```
**********************************************************************
**********************************************************************
**                                                                  **
**                                              November 1989       **
**                      A Manual to                                 **
**                                                                  **
**   The Lund Monte Carlo for Jet Fragmentation and e+e- Physics    **
**                                                                  **
**                    JETSET version 7.2                            **
**                                                                  **
**                    Torbjorn Sjostrand                            **
**                                                                  **
**               CERN/TH, CH-1211 Geneva 23                         **
**            BITNET/EARN address TORSJO@CERNVM                     **
**                 Tel. +22 - 767 28 20                             **
**                                                                  **
**        LUSHOW is written together with Mats Bengtsson            **
**                                                                  **
**              Copyright Torbjorn Sjostrand                        **
**                                                                  **
**********************************************************************
**********************************************************************
*                                                                  *
*         Table of Contents                                        *
*                                                                  *
```

```
*        References                                                    *
*                                                                      *
************************************************************************
*                                                                      *
* Legend:                                                              *
* >=       larger than or equal to                                     *
* <=       smaller than or equal to                                    *
* /=       not equal to                                                *
* ->       goes to (rightarrow)                                        *
* ^        what follows next is to be read as an upper index           *
* _        what follows next is to be read as a lower index            *
* (D=..)   default value for commonblock parameter                     *
* (R)      commonblock variable which user may read but not change      *
* (I)      commonblock variable for purely internal use                *
*                                                                      *
************************************************************************
```

1.   Introductory Material

This first section contains the objective of this program, background
information and installation notes. The experienced user, at a place
where the program is already installed, can probably go directly to
section 2 without too much of a loss.

_____

1.1. Program Objective

Jet fragmentation, in its broadest sense, occurs anytime a high energy
physics event involves the production of several hadrons. Indeed, most
of the experiments, present or planned, at high energy physics
laboratories like CERN, DESY, Fermilab, SLAC, Cornell, KEK or Serpukhov
involve jet studies, either for their own sake or as a prerequisite
for the "real" physics studies. The theory for strong interactions,
QCD, in principle should give all the properties of jet fragmentation.
The problem is only that nobody knows how to solve QCD, not even for
questions orders of magnitude easier than the real-life situation of
tens or hundreds of particles produced in one single event, each
particle with three momentum degrees of freedom, plus additional
flavour and spin quantum numbers.

The natural way out has been the introduction of phenomenological models
for jet fragmentation, implemented in terms of computer programs that
generate complete events, which can be directly compared with
experimental data. For subprocesses involving high momentum transfers
Q^2, perturbation theory can be used to give the leading order
behaviour. The generation of an event can therefore be subdivided into
two steps. In the first step, a parton configuration is selected,
using perturbative standard model (electroweak and QCD) results.
In the second step, these partons are then allowed to
fragment into hadrons, with unstable hadrons decaying
further. All elements specific to the process under study are
supposedly contained in the first step, whereas the fragmentation is
assumed to occur according to rules independent of the primary
process, "jet universality". Because of quantum mechanical effects,
neither of the two steps is of a deterministic nature. A correct

treatment would utilize amplitudes and thus contain interference terms;
since these terms are unknown anyhow, at least for fragmentation, it is
natural to choose a probabilistic approach. This approach readily lends
itself to an implementation in terms of Monte Carlo computer programs.
Ideally the events generated with these programs should not only
give the same mean behaviour as experimentally observed events, but
also contain the same degree of event-by-event fluctuations.

The first part of the present program contains the process-independent
fragmentation and decay routines, whereas the second part deals
specifically with the production of the initial parton configuration
in e+e- annihilation events. Hard interactions are also covered in
the PYTHIA program [Ben87]. A new version (5.3) of this program, fully
compatible with JETSET 7.2 conventions, is now available. Indeed,
some of the new features in JETSET have been included specifically
to simplify this compatibility. PYTHIA has traditionally been a
program for hadron-hadron collisions, but the new version also
contains a number of hard processes in lepton-lepton collisions. In
the future, the e+e- annihilation routines in JETSET will be moved
to PYTHIA.

Other programs have been built around JETSET 6.3. These include:
- the leptoproduction program LEPTO [Ing80];
- the hadron-hadron, hadron-nucleus and nucleus-nucleus program
  FRITIOF [Nil87];
- the dipole radiation program ARIADNE [Pet88] (now also available
  for JETSET 7);
- the higher twist program TWISTER [Ing87];
- the photoproduction program LUCIFR [Ing87a]; and
- the leptoproduction heavy flavour program AROMA [Ing88].

The Lund Monte Carlo is therefore a useful tool for the exploration of
jet universality questions, in that experimentalists can directly
compare e.g. flavour production parameters determined from different
fields of high energy physics.

A program like this may be used for different purposes. In one extreme,
it may offer a guideline for reasonable "conventional" physics when
planning a detector design, or help in estimating acceptance corrections
for a given detector geometry. In the other extreme, Monte Carlo results
may be compared with existing data in order to extract and study
interesting physics. The user probably wants to obtain a "best estimate"
without too much ado in the former case, whereas he/she may want to
play around with different schemes and possibilities in the latter case,
often even studying "crackpot" alternatives just to build up the
physical intuition.

Indeed, the inclusion of a specific option in the program
does not in any way imply a belief that this option is physically
sound, only that it may be useful to have around. A prime example is
independent fragmentation, a concept we consider false and which is now
strongly counterindicated by data. In order to find genuine, nontrivial
predictions of our own string fragmentation scheme, independent
fragmentation still offers a good reference.

A word of warning may therefore be in place. The program description is
fairly lengthy, and certainly could not be absorbed in one sitting.
This is not even necessary, since all switches and parameters are
provided with sensible default values, based on our best understanding
of jet fragmentation. A new user can therefore disregard
all the fancy options, and just run the program as it is, to gain some
experience. Later on, the options that might seem useful can be tried
out. No single user is ever likely to find need for more than a fraction
of the total number of switches and parameters available, yet many of
them have been added to meet special user requests.

At all times, however, it should be remembered that this program
represents a model, not a theory. Some parameter values have been
determined from experimental data, others are "informed guesses". Some
parts of the model/program seem to be supported by the data, whereas the
status of others is more uncertain. A lot of ground has to be covered,
and the models used may therefore not be equally well developed for
every aspect. There are many areas that have not even been touched
upon in the present program, like spin and polarization phenomena.
Furthermore, surprises should always be expected when performing a
new experiment, why else do it?
_____

1.2. Update History

The Lund Monte Carlo is by now a fairly old and well established
program, but has still been steadily improved on. This is a continuous
process, with the official numbered versions little more than snapshots
of this process. For the record, we below list the official versions,
with some brief notes.

```
no   date   publ.    main new or improved features
1    Nov78  [Sjo78]  single quark jets
2    May79  [Sjo79]  heavy flavour jets
3.1  Aug79   ---     two-jets in e+e-, preliminary three-jets
3.2  Apr80  [Sjo80]  three-jets in e+e- with full matrix elements,
                     toponium -> 3g decays
3.3  Aug80   ---     softer fragmentation spectrum
4.1  Apr81   ---     baryon production and diquark fragmentation,
                     fourth generation quarks, larger jet systems
4.2  Nov81   ---     low-p_T physics
4.3  Mar82  [Sjo82]  four-jets and QFD structure in e+e-,
     Jul82  [Sjo83]  event analysis routines
5.1  Apr83   ---     improved string fragmentation scheme, symmetric
                     fragmentation, full 2^nd order QCD for e+e-
5.2  Nov83   ---     momentum conservation schemes for IF,
                     initial state photon radiation in e+e-
5.3  May84   ---     "popcorn" model for baryon production
6.1  Jan85   ---     commonblocks restructured, parton showers
6.2  Oct85  [Sjo86]  error detection
6.3  Oct86  [Sjo87]  new parton shower scheme
7.1  Feb89   ---     new particle codes and commonblock structure,
                     more mesons, improved decays, vertex information,
                     Abelian gluon model, Bose-Einstein effects
7.2  Nov89  [this]   interface to new standard commonblock
```

All versions preceding 5.1 should be considered completely obsolete by
now, and there is little reason to use anything up to 6.2. From a
physics point of view, version 6.3 is very similar to 7.1 and 7.2.
For physics studies close to completion, there is therefore little to
be gained in switching to version 7. Any future developments will be
based on the latter, however (i.e. there will never be a version 6.4).
_____

1.3. Major Changes from JETSET 6.3

The present version, JETSET 7.2, of the Lund Monte Carlo for jet
fragmentation and e+e- physics represents a major break in continuity
from a programming point of view, with no backwards compatibility to
JETSET 6.3. The major rewritings have been prompted by the adoption of
the new particle numbering scheme developed under the aegis of the
Particle Data Group [PDG88], a scheme which is intended to become
standard in all event generation and detector simulation Monte Carlo
programs. This affects commonblock structure and content, as
well as calling sequences. In connection with these necessary changes,
a number of further improvements have been made. The physics content of
the program, however, is only modestly changed compared to JETSET 6.3.

Here is a more extensive list of major changes in JETSET 7.1. Some of
the changes affect the default operation of the program, while others
appear as new options.
- Some subroutine names and arguments have been changed.
- Switches and parameters have been regrouped for better consistency.
- New KF particle codes have been introduced and are consistently used
  in the program.
- The storing of particle status and origin in commonblock LUJETS has
  been reorganized and extended.
- Information on decay vertices for long-lived particles has been added.
- The lowest orbital angular momentum one meson multiplets (one scalar,
  two pseudovector and one tensor) have been implemented.
- All particle data has been updated in accordance with the 1988
  Review of Particle Properties [PDG88].
- Tau and charm decays are now mainly given in terms of individual decay
  channels, with measured branching ratios, where known.
- Top and fourth generation decays have been updated to include W
  propagator effects and a more flexible treatment of different mass
  hierarchies.
- Parton showers may develop from decay product quarks, e.g. in
  heavy flavour decays.
- Pion and eta Dalitz decays are performed with the proper matrix
  elements.
- A larger a value may be used for diquark production in the Lund
  flavour dependent symmetric fragmentation function.
- Diquark fragmentation (e.g. in a baryon target remnant) is performed
  in accordance with the "popcorn" baryon pair production scheme.
- Bose-Einstein effects may be simulated according to a simple
  algorithm.
- Matrix elements for two toy models, a scalar gluon and an Abelian
  vector gluon, have been included for the testing of QCD.
- The Abelian vector gluon model is also available as an option in the

shower evolution.
- Nonisotropic azimuthal angle in shower evolution due to gluon
  polarization.
- The alpha-strong function used for matrix elements has been made
  continuous at flavour thresholds by using a Lambda value which
  depends on the number of flavours assumed.
- The LUCLUS cluster search routine can be used also with a mass
  distance measure. A new routine LUJMAS for heavy and light jet mass
  reconstruction has been added.
- A routine LUTABU has been added to provide various event analysis
  facilities: particle content, factorial moments, energy-energy
  correlation, etc.
- A random number generator package, based on the new algorithm by
  Marsaglia and Zaman [Mar87,Jam88], comes with the program.
- The old J- and I-quark fragmentation scheme for baryon jets has been
  removed. As a consequence, so has the one-string low-p_T collision
  model.
- Also a number of other obsolete or rarely used features have been
  removed, such as the simplified Montvay scheme treatment.

As may be understood from the list above, the program has increased
significantly in size. Offset against this, the last two points
represent an effort to remove obsolete material, in particular when
this may complicate the introduction of more interesting options.
Those who wish to use one of the deleted features are heartily
recommended to run JETSET 6.3 instead. The question of diquark
fragmentation has not received its final solution yet, what is
presented here is just a (reasonable) makeshift arrangement until
further studies have been made.
_____

1.4. Major Changes from JETSET 7.1

The updates from version 7.1 to 7.2 are all minor, and just about any
program that ran with version 7.1 will also work with JETSET 7.2.
Some minor modifications have been made to bring documentation in
closer agreement with the standard developed by the QCD event generators
subgroup of the 1989 LEP physics workshop [QEG89], and also to allow
for some more switches where they logically belong. Specifically,
these are the changes where compatibility problems might exist
with programs written to run with 7.1:
- The meaning of KF codes 91 - 94 has been brought in agreement with
  the agreed standard [QEG89]. To make space, the internal diquark KC
  codes were shifted from 91 and 92 to 90; this latter should not be
  visible externally.
- The codes KF = 91 - 93 now appear as part of the event history
  (this can partly be circumvented, see MSTU(16)), i.e. primary
  hadrons now point back to an entry with KF = 92 for a fragmenting
  string, KF = 93 for an independent fragmentation jet system, and
  to KF = 91 for a small parton system that collapsed into one or
  two primary hadrons. For the designation of the CM frame of a
  shower, KF = 94 works as KF = 93 used to, except that the K(I,1)
  code has been changed from 14 to 11.
- The (rarely used) MSTJ(111) and MSTJ(112) variables have been moved
  to MSTJ(115) and MSTJ(116), respectively, to make room for some

further matrix element switches.
- In qqbarq'qbar' four-jet events generated in the matrix element
  option, the secondary q' may now also be b, whereas previously
  only d, u, s and c were allowed. The parametrized qqbarq'qbar'
  rate has as a consequence been increased by a factor 5/4 (this
  neglects the presence of a term for q = q' which does not grow
  with the number of flavours, but this term is negligible anyway).
- For matrix elements, it is no longer checked that the y value is
  sufficiently large that no negative differential three-jet rates
  may appear. Instead, the rate is assumed zero in such regions,
  while it is rescaled uniformly in regions of positive cross-section
  so as to keep the total three-jet rate "correct". (This change of
  policy was required to make the implementation of optimized matrix
  elements practicable.)
- The Z, W and t default masses have been updated, together with a few
  other minor changes of particle data.
- An error in the second order corrections to the three-jet rate,
  introduced by mistake when going from 6.3 to 7.1, has been removed.
- A few other minor errors have also been corrected; none of these
  should really make a difference, however.

In addition, a number of extensions have been made, where no
backwards incompatibilities are involved.
- The Zhu parametrization of second order corrections to the three-jet
  rate has been introduced as an alternative to the old GKS one.
- It is possible to pick an "optimized" Q^2 scale for second order
  matrix elements (and the R value).
- The parton shower can also include emission of on-mass-shell photons.
  If the photons are emitted in the first step of the shower, a
  correction is made to the matrix element probability [Gro81].
- It is possible to include nonisotropic azimuthal distributions in
  shower branchings due to soft gluon interference, in addition to
  the gluon polarization effects made available in 7.1.
- The parton shower can be run in a scalar gluon mode, in addition to
  the already existing QCD and Abelian vector gluon ones.
- An interface LUHEPC exists for conversion between the internal event
  record commonblock LUJETS and the new standard HEPEVT one [QEG89],
  see section 2.11.
- A routine LUJOIN has been added to help users rapidly to set up
  colour flow information in user-defined jet systems, as required
  e.g. for subsequent shower evolution.
- It is possible to insert separators between sections in LULIST
  for improved readability. Also, the LULIST event listing gives
  information on string systems.
- A new option LUEDIT(5) has been introduced.
- Calls to ALOG have been replaced with LOG, i.e. generic function
  names are used throughout.

A number of further additions should appear in later versions of the
program. Below is given a few examples of what this could include.
- Further updates of charm and bottom decays.
- Anisotropic decays of polarized tau leptons.
- Fragmentation of strings with a junction, as obtained when the two
  quarks of an effective diquark have a significant relative transverse
  momentum.

- A retuning of fragmentation parameters, with contributions from
  the recently included meson multiplets accounted for.
- Switch to allow top and heavier quarks to decay before they hadronize.
No guarantees are given, however, neither that these will be included,
nor when it would happen.

The present text is intended to contain a complete manual for the
JETSET 7.2 user. It is not a description of the physics of the program,
however. A new user should therefore have a look in the following
literature, at the very least.
- [And83] : the standard introduction to the Lund model, unfortunately
  with a lot of details now out of date.
- [Sjo86] : the latest complete description of the physics that goes
  into the JETSET program, as well as the complete JETSET 6.2 manual.
- [Sjo87] : update notes for JETSET 6.3, and in particular a description
  of the new parton showering routine.
- [Sjo88] : a recent summary of fragmentation models (as seen from
  the Lund horizon).
- [QEG89] : the most recent review of QCD event generators, with a
  fairly extensive description of JETSET 7.1.
Some of the new aspects included in JETSET 7 are still not well
documented anywhere, unfortunately.
_____

1.5. Installation of Program

The program is written entirely in standard Fortran 77, and should
run on any machine with such a compiler. Unfortunately, experience
with IBM, VAX and Norsk Data compilers has been uniform: the options
available for obtaining optimized code actually produce erroneous
code (e.g., operations inside DO loops are moved out before them,
where some of the variables have not yet been properly set).
Therefore the general advice is to use a low optimizing level, like
OPTIMIZE(1) on IBM, /NOOPT on VAX, OPT OFF on ND, etc. Note that
this is often not the default setting.

Specifically on the Apollo, there seems to exist a problem with
non-compliance to the Fortran 77 standard for the handling of
unformatted write and backspace, as used by RLUGET and RLUSET.
This does not form a necessary part of the event generator itself,
and so RLUGET and RLUSET could well just be commented out, or
some alternative solution found.

SAVE statements have been included in accordance with the Fortran
standard. Since most ordinary machines take SAVE for granted, this
part is not particularly well tried out, however. Users on machines
without automatic SAVE are therefore warned to be on the lookout for
any variables which may have been missed.

All default settings and particle data are stored in the BLOCK DATA
LUDATA. This subprogram must be linked for a proper functioning of
the other routines. On some machines this is not done automatically
but must be forced by the user, in particular if JETSET is maintained
as a library from which routines are to be loaded only when they are
needed. In this connection we note that the library approach does not

give any significant space advantages over a loading of the whole
JETSET package as a unit, since a normal run will call on most of the
routines anyway, directly or indirectly.

Since most machines in current use are 32 bit ones, this is the
precision normally assumed. A few pieces of code have therefore had
to be written in double precision. All double precision variables
have as first character D, and this character is never used for
single precision variables. Therefore the declaration is done with
IMPLICIT DOUBLE PRECISION(D). The only double precision constants
that appear are 0D0, 1D0 and 2D0. A user on a 64 bit machine could
therefore easily transform the program to single precision throughout.
He (she) is then strongly urged to mark this version clearly, and see
to it that this version is not handed on to a 32 bit machine user,
who could find him(her)self in deep trouble.

For applications at very high energies, like SSC, the use of single
precision for any real variable starts to become a problem. It might
then be necessary to rewrite the program completely, i.e. extend the
range of the declaration to IMPLICIT DOUBLE PRECISION(A-H,O-Z), and
change all real constants to double precision. Needless to say, the
latter is a major undertaking. In some cases, shortcuts are available.
On the IBM, e.g., the AUTODBL compiler option for automatic
precision doubling works fine, provided only that an extra dummy
integer variable (NPAD, say) is inserted directly after N in the
LUJETS commonblock, to make the number of integers even. Some
pieces of code will then actually run in quadruple precision; this
could be corrected as described above.

A test program, LUTEST, is included in the JETSET package. It is
disguised as a subroutine, so that the user has to run a main program
      CALL LUTEST(1)
      END
This program will generate six hundred events of different types,
under a variety of conditions. If the program has not been properly
installed, this program is likely to crash, or at least generate a
number of erroneous events. This will then clearly be marked in the
output, which otherwise will just contain a few sample event listings
and a table of the number of different particles produced. To switch
off the output of normal events and final table, use LUTEST(0) instead
of LUTEST(1). The final tally of errors detected should read 0.
_____

1.6. Random Numbers

The construction of a good, portable (pseudo)random generator is not a
trivial task. Therefore the Lund Monte Carlo has traditionally stayed
away from that area, and just provided the routine RLU as an interface,
which the user could modify to call on an existing routine, implemented
on the actual machine being used.

In recent years, progress has been made in constructing portable
generators with large periods and other good properties; see the review
[Jam88]. Therefore the current version contains a random number

generator based on the algorithm proposed by Marsaglia and Zaman [Mar87]. This routine should work on any machine with an at least 24-digit mantissa, i.e. all common 32-bit (or more) computers. Given the same initial state, the sequence will also be identical on different machines. This need not mean that the same sequence of events will be generated on an IBM and a VAX, say, since the different treatments of roundoff errors in numerical operations will lead to slightly different real numbers being tested against these random numbers in IF-statements. Apart from nomenclature issues, and the coding of RLU as a function rather than a subroutine, the only difference between the JETSET code and the code given in [Jam88] is that slightly different algorithms are used to ensure that the random number is not equal to 0 or 1 within machine precision.

The generator has a period of roughly 2*10^43, and the possibility to obtain almost 10^9 different and disjoint subsequences, selected by giving an initial integer numbers. The price to be paid for the long period is that the state of the generator at a given moment can not be described by a single integer, but requires about 100 words. Some of these are real numbers, and are thus not correctly represented in decimal form. The normal procedure, with being able to restart the generation from a seed value written among the run output, is therefore not convenient. The CERN library implementation keeps track of the number of random numbers generated since the start. With this value saved, in a subsequent run the random generator can be asked to skip ahead the corresponding number of numbers. The Lund Monte Carlo is a heavy user of random numbers, however: typically 30% of the full run time is spent on random number generation. Of this, half is overhead coming from the function call administration, but the other half is truly related to the speed of the algorithm. Therefore a skipping ahead would take place with 15% the time cost of the original run, i.e. an uncomfortably high figure.

Instead a different solution is chosen here. Two special routines are provided for writing and reading the state of the random number (plus some initialization information) on a sequential file, in a machine dependent internal representation. The file used for this purpose has to be specified by the user, and opened for read and write. A state is written as a single record, in free format. It is possible to write an arbitrary number of states on a file, and a record can be overwritten, if so desired. The event generation loop might then look something like:
1) save the state of the generator on file (using flag set in point 3 below),
2) generate an event,
3) study the event for errors or other reasons why to regenerate it later; set flag to overwrite previous generator state if no errors, else set flag to create new record;
4) loop back to point 1.
In the end, the file will then contain the state before each of the problematical events. An alternative approach might be to save the state every 100 events or so. It should be emphasized that this option has not been included because errors in JETSET are likely to be frequent, but because a subsequent detector simulation of this event might fail, or for other similar reasons. (The user may then

have to save also other sets of seeds, naturally.)

In addition to the service routines, the commonblock which contains
the state of the generator is available for manipulation by the user,
if he/she so desires. In particular, the initial seed value is by
default 19780503, i.e. different from the Marsaglia/CERN default
54217137. It is possible to change this value before any random numbers
have been generated, or to force reinitialization in mid-run with any
desired new seed. Inside JETSET, or other Lund family programs, some
initialization may take place in connection with the very first event
generated in a run, so sometimes it may be necessary to generate one
ordinary event before reading in a saved state to generate an
interesting event.

It should should be noted that, of course, the appearance of a random
number generator package inside JETSET does in no way preclude a user
from removing these routines. He/she can easily revert to the old
approach, where RLU is nothing but an interface to an arbitrary
external random number generator; e.g., to call a routine RNDM
all one needs to have is
```
      FUNCTION RLU(IDUM)
  100 RLU=RNDM(IDUM)
      IF(RLU.LE.0..OR.RLU.GE.1.) GOTO 100
      RETURN
      END
```

The random generator subpackage consists of the following components.

FUNCTION RLU(IDUM)
Purpose: to generate a (pseudo)random number uniformly in the range
    0 < RLU < 1, i.e. excluding the endpoints.
IDUM : dummy input argument.

SUBROUTINE RLUGET(LFN,MOVE)
Purpose: to dump the current state of the random number generator
    on a separate file, using internal representation for real and
    integer numbers. To be precise, the full contents of the LUDATR
    commonblock are written on the file, with the exception of MRLU(6).
LFN : the file number to which the state is dumped. The user must
    associate this number with a true file (with a machine-dependent
    name), and see to it that this file is open for write.
MOVE : choice of adding a new record to the file or overwriting old
    record. Normally only options 0 or -1 should be used.
    = 0 (or > 0) : add a new record to the end of the file.
    = -1 : overwrite the last record with a new one (i.e. do one
        BACSPACE before the new write).
    = -n : back up n records before writing the new record. The
        records following after the new record are lost, i.e.
        the last n old records are lost and one new added.

SUBROUTINE RLUSET(LFN,MOVE)
Purpose: to read in a state for the random number generator, from which
    the subsequent generation can proceed. The state must previously
    have been saved by a RLUGET call. Again the full contents of the
    LUDATR commonblock are read, with the exception of MRLU(6).

LFN : the file number from which the state is read. The user must
    associate this number with a true file previously written with a
    RLUGET call, and see to it that this file is open for read.
MOVE : positioning in file before a record is read. With zero value,
    records are read one after the other for each new call, while
    nonzero values may be used to navigate back and forth, and e.g.
    return to the same initial state several times.
    = 0 : read the next record.
    = +n : skip ahead n records before reading the record that sets
        the state of the random number generator.
    = -n : back up n records before reading the record that sets the
        state of the random number generator.

COMMON/LUDATR/MRLU(6),RRLU(100)
Purpose: to contain the state of the random number generator at any
    moment (for communication between RLU, RLUGET and RLUSET), and also
    to provide the user with the possibility to initialize different
    random number sequences, and to know how many numbers have been
    generated.
MRLU(1) : (D=19780503) the integer number which specifies which of the
    possible subsequences will be initialized in the next RLU call for
    which MRLU(2)=0. Allowed values are 0 <= MRLU(1) <= 900 000 000,
    the original Marsaglia (and CERN library) seed is 54217137.
    The MRLU(1) value is not changed by any of the JETSET routines.
MRLU(2) : (D=0) initialization flag, put to 1 in the first RLU call of
    run. A reinitialization of the random number generator can be made
    in mid-run by resetting MRLU(2) to 0 by hand. In addition, anytime
    the counter MRLU(3) reaches 1000000000, it is reset to 0 and MRLU(2)
    is increased by 1.
MRLU(3) : (D=0) counter for the number of random numbers generated from
    the beginning of the run. To avoid overflow when very many numbers
    are generated, MRLU(2) is used as described above.
MRLU(4), MRLU(5) : I97 and J97 of the CERN library implementation;
    part of the state of the generator.
MRLU(6) : (D=0) current position, i.e. how many records after beginning,
    in the file used by RLUGET and RLUSET.
RRLU(1) - RRLU(97) : the U array of the CERN library implementation;
    part of the state of the generator.
RRLU(98) - RRLU(100) : C, CD and CM of the CERN library implementation;
    the first part of the state of the generator, the latter two
    constants calculated at initialization.
_____

1.7. Programming Philosophy

The Monte Carlo program is built as a slave system, i.e. the user
supplies the main program, and from this the various subroutines are
called on to execute specific tasks, after which control is returned to
the main program. Some of these tasks may be very trivial, whereas the
"high-level" routines by themselves may make a large number of
subroutine calls.

It should be noted that, while the physics content is obviously at
the center of attention, the Lund Monte Carlo also contains a more
extensive setup of auxiliary service routines than any other physics

event generator. The hope is that this will provide a comfortable
working environment, where not only events are generated, but users
also linger on to perform a lot of the subsequent studies. (As for
the relatively small attention given to physics in this manual,
the reason is that the physics is documented separately in a series
of papers, but the program pieces only here.)

The general rule is that all routines have names six characters long,
beginning with LU, except for real-valued functions, which start off
with UL instead. There are three exceptions: KLU, PLU and RLU. The
former two functions are strongly coupled to the K and P matrices in
the LUJETS commonblock, the latter uses R to emphasize the role as a
random number generator. Also commonblocks have names starting with LU.

For most of the routines no initialization is necessary, except for
the one implied by the presence of BLOCK DATA LUDATA; this subprogram
must be linked, however, which does not occur automatically with all
loaders. The cases where some initialization may indeed be performed
(depending on exactly which options are used), and hence events may have
to be generated in some coherent fashion, are LUEEVT (and some routines
called by it) and LUONIA. In addition, the random number generator is
initialized at the first call; see the preceding section.

Apart from writing a header, printing error messages if need be,
and responding to explicit requests for listings, all tasks of the
program are performed silently. All output is directed to unit MSTU(11),
by default 6, and it is up to the user to see to it that this unit is
open for write. The one exception is LUUPDA where, for obvious reasons,
the input/output file is specified at each call. Here the user again
has to see to it that proper read/write access is set.

The Lund Monte Carlo is extremely versatile, but the price to be paid
for this is a large number of adjustable parameters and switches for
alternative modes of operation. No single user is ever likely to have
need for more than a fraction of the options available.
Since all these parameters and switches are assigned sensible default
values, there is no reason to worry about them until the need arises.

Unless explicitly stated (or obvious from the context) all switches and
parameters can be changed independently of each other. One should note,
however, that if only a few switches/parameters are changed, this may
result in an artificially bad agreement with data. Many disagreements
can often be cured by a subsequent retuning of some other parameters of
the model, in particular those that have anyway once been determined by
a comparison with data in the context of the default scenario. Typically
such a retuning could involve one QCD parameter (alpha_S or Lambda),
the longitudinal fragmentation function, and the average transverse
fragmentation momentum.

The program contains a number of checks that flavours specified for jet
systems make sense, that the energy is enough to allow hadronization,
that the memory space in LUJETS is enough, etc. If anything goes wrong
that the program can catch (obviously that may not always be possible)
an error message will be printed and the treatment of the corresponding
event will be cut short. So long as no error messages appear on the

output, it may not be worth the while to look into the rules for error
checking, but if but one message appears, it should be enough cause for
alarm to receive prompt attention. Also warnings are sometimes printed.
These are less serious, and the experienced used might deliberately
do operations which goes against the rules, but still can be made to
make sense in their context. Only the first few warnings will be
printed, thereafter the program will be quite. By default, the program
is set to stop execution after ten errors, however, after printing
the last erroneous event.

It must be emphasized that not all errors will be caught. In particular,
one tricky question is what happens if an integer-valued commonblock
switch or subroutine/function argument is used with a value not
defined. In some subroutine calls, a prompt return will be expedited,
but in most instances the subsequent action is entirely unpredictable,
and often completely haywire. The same goes for real-valued input
assigned values outside the physically sensible range. One example will
here suffice: if PARJ(2) is defined as the s/u suppression factor, a
value > 1 will not give more profuse s than u production, but actually
a spillover into c production. Users, beware!

1.8. The First Steps

This section is intended as a brief guided tour through some highlights
of the Lund Monte Carlo, for persons with no previous experience
in using the Lund Monte Carlo. So, Welcome to the Lund World!,
or, Please move on!, as the case may be.

As a first example, assume that you want to study the production of
uubar two-jet systems at energy 20 GeV. To do this, write a main program
```
      CALL LU2ENT(0,2,-2,20.)
      CALL LULIST(1)
      END
```
and run this program, linked together with JETSET. The routine LU2ENT
is specifically intended for storing two entries (jets or particles).
The first argument (0) is a command to perform fragmentation and decay
directly after the entries have been stored, the second and third that
the two entries are u (2) and ubar (-2), and the last that the CM energy
of the pair is 20 GeV. When this is run, the resulting event is stored
in the LUJETS commonblock. This information can then be read out by the
user. No output is produced by LU2ENT itself, except for the title
lines
```
                The Lund Monte Carlo - JETSET version 7.2
                **  Last date of change:   8 Jan 1989  **
```
which appear once for every JETSET run.

Instead the second command, to LULIST, provides a simple visible summary
of the information stored in LUJETS. The argument (1) indicates that
the short version should be used, which is suitable for viewing the
listing directly on an 80 column terminal screen. It might look
something like

                         Event listing (summary)

```
   I  particle/jet KS    KF orig   p_x     p_y     p_z      E       m

   1  (u)       A  12     2   0   0.000   0.000  10.000  10.000   0.006
   2  (u~)      V  11    -2   0   0.000   0.000 -10.000  10.000   0.006
   3  (string)     11    92   1   0.000   0.000   0.000  20.000  20.000
   4  (rho+)       11   213   3   0.098  -0.154   2.710   2.856   0.885
   5  (rho-)       11  -213   3  -0.227   0.145   6.538   6.590   0.781
   6  pi+           1   211   3   0.125  -0.266   0.097   0.339   0.140
   7  (Sigma0)     11  3212   3  -0.254   0.034  -1.397   1.855   1.193
   8  (K*+)        11   323   3  -0.124   0.709  -2.753   2.968   0.846
   9  p~-           1 -2212   3   0.395  -0.614  -3.806   3.988   0.938
  10  pi-           1  -211   3  -0.013   0.146  -1.389   1.403   0.140
  11  pi+           1   211   4   0.109  -0.456   2.164   2.218   0.140
  12  (pi0)        11   111   4  -0.011   0.301   0.546   0.638   0.135
  13  pi-           1  -211   5   0.089   0.343   2.089   2.124   0.140
  14  (pi0)        11   111   5  -0.316  -0.197   4.449   4.467   0.135
  15  (Lambda0)    11  3122   7  -0.208   0.014  -1.403   1.804   1.116
  16  gamma         1    22   7  -0.046   0.020   0.006   0.050   0.000
  17  K+            1   321   8  -0.084   0.299  -2.139   2.217   0.494
  18  (pi0)        11   111   8  -0.040   0.410  -0.614   0.751   0.135
  19  gamma         1    22  12   0.059   0.146   0.224   0.274   0.000
  20  gamma         1    22  12  -0.070   0.155   0.322   0.364   0.000
  21  gamma         1    22  14  -0.322  -0.162   4.027   4.043   0.000
  22  gamma         1    22  14   0.006  -0.035   0.422   0.423   0.000
  23  p+            1  2212  15  -0.178   0.033  -1.343   1.649   0.938
  24  pi-           1  -211  15  -0.030  -0.018  -0.059   0.156   0.140
  25  gamma         1    22  18  -0.006   0.384  -0.585   0.699   0.000
  26  gamma         1    22  18  -0.034   0.026  -0.029   0.052   0.000
                 sum:  0.00          0.000   0.000   0.000  20.000  20.000
```

(a few blanks have been removed between the columns to make it fit into
the 72 column format of this file). Look in the particle/jet column and
note that the first two lines are the original u and ubar, where 'bar'
is actually written '~' to save space in longer names. The parantheses
enclosing these names are there as a reminder that these jets actually
have been allowed to fragment. They are retained so that event
histories can be studied. Also note that the KF (flavour code) column
contains 2 in the first line and -2 in the second. These are the
actually stored codes denoting the presence of a u and a ubar, cf.
the LU2ENT call, while the names written are just conveniences used
when producing visible output. The A and V near the end of particle/jet
column indicate the beginning and end of a string (or cluster, or
independent fragmentation) parton system; any intermediate entries
belonging to the same system would have had an I in that column.
(This gives a poor man's vertical representation of a doublesided
arrow, cf. <----->.)

In the orig (origin) column, the zeros indicate that u and ubar
are two initial entries. The subsequent line, number 3, denotes the
fragmenting u+ubar string system as a whole, and has orig 1, since the
first parton of this string system is entry number 1. The particles in
lines 4 - 10 have orig 3 to denote that they come directly from the
fragmentation of this string. In string fragmentation it is not
meaningful to say that a particle comes from only the u quark or
only the ubar one. It is the string system as a whole which gives a

rho+, a rho-, a pi+, a Sigma0, a K*+, a p~-, and a pi-. Note that some
of the particle names are again enclosed by parantheses, indicating
that these particles are also not present in the final state, but
have decayed further. Thus the pi- in line 13 and the pi0 in line 14
have orig 5, as an indication that they come from the decay of the
rho- in line 5. Only the names not enclosed in parantheses are the ones
that remain at the end of the fragmentation/decay chain, and thus are
experimentally observable. The actual status code used to distinguish
between different classes of entries is given in the KS column;
codes in the range 1 - 10 correspond to remaining entries.

The columns with p_x, p_y, p_z, E and m are rather self-explanatory.
All momenta, energies and masses are given in units of GeV, with c = 1.
Note that energy and momentum is conserved at each step of the
fragmentation/decay process (although there exist options where this is
not true). Also note that the z axis plays the role of preferred
direction, along which the original partons are put. The final line is
intended as a quick check that nothing funny happened. It contains the
summed charge, summed momentum, summed energy and invariant mass of the
final entries at the end of the fragmentation/decay chain, and the
values should agree with the input ones implied by the LU2ENT arguments.
(In fact, warnings would appear on the output if anything untoward
happened, but that is another story).

The example above has illustrated roughly what information is to be had
in the event record, but not so much about how it is stored. This is
better seen by using a 132 column format for listing events. Try e.g.
the following program

```
      CALL LU3ENT(0,1,21,-1,30.,0.9,0.7)
      CALL LULIST(2)
      CALL LUEDIT(3)
      CALL LULIST(2)
      END
```

where a 3-jet dgdbar event is generated in line 1 and listed in line 2.
This listing will contain the numbers as directly stored in the
commonblock LUJETS

```
      COMMON/LUJETS/N,K(4000,5),P(4000,5),V(4000,5)
```

For particle I, K(I,1) thus gives information on whether a jet/particle
has fragmented/decayed or not, K(I,2) gives the particle code, K(I,3)
the origin, K(I,4) and K(I,5) the position of fragmentation/decay
products, and P(I,1) - P(I,5) momentum, energy and mass. The number of
lines in current use is given by N, i.e. 1 <= I <= N. The V matrix
contains decay vertices; to view those LULIST(3) has to be used. It
is important to learn the rules for how information is stored in
LUJETS, see sections 2.1 and 2.2.

The third line in the program illustrates another important point
about JETSET: a number of routines are available for manipulating
the event record after an event has been generated. Thus LUEDIT(3)
will remove everything except stable charged particles, as shown by the
result of the second LULIST call. More advanced possibilities include
things like sphericity or clustering routines.

Apart from the input arguments of subroutine calls, control on the
doings of JETSET may be imposed via the LUDAT1, LUDAT2, LUDAT3 and

LUDAT4 commonblocks. Here sensible default values are always provided.
A user might want to switch off all particle decays by putting
MSTJ(21) = 0 or increase the s/u ratio in fragmentation by putting
PARJ(2) = 0.40, to give but two examples. It is by exploring the
possibilities here that JETSET can be turned into an extremely
versatile tool, even if all the nice physics is already present in
the default values.

As a final, semirealistic example, assume that the pT spectrum of pi+
particles is to be studied in 93 GeV e+e- annihilation events, where
pT is to be defined with respect to the sphericity axis. Using the
HBOOK package (version 4, watch out for version- or installation-
specific differences) for histogramming, a complete program might look
like

```
      COMMON/LUJETS/N,K(4000,5),P(4000,5),V(4000,5)
      COMMON/PAWC/HMEMOR(10000)
      CALL HLIMIT(10000)
      CALL HBOOK1(1,'pT spectrum of pi+',100,0.,5.,0.)
      NEVT=100
      DO 110 IEVT=1,NEVT
      CALL LUEEVT(0,93.)
      IF(IEVT.EQ.1) CALL LULIST(1)
      CALL LUSPHE(SPH,APL)
      CALL LUEDIT(31)
      DO 100 I=1,N
      IF(K(I,2).NE.211) GOTO 100
      PT=SQRT(P(I,1)**2+P(I,2)**2)
      CALL HF1(1,PT,1.)
  100 CONTINUE
  110 CONTINUE
      CALL HOPERA(1,'+',1,1,20./NEVT,0.)
      CALL HISTDO
      END
```

Study this program, try to understand what happens at each step, and
run it to check that it works. After that you should be ready to look
at the relevant sections of this manual and start writing your own
programs. Good luck!

****

2.   The Fragmentation/Decay Package

The fragmentation and decay routines form the heart of the Lund family
of Monte Carlos. The hard kinematics may be specific for e+e-
annihilation (section 3), hadron collisions (as in PYTHIA),
leptoproduction (as in LEPTO), etc., but in the end they all use the
fragmentation routines described here. In particular, the particle code
and the organization of the event record is common to all members of
the Lund family. (At the time of reading this, the switchover to the
new codes may not have taken place in all programs. In the case of
PYTHIA, the work is in progress.)

_____

2.1. Particle codes

The new particle code now adopted by the Particle Data Group [PDG88] is used consistently throughout the program, and is referred to as the KF particle code. This code the user has to be thoroughly familiar with. It is described below.

Apart from a trivial printing error for the Omega- baryon in the PDG tables, one inconsistency between the KF and the PDG codes is known. This deals with the particle chi_1c, which in KF language is 20443 while the PDG code is 10443. The code KF = 10443 instead represents the h_1c, by analogy with h_1 which is 10223 in both representations. The h_1c particle does not appear in the PDG tables (not discovered!). Since the issue is fairly peripheral anyway, no action has been taken to resolve the discrepancy.

The KF code is not convenient for a direct storing of masses, decay data, or other particle properties, however, since the KF codes are so spread out. Instead a compressed code KC between 1 and 500 is used here, where the most frequently used particles have a separate code, but many heavy flavour hadrons are lumped together in groups. Normally this code is only used at very specific places in the program, not visible to the user. If need be, the correspondence can always be obtained by using the function LUCOMP, KC = LUCOMP(KF). It is therefore not the intention that a user should ever need to know any KC code at all.

Below follows a list of KF particle codes. The list is not complete; a more extensive one may be obtained with CALL LULIST(11). Particles are grouped together, and the basic rules are described for each group. Whenever a distinct antiparticle exists, it is given the same KF code with a minus sign (whereas KC codes are always positive).

The particle names printed here also corresponds to the ones obtained with the routine LUNAME, which is used extensively, e.g. in LULIST. Greek characters are spelt out in full, with a capital first letter corresponding to a capital greek letter. Generically the name of a particle is made up out of the following pieces:
a) The basic root name. This includes a * for most spin 1 (L = 0) mesons and spin 3/2 baryons, and a ' for some spin 1/2 baryons (where there are two states to be distinguished, cf. Lambda - Sigma0). The rules for heavy baryon naming are in accordance with the 1986 Particle Data Group conventions [PDG86]. For mesons with one unit of orbital angular momentum, K (D, B, ..) is used for quark spin 0 and K* (D*, B*, ..) for quark spin 1 mesons; the convention for * may here deviate slightly from the one used by the PDG.
b) Any lower indices, separated from the root with a _. For heavy hadrons, this is the additional heavy flavour content not inherent from the root itself. For a diquark, it is the spin.
c) The character ~ (alternatively bar, see MSTU(15)) for an antiparticle, wherever the distinction between particle and antiparticle is not inherent in the charge information.
d) Charge information: ++, +, 0, -, or --. Charge is not given for quarks or diquarks. Some neutral particles which customarily are given without a 0 also here lack it, like neutrinos, g, gamma, and flavour diagonal mesons other than pi0 and rho0. Note that

charge is included both for the proton and the neutron. While
nonstandard, it is helpful in avoiding misunderstandings when
looking at an event listing.

List of KF codes:

1) Quarks and leptons.
This group contains the basic building blocks of matter, arranged
according to family, with the lower member of weak isodoublets also
having the smaller code (thus d precedes u, contrary to the ordering
in previous JETSET versions). A fourth generation is included for
future reference. The quark codes are used as building blocks for
the diquark, meson and baryon codes below.

1	d	11	e-
2	u	12	nu_e
3	s	13	mu-
4	c	14	nu_mu
5	b	15	tau-
6	t	16	nu_tau
7	l	17	chi-
8	h	18	nu_chi
9		19	
10		20	

2) Gauge bosons.
This group includes all the gauge and Higgs bosons of the standard
model, as well as some of the bosons appearing in various extensions
of the standard model. The latter are not covered by the standard
Particle Data group codes. They correspond to one extra U(1) group
and one extra SU(2) one, a further Higgs doublet, and a horizontal
gauge boson R (coupling between families).

21	gluon	31	
22	gamma	32	Z'0
23	Z0	33	Z"0
24	W+	34	W'+
25	H0	35	H'0
26		36	H"0
27		37	H+
28		38	
29		39	
30		40	R0

3) Free space
The positions 41 - 80 are currently unused. In the future, they might
come to be used e.g. for supersymmetric partners to the particles above,
or for some other kind of new physics. At the moment, they are at the
disposal of the user.

4) Various special codes.
In a Monte Carlo, it is always necessary to have codes that do not
correspond to any specific particle, but are used to lump together
groups of similar particles for decay treatment, or to specify generic
decay products. These codes, which again are non-standard, are found
between numbers 81 and 100. Several are not found in the event record,
and therefore properly belong only to the KC group of codes.

In the lines below, auxiliary information is given within parentheses.
```
    81   specflav (spectator flavour; used in decay product listings)
    82   rndmflav (a random u, d, or s flavour; possible decay product)
    83   phasespa (simple isotropic phase space decay)
    84   c-hadron (information on decay of generic charm hadron)
    85   b-hadron (information on decay of generic bottom hadron)
    86   t-hadron (information on decay of generic top hadron)
    87   l-hadron (information on decay of generic low hadron)
    88   h-hadron (information on decay of generic high hadron)
    89   Wvirt    (off-mass-shell W in weak decays of t, l, h or chi)
    90   diquark  (generic code for colour information)
    91   cluster  (parton system in cluster fragmentation)
    92   string   (parton system in string fragmentation)
    93   indep.   (parton system in independent fragmentation)
    94   CMshower (four-momentum of timelike showering system)
    95   SPHEaxis (event axis found with LUSPHE)
    96   THRUaxis (event axis found with LUTHRU)
    97   CLUSjet  (jet (cluster) found with LUCLUS)
    98   CELLjet  (jet (cluster) found with LUCELL)
    99   table    (tabular output from LUTABU)
   100
```

5) Diquark codes.
A diquark made up of a quark with code i and another with code j,
where i >= j, and with total spin s, is given the code
KF = 1000*i + 100*j + 2s+1
i.e. the tens position is left empty (cf. the baryon code below).
Some of the most frequently used codes are listed below. All the
lowest-lying spin 0 and 1 diquarks are included in the program.

```
                                  1103    dd_1
   2101    ud_0                    2103    ud_1
                                   2203    uu_1
   3101    sd_0                    3103    sd_1
   3201    su_0                    3203    su_1
                                   3303    ss_1
```

The corresponding KC code is 90, and is mainly used to store colour
charge.

6) Meson codes.
A meson made up of a quark with code i and an antiquark with
code -j, and with total spin s, is given the code
KF = (100*max(i,j) + 10*min(i,j) + 2s+1) * sign(i-j) * (-1)**max(i,j)
i.e. if the heaviest quark is a down-type one, an extra - sign enters.
This is in accordance with the particle-antiparticle distinction adopted
in the 1986 Review of Particle Properties [PDG86]. The flavour-diagonal
states are arranged in order of ascending mass. The standard rule of
having the last digit being 2s+1 is broken for the K_S0 - K_L0 system,
where it is 0, and this convention should carry over to mixed states in
the B meson system. For higher multiplets with the same spin, +-10000,
+-20000, etc., are added to provide the extra distinction needed.
Some of the most frequently used codes are given below.
The full lowest-lying pseudoscalar and vector multiplets are included
in the program.

```
   211   pi+                      213   rho+
   311   K0                       313   K*0
```

321	K+	323	K*+
411	D+	413	D*+
421	D0	423	D*0
431	D_s+	433	D*_s+
511	B0	513	B*0
521	B+	523	B*+
531	B_s0	533	B*_s0
111	pi0	113	rho0
221	eta	223	omega
331	eta'	333	phi
441	eta_c	443	J/psi
551	eta_b	553	Upsilon
661	eta_t	663	Theta
130	K_L0		
310	K_S0		

Also the lowest lying orbital angular momentum L = 1 mesons are
included : one pseudovector multiplet obtained for total quark spin 0
(L = 1, S = 0 -> J = 1) and one scalar, one pseudovector and one
tensor multiplet obtained for total quark spin 1 (L = 1, S = 1 ->
J = 0, 1 or 2). Any mixing between the two pseudovector multiplets is
not taken into account. Please note that some members of these
multiplets have still not been found, and are included here only based
on guesswork. Even for known ones, particle data (mass, width, decay
modes) is highly incomplete.

10213	b_1+	10211	a_0+
10313	K_10	10311	K*_00
10323	K_1+	10321	K*_0+
10413	D_1+	10411	D*_0+
10423	D_10	10421	D*_00
10433	D_1s+	10431	D*_0s+
10113	b_10	10111	a_00
10223	h_10	10221	f_00
10333	h'_10	10331	f'_00
10443	h_1c0	10441	chi_0c0
20213	a_1+	215	a_2+
20313	K*_10	315	K*_20
20323	K*_1+	325	K*_2+
20413	D*_1+	415	D*_2+
20423	D*_10	425	D*_20
20433	D*_1s+	435	D*_2s+
20113	a_10	115	a_20
20223	f_10	225	f_20
20333	f'_10	335	f'20
20443	chi_1c0	445	chi_2c0

The corresponding meson KC codes, used for organizing mass and decay
data, range between 101 and 240.

7) Baryon codes.
A baryon made up of quarks i, j and k, with i >= j >= k, and total
spin s, is given the code
KF = 1000*i + 100*j + 10*k + 2s+1.
An exception is provided by spin 1/2 baryons made up of three different
type quarks, where the two lightest quarks form a spin 0 diquark
(Lambda-like baryons). Here the order of the j and k quarks is
changed, so as to provide a simple means of distinction to baryons

with the lightest quarks in a spin 1 diquark (Sigma-like baryons).
For hadrons with heavy flavours, the root names are Lambda or Sigma
for hadrons with two u or d quarks, Xi for those with one and
Omega for those without u or d quarks. Some of the most frequently
used codes are given below. The full lowest-lying spin 1/2 and 3/2
multiplets are included in the program.

		1114	Delta-
2112	n	2114	Delta0
2212	p	2214	Delta+
		2224	Delta++
3112	Sigma-	3114	Sigma*-
3122	Lambda0		
3212	Sigma0	3214	Sigma*0
3222	Sigma+	3224	Sigma*+
3312	Xi-	3314	Xi*-
3322	Xi0	3324	Xi*0
		3334	Omega-
4112	Sigma_c0	4114	Sigma*_c0
4122	Lambda_c+		
4212	Sigma_c+	4214	Sigma*_c+
4222	Sigma_c++	4224	Sigma*_c++
4132	Xi_c0		
4312	Xi'_c0	4314	Xi*_c0
4232	Xi_c+		
4322	Xi'_c+	4324	Xi*_c+
4332	Omega_c0	4334	Omega*_c0
5112	Sigma_b-	5114	Sigma*_b-
5122	Lambda_b0		
5212	Sigma_b0	5214	Sigma*_b0
5222	Sigma_b+	5224	Sigma*_b+

The corresponding KC codes, used for organizing mass and decay data,
range between 301 and 400, with some slots still free.

8) Diffractive states.
These codes are not standard ones, but have been defined by analogy
to be used for denoting diffractive states in the PYTHIA program,
as part of the event history. The first two or three digits give
flavour content, while the last one is 0, to denote the somewhat
unusual character of the code. Only three codes have been introduced.
   210   pi_diffr+
  2110   n_diffr
  2210   p_diffr+

9) Free compressed codes
The positions 401 - 500 of mass and decay arrays are left open.
Here a user may map any new kind of particle from the ordinary
KF codes, which probably are above 10000, into a more manageable
KC range for mass and decay data information. The mapping must be
implemented in the LUCOMP function.

2.2. The Event Record

Each new event generated is in its entirety stored in the commonblock
LUJETS, which thus forms the event record. Here each jet or particle

that appears at some stage of the fragmentation or decay chain will occupy one line in the matrices. The different components of this line will tell which jet/particle it is, from where it originates, its present status (fragmented/decayed or not), its momentum, energy and mass, and the space-time position of its production vertex.
Note that K(I,3) - K(I,5) and the P and V vectors may take special meaning for some specific applications (e.g. sphericity or cluster analysis), as described in those connections.

COMMON/LUJETS/N,K(4000,5),P(4000,5),V(4000,5)
Purpose: to contain the event record, i.e. the complete list of all partons and particles in the current event.

N : number of lines in the K, P and V matrices occupied by the current
     event. N is continuously updated as the definition of the original
     configuration and the treatment of fragmentation and decay proceed.
     In the following, the individual parton/particle number, running
     between 1 and N, is called I.

K(I,1) : status code KS, which gives the current status of the
     parton/particle stored in the line. The ground rule is that codes
     1 - 10 correspond to currently existing partons/particles, while
     larger codes contain partons/particles which no longer exist, or
     other kinds of event information.
     = 0 : empty line.
     = 1 : an undecayed particle or an unfragmented jet, the latter
           either being a single jet or the last one of a jet system.
     = 2 : an unfragmented jet, which is followed by more jets in the
           same colour singlet jet system.
     = 3 : an unfragmented jet with special colour flow information
           stored in K(I,4) and K(I,5), such that adjacent partons along
           the string need not follow after each other in the event record.
     = 4 : a particle which could have decayed, but did not do it within
           the allowed volume around the original vertex.
     = 5 : a particle which is is to be forced to decay in the next
           LUEXEC call, in the vertex position given (this code is only
           set by user intervention).
     = 11 : a decayed particle or a fragmented jet, the latter either
            being a single jet or the last one of a jet system, cf. =1.
     = 12 : a fragmented jet, which is followed by more jets in the same
            colour singlet jet system, cf. =2.
     = 13 : a jet which has been removed when special colour flow
            information has been used to rearrange a jet system, cf. =3.
     = 14 : a parton which has branched into further partons, with
            special colour flow information provided, cf. =3.
     = 15 : a particle which has been forced to decay (by user
            intervention), cf. =5.
     = 21 : documentation lines used to give a compressed story of the
            event at the beginning of the event record.
     = 31 : lines with information on sphericity, thrust or cluster
            search.
     = 32 : tabular output, as generated by LUTABU.
     < 0 : these codes are never used by the program, and are therefore
           usually not affected by operations on the record, like LUROBO,
           LULIST and event analysis routines (the exception is some

LUEDIT calls, where lines are moved but not deleted). Such
        codes may therefore be useful in some connections.

K(I,2) : parton/particle KF code, as described in a section 2.1 above.

K(I,3) : line number of parent particle or jet, where known, else 0.
    Note that the assignment of a particle to a given jet of a jet
    system is unphysical, and what is given there is only related to
    the way the event was generated.

K(I,4) : normally the line number of the first daughter;
    is 0 for an undecayed particle or unfragmented jet.
    For K(I,1) = 3, 13 or 14 it instead contains special colour
    flow information (for internal use only) on the form
    K(I,4) = 200000000*MCFR + 100000000*MCTO + 10000*ICFR + ICTO,
    where ICFR and ICTO give the line numbers of the partons from which
    the colour comes and to where it goes, respectively, and MCFR and
    MCTO originally are 0 and are set to 1 when the corresponding
    colour connection has been traced in the LUPREP rearrangement
    procedure. (The packing may be changed with MSTU(5).)
    The 'from' colour position may indicate a parton which branched
    to produce the current parton, or a parton created together with
    the current parton but with matched anticolour, while the 'to'
    normally indicates a parton that the current parton branches
    into. Thus, for setting up an initial colour configuration, it
    is normally only the 'from' part that is used, while the 'to' part
    is added by the program in a subsequent call to parton shower
    evolution (for final state radiation; it is the other way around
    for initial state radiation).
    Note: normally most users never have to worry about the exact
    rules for colour flow storage, since they are used mainly for
    internal purposes. However, when it is necessary to define this
    flow, it is recommended to use the LUJOIN routine, since that
    likely would reduce the chances of making a mistake.

K(I,5) : normally the line number of the last daughter;
    is 0 for an undecayed particle or unfragmented jet.
    For K(I,1) = 3, 13 or 14 it instead contains special colour
    flow information (for internal use only) on the form
    K(I,5) = 200000000*MCFR + 100000000*MCTO + 10000*ICFR + ICTO,
    where ICFR and ICTO give the line numbers of the partons from which
    the anticolour comes and to where it goes, respectively, and MCFR
    and MCTO originally are 0 and are set to 1 when the corresponding
    colour connection has been traced in the LUPREP rearrangement
    procedure. (The packing may be changed with MSTU(5).)
    Comments and note: as for K(I,4).

P(I,1) : p_x, momentum in the x direction, in GeV/c.

P(I,2) : p_y, momentum in the y direction, in GeV/c.

P(I,3) : p_z, momentum in the z direction. in GeV/c.

P(I,4) : E, energy, in GeV.

P(I,5) : m, mass, in GeV/c^2. In parton showers, with spacelike
    virtualities, i.e. where Q^2 = - m^2 > 0, P(I,5) = -Q.

V(I,1) : x position of production vertex, in mm.

V(I,2) : y position of production vertex, in mm.

V(I,3) : z position of production vertex, in mm.

V(I,4) : time of production, in mm/c (= 3.33 * 10^-12 s).

V(I,5) : proper lifetime of particle, in mm/c (= 3.33 * 10^-12 s).
    If the particle is not expected to decay, V(I,5) = 0.
    A line with K(I,1) = 4, i.e. a particle that could have decayed,
    but did not do so within allowed region, has the proper nonzero
    V(I,5).
    In the absence of electric or magnetic fields, or other
    disturbances, the decay vertex V' of an unstable particle may be
    calculated as V'(j) = V(I,j) + V(I,5)*P(I,j)/P(I,5), j = 1 - 4.
    _____

2.3. Definition of Initial Configuration or Commonblock Variables

With the use of the conventions described for the event record, it
is possible to specify any initial jet/particle configuration. This
task is simplified for a number of often occuring situations by
the existence of the filling routines below. Several calls can be
combined in the specification. In case one call is enough, the complete
fragmentation/decay chain may be simulated at the same time. At each
call, the value of N is updated to the last line used for information
in the call, so if several calls are used, they should be made with
increasing IP number, or else N should be redefined by hand afterwards.
It should be noted that many users do not come in direct contact with
these routines, since that is taken care of by higher-level routines
for specific processes (LUEEVT, PYTHIA, LEPTO, etc.). As an experiment,
the routine LUGIVE contains a facility for setting various comonblock
variables in a controlled and documented fashion.

SUBROUTINE LU1ENT(IP,KF,PE,THE,PHI)
Purpose: to add one entry to the event record, i.e. either a jet or
    a particle.
IP : line number for the jet/particle.
    If IP = 0 is used, line number 1 is used and LUEXEC is called.
    If IP < 0, line -IP is used, with status code KS = 2 rather than 1;
    thus a jet system may be built up by filling all but the last
    jet of the system with IP < 0.
KF : jet/particle flavour code.
PE : jet/particle energy. If PE is smaller than the mass, the
    jet/particle is taken to be at rest.
THE, PHI : polar and azimuthal angle for the momentum vector of the
    jet/particle.

SUBROUTINE LU2ENT(IP,KF1,KF2,PECM)
Purpose: to add two entries to the event record, i.e. either a two-jet
    system or two separate particles.

IP : line number for the first jet/particle, with second in line IP+1.
    If IP = 0, lines 1 and 2 are used and LUEXEC is called.
    If IP < 0, lines -IP and -IP+1 are used, with status code KS = 3,
    i.e. with special colour connection information, so that a parton
    shower can be generated by a LUSHOW call, followed by a LUEXEC
    call, if so desired (only relevant for jets).
KF1, KF2 : flavour codes for the two jets/particles.
PECM : (= W) the total energy of the system.
Remark: the system is given in the CM frame, with the first jet/particle
    going out in the +z direction.

SUBROUTINE LU3ENT(IP,KF1,KF2,KF3,PECM,X1,X3)
Purpose: to add three entries to the event record, i.e. either a
    three-jet system or three separate particles.
IP : line number for the first jet/particle, with other two in IP+1
    and IP+2.
    If IP = 0, lines 1, 2 and 3 are used and LUEXEC is called.
    If IP < 0, lines -IP through -IP+2 are used, with status code
    KS = 3, i.e. with special colour connection information, so that
    a parton shower can be generated by a LUSHOW call, followed by a
    LUEXEC call, if so desired (only relevant for jets).
KF1, KF2, KF3: flavour codes for the three jets/particles.
PECM : (= W) the total energy of the system.
X1, X3 : $x_i = 2E_i/W$, i.e. twice the energy fraction taken by the
    i:th jet. Thus $x_2 = 2 - x_1 - x_3$, and need not be given.
    Note that not all combinations of x_i are inside the physically
    allowed region.
Remark : the system is given in the CM frame, in the xz-plane, with
    the first jet going out in the +z direction and the third one
    having px > 0.

SUBROUTINE LU4ENT(IP,KF1,KF2,KF3,KF4,PECM,X1,X2,X4,X12,X14)
Purpose: to add four entries to the event record, i.e. either a
    four-jet system or four separate particles (or, for qqbarq'qbar'
    events, two two-jet systems).
IP : line number for the first jet/particle, with other three in lines
    IP+1, IP+2 and IP+3.
    If IP = 0, lines 1, 2, 3 and 4 are used and LUEXEC is called.
    If IP < 0, lines -IP through -IP+3 are used, with status code
    KS = 3, i.e. with special colour connection information, so that
    a parton shower can be generated by a LUSHOW call, followed by a
    LUEXEC call, if so desired (only relevant for jets).
KF1,KF2,KF3,KF4 : flavour codes for the four jets/particles.
PECM : (= W) the total energy of the system.
X1,X2,X4 : $x_i = 2E_i/W$, i.e. twice the energy fraction taken by the
    i:th jet. Thus $x_3 = 2 - x_1 - x_2 - x_4$, and need not be given.
X12,X14 : $x_{ij} = 2p_ip_j/W^2$, i.e. twice the four-vector product of
    the momenta for jets i and j, properly normalized. With the masses
    known, other x_{ij} may be constructed from the x_i and x_{ij} given.
    Note that not all combinations of x_i and x_{ij} are inside the
    physically allowed region.
Remark: the system is given in the CM frame, with the first jet going
    out in the +z direction and the fourth jet lying in the xz-plane
    with $p_x > 0$. The second jet will have $p_y > 0$ and $p_y < 0$ with
    equal probability with the third jet balancing this p_y (this

corresponds to a random choice between the two possible
stereoisomers).

SUBROUTINE LUJOIN(NJOIN,IJOIN)
Purpose: to connect a number of previously defined partons into a
string configuration. Initially the partons must be given with
status codes (KS = K(I,1)) 1, 2 or 3. Afterwards the partons
all have status code 3, i.e. are given with full colour flow
information. Compared to the normal way of defining a parton
system, the partons need therefore not appear in the same
sequence in the event record as they are assumed to do along the
string. It is also possible to call LUSHOW for all or some of the
entries making up the string formed by LUJOIN.
NJOIN: the number of entries that are to be joined by one string.
IJOIN: an one-dimensional array, of size at least NJOIN. The NJOIN
first numbers are the positions of the partons that are to be
joined, given in the order the partons are assumed to appear
along the string. If the system consists entirely of gluons,
the string is closed by connecting back the last to the first
entry.
Remarks: only one string (i.e. one colour singlet) may be defined per
call, but one is at liberty to use any number of LUJOIN calls for
a given event. The program will check that the parton configuration
specified makes sense, and not take any action unless it is. Note,
however, that an initially sensible parton configuration may become
nonsensical, if only some of the partons are reconnected, while the
others are left unchanged.

SUBROUTINE LUGIVE(CHIN)
Purpose: to set the value of any variable residing in the commmonblocks
LUJETS, LUDAT1, LUDAT2, LUDAT3 or LUDAT4. This is done in a more
controlled fashion than by directly including the commonblocks in
the user program, in that array bounds are checked and the old and
new values for the variable changed is written to the output for
reference.
CHIN : character expression of length at most 100 characters, with
requests for variables to be changed, stored in the form
variable1=value1;variable2=value2;variable3=value3 ...
Note that an arbitrary number of instructions can be stored in
one call if separated by semicolons, and that blanks may be included
anyplace. The variable_i may be any single variable in the JETSET
commonblocks, and the value_i must be of the correct integer, real
or character (without extra quotes) type. Array indices and values
must be given explicitly, i.e. can not be variables in their own
right. The exception is that the first index can be preceded by
a C, signifying that the index should be translated from normal
KF to compressed KC code with a LUCOMP call; this is allowed for
the KCHG, PMAS, MDCY and CHAF arrays. If a value_i is omitted,
i.e. with the construction variable=, the current value is written
to the output.

2.4.   The Physics Routines

The physics routines form the major part of the program, but once

the initial jet/particle configuration has been specified and default
parameter values changed, if so desired, only a LUEXEC call is necessary
to simulate the whole fragmentation and decay chain. The physics
involved has been described in a number of different publications,
see e.g. [Sjo86,Sjo88] and references therein. We will therefore
only give a rather brief overview. The routines RLU, RLUGET and RLUSET
have been described in section 1.6, but otherwise would also belong
to this section.

The routine LUSHOW, for timelike shower evolution, is somewhat of a
special classification problem. Its main application is to e+e-
annihilation events, as administrated from LUEEVT, and it would be
natural to place the routine in that group, as has been done in the
past. However, in some decays a qqbar pair is formed with such energy
that gluon radiation corrections should be included, and thus LUSHOW
becomes intimately intertwined with LUDECY. This is particularly
acute with the increased lower mass limits for top. A 60 GeV top, say,
would typically give a qqbar pair coming from the virtual W with an
invariant mass of 30 - 40 GeV, i.e. comparable to those PETRA/PEP
energies where gluon emission corrections are known to be quite
significant. The placing of this routine here does not preclude its
use elsewhere, however.

SUBROUTINE LUEXEC
Purpose: to administrate the fragmentation and decay chain. LUEXEC
    may be called several times, but only entries which have not yet
    been treated (more precisely, have 1 <= KS <= 10) can be affected
    by further calls. This may apply if more jets/particles have been
    added by the user, or if particles previously considered stable are
    now allowed to decay. The actions that will be taken during a
    LUEXEC call can be tailored extensively via the LUDAT1 - LUDAT3
    commonblocks, in particular by setting the MSTJ values suitably.

SUBROUTINE LUPREP(IP)
Purpose: to rearrange parton shower endproducts (marked with KS = 3)
    sequentially along strings; also to (optionally) allow small jet
    systems to collapse into two particles or one only, in the latter
    case with energy and momentum to be shuffled elsewhere in the event;
    also to perform checks that e.g. flavours of colour singlet systems
    make sense.

SUBROUTINE LUSTRF(IP)
Purpose: to generate the fragmentation of an arbitrary colour singlet
    jet system according to the Lund string fragmentation model. In many
    respects, this routine is the very heart and soul of the Lund family
    of programs. All the technical details are documented in [Sjo84].

SUBROUTINE LUINDF(IP)
Purpose: to handle the fragmentation of a jet system according to
    independent fragmentation models, and implement energy, momentum
    and flavour conservation, if so desired. Also the fragmentation of
    a single jet, not belonging to a jet system, is considered here
    (this is of course physical nonsense, but may sometimes be
    convenient for specific tasks).

SUBROUTINE LUDECY(IP)
Purpose: to perform a particle decay, according to known branching
    ratios or different kinds of models, depending on our level of
    knowledge. Various matrix elements are included for specific
    processes.

SUBROUTINE LUKFDI(KFL1,KFL2,KFL3,KF)
Purpose: to generate a new quark or diquark flavour and to combine it
    with an existing flavour to give a hadron.
KFL1: incoming flavour.
KFL2: extra incoming flavour, e.g. for formation of final particle,
    where the flavours are completely specified. Is normally 0.
KFL3: newly created flavour; is 0 if KFL2 is nonzero.
KF: produced hadron. Is 0 if something went wrong (e.g. inconsistent
    combination of incoming flavours).

SUBROUTINE LUPTDI(KFL,PX,PY)
Purpose: to give transverse momentum, e.g. for a qqbar pair created
    in the field, according to independent Gaussian distributions in
    p_x and p_y.

SUBROUTINE LUZDIS(KFL1,KFL3,PR,Z)
Purpose: to generate the longitudinal scaling variable z in jet
    fragmentation, either according to the Lund symmetric fragmentation
    function, or according to a choice of other shapes.

SUBROUTINE LUSHOW(IP1,IP2,QMAX)
Purpose: to generate timelike parton showers, conventional or coherent.
    The performance of the program is regulated by the switches
    MSTJ(41) - MSTJ(49) and parameters PARJ(82) - PARJ(84). In order
    to keep track of the colour flow information, the positions K(I,4)
    and K(I,5) have to be organized properly for showering partons.
    Inside JETSET 7.1, this is done automatically, but for external use
    proper care must be taken.
IP1 > 0, IP2 = 0 : generate a timelike parton shower for the parton
    in line IP1 in commonblock LUJETS, with maximum allowed mass
    QMAX. With only one parton at hand, one can not simultaneously
    conserve both energy and momentum: we here choose to conserve
    energy and jet direction, while longitudinal momentum (along the
    jet axis) is not conserved.
IP1 > 0, IP2 > 0 : generate timelike parton showers for the two
    partons in lines IP1 and IP2 in the commonblock LUJETS,
    with maximum allowed mass for each parton QMAX. For shower
    evolution, the two partons are boosted to their CM frame.
    Energy and momentum is conserved for the pair of partons, although
    not for each individually. One of the two partons may be
    replaced by a nonradiating particle, such as a photon or a
    diquark; the energy and momentum of this particle will then be
    modified to conserve the total energy and momentum.
IP1 > 0, IP2 < 0 : generate timelike parton showers for the -IP2
    (at most 3) partons in lines IP1, IP1+1, ... IPI-IP2-1 in the
    commonblock LUJETS, with maximum allowed mass for each parton
    QMAX. The actions for IP2 = -1 and IP2 = -2 correspond to what
    is described above, but additionally IP2 = -3 may be used to
    generate the evolution starting from three given partons (e.g.

in Upsilon -> ggg). Then the three partons are boosted to their
CM frame, energy is conserved for each parton individually and
momentum for the system as a whole.
QMAX : the maximum allowed mass of a radiating parton, i.e. the
starting value for the subsequent evolution. (In addition, the
mass of a single parton may not exceed its energy, the mass of a
parton in a system may not exceed the invariant mass of the
system.)

SUBROUTINE LUBOEI
Purpose: to include Bose-Einstein effects according to a simple
parametrization. By default, this routine is not called. If called,
this is done after decay of short-lived resonances, but before
decay of long-lived ones. See MSTJ(51) - MSTJ(52).

FUNCTION ULMASS(KF)
Purpose: to give the mass for a parton/particle.

SUBROUTINE LUNAME(KF,CHAU)
CHARACTER CHAU*16
Purpose: to give the parton/particle name (as a character string).

FUNCTION LUCHGE(KF)
Purpose: to give three times the charge for a parton/particle.

FUNCTION LUCOMP(KF)
Purpose: to give the compressed parton/particle code KC for a given
KF code, as required to find entry into mass and decay data tables.
Also checks whether the given KF code is actually an allowed one
(i.e. known by the program), and returns 0 if not. Note that KF
may be positive or negative, while the resulting KC code is never
negative.

SUBROUTINE LUERRM(MERR,MESSAG)
Purpose: to keep track of the number of errors and warnings encountered,
write out information on them, and abort the program in case of too
many errors.

FUNCTION ULALPS(Q2)
Purpose: to calculate the running strong coupling constant alpha_strong
as a function of the momentum transfer Q^2, in first or second
order QCD. See MSTU(111) - MSTU(118) and PARU(111) - PARU(118).
Is currently not used in parton showers, but only for matrix
elements.

FUNCTION ULANGL(X,Y)
Purpose: to calculate the angle from the x and y coordinates.

BLOCK DATA LUDATA
Purpose: to give default values for variables in the LUDAT1 - LUDAT4
and LUDATR commonblocks.

2.5. Event Study and Data Listing Routines

After a LUEXEC call, the event generated is stored in the LUJETS
commonblock, and whatever physical variable is desired may be
constructed from this record. An event may be rotated, boosted or
listed, and particle data may be listed or modified. Via the
functions KLU and PLU the values of some frequently appearing
variables may be obtained more easily. As described in section 2.6,
also more detailed event shape analyses may be performed simply.

SUBROUTINE LUROBO(THE,PHI,BEX,BEY,BEZ)
Purpose: to perform rotations and Lorentz boosts (in that order, if
    both in the same call) of jet/particle momenta and vertex position
    variables.
THE, PHI : standard polar coordinates theta, phi, giving the rotated
    direction of a momentum vector initially along the +z axis.
BEX, BEY, BEZ : gives the direction and size beta of a Lorentz boost,
    such that a particle initially at rest will have p/E = beta
    afterwards.
Remark: all entries 1 through N are affected by the transformation,
    unless lower and upper bounds are explicitly given by MSTU(1) and
    MSTU(2), or if status code KS <= 0.

ENTRY LUDBRB(IMI,IMA,THE,PHI,DBEX,DBEY,DBEZ)
Purpose: to perform rotations and Lorentz boosts (in that order, if
    both in the same call) of jet/particle momenta and vertex position
    variables, for a specific range of entries, and with the boost
    vector given in double precision. Is entry to LUROBO, mainly
    intended for internal use.
IMI, IMA : range of entries affected by transformation, IMI <= I <= IMA.
THE, PHI : standard polar coordinates theta, phi, giving the rotated
    direction of a momentum vector initially along the +z axis.
DBEX, DBEY, DBEZ : gives the direction and size beta of a Lorentz boost,
    such that a particle initially at rest will have p/E = beta
    afterwards. Is to be given in double precision.
Remark: all entries with status codes KS > 0 in the requested range are
    affected by the transformation.

SUBROUTINE LUEDIT(MEDIT)
Purpose: to exclude unstable or undetectable jets/particles from the
    event record. One may also use LUEDIT to store spare copies of
    events (specifically initial parton configuration) that can be
    recalled to allow e.g. different fragmentation schemes to be run
    through with one and the same parton configuration. Finally, an
    event which has been analysed with LUSPHE, LUTHRU or LUCLUS
    (see section 2.6) may be rotated to align the event axis with
    the z direction.
MEDIT : tells which action is to be taken.
    = 0 : empty (KS = 0) and documentation (KS > 20) lines are removed.
        The jets/particles remaining are compressed in the beginning of
        the LUJETS commonblock and the N value is updated accordingly.
        The event history is lost, so that information stored in K(I,3),
        K(I,4) and K(I,5) is no longer relevant.
    = 1 : as =0, but in addition all jets/particles that have
        fragmented/decayed (KS > 10) are removed.
    = 2 : as =1, but also all neutrinos and unknown particles (i.e.
        compressed code KC = 0) are removed.

= 3 : as =2, but also all uncharged, colour neutral particles are
     removed, leaving only charged, stable particles (and
     unfragmented partons, if fragmentation has not been performed).
= 5 : as =0, but also all partons which have branched or been
     rearranged in a parton shower and all particles which have
     decayed are removed, leaving only the fragmenting parton
     configuration and the final state particles.
= 11 : remove lines with K(I,1) < 0. Update event history
     information (in K(I,3) - K(I,5)) to refer to remaining entries.
= 12 : remove lines with K(I,1) = 0. Update event history
     information (in K(I,3) - K(I,5)) to refer to remaining entries.
= 13 : remove lines with K(I,1) = 11, 12 or 15, except for any line
     with K(I,2) = 94. Update event history information (in K(I,3) -
     K(I,5)) to refer to remaining entries. In particular, try to
     trace origin of daughters, for which the mother is decayed,
     back to entries not deleted.
= 14 : remove lines with K(I,1) = 13 or 14, and also any line with
     K(I,2) = 94. Update event history information (in K(I,3) -
     K(I,5)) to refer to remaining entries. In particular, try to
     trace origin of rearranged jets back through the parton shower
     history to the shower initiator.
= 15 : remove lines with K(I,1) > 20. Update event history
     information (in K(I,3) - K(I,5)) to refer to remaining entries.
= 21 : all partons/particles in current event record are stored
     (as a spare copy) in bottom of commonblock LUJETS (is e.g. done
     to save original partons before calling LUEXEC).
= 22 : partons/particles stored in bottom of event record with =21
     are placed in beginning of record again, overwriting previous
     information there (so that e.g. a different fragmentation scheme
     can be used on the same partons). Since the copy at bottom is
     unaffected, repeated calls with =22 can be made.
= 23 : primary partons/particles in the beginning of event record
     are marked as not fragmented or decayed, and number of entries
     N is updated accordingly. Is simpe substitute for =21 plus =22
     when no fragmentation/decay products precede any of the
     original partons/particles.
= 31 : rotate largest axis, determined by LUSPHE, LUTHRU or LUCLUS,
     to sit along the z direction, and the second largest axis into
     the xz plane. For LUCLUS it can be further specified to +z axis
     and xz plane with x > 0, respectively. Requires that one of
     these routines has been called before.
= 32 : mainly intended for LUSPHE and LUTHRU, this gives a further
     alignment of the event, in addition to the one implied by =31.
     The "slim" jet, defined as the side (z > 0 or z < 0) with the
     smallest summed p_T over square root of number of particles,
     is rotated into the +z hemisphere. In the opposite hemisphere
     (now z < 0), the side of x > 0 and x < 0 which has the largest
     summed p_z absolute is rotated into the z < 0, x > 0 quadrant.
     Requires that LUSPHE or LUTHRU has been called before.

SUBROUTINE LULIST(MLIST)
Purpose: to list an event, jet or particle data, or current parameter
     values.
MLIST : determines what is to be listed.
   = 0 : writes a header with program version number and last date

of change; is mostly for internal use.

= 1 : gives a simple list of current event record, in an 80 column
format suitable for viewing directly on the computer terminal.
For each entry, the following information is given: the entry
number I, the parton/particle name (see below), the status code
KS (K(I,1)), the flavour code KF (K(I,2)), the line number of
the mother (K(I,3)), and the three-momentum, energy and mass
(P(I,1) - P(I,5)). If MSTU(3) is nonzero, lines immediately
after the event record proper are also listed. A final line
contains information on total charge, momentum, energy and
invariant mass.
The particle name is given by a call to the routine LUNAME.
For an entry which has decayed/fragmented (KS = 11 - 20), this
particle name is given within parantheses. Similarly, a
documentation line (KS = 21 - 30) has the name enclosed in
expression signs (!...!) and an event/jet axis information line
the name within inequality signs (<...>). If the last character
of the name is a ?, it is a signal that the complete name has
been truncated to fit in, and can therefore not be trusted;
this is very rare. For partons which have been arranged along
strings (KS = 1, 2, 11 or 12), the end of the parton name
column contains information about the colour string arrangement:
a A for the first entry of a string, an I for all intermediate
ones, and a V for the final one (a poor man's vertical rendering
of the doublesided arrow <---->).
It is possible to insert lines just consisting of sequences
of ====== to separate different sections of the event record,
see MSTU(70) - MSTU(80).

= 2 : gives a more extensive list of the current event record, in a
132 column format, suitable for printers or workstations.
For each entry, the following information is given: the entry
number I, the parton/particle name (with padding as described
for MLIST = 1), the status code KS (K(I,1)), the flavour code
KF (K(I,2)), the line number of the mother (K(I,3)), the
decay product/colour flow pointers (K(I,4), K(I,5)), and the
three-momentum, energy and mass (P(I,1) - P(I,5)). If MSTU(3)
is nonzero, lines immediately after the event record proper
are also listed. A final line contains information on total
charge, momentum, energy and invariant mass. Lines with only
====== may be inserted as for MLIST(1).

= 3 : gives the same basic listing as = 2, but with an additional
line for each entry containing information on production vertex
position and time (V(I,1) - V(I,4)) and, for unstable particles,
invariant lifetime (V(I,5)).

= 11 : provides a simple list of all parton/particle codes defined
in the program, with KF code and corresponding particle name.
The list is grouped by particle kind, and only within each group
in ascending order.

= 12 : provides a list of all parton/particle and decay data used in
the program. Each parton/particle code is represented by one
line containing KF flavour code, KC compressed code, particle
name, antiparticle name (where appropriate), electrical and
colour charge (stored in KCHG), mass, resonance width and
maximum broadening, average invariant lifetime (in PMAS) and
whether the particle is considered stable or not (in MDCY).

Immediately after a particle, each decay channel gets one line,
containing decay channel number (IDC read from MDCY), on/off
switch for the channel, matrix element type (MDME), branching
ratio (BRAT), and decay products (KFDP).
The MSTU(14) flag can be used to set the maximum flavour
for which particles are listed, with the default (= 0)
corresponding to separately defined ones (KC > 100 if KF > 0).
In order to keep the size down, decay modes of heavy hadrons
collectively defined are never listed; these have KC codes
84 - 88, where the relevant information may be found.
= 13 : gives a list of current parameter values for MSTU, PARU,
MSTJ and PARJ, and the first 200 entries of PARF. This is
useful to keep check of which default values were changed
in a given run.

SUBROUTINE LUUPDA(MUPDA,LFN)
Purpose: to give the user the ability to update particle data, or to
keep several versions of modified particle data for special
purposes (e.g. charm studies).
MUPDA : gives the type of action to be taken.
= 1 : write a table of particle data, that the user then can
edit at leisure. For ordinary listing of decay data, LULIST(12)
should be used, but that listing could not be read back in
by the program.
For each compressed flavour code KC = 1 - 500, one line is
written containing KC (I5), the basic particle name (pieces
a and b in section 2.1) (2X,A8) in CHAF, the electric (I3),
colour charge (I3) and particle/antiparticle distinction (I3)
codes in KCHG, the mass (F12.5), the mass width (F12.5),
maximum broadening (F12.5) and average invariant lifetime
(2X,F12.5) in PMAS, and the on/off decay switch (I3) in
MDCY(KC,1).
After a KC line follows one line for each possible decay
channel, containing the MDME codes (5X,2I5), the branching
ratio (5X,F12.5) in BRAT, and the KFDP codes for the decay
products (5I8), with trailing 0:s if the number of decay
products is smaller than 5.
= 2 : read in particle data, as written with =1 and thereafter
edited by the user, and use this data subsequently in
the current run. Reading is done with fixed format,
which means that the user has to preserve the format codes
described for =1 during the editing. A number of checks will
be made to see if input looks reasonable, with warnings if not.
If some decay channel is said not to conserve charge, it should
be taken seriously. Warnings that decay is kinematically
unallowed need not be as serious, since that particular decay
mode may not be switched on unless the particle mass is
increased.
= 3 : write current particle data as data lines, which
can be edited into BLOCK DATA LUDATA for a permanent
replacement of the particle data. This option should never
be used by the ordinary user.
LFN : the file number which the data should be written to or
read from. The user must see to it that this file is properly
opened for read or write (since the definition of file names

is machine dependent).

FUNCTION KLU(I,J)
Purpose: to provide various integer-valued event data. Note that many
    of the options available (in particular I > 0, J >= 14) which
    refer to event history will not work after a LUEDIT call.
I=0, J= : properties referring to the complete event.
    = 1 : N, total number of lines in event record.
    = 2 : total number of partons/particles remaining after
        fragmentation and decay.
    = 6 : three times the total charge of remaining (stable) partons
        and particles.
I>0, J= : properties referring to the entry in line no. I of the
    event record.
    = 1 - 5 : K(I,1) - K(I,5), i.e. parton/particle status KS, flavour
        code KF and origin/decay product/colour flow information.
    = 6 : three times parton/particle charge.
    = 7 : 1 for a remaining entry, 0 for a decayed/fragmented/
        documentation entry.
    = 8 : KF code (K(I,2)) for a remaining entry, 0 for a
        decayed/fragmented/documentation entry.
    = 9 : KF code (K(I,2)) for a parton (i.e. not colour neutral entry),
        0 for a particle.
    = 10 : KF code (K(I,2)) for a particle (i.e. colour neutral entry),
        0 for a parton.
    = 11 : compressed flavour code KC.
    = 12 : colour information code, i.e. 0 for colour neutral, 1 for
        colour triplet, -1 for antitriplet and 2 for octet.
    = 13 : flavour of "heaviest" quark or antiquark (i.e. with largest
        code) in hadron or diquark (including sign for antiquark),
        0 else.
    = 14 : generation number. Beam particles or virtual exchange
        particles are generation 0, original jets/particles generation
        1 and then 1 is added for each step in the fragmentation/decay
        chain.
    = 15 : line number of ancestor, i.e. predecessor in first generation
        (generation 0 entries are disregarded).
    = 16 : rank of a hadron in the jet it belongs to. Rank denotes the
        ordering in flavour space, with hadrons containing the original
        flavour of the jet having rank 1, increasing by 1 for each step
        away in flavour ordering. All decay products inherit the rank
        of their parent. Whereas the meaning of a first-rank hadron
        in a quark jet is always well-defined, the definition of higher
        ranks is only meaningful for independently fragmenting quark
        jets. In other cases, rank refers to the ordering in the actual
        simulation, which may be of little interest.
    = 17 : generation number after a collapse of a jet system into one
        particle, with 0 for an entry not coming from a collapse, and
        -1 for entry with unknown history. A particle formed in a
        collapse is generation 1, and then one is added in each decay
        step.
    = 18 : number of decay/fragmentation products (only defined in a
        collective sense for fragmentation).
    = 19 : origin of colour for showering parton, 0 else.
    = 20 : origin of anticolour for showering parton, 0 else.

```
          = 21 : position of colour daughter for showering parton, 0 else.
          = 22 : position of anticolour daughter for showering parton, 0 else.

FUNCTION PLU(I,J)
Purpose: to provide various real-valued event data. Note that some
     of the options available (I > 0, J = 20-25), which are primarily
     intended for studies of systems in their respective CM frame,
     requires that a LUEXEC call has been made for the current initial
     parton/particle configuration, but that the latest LUEXEC call
     has not been followed by a LUROBO one.
I=0, J= : properties referring to the complete event.
     = 1 - 4 : sum of p_x, p_y, p_z and E, respectively, for the stable
          remaining entries.
     = 5 : invariant mass of the stable remaining entries.
     = 6 : sum of electric charge of the stable remaining entries.
I>0, J= : properties referring to the entry in line no. I of the event
     record.
     = 1 - 5 : P(I,1) - P(I,5), i.e. normally p_x, p_y, p_z, E and m for
          jet/particle.
     = 6 : electric charge q.
     = 7 : momentum squared p^2 = p_x^2 + p_y^2 + p_z^2.
     = 8 : momentum p.
     = 9 : transverse momentum squared p_T^2 = p_x^2 + p_y^2.
     = 10 : transverse momentum p_T.
     = 11 : transverse mass squared m_T^2 = m^2 + p_x^2 + p_y^2.
     = 12 : transverse mass m_T.
     = 13 - 14 : polar angle theta in radians (between 0 and pi) or
          degrees, respectively.
     = 15 - 16 : azimuthal angle phi in radians (between -pi and pi)
          or degrees, respectively.
     = 17 : true rapidity y = 0.5*ln((E+p_z)/(E-p_z)).
     = 18 : rapidity y_pi obtained by assuming that the particle is a
          pion when calculating the energy E, to be used in the
          formula above, from the (assumed known) momentum p.
     = 19 : pseudorapidity eta = 0.5*ln((p+p_z)/(p-p_z)).
     = 20 : momentum fraction x_p = 2p/W, where W is the total energy of
          initial jet/particle configuration.
     = 21 : x_F = 2p_z/W (Feynman-x if system is studied in CM frame).
     = 22 : x_T = 2p_T/W.
     = 23 : x_E = 2E/W.
     = 24 : z_+ = (E+p_z)/W.
     = 25 : z_- = (E-p_z)/W.
     ------------------------------------------------------------------------
```

2.6. Event Analysis Routines

The six routines LUSPHE, LUTHRU, LUCLUS, LUCELL, LUJMAS and LUFOWO give
the user the possibility to find some global event shape properties.
The routine LUTABU performs a statistical analysis of a number of
different quantities like particle content, factorial moments and
the energy-energy correlation.
Note that, by default, all remaining partons/particles except
neutrinos are used in the analysis. The JETSET 6.3 default procedure
of also including neutrinos may be obtained with MSTU(41) = 1.
Also note that axes determined are stored in LUJETS, but are not proper

four-vectors and, as a general rule (with some exceptions), should
therefore not be rotated or boosted.

SUBROUTINE LUSPHE(SPH,APL)
Purpose: to diagonalize the momentum tensor, i.e. find the
    eigenvalues lambda_1 > lambda_2 > lambda_3, with sum unity,
    and the corresponding eigenvectors.
    Momentum power dependence is given by PARU(41); default corresponds
    to sphericity, PARU(41)=1. gives measures linear in momenta.
    Which particles (or partons) are used in the analysis is determined
    by the MSTU(41) value.
SPH : 3(lambda_2 + lambda_3)/2, i.e. sphericity (for PARU(41)=2.).
    = -1. : analysis not performed because event contained less
        than two particles (or two exactly back-to-back particles, in
        which case the two transverse directions would be undefined).
APL : 3 lambda_3/2, i.e. aplanarity (for PARU(41)=2.).
    = -1. : as SPH=-1.
Remark: the lines N+1 through N+3 (N-2 through N for MSTU(43) = 2) in
    LUJETS will, after a call, contain the following information:
    K(N+i,1) = 31;
    K(N+i,2) = 95;
    K(N+i,3) : i, the axis number, i=1,2,3;
    K(N+i,4), K(N+i,5) = 0;
    P(N+i,1) - P(N+i,3) : the i:th eigenvector, x,y and z components;
    P(N+i,4) : lambda_i, the i:th eigenvalue;
    P(N+i,5) = 0;
    V(N+i,1) - V(N+i,5) = 0.
    Also, the number of particles used in the analysis is given in
    MSTU(62).

SUBROUTINE LUTHRU(THR,OBL)
Purpose: to find the thrust, major and minor axes and corresponding
    projected momentum quantities, in particular thrust and oblateness.
    The performance of the program is affected by MSTU(44), MSTU(45),
    PARU(42) and PARU(48). In particular, PARU(42) gives the momentum
    dependence, with the default value 1. corresponding to linear
    dependence. Which particles (or partons) are used in the analysis
    is determined by the MSTU(41) value.
THR : thrust (for PARU(42)=1.).
    = -1. : analysis not performed because event contained less
        than two particles.
    = -2. : remaining space in LUJETS (partly used as working area)
        not large enough to allow analysis.
OBL : oblateness (for PARU(42)=1.).
    = -1., -2. : as for THR.
Remark: the lines N+1 through N+3 (N-2 through N for MSTU(43) = 2) in
    LUJETS will, after a call, contain the following information:
    K(N+i,1) = 31;
    K(N+i,2) = 96;
    K(N+i,3) : i, the axis number, i=1,2,3;
    K(N+i,4), K(N+i,5) = 0;
    P(N+i,1) - P(N+i,3) : the thrust, major and minor axis,
        respectively, for i = 1, 2 and 3;
    P(N+i,4) : corresponding thrust, major and minor value;
    P(N+i,5) = 0;

```
    V(N+i,1) - V(N+i,5) = 0.
    Also, the number of particles used in the analysis is given in
    MSTU(62).

SUBROUTINE LUCLUS(NJET)
Purpose: to reconstruct an arbitrary number of jets using a cluster
    analysis method based on particle momenta. Two different distance
    measures are available, the traditional one roughly corresponding
    to relative transverse momentum [Sjo83], and a new one based on the
    JADE method of Bethke, which roughly corresponds to invariant mass
    [Bet86] (in both cases with some important modifications).
    The choice is controlled by MSTU(46). The distance scale d_join,
    above which two clusters may not be joined, is normally given by
    PARU(44). In general, d_join may be varied to describe different
    "jet resolution powers"; the default value, 2.5 GeV, is fairly
    well suited for e+e- physics at 30 - 40 GeV. With the mass distance,
    PARU(44) can be used to set the absolute maximum cluster mass, or
    PARU(45) to set the scaled one, i.e. in $y = m^2/W^2$, where $W^2$ is
    the total invariant mass-squared of the particles being considered.
    It is possible to continue the cluster search from the configuration
    already found, with a new higher d_join scale, by selecting MSTU(48)
    properly. In MSTU(47) one can also require a minimum number of jets
    to be reconstructed; combined with an artificially large d_join this
    can be used to reconstruct a predetermined number of jets. Which
    particles (or partons) are used in the analysis is determined by the
    MSTU(41) value, whereas assumptions about particle masses is given
    by MSTU(42). The parameters PARU(43) and PARU(48) regulate more
    technical details (for events at high energies and large
    multiplicities, however, the choice of a larger PARU(43) may be
    necessary to obtain reasonable reconstruction times).
NJET : the number of clusters reconstructed.
    = -1 : analysis not performed because event contained less than
        MSTU(47) (normally 1) particles, or analysis failed to
        reconstruct the requested number of jets.
    = -2 : remaining space in LUJETS (partly used as working area)
        not large enough to allow analysis.
Remark: if the analysis does not fail, further information is found
    in MSTU(61) - MSTU(63) and PARU(61) - PARU(63). In particular,
    PARU(61) contains the invariant mass for the system analyzed, i.e.
    the number used in determining the denominator of $y = m^2/W^2$.
    PARU(62) gives the generalized thrust, i.e. the sum of (absolute
    values of) cluster momenta divided by the sum of particle momenta
    (roughly the same as multicity). PARU(63) gives the minimum
    distance d (in p_T or m) between two clusters in the final cluster
    configuration, 0 in case of only one cluster. Further, the lines
    N+1 through N+NJET (N-NJET+1 through N for MSTU(43) = 2) in LUJETS
    will, after a call, contain the following information:
    K(N+i,1) = 31;
    K(N+i,2) = 97;
    K(N+i,3) : i, the jet number, with the jets arranged in falling
        order of absolute momentum;
    K(N+i,4) : the number of particles assigned to jet i;
    K(N+i,5) = 0;
    P(N+i,1) - P(N+i,5) : momentum, energy and invariant mass of jet i;
    V(N+i,1) - V(N+i,5) = 0.
```

Also, for a particle which was used in the analysis, K(I,4) = i,
where I is the particle number and i the number of the jet it has
ben assigned to. Undecayed particles not used then have K(I,4) = 0.
An exception is made for lines with K(I,1) = 3 (which anyhow are not
normally interesting for cluster search), where the colour flow
information stored in K(I,4) is left intact.

SUBROUTINE LUCELL(NJET)
Purpose: to provide a simpler cluster routine more in line with
    what is currently used in the study of high-p_T collider events.
    A detector is assumed to stretch in pseudorapidity between -PARU(51)
    and +PARU(51) and be segmented in MSTU(51) equally large eta
    (pseudorapidity) bins and MSTU(52) phi (azimuthal) bins. Transverse
    energy E_T for undecayed entries are summed up in each bin. For
    MSTU(53) nonzero, the energy is smeared by calorimetric resolution
    effects, cell by cell. This is done according to a Gaussian
    distribution; if MSTU(53) = 1 the standard deviation for the E_T is
    PARU(55)*sqrt(E_T), if MSTU(53) = 2 the standard deviation for the
    E is PARU(55)*sqrt(E), E_T and E expressed in GeV. The Gaussian is
    cut off at 0 and at a factor PARU(56) times the correct E_T or E.
    All bins with E_T > PARU(52) are taken to be possible initiators
    of jets, and are tried in falling E_T sequence to check whether
    the total E_T summed over cells no more distant than PARU(54) in
    sqrt((Deltaeta)^2 + (Deltaphi)^2) exceeds PARU(53). If so, these
    cells define one jet, and are removed from further consideration.
    Contrary to LUCLUS, not all particles need be assigned to jets.
    Which particles (or partons) are used in the analysis is determined
    by the MSTU(41) value.
NJET : the number of jets reconstructed (may be 0).
    = -2 : remaining space in LUJETS (partly used as working area)
        not large enough to allow analysis.
Remark: the lines N+1 through N+NJET (N-NJET+1 through N for
    MSTU(43) = 2) in LUJETS will, after a call, contain the
    following information:
    K(N+i,1) = 31;
    K(N+i,2) = 98;
    K(N+i,3) : i, the jet number, with the jets arranged in falling
        order in E_T;
    K(N+i,4) : the number of particles assigned to jet i;
    K(N+i,5) = 0;
    V(N+i,1) - V(N+i,5) = 0;
    Further, for MSTU(54) = 1
    P(N+i,1), P(N+i,2) = position in eta and phi of the center of the
        jet initiator cell, i.e. geometrical center of jet;
    P(N+i,3), P(N+i,4) = position in eta and phi of the E_T-weighted
        center of the jet, i.e. the center of gravity of the jet;
    P(N+i,5) = sum E_T of the jet;
    while for MSTU(54) = 2
    P(N+i,1) - P(N+i,5) : the jet momentum vector, constructed from the
        summed E_T and the eta and phi of the E_T-weighted center of
        the jet as (p_x, p_y, p_z, E, m) =
        E_T(cos(phi), sin(phi), sinh(eta), cosh(eta), 0);
    and for MSTU(54) = 3
    P(N+i,1) - P(N+i,5) : the jet momentum vector, constructed by adding
        vectorially the momentum of each cell assigned to the jet,

assuming that all the E_T was deposited at the center of the
　　　　cell, and with the jet mass in P(N+i,5) calculated from the
　　　　summed E and p as $m^2 = E^2 - p_x^2 - p_y^2 - p_z^2$.
　　Also, the number of particles used in the analysis is given in
　　MSTU(62), and the number of cells hit in MSTU(63).

SUBROUTINE LUJMAS(PMH,PML)
Purpose: to reconstruct high and low jet mass of an event, according to
　　the general suggestion of Clavelli and Smilga [Cla79]. Actually, a
　　simplified algorithm is used, wherein a preliminary division of the
　　event into two hemispheres is done transversely to the sphericity
　　axis. Then one particle at a time is reassigned to the other
　　hemisphere if that reduces the sum of the two jet masses-squared
　　$(m_H^2 + m_L^2)$. The procedure is stopped when no further
　　significant change (see PARU(48)) is obtained. Often, the original
　　assignment is retained as it is. Which particles (or partons)
　　used in the analysis is determined by the MSTU(41) value, whereas
　　assumptions about particle masses is given by MSTU(42).
PMH : heavy jet mass (in GeV).
　　= -2. : remaining space in LUJETS (partly used as working area)
　　　　not large enough to allow analysis.
PML : light jet mass (in GeV).
　　= -2. : remaining space in LUJETS (partly used as working area)
　　　　not large enough to allow analysis.
Remark: After a successful call, MSTU(62) contains the number of
　　particles used in the analysis, and PARU(61) the invariant mass of
　　the system analyzed. The latter number is helpful in constructing
　　scaled jet masses.

SUBROUTINE LUFOWO(H10,H20,H30,H40)
Purpose: to do an event analysis in terms of the Fox-Wolfram moments
　　[Fox79]. The moments H_i are normalized to the lowest one, H_0.
　　Which particles (or partons) are used in the analysis is determined
　　by the MSTU(41) value.
H10 : H_1/H_0. Is =0 if momentum is balanced.
H20 : H_2/H_0.
H30 : H_3/H_0.
H40 : H_4/H_0.
Remark: the number of particles used in the analysis is given in
　　MSTU(62).

SUBROUTINE LUTABU(MTABU)
Purpose: to provide a number of event analysis options which can be
　　be used on each new event, with accumulated statistics to be
　　written out on request. When errors are quoted, these refer to
　　the uncertainty in the average value for the event sample as a
　　whole, rather than to the spread of the individual events, i.e.
　　errors decrease like one over the square root of the number of
　　events analyzed. For a correct use of LUTABU, it is not
　　permissible to freely mix generation and analysis of different
　　classes of events, since only one set of statistics counters exists.
　　A single run may still contain sequential "subruns", between
　　which statistics is reset. Whenever an event is analyzed, the
　　number of particles/partons used is given in MSTU(62).
MTABU : determines which action is to be taken. Generally, a last digit

equal to 0 indicates that the statistics counters for this option
is to be reset; since the counters are reset (by DATA statements)
at the beginning of a run, this is not used normally. Last digit
1 leads to an analysis of current event with respect to the desired
properties. Note that the resulting action may depend on how the
event generated has been rotated, boosted or edited before this
call. The statistics accumulated is output in tabular form with
last digit 2, while it is dumped in the LUJETS commonblock for
last digit 3. The latter option may be useful for interfacing to
graphics output.

= 10 : statistics on parton multiplicity is reset.
= 11 : the parton content of the current event is analyzed,
   classified according to the flavour content of the hard
   interaction and the total number of partons. The flavour
   content is assumed given in MSTU(161) and MSTU(162); these
   are automatically set e.g. in LUEEVT calls.
= 12 : gives a table on parton multiplicity distribution.
= 13 : stores the parton multiplicity distribution of events in
   /LUJETS/, using the following format:
   N = total number of different channels found;
   K(I,1) = 32;
   K(I,2) = 99;
   K(I,3), K(I,4) = the two flavours of the flavour content;
   K(I,5) = total number of events found with flavour content of
      K(I,3) and K(I,4);
   P(I,1) - P(I,5) = relative probability to find given flavour
      content and a total of 1, 2, 3, 4 or 5 partons,
      respectively;
   V(I,1) - V(I,5) = relative probability to find given flavour
      content and a total of 6 - 7, 8 - 10, 11 - 15, 16 - 25 or
      above 25 partons, respectively.
   In addition, MSTU(3) = 1 and
   K(N+1,1) = 32;
   K(N+1,2) = 99;
   K(N+1,5) = number of events analyzed.
= 20 : statistics on particle content is reset.
= 21 : the particle/parton content of the current event is analyzed,
   also for particles which have subsequently decayed and partons
   which have fragmented (unless this has been made impossible by a
   preceding LUEDIT call). Particles are subdivided into primary
   and secondary ones, the main principle being that primary
   particles are those produced in the fragmentation of a string,
   while secondary come from decay of other particles. Since
   particles (top, say), may decay into partons, the distinction
   is not always unique.
= 22 : gives a table of particle content in events.
= 23 : stores particle content in events in /LUJETS/, using the
   following format:
   N = number of different particle species found;
   K(I,1) = 32;
   K(I,2) = 99;
   K(I,3) = particle KF code;
   K(I,5) = total number of particles and antiparticles of this
      species;
   P(I,1) = average number of primary particles per event;

```
                P(I,2) = average number of secondary particles per event;
                P(I,3) = average number of primary antiparticles per event;
                P(I,4) = average number of secondary antiparticles per event;
                P(I,5) = average total number of particles or antiparticles per
                    event.
                In addition, MSTU(3) = 1 and
                K(N+1,1) = 32;
                K(N+1,2) = 99;
                K(N+1,5) = number of events analyzed;
                P(N+1,1) = average primary multiplicity per event;
                P(N+1,2) = average final multiplicity per event;
                P(N+1,3) = average charged multiplicity per event.
= 30 : statistics on factorial moments is reset.
= 31 : analyzes the factorial moments of the multiplicity
       distribution in different bins of rapidity and azimuth.
       Which particles (or partons) are used in the analysis is
       determined by the MSTU(41) value. The selection between usage
       of true rapidity, pion rapidity or pseudorapidity is regulated
       by MSTU(42). The z axis is assumed to be event axis; if this
       is not desirable find an event axis e.g. with LUSPHE or LUTHRU
       and use LUEDIT(31). Maximum (pion, pseudo) rapidity, which sets
       the limit for the rapidity plateau or the experimental
       acceptance, is given by PARU(57).
= 32 : prints a table of the first four factorial moments for
       various bins of pseudorapidity and azimuth. The moments are
       properly normalized so that they would be unity (up to
       statistical fluctuations) for uniform and uncorrelated particle
       production according to Poissonian statistics, but increasing
       for decreasing bin size in case of "intermittent" behaviour
       [Bia86]. The error on the average value is based on the actual
       statistical sample (i.e. does not use any assumptions on the
       distribution to relate errors to the average values of higher
       moments), and decrease as 1/sqrt(number-of-events). Note that
       for small bin sizes, where the average multiplicity is small
       and the factorial moment therefore only very rarely is
       nonvanishing, moment values may fluctuate wildly and the errors
       given may be too low.
= 33 : stores the factorial moments in /LUJETS/, using the format:
       N = 30, with I = 1 - 10 corresponding to results for slicing
           the rapidity range in 2**(1-I) bins, I = 11 - 20 to slicing
           the azimuth in 2**(11-I) bins, and I = 21 - 30 to slicing
           both rapidity and azimuth, each in 2**(21-I) bins;
       K(I,1) = 32;
       K(I,2) = 99;
       K(I,3) = number of bins in rapidity;
       K(I,4) = number of bins in azimuth;
       P(I,1) = rapidity bin size;
       P(I,2) - P(I,5) = <F_2> - <F_5> i.e. mean of second, third,
           fourth and fifth factorial moment;
       V(I,1) = azimuthal bin size;
       V(I,2) - V(I,5) = statistical errors on <F_2> - <F_5>.
       In addition, MSTU(3) = 1 and
       K(31,1) = 32;
       K(31,2) = 99;
       K(31,5) = number of events analyzed.
```

= 40 : statistics on energy-energy correlation is reset.
= 41 : the energy-energy correlation (EEC) of the current event is
    analyzed. Which particles (or partons) are used in the analysis
    is determined by the MSTU(41) value. Events are assumed given
    in their CM frame. The weight assigned to a pair i and j is
    2*E_i*E_j/W^2, where W is the sum of energies of all analyzed
    particles in the event. Energies are determined from the
    momenta of particles, with mass determined according to the
    MSTU(42) value. Statistics is accumulated for the relative
    angle theta_ij, ranging between 0 and 180 degrees, subdivided
    into 50 bins.
= 42 : prints a table of the energy-energy correlation (EEC) and
    its asymmetry (EECA), with errors. The definition of errors is
    not unique. In our approach each event is viewed as one
    observation, i.e. an EEC and EECA distribution is obtained by
    summing over all particle pairs of an event, and then the
    average and spread of this event-distribution is calculated
    in the standard fashion, with the quoted error containing a
    factor of 1/sqrt(number-of-events). It could have been
    possible to view each single particle pair as one observation,
    which would have given somewhat lower errors, but then one
    would also be forced to do a complicated correction procedure
    to account for the pairs in an event not being uncorrelated
    (two hard jets separated by a given angle typically corresponds
    to several pairs at about that angle). Note, however, that
    in our approach the error-squared on an EECA bin is smaller
    than the sum of error-squares of the corresponding EEC bins
    (as it should be). Also note that it is not possible to
    combine the errors of two nearby bins by hand from the
    information given, since nearby bins are correlated (again a
    trivial consequence of the presence of jets).
= 43 : stores the EEC and EECA in /LUJETS/, using the format:
    N = 25;
    K(I,1) = 32;
    K(I,2) = 99;
    P(I,1) = EEC for angles between I-1 and I times 3.6 degrees;
    P(I,2) = EEC for angles between 50-I and 51-I times 3.6 degrees;
    P(I,3) = EECA for angles between I-1 and I times 3.6 degrees;
    P(I,4), P(I,5) : lower and upper edge of angular range of bin I,
        expressed in radians;
    V(I,1) - V(I,3) : errors on the EEC and EECA values stored in
        P(I,1) - P(I,3) (see =42 for comments);
    V(I,4), V(I,5) : lower and upper edge of angular range of bin I,
        expressed in degrees.
    In addition, MSTU(3) = 1 and
    K(26,1) = 32;
    K(26,2) = 99;
    K(26,5) = number of events analyzed.
= 50 : statistics on complete final states is reset.
= 51 : analyzes the particle content of the final state of the
    current event record. During the course of the run, statistics
    is thus accumulated on how often different final states appear.
    Only final states with up to 8 particles are analyzed, and there
    is only reserved space for up to 200 different final states.
    Most high energy events have multiplicities far above 8, so the

main use for this tool is to study the effective branching
ratios obtained with a given decay model for e.g. charm or
bottom hadrons. Then LU1ENT may be used to generate one decaying
particle at a time, with a subsequent analysis by LUTABU.
Depending on at what level this studied is to be carried out,
some particle decays may be switched off, like pi0.

    = 52 : gives a list of the (at most 200) channels with up to 8
particles in the final state, with their relative branching
ratio. The ordering is according to multiplicity, and within
each multiplicity according to an ascending order of KF codes.
The KF codes of the particles belonging to a given channel are
given in descending order.

    = 53 : stores the final states and branching ratios found in
/LUJETS/, using the format:

N = number of different explicit final states found (at most
200);

K(I,1) = 32;

K(I,2) = 99;

K(I,5) = multiplicity of given final state, a number between
1 and 8;

P(I,1) - P(I,5), V(I,1) - V(I,3) : the KF codes of the up to 8
particles of the given final state, converted to real
numbers, with trailing zeroes for positions not used;.

V(I,5) : effective branching ratio for the given final state.

In addition, MSTU(3) = 1 and

K(N+1,1) = 32;

K(N+1,2) = 99;

K(N+1,5) = number of events analyzed;

V(N+1,5) = summed branching ratio for finals states not given
above, either because they contained more than 8 particles
or because all 200 channels have been used up.

2.7. The General Switches and Parameters

The commonblock LUDAT1 is, next to LUJETS, the one a user is most
likely to access. Here he may control in detail what the program
is to do, if the default mode of operation is not satisfactory.

COMMON/LUDAT1/MSTU(200),PARU(200),MSTJ(200),PARJ(200)

Purpose: to give access to a number of status codes and parameters
which regulate the performance of the program as a whole. Here
MSTU and PARU are related to utility functions, as well as a
few parameters of the standard model, while MSTJ and PARJ
affect the underlying physics assumptions. Some of the variables
in LUDAT1 deal specifically with e+e- physics, and are
described in section 3.3.

MSTU(1),MSTU(2) : (D=0,0) can be used to replace the ordinary lower
and upper limits (normally 1 and N) for the action of LUROBO,
and most LUEDIT and LULIST calls. Are reset to 0 in a LUEXEC
call.

MSTU(3) : (D=0) number of lines with extra information added after
line N. Is reset to 0 in a LUEXEC call.

MSTU(4) : (D=4000) number of lines available in the commonblock

LUJETS. Should always be changed if the dimensions of the
    K and P arrays are changed by the user, but should otherwise never
    be touched. Maximum allowed value is 10000, unless MSTU(5) is
    also changed.
MSTU(5) : (D=10000) is used in building up the special colour
    flow information stored in K(I,4) and K(I,5) for K(I,3) = 3, 13
    or 14. The generic form for j = 4 or 5 is
    K(I,j) = 2*MSTU(5)**2*MCFR + MSTU(5)**2*MCTO + MSTU(5)*ICTO + ICFR
    with notation as in section 2.2. One should always have
    MSTU(5) >= MSTU(4). On a 32 bit machine, vaules MSTU(5) >
    20000 may lead to overflow problems, and should be avoided.
MSTU(6) : (D=500) number of KC codes available in the KCHG, PMAS, MDCY,
    and CHAF arrays; should be changed if these dimensions are changed.
MSTU(7) : (D=2000) number of decay channels available in the MDME,
    BRAT and KFDP arrays; should be changed if these dimensions are
    changed.
MSTU(10) : (D=2) use of parton/particle masses in filling routines
    (LU1ENT, LU2ENT, LU3ENT, LU4ENT).
    = 0 : assume the mass to be zero.
    = 1 : keep the mass value stored in P(I,5), whatever it is.
        (This may be used e.g. to describe kinematics with
        off-mass-shell partons).
    = 2 : find masses according to mass tables as usual.
MSTU(11) : (D=6) file number to which all program output is directed.
    It is the responsibility of the user to see to it that the
    corresponding file is also opened for output.
MSTU(12) : (D=1) writing of header (version number and last date of
    change) on output file.
    = 0 : not done.
    = 1 : header is written at first occasion, at which time MSTU(12)
        is set =0.
MSTU(13) : (D=1) writing of information on variable values changed by
    LUGIVE calls.
    = 0 : no information is provided.
    = 1 : information is written to standard output.
MSTU(14) : (D=0) if nonzero, this gives the maximum flavour for which a
    LULIST(12) call will give particle data on possible hadrons. With
    MSTU(14)=5 only known hadrons, i.e. up to bottom, are listed.
    If =0, only separately specified particles are listed (i.e. either
    KF <= 100 or else both KF > 100 and KC > 100).
MSTU(15) : (D=1) selection for characters used in particle names to
    denote an antiparticle; appear in LULIST listings or other LUNAME
    applications.
    = 1 : the tilde character '~'.
    = 2 : the characters 'bar'.
MSTU(16) : (D=1) choice of mother pointers for the particles produced
    by a fragmenting parton system.
    = 1 : all primary particles of a system point to a line with KF =
        92 or 93, for string or independent fragmentation, respectively,
        or to a line with KF = 91 if a jet system has so small a mass
        that it is forced to decay into one or two particles. The two
        (or more) shower initiators of a showering parton system point
        to a line with KF = 94. The entries with KF = 91 - 94 in their
        turn point back to the predecessor partons, so that the KF = 91
        - 94 entries form a part of the event history proper.

= 2 : although the lines with KF = 91 - 94 are present, and contain
                the proper mother and daughter pointers, they are not part of
                the event history proper, in that particles produced in string
                fragmentation point directly to either of the two endpoint
                partons of the string (depending on the side they were generated
                from), particles produced in independent fragmentation point
                to the respective parton they were generated from, particles in
                small mass systems point to either endpoint parton, and
                shower initiators point to the original on-mass-shell
                counterparts. Also the daugher pointers bypass the KF = 91 - 94
                entries. In independent fragmentation, a parton need not produce
                any particles at all, and then have daughter pointers 0.
        Note : MSTU(16) should not be changed between the generation of an
                event and the translation of this event record with a LUHEPC
                call, since this may give an erroneous translation of the event
                history.
MSTU(21) : (D=2) check on possible errors during program execution.
        Obviously no guarantee is given that all errors will be caught,
        but some of the most trivial user-caused errors may be found.
        = 0 : errors do not cause any immediate action, rather the
                program will try to cope, which may mean e.g. that it
                runs into an infinite loop.
        = 1 : parton/particle configurations are checked for possible
                errors. In case of problem, an exit is made from the
                misbehaving subprogram, but the generation of the event is
                continued from there on. For the first MSTU(22) errors a
                a message is printed; after that no messages appear.
        = 2 : parton/particle configurations are checked for possible
                errors. In case of problem, an exit is made from the
                misbehaving subprogram, and subsequently from LUEXEC.
                The user may then choose to correct the error, and continue
                the execution by another LUEXEC call. For the first MSTU(22)
                errors a message is printed, after that the last event is
                printed and execution is stopped.
MSTU(22) : (D=10) max number of errors that are printed.
MSTU(23) : (I) count of number of errors experienced to date.
MSTU(24) : (R) type of latest error experienced; reason that event
        was not generated in full. Is reset at each LUEXEC call.
        = 0 : no error experienced.
        = 1 : have reached end of or are writing outside LUJETS memory.
        = 2 : unknown flavour code or unphysical combination of codes;
                may also be caused by erroneous string connection information.
        = 3 : energy or mass too small or unphysical kinematical variable
                setup.
        = 4 : program is caught in an infinite loop.
        = 5 : momentum, energy or charge was not conserved (even allowing
                for machine precision errors, see PARU(11)); is evaluated only
                after event has been generated in full, and does not apply when
                independent fragmentation without momentum conservation was
                used.
        = 6 : error call from outside the fragmentation/decay package
                (e.g. the e+e- routines).
        = 7 : inconsistent particle data input in LUUPDA (MUPDA = 2) or
                other LUUPDA-related problem.
        = 8 : problems in more peripheral service routines.

```
          = 9 : various other problems.
MSTU(25) : (D=1) printing of warning messages.
     = 0 : no warnings are written.
     = 1 : first MSTU(26) warnings are printed, thereafter no warnings
           appear.
MSTU(26) : (D=10) max number of warnings that are printed.
MSTU(27) : (I) count of number of warnings experienced to date.
MSTU(28) : (R) type of latest warning given, with codes paralleling
     those for MSTU(24), but of a less serious nature.
MSTU(31) : (I) number of LUEXEC calls in present run.
MSTU(32) : (I) number of entries stored with LUEDIT(-1) call.
MSTU(41) : (D=2) partons/particles used in the event analysis routines
     LUSPHE, LUTHRU, LUCLUS, LUCELL, LUJMAS, LUFOWO and LUTABU
     (LUTABU(11) excepted) (cf. LUEDIT).
     = 1 : all partons/particles that have not fragmented/decayed.
     = 2 : ditto, with the exception of neutrinos and unknown particles.
     = 3 : only charged, stable particles, plus any partons still not
           fragmented.
MSTU(42) : (D=2) assumed particle masses, used in calculating energies
     E^2 = p^2 + m^2, as subsequently used in LUCLUS, LUJMAS and LUTABU
     (in the latter also for pseudorapidity/pion rapidity/true rapidity
     selection).
     = 0 : all particles are assumed massless.
     = 1 : all particles, except the photon, are assumed to have the
           charged pion mass.
     = 2 : the true masses are used.
MSTU(43) : (D=1) storing of event analysis information (mainly jet
     axes), in LUSPHE, LUTHRU, LUCLUS and LUCELL.
     = 1 : stored after the event proper, in positions N+1 through
           N+MSTU(3). If several of the routines are used in succession,
           all but the latest information is overwritten.
     = 2 : stored with the event proper, i.e. at the end of the event
           listing, with N updated accordingly. If several of the routines
           are used in succession, all the axes determined are available.
MSTU(44) : (D=4) is the number of the fastest (i.e. with largest
     momentum) particles used to construct the (at most) 10 most
     promising starting configurations for the thrust axis determination.
MSTU(45) : (D=2) is the number of different starting configurations
     above, which have to converge to the same (best) value before this
     is accepted as the correct thrust axis.
MSTU(46) : (D=1) distance measure used for the joining of clusters in
     LUCLUS.
     = 1 : (approximately) relative transverse momentum. Anytime two
           clusters have been joined, particles are reassigned to the
           cluster they now are closest to. The distance cutoff d_join
           is stored in PARU(44).
     = 2 : distance measure as in =1, but particles are never reassigned
           to new jets.
     = 3 : (approximately) total invariant mass. Particles may never be
           reassigned between clusters. The distance cutoff m_min is stored
           in PARU(44).
     = 4 : as =3, but a scaled distance y = m^2/W^2 is used instead of m.
           The distance cutoff y_min is stored in PARU(45).
MSTU(47) : (D=1) the minimum number of clusters to be reconstructed by
     LUCLUS.
```

MSTU(48) : (D=0) mode of operation of the LUCLUS routine.
    = 0 : the cluster search is started from scratch.
    = 1 : the clusters obtained in a previous cluster search on the
        same event (with MSTU(48)=0) are to be taken as the starting
        point for subsequent cluster joining. For this call to have any
        effect, the joining scale in PARU(44) or PARU(45) must have been
        changed. If the event record has been modified after the last
        LUCLUS call, or if any other cluster search parameter setting
        has been changed, the subsequent result is unpredictable.
MSTU(51) : (D=25) number of pseudorapidity bins that the range between
    -PARU(51) and +PARU(51) is divided into to define cell size for
    LUCELL.
MSTU(52) : (D=24) number of azimuthal bins, used to define the
    cell size for LUCELL.
MSTU(53) : (D=0) smearing of correct energy, imposed cell-by-cell in
    LUCELL, to simulate calorimeter resolution effects.
    = 0 : no smearing.
    = 1 : the transverse energy in a cell, E_T, is smeared according
        to a Gaussian distribution with standard deviation
        PARU(55)*sqrt(E_T), where E_T is given in GeV. The Gaussian is
        cut off so that $0 < E_{T\_smeared} < PARU(56)*E_{T\_true}$.
    = 2 : as =1, but it is the energy E rather than the transverse
        energy E_T that is smeared.
MSTU(54) : (D=1) form for presentation of information about
    reconstructed clusters in LUCELL, as stored in LUJETS according to
    the MSTU(43) value.
    = 1 : the P vector in each line contains eta and phi for the
        geometric origin of the jet, eta and phi for the weighted center
        of the jet, and jet E_T, respectively.
    = 2 : the P vector in each line contains a massless four-vector
        giving the direction of the jet, obtained as $(p_x, p_y, p_z, E, m) =$
        $E_T(cos(phi), sin(phi), sinh(eta), cosh(eta), 0)$, where eta and phi
        give the weighted center of a jet and E_T its transverse energy.
    = 3 : the P vector in each line contains a massive four-vector,
        obtained by adding the massless four-vectors of all cells that
        form part of the jet, and calculating the jet mass from
        $m^2 = E^2 - p_x^2 - p_y^2 - p_z^2$. For each cell, the total E_T is
        summed up, and then translated into a massless four-vector
        assuming that all the E_T was deposited in the center of the
        cell.
MSTU(61) : (I) first entry for storage of event analysis information in
    last event analyzed with LUSPHE, LUTHRU, LUCLUS or LUCELL.
MSTU(62) : (R) number of particles/partons used in the last event
    analysis with LUSPHE, LUTHRU, LUCLUS, LUCELL, LUJMAS, LUFOWO or
    LUTABU.
MSTU(63) : (R) in a LUCLUS call, the number of preclusters constructed
    in order to speed up analysis (should be equal to MSTU(62) if
    PARU(43) = 0.). In a LUCELL call, the number of cells hit.
MSTU(70) : (D=0) the number of lines consisting only of equal signs
    (======) that are inserted in the event listing obtained with
    LULIST(1), LULIST(2) or LULIST(3), so as to distinguish different
    sections of the event record on output. At most 10 such lines can
    be inserted; see MSTU(71) - MSTU(80).
MSTU(71) - MSTU(80) : line numbers below which lines consisting only
    of equal signs (======) are inserted in event listings. Only

the first MSTU(70) of the 10 allowed positions are enabled.
MSTU(101) : (D=1) procedure for alpha_em evaluation in the ULALEM
    function.
    = 0 : alpha_em is taken fixed at the value PARU(101).
    = 1 : alpha_em is running with the Q^2 scale, taking into account
        corrections from fermion loops (e, mu, tau, d, u, s, c, b).
MSTU(111) : (D=1) order of alpha_strong evaluation in the ULALPS
    function. Is overwritten in LUEEVT, LUONIA or PYINIT (in the PYTHIA
    program) calls with the value desired for the process under study.
    = 0 : alpha_strong is fixed at the value PARU(111).
    = 1 : first order running alpha_strong is used.
    = 2 : second order running alpha_strong is used.
MSTU(112) : (D=5) the nominal number of flavours assumed in the
    alpha_strong expression, with respect to which Lambda is defined.
MSTU(113) : (D=3) minimum number of flavours that may be assumed in
    alpha_strong expression, see MSTU(112).
MSTU(114) : (D=5) maximum number of flavours that may be assumed in
    alpha_strong expression, see MSTU(112).
MSTU(115) : (D=0) treatment of alpha_strong singularity for $Q^2 \to 0$.
    = 0 : allow it to diverge like $1/\ln(Q^2/Lambda^2)$.
    = 1 : soften the divergence to $1/\ln(1 + Q^2/Lambda^2)$.
    = 2 : freeze Q^2 evolution below PARU(114), i.e. the effective
        argument is $\max(Q^2, PARU(114))$.
MSTU(118) : (I) number of flavours n_f found and used in latest
    ULALPS call.
MSTU(161), MSTU(162) : hard flavours involved in current event, as used
    in an analysis with LUTABU(11). Either or both may be set 0, to
    indicate the presence of one or none hard flavours in event.
    Is normally set by high-level routines, like LUEEVT, but can also
    set by user.
MSTU(181) : (R) JETSET version number.
MSTU(182) : (R) JETSET subversion number.
MSTU(183) : (R) last year of change for JETSET.
MSTU(184) : (R) last month of change for JETSET.
MSTU(185) : (R) last day of change for JETSET.

PARU(1) : (R) pi = 3.1415927.
PARU(2) : (R) 2*pi = 6.2831854.
PARU(3) : (D=0.1973) conversion factor for $GeV^{-1} \to fm$ or $fm^{-1} \to GeV$.
PARU(4) : (D=5.068) conversion factor for $fm \to GeV^{-1}$ or $GeV \to fm^{-1}$.
PARU(5) : (D=0.3894) conversion factor for $GeV^{-2} \to mb$ or
    $mb^{-1} \to GeV^2$.
PARU(6) : (D=2.568) conversion factor for $mb \to GeV^{-2}$ or
    $GeV^2 \to mb^{-1}$.
PARU(11) : (D=0.001) relative error, i.e. nonconservation of momentum
    and energy divided by total energy, that may be attributable to
    machine precision problems before a physics error is suspected
    (see MSTU(24) = 5).
PARU(12) : (D=0.09 GeV^2) effective cutoff in mass-square, below which
    partons may be recombined to simplify (machine precision limited)
    kinematics of string fragmentation.
PARU(13) : (D=0.01) effective angular cutoff in radians for
    recombination of partons, used in conjunction with PARU(12).
PARU(21) : (I) contains the total energy W of all first generation
    jets/particles after a LUEXEC call; to be used by the PLU

function for I>0, J=20-25.

PARU(41) : (D=2.) power of momentum-dependence in LUSPHE, default
    corresponds to sphericity, =1. to linear event measures.

PARU(42) : (D=1.) power of momentum-dependence in LUTHRU, default
    corresponds to thrust.

PARU(43) : (D=0.25 GeV) maximum distance d_{init} allowed in LUCLUS
    when forming starting clusters used to speed up reconstruction.
    The meaning of the parameter is in p_T for MSTU(46) <= 2 and in
    m for MSTU(46) >= 3. If =0., no preclustering is obtained.
    If chosen too large, more joining may be generated at this stage
    than is desirable. The main application is at high energies,
    where some speedup is imperative, and the small details are not
    so important anyway.

PARU(44) : (D=2.5 GeV) maximum distance d_{join}, below which it is
    allowed to join two clusters into one in LUCLUS. Is used for
    MSTU(46) <= 3, i.e. both for p_T and mass distance measure.

PARU(45) : (D=0.05) maximum distance $y_{join} = m^2/W^2$, below which
    it is allowed to join two clusters into one in LUCLUS for
    MSTU(46) = 4.

PARU(48) : (D=0.0001) convergence criterion for thrust (in LUTHRU)
    or generalized thrust (in LUCLUS), or relative change of
    $m_H^2 + m_L^2$ (in LUJMAS), i.e. when the value changes by less
    than this amount between two iterations the process is stopped.

PARU(51) : (D=2.5) defines maximum absolute pseudorapidity used for
    detector assumed in LUCELL.

PARU(52) : (D=1.5 GeV) gives minimum E_T for a cell to be considered
    as a potential jet initiator by LUCELL.

PARU(53) : (D=7.0 GeV) gives minimum summed E_T for a collection of
    cells to be accepted as a jet.

PARU(54) : (D=1.) gives the maximum distance in sqrt((Deltaeta)^2 +
    (Deltaphi)^2) from cell initiator when grouping cells to check
    whether they qualify as a jet.

PARU(55) : (D=0.5) the calorimeter cell resolution assumed when smearing
    the transverse energy (or energy) in LUCELL (see MSTU(53)) is taken
    to be PARU(55)*sqrt(E_T) (or PARU(55)*sqrt(E)) for E_T (or E) in
    GeV.

PARU(56) : (D=2.) maximum factor of upward fluctuation in transverse
    energy or energy in a given cell when calorimeter resolution is
    included in LUCELL (see MSTU(53)).

PARU(57) : (D=3.2) maximum rapidity (or pseudorapidity or pion rapidity,
    see MSTU(42)) used in the factorial moments analysis in LUTABU.

PARU(61) : (I) invariant mass W of a system analyzed with LUCLUS or
    LUJMAS, with energies calculated according to the MSTU(42) value.

PARU(62) : (R) the generalized thrust obtained after a successful LUCLUS
    call, i.e. ratio of summed cluster momenta and summed particle
    momenta.

PARU(63) : (R) the minimum distance d between two clusters in the final
    cluster configuration after a successful LUCLUS call; is 0 if only
    one cluster left.

PARU(101) : (D=0.00729735) alpha_em, the electromagnetic fine structure
    constant at vanishing momentum transfer.

PARU(108) : (I) the alpha_em value obtained in the latest call to the
    ULALEM function.

PARU(102) : (D=0.229) sin^2(theta_W), the weak mixing angle of the
    standard electroweak model.

PARU(111) : (D=0.20) fix alpha-strong value assumed in ULALPS when
    MSTU(111) = 0 (and also in parton showers when alpha_strong is
    assumed fix there).
PARU(112) : (D=0.25 GeV) Lambda used in running alpha_strong expression
    in ULALPS. Like MSTU(111), this value is overwritten by the calling
    physics routines, and is therefore purely nominal.
PARU(113) : (D=1.) the flavour thresholds, for the effective number of
    flavours n_f to use in the alpha_strong expression, are assumed to
    sit at $Q^2 = $ PARU(113)$*m_q^2$, where m_q is the quark mass. May be
    overwritten from the calling physics routine.
PARU(114) : (D=4 GeV^2) Q^2 value below which the alpha_strong value is
    assumed constant for MSTU(115) = 2.
PARU(115) : (D=10.) maximum alpha_strong value that ULALPS will ever
    return; is used as a last resort to avoid singularities.
PARU(117) : (I) Lambda value (associated with MSTU(118) effective
    flavours) obtained in latest ULALPS call.
PARU(118) : (I) alpha_strong value obtained in latest ULALPS call.
PARU(121) : (D=1.) $\tan^2(\beta)$ parameter relevant for H+- couplings in
    a two Higgs doublet scenario.
PARU(141) : (D=1.) the W' couplings to quarks are assumed to behave
    just like those of the W, but multiplied by PARU(141). Note that
    e.g. the W' width contains couplings-squared, and thus depends
    quadratically on PARU(141).
PARU(142) : (D=1.) the W' couplings to leptons are assumed to behave
    just like those of the W, but multiplied by PARU(142). Comment as
    for PARU(141).
PARU(143) : (D=1.) the W' -> W + Z coupling, if this decay mode is at
    all assumed to exist, is assumed to behave just like the standard
    model W -> W + Z one, multiplied by PARU(143)$*(m_W/m_{W'})^2$. This
    gives a W' -> W + Z partial width that increases proportionately
    to the W' mass. Comment as for PARU(141). Also the Z' -> W+ + W-
    are assumed like in the standard model, multiplied by
    PARU(143)$*(m_W/m_{Z'})^2$.

MSTJ(1) : (D=1) choice of fragmentation scheme.
    = 0 : no jet fragmentation at all.
    = 1 : string fragmentation according to the Lund model.
    = 2 : independent fragmentation, according to specification
        in MSTJ(2) and MSTJ(3).
MSTJ(2) : (D=3) gluon jet fragmentation scheme in independent
    fragmentation.
    = 1 : a gluon is assumed to fragment like a random d, u or s
        quark or antiquark.
    = 2 : as =1, but longitudinal (see PARJ(43), PARJ(44) and PARJ(59))
        and transverse (see PARJ(22)) momentum properties of quark or
        antiquark substituting for gluon may be separately specified.
    = 3 : a gluon is assumed to fragment like a pair of a
        d, u or s quark and its antiquark, sharing the gluon energy
        according to the Altarelli-Parisi splitting function.
    = 4 : as =3, but longitudinal (see PARJ(43), PARJ(44) and PARJ(59))
        and transverse (see PARJ(22)) momentum properties of quark and
        antiquark substituting for gluon may be separately specified.
MSTJ(3) : (D=0) energy, momentum and flavour conservation options
    in independent fragmentation. Whenever momentum conservation is
    described below, energy and flavour conservation is also

```
          implicitly assumed. For physics details, see e.g. [Sjo84a].
          = 0 : no explicit conservation of any kind.
          = 1 : particles share momentum imbalance compensation according
                to their energy (roughly equivalent to boosting event to CM
                frame). This is similar to the approach in the Ali et al.
                program [Ali80].
          = 2 : particles share momentum imbalance compensation according
                to their longitudinal mass with respect to the imbalance
                direction.
          = 3 : particles share momentum imbalance compensation equally.
          = 4 : transverse momenta are compensated separately within
                each jet, longitudinal momenta are rescaled so that ratio
                of final jet to initial parton momentum is the same for
                all the jets of the event. This is similar to the approach in
                the Hoyer et al. program [Hoy79].
          = 5 : only flavour is explicitly conserved.
          = 6 - 10 : as =1-5, except that above several colour
                singlet systems that followed immediately after each other
                in the event listing (e.g. qqbarqqbar) were treated as
                one single system, whereas here they are treated as
                separate systems.
          = -1 : independent fragmentation, where also particles moving
                backwards with respect to the jet direction are kept, and
                thus the amount of energy and momentum mismatch may be large.
MSTJ(11) : (D=1) choice of longitudinal fragmentation function, i.e.
          how large a fraction of the energy available a newly-created
          hadron takes.
          = 1 : the Lund symmetric fragmentation function, see PARJ(41) -
                PARJ(45).
          = 2 : choice of some different forms for each flavour separately
                including Field-Feynman [Fie78] and SLAC heavy flavour
                [Pet83] functions, see PARJ(51) - PARJ(59).
          = 3 : hybrid scheme, where light flavours are treated with
                symmetric Lund (=1), but charm and heavier can be separately
                chosen, e.g. according to the SLAC function (=2).
MSTJ(12) : (D=2) choice of baryon production model.
          = 0 : no baryon-antibaryon pair production at all; initial diquark
                treated as a unit.
          = 1 : diquark-antidiquark pair production allowed; diquark treated
                as a unit [And82].
          = 2 : diquark-antidiquark pair production allowed;, with
                possibility for diquark to be split according to the "popcorn"
                scheme [And85].
          = 3 : as =2, but additionally the production of first rank baryons
                may be suppressed by a factor PARJ(19).
MSTJ(13) : (D=0) generation of transverse momentum for endpoint quark(s)
          of single quark jet or qqbar jet system (in multijet events
          no endpoint transverse momentum is ever allowed for).
          = 0 : no transverse momentum for endpoint quarks.
          = 1 : endpoint quarks obtain transverse momenta like ordinary
                qqbar pairs produced in the field (see PARJ(21)); for
                two-jet systems the endpoints obtain balancing transverse
                momenta.
MSTJ(14) : (D=1) treatment of colour singlet jet systems with low
          invariant masses.
```

```
           = 0 : no precautions are taken, meaning that problems may occur
                 in LUSTRF (or LUINDF) later on.
           = 1 : small jet systems are allowed to collapse into two
                 particles or, failing that, one single particle. Normally
                 all small systems are treated this way, starting with the
                 smallest one, but some systems would require more work and
                 are left untreated; they include diquark-antidiquark pairs
                 below the two-particle threshold.
MSTJ(15) : (D=0) production probability for new flavours.
           = 0 : according to standard Lund parametrization, as given by
                 PARJ(1) - PARJ(20).
           = 1 : according to probabilities stored by the user in PARF(201) -
                 PARF(1960); note that no default values exist here, i.e. PARF
                 must be set by the user. The MSTJ(12) switch can still be used
                 to set baryon production mode, with the modification that
                 MSTJ(12) = 2 here allows an arbitrary number of mesons to be
                 produced between a baryon and an antibaryon (since the
                 probability for diquark -> meson + new diquark is assumed
                 independent of prehistory).
MSTJ(21) : (D=2) form of particle decays.
           = 0 : all particle decays are inhibited.
           = 1 : a particle declared unstable in the MDCY vector, and with
                 decay channels defined, may decay within the region given by
                 MSTJ(22). A particle may decay into jets, which then fragment
                 further according to the MSTJ(1) value.
           = 2 : as =1, except that a qqbar jet system produced in a decay
                 (e.g. of a B meson) is always allowed to fragment according to
                 string fragmentation, rather than according to the MSTJ(1) value
                 (this means that momentum, energy and charge are conserved in
                 the decay).
MSTJ(22) : (D=1) cutoff on decay length for a particle that is allowed
           to decay according to MSTJ(21) and the MDCY value.
           = 1 : a particle declared unstable is also forced to decay.
           = 2 : a particle is decayed only if its average invariant lifetime
                 is larger than PARJ(71).
           = 3 : a particle is decayed only if the decay vertex is within a
                 distance PARJ(72) of the origin.
           = 4 : a particle is decayed only if the decay vertex is within a
                 cylindrical volume with radius PARJ(73) in the xy-plane and
                 extent to +-PARJ(74) in the z direction.
MSTJ(23) : (D=1) possibility of having a shower evolving from a qqbar
           pair created as decay products.
           = 0 : never.
           = 1 : whenever the decay channel matrix element code is MDME(IDC,2)
                 = 22, 23 or 33, the two first decay products (if they are
                 partons) are allowed to shower, like a colour singlet
                 subsystem, with maximum virtuality given by the invariant mass
                 of the pair.
MSTJ(24) : (D=2) particle masses.
           = 0 : discrete mass values are used.
           = 1 : particles registered as having a mass width in the PMAS
                 vector are given a mass according to a truncated Breit-Wigner
                 shape, linear in m:
                 P(m) dm = 1/((m-m_0)^2 + Gamma^2/4) dm.
           = 2 : as =1, but gauge bosons (actually all particles with
```

```
                abs(KF) <= 100) are distributed according to a Breit-Wigner
                quadratic in m, as obtained from propagators.
          = 3 : as =1, but Breit-Wigner shape is always quadratic in m:
                P(m) dm^2 = 1/((m^2 - m_0^2)^2 + m_0^2*Gamma^2) dm^2.
MSTJ(25) : (D=1) inclusion of the W propagator, in addition to the
          standard, "infinitely heavy" weak V-A matrix element, in the decay
          of a t, l or h quark, or chi lepton.
          = 0 : not included.
          = 1 : included.
MSTJ(41) : (D=1) type of branchings allowed in shower.
          = 0 : no branchings at all, i.e. shower is switched off.
          = 1 : QCD type branchings of quarks and gluons.
          = 2 : also emission of photons off quarks; the photons are assumed
                on mass-shell.
MSTJ(42) : (D=2) branching mode for timelike showers.
          = 1 : conventional branching, i.e. without angular ordering.
          = 2 : coherent branching, i.e. with angular ordering.
MSTJ(43) : (D=4) choice of z definition in branching.
          = 1 : energy fraction in grandmother's rest frame ("local,
                constrained").
          = 2 : energy fraction in grandmother's rest frame assuming massless
                daughters, with energy and momentum reshuffled for massive ones
                ("local, unconstrained").
          = 3 : energy fraction in CM frame of the showering partons ("global,
                constrained").
          = 4 : energy fraction in CM frame of the showering partons assuming
                massless daughters, with energy and momentum reshuffled for
                massive ones ("global, unconstrained").
MSTJ(44) : (D=2) choice of alpha_strong scale for shower.
          = 0 : fixed at PARU(111) value.
          = 1 : running with Q^2 = m^2/4, m mass of decaying parton,
                Lambda as stored in PARJ(81) (natural choice for conventional
                showers).
          = 2 : running with Q^2 = z*(1-z)*m^2, i.e. roughly p_T^2 of
                branching, Lambda as stored in PARJ(81) (natural choice for
                coherent showers).
MSTJ(45) : (D=5) maximum flavour that can be produced in shower by
          g -> q + qbar; also used to determine the maximum number of
          active flavours in the alpha_strong factor in parton showers
          (here with a minimum of 3).
MSTJ(46) : (D=0) nonhomogeneous azimuthal distributions in branchings
          in shower.
          = 0 : azimuthal angle is chosen uniformly.
          = 1 : nonhomogeneous azimuthal angle in gluon decays due to a
                kinematics-dependent effective gluon polarization.
          = 2 : nonhomogeneous azimuthal angle in gluon decay due to
                interference with nearest neighbour (in colour).
          = 3 : nonhomogeneous azimuthal angle in gluon decay due to both
                polarization (=1) and interference (=2).
MSTJ(47) : (D=1) corrections to the lowest order qqbarg three-jet matrix
          element at the first branching of either initial parton in a
          shower.
          = 0 : no corrections.
          = 1 : included whenever scattered partons are qqbar.
          = 2 : always included when shower starts from two partons.
```

MSTJ(48) : (D=0) possibility to impose maximum angle of first
    branching in shower.
    = 0 : no explicit maximum angle.
    = 1 : maximum angle given by PARJ(85) for single showering parton,
        by PARJ(85) and PARJ(86) for pair of showering partons.
MSTJ(49) : (D=0) possibility to change the branching probabilities
    according to some alternative toy models (note that the
    alpha-strong Q2 evolution may well be different in these models,
    but that only the MSTJ(44) options are at the disposal of the
    user).
    = 0 : standard QCD branchings.
    = 1 : branchings according to a scalar gluon theory, i.e. the
        splitting kernels in the Altarelli-Parisi equations are,
        with a common factor (alpha_strong)^*/(2*pi) omitted,
        $P(q \rightarrow qg) = (2/3) * (1-z)$, $P(g \rightarrow gg) = PARJ(87)$,
        $P(g - q\bar{q}) = PARJ(88)$ (for each separate flavour).
        The couplings of the gluon have been left as free parameters,
        since they depend on the colour structure assumed. Note that,
        since a spin 0 object decays isotropically, the gluon splitting
        kernels contain no z dependence.
    = 2 : brachings according to an Abelian vector gluon theory, i.e.
        the colour factors are changed (compared to QCD) according to
        $C\_F = 4/3 \rightarrow 1$, $N\_C = 3 \rightarrow 0$, $T\_R = 1/2 \rightarrow 3$.
MSTJ(51) : (D=0) inclusion of Bose-Einstein effects.
    = 0 : no effects included.
    = 1 : effects included according to an exponential parametrization
        $C\_2(Q) = 1 + PARJ(92)*exp(-Q/PARJ(93))$, where $C\_2(Q)$ represents
        the ratio of particle production at Q with Bose-Einstein effects
        to that without, and the relative momentum Q is defined by
        $Q^2(p\_1,p\_2) = -(p\_1 - p\_2)^2 = (p\_1 + p\_2)^2 - 4m^2$. Particles
        with width broader than PARJ(91) are assumed to have time to
        decay before Bose-Einstein effects are to be considered.
    = 2 : effects included according to a Gaussian parametrization
        $C\_2(Q) = 1 + PARJ(92)*exp(- (Q/PARJ(93))^2)$, with notation
        and comments as above.
MSTJ(52) : (D=3) number of particle species for which Bose-Einstein
    correlations are to be included, ranged along the chain pi+, pi-,
    pi0, K+, K-, K0S, K0L, eta and eta'. Default corresponds to
    including all pions (pi+, pi-, pi0), 7 to including all Kaons as
    well, and 9 is maximum.
MSTJ(91) : (I) flag when generating gluon jet with options
    MSTJ(2) = 2 or 4 (then =1, else =0).
MSTJ(92) : (I) flag that a qqbar or gg pair or a ggg triplet created
    in LUDECY should be allowed to shower, is 0 if no pair or triplet,
    is the entry number of the first parton if a pair indeed exists,
    is the entry number of the first parton, with a - sign, if a triplet
    indeed exists.
MSTJ(93) : (I) switch for ULMASS action. Is reset to 0 in ULMASS call.
    = 0 : ordinary action.
    = 1 : light (d, u, s, c, b) quark masses are taken from PARF(101) -
        PARF(105) rather than PMAS(1,1) - PMAS(5,1). Diquark masses are
        given as sum of quark masses, without spin splitting term.
    = 2 : as = 1. Additionally the constant terms PARF(121) and
        PARF(122) are subtracted from quark and diquark masses,
        respectively.

PARJ(1) : (D=0.10) is P(qq)/P(q), the suppression of diquark-antidiquark
    pair production in the colour field, compared to quark-antiquark
    production.
PARJ(2) : (D=0.30) is P(s)/P(u), the suppression of s quark pair
    production in the field compared to u or d pair production.
PARJ(3) : (D=0.4) is (P(us)/P(ud))/(P(s)/P(d)), the extra suppression
    of strange diquark production compared to the normal suppression
    of strange quarks.
PARJ(4) : (D=0.05) is (1/3)P(ud_1)/P(ud_0), the suppression of spin 1
    diquarks compared to spin 0 ones (excluding the factor 3 coming
    from spin counting).
PARJ(5) : (D=0.5) parameter determining relative occurence of baryon
    production by BMBbar and by BBbar configurations in the popcorn
    baryon production model, roughly
    P(BMBbar)/(P(BBbar)+P(BMBbar)) = PARJ(5)/(0.5+PARJ(5)).
PARJ(6) : (D=0.5) extra suppression for having a ssbar pair shared
    by the B and Bbar of a BMBbar situation.
PARJ(7) : (D=0.5) extra suppression for having a strange meson M in
    a BMBbar configuration.
PARJ(11) : (D=0.5) is the probability that a light meson (containing
    u and d quarks only) has spin 1 (with 1-PARJ(11) the probability
    for spin 0) when formed in fragmentation.
PARJ(12) : (D=0.6) is the probability that a strange meson has spin 1.
PARJ(13) : (D=0.75) is the probability that a charm or heavier meson
    has spin 1.
PARJ(14) : (D=0.) is the probability that a spin = 0 meson is produced
    with an orbital angular momentum 1, a for total spin = 1.
PARJ(15) : (D=0.) is the probability that a spin = 1 meson is produced
    with an orbital angular momentum 1, for a total spin = 0.
PARJ(16) : (D=0.) is the probability that a spin = 1 meson is produced
    with an orbital angular momentum 1, for a total spin = 1.
PARJ(17) : (D=0.) is the probability that a spin = 1 meson is produced
    with an orbital angular momentum 1, for a total spin = 2.
Note on PARJ(11) - PARJ(17) : the end result of the numbers above is
    that, with i = 11, 12 or 13, depending on flavour content,
    P(S = 0, L = 0, J = 0) = (1-PARJ(i)) * (1-PARJ(14)),
    P(S = 0, L = 1, J = 1) = (1-PARJ(i)) * PARJ(14),
    P(S = 1, L = 0, J = 1) = PARJ(i) * (1-PARJ(15)-PARJ(16)-PARJ(17)),
    P(S = 1, L = 1, J = 0) = PARJ(i) * PARJ(15),
    P(S = 1, L = 1, J = 1) = PARJ(i) * PARJ(16),
    P(S = 1, L = 1, J = 2) = PARJ(i) * PARJ(17),
    where S is the quark "true" spin and J is the total spin, usually
    called the spin of the meson.
PARJ(18) : (D=1.) is an extra suppression factor multiplying the
    ordinary SU(6) weight for spin 3/2 baryons, and hence a means
    to break SU(6) in addition to the dynamic breaking implied by
    PARJ(2), PARJ(3), PARJ(4), PARJ(6) and PARJ(7).
PARJ(19) : (D=1.) extra baryon suppression factor, which multiplies
    the ordinary diquark-antidiquark production probability for the
    breakup closest to the endpoint of a string, but leaves other
    breaks unaffected. Is only used for MSTJ(12) = 3.
PARJ(21) : (D=0.35 GeV) corresponds to the width in the Gaussian p_x
    and p_y transverse mementum distributions for primary hadrons.
PARJ(22) : (D=1.) relative increase in transverse momentum in a gluon

jet generated with MSTJ(2) = 2 or 4.

PARJ(31) : (D=0.1 GeV) gives the remaining W_+ below which the
  generation of a single jet is stopped (it is chosen smaller than a
  pion mass, so that no hadrons moving in the forward direction are
  missed).

PARJ(32) : (D=1. GeV) is, with quark masses added, used to define the
  minimum allowable energy of a colour singlet jet system.

PARJ(33) - PARJ(35) : (D=0.8 GeV, 1.5 GeV, 0.8 GeV) are, together with
  quark masses, used to define the remaining energy below which the
  fragmentation of a jet system is stopped and two final hadrons
  formed. The three alternatives refer to MSTJ(11)=1 through 3.

PARJ(36) : (D=2.) represents the dependence on the mass of the final
  quark pair for defining the stopping point of the fragmentation.
  Is strongly correlated to the choice of PARJ(33) - PARJ(35).

PARJ(37) : (D=0.2) relative width of the smearing of the stopping point
  energy.

PARJ(38) - PARJ(40) : (D=2.5, 0.6, 2.5) refers to the probability for
  reverse rapidity ordering of the final two hadrons, with transverse
  masses m_{T_1} and m_{T_2}, for a total remaining transverse mass W_{rem}.
  The probability is parametrized in the form
  $P(reverse\ ordering) = 0.5*((m_{T\_1} + m_{T\_2})/W_{rem})^d$ where
  $d = PARJ(38)*(m_{T\_1}^2 + m_{T\_2}^2)^2$ for MSTJ(11) = 1,
  $d = PARJ(39)$ for MSTJ(11) = 2, and
  $d = PARJ(40)*(m_{T\_1}^2 + m_{T\_2}^2)^2$ for MSTJ(11) = 3.

PARJ(41), PARJ(42) : (D=0.5, 0.9 GeV^-2) give the a and b parameters of
  the symmetric Lund fragmentation function for MSTJ(11) = 1 (and
  MSTJ(11) = 3 for ordinary hadrons)
  $f(z) = (1/z) * (1-z)^a * exp(-b*m_T^2/z)$
  where m_T is the transverse mass of the hadron.

PARJ(43), PARJ(44) : (D=0.5, 0.9 GeV^-2) give the a and b parameters as
  above for the special case of a gluon jet generated with IF and
  MSTJ(2) = 2 or 4.

PARJ(45) : (D=0.5) the amount by which the effective a parameter in the
  Lund flavour dependent symmetric fragmentation function is assumed
  to be larger than the normal a when diquarks are produced.
  More specifically, with
  $f(z) = (1/z) * z^{a1} * ((1-z)/z)^{a2} * exp(-b*m_T^2/z)$
  $a1 = PARJ(41)$ when considering the fragmentation of a quark and
  $= PARJ(41) + PARJ(45)$ for the fragmentation of a diquark, with
  corresponding expression for a2 depending on whether the newly
  created object is a quark or diquark (for an independent gluon jet
  generated with MSTJ(2) = 2 or 4, replace PARJ(41) by PARJ(43)).
  In the popcorn model, a meson created in between the baryon and
  antibaryon has $a1 = a2 = PARJ(41) + PARJ(45)$.

PARJ(51) - PARJ(58) : (D=3*0.77, 5*0.) give four possible ways to
  parametrize the fragmentation function for MSTJ(11) = 2 (and
  MSTJ(11) = 3 for charm and heavier). The fragmentation of each
  flavour KFL may be chosen separately; for a diquark the flavour
  of the heaviest quark is used. With $c = PARJ(50+KFL)$ the
  parametrizations are:
  $0 <= c <= 1$: Field-Feynman, $f(z) = 1 - c+ 3*c*(1-z)^2$;
  $-1 <= c < 0$: SLAC, $f(z) = 1/(z*(1-1/z-(-c)/(1-z))^2)$;
  $c > 1$: power peaked at z=0, $f(z) = (1-z)^{(c-1)}$;
  $c < -1$: power peaked at z=1, $f(z) = z^{(-c-1)}$.

PARJ(59) : (D=1.) replaces PARJ(51) - PARJ(53) for gluon jet generated

with MSTJ(2) = 2 or 4.
PARJ(61) - PARJ(63) : (D=4.5, 0.7, 0.) parametrizes the energy
    dependence of the primary multiplicity distribution in phase space
    decays.
PARJ(64) : (0.003 GeV) minimum kinetic energy in decays (safety margin
    for numerical precision errors).
PARJ(65) : (D=0.5 GeV) mass which, in addition to the spectator quark or
    diquark mass, is not assumed to partake in the weak decay of a
    heavy quark in a hadron.
PARJ(66) : (D=0.5) relative probability that colour is rearranged
    when two singlets are to be formed from decay products. Only
    applies for MDME(IDC,2) = 11 - 30, i.e. low mass phase space decays.
PARJ(71) : (D=10 mm) maximum average invariant lifetime for particles
    allowed to decay in the MSTJ(22) = 2 option. With the default value,
    K_S0, Lambda, Sigma-, Sigma+, Xi-, XI0 and Omega- are stable (in
    addition to those normally taken to be stable), but charm and
    bottom do still decay.
PARJ(72) : (D=1000 mm) maximum distance from the origin at which a
    decay is allowed to take place in the MSTJ(22) = 3 option.
PARJ(73) : (D=100 mm) maximum cylindrical distance rho =
    sqrt(x^2 + y^2) from the origin at which a decay is allowed to take
    place in the MSTJ(22) = 4 option.
PARJ(74) : (D=1000 mm) maximum z distance from the origin at which a
    decay is allowed to take place in the MSTJ(22) = 4 option.
PARJ(81) : (D=0.40 GeV) Lambda value used in running alpha_strong for
    parton showers (see MSTJ(44)).
PARJ(82) : (D=1.0 GeV) invariant mass cutoff m_min of parton showers,
    below which partons are not assumed to radiate. For Q^2 = p_T^2
    (MSTJ(44) = 2) PARJ(82)/2 additionally gives the minimum p_T of
    a branching. To avoid infinite alpha_strong values, one must have
    PARJ(82) > 2*PARJ(81) for MSTJ(44) >= 1 (this is automatically
    checked in the program, with 2.2*PARJ(81) as the lowest value
    attainable).
PARJ(83) : (D=1.0 GeV) invariant mass cutoff m_min used for photon
    emission in parton showers, below which quarks are not assumed
    to radiate. The function of PARJ(83) closely parallels that of
    PARJ(82) for QCD branchings, but there is a priori no requirement
    that the two be equal.
PARJ(85), PARJ(86) : (D=10.,10.) maximum opening angles allowed
    in the first branching of parton showers; see MSTJ(48).
PARJ(87) : (D=0.) coupling of g -> gg in scalar gluon shower, see
    MSTJ(49) = 1.
PARJ(88) : (D=0.) coupling of g -> qqbar in scalar gluon shower (per
    quark species), see MSTJ(49) = 1.
PARJ(91) : (D=0.020 GeV) minimum particle width in PMAS(KC,2), above
    which particle decays are assumed to take place before the stage
    where Bose-Einstein effects are introduced.
PARJ(92) : (D=1.) nominal strength of Bose-Einstein effects for Q = 0,
    see MSTJ(51). This parameter, often denoted lambda, expresses the
    amount of incoherence in particle production. Due to the simplified
    picture used for the Bose-Einstein effects, in particular for
    effects from three nearby identical particles, the actual lambda
    of the simulated events may be larger than the input value.
PARJ(93) : (D=0.20 GeV) size of the Bose-Einstein effect region in
    terms of the Q variable, see MSTJ(51). The more conventional

measure, in terms of the radius R of the production volume, is
    given by R = hslash/PARJ(93) = 0.2 fm GeV/PARJ(93) =
    PARU(3)/PARJ(93).
 _____

2.8. Further Parameters and Particle Data

The following commonblocks are maybe of a more peripheral interest, with
the exception of the MDCY array, which allows a selective inhibiting
of particle decays, and PMAS(6,1), the top quark mass.

COMMON/LUDAT2/KCHG(500,3),PMAS(500,4),PARF(2000),VCKM(4,4)
Purpose: to give access to a number of flavour treatment constants or
    parameters and particle/parton data.

KCHG(KC,1) : three times particle/parton charge for compressed code KC.

KCHG(KC,2) : colour information for compressed code KC.
    = 0 : colour singlet particle.
    = 1 : quark or antidiquark.
    = -1 : antiquark or diquark.
    = 2 : gluon.

KCHG(KC,3) : particle/antiparticle distinction for compressed code KC.
    = 0 : the particle is its own antiparticle.
    = 1 : a nonidentical antiparticle exists.

PMAS(KC,1) : particle/parton mass (in GeV) for compressed code KC.

PMAS(KC,2) : the total width Gamma (in GeV) of an assumed symmetric
    Breit-Wigner mass shape for compressed particle code KC.

PMAS(KC,3) : the maximum deviation (in GeV) from the PMAS(KC,1) value
    at which the Breit-Wigner shape above is truncated.

PMAS(KC,4) : the average lifetime tau for compressed particle code KC,
    with c*tau in mm, i.e. tau in units of 3.33 * 10^-12 s.

PARF(1) - PARF(60) : give a parametrization of the uubar-ddbar-ssbar
    flavour mixing for pseudoscalar, vector and tensor mesons.
PARF(61) - PARF(80) : give flavour SU(6) weights for the production
    of a spin 1/2 or spin 3/2 baryon from a given diquark-quark
    combination.
PARF(101) - PARF(108) : first five contain d, u, s, c and b constituent
    masses, as to be used in mass formulae, and should not be changed.
    For t, l and h masses the current values stored in PMAS(6,1) -
    PMAS(8,1) are copied in.
PARF(111), PARF(112) : (D=0.0, 0.11 GeV) constant terms in the mass
    formulae for heavy mesons and baryons, respectively (with diquark
    getting 2/3 of baryon).
PARF(113), PARF(114) : (D=0.16,0.048 GeV) factors which, together with
    Clebsch-Gordan coefficients and quark constituent masses, determine
    the mass splitting due to spin-spin interactions for heavy mesons
    and baryons, respectively. The latter factor is also used for the
    splitting between spin 0 and spin 1 diquarks.

PARF(115) - PARF(118) : (D=0.50, 0.45, 0.55, 0.60 GeV), constant mass
    terms, added to the constituent masses, to get the mass of heavy
    mesons with orbital angular momentum L = 1. The four numbers are for
    pseudovector mesons with quark spin 0, and for scalar, pseudovector
    and tensor mesons with quark spin 1, respectively.
PARF(121),PARF(122) : (D=0.1, 0.2 GeV) constant terms, which are
    subtracted for quark and diquark masses, respectively, in defining
    the allowed phase space in particle decays into partons.
PARF(201) - PARF(1960) : (D=1760*0) relative probabilities for flavour
    production in the MSTJ(15) = 1 option; to be defined by the user
    before any JETSET calls. The index in PARF is of the compressed form
    120 + 80*KTAB1 + 25*KTABS + KTAB3. Here KTAB1 is the old flavour,
    fixed by preceding fragmentation history, while KTAB3 is the new
    flavour, to be selected according to the relevant relative
    probabilities (except for the very last particle, produced when
    joining two jets, where both KTAB1 and KTAB3 are known).
    Only the most frequently appearing quarks/diquarks are defined,
    according to the code 1 = d, 2 = u, 3 = s, 4 = c, 5 = b, 6 = t,
    7 = dd_1, 8 = ud_0, 9 = ud_1, 10 = uu_1, 11 = sd_0, 12 = sd_1,
    13 = su_0, 14 = su_1, 15 = ss_1, 16 = cd_0, 17 = cd_1,
    18 = cu_0, 19 = cu_1, 20 = cs_0, 21 = cs_1, 22 = cc_1.
    These are thus the only possibilities for the new flavour to be
    produced; for an occasional old flavour not on this list, the
    ordinary relative flavour production probabilities will be used.
    Given the initial and final flavour, the intermediate hadron that
    is produced is almost fixed. (Initial and final diquark here
    corresponds to "popcorn" production of mesons intermediate between
    a baryon and an antibaryon). The additional index KTABS gives the
    spin type of this hadron, with
    0 = pseudoscalar meson or Lambda-like spin 1/2 baryon,
    1 = vector meson or Sigma-like spin 1/2 baryon,
    2 = tensor meson or spin 3/2 baryon.
    (Some meson multiplets, not frequently produced, are not accessible
    by this parametrization.)
    Note that some combinations of KTAB1, KTAB3 and KTABS do not
    correspond to a physical particle (a Lambda-like baryon must
    contain three different quark flavours, a Sigma-like one at least
    two), and that the user must see to it that the corresponding
    PARF entries are vanishing. One additional complication exist
    when KTAB3 and KTAB1 denote the same flavour content (normally
    KTAB3 = KTAB1, but for diquarks the spin freedom may give KTAB3 =
    KTAB1 +- 1): then a flavour neutral meson is to be produced,
    and here ddbar, uubar and ssbar states mix (heavier flavour states
    do not, and these are therefore no problem). For these cases
    the ordinary KTAB3 value gives the total probability to produce
    either of the mesons possible, while KTAB3 = 23 gives the
    relative probability to produce the lightest meson state (pi0, rho0,
    a_20), KTAB3 = 24 relative probability for the middle meson
    (eta, omega, f_20), and KTAB3 = 25 relative probability for the
    heaviest one (eta', phi, f'_20). Note that, for simplicity, these
    relative probabilities are assumed the same whether initial and
    final diquark have the same spin or not; the total probability may
    well be assumed different, however.
    As a general comment, the sum of PARF values for a given KTAB1 need
    not be normalized to unity, but rather the program will find the

sum of relevant weights and normalize to that. The same goes for
the KTAB3 = 23 - 25 weights. This makes it straightforward to use
one common setup of PARF values and still switch between different
MSTJ(12) baryon production modes.

VCKM(I,J) : squared matrix elements of the Cabibbo-Kobayashi-Maskawa
    flavour mixing matrix. Are not currently used in JETSET, but
    appear here e.g. for usage in PYTHIA.
    I : up type generation index, i.e. 1 = u, 2 = c, 3 = t and 4 = h.
    J : down type generation index, i.e. 1 = d, 2 = s, 3 = b and 4 = l.

COMMON/LUDAT3/MDCY(500,3),MDME(2000,2),BRAT(2000),KFDP(2000,5)
Purpose: to give access to particle decay data and parameters.
    In particular, MDCY(KC,1) may be used to switch on or off the
    decay of a given particle species, and MDME(IDC,1) to switch
    on or off an individual decay channel of a particle.
    For quarks, leptons and gauge bosons, a number of decay channels
    are included that are not allowed for on-mass-shell particles, see
    MDME(IDC,2) = 102. These channels are not currently used in JETSET,
    but instead find applications in PYTHIA.

MDCY(KC,1) : switch to tell whether a particle may be allowed to decay
    or not. For compressed particle code KC, MDCY(KC,1) = 1 signifies
    that the particle is allowed to decay (if decay information is
    defined below for the particle), whereas MDCY(KC,1) = 0 implies that
    the particle is not allowed to decay. The user is reminded that the
    way to know the KC value is to use the LUCOMP function, e.g.
    to switch off Lambda decay: MDCY(LUCOMP(3122),1) = 0.

MDCY(KC,2) : gives the entry point into the decay channel table for
    compressed particle code KC. Is 0 if no decay channels have
    been defined.

MDCY(KC,3) : gives the total number of decay channels defined for
    compressed particle code KC, independently of whether they have
    been assigned a nonvanishing branching ratio or not. Thus the
    last decay channel is MDCY(KC,2)+MDCY(KC,3)-1.

MDME(IDC,1) : on/off switch for individual decay channel IDC.
    In addition, a channel may be left selectively open; this has some
    special applications in PYTHIA which are not currently used in
    JETSET. Effective branching ratios are automatically recalculated
    for the decay channels left open. If a particle is allowed to decay
    by the MDCY(KC,1) value, at least one channel must be left open
    by the user. A list of decay channels with current IDC numbers
    may be obtained with LULIST(12).
    = -1 : this is a non-standard model decay mode, which by default is
        assumed not to exist. Normally, this option is used for decays
        involving fourth generation or H+- particles.
    = 0 : channel is switched off.
    = 1 : channel is switched on.
    = 2 : channel is switched on for a particle but off for an
        antiparticle. It is also on for a particle its own antiparticle,
        i.e. here it means the same as =1.
    = 3 : channel is switched on for an antiparticle but off for a

particle. It is off for a particle its own antiparticle.
           = 4 : in the production of a pair of equal or charge conjugate
                   resonances in PYTHIA, say H0 -> W+ W-, either one of the
                   resonances is allowed to decay according to this group of
                   channels, but not both. If the two particles of the pair
                   are different, the channel is on.
                   Within JETSET, this option only means that the channel is
                   switched off.
           = 5 : as =4, but an independent group of channels, such that in
                   a pair of equal or charge conjugate resonances the decay of
                   either resonance may be specified independently. If the two
                   particles in the pair are different, the channel is off.
                   Within JETSET, this option only means that the channel is
                   switched off.
           Remark: all the options above may be freely mixed. The difference,
                   for those cases where both make sense, between using values
                   2 and 3 and using 4 and 5 is that the latter automatically
                   include charge conjugate states, e.g. H0 -> W+ W- ->
                   e+ nue d ubar or dbar u e- nuebar, but the former only one
                   of them. In calculations of the joint branching ratio, this
                   makes a factor 2 difference.

     MDME(IDC,2) : information on special matrix element treatment for decay
           channel IDC. In addition to the outline below, special rules apply
           for the order in which decay products should be given, so that
           matrix elements and colour flow is properly treated.
           = 0 : no special matrix element treatment; partons and particles
                   are copied directly to the event record, with momentum
                   distributed according to phase space.
           = 1 : omega and phi decays into three pions.
           = 2 : pi0 or eta Dalitz decay to gamma e+ e-.
           = 3 : used for vector meson decays into two pseudoscalars, to signal
                   that, in the decay chain PS_0 -> PS_1 + V -> PS_1 + PS_2 + PS_3
                   (PS pseudoscalar, V vector), the momentum of PS_2 and PS_3 have
                   a cos^2(theta_02) distribution in the rest frame of V, and that,
                   in the decay chain PS_0 -> gamma + V -> gamma + PS_2 + PS_3, the
                   momentum of PS_2 and PS_3 have a sin^2(theta_02) distribution in
                   the rest frame of V.
           = 4 : decay of a spin 1 "onium" resonance to three gluons or to a
                   photon and two gluons. The gluons may subsequently develop a
                   shower if MSTJ(23) = 1.
           = 11 : phase space production of hadrons from the quarks available.
           = 12 : as =11, but for onia resonances, with the option of modifying
                   the multiplicity distribution separately.
           = 13 : as =11, but at least three hadrons to be produced (useful
                   when the two-body decays are given explicitly).
           = 14 : as =11, but at least four hadrons to be produced.
           = 15 : as =11, but at least five hadrons to be produced.
           = 22 - 30 : phase space production of hadrons from the quarks
                   available, with the multiplicity fixed to be MDME(IDC,2)-20,
                   i.e. 2 - 10.
           = 31 : two or more quarks and particles are distributed according
                   to phase space. If three or more products, the last product is
                   a spectator quark, i.e. sitting at rest with respect to the
                   decaying hadron.

= 32 : a qqbar or gg pair, distributed according to phase space
       (in angle), and allowed to develop a shower if MSTJ(23) = 1.
= 33 : a triplet qXqbar, where X is either a gluon or a colour
       singlet particle, the final particle (qbar) is assumed to
       sit at rest with respect to the decaying hadron, and the
       two first particles (qX) are allowed to develop a shower if
       MSTJ(23) = 1.
= 41 : weak decay, where particles are distributed according to
       phase space, multiplied by a factor from the expected shape of
       the momentum spectrum of the direct product of the weak decay
       (the tau neutrino in tau decay).
= 42 : weak decay matrix element for quarks and leptons. Products
       may be given either in terms of quarks or hadrons, or leptons
       for some channels. If the spectator system is given in terms
       of quarks, it is assumed to collapse into one particle from
       the onset. If the virtual W decays into quarks, these quarks
       are converted to particles, according to phase space in the
       W rest frame, as in =11. Is intended for tau, charm and
       bottom.
= 43 : as =42, but if the W decays into quarks, these will either
       appear as jets or, for small masses, collapse into a one- or
       two-body system.
= 44 : weak decay matrix element for quarks and leptons, where the
       spectator system may collapse into one particle for a small
       invariant mass. If the first two decay products are a qq'bar
       pair, they may develop a parton shower, if MSTJ(23) = 1.
       Is intended for top and beyond, but largely superseded by
       the following option.
= 45 : weak decay q -> W + q' or l -> W + l', where the W is
       registered as a decay product and subsequently treated with
       MDME = 46. To distinguish from ordinary on-shell W:s, code
       KF = +-89 is used. The virtual W mass is selected according to
       the standard weak decay matrix element, times the W propagator
       (for MSTJ(25) = 1). There may be two or three decay products;
       if a third this is a spectator taken to sit at rest. The
       spectator system may collapse into one particle. Is intended
       for top and beyond.
= 46 : W (KF = 89) decay into qq'bar or lnu_l according to relative
       probabilities given by couplings (as stored in the BRAT vector)
       times a dynamical phase space factor given by the current W
       mass. In the decay, the correct V-A angular distribution is
       generated if the W origin is known (heavy quark or lepton).
       This is therefore the second step of a decay with MDME = 45.
       A qq'bar pair may subsequently develop a shower.
= 84 - 88 : map the decay of this particle onto the generic
       c, b, t, l or h decay modes defined for KC = 84 - 88.
= 101 : this is not a proper decay channel, but only to be
       considered as a continuation line for the decay product listing
       of the immediately preceding channel. Since the KFDP array can
       contain five decay products per channel, with this code it is
       possible to define channels with up to ten decay products. It is
       not allowed to have several continuation lines after each other.
= 102 : this is not a proper decay channel for an on-mass-shell
       (or nearly so) decaying particle, and is therefore assigned
       branching ratio 0. As an off-mass-shell particle, this decay

mode is allowed, however. By including this channel among the
        others, the switches MDME(IDC,1) may be used to allow or
        forbid these channels in hard processes, with cross-sections
        to be calculated separately. As an example, gamma -> u ubar
        is not possible for a massless photon, but is an allowed
        channel in e+e- annihilation.

BRAT(IDC) : give branching ratios for the different decay channels.
    In principle, the sum of branching ratios for a given particle
    should be unity. Since the program anyway has to calculate the
    sum of branching ratios left open by the MDME(IDC,1) values
    and normalize to that, the user need not explicitly ensure this
    normalization, however. (Warnings are printed in LUUPDA(2) calls
    if the sum is not unity, but this is entirely intended as a help
    for finding user mistypings.) For decay channels with MDME(IDC,2)
    > 80 the BRAT values are dummy.

KFDP(IDC,J) : contain the decay products in the different channels,
    with five positions J = 1 through 5 reserved for each channel IDC.
    The decay products are given following the standard KF code for
    jets and particles, with 0 for trailing empty positions. Note
    that the MDME(IDC+1,2) = 101 option allows the user to double the
    maximum number of decay product in a given channel from 5 to 10,
    with the five latter products stored KFDP(IDC+1,J).

COMMON/LUDAT4/CHAF(500)
CHARACTER CHAF*8
Purpose: to give access to character type variables.

CHAF : particle names (excluding charge) according to KC code.

2.9. Miscellaneous Comments

The previous sections have dealt with the subroutine options and
variables one at a time. This is certainly important, but for a full
use of the capabilities of the Lund Monte Carlo, it is also necessary
to understand how to make different pieces work together. This is
something that can not be explained fully in a manual, but must also
be learnt by trial and error. This section contains some examples of
relationships between subroutines, commonblocks and parameters.
It also contains comments on issues that did not fit in naturally
anywhere else, but still might be useful to have on record.

Very often, the output of the program is to be fed into a subsequent
detector simulation program. It therefore becomes necessary to set up
an interface between the LUJETS commonblock and the detector model.
Preferrably this should be done via the HEPEVT standard commonblock,
see section 2.11, but sometimes this may not be convenient.
If a LUEDIT(2) call is made, the remaining entries exactly correspond
to those an ideal detector could see: all non-decayed particles, with
the exception of neutrinos. The translation of momenta should be
trivial (if need be, a LUROBO call can be made to rotate the "preferred"
z direction to whatever is the longitudinal direction of the detector),
and so should the translation of particle codes. In particular, if the

detector simulation program also uses the standard Particle Data Group codes, no conversion at all is needed. The problem then is to select which particles are allowed to decay, and how decay vertex information should be used.

Several switches regulate which particles are allowed to decay. First, the master switch MSTJ(21) can be used to switch on/off all decays (and it also contains a choice of how fragmentation should be interfaced). Second, a particle must have decay modes defined for it, i.e. the corresponding MDCY(KC,2) and MDCY(KC,3) entries must be nonzero for compressed code KC = LUCOMP(KF). This is true for all colour neutral particles except the neutrinos, the photon, the proton and the neutron. (This statement is actually not fully correct, since irrelevant "decay modes" with MDME(IDC,2) = 102 exist in some cases.) Third, the individual switch in MDCY(KC,1) must be on. Of all the particles with decay modes defined, only mu+-, pi+-, K+- and K_L0 are by default considered stable.

Finally, if MSTJ(22) does not have its default value 1, checks are also made on the lifetime of a particle before it is allowed to decay. In the simplest alternative, MSTJ(22) = 2, the comparison is based on the average lifetime, or rather c*tau, measured in mm. Thus if the limit PARJ(71) is (the default) 10 mm, then decays of K_S0, Lambda, Sigma-, Sigma+, Xi-, Xi0 and Omega- are all switched off, but charm and bottom still decay. No c*tau values below 1 micron are defined. With the two options MSTJ(22) = 3 or 4, a spherical or cylindrical volume is defined around the origin, and all decays taking place inside this volume are ignored.

Whenever a particle is in principle allowed to decay, i.e. MSTJ(21) and MDCY on, an invariant lifetime is selected once and for all and stored in V(I,5). The K(I,1) is then also changed to 4. For MSTJ(22) = 1, such a particle will also decay, but else it could remain in the event record. It is then possible, at a later stage, to expand the volume inside which decays are allowed, and do a new LUEXEC call to have particles fulfilling the new conditions (but not the old) decay. As a further option, the K(I,1) code may be put to 5, signalling that the particle will definitely decay in the next LUEXEC call, at the vertex position given (by the user) in the V vector.

This then allows the Lund decay routines to be used inside a detector simulation program, as follows. For a particle which did not decay before entering the detector, its point of decay is still well defined (in the absence of deflections by electric or magnetic fields): V'(j) = V(I,j) + V(I,5)*P(I,j)/P(I,5), j = 1 - 4. If it interacts before that point, the detector simulation program is left to handle things. If not, the V vector is updated according to the formula above, K(I,1) is set to 5, and LUEXEC is called, to give a set of decay products, that can again be tracked.

A further possibility is to force particles to decay into specific decay channels; this may be particularly interesting for charm or bottom physics. The choice of channels left open is determined by the values of the switches MDME(IDC,1) for decay channel IDC (use LULIST(12) to obtain the full listing). One or several channels may be left open;

in the latter case effective branching ratios are automatically
recalculated without the need for user intervention. It is also
possible to differentiate between which channels are left open for
particles and which for antiparticles. Lifetimes are not affected by
the exclusion of some decay channels. Note that, whereas forced decays
can enhance the efficiency for several kinds of studies, it can
also introduce unexpected biases, in particular when events may contain
several particles with forced decays.

A nontrivial question is to know which parameter values to use. The
default values stored in the program are based on comparisons with e+e-
data at around 30 GeV, using a parton shower picture followed by string
fragmentation. If fragmentation is indeed an universal phenomenon, as
we would like to think, then the same parameters should also apply at
other energies and in other processes. Note, however, that the choice
of parameters is intertwined with the choice of perturbative QCD
description. If instead matrix elements are used, a best fit to 30 GeV
data would require the values PARJ(21) = 0.40, PARJ(41) = 1.0 and
PARJ(42) = 0.7. With matrix elements one does not expect an energy
independence of the parameters, since the effective minimum invariant
mass cutoff is then energy dependent, i.e. so is the amount of soft
gluon emission effects lumped together with the fragmentation
parameters. A mismatch in the perturbative QCD treatment could also lead
to small differences between different processes.

It is often said that the string fragmentation model contains a wealth
of parameters. This is certainly true, but it must be remembered that
most of these deal with flavour properties, and to a large extent
factorize from the treatment of the general event shape. In a fit to
the latter it is therefore usually enough to consider the parameters of
the perturbative QCD treatment, like Lambda in alpha_strong and a
shower cutoff m_min (or alpha_strong itself and y_min, if matrix
elements are used), the a and b parameter of the Lund symmetric
fragmentation function (PARJ(41) and PARJ(42)) and the width of the
transverse momentum distribution (sigma = PARJ(21)). In addition, the
a and b parameters are very strongly correlated by the requirement
of having the correct average multiplicity, such that in a typical
chi^2 plot, the allowed region corresponds to a very narrow but very
long valley, stretched diagonally from small (a,b) pairs to large ones.
As to the flavour parameters, these are certainly many more, but most
of them are understood qualitiatively within one single framework, that
of tunneling pair production of flavours.

Since the use of independent fragmentation has fallen in disrespect, it
should be pointed out that the default parameters here are not
particularly well tuned to the data. This especially applies if one,
in addition to asking for independent fragmentation, also asks for
another setup of fragmentation functions, i.e. other than the standard
Lund symmetric one. In particular, note that most fits to the popular
Peterson et al. (SLAC) heavy flavour fragmentation function are based
on the actual observed spectrum. In a Monte Carlo simulation, one must
then start out with something harder, to compensate for the energy lost
by initial state photon radiation and gluon bremsstrahlung. Since
independent fragmentation is not collinear safe (i.e, the emission of
a collinear gluon changes the properties of the final event), the tuning

is strongly dependent on the perturbative QCD treatment chosen. All the parameters needed for a tuning of independent fragmentation are available, however.

The masses of most frequently used particles are taken from tables. For some rare charm and bottom hadrons, and for heavier flavour hadrons, this would be unwieldy, and instead mass formulae are used, based on the quark content. For the known quarks d, u, s, c and b, the masses used for this purpose are actually the ones stored in positions 101 - 105 in the PARF vector, rather than the ones found in PMAS. This means that the PMAS masses can be freely changed by the user, to modify the masses that appear in the event record, without courting disaster elsewhere (since mass formulae typically contain 1/m terms from spin-spin splittings, is is necessary to have the nonzero "constituent" masses here). Thus a user should never touch the mass values stored in PARF. For the heavier flavours top, low and high, the current PMAS values are always used. For these flavours, the only individually defined hadrons are the flavour neutral eta, Theta and chi_2 states. A complete change of top mass in the program thus requires changing PMAS(6,1), PMAS(LUCOMP(661),1), PMAS(LUCOMP(663),1) and PMAS(LUCOMP(665),1). Since the latter heavy flavour diagonal states are not normally produced in fragmentation, it would be no disaster to forget changing their masses.

The masses of several responances, like rho, K* and Delta, are by default distributed according to simle Breit-Wigner shapes, suitably truncated, with the truncation chosen so that no problems are encountered in decay chains. It should be emphasized that such a simple approach may be a poor approximation, and should never be used for detailed studies of resonance production. It will, however, give a first approximation to the smearing caused by variable resonance masses.

Another option (this one by default off) with a similar level of crudity is the simulation of Bose-Einstein effects. Here the detailed physics is not that well understood, see e.g. the review [Lor88]. What is offered is an algorith, more than just a parametrization (since very specific assumptions and choices have been made), and yet less than a true model (since the underlying physics picture is rather fuzzy). The fragmentation is allowed to proceed as usual, and so is the decay of short-lived particles like rho. Then pairs of identical particles, pi+ say, are considered one by one. The Q value is evaluated, and a shifted (smaller) Q' is found such that the (infinite statistics) ratio of shifted to unshifted Q distributions is given by the requested parametrization in MSTJ(51). (In fact, the distribution dips slightly below unity at Q values outside the Bose enhancement region, from conservation of total multiplicity.) This can be translated into an effective shift of the momenta of the two particles, if one uses as extra constraint that the total three-momentum of each pair be conserved in the CM frame of the event. Only after all pairwise momentum shifts have been evaluated with respect to the original momenta are these momenta actually shifted, for each particle by the sum of evaluated shifts. The total energy of the event is slightly reduced in the process, which is compensated by an overall rescaling of all CM frame momentum vectors. Finally, the decay chain is resumed with

more long-lived particles like pi0.

Two comments can be made on the approach adopted. First,
Bose-Einstein effects are here interpreted almost as a
classical force acting on the "final state", rather than
as a quantum mechanical phenomenon on the production amplitude. This
is not a credo, but just an ansatz to make things manageable. In
particular, if an event weight were introduced by the product of all
weights for individual pairs, at first sight a more correct procedure,
events with higher multiplicities would be weighted up by quite
significant amounts. Indeed, the effects are so large that they could
not be simply compensated by a modest change of the standard
fragmentation parameters. Second, since only pairwise interactions
are considered, the effects associated with three or more nearby
particles tend to get overestimated (For n identical particles with
the same momentum and Lambda = 1, the correct weight is n! but the
actually simulated one more like 2^(n(n-1)/2).) Thus the input
lambda may have to be chosen smaller than what one wants to get out.
This option should therefore be used with caution, and only as a
first approximation to what Bose-Einstein effects can mean.

The commonblock LUJETS has expanded with time, and can now house 4000
entries. This figure may seem ridiculously large, but actually the
previous limit of 2000 was often reached in studies of high-p_T
processes at the SSC. The reason for this is that the event record
contains not only the final particles, but also all intermediate partons
and hadrons, which subsequenty showered, fragmented or decayed. Included
are also a wealth of photons coming from pi0 decays; the simplest way
of reducing the size of the event record is actually to switch off
pi0 decays by MDCY(LUCOMP(111),1) = 0. Also note that some routines,
like LUCLUS and LUCELL, use memory after the event record proper as a
working area. Still, to change the size of the commonblock, upwards or
downwards, is easy: just do a global substitute in the commonblock and
change the MSTU(4) value to the new number. If more than 10000 lines are
to be used, the packing of colour information should also be changed,
see MSTU(5).

The program contains space so that additional new particles may be
introduced. Although not completely trivial, this should not be
beyond the ability of an ordinary user. Basically, three steps are
involved. First, a mechanism of production has to be introduced.
This production may well take place in an external program, like
PYTHIA or some user-written correspondence, where matrix elements are
used to select the hard process. In this case the new particle already
exists in the LUJETS commonblock when JETSET is called. A new particle,
meson, baryon or glueball, may also be a part of the fragmentation
process, in which case LUKFDI would have to be suitably modified.
The particle might also appear as a decay product from some already
existing particle, and then the decay data in /LUDAT3/ would have
to be expanded; conceivably also LUDECY would be affected.
The second step is to teach to program to recognize the new particle.
IF a KF code in the range 41 to 80 is used, this is automatically taken
care of, and in particular the compressed code KC coincides with KF.
If a whole sequence of particles is to be introduced, with KF codes
paralleling that of ordinary mesons/baryons (a supersymmetric "meson"

multiplet, made of a squark plus an antiquark, say), then LUCOMP must
be modified to include a mapping from these KF values to currently
unused KC ones, like the range 401 - 500. It is the presence of such a
mapping that the program uses to accept a given KF code as bona fide.
The third and final step is to define the properties of this new
particle. Thus particle charge information must be given in KCHG,
mass, width and lifetime in PMAS, particle name in CHAF, and decay
data in the MDCY, MDME, BRAT and KFDP arrays. This process is most
conveniently carried out by using LUUPDA(1) to produce a table of
particle data, which can then be modified by the user and read back in
with LUUPDA(2). Note that the particle data is to be introduced for
the compressed code KC, not for KF proper.

_____

2.10. Examples

What every user has to know something about is the commonblock LUJETS
        COMMON/LUJETS/N,K(4000,5),P(4000,5),V(4000,5)
A short summary follows; for details see section 2.2. The complete
event record is stored in LUJETS. This normally includes the original
jets, the particles these fragment into and the subsequent decay
chains. The number N gives the number of lines (1 through N) in the K,
P and V matrices that are actually used to store the current event.
For line I, K(I,1) gives information about current status. In
particular, anything which has 1 <= K(I,1) <= 10 represents a particle
or parton that is either considered stable or has not yet been treated,
whereas K(I,1) >= 11 corresponds to partons that have fragmented or
branched, particles that have decayed and an assortment of special
purpose lines. K(I,2) gives a code for what parton or particle we are
dealing with, e.g. 1 = d, 2 = u, 11 = e-, 12 = nu_e, 21 = g, 22 = gamma,
2103 = ud_1 diquark, 111 = pi0, 211 = pi+, 213 = rho+, 2212 = p.
Antiparticles, where existing, are given by the negative number,
-1 = dbar, -211 = pi-. This code is often referred to as KF code,
and is described in section 2.1. Codes K(I,3) - K(I,5) contain event
history information, line number where mother are stored, line number
range of daughters or colour flow information. The P vector contains
momentum information, with P(I,1), P(I,2) and P(I,3) giving the
three-momentum, P(I,4) the energy and P(I,5) the mass, in GeV/c,
GeV and GeV/c^2, respectively. The components V(I,1), V(I,2) and
V(I,3) give the production vertex position of the particle, and
V(I,4) the production time, all with respect to a primary vertex
situated in the origin at time 0. The space components are given in
mm, and the time one in mm/c = 3.33*10^-12 s. The final component
V(I,5) gives the invariant lifetime of a particle, again in mm/c.
Neglecting bending in a magnetic field etc., the decay vertex V' of
the particle is then given by V'(j) = V(I,j) + V(I,5)*P(I,j)/P(I,5),
j = 1 - 4. For a stable particle V(I,5) = 0. For an unstable,
V(I,5) is selected even if the decay is not actually carried out
because it takes place outside the allowed region, see section 2.9.

A 10 GeV u quark jet going out along the +z axis is generated with
        CALL LU1ENT(0,2,10.,0.,0.)
Note that such a single jet is not required to conserve energy,
momentum or flavour. In the generation scheme, particles with
negative p_z are produced as well, but these are automatically

rejected unless MSTJ(3) = -1. While frequently used in former days, the one-jet generation option is not of much current interest.

In e.g. a leptoproduction event a typical situation could be a u quark going out in the +z direction and a ud_0 target remnant essentially at rest. The simplest procedure is probably to treat it in the CM frame and boost it to the lab frame afterwards. Hence, if the CM energy is 20 GeV and the boost beta = 0.996 (corresponding to x_B = 0.045)

```
      CALL LU2ENT(0,2,2101,20.)
      CALL LUROBO(0.,0.,0.,0.,0.996)
```

The jets could of course also be defined and allowed to fragment in the lab frame with

```
      CALL LU1ENT(-1,2,223.15,0.,0.)
      CALL LU1ENT(2,12,0.6837,3.1416,0.)
      CALL LUEXEC
```

Note here that the target diquark is required to move in the backwards direction with (E-p_z) = m_proton*(1-x_B) to obtain the correct invariant mass for the system. This is, however, only an artefact of using a fixed diquark mass to represent a varying target remnant mass, and is of no importance for the fragmentation. If one wants a nicer-looking event record, it is possible to use the following

```
      CALL LU1ENT(-1,2,223.15,0.,0.)
      MSTU(10)=1
      P(2,5)=0.938*(1.-0.045)
      CALL LU1ENT(2,2101,0.,0.,0.)
      MSTU(10)=2
      CALL LUEXEC
```

A 30 GeV uubarg event with E_u = 8 GeV and E_ubar = 14 GeV is simulated with

```
      CALL LU3ENT(0,2,21,-1,30.,2.*8./30.,2.*14./30.)
```

The event will be given in a standard orientation with the u quark along the +z axis and the ubar in the -z, +x quadrant. Note that the flavours of the three partons have to be given in the order they are found along a string, if string fragmentation options are to work. Also note that, for three-jet events, and particularly four-jet ones, not all setups of kinematical variables x lie within the kinematically allowed regions of phase space.

All commonblock variables can obviously be changed by including the corresponding commonblock in the user-written main program. Alternatively, the routine LUGIVE can be used to feed in values, with some additional checks on array bounds then performed. A call

```
      CALL LUGIVE('MSTJ(21)=3;PMAS(C663,1)=89.;CHAF(401)=funnyino;'//
     &'PMAS(21,4)=')
```

will thus change the value of MSTJ(21) to 3, the value of PMAS(LUCOMP(663),1) = PMAS(136,1) to 89., the value of CHAF(401) to 'funnyino', and give the current value of PMAS(21,4). Since old and new values of parameters changed are written to output, this may offer a convenient way of documenting non-default values used in a given run. On the other hand, if a variable is changed back and forth frequently, the resulting voluminous output may be undesirable, and a direct usage of the commonblocks is then to be recommended (the output can also be switched off, see MSTU(13)).

A general rule of thumb is that none of the physics routines
(LUSTRF, LUINDF, LUDECY, etc.) should ever be called directly, but
only via LUEXEC. This routine may be called repeatedly for one single
event. At each call only those entries that are allowed to fragment
or decay, and have not yet done so, are treated. Thus

```
      CALL LU2ENT(1,1,-1,20.)        ! fill 2 jets without fragmenting
      MSTJ(1)=0                      ! inhibit jet fragmentation
      MSTJ(21)=0                     ! inhibit particle decay
      MDCY(LUCOMP(111),1)=0          ! inhibit pi0 decay
      CALL LUEXEC                    ! will not do anything
      MSTJ(1)=1                      !
      CALL LUEXEC                    ! jets will fragment, but no decays
      MSTJ(21)=2                     !
      CALL LUEXEC                    ! particles decay, except pi0
      CALL LUEXEC                    ! nothing new can happen
      MDCY(LUCOMP(111),1)=1          !
      CALL LUEXEC                    ! pi0:s decay
```

A partial exception to the rule above is LUSHOW. Its main application is
for internal use by LUEEVT and LUDECY, and by other Lund family programs
like PYTHIA, but it can also be directly called by the user. Note that a
special format for storing colour flow information in K(I,4) and K(I,5)
must then be used. For simple cases, the LU2ENT can be made to take care
of that automatically, by calling with the first argument negative.

```
      CALL LU2ENT(-1,1,-2,40.)       ! store dubar with colour flow
      CALL LUSHOW(1,2,40.)           ! shower partons
      CALL LUEXEC                    ! subsequent fragmentation/decay
```

It is always good practice to list one or a few events during a run to
check that the program is working as intended. With

```
      CALL LULIST(1)
```

all particles will be listed and in addition total charge, momentum and
energy of stable entries will be given. For string fragmentation these
quantities should be conserved exactly (up to machine precision errors),
and the same goes when running independent fragmentation with one of
the momentum conservation options. LULIST(1) gives a format that
comfortably fits on an 80 column screen, at the price of not giving
the complete story. With LULIST(2) a more extensive listing is obtained,
and LULIST(3) also gives vertex information. Further options are
available, like LULIST(12), which gives a list of particle data.

An event, as stored in the LUJETS commonblock, will contain the original
jets and the whole decay chain, i.e. also particles which subsequently
decayed. If parton showers are used, the amount of parton information is
also considerable: first the on-shell partons before showers have been
considered, then a KS = 22 line with total energy of the showering
subsystem, after that the complete shower history treelike structure,
starting off with the same initial partons (now off-shell), and finally
the endproducts of the shower rearranged along the string directions.
This detailed record is useful in many connections, but if one only
wants to retain the final particles, superfluous information may be
removed with LUEDIT. Thus e.g.

```
      CALL LUEDIT(2)
```

will leave you with the final charged and neutral particles, except
for neutrinos.

The information in LUJETS may be used directly to study an event. Some useful additional quantities derived from these, such as charge and rapidity, may easily be found via the KLU and PLU functions. Thus electric charge = PLU(I,6) (as integer, 3*charge = KLU(I,6)) and true rapidity y with respect to the z axis = PLU(I,17).

Event analysis routines for properties of the event as-a-whole include sphericity, thrust, cluster (2 different algorithms), jet masses and Fox-Wolfram moments. These routines may be called directly after the event generation
      CALL LU2ENT(0,5,-5,40.)
      CALL LUTHRU(THR,OBL)
and then all stable, final particles (except neutrinos) are used in the analysis. This may be changed with the MSTu(41), or by explicitly using LUEDIT, or by hand setting K(I,1) values for unwanted particles before the analysis.

A number of utility (MSTU, PARU) and physics (MSTJ, PARJ) switches and parameters are available in commonblock LUDAT1. All of these have sensible default values. Particle data is stored in commonblocks LUDAT2, LUDAT3 and LUDAT4. Note that the data in the arrays KCHG, PMAS, MDCY and CHAF is not stored by KF code, but by the compressed code KC. This code is not to be learnt by heart, but instead accessed via the conversion function LUCOMP, KC = LUCOMP(KF).

In the particle tables, the following particles are considered stable: the photon, e+-, mu+-, pi+-, K+-, K_L0, p, pbar, n, nbar and all the neutrinos. It is, however, always possible to inhibit the decay of any given particle by putting the corresponding MDCY value zero or negative, e.g. MDCY(LUCOMP(310),1) = 0 makes K_S0 and MDCY(LUCOMP(3122),1) = 0 Lambda stable. It is also possible to select stability based on the average lifetime (see MSTJ(22)), or based on whether the decay takes place within a given spherical or cylindrical volume around the origin. This is described in more detail in section 2.9.

The Field-Feynman jet model [Fie78] is available in the program by changing the following values: MSTJ(1) = 2 (independent fragmentation), MSTJ(3) = -1 (retain particles with p_z < 0; is not mandatory), MSTJ(11) = 2 (choice of longitudinal fragmentation function, with the a parameter stored in PARJ(51) - PARJ(53)), MSTJ(12) = 0 (no baryon production), MSTJ(13) = 1 (give endpoint quarks p_T as quarks created in the field), MSTJ(24) = 0 (no mass broadening of resonances), PARJ(2)=0.5 (s/u ratio for the production of new qqbar pairs), PARJ(11) = PARJ(12) = 0.5 (probability for mesons to have spin 1) and PARJ(21) = 0.35 (width of Gaussian transverse momentum distribution). In addition only d, u and s single quark jets may be generated following the FF recipe. Today the FF "standard jet" concept is probably dead and buried, so the numbers above should more be taken as an example of the flexibility of the program, than as something to apply in practice.

A wide range of independent fragmentation options are implemented, to be accessed with the master switch MSTJ(1) = 2. In particular, with MSTJ(2) = 1 a gluon jet is assumed to fragment like a

random d, dbar, u, ubar, s or sbar jet, while with MSTJ(2) = 3
the gluon is split into a ddbar, uubar or ssbar pair of jets
sharing the energy according to the Altarelli-Parisi splitting
function. Whereas energy, momentum and flavour is not explicitly
conserved in independent fragmentation, a number of options are
available in MSTJ(3) to ensure this "post facto", e.g. MSTJ(3) = 1
will boost the event to ensure momentum conservation and then
(in the CM frame) rescale momenta by a common factor to obtain
energy conservation, whereas MSTJ(3)=4 rather uses a method of
stretching the jets in longitudinal momentum along the respective
jet axis to keep angles between jets fixed.

2.11. Translation to/from Standard Commonblock

Recently a set of commonblocks has been developed and agreed on
within the framework of the 1989 LEP physics study, see [QEG89].
This standard defines an event record structure which should make
the interfacing of different event generators much more simple.

It would be a major work to rewrite JETSET to agree with this
standard event record structure. More importantly, the standard
only covers quantities which can be defined unambigously, i.e.
which are independent of the particular program used. There are
thus no provisions for the need for colour flow information in
models based on string fragmentation, etc., so the standard
commonblocks would anyway have to be supplemented with additional
event information. For the moment, the approach adopted is therefore
to retain the LUJETS event record, but supply a routine LUHEPC which
can convert to or from the standard event record. Due to somewhat
different content in the two records, some ambiguities do exist in
the translation procedure. LUHEPC has therefore to be used with
some judgment.

In this section, the new standard event structure is first presented,
i.e. the most important points in [QEG89] are recapitulated.
Thereafter the conversion routine is described, with particular
attention to ambiguities and limitations.

The standard event record is stored in two commonblocks. The second
of these is specifically intended for spin information. Since JETSET
never (explicitly) makes use of spin information, this latter
commonblock is never addressed.

In order to make the components of the standard more distinguishable
in user programs, the three characters HEP (for High Energy Physics)
have been chosen to be a part of all names.

```
 PARAMETER (NMXHEP=2000)
 COMMON/HEPEVT/NEVHEP,NHEP,ISTHEP(NMXHEP),IDHEP(NMXHEP),
&JMOHEP(2,NMXHEP),JDAHEP(2,NMXHEP),PHEP(5,NMXHEP),VHEP(4,NMXHEP)
```
Purpose: to contain an event record in a Monte Carlo-independent
     format.

NMXHEP: maximum numbers of entries (partons/particles) that can be

stored in the commonblock. The default value of 2000 can be changed
via the parameter construction. In the translation, it is
checked that this value is not exceeded.

NEVHEP: is normally the event number, but may have special meaning,
according to description below.
> 0 : event number, sequentially increased by 1 for each call to
the main event generation routine, starting with 1 for the
first event generated.
= 0 : for a program which does not keep track of event numbers,
like JETSET.
= -1 : special initialization record; not used by JETSET.
= -2 : special final record; not used by JETSET.

NHEP: the actual number of entries stored in current event. These are
found in the first NHEP positions of the respective arrays below.
Index IHEP, 1 <= IHEP <= NHEP, is below used to denote a given
entry.

ISTHEP(IHEP): status code for entry IHEP, with following meanings:
= 0 : null entry.
= 1 : an existing entry, which has not decayed or fragmented.
This is the main class of entries which represents the
"final state" given by the generator.
= 2 : an entry which has decayed or fragmented and therefore
is not appearing in the final state, but is retained for
event history information.
= 3 : a documentation line, defined separately from the event
history. This could include the two incoming reacting
particles, etc.
= 4 - 10 : undefined, but reserved for future standards.
= 11 - 200 : at the disposal of each model builder for constructs
specific to his program, but equivalent to a null line in the
context of any other program.
= 201 - : at the disposal of users, in particular for event tracking
in the detector.

IDHEP(IHEP) : particle identity, according to the Particle Data Group
standard. The four additional codes 91 - 94 have been introduced
to make the event history more legible, see section 2.1 and the
MSTU(16) description.

JMOHEP(1,IHEP) : pointer to the position where the mother is stored.
The value is 0 for initial entries.

JMOHEP(2,IHEP) : pointer to position of second mother. Normally only
one mother exist, in which case the value 0 is to be used.
In JETSET, entries with codes 91 - 94 are the only ones to have
two mothers. The flavour contents of these object, as well as
details of momentum sharing, have to be found by looking at the
mother partons, i.e. the two partons in positions JMOHEP(1,IHEP)
and JMOHEP(2,IHEP) for a cluster or a shower system, and the range
JMOHEP(1,IHEP) - JMOHEP(2,IHEP) for a string or an independent
fragmentation parton system.

JDAHEP(1,IHEP) : pointer to the position of the first daughter. If an
    entry has not decayed, this is 0.

JDAHEP(2,IHEP) : pointer to the position of the last daughter. If an
    entry has not decayed, this is 0. It is assumed that daughters are
    stored sequentially, so that the whole range JDAHEP(1,IHEP) -
    JDAHEP(2,IHEP) contains daughters. This should be done also when
    only one daughter is present, like in K^0 -> K_S^0 decays.
    Normally daughters are stored after mothers, but in backwards
    evolution of initial state radiation the opposite may appear,
    i.e. that mothers are found below the daughters they branch into.
    Also, the two daughters need then not appear one after the other,
    but may be separated in the event record.

PHEP(1,IHEP) : momentum in the x direction, in GeV/c.

PHEP(2,IHEP) : momentum in the y direction, in GeV/c.

PHEP(3,IHEP) : momentum in the z direction, in GeV/c.

PHEP(4,IHEP) : energy, in GeV.

PHEP(5,IHEP) : mass, in GeV/c**2. For spacelike partons, it is allowed
    to use a negative mass, according to PHEP(5,IHEP) = -sqrt(-m**2).

VHEP(1,IHEP) : production vertex x position, in mm.

VHEP(2,IHEP) : production vertex y position, in mm.

VHEP(3,IHEP) : production vertex z position, in mm.

VHEP(4,IHEP) : production time, in mm/c (= 3.33*10**(-12) s).

This completes the brief description of the standard. In JETSET, the
routine LUHEPC is provided as an interface.

SUBROUTINE LUHEPC(MCONV)
Purpose: to convert between the LUJETS event record and the
    HEPEVT event record.
MCONV : direction of conversion.
    = 1 : translate the current LUJETS record into the HEPEVT one,
        while leaving the original LUJETS one unaffected.
    = 2 : translate the current HEPEVT record into the LUJETS one,
        while leaving the original HEPEVT one unaffected.

The conversion of momenta is trival: it is just a matter of exchanging
the order of the indices. The vertex information is but little more
complicated; the extra fifth component present in LUJETS can be easily
reconstructed from other information for particles which have decayed.
(Some of the advanced features made possible by this component, like
the possibility to consider decays within expanding spatial volumes in
subsequent LUEXEC calls, can not be used if the record is translated
back and forth, however.) Also, the particle codes K(I,2) and IDHEP(I)
are identical, since they are both based on the PDG codes.

The remaining, nontrivial areas deal with the status codes and the event history. In moving from LUJETS to HEPEVT, information on colour flow is lost. On the other hand, the position of a second mother, if any, has to be found; this only affects lines with K(I,2) = 91 - 94. Also, for lines with K(I,1) = 13 or 14, the daughter pointers have to be found. By and large, however, the translation from LUJETS to HEPEVT should cause little problem, and there should never be any need for user intervention. (We assume that JETSET is run with the default MSTU(16) = 1, else some discrepancies compared to the proposed standard event history description will be present.)

In moving from HEPEVT to LUJETS, information on a second mother is lost. Any codes IDHEP(I) not equal to 1, 2 or 3 are translated into K(I,1) = 0, and so all entries with K(I,1) >= 30 are effectively lost in a translation back and forth. All entries with IDHEP(I) = 2 are translated into K(I,1) = 11, and so entries of type K(I,1) = 12, 13 or 14 or 15 are never found. There is thus no colour flow information available for partons which have fragmented. For partons with IDHEP(I) = 1, i.e. which have not fragmented, an attempt is made to subdivide the partonic system into colour singlets, as required for subsequent string fragmentation. To this end, it is assumed that partons are stored sequentially along strings. Normally, a string would then start at a q (qbar) or qbarqbar (qq) entry, cover a number of intermediate gluons, and end at a qbar (q) or qq (qbarqbar) entry. Particles could be interspersed in this list with no adverse effects, i.e. a u-g-gamma-ubar sequence would be interpreted as a u-g-ubar string plus an additional photon. A closed gluon loop would be assumed to be made up of a sequential listing of the gluons, with the string continuing from the last gluon up back to the first one. Contrary to the previous, open string case, the appearance of any particle but a gluon would therefore signal the end of the gluon loop. For example, a g-g-g-g sequence would be interpreted as one single four-gluon loop, while a g-g-gamma-g-g sequence would be seen as composed of two two-gluon systems.

If these interpretations, which are not unique, are not to the liking of the user, it is up to him/her to correct them, e.g. by using LUJOIN to tell exactly which partons should be joined, in which sequence, to give a string. Calls to LUJOIN (or the equivalent) are also necessary if LUSHOW is to be used to have some partons develop a shower.

For practical applications, one should note that e+e- events, which have been allowed to shower but not to fragment, do have partons arranged in the order assumed above, so that a translation to HEPEVT and back does not destroy the possibility to perform fragmentation by a simple LUEXEC call. Also the hard interactions in PYTHIA fulfil this condition, while problems may appear in the multiple interaction scenario, where several closed gg loops may appear directly following after each other, and thus would be interpreted as a single multigluon loop after translation back and forth.

****

3.   The e+e- Routines

The routines in this section are more specific than the ones in

section 2: aim is taken on applications to e+e- annihilation events,
in the continuum or on an onium resonance.
_____

3.1. e+e- Continuum Event Generation

The only routine a normal user will call is LUEEVT. The other routines
listed below, as well as LUSHOW (see section 2.4), are called by LUEEVT.

SUBROUTINE LUEEVT(KFL,ECM)
Purpose: to generate a complete event e+e- -> gamma/Z0 -> qqbar ->
    parton shower -> hadrons according to QFD and QCD cross-sections.
    As an alternative to parton showers, second order matrix elements
    are available for qqbar + qqbarg + qqbargg + qqbarq'qbar'
    production.
KFL : flavour of events generated.
    = 0 : mixture of all allowed flavours according to relevant
        probabilities.
    = 1 - 8 : primary quarks are only of the specified flavour KFL.
ECM : total CM energy of system.
Remark: Each call generates one event, which is independent
    of preceding ones, with one exception, as follows. If radiative
    corrections are included, the shape of the hard photon spectrum is
    recalculated only with each LUXTOT call, which normally is done
    only if KFL, ECM or MSTJ(102) is changed. A change of e.g. the Z0
    mass in midrun has to be followed either by a user call to LUXTOT
    or by an internal call forced e.g. by putting MSTJ(116)=3.

SUBROUTINE LUXTOT(KFL,ECM,XTOT)
Purpose : to calculate the total hadronic cross-section, including
    quark thresholds, weak, beam polarization and QCD effects and
    radiative corrections. In the process, variables necessary
    for the treatment of hard photon radiation are calculated and
    stored.
KFL, ECM : as for LUEEVT.
XTOT : the calculated total cross-section in nb.

SUBROUTINE LURADK(ECM,MK,PAK,THEK,PHIK,ALPK)
Purpose: to describe initial state hard photon radiation.

SUBROUTINE LUXKFL(KFL,ECM,ECMC,KFLC)
Purpose: to generate the primary quark flavour in case this is not
    specified by user.

SUBROUTINE LUXJET(ECM,NJET,CUT)
Purpose: to determine the number of jets (2, 3 or 4) to be generated
    within the kinematically allowed region (characterized by
    CUT = y_cut) in the matrix element approach; to be chosen such
    that all probabilities are between 0 and 1.

SUBROUTINE LUX3JT(NJET,CUT,KFL,ECM,X1,X2)
Purpose: to generate the internal momentum variables of a three-jet
    event, qqbarg, according to first or second order QCD matrix
    elements.

```
SUBROUTINE LUX4JT(NJET,CUT,KFL,ECM,KFLN,X1,X2,X4,X12,X14)
Purpose: to generate the internal momentum variables for a four-jet
    event, qqbargg or qqbarqqbar, according to second order QCD
    matrix elements.

SUBROUTINE LUXDIF(NC,NJET,KFL,ECM,CHI,THE,PHI)
Purpose: to describe the angular orientation of the jets. In first
    order QCD the complete QED or QFD formulae are used; in second
    order three-jets are assumed to have the same orientation as in
    first, and four-jets are approximated by three-jets.
_____
```

3.2. A Routine for "onium" Decay

In LUONIA we have implemented the decays of heavy onia resonances
into three gluons or two gluons plus a photon, which are the dominant
non-backgroundlike decays of Upsilon, and also would have been it for
a reasonably light top. With the present mass limits, actually weak
decays are expected to dominate, as is already implemented in the
ordinary LUDECY treatment of toponium decay. The inclusion of parton
showering for Upsilon decay is probably overkill, but is there for
completeness.

```
SUBROUTINE LUONIA(KFL,ECM)
Purpose: to simulate the process e+e- -> gamma -> 1-- "onium" resonance
    -> ggg or gggamma -> shower -> hadrons.
KFL : the flavour of the quark giving rise to the resonance.
    = 0 : generate ggg events alone.
    = 1 - 8 : generate ggg and gggamma events in mixture determined
        by the charge-squared of flavour KFL. Normally KFL = 5 or 6.
_____
```

3.3. The Commonblock Variables

The status codes and parameters relevant for the e+e- routines are
found in the commonblock LUDAT1. This commonblock also contains more
general status codes and parameters, which were described in section
2.7.

```
COMMON/LUDAT1/MSTU(200),PARU(200),MSTJ(200),PARJ(200)
Purpose: to give access to a number of status codes and parameters
    regulating the performance of the e+e- event generation routines.

MSTJ(101) : (D=5) gives the type of QCD corrections used for continuum
    events.
    = 0 : only qqbar events are generated.
    = 1 : qqbar + qqbarg events are generated according to first
        order QCD.
    = 2 : qqbar + qqbarg + qqbargg + qqbarqqbar events are generated
        according to second order QCD.
    = 3 : qqbar + qqbarg + qqbargg + qqbarqqbar events are generated,
        but without second order corrections to the three-jet rate.
    = 5 : a parton shower is allowed to develop from an original
        qqbar pair, see MSTJ(41) - MSTJ(49) for details.
    = -1 : only qqbarg events are generated (within same matrix
```

element cuts as for =1). Since the change in flavour
          composition from mass cuts or radiative corrections is not
          taken into account, this option is not intended for
          quantitative studies.
     = -2 : only qqbargg and qqbarqqbar events are generated
          (as for =2). The same warning as for =-1 applies.
     = -3 : only qqbargg events are generated (as for =2).
          The same warning as for =-1 applies.
     = -4 : only qqbarqqbar events are generated (as for =2).
          The same warning as for =-1 applies.
     Note 1: MSTJ(101) is also used in LUONIA, with
     <= 4 : ggg + gammagg events are generated according to lowest
          order matrix elements.
     >= 5 : a parton shower is allowed to develop from the original
          ggg or gggamma configuration, see MSTJ(41) - MSTJ(49) for
          details.
     Note 2: The default values of fragmentation parameters have been
          chosen to work well with the default parton shower approach
          above. If any of the other options are used, or if the parton
          shower is used in non-default mode, it may be necessary to
          retune fragmentation parameters. As an example, we note that
          the second order matrix element approach (MSTJ(101) = 2) at
          PETRA/PEP energies gives a better description with the a and
          b parameters of the symmetric fragmentation function set to
          a = PARJ(41) = 1, b = PARJ(42) = 0.7, and the width of the
          transverse momentum distribution to sigma = PARJ(21) = 0.40.
          In principle, one also ought to change the joining parameter
          to PARJ(33) = PARJ(35) = 1.1 to preserve a flat rapidity
          plateau, but if this should be forgotten, it does not make too
          much of a difference. For applications at TRISTAN or LEP, one
          must expect to have to change the matrix element approach
          parameters even more, to make up for additional soft gluon
          effects not covered in this approach.
MSTJ(102) : (D=2) inclusion of weak effects (Z0 exchange) for flavour
     production, angular orientation, cross-sections and initial state
     photon radiation in continuum events.
     = 1 : QED, i.e. no weak effects are included.
     = 2 : QFD, i.e. including weak effects.
     = 3 : as =2, but at initialization in LUXTOT the Z0 width is
          calculated from $\sin^2(\theta_W)$, alpha_em and Z0 and quark
          masses (including bottom and top threshold factors for
          MSTJ(103) odd), assuming three full generations, and the
          result is stored in PARJ(124).
MSTJ(103) : (D=7) mass effects in continuum matrix elements, in the form
     MSTJ(103) = M_1 + 2*M_2 + 4*M_3, where M_i = 0 if no mass effects
     and M_i = 1 if mass effects should be included. Here;
     M_1 : threshold factor for new flavour production according to
          QFD result;
     M_2 : gluon emission probability (only applies for
          abs(MSTJ(101)) <= 1, otherwise no mass effects anyhow);
     M_3 : angular orientation of event (only applies for
          abs(MSTJ(101)) <= 1 and MSTJ(102)=1, otherwise no mass effects
          anyhow).
MSTJ(104) : (D=5) number of allowed flavours, i.e. flavours that can
     be produced in a continuum event if the energy is big enough.

A change to 6 makes top production allowed above the threshold,
etc. Note that in qqbarqqbar events only the four first flavours
are allowed in the secondary pair, produced by a gluon breakup.

MSTJ(105) : (D=1) fragmentation and decay in LUEEVT and LUONIA calls.
= 0 : no LUEXEC calls, i.e. only matrix element and/or parton
shower treatment.
= 1 : LUEXEC calls are made to generate fragmentation and decay
chain.
= -1 : no LUEXEC calls and no collapse of small jet systems into
one or two particles (in LUPREP).

MSTJ(106) : (D=1) angular orientation in LUEEVT and LUONIA.
= 0 : standard orientation of events, i.e. q along +z axis
and qbar along -z axis or in xz plane with $p_x > 0$ for
continuum events, and $g_1g_2g_3$ or $gammag_2g_3$ in xz plane
with g_1 or gamma along the +z axis for continuum events.
= 1 : random orientation according to matrix elements.

MSTJ(107) : (D=0) radiative corrections to continuum events.
= 0 : no radiative corrections.
= 1 : initial state radiative corrections (including weak
effects for MSTJ(102) = 2 or 3).

MSTJ(108) : (D=2) calculation of alpha_strong for matrix element
alternatives. The MSTU(111) and PARU(112) values are automatically
overwritten in LUEEVT or LUONIA calls accordingly.
= 0 : fixed alpha_strong value as given in PARU(111).
= 1 : first order formula is always used, with Lambda_QCD
given by PARJ(121).
= 2 : first or second order formula is used, depending on
value of MSTJ(101), with Lambda_QCD given by PARJ(121) or
PARJ(122).

MSTJ(109) : (D=0) gives a possibility to switch from QCD matrix elements
to some alternative toy models. Is not relevant for shower
evolution, MSTJ(101) = 3, where one can instead use MSTJ(49).
= 0 : standard QCD scenario.
= 1 : a scalar gluon model. Since no second order corrections are
available in this scenario, one can only use this with
MSTJ(101) = 1 or -1. Also note that the event-as-a-whole angular
distribution is for photon exchange only (i.e. no weak effects),
and that no higher order corrections to the total cross-section
are included.
= 2 : an Abelian vector gluon theory, with the colour factors
$C_F = 1$ (= 4/3 in QCD), $N_C = 0$ (= 3 in QCD) and $T_R = 3 n_f$
(= $n_f/2$ in QCD). If one selects alpha_Abelian =
(4/3) * alpha_QCD, the three-jet cross-section will agree with
the QCD one, and differences are to be found only in four-jets.
The MSTJ(109) = 2 option has to be run with the defaults
MSTJ(110) = 1 and MSTJ(111) = 0; if need be, the latter
variables will be overwritten by the program.
Warning: second order corrections give a large negative
contribution to the three-jet cross-section, so large that
the whole scenario is of doubtful use. In order to make the
second order options work at all, the three-jet cross-section
is here by hand set exactly equal to zero for MSTJ(101) = 2.
It is here probably better to use the option MSTJ(101) = 3,
although this is not a consistent procedure either.

MSTJ(110) : (D=1) choice of second order contributions to the three-jet

rate.
         = 1 : the GKS second order matrix elements, i.e. the old JETSET
               standard.
         = 2 : the Zhu parametrization of the ERT matrix elements, based on
               the program of Kunszt and Ali, i.e. in historical sequence
               ERT/Kunszt/Ali/Zhu. The parametrization is available for
               y = 0.01, 0.02, 0.03, 0.04 and 0.05. Values outside this
               range are put at the nearest border, while values inside
               this range are given by linear interpolation between the
               two nearest points. Since this procedure is rather primitive,
               one should try to work at one of the values given above.
               Note that no Abelian CQD parametrization is available for
               this option.
MSTJ(111) : (D=0) use of optimized perturbation theory for second
      order matrix elements (it can also be used for first order matrix
      elements, but here it only corresponds to a trivial rescaling of
      the alpha_strong argument).
         = 0 : no optimization procedure; i.e. $Q^2 = E\_CM^2$.
         = 1 : an optimized Q^2 scale is chosen as $Q^2 = y' * E\_CM^2$,
               where y' = PARJ(128) for the total cross-section R factor.
               while y' = PARJ(129) for the three- and four-jet rates,
               This y' value enters via the alpha_strong, and also via a
               term proportional to alpha_strong**2*ln(y'). Some constraints
               are imposed; thus the optimized 'three-jet' contribution to R
               is assumed positive (for PARJ(128)), the total three-jet rate
               piece of second order) contribution to R is assumed positive,
               is not allowed to be negative (for PARJ(129)), etc.
               However, there is no guarantee that the differential three-jet
               cross-section is not negative (and truncated to 0) somewhere
               (this can also happen with y' = 1, but is then less frequent).
               The actually obtained y' values are stored in PARJ(168) and
               PARJ(169), respectively.
               If an optimized Q^2 scale is used, then the Lambda_QCD (and
               alpha_strong) should also be changed. With the value y' = 0.002,
               it has been shown [Bet89] that a Lambda_QCD = 0.100 GeV gives
               a reasonable agreement; the parameter to change is PARJ(122)
               for a second order running alpha_strong. Note that, since
               the optimized Q^2 scale is sometimes below the charm threshold,
               the effective number of flavours used in alpha_strong may well
               be 4 only. If one feels that it is still appropriate to use 5
               flavours (one choice might be as good as the other), it is
               necessary to put MSTU(113) = 5.
MSTJ(115) : (D=1) documentation of continuum or onium events, in
      increasing order of completeness.
         = 0 : only the parton shower, the fragmenting partons and the
               generated hadronic system are stored in the LUJETS
               commonblock.
         = 1 : also a radiative photon is stored (for continuum events).
         = 2 : also the original e+e- are stored (with KS=21).
         = 3 : also the gamma or gamma/Z0 exchanged for continuum events,
               the onium state for resonance events is stored (with KS=21).
MSTJ(116) : (D=1) initialization of total cross-section and radiative
      photon spectrum in LUEEVT calls.
         = 0 : never; can not be used together with radiative corrections.
         = 1 : calculated at first call and then whenever KFL or MSTJ(102) is

changed or ECM is changed by more than PARJ(139).
      = 2 : calculated at each call.
      = 3 : everything is reinitialized in next call, but MSTJ(116) is
            afterwards automatically put =1 for use in subsequent calls.
MSTJ(119) : (I) check on need to reinitialize LUXTOT.
MSTJ(120) : (R) type of continuum event generated with matrix element
      option (with the shower one, the result is always =1).
      = 1 : qqbar.
      = 2 : qqbarg.
      = 3 : qqbargg from Abelian (QED-like) graphs in matrix element.
      = 4 : qqbargg from non-Abelian (i.e. containing three-gluon
            coupling) graphs in matrix element.
      = 5 : qqbarqqbar.
MSTJ(121) : (R) flag set if a negative differential cross-section was
      encountered in the latest LUX3JT call. Events are still generated,
      but maybe not quite according to the distribution one would like
      (the rate is set zero in the regions of negative cross-section,
      and the differential rate in the regions of positive cross-section
      is rescaled to give the 'correct' total three-jet rate).

PARJ(121) : (D=1.5 GeV) Lambda value used in first order calculation
      of alpha_strong in the matrix element alternative.
PARJ(122) : (D=0.5 GeV) Lambda values used in second order calculation
      of alpha_strong in the matrix element alternative.
PARJ(123) : (D=91.2 GeV) mass of Z0 as used in propagators for QFD case.
PARJ(124) : (D=2.4 GeV) width of Z0 as used in propagators for QFD
      case. Overwritten at initialization if MSTJ(102) = 3.
PARJ(125) : (D=0.02) y_cut, minimum scaled invariant mass-squared of any
      two partons in 3- or 4-jet events; the main user-controlled matrix
      element cut. PARJ(126) provides an additional constraint. For each
      new event, it is additionally checked that the total three- plus
      four-jet fraction does not exceed unity; if so the effective y cut
      will be dynamically increased. The actual y cut value is stored in
      PARJ(150), event by event.
PARJ(126) : (D=2. GeV) minimum invariant mass of any two partons in
      3- or 4-jet events; a cut in addition to the one above, mainly
      for the case of a radiative photon lowering the hadronic CM
      energy significantly.
PARJ(127) : (D=1. GeV) is used as a safety margin for small colour
      singlet jet systems, cf. PARJ(32), specifically qqbar masses in
      qqbarqqbar 4-jet events and gg mass in onium gammagg
      events.
PARJ(128) : (D=0.25) optimized Q^2 scale for the QCD R (total rate)
      factor for the MSTJ(111) = 1 option is given by $Q^2 = y' * E\_CM^2$,
      where y' = PARJ(128). For various reasons the actually used y'
      value may be raised from the nominal one; while PARJ(128) gives
      the nominal value PARJ(168) gives the actual one for the current
      event.
PARJ(129) : (D=0.002) optimized Q^2 scale for the three- and four-jet
      rate for the MSTJ(111) = 1 option is given by $Q^2 = y' * E\_CM^2$,
      where y' = PARJ(129). For various reasons the actually used y'
      value may be raised from the nominal one; while PARJ(129) gives
      the nominal value PARJ(169) gives the actual one for the current
      event. The default value is in agreement with the studies of
      Bethke [Bet89].

PARJ(131), PARJ(132) : (D=2*0.) longitudinal polarizations P_L+ and
    P_L- of incoming e+ and e-.
PARJ(133) : (D=0.) transverse polarization P_T = (P_T+*P_T-)^*(1/2)
    with P_T+ and P_T- transverse polarizations of incoming e+
    and e-.
PARJ(134) : (D=0.) mean of transverse polarization directions of
    incoming e+ and e-, Deltaphi = (phi_+ + phi_-)/2, with phi
    azimuthal angle of polarization, leading to a shift in the phi
    distribution of jets by Deltaphi.
PARJ(135) : (D=0.01) minimum photon energy fraction (of beam energy)
    in initial state radiation; should normally never be changed
    (if lowered too much, the fraction of events containing a
    radiative photon will exceed unity, leading to problems).
PARJ(136) : (D=0.99) maximum photon energy fraction (of beam energy)
    in initial state radiation; may be changed to reflect actual
    trigger conditions of a detector (but must always be larger
    than PARJ(135)).
PARJ(139) : (D=0.2 GeV) maximum deviation of ECM from the
    corresponding value at last LUXTOT call, above which a new call
    is made if MSTJ(116) = 1.
PARJ(141) : (R) value of R, the ratio of continuum cross-section
    to the lowest order muon pair production cross-section,
    as given in massless QED (i.e. three times the sum of active
    quark charges-squared, possibly modified for polarization).
PARJ(142) : (R) value of R including quark mass effects (for
    MSTJ(102)=1) and/or weak propagator effects (for MSTJ(102)=2).
PARJ(143) : (R) value of R as PARJ(142), but including QCD corrections
    as given by MSTJ(101).
PARJ(144) : (R) value of R as PARJ(143), but additionally including
    corrections from initial state photon radiation (if MSTJ(107)=1).
    Since the effects of heavy flavour thresholds are not simply
    integrable, the initial value of PARJ(144) is updated during the
    course of the run to improve accuracy.
PARJ(145) - PARJ(148) : (R) absolute cross-sections in nb as for
    the cases PARJ(141) - PARJ(144) above.
PARJ(150) : (R) current effective matrix element cutoff y_cut, as given
    by PARJ(125), PARJ(126) and the requirements of having non-negative
    cross-sections for two-, three- and four-jet events. Not used in
    parton showers.
PARJ(151) : (R) value of CM energy ECM at last LUXTOT call.
PARJ(152) : (R) current first-order contribution to the three-jet
    fraction; modified by mass effects. Not used in parton showers.
PARJ(153) : (R) current second-order contribution to the three-jet
    fraction; modified by mass effects. Not used in parton showers.
PARJ(154) : (R) current second-order contribution to the four-jet
    fraction; modified by mass effects. Not used in parton showers.
PARJ(155) : (R) current fraction of four-jet rate attributable to
    qqbarqqbar events rather than qqbargg ones; modified by mass
    effects. Not used in parton showers.
PARJ(156) : (R) has two functions when using second order QCD.
    For a three-jet event, it gives the ratio of the second-order
    to the total three-jet cross-section in the given kinematical
    point. For a four-jet event, it gives the ratio of the
    modified four-jet cross-section, obtained when interference terms
    with not well defined colour flow are neglected, to the full

unmodified one, all evaluated in the given kinematical point.
        Not used in parton showers.
PARJ(157) - PARJ(159) : (I) used for cross-section calculations to
        include mass threshold effects to radiative photon cross-section.
        What is stored is basic cross-section, number of events
        generated and number that passed cuts.
PARJ(160) : (R) nominal fraction of events that should contain
        a radiative photon.
PARJ(161) - PARJ(164) : (I) give shape of radiative photon spectrum
        including weak effects.
PARJ(168) : (R) actual y' value of current event in optimized
        perturbation theory for R; see MSTJ(111) and PARJ(128).
PARJ(169) : (R) actual y' value of current event in optimized
        perturbation theory for three- and four-jet rate; see MSTJ(111)
        and PARJ(129).
        _____

3.4. Examples

An ordinary e+e- annihilation event in the continuum, at a CM energy
of 40 GeV, may be generated with
        CALL LUEEVT(0,40.)
In this case a qqbar event is generated, including weak effects,
followed by parton shower evolution and fragmentation/decay treatment.
Before a call to LUEEVT, however, a number of default values may be
changed, e.g. MSTJ(101) = 2 to use second order QCD matrix elements,
giving a mixture of qqbar, qqbarg, qqbargg and qqbarq'qbar' events,
MSTJ(102) = 1 to have QED only, MSTJ(104) = 6 to allow ttbar production
as well, MSTJ(107) = 1 to include initial state photon radiation
(including a treatment of the Z0 pole), PARJ(123) = 92.0 to change the
Z0 mass, PARJ(81) = 0.3 to change the parton shower Lambda value, or
PARJ(82) = 1.5 to change the parton shower cutoff. If initial state
photon radiation is used, some restrictions apply on how one can
alternate the generation of events at different energies or with
different Z0 mass etc. These restrictions are not there for
efficiency reasons (the extra time for recalculating the extra
constants every time is small), but because it ties in with the
the cross-section calculations (see PARJ(144)).

Most parameters can be changed independently of each other. However,
if just one or a few parameters/switches are changed, one should not
be surprised to find a rather bad agreement with the data, like e.g.
a too low or high average hadron multiplicity. It is therefore usually
necessary to retune one parameter related to the perturbative QCD
description, like alpha_S or Lambda, one of the two parameters a and b
of the Lund symmetric fragmentation function (since they are so
strongly correlated, it is often not necessary to retune both of
them), and the average fragmentation transverse momentum - see Note 2
of the MSTJ(101) description for one example. For very detailed studies
it may be necessary to retune even more parameters.

The three-gluon or gluon-gluon-photon decay Upsilon may be simulated
by a call
        CALL LUONIA(5,9.46)
Unfortunately, with present top mass limits, this routine will not be

of much interest for toponium studies (weak decays will dominate).

A typical program for analysis of e+e- annihilation events at 100 GeV
might look something like

```
      COMMON/LUJETS/N,K(4000,5),P(4000,5),V(4000,5)
      COMMON/LUDAT1/MSTU(200),PARU(200),MSTJ(200),PARJ(200)
      COMMON/LUDAT2/KCHG(500,3),PMAS(500,4),PARF(2000),VCKM(4,4)
      COMMON/LUDAT3/MDCY(500,3),MDME(2000,2),BRAT(2000),KFDP(2000,5)
      MDCY(LUCOMP(111),1)=0        ! put pi0 stable
      MSTJ(104)=6                  ! allow top-antitop production
      PMAS(6,1)=45.                ! change top quark mass
      MSTJ(107)=1                  ! include initial state radiation
      PARU(41)=1.                  ! use linear sphericity
      .....                        ! other desired changes
      .....                        ! initialize analysis statistics
      DO 100 IEVENT=1,1000         ! loop over events
      CALL LUEEVT(0,100.)          ! generate new event
      IF(IEVENT.EQ.1) CALL LULIST(2) ! list first event
      CALL LUTABU(11)              ! save particle composition
                                   !   statistics
      CALL LUEDIT(2)               ! remove decayed particles
      CALL LUSPHE(SPH,APL)         ! linear sphericity analysis
      IF(SPH.LT.0.) GOTO 100       ! too few particles in event for
                                   !   LUSPHE to work on it (unusual)
      CALL LUEDIT(31)              ! orient event along axes above
      IF(IEVENT.EQ.1) CALL LULIST(2) ! list first treated event
      .....                        ! fill analysis statistics
      CALL LUTHRU(THR,OBL)         ! now do thrust analysis
      .....                        ! more analysis statistics
  100 CONTINUE                     !
      CALL LUTABU(12)              ! print particle composition
                                   !   statistics
      .....                        ! print analysis statistics
      END
```

****

      Acknowledgements

Constructive criticism from a number of users has been very helpful.
In the work with JETSET 7.1 and 7.2, comments from Hans-Uno Bengtsson
and Bill Gary have been of particular importance. In some cases, users
have submitted pieces of code, with suggested extensions. This
correspondence has been valuable, but the new code actually used in
the program has been written entirely by the author, to ensure a
uniform programming style. Also, the (moral) responsibility for any
errors and other shortcomings rests solely with the author.

****

      References

Ali80   A. Ali, J. G. Korner, G. Kramer, J. Willrodt, Nucl. Phys. B168
        (1980) 409
        A. Ali, E. Pietarinen, G. Kramer, J. Willrodt, Phys. Lett. B93

(1980) 155

And82   B. Andersson, G. Gustafson, T. Sjostrand, Nucl. Phys. B197
         (1982) 45

And85   B. Andersson, G. Gustafson, T. Sjostrand, Physica Scripta 32
         (1985) 574

And83   B. Andersson, G. Gustafson, G. Ingelman, T. Sjostrand,
         Phys. Rep. 97 (1983) 31

Ben87   H.-U. Bengtsson, T. Sjostrand, Computer Phys. Comm. 46 (1987) 43

Bet86   JADE Collaboration, W. Bartel et al., Z. Physik C33 (1986) 23
         S. Bethke, Habilitation thesis, LBL 50-208 (1987)

Bet89   S. Bethke, Z. Physik C43 (1989) 331

Bia86   A. Bialas, R. Peschanski, Nucl. Phys. B273 (1986) 703

Cla79   L. Clavelli, Phys. Lett. B85 (1979) 111
         A. V. Smilga, Nucl. Phys. B161 (1979) 449

Fie78   R. D. Field, R. P. Feynman, Nucl. Phys. B136 (1978) 1

Fox79   G. C. Fox, S. Wolfram, Nucl. Phys. B149 (1979) 413

Gro81   T. R. Grose, K. O. Mikaelian, Phys. Rev. D23 (1981) 123

Hoy79   P. Hoyer, P. Osland, H. G. Sander, T. F. Walsh, P. M. Zerwas,
         Nucl. Phys. B161 (1979) 349

Ing80   G. Ingelman, T. Sjostrand, LUTP 80-12 (1980)
         G. Ingelman, LEPTO version 4.3, CERN program pool long writeup,
         program W5046 (1986)

Ing87   G. Ingelman, Computer Phys. Comm. 46 (1987) 217

Ing87a   G. Ingelman, A. Weigend, Computer Phys. Comm. 46 (1987) 241

Ing88   G. Ingelman, G. A. Schuler, Z. Phys. C40 (1988) 299

Jam88   F. James, CERN-DD/88/22 (1988)

Kra86   G. Kramer, B. Lampe, Fortschr. Physik 37 (1989) 161

Lor88   B. Lorstad, Int. J. of Mod. Phys. A4 (1989) 2861

Mar87   G. Marsaglia, A. Zaman, FSU-SCRI-87-50 (1987)

Nil87   B. Nilsson-Almqvist, E. Stenlund, Computer Phys. Comm. 43 (1987)
         387

PDG86   Particle Data Group, M. Aguilar-Benitez et al.,
         Phys. Lett. B170 (1986) 1

PDG88   Particle Data Group, G. P. Yost et al.,
        Phys. Lett. B204 (1988) 1

Pet83   C. Peterson, D. Schlatter, I. Schmitt, P. Zerwas, Phys. Rev. D27
        (1983) 105

Pet88   U. Pettersson, LU TP 88-5 (1988)
        L. Lonnblad, U. Pettersson, LU TP 88-15 (1988)
        L. Lonnblad, LU TP 89-10 (1989)

QEG89   QCD event generators subgroup of the 1989 LEP physics workshop,
        B. Bambah et al., CERN-TH.5466/89 (1989), to appear in the
        proceedings of the workshop

Sjo78   T. Sjostrand, B. Soderberg, LU TP 78-18 (1978)

Sjo79   T. Sjostrand, LU TP 79-8 (1979)

Sjo80   T. Sjostrand, LU TP 80-3 (1980)

Sjo82   T. Sjostrand, Computer Phys. Comm. 27 (1982) 243

Sjo83   T. Sjostrand, Computer Phys. Comm. 28 (1983) 227

Sjo84   T. Sjostrand, Nucl. Phys. B248 (1984) 469

Sjo84a T. Sjostrand, Z. Physik C26 (1984) 93

Sjo86   T. Sjostrand, Computer Phys. Comm. 39 (1986) 347

Sjo87   T. Sjostrand, M. Bengtsson, Computer Phys. Comm. 43 (1987) 367

Sjo88   T. Sjostrand, Int. J. of Mod. Phys. A3 (1988) 751

****