```
PPPPP   EEEEEE  N    N  EEEEEE  L        OOOO   PPPPP   EEEEEE
P    P  E       NN   N  E       L       O    O  P    P  E
P    P  E       N N  N  E       L       O    O  P    P  E
PPPPP   EEEE    N  N N  EEEE    L       O    O  PPPPP   EEEE
P       E       N   NN  E       L       O    O  P       E
P       EEEEEE  N    N  EEEEEE  LLLLLL   OOOO   P       EEEEEE
```

(version 2005).

F. Salvat, J.M. Fernandez-Varea and J. Sempau

Facultat de Fisica (ECM). Universitat de Barcelona.
Diagonal 647. 08028 Barcelona. Spain

---- GENERAL INFORMATION ----

The FORTRAN 77 code system PENELOPE performs Monte Carlo simulation of coupled electron and photon transport in arbitrary materials. The name is an acronym that stands for PENetration and Energy LOss of Positrons and Electrons in matter; photon transport was introduced later. The simulation algorithm is based on a scattering model that combines numerical databases with analytical cross-section models for the different interaction mechanisms and is applicable to energies (kinetic energies in the case of electrons and positrons) from 50 eV to 1 GeV. PENELOPE generates random electron-photon showers in complex material structures consisting of any number of distinct homogeneous regions (bodies) of different compositions.

PENELOPE is structured as a set of subroutine packages with compact input/output (essentially all particle-state and geometry variables are transferred through a single common block). To have complete control on the simulation and to get access to all possible information, the user should write his/her own MAIN program. This program only has to control the evolution of the simulated tracks and keep score of the relevant quantities. Since PENELOPE does a great part of the simulation work internally, no previous knowledge of the intricate features of scattering and transport theories is required. Nevertheless, to spare occasional users from having to write their MAIN program, the PENELOPE distribution package contains a generic MAIN program called PENMAIN (see Appendix 1), which is flexible enough to solve a broad variety of practical problems.

For the sake of brevity, we use the term ''particle'' to refer to either electrons, positrons or photons. Interactions with the medium cause particles to lose energy, change their direction of movement and, occasionally, produce secondary particles. PENELOPE incorporates a scattering model that combines information from numerical databases with simple analytical differential cross section models. The considered interactions and the corresponding differential cross sections are the following:

A) Elastic scattering of electrons and positrons:
   -- For energies below 100 MeV, relativistic (Dirac) partial-wave differential cross sections generated by using the computer code ELSEPA.
   -- For E>100 MeV, modified Wentzel (MW) differential cross section model with parameters determined from the mean free path and first and second transport mean free paths computed with ELSEPA.

B) Inelastic collisions of electrons and positrons: Born differential cross section obtained from the Sternheimer-Liljequist generalised oscillator strength model, including the density-effect correction. The differential cross section is renormalised to reproduce the

collision stopping power read from the input file.

C) Bremsstrahlung emission by electrons and positrons: the energy of the emitted photons is sampled from energy-loss differential cross sections obtained from the scaled cross section tabulated by Seltzer and Berger. These differential cross sections are renormalised to reproduce the radiative stopping power read from the input file. The intrinsic angular distribution of emitted photons is described by an analytical expression with parameters determined by fitting the benchmark partial-wave shape functions of Kissel, Quarles and Pratt.

D) Positron annihilation: Heitler differential cross section for two-photon annihilation in flight.

E) Inner-shell ionisation by electron and positron impact: total cross sections for ionisation of K, L and M shells, obtained from an optical-data (virtual quanta) model. Correlations between the energy lost by the projectile and the emitted fluorescent radiation (Auger electrons and x rays) are disregarded.

F) Coherent (Rayleigh) scattering of photons: Born differential cross section with an analytical atomic form factor.

G) Incoherent (Compton) scattering of photons: differential cross section calculated using the relativistic impulse approximation with analytical one-electron Compton profiles.

H) Photoelectric absorption of photons: total atomic cross sections and K-, L- and M-shell partial cross sections from the LLNL Evaluated Photon Data Library. The initial direction of photoelectrons is sampled from Sauter's K-shell hydrogenic differential cross section.

I) Electron-positron pair production: total cross sections obtained from the XCOM program of Berger and Hubbell. The initial kinetic energies of the produced particles are sampled from the Bethe-Heitler differential cross section, with exponential screening and Coulomb correction, empirically modified to improve its reliability for energies near the pair-production threshold.

The simulation of photon transport follows the usual detailed procedure, i.e. all the interaction events in a photon history are simulated in chronological succession.

The simulation of electron and positron tracks is performed by means of a mixed (class II) algorithm. Individual hard elastic collisions, hard inelastic interactions and hard bremsstrahlung emission are simulated in a detailed way, i.e. by random sampling from the corresponding restricted differential cross sections. The track of a particle between successive hard interactions, or between a hard interaction and the crossing of an interface (i.e. a surface that separates two media with different compositions) is generated as a series of steps of limited length (see below). The combined effect of all (usually many) soft interactions that occur along a step is simulated as a single ''artificial'' soft event (a random hinge) where the particle loses energy and changes its direction of motion. The energy loss and angular deflection at the hinge are generated according to a multiple-scattering approach that yields energy loss distributions and angular distributions with the correct mean and variance.

Secondary particles emitted with initial energy larger than the absorption energy -see below- are stored, and simulated after completion of each primary track. Secondary particles are produced in direct interactions (hard inelastic collisions, hard bremsstrahlung emission, positron annihilation, Compton scattering, photoelectric absorption and pair production) and as fluorescent radiation (characteristic x rays and Auger electrons). PENELOPE simulates the emission of characteristic x rays and Auger electrons that result from vacancies produced in K, L and M shells by photoelectric absorption and Compton scattering of photons and by electron/positron impact. The relaxation of these vacancies is followed until the K, L and M shells are filled up, i.e. until the

vacancies have migrated to N and outer shells. The adopted transition probabilities were extracted from the LLNL Evaluated Atomic Data Library. The energies of the emitted x rays are set equal to their experimental values; the initial energies of Auger electrons are calculated from the ionisation energies of the active shells, i.e. neglecting the relaxation of the ionised atom.

A detailed description of the cross sections and simulation methods adopted in PENELOPE, and a discussion of their reliability and domains of validity, is given in the following references:

- J. Baro, J. Sempau, J.M. Fernandez-Varea and F. Salvat, ''PENELOPE: An algorithm for Monte Carlo simulation of the penetration and energy loss of electrons and positrons in matter''. Nucl. Instrum. and Meth. B 100 (1995) 31-46.
- J. Sempau, E. Acosta, J. Baro, J.M. Fernandez-Varea and F. Salvat, ''An algorithm for Monte Carlo simulation of coupled electron-photon transport''. Nucl. Instrum. and Meth. B 132 (1997) 377-390.
- J. Sempau, J.M. Fernandez-Varea, E. Acosta and F. Salvat, ''Experimental benchmarks of the Monte Carlo code PENELOPE'', Nucl. Instrum. and Meth. B 207 (2003) 107-123.
- F. Salvat, A. Jablonski and C.J. Powell, ''ELSEPA-- Dirac partial-wave calculation of elastic scattering of electrons and positrons by atoms, positive ions and molecules''. Comput. Phys. Commun. 165 (2005) 157-190.
- F. Salvat, J.M. Fernandez-Varea and J. Sempau, ''PENELOPE, A Code System for Monte Carlo Simulation of Electron and Photon Transport''. OECD Nuclear Energy Agency (Issy-les-Moulineaux, France, 2003). The PDF version of this document can be downloaded from the web site of the Nuclear Energy Agency Data Bank (www.nea.fr).

---- MATERIAL DATA FILE ----

PENELOPE reads the required information about each material (which includes tables of physical properties, interaction cross sections and physical information) from the input material data file (identified as UNIT=IRD in the code source listing). The material data file is created by means of the auxiliary program MATERIAL, which extracts atomic interaction data from the database. This program runs interactively and is self-explanatory. Basic information about the considered material is supplied by the user from the keyboard, in response to prompts from the program. The required information is: 1) chemical composition (i.e. elements present and stoichiometric index of each element), 2) mass density, 3) mean excitation energy and 4) energy and oscillator strength of plasmon excitations. Alternatively, for a set of 280 prepared materials, the program MATERIAL can read data directly from the PDCOMPOS.P05 file (see below). Alloys and mixtures are treated as compounds, with stoichiometric indices equal, or proportional, to the percent number of atoms of the elements.

The database consists of the following 797 ASCII files,

-- PDATCONF.P05: atomic ground-state configurations, ionisation energies and central values of the one-electron shell Compton profiles for the elements, from hydrogen to einsteinium.
-- PDCOMPOS.P05: prepared composition data for 280 different materials of radiological interest (adapted from Berger, NISTIR 4999, 1992).
-- PDEFLIST.P05: list of materials included in the PDCOMPOS.P05 file, with their identification numbers (see Appendix 2).
-- PDRELAX.P05: data on atomic relaxation, extracted from the LLNL Evaluated Atomic Data Library. To describe atomic transitions, each atomic shell is assigned a numerical label IS as follows;

```
   1 = K  (1s1/2),   11 = N2 (4p1/2),   21 = O5 (5d5/2),
   2 = L1 (2s1/2),   12 = N3 (4p3/2),   22 = O6 (5f5/2),
   3 = L2 (2p1/2),   13 = N4 (4d3/2),   23 = O7 (5f7/2),
```

```
          4 = L3 (2p3/2),   14 = N5 (4d5/2),   24 = P1 (6s1/2),
          5 = M1 (3s1/2),   15 = N6 (4f5/2),   25 = P2 (6p1/2),
          6 = M2 (3p1/2),   16 = N7 (4f7/2),   26 = P3 (6p3/2),
          7 = M3 (3p3/2),   17 = O1 (5s1/2),   27 = P4 (6d3/2),
          8 = M4 (3d3/2),   18 = O2 (5p1/2),   28 = P5 (6d5/2),
          9 = M5 (3d5/2),   19 = O3 (5p3/2),   29 = Q1 (7s1/2),
         10 = N1 (4s1/2),   20 = O4 (5d3/2),   30 = outer shells.
      In the case of non-radiative transitions the label 30 indicates
      shells beyond the N7 shell.
-- 99 files named PDEELZZ.P05 with ZZ=atomic number (001-099). These
      files contain tables of total cross sections and first and second
      transport cross sections for elastic scattering of electrons and
      positrons by neutral free atoms, generated by the program ELSEPA.
      The same grid of energies, that covers the interval from 50 eV
      up to 1 GeV, is used for all elements.
-- 99 files named EELDXZZZ.P05 with ZZZ=atomic number (001-099). These
      files contain tables of differential cross sections for elastic
      scattering of electrons by free neutral atoms. They where
      generated by using the program ELSEPA and cover the energy
      interval from 50 eV to 100 MeV.
-- 99 files named PELDXZZZ.P05 with ZZZ=atomic number (001-099). These
      files contain tables of differential cross sections for elastic
      scattering of positrons by free neutral atoms. They where
      generated by using the program ELSEPA and cover the energy
      interval from 50 eV to 100 MeV.
-- 99 files named PDEBRZZ.P05 that contain electron bremsstrahlung data.
      These files were produced from the database of Seltzer and Berger.
      The same grid of energies for all elements.
-- PDBRANG.P05: parameters of the intrinsic angular distribution of
      bremsstrahlung photons. Determined by fitting the set of benchmark
      partial-wave shape functions of Kissel, Quarles and Pratt.
-- 99 files named PDGPPZZ.P05 with cross sections for pair production
      in the field of neutral atoms (sum of contributions from pair and
      triplet production), obtained from the XCOM program of Berger and
      Hubbell. The same energy grid for all elements.
-- 99 files named PDGPHZZ.P05, containing total atomic photoelectric
      cross sections and partial cross sections for inner (K, L and M)
      shells, generated from the EPDL97 data library of Cullen et al.
-- 99 files named PDESIZZ.P05 with cross sections for ionisation of
      inner (K, L and M) shells by electron impact.
-- 99 files named PDPSIZZ.P05 with cross sections for ionisation of
      inner (K, L and M) shells by positron impact.
Notice that PENELOPE does not work for elements with atomic number Z>99.

   The energy-dependent quantities tabulated in the input material data
file determine the most relevant characteristics of the scattering
model. Thus, the MW differential cross section for electron and positron
elastic scattering is completely defined by the mean free paths and
transport mean free paths. Collision and radiative stopping powers read
from the input file are used to renormalise the built-in analytical
differential cross sections, i.e. these are multiplied by an
energy-dependent factor such that the input stopping powers are exactly
reproduced. The mean free paths used in the simulation of photon
transport are directly obtained from the input cross sections. Notice
that one can modify the scattering model, without altering the program,
by simply modifying these energy-dependent quantities in the input
material data file.

   To simulate geometrical structures with several materials, the
corresponding material data files generated by the program MATERIAL must
be concatenated in a single input file. PENELOPE labels the M-th
material in this file with the index MAT=M, which is used during the
simulation to identify the material where the particle moves. The
maximum number of different materials that PENELOPE can handle
```

simultaneously is fixed by the parameter MAXMAT, which in the present version is set equal to 10. The required memory storage is roughly proportional to the value of this parameter. The user can increase MAXMAT by editing the program source files. Notice that the value of MAXMAT _must_ be the same in all subprograms.

---- STRUCTURE OF THE MAIN PROGRAM ----

As mentioned above, PENELOPE should be complemented with a steering MAIN program, which controls the geometry and the evolution of tracks, keeps score of the relevant quantities, and performs the required averages at the end of the simulation.

The connection of PENELOPE and the MAIN program is done via the named common block

```
--> COMMON/TRACK/E,X,Y,Z,U,V,W,WGHT,KPAR,IBODY,MAT,ILB(5)
```

that contains the following particle state variables and labels:

- KPAR: kind of particle (1: electron, 2: photon, 3: positron).
- E: current particle energy (eV) (kinetic energy for electrons and positrons).
- X, Y, Z: position coordinates (cm).
- U, V, W: direction cosines of the direction of movement.
- WGHT: in analogue simulations, this is a dummy variable. When using variance-reduction methods, the particle weight can be stored here.
- IBODY: this auxiliary flag serves to identify different bodies in complex material structures.
- MAT: material where the particle moves (i.e. the one in the body labelled IBODY).
- ILB(5): an auxiliary array of 5 labels that describe the origin of secondary particles. It is useful e.g. in studying partial contributions from particles originated by a given process.

The position coordinates (X,Y,Z) and the direction cosines (U,V,W) of the direction of movement are referred to the ''laboratory'' frame, which can be arbitrarily defined. During the simulation, all energies and lengths are expressed in eV and cm, respectively.

The label KPAR identifies the kind of particle: KPAR=1, electron; KPAR=2, photon; KPAR=3, positron. A particle that moves in material M is assumed to be absorbed when its energy becomes less than a value EABS(KPAR,M) (in eV) specified by the user. Positrons are assumed to annihilate, by emission of two photons, when absorbed. In dose calculations, EABS(KPAR,M) should be determined so that the residual range of particles with this energy is smaller than the dimensions of the volume bins used to tally the spatial dose distribution. As the interaction database is limited to energies above 50 eV, absorption energies EABS(KPAR,M) must be larger than this value.

The transport algorithm for electrons and positrons in each material M is controlled by the following simulation parameters,

- C1(M): Average angular deflection, 1-<cos(theta)>, produced by multiple elastic scattering along a path length equal to the mean free path between hard elastic events. C1(M) should be of the order of 0.05; its maximum allowed value is 0.2.
- C2(M): Maximum average fractional energy loss between consecutive hard elastic events. Usually, a value of the order of 0.05 is adequate. The maximum allowed value of C2(M) is 0.2.
- WCC(M): Cutoff energy loss (in eV) for hard inelastic collisions.
- WCR(M): Cutoff energy loss (in eV) for hard bremsstrahlung emission.

These parameters determine the accuracy and speed of the simulation. To ensure accuracy, C1(M) and C2(M) should have small values (of the order of 0.05 or so). With larger values of C1(M) and C2(M) the simulation

gets faster, at the expense of a certain loss in accuracy. The cutoff energies WCC(M) and WCR(M) mainly influence the simulated energy distributions. The simulation speeds up by using larger cutoff energies, but if these are too large, the simulated energy distributions may be somewhat distorted. In practice, simulated energy distributions are found to be insensitive to the adopted values of WCC(M) and WCR(M) when these are less than the bin width used to tally the energy histograms. Thus, the desired energy resolution determines the maximum allowed cutoff energies. The reliability of the whole simulation rests on a single condition: the number of steps (or random hinges) per primary track must be ''statistically sufficient'', i.e. larger than 10 or so.

The simulation package is initialised from the MAIN program with the statement

```
--> CALL PEINIT(EPMAX,NMAT,IRD,IWR,INFO)
```

Subroutine PEINIT reads the data files of the different materials, evaluates relevant scattering properties and prepares look-up tables of energy-dependent quantities that are used during the simulation. Its input arguments are:

- EPMAX: Maximum energy (in eV) of the simulated particles. Notice that if the primary particles are positrons with initial kinetic energy EP, the maximum energy of annihilation photons equals EPMAX=1.21*(EP+5.11E5) eV; in this special case, the maximum energy is larger than the initial kinetic energy.
- NMAT: Number of different materials (less than or equal to MAXMAT).
- IRD : Input unit.
- IWD : Output unit.
- INFO: Determines the amount of information that is written in the output file. Minimal for INFO=0 and increasingly detailed for INFO=1, 2, ...

For the preliminary computations, PEINIT needs to know the absorption energies EABS(KPAR,M) and the simulation parameters C1(M), C2(M), WCC(M) and WCR(M). This information is introduced through the named common block

```
--> COMMON/CSIMPA/EABS(3,MAXMAT),C1(MAXMAT),C2(MAXMAT),WCC(MAXMAT),
    1  WCR(MAXMAT)
```

that has to be loaded before invoking the PEINIT subroutine. Notice that we can employ different values of the simulation parameters for different materials. This possibility can be used to speed up the simulation in regions of lesser interest.

PENELOPE has been structured in such a way that a particle track is generated as a sequence of track segments (free flights or ''jumps''); at the end of each segment the particle suffers an interaction event (a ''knock'') where it loses energy, changes its direction of movement and, in certain cases, produces secondary particles. Electron-photon showers are simulated by successively calling the following subroutines:

--> SUBROUTINE CLEANS
Initiates the secondary stack.

--> SUBROUTINE START
For electrons and positrons, this subroutine forces the following interaction event to be a soft artificial one. It must be called before starting a new -primary or secondary- track and also when a particle crosses an interface.

Calling START is strictly necessary only for electrons and positrons; for photons this subroutine has no physical effect. However, it is advisable to call START for any kind of particle since it checks whether the energy is within the expected range, and can thus help to detect ''bugs'' in the MAIN program.

--> SUBROUTINE JUMP(DSMAX,DS)
Determines the length DS of the track segment to the following interaction event.

The input parameter DSMAX defines the maximum allowed step length for electrons/positrons; for photons, it has no effect. To

limit the step length, PENELOPE places delta interactions along the particle track. These are fictitious interactions that do not alter the physical state of the particle. Their only effect is to interrupt the sequence of simulation operations (which requires altering the values of inner control variables to allow the simulation to be resumed consistently). The combined effect of the soft interactions that occur along the step preceding the delta interaction is simulated by the usual random hinge method. Owing to the Markovian nature of hard interactions, the introduction of delta interactions does not alter the distribution of path lengths between consecutive hard events.

As mentioned above, to ensure the reliability of the mixed simulation algorithm, the number of artificial soft events per particle track in each body should be larger than, say, 10. For relatively thick bodies (say, thicker than 10 times the mean free path between hard interactions), this condition is automatically satisfied. In this case we can switch off the step-length control by setting DSMAX=1.0D35 (or any other very large value). On the other hand, when the particle moves in a thin body, DSMAX should be given a value of the order of one tenth of the ''thickness'' of that body. Limiting the step length is also necessary to simulate particle transport in external electromagnetic fields.

--> SUBROUTINE KNOCK(DE,ICOL)

Simulates an interaction event, computes new energy and direction of movement, and stores the initial states of the generated secondary particles, if any. On output, the arguments are:

DE: deposited energy in the course of the event,

ICOL: kind of event that has been simulated, according to the following convention,

-- Electrons (KPAR=1)

ICOL=1, artificial soft event (random hinge).
=2, hard elastic collision.
=3, hard inelastic collision.
=4, hard bremsstrahlung emission.
=5, inner-shell ionisation.
=7, delta interaction

-- Photons (KPAR=2):

ICOL=1, coherent (Rayleigh) scattering.
=2, incoherent (Compton) scattering.
=3, photoelectric absorption.
=4, electron-positron pair production.
=7, delta interaction

-- Positrons (KPAR=3):

ICOL=1, artificial soft event (random hinge).
=2, hard elastic collision.
=3, hard inelastic collision.
=4, hard bremsstrahlung emission.
=5, inner-shell ionisation.
=6, annihilation.
=7, delta interaction

The value ICOL=8 is used for the ''auxiliary'' interactions (an additional mechanism that may be defined by the user, e.g. to simulate photonuclear interactions).

--> SUBROUTINE SECPAR(LEFT)

Sets the initial state of a secondary particle and removes it from the secondary stack. The output value LEFT is the number of secondary particles remaining in the stack at the calling time.

--> SUBROUTINE STORES(E,X,Y,Z,U,V,W,WGHT,KPAR,ILB)

Stores a particle in the secondary stack. Arguments have the same meaning as in COMMON/TRACK/, but refer to the particle that is being stored. The variables IBODY and MAT are set equal to the current values in COMMON/TRACK/.

Calling STORES from the MAIN program is useful e.g. to store particles produced by splitting, a variance-reduction method.

The sequence of calls to generate a random track is independent of the kind of particle that is being simulated. The generation of random showers proceeds as follows:

1) Set the initial state of the primary particle, i.e. assign values to the state variables KPAR, E, position coordinates =(X,Y,Z) and direction of movement =(U,V,W). Specify the body and material where the particle moves by defining the values of IBODY and MAT, respectively. Optionally, set the values of WGHT and ILB.

2) CALL CLEANS to initialise the secondary stack.

3) CALL START to initiate the simulation of the track.

4) CALL JUMP(DSMAX,DS) to determine the length DS of the next track segment (for electrons and positrons, DS will never exceed the input value DSMAX).

5) Compute the position of the following event:
   -- If the track has crossed an interface, stop the particle at the position where the track intersects the interface.
      Change to the new body and material (the ones behind the interface) by redefining the values of IBODY and MAT.
      -- When the particle escapes from the system, the simulation of the track has been finished.
         Increment counters and go to step 7.
      Go to step 3.

6) CALL KNOCK(DE,ICOL) to simulate the following event.
   -- If the energy is less than EABS(KPAR,MAT), end the track, increment counters and go to step 7.
   -- Go to step 4.

7) CALL SECPAR(LEFT) to start the track of a particle in the secondary stack (this particle is then automatically removed from the stack).
   -- If LEFT>0, go to step 3 (the initial state of a secondary particle has already been set).
   -- If LEFT=0, the simulation of the shower produced by the primary particle has been completed. Go to step 1 to generate a new primary track (or leave the simulation loop after simulating a sufficiently large number of showers).

Notice that subroutines JUMP and KNOCK keep the position coordinates unaltered; the positions of successive events have to be followed by the MAIN program (simply by performing a displacement of length DS along the direction of movement after each call to JUMP). The energy of the particle is automatically reduced by subroutine KNOCK, after generating the energy loss from the relevant probability distribution function. KNOCK also modifies the direction of movement according to the scattering angles of the simulated event. Thus, at the output of KNOCK, the values of the energy E, the position (X,Y,Z) and the direction of movement (U,V,W) define the particle state immediately after the interaction event.

In order to avoid problems related to possible overflows of the secondary stack, when a secondary particle is produced its energy is temporarily assumed to be locally deposited. Hence, the energy E of a secondary must be subtracted from the corresponding dose counter when the secondary track is started. Occasional overflows of the secondary stack are remedied by eliminating the less energetic secondary electron or photon in the stack (positrons are not eliminated since they will eventually produce quite energetic annihilation radiation). As the main effect of secondary particles is to spread out the energy deposited by the primary one, the elimination of the less energetic secondary electrons and photons should not invalidate local dose calculations.

It is the responsibility of the user to avoid calling subroutines JUMP and KNOCK with energies outside the interval (EABS(KPAR,M),EMAX). This could cause inaccurate interpolation of the cross sections. The simulation is aborted (and an error message is printed in unit 6) if the

conditions EABS(KPAR,M)<E<EMAX are not satisfied when a primary or secondary track is started (whenever subroutine START is called at the beginning of that track).

Pseudo-random numbers uniformly distributed in the interval (0,1) are supplied by function RAND(DUMMY) that implements a 32-bit generator due to L'Ecuyer. The seeds of the generator (two integers) are transferred from the main program through the named common block RSEED. The random number generator can be changed by merely replacing that FUNCTION subprogram (the new one has to have a single dummy argument). Some compilers incorporate an intrinsic random number generator with the same name (but with different argument lists). To avoid conflict, the user should declare RAND as an external function in all subprograms that call it.

Owing to the long execution time, the code will usually be run in batch mode. It is advisable to limit the simulation time rather than the number of tracks to be simulated, since the time required to follow each track is difficult to predict. To this end, one can link a clock routine to the simulation code and stop the computation after exhausting the allotted time.

**** Notice that

1) In the simulation routines, real and integer variables are declared as DOUBLE PRECISION and INTEGER*4, respectively. To prevent type mismatches, it is prudent to use the following IMPLICIT statement
   IMPLICIT DOUBLE PRECISION (A-H,O-Z), INTEGER*4 (I-N)
   in the MAIN program and other user program units.
2) The MAIN program must include the following three common blocks:

```
      COMMON/TRACK/E,X,Y,Z,U,V,W,WGHT,KPAR,IBODY,MAT,ILB(5)
      COMMON/CSIMPA/EABS(3,MAXMAT),C1(MAXMAT),C2(MAXMAT),WCC(MAXMAT),
     1  WCR(MAXMAT)
      COMMON/RSEED/ISEED1,ISEED2
```

As mentioned above, ILB(5) is an array of labels that describe the origin of secondary particles. It is assumed that the user has set ILB(1) equal to 1 (one) when a primary (source) particle history is initiated. PENELOPE then assigns the following labels to each particle in a shower;

- ILB(1): generation of the particle. 1 for primary particles, 2 for their direct descendants, etc.
- ILB(2): kind KPAR of the parent particle, only if ILB(1)>1 (secondary particles).
- ILB(3): interaction mechanism ICOL (see above) that originated the particle, only when ILB(1)>1.
- ILB(4): a non-zero value identifies particles emitted from atomic relaxation events and describes the atomic transition where the particle was released. The numerical value is
  = Z*10**6+IS1*10**4+IS2*100+IS3,
  where Z is the atomic number of the parent atom and IS1, IS2 and IS3 are the numerical labels of the active electron shells (see above).
- ILB(5): this label can be defined by the user; it is transferred to all descendants of the particle.

The ILB label values are delivered by subroutine SECPAR, through common TRACK, and remain unaltered during the simulation of the track.

The subroutine package PENELOPE.F is intended to perform analogue simulation and, therefore, does not include any variance-reduction methods. The source file PENVARED.F contains subroutines to perform splitting (VSPLIT), Russian roulette (VKILL) and interaction forcing (JUMPF, KNOCKF) in an automatic way. Splitting and Russian roulette do not require changes in PENELOPE; the necessary manipulations on the numbers and weights WGHT of particles could be done directly in the main

program. Particles resulting from splitting are stored in the secondary stack by calling subroutine STORES. Interaction forcing implies changing the mean free paths of the forced interactions and, at the same time, redefining the weights of the generated secondary particles. In principle, it is possible to apply interaction forcing from the MAIN program by manipulating the interaction probabilities, that are made available through the named common block CJUMP0. These manipulations are performed automatically by calling the subroutines JUMPF and KNOCKF instead of JUMP and KNOCK.

---- QUADRIC GEOMETRY PACKAGE ----

PENELOPE incorporates the geometry subroutine package PENGEOM, which performs particle tracking in material systems consisting of homogeneous regions (bodies) limited by quadric surfaces. The structure and operation of PENGEOM are described in detail in chapter 5 of the write-up. Here we just mention the information that is essential for using this package.

A quadric surface is defined by the implicit equation

$$F(x,y,z) = AXX*x*x+AXY*x*y+AXZ*x*z+AYY*y*y+AYZ*y*z+AZZ*z*z+AX*x+AY*y+AZ*z+A0 = 0,$$

which includes planes, pairs of planes, spheres, cylinders, cones, ellipsoids, paraboloids, hyperboloids, etc. Positions are referred to the laboratory coordinate system; all lengths are in cm.

In practice, limiting surfaces are frequently known in ''graphical'' form and it may be very difficult to obtain the corresponding quadric parameters. Try with a simple example: calculate the parameters of a circular cylinder of radius R such that its symmetry axis goes through the origin and is parallel to the vector (1,1,1). To facilitate the definition of the geometry, each quadric surface can be specified either through its implicit equation or by means of its reduced form, which is easily visualised, and a few simple geometrical transformations. A reduced quadric is defined by an expression of the form

$$FR(x,y,z) = I1*x*x+I2*y*y+I3*z*z+I4*z+I5 = 0,$$

where the coefficients (indices) I1, I2, I3, I4 and I5 can only take the values -1, 0 or 1. Notice that reduced quadrics have central symmetry about the z-axis, i.e. FR(-x,-y,z)=FR(x,y,z). The possible (real) reduced quadrics are:

reduced form	indices	quadric
z-1=0	0 0 0 1 -1	plane
z*z-1=0	0 0 1 0 -1	pair of parallel planes
x*x+y*y+z*z-1=0	1 1 1 0 -1	sphere
x*x+y*y-1=0	1 1 0 0 -1	cylinder
x*x+y*y-z*z=0	1 1 -1 0 0	cone
x*x-y*y-1=0	1 -1 0 0 -1	hyperbolic cylinder
x*x+y*y-z*z-1=0	1 1 -1 0 -1	one sheet hyperboloid
x*x+y*y-z*z+1=0	1 1 -1 0 +1	two sheet hyperboloid
x*x-z=0	1 0 0 -1 0	parabolic cylinder
x*x+y*y-z=0	1 1 0 -1 0	paraboloid
x*x-y*y-z=0	1 -1 0 -1 0	hyperbolic paraboloid
(... and permutations of x, y and z that preserve the central symmetry with respect to the z-axis).		

A quadric is obtained from the corresponding reduced form by applying the following transformations (in the quoted order):

1) An expansion along the directions of the axes, defined by the scaling factors X-SCALE=a, Y-SCALE=b and Z-SCALE=c. The equation of the scaled quadric is

$$F(x,y,z) = I1*(x/a)**2+I2*(y/b)**2+I3*(z/c)**2+I4*(z/c)+I5 = 0.$$

Thus, for instance, the reduced sphere transforms into an ellipsoid with semiaxes equal to the scaling factors.

2) A rotation, defined through the Euler angles OMEGA, THETA and PHI, which specify a sequence of rotations about the coordinate axes: first a rotation of angle OMEGA about the z-axis, followed by a rotation of angle THETA about the y-axis and, finally, a rotation of angle PHI about the z-axis. Notice that rotations are active; the coordinate axes remain fixed and only the quadric surface is rotated. A positive rotation about a given axis would carry a right-handed screw in the positive direction along the axis. Positive (negative) angles define positive (negative) rotations. The global rotation transforms a plane perpendicular to the z-axis into a plane perpendicular to the direction defined by the polar and azimuthal angles THETA and PHI, respectively. The first rotation R(z,OMEGA) has no effect when the initial (expanded) quadric is symmetric about the z-axis.

3) A shift, defined by the components of the displacement vector (X-SHIFT,Y-SHIFT,Z-SHIFT).

Thus, a quadric is completely specified by giving the set of indices (I1,I2,I3,I4,I5), the scale factors (X-SCALE,Y-SCALE,Z-SCALE), the Euler angles (OMEGA,THETA,PHI) and the displacement vector (X-SHIFT,Y-SHIFT, Z-SHIFT). Any quadric surface can be expressed in this way.

A point with coordinates (x,y,z) is said to be inside a surface F(x,y,z)=0 if F(x,y,z)<0, and outside it if F(x,y,z)>0. A quadric surface divides the space into two exclusive regions that are identified by the sign of F(x,y,z), the surface side pointer. A body can be defined by its limiting quadric surfaces and corresponding side pointers (+1 or -1). Previously defined bodies can also be used to delimit a new body; this is very convenient when the new body contains inclusions or when it is penetrated by other bodies. However, the use of limiting bodies may lengthen the calculation.

To speed up the geometry operations, the bodies of the material system can be grouped into modules (connected volumes, limited by quadric surfaces, that contain one or several bodies); modules can in turn form part of larger modules, and so on. This hierarchic modular structure allows a reduction of the work of the geometry routines, which becomes more effective when the complexity of the system increases. The present version 2005 of PENGEOM contains several new features (a CLONE operation for modules; ''C '' comment lines can be inserted anywhere in the geometry definition file; each body can be made part of an impact detector; and others).

The geometry is defined from the input file (UNIT=IRD in the source code), which consists of a number of data sets that define the different elements (surfaces, bodies and modules). For details on the structure of the geometry definition file see section 5.4 in the write-up (see also the examples in directory GVIEW). Except for trivial cases, the correctness of the geometry definition is difficult to check and, moreover, 3D structures with interpenetrating bodies are difficult to visualise. A pair of programs, named GVIEW2D and GVIEW3D, have been written to display the geometry on the computer screen. These programs use specific computer graphics software and, therefore, they are not portable. The executable files included in the PENELOPE distribution package run on personal computers under Microsoft Windows; they are simple and effective tools for debugging the geometry definition file.

In practical simulations, the following PENGEOM routines are to be invoked from the MAIN program:

--> SUBROUTINE GEOMIN(PARINP,NPINP,NMAT,NBOD,IRD,IWR)

Reads geometry data from the input file and initialises the geometry package.

Input arguments:

```
        PARINP ... Array containing optional parameters, which may
                   replace the ones entered from the input file. This
                   array must be declared in the MAIN program, even
                   when NPINP=0.
        NPINP .... Number of parameters defined in PARINP (positive).
        IRD ...... Input file unit (opened in the main program).
        IWR ...... Output file unit (opened in the main program).
      Output arguments:
        NMAT ..... Number of different materials in full bodies
                   (excluding void regions).
        NBOD ..... Number of defined bodies.
      The program is stopped when a clearly incorrect input datum is
      found, the wrong quantity appears in the last printed line.
-->  SUBROUTINE LOCATE
      Determines the body that contains the point with coordinates
      (X,Y,Z).
      Input values (through COMMON/TRACK/):
        X, Y, Z ... particle position coordinates.
        U, V, W ... direction cosines of the particle velocity.
      Output values (through COMMON/TRACK/):
        IBODY ..... Body where the particle moves.
        MAT  ...... Material in IBODY. The output MAT=0 indicates that
                    the particle is in a void region.
-->  SUBROUTINE STEP(DS,DSEF,NCROSS)
      This subroutine handles the geometrical part of the track
      simulation. The particle starts from the point (X,Y,Z) and
      proceeds to travel a length DS in the direction (U,V,W) within
      the material where it moves. STEP displaces the particle and
      stops it at the end of the step, or just after entering a new
      material. The output value DSEF is the distance travelled within
      the initial material. If the particle enters a void region
      (MAT=0), STEP continues the particle track, as a straight
      segment, until it penetrates a material body or leaves the system
      (the path length through void regions is not included in DSEF).
      When the particle arrives from a void region (MAT=0), it is
      stopped after entering the first material body. The output value
      MAT=0 indicates that the particle has escaped from the system.
      Input-output values (through COMMON/TRACK/):
        X, Y, Z ... Input: coordinates of the initial position.
                    Output: coordinates of the final position.
        U, V, W ... direction cosines of the displacement. They
                    are kept unaltered
        IBODY . ... Input: initial body, i.e. the one that contains the
                    initial position.
                    Output: final body.
        MAT ....... material in body IBODY (automatically changed when
                    the particle crosses an interface).
      Input argument:
        DS ........ distance to travel (unaltered).

      Output arguments:
        DSEF ...... travelled path length before leaving the initial
                    material or completing the jump (less than DS if
                    the track crosses an interface).
        NCROSS .... number of interface crossings (=0 if the particle
                    does not leave the initial material, greater than 0
                    if the particle enters a new material).
```

Before starting the simulation, the user should make sure that the geometry has been defined correctly. To this end, subroutine GEOMIN writes a geometry report in the output file (UNIT=IWR), which is a duplicate of the input definition file. When the input file is formally incorrect, the program stops and an error message is issued in unit IWR, usually just after printing the conflicting information (i.e. very

likely the error is in the last printed line of the geometry report). When the geometry definition is formally correct, the only differences between the input file and the output report are the labels assigned to the different surfaces, bodies and modules; in the output report, these elements are numbered in strictly increasing order. It is important to bear in mind that PENGEOM internally uses this sequential labelling to identify bodies and surfaces. Knowing the internal number label assigned to each element is necessary for scoring purposes, e.g. to determine the distribution of energy deposited within a particular body, and also to apply certain variance reduction techniques. The new version 2005 of the viewer GVIEW2D allows two display modes, ''materials'' and ''bodies''; the latter is useful for finding the internal number labels of bodies and modules and also for defining detectors embedded in homogeneous media.

---- EXAMPLES OF MAIN PROGRAMS ----

In principle, the user should provide the MAIN program for each specific geometry. The distribution package includes various examples of MAIN programs for simple geometries (slab and cylindrical) and for general quadric geometries, with limited scoring and variance reduction options. For details on the operation of these codes, see the heading comments in the corresponding source files.

-- PENSLAB

The program PENSLAB simulates electron/photon showers within a material slab. It illustrates the use of the simulation routines for the simplest geometry (as geometry operations are very simple, this program is faster than the ones described below). PENSLAB generates detailed information on many quantities and distributions of physical interest.

The slab is limited by the planes Z=0 and Z=thickness (its lateral extension is assumed to be infinite, i.e. much larger than the maximum range of the particles). Primary particles of a given kind, KPARP, are emitted from a point source at the position (SX0,SY0,SZ0), either with fixed energy SE0 or with a specified (histogram-like) energy spectrum. The initial direction of the primary particles is sampled uniformly inside a cone of semi-aperture SALPHA and with central axis in the direction (STHETA,SPHI). Thus, SALPHA=0 defines a monodirectional source and SALPHA=180 deg corresponds to an isotropic source.

A dump/resume option allows the user to stop the simulation at any time and to resume it from the last dumping point in a completely consistent way. The program can also write simulation results in the output files at regular time intervals. This option is useful to check the progress of long simulations. It also allows running the program with a long execution time and stopping it when the required statistical uncertainty has been reached.

-- PENCYL

The program PENCYL simulates electron-photon showers in multilayered cylindrical structures. The material system consists of one or several layers of given thicknesses. Each layer contains a number of concentric homogeneous rings of given compositions and radii (and thickness equal to that of the layer). The layers are perpendicular to the Z-axis and the centre of the rings in each layer is specified by giving its X and Y coordinates. When all the centres are on the Z-axis, the geometrical structure is symmetrical about the Z-axis.

Primary particles of a given kind, KPARP, are emitted from the active volume of the source, either with fixed energy SE0 or with a

specified (histogram-like) energy spectrum. The initial direction of the primary particles is sampled uniformly inside a cone of (semi-) aperture SALPHA and with central axis in the direction (STHETA,SPHI). The program can simulate two different types of sources:

a) An external source with uniform activity over a cylindrical volume, which is defined separately from the geometry of the material system.
b) A set of internal sources spread over specified bodies, each one with uniform activity concentration. The original position of the primary particle is sampled uniformly within the active cylinder or ring, which is selected randomly with probability proportional to the total activity in its volume.

In the distributed form of the program, we assume that both the source and the material structure are symmetrical about the Z-axis, because this eliminates the dependence on the azimuthal angle PHI. It is possible to consider geometries that are not axially symmetrical, but then the program only delivers values averaged over PHI. To obtain the dependence of the angular distributions on the azimuthal angle, we need to increase the value of the parameter NBPHM (the maximum number of bins for PHI, which is set equal to 1 in the distributed source file) and, in the input data file, set NBPH equal to NBPHM.

The source file PENCYL.F includes a (self-contained) set of geometry routines for tracking particles through multilayered cylindrical structures. These routines can be used for simulation even when the source is off-axis. Cylindrical geometries can be viewed with the program GVIEWC (which is similar to GVIEW2D and runs under Microsoft Windows). This program reads the geometry definition directly from the input file of PENCYL and displays a two-dimensional map of the materials intersected by the window plane on the screen. It is useful for debugging the geometry definition list.

PENCYL delivers detailed information on the transport and energy deposition, which includes energy and angular distributions of emerging particles, depth-dose distribution, depth-distribution of deposited charge, distributions of deposited energy in selected materials and 2D (depth-radius) dose and deposited charge distributions in selected bodies (cylinders). This program can be directly used to study radiation transport in a wide variety of practical systems, e.g. planar ionisation chambers, cylindrical scintillation detectors, solid state detectors and multilayered structures.

A dump/resume option allows the user to stop the simulation at any time and to resume it from the last dumping point in a completely consistent way. The program also offers the option of writing simulation results in the output files at regular time intervals.

-- PENMAIN

This is a generic main program that performs simulations of electron-photon transport in complex material structures. PENMAIN is devised to allow occasional users to employ PENELOPE without having to write their main program. The geometry of the material system is described by means of the package PENGEOM, presented above, which is able to handle complicated geometries very efficiently (provided only the user takes care of defining a sufficiently ramified genealogical tree of modules). The operation of PENMAIN is completely controlled from the input data files (see Appendix 1). Although it is impossible to cover all possible cases with a ''closed'' program, PENMAIN is flexible enough to solve a broad class of practical problems.

In the default mode, PENMAIN assumes that primary particles of a

given kind are emitted from a point source, either with fixed energy or with a specified (histogram-like) energy spectrum. The initial direction of the primary particles is sampled uniformly within a cone of given (semi-) aperture and direction. Alternatively, the program can read the initial state variables of ''primary'' particles from a pre-calculated phase-space file. This option is useful for splitting the simulation of complex problems into several consecutive stages.

PENMAIN provides global simulation results such as the energy and angular distributions of particles that emerge from the material system, the average energy deposited in each body, etc. To generate more specific information, the user can define impact detectors and energy-deposition detectors. An impact detector consists of a set of active (non-void) bodies, which must have been defined as parts of the geometry. The output spectrum from an impact detector is the energy distribution of particles that entered any of the active bodies coming from a body that is not active (i.e. that is not part of the detector). Optionally, the state variables of all particles that enter an impact detector can be written in a phase-space file. An energy-deposition detector consists of a set of active bodies; the output spectrum is the distribution of absorbed energy (per shower) in the active bodies. Optionally, a deposited dose map can be tallied on a Cartesian mesh defined by the user.

The program PENMAIN is designed to be run on single-processor computers. A dump/resume option allows us to stop the simulation at any time and to resume it from the last dumping point in a completely consistent way. The program also includes the option of writing simulation results in the output files at regular time intervals.

WARNING: In output files of programs PENSLAB, PENCYL and PENMAIN the terms ''transmitted'' and ''backscattered'' are used to denote particles that leave the material system moving upwards (W>0) and downwards (W<0), respectively. Notice that this agrees with the usual meaning of these terms only when primary particles impinge on the system coming from below (i.e. with W>0).

---- INSTALLATION ----

The FORTRAN 77 source files of PENELOPE, the auxiliary programs, the database and the documentation files are distributed as a single ZIP compressed file named PENELOPE.ZIP. To install the code system on your computer, you only have to inflate (unzip) this file, keeping the original directory structure, and copy its content into the root directory of your hard disk. The directory structure and contents of the PENELOPE code system are the following:

-- Directory FSOURCE (FORTRAN source files, 5 files):
- PENELOPE.F ..... basic subroutine package for simulation of electron-photon showers in unlimited media.
- PENGEOM.F ..... subroutine package for tracking particles through modular quadric geometries (up to 2500 surfaces and 1250 bodies).
- PENVARED.F ..... variance-reduction subroutines (splitting, Russian roulette and interaction forcing).
- TIMER.F ..... timing subroutines. Although they do work with several different FORTRAN compilers, their portability is not guaranteed.
- MATERIAL.F ..... main program to generate material data files.

To obtain the executable binary file of MATERIAL, compile and link the files MATERIAL.F and PENELOPE.F. This executable file must be

placed and run in the same subdirectory as the database files (PENDBASE).

-- Directory DOC (documentation, 4 files):
- MANUAL.TXT ..... this ascii file.
- MANUAL.PDF ..... this file in pdf format.
- TUTORIAL.PDF ... a guided tour through the PENELOPE code system. Includes brief descriptions of the programs and subroutine packages, instructions to install the system, and to build binary executable files.
- PENELOPE_2003_NEA.PDF ... very detailed documentation. Covers basic aspects of Monte Carlo simulation methods, electron and photon interaction models, multiple scattering approximations for electrons and positrons, constructive quadric geometries, installation and operation of the code system, numerical methods,... WARNING: Version 2005 contains a few extensions that are not described in this document.

-- Directory MAINS (examples of MAIN programs, 3 subdirectories):
-- Directory PENSLAB. Contains the main program PENSLAB.F for particle transport in a slab, an example of input file and the associated material data file. The file PENSLAB.GNU is a GNUPLOT script (see below) that displays the continuous distributions generated by the program PENSLAB; PSOURCE.GNU displays the energy spectrum of primary particles (if the source is not monoenergetic).
-- Directory PENCYL. Contains the main program PENCYL.F for particle transport in material systems with cylindrical structures, the geometry viewer GVIEWC.EXE for these structures, examples of input and material-data files and GNUPLOT scripts to visualise the results.
-- Directory PENMAIN. Contains the generic main program PENMAIN.F for particle transport in quadric geometries, input files for two sample problems and various GNUPLOT scripts.

The executable files of PENSLAB, PENCYL and PENMAIN are obtained by compiling and linking the following groups of source files:
PENSLAB: PENSLAB.F, PENELOPE.F, PENVARED.F, TIMER.F
PENCYL : PENCYL.F, PENELOPE.F, PENVARED.F, TIMER.F
PENMAIN: PENMAIN.F, PENELOPE.F, PENGEOM.F, PENVARED.F, TIMER.F

The programs PENSLAB, PENCYL and PENMAIN generate multiple files with simulated probability distributions. Each output file has a heading describing its content, which is in a format ready for visualisation with a plotting program. We use GNUPLOT, which is small in size, available for various platforms (including Linux and Windows) and free (distribution sites are listed at the Gnuplot Central site, http://www.gnuplot.info). Each main program is accompanied by GNUPLOT scripts that, if you have GNUPLOT installed on your computer, can be used to display on your terminal the probability distributions evaluated by the simulation code. For instance, after running PENSLAB you can visualise the results by simply 1) copying the file PENSLAB.GNU from the directory PENSLAB to the directory that contains the results and 2) entering the command ''WGNUPLOT PENSLAB.GNU'' (or clicking the icon).

-- Directory PENDBASE: PENELOPE database. 797 files with the extension ''.P05'' (see above).

-- Directory OTHER: Consists of the following 5 subdirectories,
-- GVIEW. Contains the geometry viewers GVIEW2D, GVIEW3D (which are operable under Microsoft Windows), and several examples of geometry definition files.

-- TABLES. This directory contains the program TABLES.F, which reads a material data file and generates tables of interaction data (cross sections, mean free paths, ranges, ...), and the GNUPLOT script TABLES.GNU, which plots the most relevant of these quantities as functions of the energy. The program TABLES also calculates interpolated values of these quantities at specified energies.
-- EMFIELDS. Contains the subroutine package PENFIELD.F, which does simulation of electron/positron transport under external static magnetic (and electric) fields, and examples of programs that use it.
-- SHOWER. Contains a single binary file named SHOWER.EXE, which operates under Microsoft Windows. This code generates electron-photon showers within a slab (of one of the 280 materials defined in PDCOMPOS.P05) and displays the showers projected on the screen plane. To use SHOWER, just copy the file SHOWER.EXE into the directory PENDBASE and run it from there. This little tool is particularly useful for teaching purposes, it makes radiation physics ''visible''.

The simulation programs are written in standard FORTRAN 77 language, so that they should run on any computer with a FORTRAN compiler. The only exception is the clock subroutine, which may need to be adapted to your computer's compiler.

---- APPENDIX 1 (Operation of PENMAIN.F) ----

The program PENMAIN.F performs Monte Carlo simulation of electron-photon showers in material structures described with the constructive quadric geometry package PENGEOM.F. To specify the kind and amount of information that the program must generate, the user can define two different types of detectors, as well as a Cartesian mesh for local dose tallying.

In the default mode, the program assumes that primary particles of a given kind, KPARP, are emitted from a point source, with coordinates (SX0,SY0,SZ0), either with fixed energy SE0 or with a specified (histogram-like) energy spectrum. The initial direction of the primary particles is sampled uniformly within a cone of semi-aperture SALPHA and central axis in the direction (STHETA, SPHI). Thus, the case SALPHA=0 defines a monodirectional source and SALPHA=180 deg corresponds to an isotropic source. When SX0=SY0=0 and STHETA=0 or 180 deg, the source is axially symmetrical about the Z-axis.

Alternatively, the program can read the initial state variables of ''primary'' particles from a pre-calculated phase-space file. This option may be used to split the simulation of complex problems into several consecutive stages.

********** Structure of the input data file.

Each line in the input data file consists of a 6-character keyword (columns 1-6) followed either by numerical data (in free format) or by a character string, which start at the 8th column. Keywords are explicitly used/verified by the program (which is case sensitive!). Notice also that the order of the data lines is important. The keyword ''_____'' (6 blanks) indicates comment lines, these can be placed anywhere in the input file. The program ignores any text following the first blank after the last numerical datum, or after the character string, in each line (thus, in the table given below, the comments in square brackets are ignored by the program). Lines with some keywords (e.g., ''SPECTR'',

''IPSFN'') can appear an arbitrary number of times, limited only by the allocated amount of memory.

The program assigns default values to many input variables; lines that declare default values may be eliminated from the input file.

The structure of the input file is the following,

```
....+....1....+....2....+....3....+....4....+....5....+....6....+....
TITLE  Title of the job, up to 120 characters.

       >>>>>>>> Source definition.
SKPAR  KPARP    [Primary particles: 1=electron, 2=photon, 3=positron]
SENERG SE0              [Initial energy (monoenergetic sources only)]
SPECTR Ei,Pi                 [E bin: lower-end and total probability]
SPOSIT SX0,SY0,SZ0                        [Coordinates of the source]
SDIREC STHETA,SPHI               [Beam axis direction angles, in deg]
SAPERT SALPHA                              [Beam aperture, in deg]

       >>>>>>>> Input phase-space file (psf).
IPSFN  psf_filename.ext              [Input psf name, 20 characters]
IPSPLI NSPLIT                                    [Splitting number]
EPMAX  EPMAX                [Maximum energy of particles in the psf]

       >>>>>>>> Material data and simulation parameters.
NMAT   NMAT                   [Number of different materials, .le.10]
SIMPAR M,EABS(1:3,M),C1,C2,WCC,WCR   [Sim. parameters for material M]
PFNAME mat_filename.ext          [Material definition file, 20 chars]

       >>>>>>>> Geometry definition file.
GEOMFN geo_filename.ext          [Geometry definition file, 20 chars]
DSMAX  IBODY,DSMAX(IBODY)   [IB, maximum step length (cm) in body IB]

       >>>>>>>> Interaction forcing.
IFORCE KB,KPAR,ICOL,FORCER,WLOW,WHIG            [Interaction forcing]

       >>>>>>>> Emerging particles. Energy and angular distributions.
NBE    EMIN,EMAX,NBE              [E-interval and no. of energy bins]
NBTH   NBTH                  [No. of bins for the polar angle THETA]
NBPH   NBPH                [No. of bins for the azimuthal angle PHI]

       >>>>>>>> Impact detectors (up to 5 different detectors).
IMPDET EDIL,EDIU,NCHI,IPSF  [Energy window, no. of channels and IPSF]
IDPSF  pm_psf_impdet_#.dat           [Output psf file name, 20 chars]
IDSPC  pm_spc_impdet_#.dat      [Output spectrum file name, 20 chars]
IDBODY KB                       [Active body; one line for each body]
IDKPAR KPAR               [Kind of detected particles, one line each]

       >>>>>>>> Energy-deposition detectors (up to 5).
ENDDET EDEL,EDEU,NCHE          [Energy window and number of channels]
EDSPC  pm_spc_enddet_#.dat      [Output spectrum file name, 20 chars]
EDBODY KB                       [Active body; one line for each body]

       >>>>>>>> Dose distribution.
GRIDX  XL,XU                [X coordinates of the enclosure vertices]
GRIDY  YL,YU                [Y coordinates of the enclosure vertices]
GRIDZ  ZL,ZU                [Z coordinates of the enclosure vertices]
GRIDBN NX,NY,NZ                                    [Numbers of bins]

       >>>>>>>> Job properties.
RESUME dumpfile_1.dat          [Resume from this dump file, 20 chars]
DUMPTO dumpfile_2.dat             [Generate this dump file, 20 chars]
DUMPP  DUMPP                                 [Dumping period, in sec]
```

```
NSIMSH DSHN                      [Desired number of simulated showers]
RSEED  ISEED1,ISEED2           [Seeds of the random number generator]
TIME   TIMEA                        [Allotted simulation time, in sec]
....+....1....+....2....+....3....+....4....+....5....+....6....+....
```

The following listing describes the function of each of the keywords, the accompanying data and their default values. For clarity, blanks in keywords are indicated as '_'.

```
TITLE_ : Title of the job (an alphanumeric string with up to 120
         characters).
           DEFAULT: none (the input file must start with this line)

         The TITLE string is used to mark dump files. To prevent the
         improper use of wrong resuming files, change the title each
         time you modify basic parameters of your problem. The code
         will then be able to identify the inconsistency and to print
         an error message before stopping.

>>>>>>>> Source definition.

SKPAR_ : Kind of primary particle (1=electrons, 2=photons or
         3=positrons).
           DEFAULT: KPARP=1

SENERG : For a monoenergetic source, initial energy SE0 of primary
         particles.
           DEFAULT: SE0=1.0E6

SPECTR : For a source with continuous (stepwise constant) spectrum,
         each 'SPECTR' line gives the lower end-point of an energy
         bin of the source spectrum (Ei) and the associated relative
         probability (Pi), integrated over the bin. Up to NSEM=200
         lines, in arbitrary order. The upper end of the spectrum is
         defined by entering a line with Ei equal to the upper energy
         value and with Pi set to a negative value.
           DEFAULT: none

SPOSIT : Coordinates of the (point) source.
           DEFAULT: SX0=SY0=0, SZ0=-1.0E15
SDIREC : Polar and azimuthal angles of the source beam axis direction,
         in deg.
           DEFAULTS: STHETA=0.0, SPHI=0.0
SAPERT : Angular aperture of the source beam, in deg.
           DEFAULT: SALPHA=0.0

         --> Notice that the default source is a pencil beam that
         moves upwards along the Z-axis.

>>>>>>>> Input phase-space file.
```

The initial state variables of primary particles can be read directly from a set of pre-calculated phase-space files (psf). When this option is active, previous definitions about the source are ignored.

```
IPSFN_ : Name of an input psf (up to 20 characters).
           DEFAULT: none
         Up to 100 psf's may be declared. They are read sequentially.
```

The input psf is in ASCII format. Each line defines the initial state of a particle; it contains the following quantities in free format (and in the order they are listed here):

```
  -- KPAR, type of particle (1, electron; 2, photon; 3, positron).
  -- E, energy (eV).
  -- X,Y,Z, position coordinates (cm).
  -- U,V,W, direction cosines.
  -- WGHT, weight.
  -- ILB(1),ILB(2),ILB(3),ILB(4), a set of indices that provide
        information on how the particle was generated (see above).
  -- NSHI, incremental shower number (difference between the shower
        numbers of the present particle and the one preceding it
        in the psf).
Phase-space files can be generated by running PENMAIN using an impact
detector with the flag IPSF=1 or -1 (see below).

   Because of the limited size of the psf's, the results of analogue
simulations tend to be ''too noisy''. This can be partially corrected by
splitting the particles from the psf.

IPSPLI : Splitting number. Each particle in the psf's will be split
         into NSPLIT equivalent particles, with weights equal to
         WGHT/NSPLIT.
           DEFAULT: NSPLIT=1 (no splitting)

--> WARNING: Notice that there is a ''latent'' uncertainty in the psf,
which sets a limit to the accuracy that can be attained by using large
splitting numbers.

EPMAX_ : Maximum energy (in eV) of particles in the psf's.
         EPMAX is the upper limit of the energy interval covered by
         the simulation lookup tables. To minimise interpolation
         errors, EPMAX should not be much larger than the maximum
         energy actually occurring during the simulation.

         When the initial state variables of particles are read from
         a psf, this parameter is required to initialise PENELOPE and
         is critical; the code crashes if it finds a particle that
         has energy larger than EPMAX.
           DEFAULT: EPMAX=1.0E9 (interpolation is not optimal)

>>>>>>>> Material data and simulation parameters.

NMAT__ : Number of different materials. The original programs in the
         distribution package allow up to 10 materials. This number
         can be increased by changing the value of the parameter
         MAXMAT in the original source files.
         Materials are identified by their ordering in PENELOPE's
         input material data file.
           DEFAULT: NMAT=1

SIMPAR : Set of simulation parameters for material M; absorption
         energies, EABS(1:3,M), elastic scattering parameters, C1(M)
         and C2(M), and cutoff energy losses for inelastic collisions
         and bremsstrahlung emission, WCC(M) and WCR(M). One line for
         each material.
           DEFAULTS: EABS(1,M)=EABS(3,M)=0.01*EPMAX,
                     EABS(2,M)=0.001*EPMAX
                     C1(M)=C2(M)=0.1, WCC=EABS(1,M), WCR=EABS(2,M)
           EPMAX is the upper limit of the energy interval covered
           by the simulation lookup tables.

PFNAME : Name of the PENELOPE input material data file (up to 20
         characters).
           DEFAULT: none

>>>>>>>> Geometry.
```

GEOMFN : PENGEOM geometry definition file name (a string of up to 20 characters).
DEFAULT: none.

--> The geometry definition file can be debugged/visualised with the viewers GVIEW2D and GVIEW3D (operable only under Windows).

DSMAX_ : Maximum step length DSMAX(IB) (in cm) of electrons and positrons in body IB. This parameter is important only for thin bodies; it should be given a value of the order of one tenth of the body thickness or less.
DEFAULT: DSMAX=1.0E20 (no step length control)

>>>>>>>> Interaction forcing.

IFORCE : Activates forcing of interactions of type ICOL of particles KPAR in body KB. FORCER is the forcing factor, which must be larger than unity. WLOW and WHIG are the lower and upper limits of the weight window where interaction forcing is applied.
DEFAULT: no interaction forcing

If the mean free path for real interactions of type ICOL is MFP, the program will simulate interactions of this type (real or forced) with an effective mean free path equal to MFP/FORCER.

TRICK: a negative input value of FORCER, -FN, is assumed to mean that each particle should interact, on average and approximately, +FN times in a path length equal to the range of that kind of particle with E=EPMAX. This is very useful to generate x-ray spectra from bulk samples.

The real effect of interaction forcing on the efficiency is not easy to predict. Please, do tentative runs with different FORCER values and check the efficiency gain (or loss!).

>>>>>>>> Energy and angular distributions of emerging particles.

NBE___ : Limits EMIN and EMAX of the interval where the energy distributions of emerging particles are tallied. Number of energy bins.
DEFAULT: EMIN=0.0, EMAX=EPMAX, NBE=100

NBTH__ : Number of bins for the polar angle THETA.
DEFAULT: NBTH=90

WARNING: In the output files, the terms ''transmitted'' and ''backscattered'' are used to denote particles that leave the material system moving upwards (THETA>0) and downwards (THETA<0), respectively. Notice that this agrees with the usual meaning of these terms only when primary particles impinge on the system coming from below (i.e. with THETA>0).

NBPH__ : Number of bins for the azimuthal angle PHI.
DEFAULT: NBPH=1 (azimuthal average).

>>>>>>>> Impact detectors.

Each impact detector consists of a set of non-void active bodies, which must have been defined as parts of the geometry. The output spectrum is the energy distribution of particles that entered any of the

active bodies coming from a body that is not active (i.e. that is not part of the detector). Notice that a detected particle can re-enter the detector volume and, consequently, be ''counted'' several times (except when the flag IPSF is set equal to -1, see below).

Active bodies cannot be void, because the geometry routines would not stop particles at their limiting surfaces. In case you need to define detectors outside the material system, fill them with an arbitrary material of very small density to avoid perturbing the transport process.

To define each impact detector insert the following block of lines;

IMPDET : Starts the definition of a new detector. Up to 5 different detectors can be considered.
EDIL and EDIU are the lower and upper limits of the energy window covered by the impact detector.
NCHI is the number of channels in the output spectrum of the detector (.LE. 1000).
The integer flag IPSF serves to activate the creation of a phase-space file (psf), which contains the state variables of all particles that enter the detector. Use this option with care, because psf's may grow very fast.
IPSF=0, the psf is not created.
IPSF=1, a psf is created. Particles that enter the detector are transported as usually.
IPSF=-1, a psf is created. The simulation of a detected particle is discontinued when it enters the detector.
DEFAULTS: None

IDPSF_ : Name of the output phase-space file (up to 20 characters).
DEFAULT: ''pm_psf_impdet_#.dat''

IDSPC_ : Name of the output energy spectrum file (20 characters).
DEFAULT: ''pm_spc_impdet_#.dat''

IDBODY : Active body of the detector. One line for each active body.
DEFAULT: None
--> Notice that a body cannot be part of more than one impact detector.

IDKPAR : Kind of particle that is detected (1=electrons, 2=photons or 3=positrons). One line for each kind.
DEFAULT: All particles are detected

>>>>>>>> Energy-deposition detectors.

Each energy-deposition detector consists of a set of active bodies, which must have been defined as parts of the geometry. The output spectrum is the distribution of absorbed energy (per primary shower) in the active bodies.

***** WARNING: The energy-deposition spectrum may be strongly biased when interaction forcing is applied.**

To define each energy-deposition detector insert the following block of lines;

ENDDET : Starts the definition of a new energy-deposition detector. Up to 5 different detectors can be considered.
EDEL and EDEU are the lower and upper limits of the energy window covered by the detector.
NCHE is the number of energy channels in the output spectrum (.LE. 1000).

EDSPC_ : Name of the output spectrum file (up to 20 characters).
DEFAULT: ''pm_spc_enddet_#.dat''

EDBODY : Active body of the detector. One line for each active body.
DEFAULT: None
--> Notice that a body cannot be part of more than one energy-deposition detector.

>>>>>>>> Dose map.

The program can calculate the dose distribution inside a parallelepiped (dose enclosure) whose edges are parallel to the axes of the laboratory frame. The enclosure is defined by giving the coordinates of its vertices. The dose is tallied using a uniform orthogonal grid with NX, NY and NZ (.LE. 100) bins along the directions of the coordinate axes.

GRIDX_ : X-coordinates of the vertices of the dose enclosure.
DEFAULT: None
GRIDY_ : Y-coordinates of the vertices of the dose enclosure.
DEFAULT: None
GRIDZ_ : Z-coordinates of the vertices of the dose enclosure.
DEFAULT: None
GRIDBN : Numbers of bins NX, NY, and NZ in the X, Y and Z directions, respectively.
DEFAULTS: NX=1, NY=1, NZ=1

--> The grid defined here to calculate the dose distribution can be used to tally other 3D distributions (e.g. the space distribution of inner-shell ionisation events, used in electron-probe microanalysis). This, however, requires to edit the PENMAIN.F source file.

>>>>>>>> Job properties.

RESUME : The program will read the dump file named `dumpfile_1.dat' (20 characters) and resume the simulation from the point where it was left. Use this option very, _VERY_ carefully. Make sure that the input data file is fully consistent with the one used to generate the dump file.
DEFAULT: off

DUMPTO : Generate a dump file named ''dumpfile_2.ext'' (20 characters) after completing the simulation run. This allows resuming the simulation later on to improve statistics.
DEFAULT: off

NOTE: If the file ''dumpfile_2.ext'' already exists, it is overwritten.

DUMPP_ : When the DUMPTO option is activated, simulation results are evaluated and written in the output files every DUMPP seconds. This option is useful to check the progress of long simulations. It also allows running the program with a long execution time and stopping it when the required statistical uncertainty has been reached.
DEFAULT: DUMPP=1.0E15

NSIMSH : Desired number of simulated showers.
DEFAULT: DSHN=2.0E9

RSEED_ : Seeds of the random number generator.
DEFAULT: ISEED1=12345; ISEED2=54321

TIME__ : Allotted simulation time, in sec.
DEFAULT: TIMEA=100.0E0

The program is aborted when an incorrect input datum is found. The conflicting quantity usually appears in the last line of the output file ''penmain.dat''. If the trouble is with arrays having dimensions smaller than required, the program indicates how the problem can be solved (this usually requires editing the source file, be careful).

********** Generating the executable PENMAIN and running it.

To generate the executable binary file PENMAIN.EXE, compile and link the FORTRAN 77 source files PENMAIN.F, PENELOPE.F, PENGEOM.F, PENVARED.F and TIMER.F. For example, if you are using the g77 compiler under Windows, place these five files in the same directory, open a DOS window and from that directory enter the command
''g77 -Wall -O PENMAIN.F -o PENMAIN.EXE''
(The same, with filenames in lowercase, should work under Linux).

To run PENMAIN, you have to generate an input data file, let's call it PENMAIN.IN, and the corresponding geometry definition and material data files. Place these three files and the binary file PEMAIN.EXE in the same directory and, from there, issue the command
''PENMAIN.EXE < PENMAIN.IN''

The calculated distributions are written in separate files, whose names start with ''pm_'' (for PenMain) and have the extension ''.dat''. These files are in a format suited for direct visualisation with GNUPLOT (version 4.0).

---- APPENDIX 2 (File PDEFLIST.P05) ----

Materials included in the PDCOMPOS.P05 file, with identifying numbers.
(adapted from Berger, NISTIR 4999, 1992).

*** ELEMENTS (id. number = atomic number):

1	Hydrogen	51	Antimony
2	Helium	52	Tellurium
3	Lithium	53	Iodine
4	Beryllium	54	Xenon
5	Boron	55	Cesium
6	Amorphous carbon	56	Barium
7	Nitrogen	57	Lanthanum
8	Oxygen	58	Cerium
9	Fluorine	59	Praseodymium
10	Neon	60	Neodymium
11	Sodium	61	Promethium
12	Magnesium	62	Samarium
13	Aluminum	63	Europium
14	Silicon	64	Gadolinium
15	Phosphorus	65	Terbium
16	Sulfur	66	Dysprosium
17	Chlorine	67	Holmium
18	Argon	68	Erbium
19	Potassium	69	Thulium
20	Calcium	70	Ytterbium
21	Scandium	71	Lutetium
22	Titanium	72	Hafnium
23	Vanadium	73	Tantalum
24	Chromium	74	Tungsten
25	Manganese	75	Rhenium
26	Iron	76	Osmium

27	Cobalt	77	Iridium
28	Nickel	78	Platinum
29	Copper	79	Gold
30	Zinc	80	Mercury
31	Gallium	81	Thallium
32	Germanium	82	Lead
33	Arsenic	83	Bismuth
34	Selenium	84	Polonium
35	Bromine	85	Astatine
36	Krypton	86	Radon
37	Rubidium	87	Francium
38	Strontium	88	Radium
39	Yttrium	89	Actinium
40	Zirconium	90	Thorium
41	Niobium	91	Protactinium
42	Molybdenum	92	Uranium
43	Technetium	93	Neptunium
44	Ruthenium	94	Plutonium
45	Rhodium	95	Americium
46	Palladium	96	Curium
47	Silver	97	Berkelium
48	Cadmium	98	Californium
49	Indium	99	Einsteinium
50	Tin		

*** COMPOUNDS AND MIXTURES (in alphabetical order):

100 Acetone
101 Acetylene
102 Adenine
103 Adipose tissue (ICRP)
104 Air, dry (near sea level)
105 Alanine
106 Aluminum oxide
107 Amber
108 Ammonia
109 Aniline
110 Anthracene
111 B-100 bone-equivalent plastic
112 Bakelite
113 Barium fluoride
114 Barium sulfate
115 Benzene
116 Beryllium oxide
117 Bismuth germanium oxide
118 Blood (ICRP)
119 Bone, compact (ICRU)
120 Bone, cortical (ICRP)
121 Boron carbide
122 Boron oxide
123 Brain (ICRP)
124 Butane
125 N-butyl alcohol
126 C-552 air-equivalent plastic
127 Cadmium telluride
128 Cadmium tungstate
129 Calcium carbonate
130 Calcium fluoride
131 Calcium oxide
132 Calcium sulfate
133 Calcium tungstate
134 Carbon dioxide
135 Carbon tetrachloride
136 Cellulose acetate, cellophane
137 Cellulose acetate butyrate

138 Cellulose nitrate
139 Ceric sulfate dosimeter solution
140 Cesium fluoride
141 Cesium iodide
142 Chlorobenzene
143 Chloroform
144 Concrete, portland
145 Cyclohexane
146 1,2-dichlorobenzene
147 Dichlorodiethyl ether
148 1,2-dichloroethane
149 Diethyl ether
150 N,n-dimethyl formamide
151 Dimethyl sulfoxide
152 Ethane
153 Ethyl alcohol
154 Ethyl cellulose
155 Ethylene
156 Eye lens (ICRP)
157 Ferric oxide
158 Ferroboride
159 Ferrous oxide
160 Ferrous sulfate dosimeter solution
161 Freon-12
162 Freon-12b2
163 Freon-13
164 Freon-13b1
165 Freon-13i1
166 Gadolinium oxysulfide
167 Gallium arsenide
168 Gel in photographic emulsion
169 Pyrex glass
170 Glass, lead
171 Glass, plate
172 Glucose
173 Glutamine
174 Glycerol
175 Graphite
176 Guanine
177 Gypsum, plaster of Paris
178 N-heptane
179 N-hexane
180 Kapton polyimide film
181 Lanthanum oxybromide
182 Lanthanum oxysulfide
183 Lead oxide
184 Lithium amide
185 Lithium carbonate
186 Lithium fluoride
187 Lithium hydride
188 Lithium iodide
189 Lithium oxide
190 Lithium tetraborate
191 Lung (ICRP)
192 M3 wax
193 Magnesium carbonate
194 Magnesium fluoride
195 Magnesium oxide
196 Magnesium tetraborate
197 Mercuric iodide
198 Methane
199 Methanol
200 Mix d wax
201 Ms20 tissue substitute

202 Muscle, skeletal (ICRP)
203 Muscle, striated (ICRU)
204 Muscle-equivalent liquid, with sucrose
205 Muscle-equivalent liquid, without sucrose
206 Naphthalene
207 Nitrobenzene
208 Nitrous oxide
209 Nylon, du Pont elvamide 8062
210 Nylon, type 6 and type 6/6
211 Nylon, type 6/10
212 Nylon, type 11 (rilsan)
213 Octane, liquid
214 Paraffin wax
215 N-pentane
216 Photographic emulsion
217 Plastic scintillator (vinyltoluene based)
218 Plutonium dioxide
219 Polyacrylonitrile
220 Polycarbonate (makrolon, lexan)
221 Polychlorostyrene
222 Polyethylene
223 Polyethylene terephthalate (mylar)
224 Polymethyl methacrilate (lucite, perspex, plexiglass)
225 Polyoxymethylene
226 Polypropylene
227 Polystyrene
228 Polytetrafluoroethylene (teflon)
229 Polytrifluorochloroethylene
230 Polyvinyl acetate
231 Polyvinyl alcohol
232 Polyvinyl butyral
233 Polyvinyl chloride
234 Polyvinylidene chloride (saran)
235 Polyvinylidene fluoride
236 Polyvinyl pyrrolidone
237 Potassium iodide
238 Potassium oxide
239 Propane
240 Propane, liquid
241 N-propyl alcohol
242 Pyridine
243 Rubber, butyl
244 Rubber, natural
245 Rubber, neoprene
246 Silicon dioxide
247 Silver bromide
248 Silver chloride
249 Silver halides in photographic emulsion
250 Silver iodide
251 Skin (ICRP)
252 Sodium carbonate
253 Sodium iodide
254 Sodium monoxide
255 Sodium nitrate
256 Stilbene
257 Sucrose
258 Terphenyl
259 Testes (ICRP)
260 Tetrachloroethylene
261 Thallium chloride
262 Tissue, soft (ICRP)
263 Tissue, soft (ICRU four-component)
264 Tissue-equivalent gas (methane based)
265 Tissue-equivalent gas (propane based)

```
 266  Tissue-equivalent plastic (A-150)
 267  Titanium dioxide
 268  Toluene
 269  Trichloroethylene
 270  Triethyl phosphate
 271  Tungsten hexafluoride
 272  Uranium dicarbide
 273  Uranium monocarbide
 274  Uranium oxide
 275  Urea
 276  Valine
 277  Viton fluoroelastomer
 278  Water, liquid
 279  Water vapor
 280  Xylene


 *** END ***


                          XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
                          X  Please report any bugs to F. Salvat,      X
                          X      e-mail: cesc@ecm.ub.es                X
                          X      Tel: 34-934021186, Fax: 34-934021174  X
                          XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C  PENELOPE/PENGEOM (version 2005)                                     C
C  Copyright (c) 2001-2005                                             C
C  Universitat de Barcelona                                            C
C                                                                      C
C  Permission to use, copy, modify, distribute and sell this software  C
C  and its documentation for any purpose is hereby granted without     C
C  fee, provided that the above copyright notice appears in all        C
C  copies and that both that copyright notice and this permission      C
C  notice appear in all supporting documentation. The Universitat de   C
C  Barcelona makes no representations about the suitability of this    C
C  software for any purpose. It is provided "as is" without express    C
C  or implied warranty.                                                C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```