LanHEP - a package for automatic generation of Feynman rules from the Lagrangian. Updated version 1.3.

A. Semenov
email: `semenov@theory.npi.msu.su`

INP MSU PREPRINT 98–2/503

March 8, 1998

Abstract

This short note describes new features of the version 1.3 of the LanHEP software package; in particular, 1-loop counterterms generation and simplification of large trigonometric expressions. The LanHEP program is aimed for Feynman rules generation in momentum representation. It reads the Lagrangian written in the compact form close to one used in publications, with summation over indices of broken symmetries and using special symbols for complicated expressions. The output is Feynman rules in terms of physical fields and independent parameters. Two formats are available for this output: CompHEP tables and tables in LaTeX code.

Introduction

The LanHEP program is aimed to generate automatically the Feynman rules from the Lagrangian. The output can be written in LaTeX format and in the form of CompHEP [1] tables. The CompHEP package allows symbolic computation of the matrix element squared of any process with up to 6 incoming and outgoing particles and calculate cross-sections and various distributions. The physical model in CompHEP is presented as a table of all vertices.

LanHEP reads a description of the physical model from the specified file (or console input) and than writes out Feynman rules for propagators and vertices in momentum representation in CompHEP (or LaTeX) format. The input file contains a number of statements describing the Lagrangian written in evident notation and the set of particles and coupling constants it uses.

The format of the LanHEP input file is described in details in the LanHEP User's Manual [2]. The LanHEP program and related papers can be found in the WWW page:

`http://theory.npi.msu.su/~semenov/lanhep.html`

Here we describe new features implemented into LanHEP package:

- generation of 1-loop counterterms;
- simplification of large trigonometric expressions;
- extraction of some classes of vertices;

- extracting factors in the output expression;
- inspecting the stage of processing the model;
- options allowing fine tuning of the program processing and output.

1 Generation of counterterms

Ultraviolet divergencies in renormalized quantum field theories are compensated by renormalization of wave functions $\phi_i \to \phi_i(1+\delta z_i)$, masses $M_i \to M_i + \delta M_i$, charges $e_i \to e_i + \delta e_i$, and may be some other Lagrangian constants. Such transformation of the Lagrangian assumes changes in Feynman rules. In particular vertices are changed due to appearing new terms — *counterterms*. It is possible now to treat this transformation at 1-loop level by means of LanHEP package.

The statement

`transform` *obj* `->` *expr*.

forces LanHEP to substitute symbol of a parameter or a particle, *obj*, by the expression *expr* when this object appear in the Lagrangian expressions of the Lagrangian.

For example, if electric charge e in QED is renormalized than the statement

```
transform e -> e+d_e.
```

makes LanHEP to substitute each occurence of `e` symbol in further expressions by `e+d_e`. Of course, `e` and `d_e` should be declared as parameters before this statement.

The `transform` statement is very similar to the `let` [2] statement and uses the same syntax for expression *expr*. It has however two advantages. First, one does not need to introduce new name for transformed parameter or field. Second, if one has the LanHEP file for some physical model, it is enough to add counterterms statements into the input file just after the statements declaring parameters and particles.

In 1-loop approximation renormalization parameters appear in expressions only at first power, i.e. $\delta Z_i \delta Z_j = 0$, where δZ_i means all renormalization parameters (δe_i, δM_i, δz_i, ...). This should be done with the help of the following statement

`infinitesimal` *d_Z1, d_Z2, ..., d_Zn.*

It declares that a monomial should be omitted if it contains more than one of parameters *d_Z1, d_Z2, ..., d_Zn.*

An example for QED model with renormalization reads as:

```
parameter e = 0.3133:  'Electric charge'.
vector A/A:photon.
spinor e1/E1:(electron, mass me=0.000511).
parameter d_e, d_me, d_e1, d_A.
infinitesimal d_e, d_me, d_e1, d_A.
transform e -> e*(1+d_e), me -> me+d_me,
      A -> A*(1+d_A),
      e1 -> e1 + d_e1*e1,
      E1 -> E1 + d_e1*E1.
lterm e*E1*(i*gamma*deriv+gamma*A+me)*e1.
```

2 Reducing trigonometric expressions

Some physical models, such as Minimal Supersymmetric Standard Model [3] and Two Higgs Doublet Model [4] (Feynman rules generations for these models by means of LanHEP are described in [5, 6]) involve large expressions built of trigonometric functions. In particlular, in the models mentioned above two angles, α and β, are involved, thus the Lagrangian may be written in LanHEP notation using the following definitions:

```
parameter sa=0.5:'sinus alpha',
          ca=Sqrt(1-sa**2):'cosine alpha'.
parameter sb=0.9:'sinus beta',
          cb=Sqrt(1-sb**2):'cosine beta'.
```

One can find in the output expressions like `sa*cb+ca*sb` $= \sin(\alpha+\beta)$ and much more complicated ones. To simplify the output and make it more readable user can define new parameters, in the example above it is:

```
parameter sapb=sa*cb+ca*sb:'sin(a+b)'.
```

To force LanHEP to substitute the expression `sa*cb+ca*sb` by the parameter `sapb`, one should use the statement

```
SetAngle(sa*cb+ca*sb=sapb).
```

It is possible also to substitute an expressions by any polynomial involving paramters, for example,

```
SetAngle((sa*cb+ca*sb)**2+(sa*cb-ca*sb)**2=sapb**2+samb**2).
```

where `samb` should be previously declared as a parameter. Note, that LanHEP expands both expressions before setting the substitution rule. The left side expression can also contain symbols defined by `let` statement. In the Lagrangian expressions LanHEP express powers of cosines, `ca**`N and `cb**`N through sines using recursively the formula $\cos^N\alpha = \cos^{N-2}(1-\sin^2\alpha)$ to combine all similar terms. The same procedure is performed for the left side expression in `SetAngle` statement.

If the substitution for one angle combination is defined, LanHEP offers to define the other ones for expressions consisting of parameters used in previous `SetAngle` statements. In our example, for all expressions consisting of parameters `ca, cb, sa, sb`, the following message is printed (e.g. for $\cos(\alpha+\beta)$):

```
Warning:  undefined angle combination; use:
          SetAngle(ca*cb-sa*sb=aa000).
```

Here `aa000` is an automatically generated parameter. The message is printed only once for each expression. This feature is disabled by default; to enable producing these messages, one should issue the statement

```
option UndefAngleComb=1.
```

3 Extracting vertices

New statement allows to extract a class of vertices into separate file. This feature is useful in checking large models with thousands of vertices, such as MSSM. The statement has the form

`SelectVertices(`*file, vlist* `).`

Here *file* is a file name to output selected vertices, and *vlist* determines the class of vertices. The pattern *vlist* may have two possible formats.

In the first format *vlist* is a list of particles:

`[`*P1, P2, P3, ... Pn* `]`

All vertices consisting only of the listed particles *P1, P2,...* are extracted. For example, the pattern

```
[A, Z, 'W+', 'W-', e1, E1, n1, N1]
```

selects vertices of leptons of first generation interaction with gauge fields and the self-interaction of gauge fields (we used here particles notation of CompHEP).

The second format reads

`[`*P11/P12../P1n, P21/.../P2m, ...* `]`

Here are several groups of particles, each group is joined by slash '/' symbol. Selected vertices have the number of legs equal to the number of groups in the list, each leg in the vertex corresponds to one group. For example,

```
[e1/E1/n1/N1, e1/E1/n1/N1, A/Z/'W+'/'W-']
```

selects vertices with two legs corresponding to leptons, third leg is gauge boson. Thus, this pattern extracts vertices of leptons interactions with gauge bosons. Note that the vertices of gauge bosons interaction are not selected in this case. If a group consists of one particle, it must be typed twice to enable second format (e.g. `[P1/P1, P2/P2, P3/P3]` for the vertex consisting of particles `P1, P2, P3`.

It is possible to specify some options in `SelectVertices` statement. The `WithAnti` option adds antiparticle names for each particle in the list (in the first format) or in the each group (in the second format. The `Delete` option removes selected vertices from the LanHEP's internal vertices list, so these vertices are not written in output file `lgrng`N`.mdl (lgrng`N`.tex)`. This could be useful if `SelectVertices` statements are supposed to distribute all vertices which should be in the model, than some 'unexpected' vertices can be found in output `lgrng`N`.mdl` file.

An example of lepton-gauge vertices selection could read as

```
SelectVertices( 'lept-gauge.mdl', [e1/n1, e1/n1, A/Z/'W+'], WithAnti, Delete).
```

4 Extracting factors in the output expressions.

By default, output vertices are presented as merely sums of monomials. It is possible to extract the common factor for the group of monomials. Parameters *Pi* which can be common factors, should be declared by the statement

`coeff` *P1, P2, ..., Pn.*

5 Inspecting the current stage of processing

During the processing large model, it is possible to learn what is LanHEP doing now. One can use UNIX 'kill' command aimed to terminate processes and (in more general case) to send a signal to a process:

`kill -10` *pid*

Here the option `-10` specifies this command to send the signal 10 to the process; *pid* is the process number, it can be learned by means of UNIX `ps` command. Having received this signal, LanHEP prints current line of input file, the stage of processing the statement and resume processing. It is possible also to use `-v` command line option to turn on the verbose mode. In this mode LanHEP prints each `lterm` statement on console before executing it.

6 Tuning the Lagrangian processing and program output.

New option allows to simplify debugging the model using CompHEP output format. The statement

```
option chepBreakLines=1.
```

forces LanHEP to print each monomial of vertex expression in the new line. This is useful when inspecting vertices which contain very long expressions. Of course, CompHEP can not read vertices table in such format.

It is possible to include into CompHEP output the 2-legs vertices (and 1-legs ones, they can appear from incorrectly written Higgs potential) by means of `-allvrt` command line option. This feature is also aimed to check the model. CompHEP does not allow such vertices in model file.

Acknowledgments

The work was partially supported by RFBR grant 96-02-18635.

References

[1] E. Boos et al., INP MSU–94–36/358, SNUTP 94-116 (Seoul, 1994); hep-ph/9503280
P. Baikov et al., Physical Results by means of CompHEP, in Proc.of X Workshop on High Energy Physics and Quantum Field Theory (QFTHEP-95), ed.by B.Levtchenko, V.Savrin, Moscow, 1996, p.101 , hep-ph/9701412

[2] A. Semenov. *LanHEP — a package for automatic generation of Feynman rules. User's manual.* INP MSU 96–24/431, Moscow, 1996; hep-ph/9608488
A. Semenov. Nucl.Inst.&Meth. **A393** (1997) p. 293.

[3] J. Rosiek. Phys. Rev. **D41** (1990) p. 3464.

[4] J.F. Gunion, H.E. Haber, G. Kane and S. Dawson. *The Higgs Hunter Guide,* Addison-Wesley, 1990.

[5] A. Belyaev, A. Gladyshev and A. Semenov. IFT–P–093–97, hep-ph/9712303.

[6] M. Dubinin and A. Semenov, *in preparation.*