7. Program BSR_MAT

7.1. Outline of the BSR_MAT calculation

This program generates the interaction matrices (2.17) in the *B*-spline representation. It can be done either in *LS*-coupling or in some intermediate coupling scheme with direct inclusion of selected Breit-Pauli terms. The eigenproblem (2.17) in the *B*-spline basis leads to a generalized eigenvalue problem of the form

$$\boldsymbol{Hc} = E\boldsymbol{Sc} \tag{7.1}$$

The solution vector, $\boldsymbol{c}$, can be written as

$$\boldsymbol{c} = [\boldsymbol{a}_1,...,\boldsymbol{a}_{N_c},\boldsymbol{b}]^T \tag{7.2}$$

where each $\boldsymbol{a}_i$ is a column vector of *B*-spline coefficients for the given channel functions $\overline{\Phi}_i^{\Gamma}$

$$\boldsymbol{a}_i = [a_{1i}, a_{2i},..., a_{n_s i}]^T \tag{7.3}$$

and $\boldsymbol{b}$ is the column vector of correlation functions coefficients

$$\boldsymbol{b}_j = [b_1, b_2,..., b_{N_p}]^T . \tag{7.4}$$

Here N_c is the number of channels while N_p is the number of correlation functions. Schematically, the interaction matrix $\boldsymbol{H}$ has the form

$$\begin{pmatrix} H(11) & H(12) & \cdots & H(1N_c) & h(1p) \\ H(21) & H(22) & \cdots & H(2N_c) & h(2p) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ H(N_c1) & H(N_c2) & \cdots & H(N_cN_c) & h(N_cp) \\ h(1p)^T & h(2p)^T & \cdots & h(N_cp)^T & h(pp) \end{pmatrix} \tag{7.5}$$

where $h(pp)$ is an $N_p \times N_p$ matrix that comes from the bound-bound interaction, the $h(ip)$ are $n_s \times N_p$ matrices representing the interaction between the i-th channel and the bound states, and $H(ij)$ are $n_s \times n_s$ matrices for channel-channel interaction. Furthermore, $\boldsymbol{S}$ in equation (7.1) is the overlap matrix. In the usual case of orthogonal conditions imposed on the scattering orbitals, it reduces to a matrix consisting of diagonal blocks of the banded overlap matrix $\boldsymbol{B}$ between individual *B*-splines B_i

$$B_{ij} = \langle B_i \mid B_j \rangle , \tag{7.6}$$

one for each channel. In the more general case of non-orthogonal orbitals, however, it has a more complicated structure (see table 7.1).

Each individual matrix element in (7.5) is expressed by the program BSR_BREIT in the form

$$\sum c_i L(a,b) \times D(\{nl\};\{n'l'\}) + \sum c_i R^{\lambda}(a,b;c,d) \times D(\{nl\};\{n'l'\}) \tag{7.7}$$

where the c_i are numeric coefficients which depend only on the angular symmetry of the involved CSFs, L stands for one-electron integrals, R^{λ} are Slater integrals, and $D(\{nl\},\{n'l'\})$ is the overlap factor, which depends only on the orthogonality of the radial orbitals used in the construction of the CSFs. In general, the overlap factor is the multiplier of determinants of matrices consisting of one-electron overlaps

$$\begin{vmatrix} < n_1 l \mid n'_1 l > & < n_1 l \mid n'_2 l > & \dots & < n_1 l \mid n'_q l > \\ < n_2 l \mid n'_1 l > & < n_2 l \mid n'_2 l > & \dots & < n_2 l \mid n'_q l > \\ \dots & \dots & \dots & \dots \\ < n_q l \mid n'_1 l > & < n_q l \mid n'_2 l > & \dots & < n_q l \mid n'_q l > \end{vmatrix} \tag{7.8}$$

The scattering orbitals can appear in the radial integrals L and R^{λ}, and also in the determinant factors. To derive the final expression for the matrix elements in the B-spline basis, let us first simplify the overlap factors (7.8) by expanding them over those row (or column) that contain one-electron overlaps with the scattering orbitals. At most, there can be one such row or/and collumn. The residual overlap determinants depend only on the known bound orbitals and they can be calculated in a standard manner. It is convenient to redefine the angular coefficients c_i. multiplying them by these residual overlap factors. As a result, the initial overlap factor $D(\{nl\},\{n'l'\})$ in expression (7.7) is reduced to one- or two-electron overlaps of the form

$$\langle kl \mid nl \rangle, \quad \langle kl \mid k'l \rangle \quad \text{or} \quad \langle kl \mid nl \rangle \langle n'l' \mid k'l' \rangle, \tag{7.9}$$

where $|kl>$ stands for the radial function of the continuum orbital in the inner region

$$u_{kl}(r) = \sum_i a_i B_i(r). \tag{7.10}$$

Each term in (7.7) gives rise to a matrix in the spline basis. To elucidate the structure of these matrices, let us introduce, in addition to the overlap B-matrix defined in equation (7.6), $n_s \times n_s$ matrices with elements

$$L(\bullet\bullet)_{ij} = \langle B_i \mid \frac{\mathrm{d}^2}{\mathrm{d}r^2} - \frac{l(l+1)}{r^2} + \frac{2Z}{r} \mid B_j \rangle$$

$$R^{\lambda}(\bullet a\, b\, \bullet)_{ij'} = \sum_{i'j} a_{i'} b_{j'} R^{\lambda}(ij; i'\, j') \tag{7.11}$$

$$R^{\lambda}(\bullet\, a \bullet\, b)_{ij} = \sum_{jj'} a_{i'} b_{j'} R^{\lambda}(ij; i'\, j')$$

where

$$R^{\lambda}(ij, i'\, j') = \langle B_i(r_1) B_j(r_2) \mid U_{12}^{\lambda} \mid B_{i'}(r_1) B_{j'}(r_2) \rangle \tag{7.12}$$

Here U_{12}^{λ} is the radial part of the λ'th term in the multipole expansion of $1/r_{12}$, and a_i, b_j are the expansion coefficients in the spline basis for the bound orbitals $P_a(r)$, $P_b(r)$ as in (7.10). The four-dimensional array, $R^{\lambda}(ij;i'j')$, effectively represents the Slater integral in the *B*-spline basis. The integrals are zero if either $|i-i'| \geq k_s$ or $|j-j'| \geq k_s$. Methods for computing these quantities, which depend only on the basis, are given in Section 12. In order to describe the channel-bound interaction, let us introduce the following vectors with elements

$$B(\bullet a)_i = \sum_{j=1}^{n_s} a_j B_{ij} \tag{7.13}$$

$$L(\bullet\, a)_i = \sum_{j=1}^{n_s} a_j L_{ij} \tag{7.14}$$

$$R^{\lambda}(\bullet\, a\, b\, c)_i = \sum_{ji'j'=1}^{n_s} a_j b_{i'} c_{j'} R^{\lambda}(ij; i'\, j') \tag{7.15}$$

With this notation, table 7.1 represents the contribution of the different terms from the energy expression (7.7), which is usually produced by the angular integration codes, into the interaction matrix (7.5), along with an indication of its structure. Table 7.1 also indicates the contribution to the total overlap matrix arising from the matrix element on the right hand side of equation (7.1). Note that the target wave functions are assumed to form an orthogonal set and diagonalize the target Hamiltonian. Hence, of all the matrix elements containing $\langle kl|k'l\rangle$, only the diagonal ones, with $k=k'$, contribute to the interaction matrix. The same concerns the matrix elements with $L(kl,k'l)$.

Table 7.1. Contribution of different terms (without indication of angular coefficients) in the interaction matrix. The symbols *a,b,c,d* stand for the bound orbitals, while k_i indicates the continuum orbital in channel *i*.

Term	Contribution	Remarks
Channel-channel interaction		
$R^\lambda(a,k_i;b,k_j)$ $R^\lambda(k_i,a;k_j,b)$	$R^\lambda(\,a\,.\,b\,.)$ $R^\lambda(\,.\,a\,.\,b)$	Banded matrix, direct interaction
$R^\lambda(k_i,a;b,k_j)$ $R^\lambda(a,k_i,;k_j,b)$	$R^\lambda(\,.\,a\,b\,.)$ $R^\lambda(\,a\,.\,.\,b)$	Full matrix, exchange interaction
$R^\lambda(k_i,a;b,c)\ \langle d \mid k_j\rangle$ $R^\lambda(a,k_i;b,c)\ \langle d \mid k_j\rangle$ $R^\lambda(a,b;k_i,c)\ \langle d \mid k_j\rangle$ $R^\lambda(a,b;c,k_i)\ \langle d \mid k_j\rangle$	$R^\lambda(\,.\,a\,b\,c)\bullet B(\,.\,d)$ $R^\lambda(\,a\,.\,b\,c)\bullet B(\,.\,d)$ $R^\lambda(\,a\,b\,.\,c)\bullet B(\,.\,d)$ $R^\lambda(\,a\,b\,c\,.)\bullet B(\,.\,d)$	Full matrix, contribution due to non-orthogonality
$\langle k_i \mid a\rangle\,\langle b \mid k_j\rangle\,R^\lambda$(or L)	$B(\,.\,a)\bullet B(\,.\,b)\,*\,R^\lambda$ (or L)	Full matrix, contribution due to non-orthogonality
$\langle k_i \mid k_j\rangle\,R^\lambda$ (or L)	$B(\,.\,.)\,*\,E_i$	Banded matrix, only for $i = j$
$L(k_i, k_j)$	$L(\,.\,.)$	banded matrix, only for $i = j$
$L(k_i,a)\,\langle b \mid k_j\rangle$	$L(\,.\,a\,)\bullet B(\,.\,b)$	full matrix, contribution due to non-orthogonality
$\langle k_i \mid k_j\rangle$	$B(\,.\,.)$	banded matrix, only for $i = j$, contribution to overlap matrix
$\langle k_i \mid a\rangle\,\langle b \mid k_j\rangle$	$B(\,.\,a)\bullet B(\,.\,b)$	full matrix, contribution to overlap matrix due to non-orthogonality
Channel-bound interaction		
$R^\lambda(k_i,a;b,c)$	$R^\lambda(\,.\,a\,b\,c)$	vector
$R^\lambda(a,k_i;b,c)$	$R^\lambda(\,a\,.\,b\,c)$	
$R^\lambda(a,b;k_i,c)$	$R^\lambda(\,a\,b\,.\,c)$	
$R^\lambda(a,b;c,k_i)$	$R^\lambda(\,a\,b\,c\,.)$	
$L(k_i,a)$	$L(\,.\,a\,)$	vector
$\langle k_i \mid a\rangle\,R^\lambda$(or L)	$B(\,.\,a\,)\,*\,R^\lambda$ (or L)	vector, contribution due to non-orthogonality
$\langle k_i \mid a\rangle$	$B(\,.\,a\,)$	vector, contribution to overlap matrix
Bound-bound interaction		

$R^{\lambda}(a, b; c, d)$	scalar
$L(a, b)$	scalar
c	scalar, contribution to overlap matrix

7.2. Structure and data flow.

The block diagram of the program BSR_MAT, along with the data flow, is shown in Fig.7.1. First, the program reads the data common to all partial waves. It includes the parameters for the calculation (routine **read_arg**), the *B*-spline parameters (**define_spline**), and the description of the target states **(read_target**). Then the program executes fully independent calculations for each partial wave. First, the program reads the configuration list from the relevant *c*-file **cfg.nnn** and allocates all main arrays with dimensions specific for a given partial wave. The most critical dimension here is the size of the interaction matrix, which usually occupies most of the memory. The size of the interaction matrix is mainly defined by the product of the number of scattering channels and the number of *B*-splines, because usually the (*N*+1)-electron bound channels in the close-coupling expansion are either absent or their number is relatively small in the present approach. In order to speed up the calculation, we place the entire interaction matrix in the RAM memory. On the other side, this limits the number of scattering channels which we can include in the physical model by the size of the computer memory. (Note that virtual memory will not help much here.)

The following calculations of the matrix elements are performed for each type of integrals sequentially. This part is controlled by the routine **state_res**, whose flow diagram is placed at the bottom of Fig.7.1. Each time this routine reads the relevant **int_bnk** file from the beginning, chooses the integrals under consideration and provides their contribution to the total interaction matrix. Such a procedure reduces the number of integrals which should be considered at one time, and this increases the effectiveness of their processing. One-electron and two-electron integrals are considered in different ways. For one-electron integrals, such as kinetic-energy *L*-integrals (7.11) or spin-orbit *Z*-integrals (2.47), the program first calculates all possible integrals and their *B*-spline representation for all one-electron orbitals in the configuration list, employing the routines **gen_Lval** and **gen_Zval**. The relevant data are stored in the modules **L_core** and **Z_core**. Their storing requires only two-dimension arrays and can be handled even for a large number of one-electron orbitals. For the two-electron integrals, their storing requires four-dimensional arrays. This cannot be realized in the case of a large number of different one-electron orbitals, which often occur in the present approach. Hence, the two-electron integrals are

calculated dynamically when they are required. In contrast to the standard *R*-matrix approach, the present implementation uses a larger number of non-orthogonal orbitals for a more accurate representation of the target states. Typical calculations may involve up to several hundred of different one-electron radial functions. The present code has no practical limit on the maximum number of different one-electron orbitals that may be used to describe the target and the scattering states.

The list of specific integrals needed for the given configurations is generated dynamically from the information in the databank **int_bnk**. The integrals are then sorted according to their structure (see table 7.1) and stored in the module **cmdata** in different ordered blocks, using routines **check_det**, **idef_type, add_coef**, and **add_cdata**. The number and size of the individual blocks is defined by the input parameters and could be adjusted to the speed and memory of a given computer. When all blocks are full, they are merged, if necessary, with the routine **merge_cdata** and then are processed by the routine **gen_matrix** to get the *B*-spline representation for specific integrals (using routines **O_data**, **L_data**, **Z_data** and **I_data**; the latter deals with two-electron integrals). All these routines extensively use subroutines from the BSPLINE library. The corresponding contributions are then added to the interaction matrix, located in module **bsr_matrix**.

The next step is the correction of the interaction matrix according to the orthogonal conditions imposed on the scattering orbitals, using the routine **BS_orth**. This is done according to the procedure suggested by Bentley [55]. Suppose a continuum orbital, $u_{kl}(r)$, in some *i*-th channel is orthogonal to the bound orbital $P_{nl}(r)$. This can be accounted for by modifying the Hamiltonian according to

$$\boldsymbol{H} \rightarrow (1 - \boldsymbol{B}\boldsymbol{c}\boldsymbol{c}^T)\boldsymbol{H}(1 - \boldsymbol{c}\boldsymbol{c}^T\boldsymbol{B}) \qquad (7.16)$$

where $\boldsymbol{c}$ is the full solution vector (7.2) with all zero elements, except that the channel *i* is replaced by the *B*-spline expansion $\{p_i\}$ of the orbital $P_{nl}(r)$. This leads to transformation of only those blocks $H(i,j)$ in (7.5) which have one index equal to *i*. Recall that the orthogonal conditions to the common core orbitals are implied automatically in the present implementation. All other conditions should be specified through input data in the **bsr_par** or **cfg.nnn** files.

At the last step, the interaction matrix, along with the long-range potential coefficients, is recorded in the relevant **bsr_mat.nnn** file. The above calculations are repeated for each partial wave under consideration.

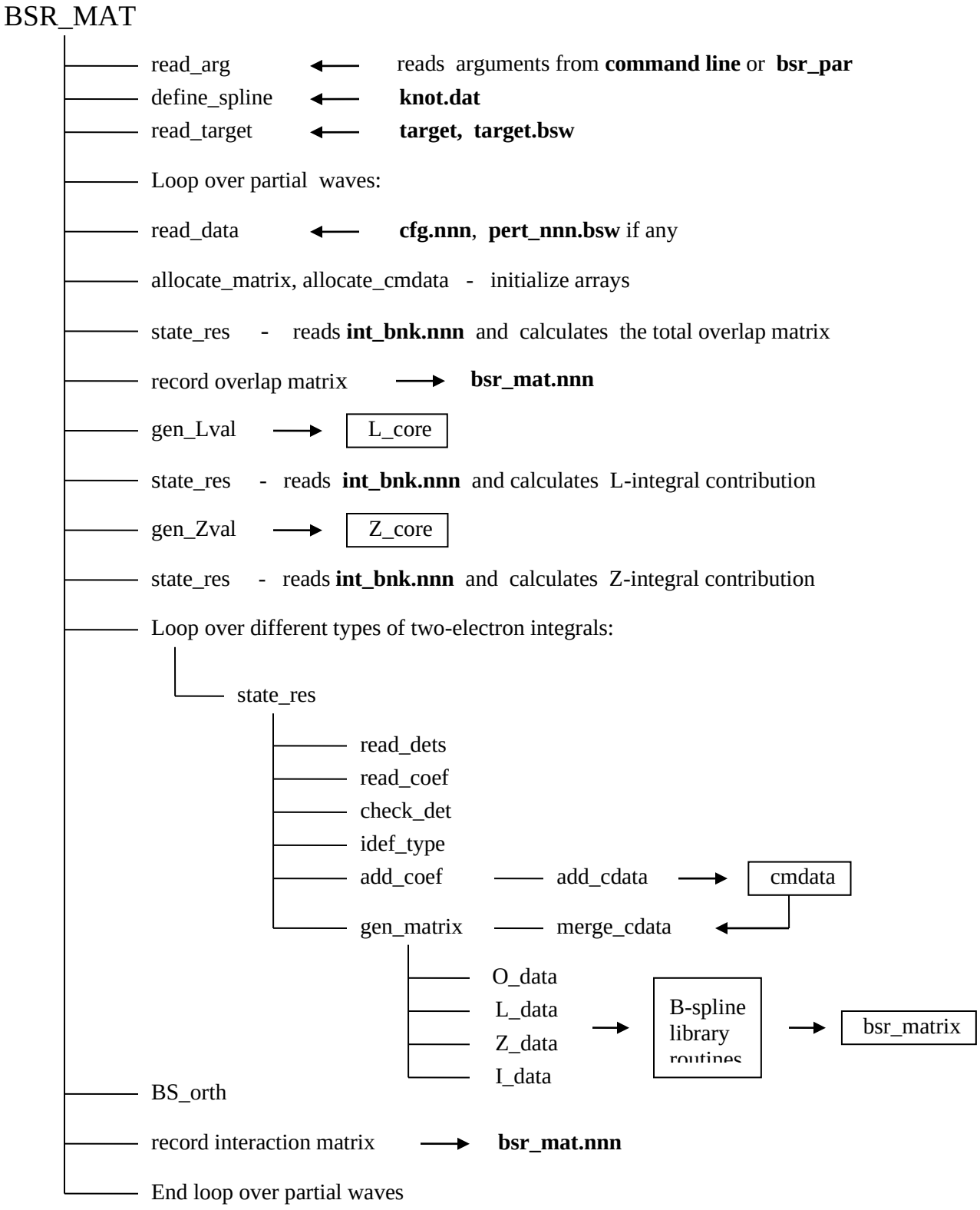

Fig. 7.1. Block diagram for the program BSR_MAT and data flow (see text).

7.3. Data files

bsr_par	File type: formatted sequential input. Written by user. Read by routine **read_arg**. Description: input parameters for given run.
target	File type: formatted sequential input. Written by user and modified by BSR_PREP and BSR_CONF programs. Read by routine **read_target**. Description: contains description of target states and scattering channels.
target_orb	File type: formatted sequential input. Created by BSR_PREP. Read by routine **read_target**. Description: contains list of physical orbitals for all target states and the corresponding substitution orbitals used for orthogonal constraints if needed.
knot.dat	File type: formatted sequential input. Written by user. Read by routine **define_grid** from BSPLINE library. Description: input parameters that define the *B*-spline grid.
cfg.nnn	File type: formatted sequential input Created by program BSR_CONF. Read by routine **read_conf**. Description: list of configurations for given partial wave nnn.
int_bnk.nnn	File type: unformatted sequential input. Created by program BSR_BREIT. Read by routine **state_res**. Description: databank for angular coefficients.
target.bsw	File type: unformatted sequential input. Created by program BSR_PREP. Read by routine **read_bsw**. Description: target's one-electron orbitals in *B*-spline basis.
pert_nnn.bsw	File type: unformatted sequential input. Created by program BSR_PREP. Read by routine **read_bsw**. Description: perturber's one-electron orbitals in *B*-spline basis, optional.
int_mat.nnn	File type: formatted sequential output. Created by program BSR_MAT. Description: list of one- and two-electron integrals. This output is only for debug purposes and controlled by the input parameter **nud**.
bsr_mat.log	File type: formatted sequential output. Written by program BSR_MAT. Read by user. Description: parameters description common for all partial waves.
mat_log.nnn	File type: formatted sequential output. Written by program BSR_MAT. Read by user. Description: running information.
bsr_mat.nnn	File type: unformatted sequential output **(main result)** Created by program BSR_MAT. Read by program BSR_HD. Description: **Hamiltonian matrix and asymptotic coefficients.**

7.4. Input data

Input data can be provided in the command line or in the input file **bsr_par** (the data from the command line overwhelm the data from input file). Below we describe those data from **bsr_par** which are read by program BSR_MAT. All data have the default values indicated in the brackets. All default values, along with unit numbers and default file names, are placed in the module **bsr_mat**.

klsp1 [1] — first partial wave under consideration.

klsp2 [nlsp] — last partial wave under consideration.

klsp [1] — partial wave under consideration (overwrites **klps1** and **klsp2**) .

mk [7] — maximum multipole index in two-electron integrals.

mb [5000] — the size of one block in module **c_data** used for accumulation of given type of integrals.

nb [1000] — number of blocks in module **c_data.**

kb [10] — number of blocks for one type of integrals in module **c_data.**

maxnc [100000] — size (in records) of buffer for coefficients read from **int_bnk**. The buffer was introduced in order to reduce the number of read-write operations. It is especially important if several **bsr_mat** programs are run with a single disk.

mrel [0] — indicates the terms from Breit-Pauli Hamiltonian which should be included

`mrel=1` – include only one-electron scalar relativistic integrals.
`mrel=2` – include additionally one-electron spin-orbit interaction.
`mrel=3` – plus two-electron spin-other-orbit interaction.
`mrel=4` – plus spin-spin interaction.
`mrel=5` – plus orbit-orbit interaction.

mso [0] — controls the inclusion of one-electron spin-orbit interaction. If set to 0, then a decision will be made according to the **mrel** parameter; if set to -1(1), then the spin-orbit interaction will be excluded (included) anyway.

msoo [0] — controls the inclusion of the two-electron spin-other-orbit interaction in the same way as the **mso** parameter.

mss [0] — controls the inclusion of the spin-spin interaction in the same way as the **mso** parameter.

moo [0] — controls the inclusion of the orbit-orbit interaction in the same way as the **mso** parameter.

imvc[-2] — indicates the mode for processing the mass-velocity term:

`imvc=+1` – include mass-velocity term directly in the Hamiltonian (7.5) (recommended only for light atoms, $Z < 5$).
`imvc=-1` – include the mass-velocity term only for the bound orbitals. For scattering orbitals this interaction can then be included in the BSR_HD program as a first-order correction.

	`imvc=-2` – the same as imvc= -1 but additionally exclude the mass-velocity term for bound orbitals with principal quantum number > **nmvc**. This mode was introduced for consistency in cases when some bound orbitals were generated in previous bound-state BSR calculations without inclusion of relativistic corrections.
nmvc[0]	the principal quantum number of orbitals for which the mass-velocity term is not included (see the `imvc=-2` mode above).
nud [0]	if > 0, the debug file **int_mat.nnn** with list of integrals will be created.
pri_f [0]	if > 0, the radiative data (f-values and A-values) for transitions between target states will be output in file **mat_log.nnn .** The same information can be got with utility program **f_values .**
pri_ac [0]	if > 0, the asymptotic coefficients will be output in file **mat_log.nnn .**
debug [0]	if > 0, some additional timing information is provided.
eps_c [1.d-10]	tolerance for angular coefficients combined with numerical overlap factor. Coefficients less than **eps_c** are ignored.
eps_det [1.d-10]	tolerance for the value of the overlap factor. Overlap factors less than **eps_det** are taken as zero.
eps_ovl [1.d-8]	tolerance for the value of the on-electron overlaps. Overlap less than **eps_ovl** are taken as zero.
eps_soo [1.d-2]	tolerance for the angular coefficients for the two-electron Breit-Pauli terms. The treatment of these terms may take most of the time. On the other hand, they give only small corrections, and thus this parameters allows us to consider contributions only from the principal configurations.
eps_tar [1.d-6]	tolerance for target energies and overlaps to provide warning messages
eps_acf [1.d-5]	tolerance for the value of the asymptotic coefficients. Values less than **eps_acf** are taken as zero.
zcorr [1.0]	semi-empirical correction for the spin-orbit integrals
izcorr [1]	if =1, the cut-off procedure for the spin-orbital integrals near nuclear is applied
s_pert [0.5]	tolerance for overlaps between perturbers to provide warning messages
s_ovl [0.75]	tolerance for overlaps between channel functions to impose additional orthogonal conditions
iitar [0]	`iitar = 0` – target states are supposed to be orthogonal eigenfunctions of the target Hamiltonian `iitar = 1` – target states are supposed to be orthogonal but may not diagonalize the target Hamiltonian `iitar = 2` – target states may be not orthogonal and may not diagonalize the target Hamiltonian

7.5 *Output of Hamiltonian matrices and asymptotic coefficients*

The main result of the **bsr_mat** program is the generation of the interaction matrix (2.17) in the *B*-spline representation (7.5). The latter is stored in the **bsr_mat.nnn** files for each partial wave separately. Below is a summary of the output records in these files.

1. Main dimensions:
   `ns,nch,kcp`

   `ns` – number of *B*-splines.
   `nch` – number of channels.
   `ncp` – number of bound (*N*+1)-electron configurations.
   The total dimension of interaction matrix is equal `nhm=ns×nch+ncp`.

2. Diagonal blocks of overlap matrix:
   `hcc(:,:,i*(i+1)/2)` – repeat for *i*=1,nch

3. Non-diagonal blocks of overlap matrix:
   `i,j`
   `hcc`(:,:) - if $i,j \le$ nch (channel-channel blocks)
   hcb(:) - if $i >$ nch (channel-bound block)
   hbb - if $i,j >$ nch (bound-bound block)

   if $i \neq 0$ - repeat the record 3

4. Diagonal blocks of Hamiltonian matrix:
   `hcc(:,:,i*(i+1)/2)` – repeat for *i*=1,nch

5. Non-diagonal blocks of Hamiltonian matrix:
   `i,j`
   `hcc`(:,:) - if $i,j \le$ nch (channel-channel blocks)
   hcb(:) - if $i >$ nch (channel-bound block)
   hbb - if $i,j >$ nch (bound-bound block)

   if $i \neq 0$ - repeat the record 5

6. `mk` - maximum multipole index.

7. `CF`(1:nch,1:nch,0:mk) – matrix of asymptotic coefficients (2.26).

8. `t(ns+1), ns`

   Array `t(1:ns+ks)` is the set of knot points used for the generation of *B*-splines.
   Value `t(ns+1)` defines the maximum radius for the *B*-splines, i.e. the *R*-matrix radius.

7.6 *Version 3 updates.*

The BSR_MAT3 version provide new options in comparison to the original version. Most important is the **s_ovl** amd **iitar** parametrs. The program now additionally check the orthogonality of the possible channel functions. If overlaps exceed **s_ovl**, additional orthogonality constraint are imposed. Default value **s_ovl=0.75** was chosen as optimal for the scattering calculations. For the bound calculations we can increase this value up to 0.999 (recommended).

Additional option is the possibility to use non-orthogonal target states in the close-coupling expansions (**iitar = 1**). If **iitar=2**, target states may not diagonalize the target Hamiltonian. In this case, program will create target_new.nnn files with new target energies, obtained after diagonalization of target Hamiltonian. These options provide more flexibility in generation target states from the independent calculation. These options, however, are recommended to use only for bound state calculations, because is not clear how introduce new target energies in H.DAT files and in following scattering calculations.

Example of the **mat_log.nnn** file is given in Fig. 7.1. It provide main parameter for given partial wave (recall that the parameters which are common for all partial wave are given in the **bsr_mat.log** file together with their short description) . Along with parameters, the **mat_log.nnn** file provide the main dimensions for the allocatable arrays, the required memory and the timing for calculations of different integrals in the Hamiltonian matrix. Important is the possible warning about the correctness of target expansions in the end of file. As a side product, the BSR_MAT calculates the target Hamiltonian. By default (**iitar=0**) it should be represented by diagonal matrix, with target energies as the diagonal matrix elements. If calculated Hamiltonian matrix is different, the program provides the corresponding warnings with the indication of incorrect target states. The user is advice to check these possible warning before to go to next step.

As another side product, the BSR_MAT program may provide the radiative data for dipole-allowed transitions between target states. The output of the corresponding f- and A-values is indicated by parameter **pri_f**. The user also can additionally output the asymptotic coefficients, using the parameter **pri_ac**.

```
BSR_MAT:   klsp =     1
************************

c-file data:

ncfg   =    149  -  number of configurations
nwf    =     31  -  number of orbitals:


Partial wave LSP (JP):

lpar   =      0  -  total L
ispar  =      2  -  total 2S+1
ipar   =      1  -  parity

Main dimensions in bsr_matrix module:

kch    =      4  -  number of channels
kcp    =      1  -  number of perturbers
kns    =    107  -  number of splines

Interaction matrix dimension:        429

Reqired memory for bsr-matrix:       0.7 Mb

Reqired memory for cmdata:         267.2 Mb

Reqired memory for bufer:           19.1 Mb

ndet  =       90     161
ndef  =     2827    8640

Checking the overlap matrix for overlaps > S_ovl = 0.750

Overlaps:           0.00 min

L-integrals:        0.00 min

R-integrals:        0.03 min

Orthogonal conditions:

ks4  ->      1s  2s1
kp4  ->     2p1
kp5  ->     2p1

BS_ORTH:            0.00 min

Asymptotic coefficients: mk =  7

Target hamiltonian errors if > eps_tar =  1.0E-06

Target overlaps errors if > eps_tar =  1.0E-06

Total time:         0.04 min
```

Fig.7.1 Example of the mat_log.nnn file after BSR_MAT run.

7.6. MPI version

The MPI version, BSR_MAT3_MPI, employs the same algorithm for the calculation of angular coefficients as the serial version, and used the same input parameters and the same structure for the databank. Parallelization consists in the parallel calculations of different blocks in the Hamiltonian matrix, see Eq. 7.5. In BSR_MAT3_MPI, we use one processor as the master which organizes the calculations and records the results. The master broadcast all needed parameters and assigns the Hamiltonian blocks to be considered on the specific slave processor, based on number of processors. Then the master reads, by blocks, the angular coefficients from **int_bnk** and broadcast them to all processors. The each slave processor picks the angular coefficient which contribute to the local blocks and carries out the needed calculations to update the Hamiltonian matrix. After checking whole **int_bnk** database, the master receives, block by block, from the corresponding slave the final Hamiltonian matrix and records it in the **bsr_mat.nnn** file. The same procedure is employed for the asymptotic coefficients. Note that in this way, the whole Hamiltonian matrix is spread over all processors.

7.7. Further development (version 4)

The BSR_MAT4 version includes two main modifications: (i) using the angular coefficient databank after BSR_BREIT4; (ii) preparing for the extremely large-scale case and to provide possibility to generate Hamiltonian matrix in parts. It require new parameters:

time [0] - limit on time of calculations.

If parameter time is not zero, the program stops the calculations and record the partially calculated Hamiltonian matrix in file **bsr_mat.nnn**. This file, in the last record, also contains the information indicating at what stage of calculations the program has been stopped. The user can then adjust parameters or number of processors and re-run the BSR_MAT4 to continue the matrix generation. The time parameter allow to avoid the loss of data when program was terminated by the system due the time limits (what frequently happens in the supercomputer environment) .

mode [0] – if $\neq 0$, it indicates that we continue the matrix generation.