                BAKSUB      NORM
```

Routine HSLDR

This routine accepts the upper triangle of an N∗N symmetric matrix and on the first call determines all the eigenvalues and the first eigenvector. On each further call one more of the remaining eigenvectors is calculated.

```
    SUBROUTINE HSLDR(N,A,LENGTH,EPSI,EIG,X,NO)
```

```
N      : the dimension of the symmetric matrix to be diagonalised.
A      : the linear array containing the upper triangle of the
         original matrix, overwritten on return by the main diagonal
         elements of the tridiagonal matrix and details of the
         transforming matrices.
LENGTH : =(N*(N+1))/2 , the dimension of the array A.
EPSI   : the accuracy to which the eigenvalues are to be determined.
EIG    : this array contains the eigenvalues on return,
         in descending order.
X      : this array contains one eigenvector on return.
NO     : this runs from 1 to N and specifies which eigenvector is
         stored in X on return
```

Routine BAKSUB

This routine solves for X the vector equation UPP X=B where UPP is an upper triangular matrix with only three non-zero diagonals, X and B are column vectors. The arrays U,V and W are as described in routine VECTOR.

Routine EIGEN

This routine accepts the arrays R and P of main and super diagonal elements respectively. Using the Sturm sequence property a bisection method is applied to determine the eigenvalues (stored in the array EIG on return) to an accuracy specified by EPSI. N is as defined in the routine HSLDR.

Routine EIGVEC

This routine takes the eigenvector of the tridiagonal matrix stored in X and details of the matrices used in transforming the original matrix to tridiagonal form, stored in A, and obtains the corresponding eigenvector of the original matrix. N and LENGTH are as defined in the routine HSLDR.

Routine HOUSE

This routine accepts the upper triangle of elements of a symmetric matrix, stored in the linear array A, and using the Householder method reduces this to tridiagonal form, storing the new main diagonal elements in position in A and also in the array R, and the super-diagonal elements in the array P. Details of the transforming matrices are overwritten in the redundant space of A. N and LENGTH are as defined in routine HSLDR.

Routine NORM

This routine normalises the vector X of dimension N such that the largest component is unity.

Routine VECTOR

This routine takes arrays R of main diagonal elements, P of super diagonal elements, EIG of eigenvalues, of the tridiagonal matrix, and by means of inverse iterations determines an eigenvector of the tridiagonal matrix.

13.5 LIB — NJGRAF

This program calculates a general recoupling coefficient.

```
NJGRAF
    BUBBLE
        DELTA
        PHASE
        PHASE2
    CHKLP1
        PRINTJ
        ZEROJ
            DELTA
            NEIBOR
            OTHERJ
            PHASE2
            PRINTJ
    CUT1L
        DELTA
        PRINTJ
        ZEROJ (see above)
    CUT2L
        DELTA
        OTHERJ
        PHASE2
        PRINTJ
    CUTNL
        DELTA
        PHASE
        PHASE2
        PRINTJ
    DIAGRM
        INTAB
        NEIBOR
        OTHERJ
        PHASE
        PRINTJ
        WAY
            NEIBOR
            OTHERJ
    GENSUM
        DRACAH
        RDIAG
    LOLPOP
        DELTA
        PHASE2
    ORDTRI
        CHANGE
            PHASE
        PRINTJ
```

```
POLYGN
    PHASE2
    PRINTJ
PRINTJ
SEARCH
    CHANGE (see above)
    PHASE
    PRINTJ
SETDM
SETTAB
    DELTA
    OTHERJ
    PRINTJ
SPRATE
    CHVAR
    VAR
SQUARE
    PHASE2
TRIANG
    PHASE2
    TRDEL
ZEROJ (see above)
```

Routine NJGRAF

This is the main program. It handles all the analysis of the recoupling coefficient without referring explicitly to the values of angular momenta which are in J1(J),except for zero in case FREE = .FALSE. . Like NJSYM it prepares arrays of arguments for phase factors, $(2J + 1)$ factors and $6j$-symbols to be computed in GENSUM, which can also be called separately when only the numerical values of angular momenta change. These variable angular momenta should be declared FREE(J) = .TRUE. , so that the formula prepared for GENSUM should be correct when J1 is not zero. FAIL will be TRUE when the recoupling coefficient is zero because of unsatisfied delta or other similar causes.

Routine BUBBLE

This routine reduces a circuit of order 2, giving delta function and phase factors.

Routine CHANGE

This routine exchanges the free ends in either first or last triad of JDIAG.

Routine CHKLP1

This routine checks if there are active triads with two identical arguments. This is a loop of order 1 (a ”lollypop”). The other argument must then be zero; i.e. j1(j) = 1 in $2j + 1$ notation. Suppression of the loop introduces factors and phases. Two Two triads become inactive. All this is performed by invoking ZEROJ with first argument 1.

Written by Marcel Klapisch for correcting an error detected by Charlotte F Fischer. This version includes Marcel's fix of 12 June 1992.

Routine CHVAR

This routine changes the order of summation variable to be able to perform separately the summations in GENSUM.

Routine CUT1L

This routine deals with a cut on one line, that was left as a free end in JDIAG. Puts corresponding delta in J23.

Routine CUT2L

This routine deals with a cut on two lines that were left as free ends in JDIAG. Puts corresponding delta in J23.

Routine CUTNL

This routine examines the case where there are more than two free ends, but they are contiguous, so that the graph can be cut without destroying the flat structure.

Routine DELTA

This routine tests for delta(JA,JB). If they are summation variables, the second is changed into the first everywhere. If they are fixed, their value is checked, and fail put to .TRUE. if they differ.

Routine DIAGRM

This routine builds up a flat diagram from the triads J23 and places them in JDIAG. Arrows are in ARR (INTEGER). The diagram is built so as to maximise the number of triads involved, within a one-step-forward-check process. If the diagram does not include all the NBTR triads, it will have 'free ends'. JDIAG has dimension double that of J23, because the path may proceed either way.

Routine GENSUM

This routine carries out the summation over coefficients defined by the arrays J6, J7, J8, LDEL and JW to give RECUP. The entry is either made from NJGRAF or directly assuming that the arrays J6,...,JW have already been determined by a previous entry to NJGRAF and that the summation is required for another set of j values defined by the array J1. RECUP is the recoupling coefficient.

Routine INTAB

This routine is called at the end of DIAGRM, fixes the arrays IH and IL — hardware and logical addresses of triads in JDIAG. Also determines the number of free ends NFREE and their location ITFREE.

Routine LOLPOP

This routine reduces a loop with one line and one node in the flat graph.

Routine NEIBOR

This routine gives the positions of the other two arguments in the triad.

Routine ORDTRI

This routine orders the triads which were left with free ends as consequence of cutting,so that the new graph will start there.

Routine OTHERJ

This routine gives the other triad where a given J occurs and its position.

Routine PHASE

This routine computes the phase factor arising from non-cyclic permutation of arguments in triad L. JM may be either J23 or JDIAG.

Routine PHASE2

This routine adds a phase factor (-1)**2J.

Routine POLYGN

This routine reduces a circuit of arbitrary order NC. It exchanges nodes on the flat diagram until the distance on the axis between nodes equals one. Each exchange introduces a summation variable and a $6j$-symbol. The circuit has a maximum of NPART = 2 disconnected parts on the axis.

Routine PRINTJ

This routine prints intermediate results in standard form from wherever it is called.

Routine RDIAG

This routine is called by GENSUM to establish the range of values of the summation variables. This routine replaces an extended range do loop in GENSUM, to conform with the FORTRAN 77 standard.

Routine SEARCH

This routine locates circuits or loops of order NC. NPOINT(NC) are the indices of the points (triads) pertaining to the first such loop found. NPART is the number of separate parts (groups of contiguous points) on the axis of the flat graph. IPARTS is the number of points in the smallest part. IPARTL is the number of points in the largest part. The routine finds all the possible loops of order 3 and 4. For NC.GE.5, it looks for only those who are partitioned in NPART.LE.2. which can eventually reduce to a loop of order 4 without breaking the basic structure of the flat graph. ICROSS = -1, if lines cross.

Routine SETDM

This routine sets dimensions of arrays.

Routine SETTAB

This routine builds up the unstructured graph. Sets the array J23, containing the two lists of original triads J2 and J3, and the corresponding arrows on the angular momenta lines. Also establishes the numerical and phase factors connecting recoupling coefficient and graphs, according to Yutsis et al. For this purpose determines the total J.

Routine SPRATE

This routine prepares the information to be transfered to GENSUM for numerical evaluation.

Routine SQUARE

This routine reduces a circuit of order 4 in the two cases which are left over by POLYGN, namely two disconnected groups of two points and one group of two points plus the two ends of the axis. In the latter, the end of the axis is transferred to the beginning. In this process, one summation variable and two $6j$-symbols are introduced.

Routine TRDEL

This routine tests for triangular delta. If not satisfied FAIL = .TRUE. .

Routine TRIANG

This routine reduces a triangle having one apex at either end of the axis of the flat diagram. This introduces one $6j$-symbol and some phase factors.

Routine VAR

This routine tests for variable character and put in JNS if yes, and JN now contains 0.

Routine WAY

This routine tests one step forward if the way is free. First and second arguments are interchanged or not according to ICH = −1, or +1.

Routine ZEROJ

This routine suppresses one line and two nodes of the unstructured graph introduces zeros in the triads J23. As a consequence the other two arguments of the triad are put equal. If there was already a zero in the triad which is changed, it is a special case.

13.6 LIB — NJSYM

This program calculates a general recoupling coefficient. It is an adaption of the revised NJSYM program due to Scott and Hibbert (1982 [55]). The original program was written by Burke.

```
NJSYM
    GENI9
    GENJ45
    GENSM1
        DRACAH
```

Routine NJSYM

This routine controls the calculation.

Routine GENI9

This routine determines the number of recoupling necessary to bring J2(I5,I3) and J2(I6,I4) into the same triad. This will give a triad identical with one in J3. On exit I9 contains the number of recouplings plus two, I7 contains the level of the I5 triad below the common triad and I8 contains the level of the I6 triad below the common triads.

Routine GENJ45

This routine finds the level of each J in the coupling trees of J2 and J3 and store in the J4 and J5 arrays respectively. If an element of J1 does not occur in J2 the J4 entry is −1 and if an element does not occur in J3 the J5 entry is −1.

Routine GENSM1

This routine carries out the summation over coefficients defined by the arrays J6, J7, J8 and JW to give RECUP.

The entry is either made from NJSYM or directly assuming that the arrays J6, J7, J8 and JW have already been determined by a previous entry to NJSYM and that the summation is required for another set of J values defined by the array J1.

13.7 LIB — ASYPCK

This program provides nonrelativistic solutions to the asymptotic scattering equations.

Routine AFACE

This routine is an interface between DSTG3/PDSTG3 and ASYPCK.
Calling tree:

```
AFACE
    ASYPCK
        ASYMPT
            ASYSM
                COULGM
                EXPAN
                ITERA
                    POP
                        BOOLE
                        DCHAIN
                    PPFS
                    QROP
                        DCHAIN
                    ZETA
                PHAS
                POTS
                SOLV
                    WOUTER
            FG
                DGETRF
                DGETRI
                WW
            SOLV (see above)
        BOUND
        INTERP
        OMEGA
        SOLNS
            PRTFNS
            SECDRV
                WW
    DGETRF
    DGETRS
    MATOUT
```

Input variables:

- IWRITE
  stream for printed output
- IBUG1
  debug parameter

- RPARM
  real parameters (20 can be used)
  - DR = RPARM(1) — step length for Fox-Goodwin integration
  - EROR = RPARM(2) — accuracy parameter for asymptotic solutions
- IPARM
  integer parameters (20 can be used)
  - MFG = IPARM(1) — number of steps with Fox-Goodwin
  - IBUG = IPARM(2) — debug parameter (=0 for no L/P output from ASYPCK)
  - IAUTO = IPARM(3) — controls variation in MFG
    = 0 for using specified value of MFG
    = 1 for conditional automatic MFG
    = 2 for unconditional automatic MFG
  - IKMAT = IPARM(20) — controls calculation of K-matrix
    = 0 for no calculation of K-matrix
    = 1 for calculation of K-matrix
- ION
  degree of ionisation
- NCHAN
  number of channels
- NOPEN
  number of open channels
- KKMAX
  maximum λ for the asymptotic coefficients
- RR
  R-matrix boundary radius
- BB
  boundary condition
- L2P
  channel l-values
- ETT
  channel k^2
- CF
  asymptotic coefficients
- RMAT
  R-matrix

Output variables:

- AKMAT
  K-matrix
- IERR
  error parameter (=0 ok, =1 failure) — NOT USED

Routine ASYPCK

This routine solves the coupled differential equations of electron-atom collision theory that are valid at large radial distances (the asymptotic region). It computes:

- solutions of the equations at two radial points
- solutions and their first derivatives at a single radial point
- radial functions at a mesh of radial points

References:

- Program ASYPCK2, Crees (1981 [32]).
- Program ASYPCK, Crees (1980 [31]). Referred to as CPC on comment cards
- Program IMPACT, Crees et al (1978 [30]). Referred to as Paper III on comment cards
- Program ASYM, Norcross (1969 [47]).
- Fox-Goodwin technique, Norcross and Seaton (1973 [48]).

Automatic MFG facility

The program ASYPCK has been changed to include a facility for automatically setting the Fox-Goodwin integration parameter MFG. By this it is meant that if the program finds that long inward integration are required with the given input value of MFG, the program can reset MFG to avoid such integrations and recall routine ASYSM to calculate solutions at the reset value of RB

Control of this facility is by a new input parameter (IAUTO). IAUTO takes the values 0, 1 and 2.

- IAUTO = 0
  ASYPCK runs as the standard published version, i.e. without the automatic MFG facility.

- IAUTO = 1
  ‘conditional MFG facility’. If IAUTO = 1, ASYPCK will calculate any increment in MFG but will not implement the automatic facility if either the increase in MFG is less than 150 or radial function output required for that run (i.e. JFILE specified as negative).

- IAUTO = 2
  ‘unconditional MFG facility’. If IAUTO = 2, ASYPCK will calculate any increment to MFG and will always reset MFG and recall ASYSM with the one following exception. If radial function output is required, the maximum value of MFG is limited by dimensions. If the increased MFG is greater than the maximum allowed, MFG is set to the maximum.

This automatic MFG facility operates on both the cases of some open channels and all closed channels. In the latter, if solutions are required at both the input and incremented energies, any increase in MFG is determined from the calculations for the input energies.

Routine ASYMPT

(see CPC section 5.1)

ASYMPT organises the calculation of the solutions at `R1` and `R2`. These solutions are returned in arrays `F1` and `F2` in common /SAVE/.

The method of calculation is

- To call ASYSM to calculate solutions at `RB` (ASYSM is a modified version of the program ASYM of Norcross (1969) [47]) with the introduction of the automatic `MFG` feature, this may mean a redefinition of `MFG`, `RB` and `RA` and also a second call to ASYSM. See comments in MAIN.
- Call SOLV to integrate inwards (using 4th order Runge-Kutta) to obtain solutions at `RA`
- For `MFG` > 0 call FG to obtain solutions at (`R1`,`R2`) from the solutions at (`RA`,`RB`) using Fox-Goodwin integration
- Renormalise solutions at `R1` and `R2`

For the definition of `RA`, `RB`, `R1` and `R2` see CPC eqs.(5.1) and (5.2).

Routine ASYSM

(see CPC section 5.2)

ASYSM is called by ASYMPT to calculate solutions at `RB`, which is known internally as `R3`.

This is a modified version of the program ASYM of Norcross (1969) [47]. The modifications are described in section 7 and appendix c of paper III and on comment cards below.

Input data:

- ET(N)
  energies (Rydberg units) All open channels must precede all closed channels, degenerate channels must be grouped together.

- EL(N)
  angular momenta values. Must always be included in array BLAM

- NCHF
  number of channels (less than or equal to NCHFMX)

- SZ
  residual charge (greater than or equal to 0)

- BLAM
  coefficients of $(1/r)^(k+1)$. A symmetric matrix positive/negative for attractive/repulsive potentials.

- LAMMX
  maximum value of k in above series

- R3
  value of r (atomic units) at which solution desired

- DR
  initial estimate of step-length for inward integrations

- EROR
  criterion for convergence, energy degeneracy and integration. must be greater than or equal to 1.E−5

Parameters set-up internally:

- NZD
  residual charge (integer SZ)

- DELE
  some small positive energy. Typically 0.07∗SZ∗SZ For details see Norcross (1969) [47].

- IOUT
  unit number for output.

- JBUG
  debugging parameter, taking values 0 and 1. The input parameter is called IBUG in ASYSM. JBUG = 0 (no datasets written onto unit IOUT) if IBUG = 0 or 1. JBUG = 1 (datasets written onto unit IOUT) if IBUG = 2.

Output results:

F(K,I,J) is I-th member of J-th boundary condition (ordered as input ET(J))

- F(1,I,J) — regular solutions
- F(2,I,J) — irregular solutions
- FDR(1,I,J) — derivatives of regular solutions
- FDR(2,I,J) — derivatives of irregular solutions

For open channels:

- F(1,I,I) asymptotic to (ET(I)**(-1/4))*SIN(PHI)
- F(2,I,I) asymptotic to (ET(I)**(-1/4))*COS(PHI)

For closed channels:

- F(1,I,I) asymptotic to (ET(I)**(-1/4)*EXP(-PHI) For ET(I) = 0.0, ET(I) are replaced in asymptotic amplitude by 2.0*SZ/R
- F(2,I,I) asymptotic solutions not calculated

Comments:

- Not applicable for
  - SZ = 0 if any ET(J) = 0
  - SZ less than 0
- R3 increased to RF until solution accurate to 1.E−4 obtained. Result then integrated back to R3
- Large RF may be required (warning printed) for cases
  - large positive energies just below threshold
  - SZ = 0 just above threshold
  - nearly degenerate channels
- ISF(I) is the number of terms taken in asymptotic series for channel I (see eq.(c6) of appendix C of paper III).

13.8 LIB — COUL

This program provides nonrelativistic Coulomb solutions to the asymptotic scattering equations.

Routine CFACE

This routine is an interface between DSTG3/PDSTG3 and COUL/COULFG.

Calling tree:

```
CFACE
    COUL         - closed channel solutions
        CHEBCO
        DIGAMM
        FC03AD
    COULFG       - open channel solutions
        JWKB
    DGETRF
    DGETRS
    MATOUT
```

Routine COULFG

This routine is the revised Coulomb wavefunction program using Steed's method.

	reference
original program 'RCWFN'	Barnett (1974 [2])
modified version 'RCWFF'	Barnett (1976 [3])
full description of algorithm	Barnett (1981 [4])
this version written up	Barnett (1982 [5])

COULFG returns F, G, F', G', for real $x > 0$, real η (including 0), and real $\lambda > -1$ for integer-spaced λ values, thus giving positive-energy solutions to the Coulomb Schrödinger equation, to the Klein-Gordon equation and to suitable forms of the Dirac equation, also spherical and cylindrical Bessel equations.

```
CALL COULFG(XX,ETA1,XLMIN,XLMAX,FC,GC,FCP,GCP,MODE1,KFN,IFAIL)
```

	input variable
x	XX
η	ETA1
λ	XLM in range XLMIN$\cdots$XLMAX

For a range of λ values, (XLMAX $-$ XLMIN) must be an integer, starting array element is M1 = MAX(INT(XLMIN+ACCUR),0) + 1. See text for modifications for integer l-values.

MODE1	functions calculated
1	F, G, F', G'
2	F, G
3	F

For integer-spaced lambda values unused arrays must be dimensioned in the call to at least length (1).

KFN	type of function
0	real Coulomb
1	spherical Bessel
2	cylindrical Bessel

The use of `MODE1` and `KFN` is independent.

Precision: results to within 2-3 decimals of machine accuracy in the oscillating region

$$x \geq \eta + \sqrt{\eta^2 + \lambda(\lambda + 1)} \tag{13.1}$$

This version of COULFG has been taken from the program CFASYM.

1. It is in double precision. `ACCUR` (or `ACC`) is set to 1D−16. You should refer to the writeup or the original code for the modifications for higher (or lower) precision.
2. The implicit function names have been replaced by generic names.
3. The code has been tidied. A number of cosmetic changes have been made.
4. Warning and error messages have been commented out. In routine CFACE there are warning messages for routine COULFG.
5. This code differs from the original in the use of `ACCH` rather than `ACC` in a test. `ACCH` is the square root of `ACC`, i.e. it is larger.

Routine JWKB

This routine computes JWKB approximations to Coulomb functions for `DL` ≥ 0 as modified by Biedenharn et al (1955 [14]).

Routine COUL

This routine calculates the exponentially decaying solutions and first derivatives at r of the Coulomb equation

$$u_l'' + \left(k^2 + \frac{2z}{r} - \frac{l(l+1)}{r^2}\right) u_l = 0 \tag{13.2}$$

i.e. the solution tends to infinity as

$$u = B \exp\left(-|k|r + \frac{z}{|k|\ln(2|k|r)}\right) \tag{13.3}$$

where B is a constant.

```
SUBROUTINE COUL(SK,Z,L,R,U1,U1P,NORM)
```

Input supplied by the user

- SK
  k^2 and must be a real negative quantity.
- Z
  the nuclear charge z which must be greater than zero.
- L
  the maximum angular momentum required i.e. solutions will be provided for $l = 0, 1, 2, \ldots$L in one call of the routine. Be careful to dimension the arrays U1 and U1P correctly in the calling routine.
- R
  the radius r at which the solution is required.
- NORM

  - =1
    In this case the solutions are required to have the above asymptotic form with $B = 1$.
  - =0
    In this case solutions are found which have the above asymptotic form but with B defined as follows,
    * If $\frac{z}{|k|}$ is integral and $l + 1 - \frac{z}{|k|} = -n$ for $n = 0, 1, 2, \ldots$ then

      $$B = \frac{(2l+1)!}{(n+2l+1)!(2|k|r)^{(l+1)}} \tag{13.4}$$

    * If $\frac{z}{|k|}$ is integral and $l + 1 - \frac{z}{|k|} = n$ for $n = 1, 2, 3, \ldots$ then

      $$B = \frac{(n-1)!(2|k|r)^l}{(2l)!} \tag{13.5}$$

    * If $\frac{z}{|k|}$ is non-integral and $\frac{|k|}{z}$ is less than 0.25 and $(zr + 100.0)$ is less than or equal to $\frac{0.5}{\left(\frac{|k|}{z}\right)^3}$ then

      $$B = \Gamma\left(1 - \frac{z}{|k|}\right) \tag{13.6}$$

- Otherwise $B = 1.0$

Output from the routine.

- U1
  an array containing the function for angular momentum values from $l = 0, 1, 2, \ldots$L i.e. U1(I) holds the solution with angular momentum (I-1)
- U1P
  an array equivalent to U1 but containing the first derivative.

This version has been modified as follows:

1. The code is in double precision.
2. The variable BIG is set to BIG=1.D60.

Routine CHEBCO

This routine obtains the unnormalised coefficients for the Chebyshev expansion approximation.

Routine DIGAMM

This routine calculates

$$\Psi = \frac{d(\ln \Gamma(p))}{dp} \tag{13.7}$$

where p is a real variable. The method used is the Chebyshev polynomial approximation together with the fundamental properties of Ψ as given by Luke (1969 [44]) p.301.

Routine FC03AD

This routine calculates the Γ function i.e. VALUE=Γ(X) for all values of X except X=0 and when X is a negative integer.

This routine has been obtained from the Harwell library.

13.9 LIB — VPM

This program provides nonrelativistic solutions to the asymptotic scattering equations.

Routine VFACE

This routine is an interface between DSTG3 and VPM.

```
 SUBROUTINE VFACE(
+ IWRITE,IBUG1,RPARM,IPARM,NCHAN,LAMAX,RA,BSTO,L2P,
+ ET,LRPC,RMAT,TEMPK,IERR)
```

Input:

- IWRITE
  stream for printed output
- IBUG1
  debug parameter
- RPARM
  real parameters (20 can be used)
  - ABSERR = RPARM(1)
  - ENDEG = RPARM(2)
- IPARM
  integer parameters (20 can be used)
  - IBSG = IPARM(1)
- NCHAN
  number of channels
- LAMAX
  max λ for the asymptotic coefficients
- RA
  R-matrix boundary
- BSTO
  boundary condition
- L2P
  channel l-values
- ET
  channel k^2

- LRPC
  asymptotic coefficients
- RMAT
  R-matrix

Output:

- TEMPK
  K-matrix
- IERR
  error parameter (=0 ok, =1 failure)

Routine VPM

VPM is an asymptotic package based on the variable phase method of Calogero (1967 [25]).

It was written by Croskery et al (1982 [33]).

This package solves the coupled second order differential equations which arise in close-coupling calculations on neutral atoms. The method involves reducing them to coupled first order non-linear equations which are solved using the package of Shampine and Gordon (1975 [60]).

Calling tree:

```
VPM
    ASYBSG
        MACHIN
        VGAIEX
    DISTNT
    SOLVE
        DGETRF
        DGETRI
        F
            JNBAR
                RICBFH
                    RBESI
                        MACHIN
                SBESSF
                    MACHIN
        FFASY
            JNBAR (see above)
        JNBAR (see above)
        RAMTCH
            DGETRF
            DGETRI
            FFASY (see above)
```

```
          VDE
              CHECK
                  DGETRF
                  DGETRI
              MACHIN
              VSTEP
                  MACHIN
```

```
      SUBROUTINE VPM(
     + ABSERR, ENDEG, IBSG, BSTO, RA, RMAT, ET, LRPC, LAMAX,
     + L2P, NCHAN, ITAPE, ITAPE2, KONLY, NTASY,
     + IERR, TEMPK)
```

Input:

- ABSERR
  the accuracy to which the asymptotic expansions are required
- ENDEG
  If absolute channel energy differences are less than or equal to ENDEG the energies are assumed to be degenerate
- IBSG
  = 0 — Burke-Schey expansion to be used
  = 1 — Gailitis expansion to be used
- BSTO
  The logarithmic derivative
- RA
  The boundary radius of the inner region from which the integration is started
- RMAT
  The NCHAN × NCHAN R-matrix
- ET
  The NCHAN channel energies
- LRPC
  A real 3-dimensional array containing the long-range potential coefficients
- LAMAX
  The order of the long-range potential coefficients
- L2P
  The NCHAN orbital angular momenta of the continuum electron
- NCHAN
  The number of coupled channels

- ITAPE
  The channel number of the scratch file. This is only required if KONLY=.FALSE.

- ITAPE2
  The channel number of the permanent output file. This is only required if KONLY=.FALSE.

- KONLY
  =.TRUE. — only the K-matrix is required
  =.FALSE. — both the K-matrix and information relating to the wavefunctions are to be output

- NTASY
  The maximum number of terms to be included in the asymptotic expansions. This should be at least LAMAX + 2. If it is not the code will reset NTASY to this value

Output:

On completion the array TEMPK contains the NCHAN × NOPEN K-matrix evaluated at R = RB. The elements ((TEMPK(I,J),I=1,NOPEN),J=1,NOPEN) hold the symmetric K-matrix (reactance matrix) where NOPEN is the number of open channels.

Dimensions:

- NDIM1
  the number of coupled channels (NCHAN)

- NDIM2
  the maximum number of terms in the asymptotic expansion (NTASY)

- NDIM3
  the number of coupled channels (NCHAN)

- NDIM4
  the number of points at which ABAR, BBAR and R are to be stored before being output to disc (NOPTS). Note that if KONLY=.TRUE. this need only equal 1, in which case a great deal of storage will be saved.

- NDIM5
  the maximum order of the long-range potential coefficients (LAMAX)

- NDIM6
  the maximum value plus one of the orbital angular momenta i.e. the maximum value plus one of the L2P array

The variables NDIM1 to NDIM6 are set as follows where variables beginning with MX are set in the file darc.inc.

```
      INTEGER NDIM1
      PARAMETER (NDIM1=MXCH)
      INTEGER NDIM2
      PARAMETER (NDIM2=14)
      INTEGER NDIM3
      PARAMETER (NDIM3=MXCH)
      INTEGER NDIM4
      PARAMETER (NDIM4=2)
      INTEGER NDIM5
      PARAMETER (NDIM5=MXNA)
      INTEGER NDIM6
      PARAMETER (NDIM6=MXNK)
```

Routine ASYBSG

This routine evaluates the expansion coefficients GAM1(I,J,N) and GAM2(I,J,N) arising in the inverse power series factors of the asymptotic expansion methods of Burke-Schey and Gailitis.

If IBSG=0 the Burke-Schey coefficients are calculated.

If IBSG≠0 the Gailitis coefficients are calculated (expansion 15 of Gailitis' paper is used).

The maximum number of terms to be considered is input in the variable NTASY. An accuracy parameter ABSERR must also be assigned on input. This parameter is used to determine a maximum number of terms NTIJ(I,J) (.LE.NTASY) and a radius RB(I,J) such that the series for each (I,J) is terminated at the term NTIJ(I,J) and for R.GE.RB(I,J) the last term has absolute value less than or equal to ABSERR.

Routine MACHIN is called on the first call to this routine. This evaluates the machine dependent constant FOURU.

Output:

- GAM1(I,J,N) GAM2(I,J,N)
  expansion coefficients
- NTIJ(I,J)
  number of expansion terms
- RB
  maximum value of all RB(I,J)

Degeneracy of channels:

The routine assumes that channels are degenerate if the absolute value of the energy difference between two channels is less than or equal to ENDEG where ENDEG is input.

Routine CHECK

This routine checks the equations at each step to determine whether any blow-up has occurred. Consequently it determines which equations are to be solved at each step in the integration.

Routine DISTNT

This routine examines the channel energies and determines those energies which are degenerate, setting up also the maximum angular momenta corresponding to a set of degenerate energies.

Routine F

This routine evaluates the right-hand-side of Eq.(13)-(17) as appropriate.

Routine FFASY

This routine evaluates the expansion functions F used in the analytic expansion of the wavefunctions in the asymptotic region.

Input:

- ID=0
  only the functions are evaluated
- ID≠0
  functions and their (modified) derivatives are evaluated
- IBSG=0
  Burke-Schey expansion is assumed
- IBSG≠0
  Gailitis expansion is assumed

Routine JNBAR

This routine calculates the Jbar and Nbar functions and their derivatives with respect to R.

```
SUBROUTINE JNBAR(R,ID
+ , ET, L2I, L2P, NCOLEV, NLEVEL
+ , DJ, DN, FJ, FN
+ )
```

Input:

- R
  the R-value at which the functions are required

- ID
  If ID = 0 Jbar and Nbar only are calculated
  If ID ≠ 0 Jbar and Nbar and their derivatives are calculated

- ET
  the array of free electron energies

- L2I
  the array ...

- L2P
  the array of free electron angular momenta

- NCOLEV
  the array ...

- NLEVEL
  the variable ...

Output:

- FJ,FN
  arrays containing Jbar and Nbar, respectively

- DJ,DN
  arrays containing their derivatives

Other routines required:

- SBESSF
  a routine to calculate Ricatti-Bessel functions and their derivatives

- RICBFH
  a routine to calculate modified Bessel functions ('I' AND 'K' functions) and their derivatives

Routine MACHIN

This routine is from the Shampine and Gordon (1975 [60]) package. The output variable `U` is the machine accuracy. `TWOU` is 2 times `U` while `FOURU` is four times `U`.

Routine RAMTCH

This routine is called if RAGERB=.TRUE.. It determines the Kbar by direct matching to the asymptotic expansions at RA.

Routine RBESI

This is a re-coded, modified and restricted version of the general routine RBESI given in the package by Campbell (1981 [26]).

It calculates the modified Bessel functions:

$$\sqrt{(2\pi x)}\ \exp(-x)\ I_\nu(x) \qquad \text{for } \nu = \frac{1}{2}, \frac{3}{2}, \frac{5}{2}, \ldots, \frac{1}{2} + n \tag{13.8}$$

```
SUBROUTINE RBESI(N,X,BI,IER)
```

Input:

- N
  N+0.5 is the final value of ν required, N must be .GE. 0
- X
  X value, X must be .GT. ZERO

Output:

- IER
  an error indicator.
  IER = -1 indicates that calculations cannot be performed because of overflow or underflow problems
  IER > 0 indicates the output contained in the array BI. See below.
- BI
  The array BI contains the results as follows:
  For IER .GE. N+1
  BI(L) has the result for ν=0.5+L-1, L=1,N+1
  For IER .LT. N+1
  BI(L) has the result for ν=0.5+L-1, L=1,IER and BI(L) has the ratio of the functions for ν=0.5+L-1 and ν=0.5+L-2, L.GT.IER

Other routines required:

MACHIN is the Shampine and Gordon (1975 [60]) routine. This determines the variable U, the machine accuracy.

The dimension of array BI must be at least N+1.

The user must insert the value of DMIN appropriate to the machine. DMIN = the minimum positive number held by the machine. At present (DMIN=1.D-37).

Routine RICBFH

This routine calculates modified Bessel functions defined by:

$$\sqrt{2\pi x}\ \exp(-x)\ I_{l+\frac{1}{2}}(x) \qquad \text{and} \qquad -\sqrt{\frac{2x}{\pi}}\ \exp(x)\ K_{l+\frac{1}{2}}(x) \tag{13.9}$$

together with their derivatives for all $l = 0, 1, \ldots,$LMAX. The functions I and K are as defined by Abramowitz and Stegun, p.374.

```
    SUBROUTINE RICBFH(X,LMAX,BESSI,BESSK,DBESSI,DBESSK,ID)
```

Input:

- X
  value of variable x
- LMAX
  the maximum value of l desired
- ID
  If ID = 0, only the functions are evaluated If ID $\neq$ 0, functions and derivatives are evaluated

Output:

- BESSI,BESSK
  contain functions
- DBESSI,DBESSK
  contain the derivatives (only if input ID $\neq$ 0)

Other routines required:

RBESI
A routine to calculate $\sqrt{2\pi x}\ \exp(-x)\ I_{l+\frac{1}{2}}(x)$ for $l = 0, 1, \ldots,$LMAX

NDIM6 determines the dimensions used in the routine. It must be at least LMAX+1.

The user must insert the value of OVEFLW appropriate to the machine. OVEFLW = −0.5∗LOG(DMIN) where DMIN is the minimum positive number held by the machine. At present (OVEFLW=42.5D0)

Routine SBESSF

This routine calculates Riccati-Bessel functions:

$$x\ j_l(x) \qquad \text{and} \qquad x\ y_l(x) \tag{13.10}$$

for $x > 0$; $l = 0,$LMAX. As well as their derivatives with respect to x.

The methods used are described in Croskery et al (1982 [33]).

```
      SUBROUTINE SBESSF(LMAX,X,RBJ,RBY,DERRBJ,DERRBY,IDERIV)
```

Input:

- `LMAX`
  the maximum value of l

- `X`
  the value of x

- `IDERIV`
  If `IDERIV` $\neq 0$, all functions and derivatives are calculated.
  If `IDERIV` $= 0$, only the functions are calculated.

Output:

- `RBJ`
  the array containing $xj_l(x)$

- `RBY`
  the array containing $xy_l(x)$

- `DERRBJ`
  the array containing $\frac{d(xj_l(x))}{dx}$

- `DERRBY`
  the array containing $\frac{d(xy_l(x))}{dx}$

Element `I` in these arrays contains the result for $l = \mathtt{I} - 1$, for $l = 0, \mathtt{LMAX}$.

Accuracy of the results:

The results have a relative accuracy close to that specified by the variable: `ACCUR`. In this version of the routine `ACCUR` is set to: `U`, where `U` is a machine dependent quantity determined by the routine MACHIN. `U` is the smallest positive number such that $1 + \mathtt{U} > 1$ and is a measure of machine accuracy. If less accuracy is desired, change the statement `ACCUR` = `U` to: `ACCUR` = (user specified accuracy)

Routine MACHIN is the Shampine and Gordon (1975 [60]) routine.

`NDIM6` (which must be at least `LMAX` + 1) is the array storage dimension for the arrays `RBJ`, `RBY`, `DERRBJ`, `DERRBY`. This must be set correctly in the calling routine.

Routine SOLVE

This routine drives the integration, matching and normalisation procedures.

Routine VDE

This routine is a modified version of the one provided by Shampine and Gordon. It controls the use of routine STEP.

Routine VGAIEX

This routine transforms the Burke-Schey expansion coefficients into the corresponding coefficients appropriate to the Gailitis expansion.

```
    SUBROUTINE VGAIEX(G1,G2,JOC,NT,NTE,LV,EV,J)
```

Input:

- G1(I,N) and G2(I,N)
  Burke-Schey coefficients for all channels I and all expansion terms N for a specified channel J
- J
  specified channel
- JOC
  indicates the nature of channel J. If JOC=0 the channel is open. If JOC=1 the channel is closed.
- NT
  total number of channels
- NTE
  number of terms in the expansion
- LV,EV
  l and k^2 values for the channel set

Output:

G1 and G2 are overwritten by this routine and contain on return the Gailitis coefficients

Routine VSTEP

Routine STEP integrates a system of first order ordinary differential equations one step, normally from X to X+H, using a modified divided difference form of the Adams PECE formulae. Local extrapolation is used to improve absolute stability and accuracy. The code adjusts its order and stepsize to control the local error per unit step in a generalised sense. Special devices are included to control roundoff error and to detect when the user is requesting too much accuracy.

This code is completely explained and documented in Shampine and Gordon.

```
 SUBROUTINE VSTEP(X,Y,F,NEQN,H,EPS,WT,
+                 START,HOLD,K,KOLD,
+                 CRASH,PHI,P,YP,PSI)
```

The parameters represent:

- X
  independent variable
- Y(*)
  solution vector at X
- YP(*)
  derivative of solution vector at X after successful step
- NEQN
  number of equations to be integrated
- H
  appropriate stepsize for next step. Normally determined
- EPS
  local error tolerance. Must be variable
- WT(*)
  vector of weights for error criterion
- START
  logical variable set .TRUE. for first step, .FALSE. otherwise
- HOLD
  stepsize used for last successful step
- K
  appropriate order for next step (determined by code)
- KOLD
  order used for last successful step
- CRASH
  logical variable set .TRUE. when no step can be taken, .FALSE. otherwise

The arrays PHI,PSI are required for the interpolation routine INTRP. The array P is internal to the code.

Input to STEP

First call —

The user must provide storage in his driver program for all array in the call list, namely DIMENSION Y(NDIM),WT(NDIM),PHI(NDIM,16),P(NDIM),YP(NDIM),PSI(12) The user must also declare START and CRASH logical variables and F an external routine, supply the routine F(X,Y,YP,NDIM) with internal dimension statement:

DIMENSION Y(NDIM),YP(NDIM)

to evaluate
DY(I)/DX=YP(I)=F(X,Y(1),Y(2) — Y(NEQN))
and initialise only the following parameters:

- X
  initial value of the independent variable
- Y(*)
  vector of initial values of dependent variables
- NEQN
  number of equations to be integrated
- H
  normal stepsize indicating direction of integration and maximum size of step. Must be variable.
- EPS
  local error tolerance per step. Must be variable.
- WT(*)
  vector of non-zero weights for error criterion
- START
  .TRUE.
- NDIM
  array storage dimension

STEP requires the L2 norm of the vector with components local error(L)/WT(L) be less than EPS for a successful step. The array WT allows the user to specify an error test appropriate for his problem. For example, WT(L)

- = 1.0 specifies absolute error
- = ABS(Y(L)) error relative to most recent value of the L-th component of the solution
- = ABS(YP(L)) error relative to the most recent value of the L-th component of the derivative
- = MAX(WT(L),ABS(Y(L))) error relative to the largest magnitude of L-th component obtained so far
- = ABS(Y(L))*RELERR/EPS+ABSERR/EPS specifies a mixed relative-absolute test where RELERR is relative error, ABSERR is absolute error and EPS = MAX(RELERR, ABSERR)

Subsequent calls:

Routine STEP is designed so that all information needed to continue the integration, including the stepsize H and the order K, is returned with each step. With the exception of the stepsize, the error tolerance, and the weights, none of the parameter should be altered. The array WT must be updated after each step to maintain relative error tests like those above. Normally the integration is continued just beyond the desired endpoint and the solution interpolated there with routine INTRP. If it is impossible to integrate beyond the endpoint, the stepsize may be reduced to hit the endpoint since the code will not take a step larger than the H input. Changing the direction of integration, i.e. the sign of H, requires the user set START=.TRUE. before calling STEP again. This is the only situation in which START should be altered.

Output from STEP:

Successful step —

The routine returns after each successful step with START and CRASH set .FALSE. X represents the independent variable advanced one step of length HOLD from its value on input and Y the solution vector at the new value of X. all other parameters represent information corresponding to the new X needed to continue the integration.

Unsuccessful step —

When the error tolerance is too small for the machine precision, the routine returns without taking a step and CRASH=.TRUE. An appropriate stepsize and error tolerance for continuing are estimated and all other information is restored as upon input before returning. To continue with the larger tolerance, the user just calls the code again. A restart is neither required nor desired.

The only machine dependent constants are based on the machine unit roundoff error U which is the smallest positive number such that 1.O+U.GT.1.0. The user must calculate U and insert TWOU=2.0*U and FOURU=4.0*U. This is done using the routine MACHIN which calculates the required quantities. MACHIN is called on the first call to this routine.

Chapter 14

References

Bibliography

[1] DCS Allison, PG Burke and WD Robb *J. Phys. B: At. Mol. Phys.* **5** (1972) 55–65

[2] AR Barnett, DH Feng, JW Steed and LJB Goldfarb *Comput. Phys. Commun.* **8** (1974) 377–395
RCWFN: Coulomb functions for real eta and rho

[3] AR Barnett *Comput. Phys. Commun.* **11** (1976) 141–142
RCWFF: a modification of the real Coulomb wavefunction program RCWFN

[4] AR Barnett *Comput. Phys. Commun.* **21** (1981) 297–314
An algorithm for regular and irregular Coulomb and Bessel functions of real order to machine accuracy

[5] AR Barnett *Comput. Phys. Commun.* **27** (1982) 147–166
COULFG: Coulomb and Bessel functions and their derivatives, for real arguments, by Steed's method

[6] A Bar-Shalom and M Klapisch *Comput. Phys. Commun.* **50** (1988) 375–393
NJGRAF: an efficient program for calculation of general recoupling coefficients by graphical analysis, compatible with NJSYM

[7] KL Bell, PG Burke and AE Kingston *J. Phys. B: At. Mol. Phys.* **10** (1977) 3117–3127

[8] KL Bell and NS Scott *Comput. Phys. Commun.* **20** (1980) 447–458
Coulomb functions (negative energies)

[9] KA Berrington, PG Burke, JJ Chang, AT Chivers, WD Robb and KT Taylor *Comput. Phys. Commun.* **8** (1974) 149–198
A general program to calculate atomic continuum processes using the R-matrix method

[10] KA Berrington, PG Burke, M Le Dourneuf, WD Robb, KT Taylor and Vo Ky Lan *Comput. Phys. Commun.* **14** (1978) 367–412
A new version of the general program to calculate atomic continuum processes using the R-matrix method

[11] KA Berrington, PG Burke, K Butler, MJ Seaton, PJ Storey, KT Taylor and Yu Yan *J. Phys. B: At. Mol. Phys.* **20** (1987) 6379–6397

[12] KA Berrington, WB Eissner and PH Norrington *Comput. Phys. Commun.* **92** (1995) 290–420
RMATRX1: Belfast atomic R-matrix codes

[13] HA Bethe and EE Salpeter *Quantum Mechanics of One- and Two-electron Atoms* (Springer, 1957 — reprint Plenum 1977)

[14] LC Biedenharn, RL Gluckstern, MH Hull and G Breit *Phys. Rev.* **97** (1955) 542–554

[15] BH Bransden *Atomic Collision Theory* (2nd edition, Benjamin/Cummings, 1983)

[16] BH Bransden and CJ Joachain *Physics of Atoms and Molecules* (Longman, 1983)

[17] DM Brink and GR Satchler *Angular Momentum* (3rd edition, Oxford University Press, 1993)

[18] PG Burke *Comput. Phys. Commun.* **1** (1969) 241–250
A program to calculate a general recoupling coefficient

[19] PG Burke, A Hibbert and WD Robb *J. Phys. B: At. Mol. Phys.* **4** (1971) 153–161

[20] PG Burke and KT Taylor *J. Phys. B: At. Mol. Phys.* **8** (1975) 2620–2639

[21] PG Burke and BL Moiseiwitsch (Ed.) *Atomic Processes and Applications* (North-Holland, 1976, chap.7 p.202)

[22] PG Burke and MJ Seaton *J. Phys. B: At. Mol. Phys.* **17** (1984) L683–687

[23] PG Burke and KA Berrington *Atomic and Molecular processes: an R-matrix approach* (IOP publishing, 1993)

[24] PJA Buttle *Phys. Rev.* **160** (1967) 719–729

[25] F Calogero *Variable Phase Approach to Potential Scattering* (Academic Press, 1967)

[26] JB Campbell *Comput. Phys. Commun.* **24** (1981) 97–105

[27] JJ Chang *J. Phys. B: At. Mol. Phys.* **8** (1975) 2327–2335
The R-matrix theory of electron-atom scattering using the Dirac Hamiltonian

[28] JJ Chang *J. Phys. B: At. Mol. Phys.* **10** (1977) 3195

[29] ER Cohen and BN Taylor *CODATA Bull.* (1986) no. 63

[30] MA Crees, MJ Seaton and PMH Wilson *Comput. Phys. Commun.* **15** (1978) 23–83

[31] MA Crees *Comput. Phys. Commun.* **19** (1980) 103–137
ASYPCK, a program for calculating asymptotic solutions of the coupled equations of electron collision theory

[32] MA Crees *Comput. Phys. Commun.* **23** (1981) 181–198
ASYPCK2, an extended version of ASYPCK

[33] JP Croskery, NS Scott, KL Bell and KA Berrington *Comput. Phys. Commun.* **27** (1982) 385–401
VPM: a new asymptotic package

[34] KG Dyall, IP Grant, CT Johnson, FA Parpia and EP Plummer *Comput. Phys. Commun.* **55** (1989) 425–456 *erratum* **58** (1990) 345
GRASP: a general-purpose relativistic atomic structure program

[35] G Goertzel *Phys. Rev.* **73** (1948) 1463

[36] IP Grant *Comput. Phys. Commun.* **4** (1972) 377–381
CFPJJ – fractional parentage coefficients for equivalent electrons in jj-coupling

[37] IP Grant *Comput. Phys. Commun.* **5** (1973) 263–282
A general program to calculate angular momentum coefficients in relativistic atomic structure

[38] IP Grant *J. Phys. B: At. Mol. Phys.* **7** (1974) 1458–1475
Gauge invariance and relativistic radiative transitions

[39] IP Grant, BJ McKenzie, PH Norrington, DF Mayers and NC Pyper *Comput. Phys. Commun.* **21** (1980) 207
An atomic multi-configurational Dirac-Fock package

[40] RJW Henry and L Lipsky *Phys. Rev.* **153** (1967) 51

[41] DG Hummer, KA Berrington, W Eissner, AK Pradhan, HE Saraph and JA Tully *Astron. and Astrophys.* **279** (1993) 298–309

[42] VL Jacobs *J. Phys. B: At. Mol. Phys.* **5** (1972) 2257

[43] NM Kabachnik and IP Sazhina *J. Phys. B: At. Mol. Phys.* **9** (1976) 1681

[44] YL Luke *The Special Functions and their Applications Vol.2* (Academic Press, 1969)

[45] BJ McKenzie, IP Grant and PH Norrington *Comput. Phys. Commun.* **21** (1980) 233 *erratum* **23** (1981) 222
A program to calculate transverse Breit and QED corrections to energy levels in a multi-configurational Dirac-Fock environment

[46] CJ Noble and RK Nesbet *Comput. Phys. Commun.* **33** (1984) 399–411
CFASYM: a program for the calculation of the asymptotic solutions of the coupled equations of electron collision theory

[47] DW Norcross *Comput. Phys. Commun.* **1** (1969) 88–96

[48] DW Norcross and MJ Seaton *J. Phys. B* **6** (1973) 614

[49] Opacity Project Team *The Opacity Project Volume 1* (Institute of Physics, 1995)

[50] FA Parpia, C Froese Fischer and IP Grant *Comput. Phys. Commun.* **94** (1996) 249
GRASP92: a package for large-scale relativistic structure calculations

[51] JP Pelan *The IRON Project, World Wide Web homepage* (1996)
http://www.am.qub.ac.uk/projects/iron/

[52] ME Rose *Elementary Theory of Angular Momentum* (Wiley, 1957)

[53] NS Scott and PG Burke *J. Phys. B: At. Mol. Phys.* **13** (1980) 4299–4314

[54] NS Scott *Relativistic Effects In Electron Scattering and Photoionization of Atoms and Ions* (PhD Thesis, The Queen's University of Belfast, 1980)

[55] NS Scott and A Hibbert *Comput. Phys. Commun.* **28** (1982) 189–200
A more efficient version of the WEIGHTS and NJSYM packages

[56] NS Scott and KT Taylor *Comput. Phys. Commun.* **25** (1982) 347–387
A general program to calculate atomic continuum processes incorporating model potentials and the Breit-Pauli Hamiltonian within the R-matrix method

[57] MJ Seaton *J. Phys. B: At. Mol. Phys.* **18** (1985) 2111–2131

[58] MJ Seaton *J. Phys. B: At. Mol. Phys.* **19** (1986) 2601–2610

[59] MJ Seaton *J. Phys. B: At. Mol. Phys.* **20** (1987) 6363–6378

[60] LF Shampine and MK Gordon *Computer Solution of Ordinary Differential Equations: The Initial Value Problem* (Freeman, 1975)

[61] P Shorer *J. Phys. B: At. Mol. Phys.* **13** (1980) 2921–2929
An R-matrix approach to the calculation of dynamic dipole polarisabilities and two-photon photoionisation cross sections

[62] R Szmytkowski and J Hinze *J. Phys. B: At. Mol. Phys.* **29** (1996) 761–777
Convergence of the non-relativistic and relativistic R-matrix expansions at the reaction volume boundary

[63] KT Taylor *Photoionization of Atomic Systems* (PhD Thesis, The Queen's University of Belfast, 1975)

[64] TEH Walker and JT Waber *J. Phys. B: At. Mol. Phys.* **6** (1973) 1163

[65] EP Wigner and L Eisenbud *Phys. Rev.* **72** (1947) 29–41

[66] IG Young *The Solution of Relativistic Asymptotic Equations in Electron-Ion Scattering* (PhD Thesis, The Queen's University of Belfast, 1992)

[67] IG Young and PH Norrington *Comput. Phys. Commun.* **83** (1994) 215