ELSEVIER

Computer Physics Communications 104 (1997) 245–258

Computer Physics Communications

Fortran code of the Projected Shell Model: feasible shell model calculations for heavy nuclei

Yang Sun [a,b,c,1], Kenji Hara [d,2]

[a] *Joint Institute for Heavy Ion Research, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA*
[b] *Department of Physics and Astronomy, University of Tennessee, Knoxville, TN 37996, USA*
[c] *Physics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA*
[d] *Physik-Department, Technische Universität München, D-85747 Garching bei München, Germany*

Received 19 March 1997; revised 28 May 1997

Abstract

A Fortran program is presented which conforms with the framework of the Projected Shell Model. The theory is a genuine shell model configuration mixing approach but requires only a very small configuration space. This feature enables us to interpret numerical results in simple physical terms. The present code allows detailed spectroscopic calculations for low- and high-spin states in axially deformed nuclei. It is written using a very efficient algorithm published earlier and runs extremely fast not only on Mainframes but also on Workstations or even on modern PCs. © 1997 Elsevier Science B.V.

PACS: 21.60.Cs; 21.10.Re
Keywords: Projected shell model; Configuration mixing; Rotational symmetry; Deformed nuclei; Spectroscopic calculations; Nuclear high-spin states

PROGRAM SUMMARY

Title of program: PSM_EE, PSM_EO, PSM_OE, PSM_OO

Catalogue identifier: ADGJ

Program obtainable from: CPC Program Library, Queen's University of Belfast, N. Ireland

Licensing provisions: none

Computer for which the program is designed and others on which it is operable: Mainframes, Workstations, PCs (486 and Pentium)

Operating systems under which the program has been tested: UNIX, Windows 95, LINUX, DOS

Programming language used: FORTRAN 77

Memory required to execute with typical data: 16 MB

No. of bytes in distributed program, including test data, etc.: 893741

Distribution format: ASCII

Keywords: Projected shell model, configuration mixing, rotational symmetry, deformed nuclei, spectroscopic calculations, nuclear high-spin states

[1] E-mail: yangsun@csep1.phy.ornl.gov
[2] E-mail: Kenji.Hara@physik.tu-muenchen.de

PII S0010-4655(97)00064-7

Nature of physical problem
These Fortran codes conform with the framework of the Projected Shell Model [1]. The theory is a genuine shell model configuration mixing approach but requires only a very small configuration space. This feature enables us to interpret numerical results in simple physical terms. The present code allows detailed spectroscopic calculations for low- and high-spin states in axially deformed nuclei.

Method of solution: Starting from the Nilsson+BCS procedure, the shell model truncation is done in the multi-quasiparticle basis by selecting low-lying states. The rotational symmetry is then restored for these multi-quasiparticle states by the projection method to form a spherical (many-body) basis in the laboratory frame. Finally, the Hamiltonian is diagonalized in this basis. The large part of numerical effort is devoted to the second step, where rotational matrix elements are calculated for each integration mesh.

Typical running time
1.5 minutes on Pentium/133MHz (DOS).

References
[1] K. Hara and Y. Sun, Int. J. Mod. Phys. E 4 (1995) 637.

LONG WRITE-UP

1. Introduction

The shell model is the most fundamental way of describing many-nucleon systems fully quantum mechanically. However, it soon becomes unfeasible when going to systems with large nucleon numbers because of the dimension explosion. Even with the today's computer power, the standard shell model diagonalization can be done only in the full pf-shell space [3], for which the dimension of the configuration space well exceeds one million. One problem of this approach is that the interpretation of the numerical result is very difficult. The main reason is that the eigenstates consist of a vast amount of tiny components.

Thus, in a shell model study of heavy nuclear systems, an efficient truncation of basis is essential. On the one hand, in order to be able to extract the essence of the physics, it is desirable to use a shell model basis which has a good classification scheme (hierarchy) such that a simple configuration corresponds to a simple excitation mode of the nucleus. In fact, the truncation of the shell model space could be done quite efficiently in such a basis. On the other hand, it is known that most of nuclei in the nuclear chart are deformed, so that the use of the spherical single-particle basis is not ideal. All this suggests that it is advantageous to use a deformed basis from the outset corresponding to the optimal set of basis states in the sense of the Hartree–Fock(–Bogolyubov) theory, although it spontaneously violates the rotational (and gauge) symmetry. However, the symmetry thus broken can be restored easily and this is where the projection technique comes into play. Namely, we project the deformed intrinsic states onto a good angular momentum (and particle number) to obtain a proper shell model basis. We emphasize that the most important aspect of using a deformed basis lies not only in the fact that it makes a shell model calculation feasible but also in the fact that it allows us to interpret the numerical result in simple physical terms.

In the long history of the shell model, Elliott was the first to use the advantage of a deformed (intrinsic) many-body basis and developed the SU(3) Shell Model [4] for sd-shell nuclei. The projection of the intrinsic states in this scheme can be done easily by using the group theoretical method. It works nicely so long as the spin-orbit force is weak. However, in heavier nuclei where the presence of a strong spin-orbit force is essential for the correct shell closures (magic numbers), the SU(3) scheme is no longer valid. For such systems, we may resort to the Nilsson scheme [5] (more precisely the Nilsson+BCS scheme in order to take the strong pairing correlations into account) and project numerically the deformed basis onto a good angular momentum (and particle number). In this way, we come to the basic idea of the Projected Shell Model (PSM) approach [1]. It can be considered as a natural extension of the SU(3) shell model to heavier systems.

In this paper, we present a Fortran program which conforms with the framework of the PSM. The PSM code was first created in Munich in 1979 and has been developed nearly over the past twenty years through the

following three stages.

In the first stage, K. Hara and S. Iwasaki derived the algorithm [2] and wrote the first code. Although the original program was written for a Mainframe (a CDC Cyber), it had to heavily rely on the direct disk access method because of the memory limitation at that time (500K machine words per user, which may seem unbelievable from today's standard). Accordingly, the early calculations published by these authors [6–8] were based on restricted configuration spaces and the applicable (energy and spin) range was rather limited.

In the second stage of the development of the code, the present authors began to work in 1988. They partly borrowed subroutines from the original version, but reconstructed the program completely so that any later extension becomes possible. This extended version can indeed cope with a large number of high-spin phenomena [9–11]. This version is still in use but it relies on the direct disk access method inherited from the original version.

In 1991, Z.-y. Zhu at the Institute of Nuclear Research, Academia Sinica in Shanghai ran the program on a PC for the first time. This initiated the third stage of the program development, which actually began in 1994 when he visited Munich. Following the suggestion of K. Hara, Z.-y. Zhu worked over the extended model to remove the direct disk accesses entirely from the program, which was then debugged and largely rewritten by K. Hara in 1996. The new code is user friendly (errors will be trapped) and, above all, runs at a turbo speed (see the appendix for the speed test). This is the version we publish in the present article since it is hardware-independent and thus runs on any machine without modification.

An extensive review of the PSM has been published in Ref. [1]. Thus, we shall not spend much words about the theory except for a short outline presented in Section 2. The whole PSM program consists of four independent codes for even–even (EE), odd–neutron (OE), odd–proton (EO) and odd–odd (OO) nuclei. Since all of them have essentially the same structure, we describe the EE program in Section 3. Finally, a worked example is presented in Section 4. Some practical hints (not only) for PC users will be given in the appendix. In the following discussions, we shall closely follow the formulas[3] given in Ref. [1].

2. The Projected Shell Model

The PSM is a spherical shell model truncated in a deformed basis. The single-particle basis obtained by the Hartree–Fock–Bogolyubov procedure may theoretically serve as the optimal basis but we have chosen to use that of the Nilsson+BCS procedure since then the interpretation becomes easier while the error committed by this is practically negligible [1]. The PSM proceeds as follows: first, the truncation is done in the multi-quasiparticle (multi-qp) basis by selecting low-lying states; then the rotational symmetry (and the number conservation, if necessary) is restored for these (multi-qp) states by the projection method to form a spherical (many-body) basis in the laboratory frame; finally, the Hamiltonian is diagonalized in this basis. The truncation achieved in this way is very efficient. In fact, quite satisfactory results can be obtained by choosing only a few orbitals near the Fermi surface since the deformed quasiparticle basis already contains most of important (pairing and quadrupole) correlations.

Suppose that we have selected a set of multi-qp states $\{|\Phi_\kappa\rangle\}$ which we want to take into account in the shell model configuration space by projecting them onto good angular momentum I. Their concrete specification will be given later for different types of nuclei. Once the quasiparticle basis is prepared, we diagonalize the Hamiltonian in the shell model space spanned by $\{\hat{P}^I_{MK}|\Phi_\kappa\rangle\}$. The normalized eigenstate takes the form

$$|\Psi_{IM}\rangle = \sum_\kappa F^I_\kappa \hat{P}^I_{MK}|\Phi_\kappa\rangle\,. \tag{1}$$

[3] Unless otherwise stated, the equation numbers in the following text are those of Ref. [1].

The coefficients F^I_κ are to be determined by diagonalizing the Hamiltonian in the basis of $\{\hat{P}^I_{MK}|\Phi_\kappa\rangle\}$. This leads to the PSM eigenvalue equation

$$\sum_{\kappa'}(H^I_{\kappa\kappa'} - EN^I_{\kappa\kappa'})F^I_{\kappa'} = 0\,, \tag{2}$$

where the Hamiltonian and Norm matrix elements have been defined by

$$H^I_{\kappa\kappa'} = \langle\Phi_\kappa|\hat{H}\hat{P}^I_{KK'}|\Phi_{\kappa'}\rangle\,, \qquad N^I_{\kappa\kappa'} = \langle\Phi_\kappa|\hat{P}^I_{KK'}|\Phi_{\kappa'}\rangle\,. \tag{3}$$

Projecting an intrinsic state $|\Phi_\kappa\rangle$ onto a good angular momentum generates the rotational band associated with this intrinsic configuration, whose rotational energy is given by the expectation value of the Hamiltonian as a function of spin I,

$$E_\kappa(I) = \frac{\langle\Phi_\kappa|\hat{H}\hat{P}^I_{KK}|\Phi_\kappa\rangle}{\langle\Phi_\kappa|\hat{P}^I_{KK}|\Phi_\kappa\rangle} = \frac{H^I_{\kappa\kappa}}{N^I_{\kappa\kappa}}\,. \tag{4}$$

A diagram in which $E_\kappa(I)$ for various bands are plotted against spin I is referred to as a band diagram [1]. The solution of Eq. (2) can be compared with the experiment.

In the present code, the following Hamiltonian is employed:

$$\hat{H} = \hat{H}_0 - \tfrac{1}{2}\chi\sum_\mu \hat{Q}^\dagger_\mu\hat{Q}_\mu - G_M\hat{P}^\dagger\hat{P} - G_Q\sum_\mu \hat{P}^\dagger_\mu\hat{P}_\mu\,, \tag{5}$$

where $\hat{H}_0$ is the spherical single-particle Hamiltonian with a proper $L\cdot S$ force, the second term the $Q\cdot Q$ force and the last two terms the monopole and quadrupole pairing forces, respectively. Determination of the interaction strengths is discussed in Ref. [1], in which the use of schematic forces is also justified by showing that the essential results do not depend on details of the two-body forces so long as they fulfill certain (simple) criteria.

Below, we list the quasiparticle configurations $\{|\Phi_\kappa\rangle\}$ used in the present code for different types of nuclei:

$$\begin{aligned}
&\text{Doubly even nucleus}: && |0\rangle,\ a^\dagger_{\nu_1}a^\dagger_{\nu_2}|0\rangle,\ a^\dagger_{\pi_1}a^\dagger_{\pi_2}|0\rangle,\ a^\dagger_{\nu_1}a^\dagger_{\nu_2}a^\dagger_{\pi_1}a^\dagger_{\pi_2}|0\rangle\,;\\
&\text{Doubly odd nucleus}: && a^\dagger_\nu a^\dagger_\pi|0\rangle\,;\\
&\text{Odd neutron nucleus}: && a^\dagger_\nu|0\rangle,\ a^\dagger_\nu a^\dagger_{\pi_1}a^\dagger_{\pi_2}|0\rangle\,;\\
&\text{Odd proton nucleus}: && a^\dagger_\pi|0\rangle,\ a^\dagger_\pi a^\dagger_{\nu_1}a^\dagger_{\nu_2}|0\rangle\,,
\end{aligned} \tag{6}$$

where ν's (π's) denote the neutron (proton) Nilsson quantum numbers which run over properly selected (low-lying) orbitals. We have discarded configurations that contain three or more like-nucleon quasiparticles because they have higher excitation energies due to mutual blocking of levels and thus affect the results little in the energy (and the spin) range that interests us. This restriction can be released if necessary.

3. Program organization

Apart from the I/O parts, it is convenient to divide the code (logically) into three main parts: preparation of the configuration space, calculation of the rotated matrix elements, and diagonalization in the projected basis. Fig. 1 illustrates the structure of the whole code. Its left most part shows the main stream, which proceeds along the vertical arrows.

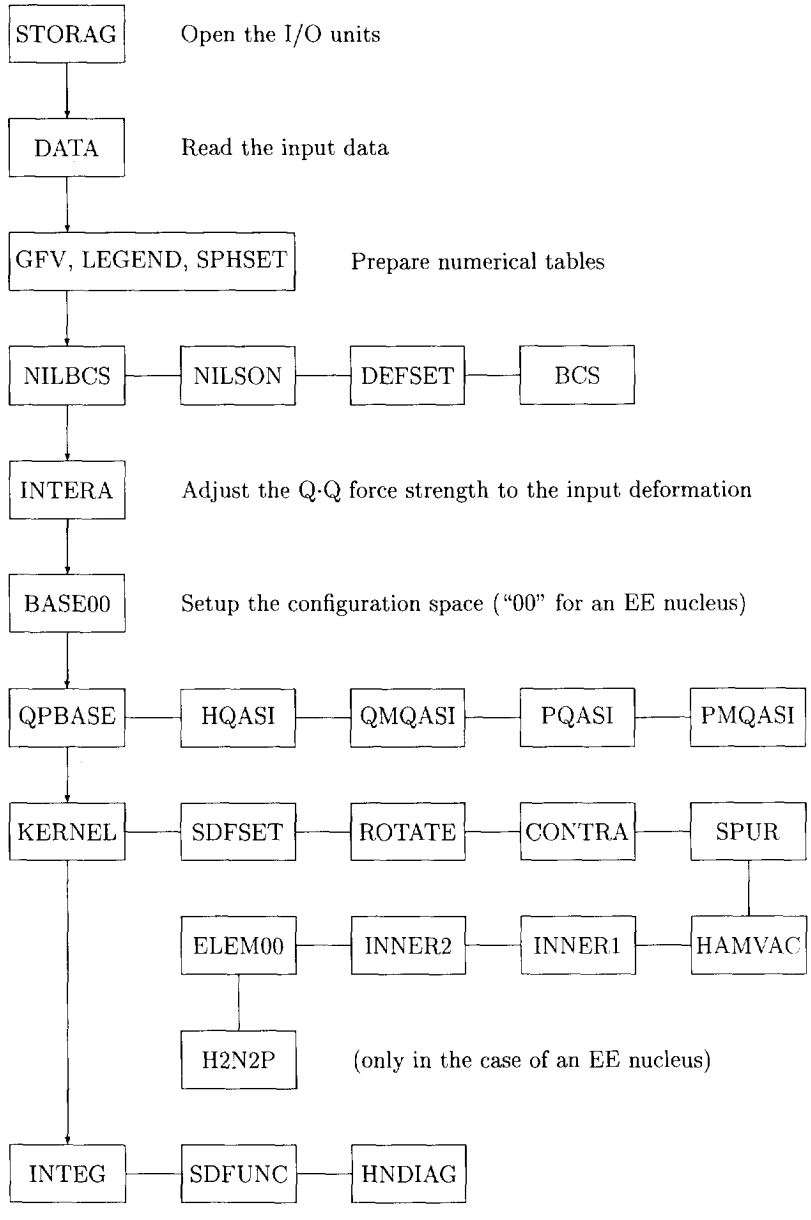

Fig. 1. A diagram showing the structure of the (EE) program; the main stream of the program proceeds along the vertical arrows (the left most part).

3.1. The input and include file

Let us begin with the description of the input and include file. The input file **XY_DATA** contains the sample data[4]. Before running the program, one must of course check carefully all input items to ensure that they correspond to the nucleus one wants to compute. The file is well commented and there should be no difficulty

[4] **XY** refers to any of the four codes, i.e. the code for EE, OE, EO or OO nuclei.

in understanding what the items represent. However, we want to comment on the choice of some parameters which one may frequently encounter in actual calculations.

The spin-orbit force parameters, κ and μ, appearing in Nilsson potential are essential in reproducing correct shell filling and are thus especially important for odd mass and odd–odd nuclei. The program offers two options selected by the constant LSF (1 or 2) read from the **XY_DATA** file. LSF = 1: the compilation of Nilsson et al. [12] which was adjusted to the rare-earth region (A-dependent) and LSF = 2: the compilation of Zhang et al. [13] which is a modified version of Bengtsson and Ragnarsson [14] and has been fitted to latest experimental data. It is supposed to apply over sufficiently wide range of shells. These κ and μ are different for different major shells (N-dependent).

The deformation parameter used in the code is the stretched deformation ε, not β, although the difference is not of much relevance. The quadrupole (ε_2) and hexadecapole (ε_4) deformation are denoted as EPS2 and EPS4 in **XY_DATA**, respectively. We usually take the deformation parameters from Refs. [15,16]. For nuclei whose deformation parameters are not listed in these two articles, one may use estimated values by extrapolating (along isotopic chains) or interpolating (along isotonic chains) the known values. To obtain the correct shell filling, one may play with these parameters within reasonable range so long as they are not too far from the standard values.

The pairing force strengths are important to obtain correct spectra. The resulting energy gap should be appropriate for the nucleus interested. The sample value for the monopole pairing force given in **XY_DATA** may not be the right one for your nucleus. On the other hand, the ratio of quadrupole and monopole pairing strengths, originally set to 0.18 in the sample data, is an adjustable quantity. One may adjust it to get the right position for the g–s band crossing [1]. According to our experience, it is in the range of 0.16–0.18 (0.18–0.20) for heavier (lighter) rare-earth nuclei.

It is sensible to keep the many-body basis (configuration space) as small as possible within the limit physics would allow. This will make not only the computation faster but also the interpretation of the result easier. Truncation of the basis can be achieved by three controllers NMAX (maximum number of the basis states), KMAX (maximum K-quantum number of the basis states) and EMAX (maximum quasiparticle energy of the basis states), which are read from the file **XY_DATA**. One needs certain experience until one gets some feeling about the proper choice of these values.

It is possible to configure the program such that one can stop (or continue) the execution in order to inspect the configuration space immediately after it is constructed. This can be done by removing few comment symbols in the MAIN program.

The subroutine **DATA** reads the above-mentioned default input file **XY_DATA**. In practice, however, you might want to use a data file named other than **XY_DATA**. For example, a file name such as **YB166** reminds you immediately the nucleus being computed, and is very convenient as you need not recompile the program for different nuclei. In this case, the corresponding three output files will be named as **YB166.O1**, **YB166.O2** and **YB166.O3**, which will be created in the same path as the input file **YB166**. Such a possibility is already implemented to the program. A big advantage of this method over the default one is that I/O files for different nuclei can "coexist" without overriding each other. To do this, it is sufficient to change just one line in the subroutine **STORAG** (follow the comment given there). You will then be prompted to enter the name of the input file from the keyboard. In this case, running the program in background is not possible but, since the computing time is anyway very short, we highly recommend this method.

Finally, we comment on the include file, **XY_PARA**, which contains a set of integers defining the size of arrays in the code. The size of arrays should be adjusted by keeping two facts in mind. On the one hand, the arrays must have sufficient spaces so that the necessary values are properly stored. On the other hand, one should keep them as small as possible in order to save the memory. We list below the parameters in **XY_PARA**.

NMD: Number of Nilsson levels in the largest major shell (28 in our example with $N = 6$).
NJMAX: j+1/2 (=N+1) of the largest j-subshell (7 in our example with $j = 13/2$ ($N = 6$)).

NNBCS: Total Nilsson levels in three major shells ($64 = 15_{(N=4)} + 21_{(N=5)} + 28_{(N=6)}$).
NNRANK: Maximum number of basis for a partial configuration.
NNRANT: Maximum number of basis for the total configuration.
NNATA: = (2 × number of the selected Nilsson levels)^2.
NJOTMX: Number of angular momenta (spins) to be computed.
NKMAX: Maximum K-value for the multi-qp states.
NNDEG: Maximum number of mesh points for the β-integration.
NNTQ3: Maximum number of 3-qp basis (used only in EO and OE codes).

If there is some conflict between these parameters and those which are expected from the input data, the nature of the conflict(s) in question will be written to **XY_OUT1** and the program stops, so that the user can immediately correct the error(s).

3.2. *Preparation of the configuration space*

After calling three subroutines **GFV**, **LEGEND** and **SPHSET**, which prepare numerical tables to be used in later calculations, we enter into the three major parts of the program. The first part constructs the quasiparticle basis and the multi-qp configuration space.

3.2.1. *Construction of the Nilsson+BCS Basis*

The MAIN program calls **NILBCS** which invokes three subroutines one after the other: **NILSON**, **DEFSET**, and **BCS**. The role of each subroutine is as follows.

The subroutine **NILSON** diagonilizes the Nilsson Hamiltonian (Eq. (A.10)) for the chosen spin-orbit force and deformation parameters. As the output, the deformed single-particle energies are stored in the array EP. The resulting wavefunctions define the transformation between the spherical and the deformed basis. This subroutine can be used also as a general code to produce a Nilsson diagram if one runs it in an ε_2-loop instead of fixed deformation.

The subroutine **DEFSET** transforms the matrix elements of relevant one-body operators originally defined in the spherical basis into the deformed basis (Eqs. (A.8), (A.9)). This includes the spherical single-particle Hamitonian, the quadrupole operator, the hexadecapole operator, the monopole and quadrupole pairing operators as well as the angular momentum and spin operators. The last two operators are defined but are not used in the present code. They will be necessary when evaluating the g-factor and BM1 transitions.

The subroutine **BCS** solves the standard BCS problem for a given set of single-particle levels (stored in array EP) and the particle number (first for neutron and then for proton). As the output, the Fermi energies are stored in the array ALAMB, the gap energies in DELTA, the quasiparticle energies in EQ and the BCS occupation factors in U and V.

Finally, we call the subroutine **INTERA**, which adjusts the strength of the $Q\cdot Q$ force to the input deformation. This is done by using Eq. (2.51) of Ref. [1], which is essentially the Hartree–Fock–Bogolyubov (HFB) minimization condition and ensures that the HFB procedure will indeed lead to the input quadrupole deformation. The self-consistency between the Hamiltonian and the quadrupole deformation is retained in this way.

3.2.2. *Construction of the configuration space*

The BCS calculation defines a set of quasiparticle states as well as the energy gap and the Fermi energy. The configuration space is now constructed by selecting the quasiparticle states close to the Fermi energy and forming multi-qp states from them. This is done in the subroutine **BASE00**[5] and is the part that should be carefully controlled through the input data (see below and the Section 3.1).

[5] The symbol 00 designates an exclusive subroutine for the EE code. Similarly, 01, 10 and 11 symbolize EO, OE and OO, respectively.

There are two aspects to be kept in mind. On the one hand, all physically significant single-particle states should be contained in the selection. Such states are around the Fermi energy, but not necessarily the closest ones (see, for example, the problem of signature inversion discussed in Ref. [10]). On the other hand, we should keep the configuration space as small as possible to avoid unnecessary numerical work. As mentioned in Section 3.1, the size of the configuration space is determined by three parameters read from the input file **XY_DATA**: NMAX, KMAX and EMAX.

Another important aspect of the subroutine **BASE00** is related to the physical interpretation of the result. To find out the role played by a particular band, one has to carefully keep the track of its configuration throughout the shell model diagonalization procedure which mixes up all bands. Such information is prepared by this subroutine and is printed out in a table form to **XY_OUT1**. This is essential when "reading" the band diagram.

3.3. Calculation of the rotated matrix elements

The heart of the angular momentum projection theory is how to compute the projected matrix elements. In fact, the idea of projection is old, but it had not been used for heavy nuclear systems until an efficient algorithm was developed by K. Hara and S. Iwasaki [2] in 1979. Computation of the projected matrix elements in the present code follows exactly this algorithm. For details, we refer to the original paper [2] or to Appendix A of Ref. [1].

3.3.1. Construction of operators in the quasi-particle basis

Operators, originally defined in the spherical basis in the subroutine **SPHSET** and later transformed to the deformed basis in the subroutine **DEFSET**, are further transformed to the quasiparticle basis (Eqs. (A.8), (A.9)). These Bogolyubov transformations are done in the subroutine **QPBASE**, which successively calls four subroutines: **PQASI** (pairing operator), **QMQASI** (quadrupole operator), **PMQASI** (quadrupole-pairing operator), and **HQASI** (spherical single-particle Hamiltonian).

3.3.2. Computation of the rotated matrix elements

The subroutine **KERNEL** is the key part of the program and takes most of the computing time. Referring to the "stream" chart Fig. 1, the first three subroutines **SDFSET**, **ROTATE** and **CONTRA** called from **KERNEL** evaluate contractions and overlap, whose detailed description can be found in Appendix A.2 of Ref. [1]. This part carries out the rotation of the deformed quasiparticle basis, which is one of the basic ingredients of our projection method. The role of each subroutine is as follows.

The first subroutine, **SDFSET**, creates a set of the standard d-function [17], which is necessary to describe the rotation of the single-particle basis. In our representation of the basis, the set of d-functions (and the following matrices to be discussed) has a 2×2 block form (Eq. (A.20)). The subroutine **ROTATE** defines two matrices $X(\Omega)$ and $Y(\Omega)$ as given by Eq. (A.23). In our Nilsson+BCS representation, they are computed by Eqs. (A.25)–(A.27). The subroutine **CONTRA** evaluates three contraction matrices $A(\Omega)$, $B(\Omega)$, $C(\Omega)$ and the overlap $\langle\hat{R}(\Omega)\rangle$ according to Eqs. (A.34) and (A.44). These quantities are the building blocks of the theory in the sense that all more complicated matrix elements can be constructed by them. In the next three subroutines, linked contractions of relevant one- and two-body operators are prepared based on the contraction matrices obtained above. Namely, for the one-body operators, the subroutine **SPUR** evaluates the trace defined in the first relation of Eq. (A.49) and **INNER1** the rest of three quantities of Eq. (A.49) while the two-body operators are treated in the subroutine **INNER2** according to Eqs. (A.58)–(A.63). Finally, the rotated Hamiltonian matrix elements are constructed in the subroutine **ELEM00** with the help of **HAMVAC** which evaluates the quantity $\langle\hat{H}[\Omega]\rangle$.

We remark that all operations in the subroutine **KERNEL** are done in a β-loop and that the rotated matrix

elements are stored as functions of the Euler angle[6] $\Omega = \beta$ for all mesh points. This is where the direct disk access method has been used in the old versions.

3.4. Diagonalization in the projected basis

In the last subroutine **INTEG** of the main stream (cf. Fig. 1), evaluation of the projected matrix elements and diagonalization of the PSM eigenvalue equation are done in the I-loop for successive spin values. Having obtained the rotated matrix elements as functions of β, the calculation of the projected matrix elements is nothing other than an integration over the variable β as given in Eq. (A.68). The d-function appearing in the integrand is computed by **SDFUNC**. The projected Hamiltonian and the Norm matrix are then stored in the arrays HH and HN, respectively. The PSM eigenvalue equation (2) of the present paper can be solved now. This is done by the subroutine **HNDIAG** which achieves two successive diagonalizations. We first diagonalize the Norm matrix and use the resulting wavefunctions to transform the Hamiltonian matrix into the representation in which the Norm matrix is diagonal (see Eq. (A.83)). We then diagonalize the Hamiltonian matrix in the new basis to obtain the final eigenvalues, i.e. the energies for a given spin I. The amplitudes F_κ^I of Eq. (1) in our original projected basis are evaluated by Eq. (A.84), which satisfy the normalization condition Eq. (A.85) automatically.

3.5. The output files

There are three output files: **XY_OUT1**, **XY_OUT2** and **XY_OUT3** corresponding to the I/O unit 1, 2 and 3, respectively. These are default files associated with the input file **XY_DATA**. The I/O file names can be changed as described before (see the Section 3.1).

The file **XY_OUT1** is the main output and contains the information about the configuration space, the band energies, the eigenvalues of the PSM equation and many more. The file **XY_OUT2** contains the yrast energy (the lowest eigenvalue of the Hamiltonian for a given spin) and the band energies (energies of the unperturbed rotational bands) defined in Eq. (4) of the present paper as functions of the spin. The file **XY_OUT3** contains the data for the backbending plot (for the case of EE) or $E(I) - E(I-1)$ plot (for the cases of OE, EO and OO). The last two output files are designed for drawing diagrams and are at the moment formatted for XMGR available on UNIX Workstations under the X-Windows environment but one may rewrite the program for other graphics package such as GLE for PC. See the end of the subroutine **INTEG** if one wants to change the output format.

4. An illustrative example

We take an even–even nucleus ^{166}Yb as an example to show how to use the code. The relevant files are **EE_NUCL.F**, **EE_DATA** and **EE_PARA** which must be in the same directory. A caution for Workstation users: the name of the last two files must be written in capital letters. The program will fail to find these files if they are stored as "ee_data" and/or "ee_para" since the upper and lower cases represent different file names.

The input data file **EE_DATA** has been prepared for this nucleus and we gave already the corresponding neutron and proton number, 96 and 70. The deformation parameters $\varepsilon_2 = 0.246$ and $\varepsilon_4 = 0.004$ for this nucleus are taken from Ref. [16]. The ratio of the quadrupole pairing force strength to that of the monopole pairing force is taken to be 0.18. Later, as an exercise, one may change this parameter around this value to study how the position of the backbending in the yrast states is affected by the quadrupole pairing interaction. Other quantities in **EE_DATA** are generally appropriate for rare-earth nuclei and may be left as they are. However,

[6] The α and γ integrations have been explicitly carried out using the axial symmetry.

prior to the first run, we recommend to read the appendix, which contains some facts including the memory requirement, the execution speed and few more.

After executing the EE code, we obtain three output files **EE_OUT1**, **EE_OUT2** and **EE_OUT3** (which are available also as sample output files). The first output file **EE_OUT1** is written in several groups, each of which starts with a subroutine name as a headline.

First, under the headline **DATA**, the input data are printed. Among others, one should confirm whether the neutron and proton numbers correspond to the nucleus one wants to compute and whether the input deformation parameters are properly chosen.

In the next headline **NILBCS**, one can see which set of the Nilsson parameters κ and μ are used in the calculation. The Fermi energy λ and the gap energy Δ resulting from the BCS calculation are also given for neutron and proton.

Under the headline **INTERA**, interaction strengths and some related quantities are printed. If you see two numbers for a certain item, the first one is for neutron and the second for proton. For the $Q \cdot Q$ interaction, the three numbers for X_Q are the *nn*, *pp* and *np* force strength, respectively. In addition, the HFB energy $\langle \hat{H} \rangle$ and the semi-classical value of the intrinsic quadrupole moment $\langle \hat{Q}_0 \rangle$ are also printed.

The following output is particularly important for a later analysis of the numerical result. In **BASE00**, after an ordering number, we print the K-quantum number (presented as $2*K$) and the Nilsson single-particle energy for three neutron shells (the next three columns) as well as three proton shells (the last three columns). We then print the corresponding quasiparticle energies in a similar way. In the present example of ^{166}Yb, one finds that the neutron Fermi energy lies close to the $2*K = 5$ quasiparticle state of the $N = 6$ high-j intruder orbitals and the proton Fermi energy just above the $2*K = 7$ quasiparticle state of the $N = 5$ high-j intruder orbitals. Therefore, orbitals around these levels are physically most important. To build multi-qp basis, 5 such neutron quasiparticle states are selected, from which 16 neutron 2-qp states are formed. Similarly, 14 proton 2-qp states are built from the selected 5 proton quasiparticle states. Furthermore, 32 4-qp states are constructed from these neutron and proton 2-qp states. As mentioned in Section 3.1, the selection of these states is done according to the quantities NMAX, KMAX and EMAX. In the present example, the dimension of our configuration space finally becomes 63 (= 16 + 14 + 32 + 1), the last state being the 0-qp state or the Nilsson+BCS vacuum. The shell model configuration space is constructed by projecting each of these multi-qp states onto a good angular momentum and the Hamiltonian is diagonalized in this space.

In **KERNEL**, we print out some rotated matrix elements at each β mesh point. We have 20 mesh points in the range of $\beta = [0, \pi/2]$. For each β-value, we tabulate the quantities $\langle H[R] \rangle$ and $\langle R \rangle$. The former is the sum of Eqs. (A.55) and (A.57) with the Hamiltonian given in Eq. (5) of this paper and the latter computed according to Eq. (A.44) of Ref. [1].

Finally, under the headline **INTEG**, we list results for rotational bands before and after the shell model diagonalization (band mixing). For each angular momentum, one can see 9 columns in the output. From the first 6 columns, one obtains information about the projected states (rotational bands) before mixing. The first column gives the ordering number (BAND) of the configuration space as defined in **BASE00** (total 63 in this example). Note that not all states in the configuration space can participate in the calculation at lower angular momenta because of the restriction $|K| \leq I$. The next 2 columns respectively give the K value and the quasiparticle energy of a multi-qp state. The forth column is the rotational energy defined by Eq. (4) of the present paper. It is followed by the Hamiltonian and the Norm diagonal element. In the seventh column, we give the eigenvalues obtained by diagonalizing the Norm matrix. This is the result of the first of the two successive diagonalizations discussed in Section 3.4. In the last column, energies obtained by the final diagonalization are listed according to the new ordering number (STATE). In the last part of **EE_OUT1**, we print out the yrast energy $E(I)$ and the stretched γ-ray transition energy $E(I) - E(I-2)$ for each (even) angular momentum I.

The output file **EE_OUT2** contains the yrast energies and the band energies (of 63 bands) as functions of angular momentum. The band diagram resulting from this output is shown in Fig. 2. Filled circles in this figure

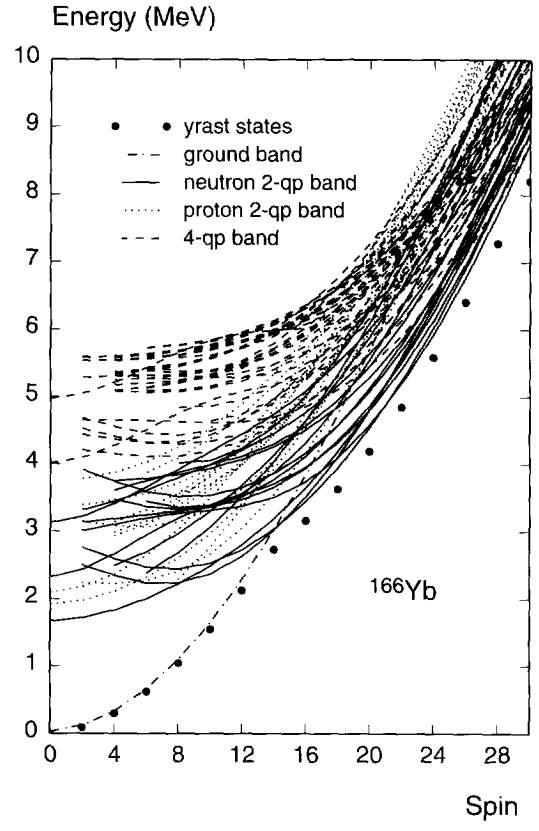

Fig. 2. A band diagram of the nucleus ^{166}Yb with yrast states (filled circles); Different multi-qp (0-, 2- and 4-qp) bands are shown in different line types. Well-aligned decoupled bands first come down before going upwards as explained in Ref. [1]. Note that two closely lying 2-qp bands cross each other just after they crossed the ground band at $I \approx 14$. This explains a tiny kink which is seen in the backbending plot (cf. Fig. 3), see the text for the configurartions of these two bands.

represent the yrast states, which are obtained from the final diagonalization procedures (band mixing). It is this kind of diagrams which one can conveniently use to extract physics out of the numerical results [1]. In the lowest spin region, the yrast state nearly coincides with the ground band (g-band), indicating that the higher multi-qp bands have little effect in the yrast state in this spin range. As the spin increases, the yrast states are pushed down more and more from the lowest band, which is a sign that the effect of the band coupling (and thus band mixing) increases. Fig. 2 also shows that the level density rapidly increases with spin.

At spin $I \approx 14$, the first band crossing is seen to take place. In the present case, there are accidentally two closely lying (2-qp) bands which cross the g-band one after the other and the higher one of which becomes the lowest band by crossing its "partner". In other words, we find nearly degenerate two "s-bands" and their mutual crossing here. This occurs whenever the Fermi energy lies around the middle point of two successive Nilsson levels. The physical consequence of such a constellation has been discussed in Ref. [1]. In fact, the effect of such two s-bands will be seen in the backbending plot (to be discussed later).

We thus want to know the configurations of these two closely lying (2-qp) bands. By looking at the output printed from the subroutine **INTEG** to the file **EE_OUT1**, we can identify that they are the neutron 2-qp band # 1 and # 2. The band # 1 lies slightly lower in energy than the band # 2 before spin $I \approx 14$ and they interchange their order shortly after this spin. This behavior will clearly show up as a tiny kink in the backbending plot. Now, to find the structures of these two bands, we must go back to the earlier part **BASE00** of the file

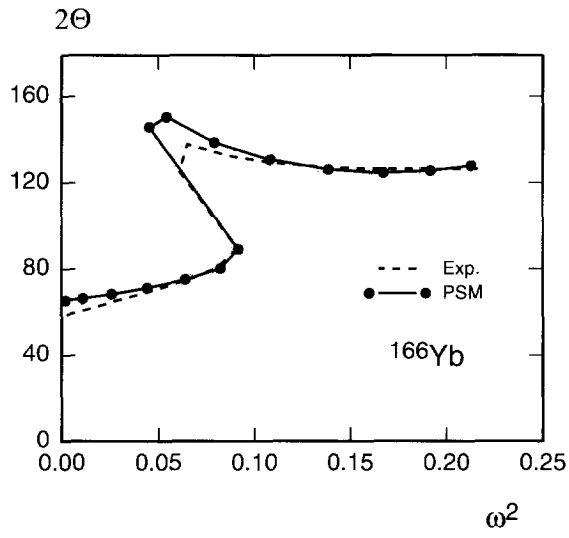

Fig. 3. The backbending plot of the nucleus ^{166}Yb; The result of the PSM (filled circles) is compared with the experimental data (open squares). A tiny kink which is seen at the largest value of the experimental 2Θ can be explained by the mutual crossing of the two closely lying 2-qp bands (cf. Fig. 2) which occurs just after their crossing with the ground band at $I \approx 14$.

EE_OUT1. There, we see that the band #1 is an $i_{13/2}$ 2-neutron state with $K_1 = 5/2$ and $K_2 = -7/2$ coupled to total $K = -1$ while the band #2 is an $i_{13/2}$ 2-neutron state with $K_1 = -3/2$ and $K_2 = 5/2$ coupled to total $K = 1$.

In the output file **EE_OUT3**, twice the moment of inertia, defined by $2\mathcal{I} = (2I - 1)/\omega$, is given as a function of ω^2, where the rotational frequency ω is calculated along the yrast level by $\omega = (E(I) - E(I-2))/2$. In the band diagram Fig. 2, the rotational frequency ω represents the slope of the rotational energy as it is defined according to the (classical) relation $\omega = dE(I)/dI$.

The backbending plot obtained from **EE_OUT3** is presented in Fig. 3. One sees that the moment of inertia smoothly increases for smaller rotational frequencies, goes suddenly backwards at about $\omega^2 = 0.1$ where it makes a sudden jump and then decreases smoothly until a plateau is reached. From the band diagram, it is easy to understand the undergoing physics. The sudden change in the moment of inertia at $\omega^2 = 0.1$ is due to the first band crossing which occurs at $I \approx 14$ (cf. Fig. 2). There, the rotational frequency ω changes from that of the g-band (larger) to that of the s-band (smaller), so that the moment of inertia suddenly increases ($\mathcal{I} \sim 1/\omega$). Therefore, a sudden decrease of ω and a sudden increase of $\mathcal{I}$ are correlated. The theory correctly predicts a small kink seen at the largest value of $2\mathcal{I}$ in Fig. 3. This is due to the mutual crossing of the above-mentioned lowest two 2-qp bands (#1 and #2, cf. Fig. 2) which occurs immediately after the first band crossing. The PSM is indeed able to describe such a fine detail of the experimental data.

The band diagram Fig. 2 tells us many more things. For example, one can see that the lowest 4-qp band is about to cross the lowest (2-qp) band at spin $I \approx 28$ with a shallow crossing angle. The rotational frequency (which is the slope of the rotational energy) changes little before and after such a crossing. Thus, in contrast to the first band crossing, it does not lead to any drastic change in the moment of inertia of ^{166}Yb. Otherwise, it would have led to the second backbending, which indeed can be seen in some nuclei. The reason why one sees a strong backbending (or the second backbending) in some nuclei and only upbending (or no second backbending) in others has been discussed in Ref. [1].

To conclude, the PSM is a powerful method of carrying out the shell model configuration mixing calculations. At present, this is probably the only way of treating heavy nuclei fully quantum mechanically. However, the most important and unique aspect of the PSM is that it allows us to interpret the result in very simple physical

terms within the framework of the theory. We believe that the PSM will continue to be useful even if one day it becomes possible to carry out the shell model calculation for an arbitrary (heavy) nucleus with the help of an ultra-supercomputer because it is difficult to build a physical picture from such a calculation as we know already from the case of the exact pf-shell diagonalization.

Acknowledgements

As documented in Section 1, the present code is the result of a long year collaboration. The authors sincerely acknowledge S. Iwasaki and Z.-y. Zhu for their valuable contributions to the development of the code in the first and the third stage, respectively.

The Joint Institute for Heavy Ion Research has as member institutions the University of Tennessee, Vanderbilt University, and the Oak Ridge National Laboratory; it is supported by the member institutions and by the U.S. Department of Energy through Contract No. DE–AS05–76ER04936 with the University of Tennessee. Theoretical nuclear physics research at the University of Tennessee is supported by the U. S. Department of Energy through Contract No. DE–FG05–93ER40770. Oak Ridge National Laboratory is managed by Lockheed Martin Energy Research Corp. for the U. S. Department of Energy under Contract No. DE–AC05–96OR22464.

Appendix A. Practical hints (not only) for PC users

The present program requires at least 16 MB RAM. If a PC has less memory than this, a virtual memory has to be used. In fact, Fortran compilers for "modern" PCs (e.g. NDP, LAHEY, NAG, MS Fortran PowerStation) have such facilities. We tested the NDP Fortran Version 3.2.0 and the MS Fortran PowerStation Version 1 on a 100 MHz 486/PC with 8MB RAM and confirmed that their virtual memory facilities work properly. In practice, the latter compiler is easier to use since it switches automatically to the virtual memory mode if the available RAM is insufficient for the program being compiled.

In addition, we tested the compilers mentioned above for their speed using the most "CPU-intensive" EE code. We quote two results. The execution time (with full optimization) was 85 sec for the MS Fortran PowerStation Version 1 (for DOS and Windows 3.11) and 105 sec for the NDP Fortran (LAHEY F77L-EM/32 was more or less the same) on a 133MHz Pentium/PC with 16MB RAM. These tests were done under the DOS (extender) environment. It should be interesting to compare with other operating systems.

Windows 95 operating system:

The latest release of the MS Fortran PowerStation is Version 4 (for Windows 95 and NT). One may learn the performance difference between 32Bit and 16Bit operating system by comparing Version 4 with Version 1. We thus tried the EE code with this compiler on a 120MHz Pentium/PC running under Windows 95. To our surprise, the execution took only 55 sec, which means that the execution time would have been 50 sec on a 133MHz Pentium/PC. In comparison to 85 sec of the Version 1, this result seems to show the power of a 32Bit operating system (no paging mechanism to access "extended memory" etc.). Windows 95 is without doubt an interesting operating system.

LINUX operating system:

LINUX is one of the most popular 32Bit operating systems and has been well established as the "UNIX for PC". We have thus tested our code also under this environment. Actually, LINUX has no generic Fortran compiler (G77 is still a beta version). Thus, one has to first apply F2C (a source level converter from Fortran to C) and then use the C compiler GCC. The whole procedure was as follows. We converted the original EE code to C via "F2C -A" (ANSI standard), compiled the generated C code with "GCC -O3" (Optimization level 3) and then ran the resulting code. The execution time was 147 sec on a 90MHz Pentium/PC running under

LINUX Version 2.0.25. This means that, if it were a 133MHz Pentium/PC, the execution time would have been 100 sec, which is slower than the Fortran PowerStation Version 1 (and is about the same as the NDP) running under DOS extender (interestingly, the LINUX system reported the memory usage to be 10MB instead of 16MB). As a matter of fact, we had expected more speed for LINUX. The reason for this disappointing result is most probably due to an inefficiency inherent to the use of an intermediate translator (F2C). It may therefore change when a "legitimate" version of G77 is released.

NDP and DOS extender:

It should be mentioned that the NDP Fortran of MicroWay is incompatible with the DOS extender HIMEM.SYS and EMM386.EXE provided by DOS. Therefore, these device drivers should not be installed to the system if one wants to use the compiler (NDP has own DOS extender provided by PharLap). However, the product like QEMM of Quaterdeck (which is designed to replace HIMEM.SYS and EMM386.EXE but is more comfortable to use) has no such problem and thus may remain installed in the system. Actually, there is a few more DOS extenders on the market which is supposed to be compatible with the NDP (PharLap) but we have not tested them.

A bug in NAG FTN90:

The NAG FTN90 Version 2.11 (for DOS and Windows 3.11 distributed by Salford Software) which is their latest product seems to have some unknown bug in the optimization procedure. In fact, depending on source codes, it fails to compile when optimized. Accordingly, this compiler could not be tested. We informed this to NAG and sent them the EE code as a sample. We do not know whether FTN77 has a similar problem or not.

References

[1] K. Hara and Y. Sun, Int. J. Mod. Phys. E 4 (1995) 637.
[2] K. Hara and S. Iwasaki, Nucl. Phys. A 332 (1979) 61.
[3] E. Caurier, A.P. Zuker, A. Poves and G. Martínez-Pinedo, Phys. Rev. C 50 (1994) 225.
[4] J.P. Elliott, Proc. Roy. Soc. 245 (1968) 128, 557.
[5] S.G. Nilsson, Dan. Mat. Fys. Med. 29 (1955) nr.16.
[6] K. Hara and S. Iwasaki, Nucl. Phys. A 348 (1980) 200.
[7] S. Iwasaki and K. Hara, Prog. Theor. Phys. 68 (1982) 1782.
[8] K. Hara and S. Iwasaki, Nucl. Phys. 430 (1984) 175.
[9] K. Hara and Y. Sun, Nucl. Phys. A 529 (1991) 445.
[10] K. Hara and Y. Sun, Nucl. Phys. A 531 (1991) 221.
[11] K. Hara and Y. Sun, Nucl. Phys. A 537 (1992) 77.
[12] S.G. Nilsson et al., Nucl. Phys. A1 31 (1969) 1.
[13] J.-y. Zhang, A.J. Larabee and L.L. Riedinger, J. Phys. G 13 (1987) L75.
[14] T. Bengtsson and I. Ragnarsson, Nucl. Phys. A 436 (1985) 14.
[15] I.L. Lamm, Nucl. Phys. A 125 (1969) 504.
[16] R. Bengtsson, S. Frauendorf and F.-R. May, Atom. Data and Nucl. Data Tables 35 (1986) 15.
[17] A.R. Edmonds, Angular Momentum in Quantum Mechanics, 3rd ed. (Princeton University Press, Princeton, 1974).