Computer Physics Communications 182 (2011) 2567–2579

Contents lists available at ScienceDirect

Computer Physics Communications

www.elsevier.com/locate/cpc

MsSpec-1.0: A multiple scattering package for electron spectroscopies in material science ☆

Didier Sébilleau [a,*], Calogero Natoli [b,c], George M. Gavaza [d], Haifeng Zhao [e], Fabiana Da Pieve [f], Keisuke Hatada [c]

[a] Institut de Physique de Rennes, UMR UR1-CNRS 6251, Campus de Beaulieu, Université de Rennes-1, 35042 Rennes-cedex, France
[b] Instituto de Ciencia de Materiales de Aragón, CSIC-Universidad de Zaragoza, 50009 Zaragoza, Spain
[c] INFN Laboratori Nazionali di Frascati, c.p. 13, I-00044 Frascati, Italy
[d] Institute of High Performance Computing, 1 Fusionopolis Way, #16-16 Connexis, Singapore 138632, Singapore
[e] Beijing Synchrotron Radiation Facility, Institute of High Energy Physics, Chinese Academy of Sciences, P.O. Box 918, Beijing 100049, PR China
[f] EMAT, University of Antwerp, Groenenborgerlaan 171, B-2020 Antwerp, Belgium

ARTICLE INFO

Article history:
Received 13 December 2010
Accepted 9 July 2011
Available online 26 July 2011

Keywords:
Spectroscopies
Multiple scattering
Cross-sections

ABSTRACT

We present a multiple scattering package to calculate the cross-section of various spectroscopies namely photoelectron diffraction (PED), Auger electron diffraction (AED), X-ray absorption (XAS), low-energy electron diffraction (LEED) and Auger photoelectron coincidence spectroscopy (APECS). This package is composed of three main codes, computing respectively the cluster, the potential and the cross-section. In the latter case, in order to cover a range of energies as wide as possible, three different algorithms are provided to perform the multiple scattering calculation: full matrix inversion, series expansion or correlation expansion of the multiple scattering matrix. Numerous other small Fortran codes or bash/csh shell scripts are also provided to perform specific tasks. The cross-section code is built by the user from a library of subroutines using a makefile.

Program summary

Program title: MsSpec-1.0
Catalogue identifier: AEJT_v1_0
Program summary URL: http://cpc.cs.qub.ac.uk/summaries/AEJT_v1_0.html
Program obtainable from: CPC Program Library, Queen's University, Belfast, N. Ireland
Licensing provisions: Standard CPC licence, http://cpc.cs.qub.ac.uk/licence/licence.html
No. of lines in distributed program, including test data, etc.: 504 438
No. of bytes in distributed program, including test data, etc.: 14 448 180
Distribution format: tar.gz
Programming language: Fortran 77
Computer: Any
Operating system: Linux, MacOs
RAM: Bytes
Classification: 7.2
External routines: Lapack (http://www.netlib.org/lapack/)
Nature of problem: Calculation of the cross-section of various spectroscopies.
Solution method: Multiple scattering.
Running time: The test runs provided only take a few seconds to run.

☆ This paper and its associated computer program are available via the Computer Physics Communications homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).

* Corresponding author.
E-mail address: didier.sebilleau@univ-rennes1.fr (D. Sébilleau).

0010-4655/$ – see front matter
doi:10.1016/j.cpc.2011.07.012

1. Introduction

Electrons are a favored tool to investigate the structure of materials, be it crystallographic, electronic or magnetic. Among their essential advantages are the facts that they are non-destructive and usually do not penetrate deep into the matter which makes them ideal for surface structure determination, or to characterize adsorption sites. In any case, when the probe electron is excited on a particular atom (which is not the case in LEED) it will essentially probe the local structure around this atom. In this case, the excitation can be made site and element specific which makes the corresponding spectroscopy even more powerful. The fact that the electrons can be made internally spin-polarized give them also the possibility to unravel magnetic structure without the need of a spin detector. For all these reasons, these types of spectroscopies have enjoyed a widespread popularity. Although in some cases, it is possible to extract directly from the experimental spectra some information about the sample, using the so-called inverse methods such as Fourier transform or holographic reconstruction, it is only through a careful comparison with simulations obtained from a suitable model that a great accuracy can be achieved. The present package proposes a computer implementation to simulate the cross-section of several spectroscopies within the framework of multiple scattering (MS) theory. Although MS can be declined in many ways, we will use here the particular formulation in terms of the so-called scattering path operator which has among its advantages the particularity to give a very straightforward picture of the physics involved, and to disconnect completely the MS part from the geometry of the incoming and outgoing beams whatever their nature. Notice that, among the electron spectroscopies treated here, we have also considered XAS. Indeed, even though in XAS no electron is detected, this spectroscopy is connected to PED by an integration over the electron emission angle [1].

2. Theoretical background

2.1. General ideas

A few codes already exist for specific spectroscopies such as PED [2–5], XAS [6,7] or AED [2]. In all cases, they are generally devoted either to one spectroscopy or to one range of energy (because of the choice of the algorithm used to solve the MS problem).

As some of us showed in a recent review article [1], MS within the scattering path operator formalism allows to describe a whole range of spectroscopies, with a minor cost for any of them once the MS part has been computed. The present program is based on the theory described in this article. We will now recall the main points and refer the reader to [1] for a more detailed presentation.

As introduced above, the scattering path operator provides a particularly convenient framework do describe spectroscopies where it is an electron that collects the information that is sought after. This operator describes all the possibilities that the electron has to go from a given atom to another given one, taking into account the whole structure of the material under study. As such it contains all the information (both structural and electronic/magnetic) in the vicinity of the electron because of the interaction of this electron with its surrounding. The size of this vicinity depends essentially on the mean free path of the probe particle, although the geometry of the experiment can also play some role in it.

To go into more details, if we represent the potential describing the interaction of the probe electron with atom i as V_i and the propagator describing the behavior of this probe within the material under study by G, we can define the scattering path operator as

$$\bar{\tau}^{ji} = \bar{V}_i \delta_{ji} + \bar{V}_j G \bar{V}_i \tag{1}$$

Here, the overbar is simply a remainder that, because atom i and atom j do not necessarily coincide with the arbitrary origin we choose, the corresponding operator contains translation operators referring to this origin.

As explicited in details in Ref. [1], all the spectroscopies using one or several electrons to probe the matter can be described using this scattering path operator, even if the electron is virtual as in the case of Resonant Elastic X-ray Scattering (REXS). For instance, the cross-section of low-energy electron diffraction (LEED) or valence photoelectron diffraction (valence PED) can be expressed as a function of $\bar{\tau}^{ji}$ with i and j scanning the material while that of core PED is directly related to $\bar{\tau}^{j0}$ where 0 is the atom where the core excitation takes place. Likewise, X-ray absorption spectroscopy (XAS) depends only on $\bar{\tau}^{00}$ and the electron energy loss spectroscopy (EELS) cross-section involves three scattering path operators at three different energies (one per electron contributing to the physical process).

Computationally speaking, Eq. (1) is not convenient and it is normally replaced by the equation of motion satisfied by this scattering path operator

$$\begin{cases} \bar{\tau}^{ji} = \bar{T}_i \delta_{ji} + \sum_{k \neq j} \bar{T}_j G_o \bar{\tau}^{ki} \\ \quad\;\; = \bar{T}_i \delta_{ji} + \sum_{k \neq i} \bar{\tau}^{jk} G_o \bar{T}_i \end{cases} \tag{2}$$

$\bar{T}_i$ is the usual atomic transition operator associated to potential $\bar{V}_i$.

A formulation such as (2) provides directly two straightforward ways to implement the computation of this scattering path operator. One is to factorize this equation of motion. Using bold characters to represent matrices, this leads to the so-called full MS

$$\boldsymbol{\tau} = \left(\boldsymbol{T}^{-1} - \boldsymbol{G}_o\right)^{-1} = \boldsymbol{T}(\boldsymbol{I} - \boldsymbol{G}_o \boldsymbol{T})^{-1} = (\boldsymbol{I} - \boldsymbol{T}\boldsymbol{G}_o)^{-1}\boldsymbol{T} \tag{3}$$

In this case, the core of the algorithm consists in the inversion of the so-called multiple scattering matrix $(\boldsymbol{T}^{-1} - \boldsymbol{G}_o)^{-1}$. This matrix will become large very quickly when increasing the energy and the number of atoms involved in the description of the material. Furthermore, it is not sparse, so that the inversion process will scale as N^2 for the memory and N^3 for the CPU time. Although being the best approach from a scattering point of view where it gives an exact result, it is restricted to low energies and relatively small clusters.

An alternative to this algorithm is to iterate the equation of motion. This gives

$$\bar{\tau}^{ji} = \bar{T}_i \delta_{ji} + \bar{T}_j G_o \bar{T}_i + \sum_{k \neq i} \bar{T}_j G_o \bar{T}_k G_o \bar{T}_i + \cdots \tag{4}$$

which, when rewritten into matrix form is formally equivalent to the series expansion of Eq. (3). We know that such an expansion is only valid if it converges, in which case its limit is indeed $(\boldsymbol{T}^{-1} - \boldsymbol{G}_o)^{-1}$. Expansion (4) is generally called the MS series expansion.

With the two algorithms (3) and (4), we are able to cover most of the energy range [0 keV, 1.5 keV]. Because it is a matrix method, the former is limited to low energies and small clusters, while the latter as it involves a perturbation expansion cannot be used at lower energies due to a lack of convergence [8]. Moreover, as the partial wave description of the scattering process on which these methods are based involves angular momenta that increase dramatically with energy, the series expansion (4) is also limited in the higher energy range, hence the upper limit. For all these reasons, it does happen that there is a gap between the use of the full MS method (3) and the series expansion method (4) where none can be used (for lack of memory space in one case and lack of convergence in the other case) [8]. This is the reason why we provide also an alternative algorithm which, despite being an expansion (but a finite one), always converges whatever the energy, being based on a combinatorial result and not on a perturbative analysis. This is the correlation expansion [9].

2.2. Cross-section of spectroscopies

Multiple scattering, especially in the scattering path operator representation, is the toolbox we will use to derive the cross-section of all the spectroscopies involving an electron (or several) to probe the material. The physical interpretation that results from Eq. (1) gives us a key to the expression of these cross-sections. Indeed, depending on the physical processes underlying them, we see that they will involve some particular matrix elements of the scattering path operator. Let us give three precise examples as an illustration. We will not derive any formula but rather refer the reader to the specialized literature for it.

The cross-sections we give in the following do not make use of any basis so as not to obscure its relationship to a particular matrix element of the scattering path operator. Recovery of the standard result is obtained by inserting whenever necessary the closure relation of the normalized spherical wave basis defined by [10]

$$\langle \vec{\mathbf{r}} | kL \rangle = k \sqrt{\frac{2}{\pi}} i^l j_l(kr) Y_L(\hat{\mathbf{r}}) \tag{5}$$

2.2.1. LEED

In a LEED experiment, a beam of electrons is focused onto the sample, and only the electrons elastically scattered into a given direction are detected. The cross-section can then be expressed as

$$\frac{d\sigma}{d\hat{\mathbf{k}}_{sc}} = 4\pi^4 \left| \sum_{i,j} e^{i\vec{\mathbf{k}}_{in}\cdot\vec{\mathbf{R}}_i} e^{-i\vec{\mathbf{k}}_{sc}\cdot\vec{\mathbf{R}}_j} \langle \vec{\mathbf{k}}_{sc} | \tau^{ji} | \vec{\mathbf{k}}_{in} \rangle \right|^2 \tag{6}$$

$\vec{\mathbf{k}}_{in}$ and $\vec{\mathbf{k}}_{sc}$ represent respectively the incoming and scattered directions.

2.2.2. Photoelectron diffraction

Photoelectron diffraction consists in monitoring a photoemission structure (usually a core level peak, but it can also be a valence band structure or a plasmon peak) as a function of the direction or the energy. The resulting modulations reflect the structure around the absorbing atom. As it is based on photoemission, it consists primarily into the ejection of an electron from a given shell of an atom we will call the absorber under the interaction with a photon, followed by the subsequent scattering by the surrounding atoms. We will label the absorbing atom as atom 0. Because the physical process giving rise to the electron is localized on atom 0 – for core levels –, it is clear that the cross-section of photoelectron diffraction should only contains terms of the form τ^{j0}, in contrast to LEED where no such localization occurs. Indeed, we can work out the cross-section as

$$\frac{d\sigma}{d\hat{\mathbf{k}}} = 8\pi^2 \alpha k \frac{m\omega_q}{\hbar} \sum_0 \sum_{m_i} \left| \sum_j e^{-i\vec{\mathbf{k}}\cdot\vec{\mathbf{R}}_j} \langle \vec{\mathbf{k}} | \tau^{j0} \widetilde{\Omega}_0^{(-)\dagger} \hat{\boldsymbol{\epsilon}} \cdot \vec{\mathbf{r}} | i \rangle \right|^2 \tag{7}$$

Here, $\widetilde{\Omega}_0^{(-)\dagger}$ is related to the Møller wave operator defined through $\widetilde{\Omega}^{(\pm)} = \Omega^{(\pm)} T^{\pm -1}$ [10]. The Møller wave operator is the operator that maps the unperturbed states $|\phi\rangle$ onto the scattering states $|\psi^{\pm}\rangle$, i.e. $|\psi^{\pm}\rangle = \Omega^{(\pm)}|\phi\rangle$.

2.2.3. X-ray absorption

In an X-ray absorption experiment, a beam of monochromatic light is directed onto the material. The fraction that is not absorbed is then measured as a function of the energy of the photons to deduce the absorbed fraction. From this definition, we see that it is strongly related to photoelectron diffraction. Indeed, if there is no other loss (*i.e.* if the sample potential is real), it is clear that the fraction of the photons that has been absorbed into the material has served to excite electrons through a photoionization process. Therefore, if we sum the photoelectron diffraction cross-section over all the directions in space, we should recover the absorption cross-section. It is in fact a consequence of the so-called optical theorem that ensures particle conservation. A numerical demonstration of this is given in the user's guide. The cross section can then be written as

$$\sigma_a = -16\pi\alpha \frac{m\omega_q}{\hbar} \sum_0 \sum_{m_{l_i}} \Im\left[\langle i | (\overline{\hat{\boldsymbol{\epsilon}} \cdot \vec{\mathbf{r}}})_0^{\dagger} \widetilde{\Omega}_0^{(-)} \tau^{00} \widetilde{\Omega}_0^{(-)\dagger} (\overline{\hat{\boldsymbol{\epsilon}} \cdot \vec{\mathbf{r}}})_0 | i \rangle \right] \tag{8}$$

where $\Im[\ ]$ stands for the imaginary part.

2.2.4. Summary

It is by no means the place to derive or even collect here the expression of the cross-section for all the spectroscopies based on an electron probe. We only give here a summary giving for some of them the matrix element of the scattering path operator they are related to. The systematic derivation of the cross-section for many such spectroscopies can be found in [1]. Note that when the spectroscopy does not involve a core level but valence states, there is no more localization. As a consequence, the valence wave function has to be expanded onto a set of atomic functions localized on every atom of the cluster used for the calculation so that multiple scattering can be applied. For photoelectron diffraction for instance, this implies that the cross-section will depend on τ^{ji} and not τ^{j0} as before, as in the LEED case. Furthermore, when several electrons are involved as in the case of EELS (electron energy loss spectroscopy) or APECS (Auger photoelectron coincidence spectroscopy), scattering path operators at different energies will appear:

$$\begin{cases} \tau^{ji}(k) & \text{LEED, valence PhD/AED} \\ \tau^{j0}(k) & \text{core PhD/AED} \\ \tau^{j0}(k_P) \quad \text{and} \quad \tau^{j0}(k_A) & \text{core APECS} \\ \tau^{00}(k) & \text{XAS, DAFS, REXS} \\ \tau^{00}(k) \quad \text{and} \quad \tau^{0i}(k_{in}) & \text{BIS} \\ \tau^{00}(k), \tau^{j0}(k_{sc}) \quad \text{and} \quad \tau^{0i}(k_{in}) & \text{core EELS} \end{cases} \tag{9}$$

3. Overview of the software structure

The MsSpec package is organized as detailed in Fig. 1.

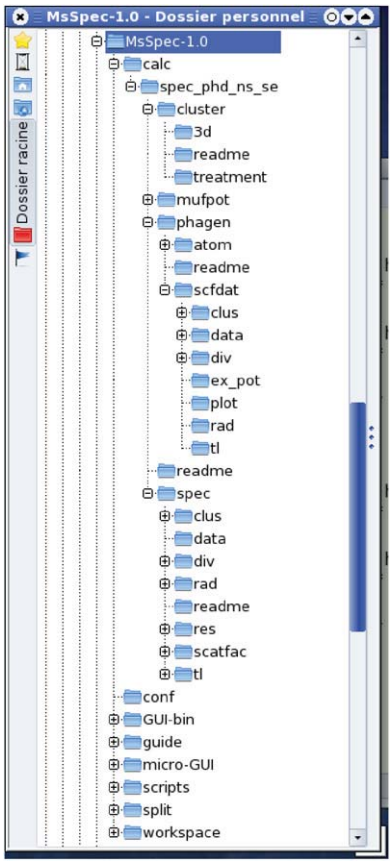

Fig. 1. Directory structure of the MsSpec-1.0 package.

It is composed of eight main directories: *calc, conf, GUI-bin, guide, micro-GUI, script, split* and *workspace*. The *GUI-bin* directory contains the graphical user interface (GUI) that is still under development. Therefore, we will not discuss it anylonger (just note that *workspace* will be the work area for this GUI). A light version of the GUI, called μ-GUI is provided instead, in the *micro-GUI* directory. It is based on the Zenity package which is available with every Linux/Unix/BSD distributions (including Mac OS). The directory *conf* is the repository of all the configuration files necessary to run this μ-GUI. As its name indicates it, *guide* is the place of the user's guide and all the related files or figures. The `makefile` that will build the `spec.f` Fortran code and some scripts used by the μ-GUI to test the software available on the computer are located into the *script* directory. The *split* directory contains all the Fortran subroutines of the MsSpec-1.0 package together will all the files necessary to run the various codes (input data files, dimensioning files, example files, Fortran scripts performing input/output operations, execution scripts, help files, readme files, ...). It is the reservoir the `makefile` will access to build a specific case environment or to update the actual one. When running the μ-GUI, *calc* is the directory where all calculations will be made. It is therefore

the user's work area. More precisely, it is the directory where the μ-GUI controlled `makefile` will build the `spec_xxx.tar.gz` file when running it with the `TAR=YES` option or the `spec_xxx` workspace directory with the (`TEST=YES` option).

The structure of the *spec_xxx* workspace area, obtained either directly or by uncompressing[1] the `spec_xxx.tar.gz` file, consists basically into four subdirectories: *cluster, mufpot, phagen* and *spec*. They correspond to the three codes that must be run to perform a calculation. The `mufpot` code by J. Pendry and the `phagen` code are close in their purpose: the former computes the phase shifts while the latter evaluates the atomic T-matrix elements. The `phagen` code should be used whenever possible as it is more accurate because it allows to use complex exchange and correlation potentials to account for inelastic losses. For this reason, `mufpot` has not been implemented into the μ-GUI.

The *cluster* directory contains the `cluster_gen.f` code and all the related files that allow to build a cluster from scratch. It itself contains the following directories: *3d, readme* and *treatment*. The *readme* directory contains some information about the building of a cluster. The *treatment* directory is composed of two interactive Fortran scripts. One, named `chg_clus.f`, allows to modify the cluster generated by `cluster_gen.f`. The possibilities offered by this script are detailed in the next section. With the second, `rot_clus.f`, it is possible to rotate the cluster in any way by giving either the rotation matrix or the Euler angles to characterize this rotation. Note that this script applies to *all* the files containing a cluster within the MsSpec-1.0 package.

The *3d* subdirectory contains Fortran scripts that will rewrite the cluster into a format suitable for molecular visualization sofware. Among the popular sofware that allows a 3D view of a cluster of atoms, we can mention

- `rasmol` (http://www.umass.edu/microbio/rasmol/index2.htm)
- `mercury` (http://www.ccdc.cam.ac.uk/products/csd_system/mercury/)
- `vmd` (http://www.ks.uiuc.edu/Research/vmd/)
- `kmovisto` (http://mitglied.lycos.de/PageOfMH/)

The first three ones, like many of others not mentioned here, rely on the so-called *pdb* format (which stands for *protein data bank*). The Fortran script `cluster_to_pdb.f` does the transformation. The KDE-based `kmovisto` software does not support the *pdb* format but uses instead a native *xyz* format. The `cluster_to_xyz.f` takes care of it. The `kmovisto` files can also be rotated with the `rot_clus.f` programme.

The *phagen* directory contains the *readme* and *scfdat* directories. This latter contains the `phagen_scf.f` Fortran code used to compute the T-matrix elements (written into the *tl* directory), the excitation radial matrix elements – dipole or Coulomb – (written into the *rad* directory) and the symmetry-reorganized cluster[2] (written into the *clus* directory) that will be used as input data files by the `spec.f` program. The input data file for running this code is in *data*. Additional output files are stored into *div*. The `msxas3.inc` file serves for the dimensioning while `procfase3` is a C-shell script to run the code.

The *spec* directory contains all what is necessary to run the `spec.f` code. The input data file, which is edited through the μ-GUI, is in the *data* subdirectory, while the T-matrix matrix elements, the radial matrix elements and symmetrized cluster are respectively into the *tl*, *rad* and *clus* directories. Into the latter, a *3d* subdirectory allows to perform the same format changes as for the results of the `cluster_gen.f` code. The structure of this cluster file being slightly different, modified Fortran scripts `clus_to_pdb.f` and `clus_to_xyz.f` are provided. In addition, the `proj_cluster.f` script allows to plot a stereographic (or orthographic, gnomonic or equal area) projection of the cluster (to compare to the corresponding projection of the cross-section). The *tl* directory contains a *treatment* subdirectory where a script called `chg_lmax.f` allows to change the l_{max} value in the T-matrix file. In the same directory, another small code named `plot_tl.f` allows to prepare the t_l file to plot either the modulus or the phase of the t_l as a function of energy, or the Argand representation of the t_l. This code splits the t_l file in as many files as there are values of l, each new file being indexed with the corresponding l value. Note that `plot_tl.f` can also plot the scattering factor at a given angle as a function of energy. The *dir* directory contains the script `ext_dir.f` which can be used to compute directions and weights to perform an integration of the cross-section over a sphere using the Lebedev–Laikin grid method, or to generate weighted sets of directions that will be summed over to account for the existence of different domains slightly disorientated with respect to the monocrystalline calculation. Results are printed into the *res* directory for cross-sections and into the *scatfac* one for scattering factors. Both directories contain a *treatment* subdirectory. The latter has the `fdplot.f` interactive script that allows to format into a two-column file the output of `spec.f` for scattering factors. The former contains three scripts. `specplot.f` has the same use as `fdplot.f` but for the cross-section case. The script `proj_interp.f` allows to perform a stereographic projection of the results together with some interpolation if the number of detection directions is not sufficient for a high accuracy plot. On the other hand, `amp_to_cs.f` allows to reconstruct a cross-section for orientated orbitals from an amplitude file.

More details will be found in the user's guide file `users_guide.pdf` which is also available through the main μ-GUI window.

4. Description of the individual software components

The MsSpec-1.0 package is composed of three main codes: `cluster_gen.f` which allows to generate a cluster, `phagen_scf.f` which computes the matrix elements of the atomic T-matrices in a partial wave representation, and if necessary matrix elements describing the excitation process, and finally `spec.f` which computes a cross-section. The latter code is build-in automatically by a μ-GUI-controlled makefile according to the user's request. The flow-chart corresponding to the use of these three codes is represented in Fig. 2.

We review now these individual codes. More details, and especially a thorough description of the input data files, can be found in the user's manual which can be obtained separately from the package from the MsSpec website http://www.ipr.univ-rennes1.fr/EN/MsSpec.

[1] The line command is: `tar zxvf spec_xxx.tar.gz`.

[2] The same Fortran script `chg_clus.f` as in the `cluster/treatment` subdirectory is present here to modify the `phagen` output cluster if necessary.

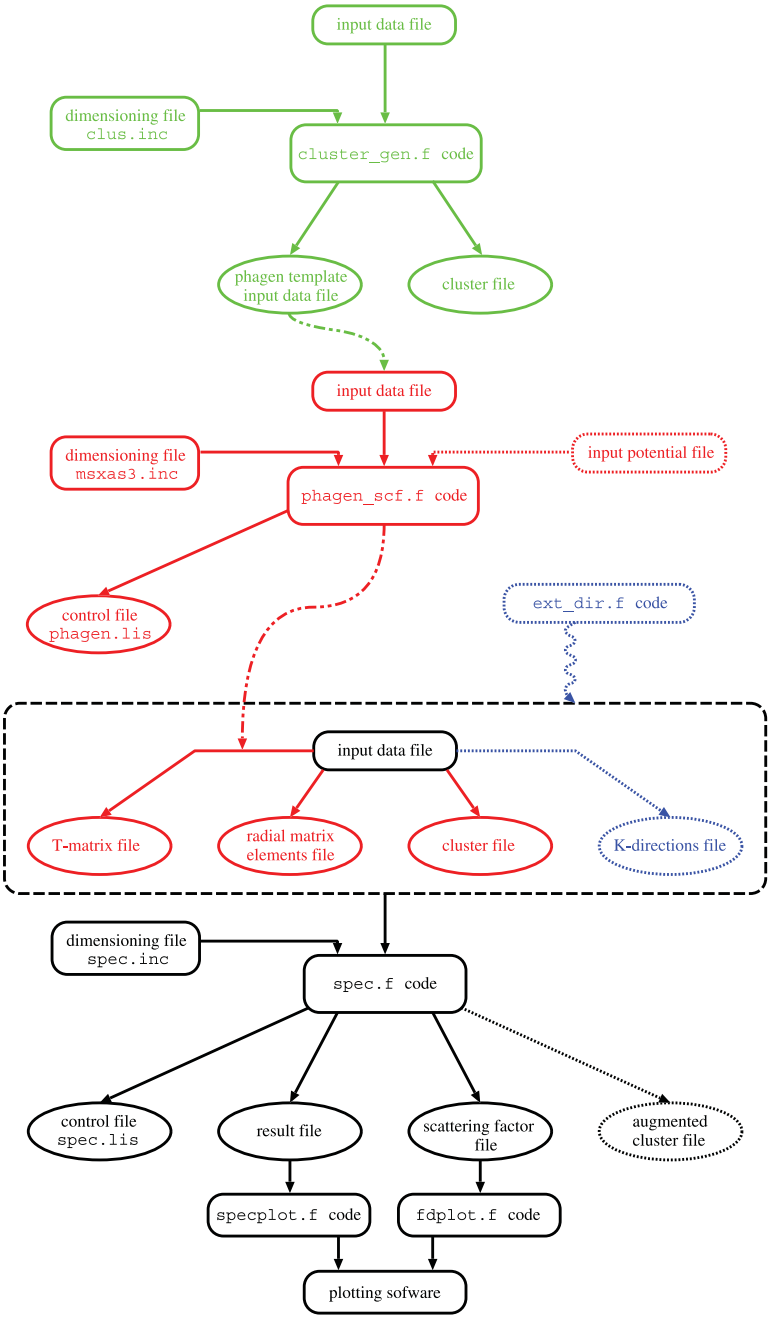

Fig. 2. A flow chart of the MsSpec package.

4.1. The `cluster_gen.f` code

`cluster_gen.f` allows to build a cluster that will then be used to perform a cross-section calculation. It is the first step in the series of codes that make the MsSpec package.

Once it has been compiled, it runs interactively, although most of the information that it requires is passed on through an input data file, `cluster.dat` that can be found in the */spec_xxx/cluster* directory. A dimensioning file named `clus.inc` is used to externally define the dimensions of the arrays. Note that a script called `proc_clus` is also provided to automate the running of this code.

If anything goes wrong, it will be printed in the control file `cluster.lis`. Therefore, it is good practice to always check this file, if only to verify that the information contained in the input data file has been passed correctly to the code.

The parameters the programme needs to construct a cluster are entered in the input data file `cluster.dat`, which is in the */spec_xxx/cluster* directory. This file contains formatted data, so it is important to respect the format of each variable so that the information is passed correctly. To this effect, tabulations are indicated in the `cluster.dat` as a guideline to the eyes. They are embedded into the separation lines '`-----`' and represented by '+'. So, a typical separation line with its tabulations looks like

```
*---+---------+---------+---------+---------+-------------------------*
```

As a general rule, the last digit of integers should coincide with a tabulation. Likewise, for real numbers, the tabulation mark corresponds to the last digit before the point.

The rule is not so strict for characters. In the unrelaxed crystalline structure, character strings start at the mark, while in the two other sections, they finish at the mark.

The input data file is composed of three section:

- unrelaxed crystalline structure
- changes to this structure (relaxation, reconstruction, ...)
- position of adsorbate atoms added by hand

Once this input data file has been filled, the user must check the dimensions of the code. The `cluster_gen.f` programme is dimensioned with the `clus.inc` file which is added at compilation time. A careful check of this file should be made prior to compilation. The `clus.inc` file is composed of:

```
      PARAMETER (NAT_M=2,NIV_M=8,NTOT=(2*NIV_M+1)**3)
      PARAMETER (NATM=NAT_M+3,NIV1=NIV_M+1,NTOT1=(2*NIV1+1)**3)
C
C
C               NAT_M    : MAXIMUM NUMBER OF ATOMS IN THE SUBSTRATE UNIT CELL
C               NIV_M    : MAXIMUM ALLOWED VALUE FOR THE MODULUS OF
C                             THE BULK LATTICE VECTORS LINEAR COMBINATION
C                             COEFFICIENTS IN THE CLUSTER
C               NTOT     : MAXIMUM NUMBER OF ATOMS IN THE CLUSTER
C               NATM     : MAXIMUM NUMBER OF ATOMS IN THE UNIT CELL
C                             (SUBSTRATE+ADSORBATES)
C
C               NIV1 ET NTOT1 ARE ONLY USED WHEN ITEST=1;
C               THEY ALLOW TO CHECK THE CONVERGENCE OF THE CLUSTER
C
C
C   **********     CAUTION : ALWAYS CHECK THAT NAT_M AND NIV_M ARE      *********
C   **********     LARGER OR EQUAL THE VALUE IN THE INPUT DATA FILE.    *********
```

Only two parameters should be given: the maximum number of atoms in the unit cell, and the maximum bound for the `NIV` parameter controlling the linear combinations of the lattice vectors used to generate the cluster. The `cluster_gen.f` code should be recompiled whenever the `clus.inc` is modified.

Running is interactive. Alternatively, the script called `proc_clus` can be used to bypass the interactive mode. Note that in addition to the cluster, whose name is chosen by the user, the code can also generate a `phagen` template input data file containing the same cluster. This file can then be edited and used as the input data file for `phagen_scf.f`. For this, select at the prompt the `SCFDAT` version, the other version being obsolete and only kept for testing and checking purposes.

Two small Fortran codes respectively named `chg_clus.f` and `rot_clus.f` are provided in the */spec_xxx/cluster/treatment* directory as a convenience to perform some changes on the cluster generated by the `cluster_gen.f` programme. The first of these interactive scripts can operate either on the cluster file or on the `phagen` template input data file indifferently. It performs the following tasks:

- translation of the coordinates
- rotation of the coordinates (about the z axis)
- scaling of the coordinates
- suppression of the last $x = constant$ planes
- suppression of the last $y = constant$ planes
- suppression of the last $z = constant$ planes
- suppression of distant atoms:
  - outside a sphere centered on the first atom
  - outside a cylinder centered on the z axis going through the first atom
  - outside a cone centered on the z axis going through the first atom

The second interactive script, `rot_clus.f`, can operate on the cluster file generated by the `cluster_gen.f` programme, on the `phagen` input data file, on a file formatted for the `kmovisto` (3D visualization software), or on the cluster generated by `phagen` and used as an input file by the `spec.f` code. It allows to rotate the cluster about *any* axis by giving either the rotation matrix or the Euler angles of the rotation considered.

There are many tools available that allow to visualize in 3D atoms, molecules, clusters, ... Most of these software accept as an input a list of atoms in the so-called `pdb` format (`Protein Data Bank`). This is the format usually used in protein studies. `kmovisto` is an exception as the input cluster should be given in a native `xyz` format.

Two small Fortran codes are provided in the *workspace/cluster/3d* directory to rewrite the output cluster file generated by `cluster_gen.f` in a format suitable for these software. They are called respectively `cluster_to_pdb.f` and `cluster_to_xyz.f`. They are both run interactively.

4.2. The phagen_scf.f code

The phagen_scf.f programme computes the T-matrix elements, the radial matrix elements (describing the radial part of the excitation process both for a photoelectron and an Auger electron from a core level) and provides also to the spec.f code a cluster reorganized in terms of symmetry. For this, it constructs a (muffin-tin) potential corresponding to the input cluster originating from cluster_gen.f and solves the Schrödinger equation inside each muffin-tin sphere. Note that the potential generated by phagen_scf.f is very basic as it consists simply in a superposition of atomic potentials to which an exchange and correlation potential, chosen by the user, is added. In particular, in this version there is no attempt to perform a self-consistent loop, as in bandstructure programmes, to generate a self-consistent charge. This type of potential however is largely sufficient to describe the scattering processes when the electrons involved have a reasonably high kinetic energy. In this case, the electron penetrates sufficiently into the atom to be scattered by the core electrons and the nucleus. Therefore, the influence of the surrounding atoms on this atomic potential can be safely neglected. On the contrary, when the energy of the electron is small (close to the absorption edge) this is not true anymore and the potential generated by phagen_scf.f might not be sufficiently accurate. Furthermore, in this case we know that the muffin-tin approximation (spherically symmetric potential within the atomic spheres and constant potential outside) breaks down. Therefore, there is no guaranty of accuracy for kinetic energies below $\sim$ 25 eV. In this case, if the accuracy is not sufficient, it is advisable to use a bandstructure code to obtain both the T-matrix elements and the radial matrix elements. The actual version of MsSpec accepts outputs from the TB-LMTO-ASA code [11].

The only input file to phagen_scf.f is the input data file. This file is called data3.ms and is situated in the */spec_xxx/phagen/scfdat/data* directory.

The data3.ms is a template file that contains a test cluster. To avoid to have to change all the cluster in it, cluster_gen.f generates also a template input data file for phagen_scf.f, but containing the right cluster. It is therefore advisable to transfer this file to the *phagen* directory */spec_xxx/phagen/scfdat/data* and start from it.

The data3.ms file is composed of three sections:

- the general parameters section
- the cluster section
- the ionization section

The rest of the file is composed of text explaining the different parameters.

Two files are included into phagen_scf.f at compilation time. They are called respectively msxas3.inc and msxas3c.inc. The latter only contains declarations and should never be edited. The former is the dimensioning file. Although most of the time the default value will be sufficient, it might be necessary to modify it from time to time.

A script file called procfase3 is provided to run the code. It is in the */spec_xxx/phagen/scfdat* directory, like the Fortran code and the dimensioning file. This script should be edited prior to running a calculation as it contains all the names of the input/output files. It is a C-shell script, therefore the user must make sure that C-shell csh, tcsh, ... is available on his system. This can be done very simply by checking whether the file */bin/csh* or */bin/tcsh* exists.

4.3. The spec.f code

The spec.f code does not exist as such. It is build up from a collection of subroutines, either by the μ-GUI or by running the makefile provided in the *script* directory of the MsSpec package. Therefore, the resulting spec.f programme is specific to one spectroscopy and one type of algorithm to compute the scattering path operator. More precisely, the following possibilities are available in the present version of MsSpec (inside the parenthesis are the options to be passed to the makefile):

- spectroscopies: photoelectron diffraction (SPEC=PHD), Auger electron diffraction (SPEC=AED), X-ray absorption (SPEC=XAS), low-energy electron diffraction (SPEC=LED), Auger photoelectron coincidence spectroscopy (SPEC=ACS)
- eigenvalue calculation: the eigenvalues of the multiple scattering matrix can be calculated in order to obtain the spectral radius that governs the convergence of the series expansion (SPEC=EIG)
- spin resolution: non spin-polarized calculation (SPIN=NO), spin-polarized one (SPIN=YES)[3]
- photoelectron final state: non spin–orbit resolved (SO=NO), spin–orbit resolved (SO=YES)
- Auger electron final state: non-multiplet resolved (MU=NO), multiplet resolved (MU=YES)
- algorithm: matrix inversion (INV=YES), correlation expansion (COR=YES), series expansion (SER=YES). The latter being the default, it can be omitted when chosen. Note that in the Auger photoelectron coincidence spectroscopy mode, a distinction is made between INV (photoelectron) and INVA (Auger electron)
- symmetrization: non-symmetrized calculation (SYM=NO), symmetrized calculation (SYM=YES)[4]

In console mode, the makefile is run by typing make SPEC=XXX plus the other options chosen, each option separated by a blank. For instance, the photoelectron diffraction spec.f programme using the matrix inversion will be obtained by typing 'make SPEC=PHD INV=YES'. Apart from SPEC=XXX which is necessary, all the other options have a default value. By default, the code produced will use a series expansion, be non spin-polarized, non-symmetrized and have nothing resolved in the final state so that the whole structure is taken into account. In μ-GUI mode, you just select from the prompted windows.

All the files necessary for a full run of the MsSpec package are present in the *calc* directory, which is the directory where the user should work in console mode or in μ-GUI mode. Running the makefile as specified above will not produce any Fortran code,

[3] Available only for photoelectron diffraction in the present release.
[4] *Idem.*

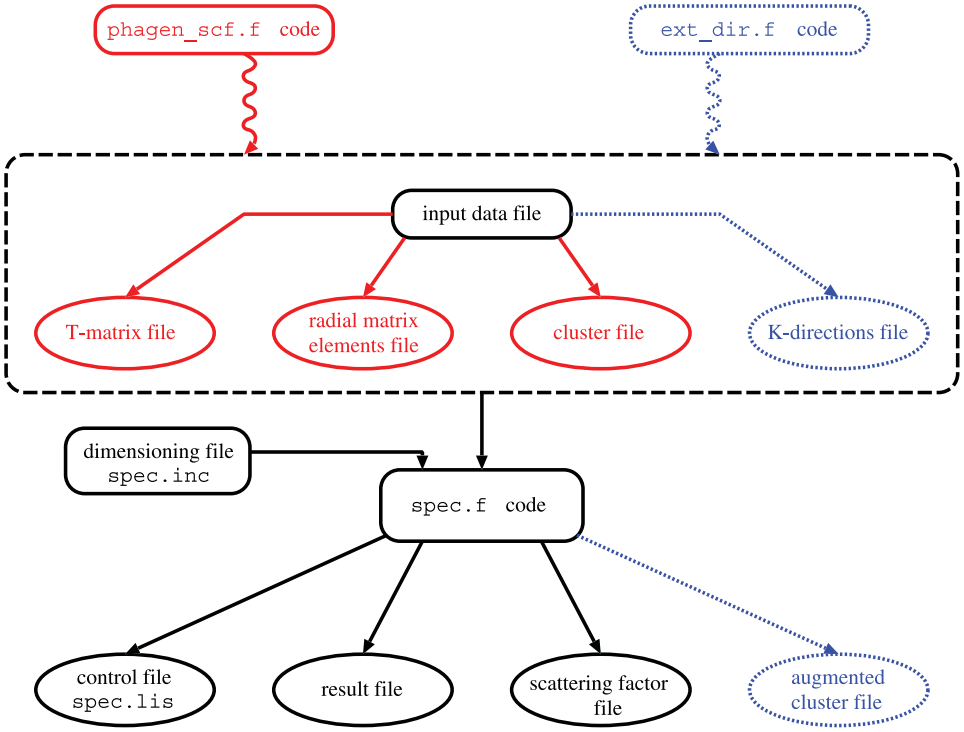

Fig. 3. Flow chart of the `spec.f` code with all the input/output files.

but only an executable file `spec` that is located in the */spec_xxx/spec* directory. Except otherwise stated, this executable file is obtained using the GNU `g77` compiler. Intel Fortran compiler or Portland Group Fortran compiler can be selected by passing to the `makefile` the option `COMP=IFC` and `COMP=PGF` respectively. Likewise, the debugging option can be activated by setting `DEBUG=YES`. If the user wishes to have a Fortran code to look at, then the makefile option `SRC=YES` should be used. Note also that there is the possibility to produce a whole workspace, specific to a given choice of the different calculation options, in the archived form `spec_xxx_xx_xx.tar.gz` to implement it elsewhere on the user's system. Here `xxx_xx_xx` is a chain of characters specific to each case. For instance, `spec_xas_ns_se.tar.gz` will correspond to a XAS case (`xas`) with no spin polarization (`ns`) or final state resolution and based on a series expansion approach (`se`). This is done with the option `TAR=YES`.

For matrix inversion, *i.e.* when passing to the `makefile` the option `INV=YES`, two possible choices are provided to perform this matrix inversion. One corresponds to the well-known *LAPACK* library (http://www.netlib.org/lapack/index.html). It is the default value but can be accessed otherwise through the option `LIB=LAP`. The other choice is to use the lesser known *SLATEC* library (http://www.netlib.org/slatec/index.html). It is selected with the option `LIB=SLA`. These two libraries are provided because the tests made with them on computers running different processors led to opposite results as to the CPU time: the *SLATEC* version was about 15% faster than the *LAPACK*[5] one on an Athlon64 (Opteron) processor while it was roughly 60% slower on an Intel Pentium M processor.

In the case of the *LAPACK* library, most Linux distributions provide a compiled version which is normally much faster that the one we compile by ourselves. This means that whenever possible, it is better to remove the *LAPACK* subroutines from the `spec.f` and link to these compiled libraries at compilation time using the options `-lblas -llapack`. The *LAPACK*-free version of `spec.f` can be obtained by using the option `LINK=YES` combined with `TAR=YES`. This will have the effect to produce a compressed whole workspace in which the `spec.f` does not contain the *LAPACK* subroutines.

The structure of the input/output file system for `spec.f` is represented as a flow chart in Fig. 3. The name of all these files should be entered in the corresponding section of the input data file `spec.dat`. Further information is contained into the MsSpec-1.0 user's guide, but note that the input data file contains also all this information, as comments.

Input files resulting from the running of the `phagen_scf.f` code are represented in red. They are three: the T-matrix file, the file containing the radial matrix elements and the cluster file. Note that `spec.f` takes care automatically of empty spheres if these have been generated by the `phagen_scf.f` code through an TB-LMTO-ASA potential calculation [11]. Their content can be modified (T-matrix file) or partially taken into account (T-matrix file and radial matrix elements file) if necessary. The cluster file however should not be touched. If a modified cluster is needed, then the `phagen_scf.f` should be run again. All the input files in blue are not necessary to run the `spec.f` code.

Whenever necessary, the $\vec{\mathbf{k}}$-directions of collection of the excited electron (if any) can be read in from an external file (in blue and dotted lines in the flow chart) rather than generated externally. This external file can be user-defined or generated by the `ext_dir.f` Fortran script provided in the *spec_xxx/spec/dir* subdirectory.

Note that for spectroscopies for which two electrons are involved, the T-matrix file, the radial matrix elements file and if necessary the $\vec{\mathbf{k}}$-directions file should be doubled, as one set of files is necessary for each electron. Obviously, the cluster file is common to the two electrons.

[5] The tests were performed using the *LAPACK* 3.0 version. The version provided here is now the 3.1 which is slightly faster.

There are three standard output files: the control file `spec.lis`, the cross-section result file (which is in the */spec_xxx/spec/res* subdirectory) and the scattering factor file (located in the */spec_xxx/spec/scatfac* subdirectory). As this version of MsSpec does either a cross-section calculation or a scattering factor calculation but never both, only one of these two result files is constructed. In the symmetrized version of the series expansion code, an augmented cluster can be constructed by the code with more symmetries than the original cluster. The name of the new cluster file should be given in the relevant section of the input data file. It is important to check the control file `spec.lis` as any problem that occurred during the running will normally appear into this file.

The dimensioning file is called `spec.inc`. Note that it contains maximal values of the various parameters. Each time it is modified, the `spec.f` code must be recompiled. Then, the code is run using the `procspec` script which must be customized by the user. It looks like

```
#! /bin/bash -f
clear
echo "                                                  "
echo "                                                  "
echo "**************************************************"
echo "*                                                *"
echo "*             PHD NON SPIN-POLARIZED             *"
echo "*         MULTIPLE SCATTERING CALCULATION        *"
echo "*                                                *"
echo "**************************************************"
echo "                                                  "
echo "                                                  "
#
#  If you want the CPU time of your calculation to be
#    printed into a separate file, uncomment and
#    customize the commented lines to your needs
#
#
#echo "  " >> cpu.res 2>&1
#echo "============================" >> cpu.res 2>&1
#echo "  " >> cpu.res 2>&1
#date >> cpu.res 2>&1
#echo "              +++             " >> cpu.res 2>&1
#echo " Input data file : spec.dat :" >> cpu.res 2>&1
#echo "              +++             " >> cpu.res 2>&1
#echo " CPU time :" >> cpu.res 2>&1
#echo "  " >> cpu.res 2>&1
#
#(time  -p  ./spec  <<Fin)  >> cpu.res 2>&1
#
time -p ./spec <<Fin
 3                          #  Number of input files : Format(I2) NFICHLEC
 5                          #  Unit number     : Format(I2)
data/spec1.dat              #  Input data file : Format(A24) (one line per file)
data/spec2.dat              #  Input data file : Format(A24) (one line per file)
data/spec3.dat              #  Input data file : Format(A24) (one line per file)
Fin
#
exit
```

`spec` is the name of the executable file as it was defined when compiling. Then follows:

- the number of input files (several calculations can be run successively. See the user's guide for more details)
- a unit number that should be left to 5 for security
- the name(s) of the input data file(s) with the whole path starting from `spec` directory.

5. Installation instructions

A file containing installation notes is provided with the package and can be downloaded separately together with the user's guide from the MsSpec website. We recall here the main points contained in this file.

The MsSpec-1.0 package comes with an installation script based on Zenity which is normally available in any Linux/Unix/BSD distribution. Please note that the MsSpec-1.0 package can only be installed on Linux and Unix-like systems. For Windows users, it is recommended to install VirtualBox or VMware and then run a Linux distribution inside a virtual machine. It works well, the loss in CPU time is small, but you must take care as how much memory you devote to your virtual machine.

To install the package, once it has been uncompressed, you just need to `cd` to the MsSpec-1.0 directory and type:

```
./MsSpec-1.0_wizard
```

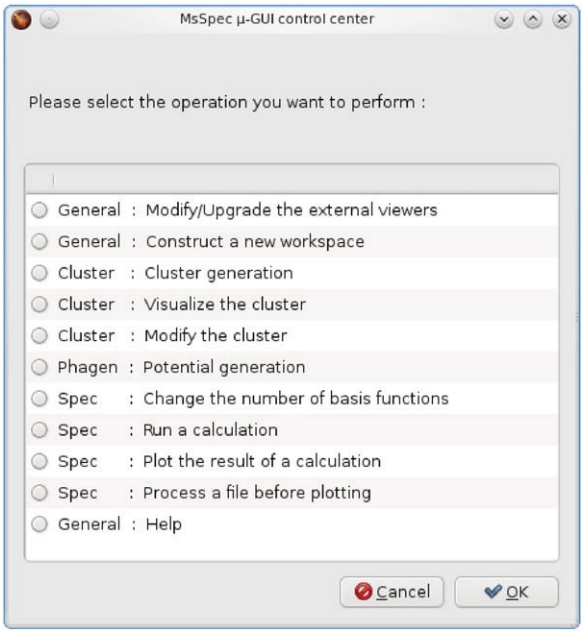

Fig. 4. The MsSpec-1.0 μ-GUI main window.

Please make sure that the execution rights have not disappeared when copying the directory you have downloaded. If it were the case, you just need in the MsSpec-1.0 directory to type:

```
chmod a+x -R GUI micro-GUI scripts
```

All graphics, editing, … is done using external software that should be installed before. The μ-GUI supported external software (this means that there is a script dedicated to each one and that the configuration script that looks for the software available on the computer specifically searches for it) at present is

- Terminals: konsole, Gnome-terminal, Terminal, xterm, urxvt
- Molecular viewers: rasmol, garlic, pymol, gdis, xmakemol, mercury
- 2D plotters: grace, gnuplot, qtiplot, kst (with KDE 3 systems only)
- 3D plotters: gnuplot (4.2 preferably)

Once all the installation has been done, the configuration should be run from the μ-GUI. This μ-GUI is shown in Fig. 4. Basically the order of the operations proposed in the μ-GUI corresponds to the order to follow for configuring and running a calculation.

The configuration of the system is done by running: “General: Modify/Upgrade the external viewers”. Once this is done, the user must construct a workspace. This is done by running “General: Construct a new workspace”. Once this is done, all subsequent calculations will be done within this workspace.

6. Test run description

The MsSpec-1.0 comes with default files for all the three codes that compose it. We describe here briefly the default results for the `spec_phd_ns_se` code. This code corresponds to non spin polarized (`ns`) photoelectron diffraction (`phd`) using the series expansion (`se`) algorithm. The default files for the `spec.f` code are `data/spec.dat` (input data file), `clus/test.clu` (cluster file), `tl/tl_test.dat` (T-matrix file) and `rad/rad_test.dat` (radial matrix elements). The three last file are generated by `phagen_scf.f` and have to be moved from the *phagen* directories to the *spec* directories when doing a calculation. The default `data/spec.dat` input data file does a double scattering calculation. The beginning of the default result file `res/res_test.dat` should look like

```
PHD            CROSS-SECTION
0  0   0   0   0   0   1   0
 1     81      1     3   0
   1  -10.00     0.00    1208.00  0.000000E+00  0.000000E+00
   2  -10.00     0.00    1208.00  0.000000E+00  0.000000E+00
   3  -10.00     0.00    1208.00  0.114322E+01  0.726914E+00
   0  -10.00     0.00    1208.00  0.114322E+01  0.726914E+00
  -1  -10.00     0.00    1208.00  0.114322E+01  0.726914E+00
   1   -9.00     0.00    1208.00  0.000000E+00  0.000000E+00
   2   -9.00     0.00    1208.00  0.000000E+00  0.000000E+00
   3   -9.00     0.00    1208.00  0.114398E+01  0.814278E+00
```

```
  0   -9.00    0.00   1208.00  0.114398E+01  0.814278E+00
 -1   -9.00    0.00   1208.00  0.114398E+01  0.814278E+00
  1   -8.00    0.00   1208.00  0.000000E+00  0.000000E+00
  2   -8.00    0.00   1208.00  0.000000E+00  0.000000E+00
  3   -8.00    0.00   1208.00  0.114465E+01  0.874288E+00
  0   -8.00    0.00   1208.00  0.114465E+01  0.874288E+00
 -1   -8.00    0.00   1208.00  0.114465E+01  0.874288E+00
  1   -7.00    0.00   1208.00  0.000000E+00  0.000000E+00
  2   -7.00    0.00   1208.00  0.000000E+00  0.000000E+00
  3   -7.00    0.00   1208.00  0.114525E+01  0.903004E+00
  0   -7.00    0.00   1208.00  0.114525E+01  0.903004E+00
 -1   -7.00    0.00   1208.00  0.114525E+01  0.903004E+00
  1   -6.00    0.00   1208.00  0.000000E+00  0.000000E+00
  2   -6.00    0.00   1208.00  0.000000E+00  0.000000E+00
  3   -6.00    0.00   1208.00  0.114576E+01  0.907184E+00
  0   -6.00    0.00   1208.00  0.114576E+01  0.907184E+00
 -1   -6.00    0.00   1208.00  0.114576E+01  0.907184E+00
  1   -5.00    0.00   1208.00  0.000000E+00  0.000000E+00
  2   -5.00    0.00   1208.00  0.000000E+00  0.000000E+00
  3   -5.00    0.00   1208.00  0.114620E+01  0.897653E+00
  0   -5.00    0.00   1208.00  0.114620E+01  0.897653E+00
 -1   -5.00    0.00   1208.00  0.114620E+01  0.897653E+00
  1   -4.00    0.00   1208.00  0.000000E+00  0.000000E+00
  2   -4.00    0.00   1208.00  0.000000E+00  0.000000E+00
  3   -4.00    0.00   1208.00  0.114655E+01  0.882110E+00
  0   -4.00    0.00   1208.00  0.114655E+01  0.882110E+00
 -1   -4.00    0.00   1208.00  0.114655E+01  0.882110E+00
  1   -3.00    0.00   1208.00  0.000000E+00  0.000000E+00
  2   -3.00    0.00   1208.00  0.000000E+00  0.000000E+00
  3   -3.00    0.00   1208.00  0.114683E+01  0.863893E+00
  0   -3.00    0.00   1208.00  0.114683E+01  0.863893E+00
 -1   -3.00    0.00   1208.00  0.114683E+01  0.863893E+00
  1   -2.00    0.00   1208.00  0.000000E+00  0.000000E+00
  2   -2.00    0.00   1208.00  0.000000E+00  0.000000E+00
  3   -2.00    0.00   1208.00  0.114702E+01  0.846650E+00
  0   -2.00    0.00   1208.00  0.114702E+01  0.846650E+00
 -1   -2.00    0.00   1208.00  0.114702E+01  0.846650E+00
  1   -1.00    0.00   1208.00  0.000000E+00  0.000000E+00
  2   -1.00    0.00   1208.00  0.000000E+00  0.000000E+00
  3   -1.00    0.00   1208.00  0.114714E+01  0.838066E+00
  0   -1.00    0.00   1208.00  0.114714E+01  0.838066E+00
 -1   -1.00    0.00   1208.00  0.114714E+01  0.838066E+00
  1    0.00    0.00   1208.00  0.000000E+00  0.000000E+00
  2    0.00    0.00   1208.00  0.000000E+00  0.000000E+00
  3    0.00    0.00   1208.00  0.114718E+01  0.847309E+00
  0    0.00    0.00   1208.00  0.114718E+01  0.847309E+00
 -1    0.00    0.00   1208.00  0.114718E+01  0.847309E+00
```

The first column contains the plane number, with -1 being the sum of all the planes. The next columns are respectively the polar angle, the azimuthal angle, the kinetic energy. The last two columns give the atomic cross-section (no scattering) and the total cross-section.

Once treated with `specplot.f`, we can plot it. Fig. 5 shows the result. Note that the direct signal has been substracted in this figure.

Acknowledgements

An important part of this work was made within the framework of the LighTnet European network (2006–2010). Some other essential parts were made possible through the Chinese Academy of Sciences/Centre National de la Recherche Scientifique projects (2004–2006 and 2009–2010) with the groups of Wu Ziyu in BSRF (Beijing, China) and NSRL (Hefei, China). Part of this work was also supported by the National Natural Science Foundation of China (project No. 10905067).

K.H. acknowledges a post-doc funding from the Région Bretagne.

D.S. thanks J. Osterwalder (University of Zürich) for using his piece of code to generate the θ-dependent number of ϕ points for the stereographic representation, H. Bulou (University of Strasbourg) for writing the program allowing to use the g2 graphical library, and P. Krüger (University of Bourgogne, Dijon, France) for his help in building an interface between the TB-LMTO-ASA code [11] and MsSpec.

C.N. thanks J.J. Rehr and A. Ankudinov (University of Washington) for permission to use some of their subroutines which are included in `phagen_scf.f`.

M.G. acknowledges financial support from the French embassy in Romania, and H.Z. from the French embassy in China.

D.S. wants also to thank more particularly D. Agliz (Ibnou-Zohr University, Agadir, Morocco), Wu Ziyu (NSRL, University of Science and Technology of China, Hefei, P.R. China and BSRF, Institute of High Energy Physics, Beijing, P.R. China), Wang Junyue and Li Enrong (BSRF,

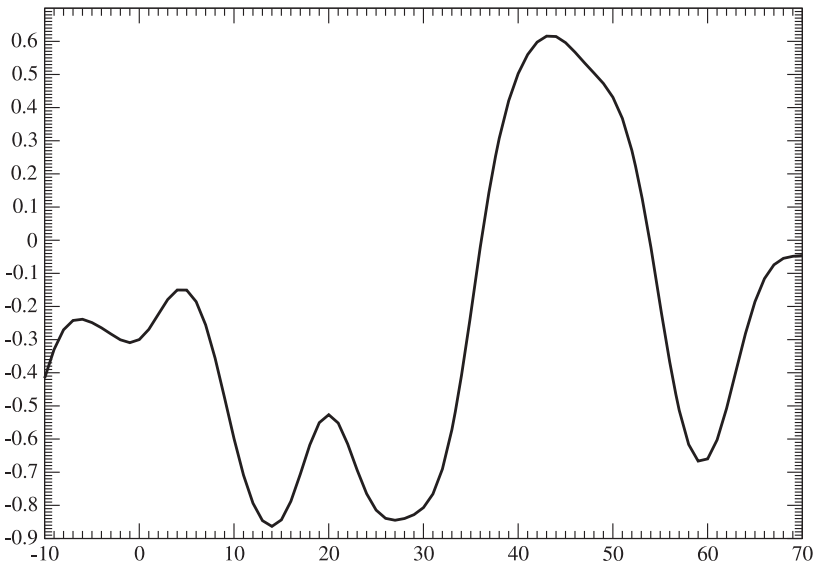

Fig. 5. The result of the test calculation.

Beijing, P.R. China), L. Frein (University of Rennes-1, France) for their contribution to various parts of the MsSpec package and A. Carré (University of Rennes-1, France) for constructing the MsSpec website (http://www.ipr.univ-rennes1.fr/EN/MsSpec).

Discussions with B. Lépine, P. Schieffer (University of Rennes-1, France), Y. Lu (University of Nancy), R. Gunnella (University of Camerino, Italy), F. Scheurer (University of Strasbourg) and R. Belkhou (SOLEIL synchrotron, St Aubin, France) who suggested improvements are also acknowledged.

References

[1] D. Sébilleau, R. Gunnella, Z.-Y. Wu, S. Di Matteo, C.R. Natoli, J. Phys.: Condens. Matter 18 (2006) 175–230, Topical review.
[2] X. Chen, D.K. Saldin, Comput. Phys. Comm. 112 (1998) 67.
[3] R. Gunnella, F. Solal, D. Sébilleau, C.R. Natoli, Comput. Phys. Comm. 132 (2000) 251.
[4] Y. Chen, M.A. Van Hove, http://www.ap.cityu.edu.hk/personal-website/Van-Hove_files/mscd/mscdpack.html.
[5] F. Javier García de Abajo, http://maxwell.optica.csic.es/software/edac/index.html.
[6] http://leonardo.phys.washington.edu/feff/.
[7] http://gnxas.unicam.it/XASLABwww/pag_gnxas.html.
[8] D. Sébilleau, C.R. Natoli, J. Phys.: Conf. Ser. 190 (2009) 012202.
[9] H.-F. Zhao, D. Sébilleau, Z.-Y. Wu, J. Phys.: Condens. Matter 20 (2008) 275241.
[10] D. Sébilleau, Phys. Rev. B 61 (2000) 14167.
[11] http://www.fkf.mpg.de/andersen/LMTODOC/LMTODOC.html.