CERN COMPUTER NEWSLETTER

CERN - CN Division
CH-1211 Geneve 23
SWITZERLAND

JANUARY – MARCH 1996

No. 222 – "MIGRATION SPECIAL – MAIL ISSUES"

Contents

Special Wide Distribution

6. Text Processing

6.1 Drawing Feynman Diagrams with LaTeX and METAFONT. Part 3: Extension Mechanism and Frequently Asked Questions

Thorsten Ohl (TH Darmstadt, Theoretical High Energy Physics Group)
e-mail: `Thorsten.Ohl@Physik.TH-Darmstadt.de`

Abstract

This is the third and final article in a series describing the LaTeX package `feynMF` for easy drawing of professional quality Feynman diagrams with METAFONT (or METAPOST). This time the mechanisms for extending `feynMF` are covered and some trouble-shooting advice is given.

6.1.1 Introduction

In [1,2], I have described the design principles and simple applications of `feynMF` [3,4]. Now it is time to discuss the more esoteric features of `feynMF`, those that allow efficient creation of rather complicated Feynman diagrams.

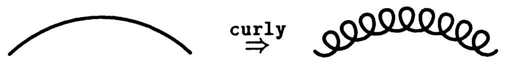
6.1.2 The extension mechanism

`feynMF` is an extensible system, i.e. a documented extension mechanism exists via the METAFONT macro `style_def`. It enables the user to define new styles for propagators, which are not provided in the standard distribution. These styles can be used just like any predefined style. Several concrete examples will be given below.

The examples presented in this part can be used as a cookbook from which recipes can be copied without intimate knowledge of METAFONT. Nevertheless, familiarity with some chapters of [5] will certainly be helpful for a deeper understanding of the macros.

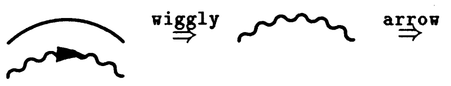
Transformers

An important tool for generating complex line styles is provided by *transformers*. These are functions that take a path as argument and return another path which corresponds to a embellished version. Here is an example that is used for implementing gluon lines:



Such transformers can be combined with other transformers or with drawing functions and packaged in a `style_def`:

```
\fmfcmd{%
style_def charged_boson expr p =
 draw (wiggly p);
 fill (arrow p)
enddef;}
```



The implementation does not attempt to force embellishments (e.g. arrows) into the transformer paradigm, because drawing along a path is inherently different from filling an outline. Therefore embellishments are drawn after each other and not added to the object in a pipeline.

6.1.3 Examples

Crosses

Double lines and crosses are popular among physicists working on ETC. There is no default line style with a cross at the center, but we can use the extension mechanism to define our own rather easily. We start in with a little utility function that returns a rotated bar at the center of the path p. This function is combined with the existing `double` style to the new style `tfermion`:

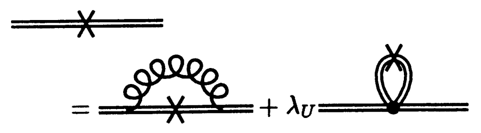
```
\fmfcmd{%
vardef bar (expr p, len, ang) =
 ((-len/2,0)--(len/2,0))
    rotated (ang + angle
      direction length(p)/2 of p)
    shifted point length(p)/2 of p
enddef;
style_def tfermion expr p =
 draw_double p;
 ccutdraw bar (p, 4mm,  60);
 ccutdraw bar (p, 4mm, -60)
enddef;}
```

The `tfermion` line style is used in Figure 6.1 in a Schwinger-Dyson equation. Such definitions can be placed in a little library of personal preferences and reused later in publications dealing with similar diagrams.

Majorana neutrinos

A popular species from beyond the standard model are Majorana neutrinos, which are usually denoted by double arrows.

```
\fmfcmd{%
style_def majorana expr p =
 cdraw p;
 cfill (harrow (reverse p, .5));
 cfill (harrow (p, .5))
enddef;
style_def dbl_majorana expr p =
 draw_double p;
 cfill (tarrow (reverse p, .55));
 cfill (tarrow (p, .55))
enddef;}
```

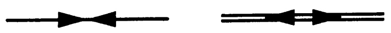


(6.1)

```
\begin{equation}
\begin{split}
 &\parbox{20mm}{\begin{fmfgraph}(20,5)
    \fmfleft{i} \fmfright{o}
    \fmf{tfermion}{i,o}
  \end{fmfgraph}} \\
 &\qquad =
  \parbox{20mm}{\begin{fmfgraph}(20,15)
    \fmfleft{i} \fmfright{o}
    \fmf{double}{i,v1} \fmf{double}{v2,o}
    \fmf{tfermion,tension=.3}{v1,v2}
    \fmf{gluon,left,tension=0}{v1,v2}
  \end{fmfgraph}}
  +\lambda_U
  \parbox{20mm}{\begin{fmfgraph}(20,15)
    \fmfleft{i} \fmfright{o} \fmfdot{v}
    \fmf{double}{i,v,o}
    \fmf{tfermion}{v,v}
  \end{fmfgraph}}
\end{split}
\end{equation}
```

Figure 6.1: Schwinger-Dyson equation in ladder approximation for extended technicolor theories. This example demonstrates the extension mechanism and graphical equations.

The first kind has the arrows facing each other at the tip (`harrow` has the reference point at the head) and the second kind has them facing st the base (`harrow` has the reference point at the tail):



Momentum flow arrows

The most complicated aspect of Figure 6.2 is the placement of the little arrows indicating the momentum flow. One solution, which would be preferable in a case were more of these diagrams are to be drawn, would be to define several line styles corresponding to arrows in various orientations combined with the basic line styles.

Here we choose a more pragmatic approach and define some pseudo-line styles consisting of reduced and shifted arrows *without* a line:

```
\fmfcmd{%
 vardef middir(expr p,ang) =
  dir(angle direction length(p)/2 of p + ang)
 enddef;
 style_def arrow_left expr p =
  shrink(.7);
   cfill(arrow p
    shifted(4thick*middir(p,90)));
  endshrink
 enddef;
 style_def arrow_right expr p =
  shrink(.7);
   cfill(arrow p
    shifted (4thick*middir(p,-90)));
  endshrink
 enddef;}
```

Because the `gluon` lines occupy more space, we have to define another style `warrow_left` which is identical to `arrow_left` except for a larger distance `8thick` from the line. These pseudo-line styles are used in lines 23–26 of Figure 6.2 to place the little arrows. Since the vertices that are connected by these arcs are already fixed, we can omit the `tension=0` option.

The gluons in lines 18–21 do not affect the layout, because they are added after the position of the vertices are fixed by `\fmffreeze`. The same effect could be achieved by adding a `tension=0` option to the gluons.

"Brown muck"

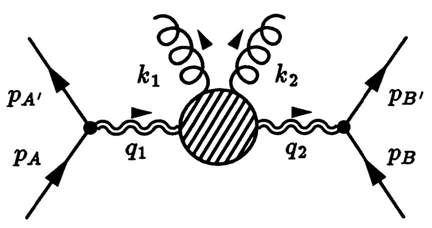
A common problem in phenomenological papers is the representation of non-perturbative contributions, also known as "brown muck". To distinguish them from the perturbative propagators, they are usually shown as grey or shaded areas. `feynMF`'s extension mechanism allows for a systematic treatment of these contributions.

We start by defining a couple of utility macros: the macro port(t,p) returns a unit vector pointing to the left of the path p at t:

```
\fmfcmd{%
 vardef port (expr t, p) =
  (direction t of p rotated 90)
   / abs (direction t of p)
 enddef;}
```

The macro portpath(a,b,p) returns a copy of the path p bent to the left by a at the beginning and end and by b in the middle.

```
\fmfcmd{%
 vardef portpath (expr a, b, p) =
  save l; numeric l; l = length p;
  for t=0 step 0.1 until l+0.05:
   if t>0: .. fi point t of p
    shifted ((a+b*sind(180t/l))*port(t,p))
  endfor
  if cycle p: .. cycle fi
 enddef;}
```



```
\begin{fmfgraph*}(50,30)
 \fmfbottom{pA,pB}
 \fmftop{pA',k1,k2,pB'}
 \fmf{fermion,lab.side=left,
  lab=$p_A$}{pA,vA}
 \fmf{fermion,lab.side=left,
  lab=$p_{A'}$}{vA,pA'}
 \fmf{fermion,lab.side=right,
  lab=$p_B$}{pB,vB}
 \fmf{fermion,lab.side=right,
  lab=$p_{B'}$}{vB,pB'}
 \fmf{dbl_wiggly,lab.side=right,
  lab=$q_1$}{vA,v}
 \fmf{dbl_wiggly,lab.side=left,
  lab=$q_2$}{vB,v}
 \fmfdot{vA,vB} \fmfblob{.2w}{v}
 \fmffreeze
 \fmf{gluon,lab.side=left,
   lab=$k_1$}{v,k1}
 \fmf{gluon,lab.side=left,
   lab=$k_2$}{k2,v}
 \fmf{warrow_right}{v,k1}
 \fmf{warrow_left}{v,k2}
 \fmf{arrow_left}{vA,v}
 \fmf{arrow_left}{v,vB}
\end{fmfgraph*}
```

Figure 6.2: Quasi-Multi-Regge-Kinematics: a typical diagram from a paper on the high-energy asymptotics of QCD.

This macro is now used to draw a fat, shaded "propagator" along the path p:[1]

```
\fmfcmd{%
 style_def brown_muck expr p =
  shadedraw(portpath(thick/2,2thick,p)
   ..reverse(portpath(-thick/2,-2thick,p))
   ..cycle)
 enddef;}
```

This brown_muck style can now be used just like any other predefined line style, cf. Figure 6.3 and Figure 6.4 below.

Protons

The solution used for protons in [2] has been straightforward and sufficient for most applications. Another, more elegant, method is to use style_def for defining a proton line style that can be used instead of fermion. It has to make more use of METAFONT functions, though:

```
\fmfcmd{%
style_def proton expr p =
 save mp, sp; pair mp; path sp;
 mp = point length(p)/2 of p;
 sp = p shifted -mp scaled 1.05 shifted mp;
 cdraw sp shifted (2thick*middir(p, 90));
 cdraw sp shifted (2thick*middir(p,-90));
 draw_fermion p
enddef;}
```

The outer lines are slightly enlarged to match with the circular blobs. This solution is not perfect (it would be a default line style in feynMF otherwise) because it is not guaranteed to match *always* satisfactorily at the vertices. It is nevertheless a good example for how the style_def mechanism can be used to streamline feynMF applications.

Instead of redisplaying the Higgs production diagram from [2] once more, we apply the proton style in Figure 6.4 in a photoproduction diagram. This diagram also shows another application of the brown_muck style.

Two-loop diagrams

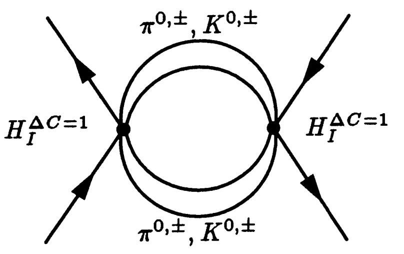
Finally, let us return to the subject of [2], the *immediate mode*. One notorious set of diagrams is given by the NLO contributions to $b \to s\gamma$. Figure 6.5 shows how aesthetically pleasing results can be produced with little effort in feynMF. Topologically, the second and fourth diagram are mirror images and it is a matter of taste which layout style is preferred.

Lines 2–16 are straightforward, except for the trick in line 7, where the quark loop is first drawn invisibly (phantom). Later (lines 10–15) a fermion is drawn on each half (using the undocumented feature that length(p) = 2).

The first four arguments of the \NLO macro describe the starting point of the gluon and the final four the end point. The first argument gives the position on the path connecting the vertices specified in the third argument, while the second argument gives the direction *relative to this path* (this is the non-trivial part). The fourth argument is needed to disambiguate multiple paths connecting the same pair of vertices. The actual path is defined in lines 25–26. Note that the value of the group is the value of the final expression in the group.

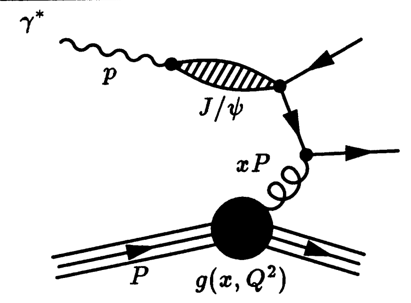
The first and second diagram reveal a slight problem with the current implementation of feynMF: since the coordinate along a METAFONT path doesn't vary uniformly with the actual length, the gluons can appear a bit distorted. The effect is

[1] Note that brown_muck is not as general as it could be, because it doesn't properly handle cyclic paths. This is in principle easy to fix. But the implementation of shading areas of genus > 0 has to behave differently for METAFONT and METAPOST and would occupy too much space in this short article. It will be provided in a future version of feynMF.



```
\begin{fmfgraph*}(50,30)
\fmfleft{i1,i2} \fmfright{o1,o2}
\fmf{fermion}{i1,v1,i2}
\fmf{fermion}{o2,v2,o1}
\fmf{brown_muck,tension=.5,right,%
 label=$\pi^{0,,\pm},,K^{0,,\pm}$}%
 {v1,v2,v1}
\def\H#1{\fmfv{%
 label=$H_I^{\Delta C=1}$,%
 label.dist=6thick}{#1}}
\H{v1}\H{v2}\fmfdotn{v}{2}
\end{fmfgraph*}
```

Figure 6.3: "Brown Muck" (i.e. long distance strong interactions) in a mixing amplitude.



```
\begin{fmfgraph*}(50,30)
\fmfleft{P,gamma}\fmfright{P',c,cbar}
\fmflabel{$\gamma^*$}{gamma}
\fmf{proton,lab=$P$}{P,g1}
\fmf{proton}{g1,P'}
\fmfv{d.sh=circle,d.fi=shaded,d.si=.16w,%
 lab=$g(x,,Q^2)$,lab.d=.1w,lab.a=-90}{g1}
\fmf{gluon,lab.s=left,lab=$xP$}{g1,g2}
\fmf{photon,lab.s=right,lab=$p$}{gamma,j1}
\fmf{brown_muck,lab.s=right,lab.d=4thick,%
 lab=$J/\psi$}{j1,j2}
\fmf{fermion}{cbar,j2,g2,c}
\fmfdot{j1,j2,g2}
\end{fmfgraph*}
```

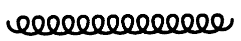
Figure 6.4: Vector meson contribution to heavy flavor photoproduction.

not dramatic, but it might be fixed in a future version anyway.

6.1.4 Frequently asked questions

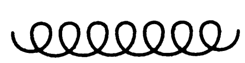
How can I change the layout of gluons, photons and fermions?

The form of the gluon curls is governed by the global `curly_len` variable. It default value is 3mm. Changing it to 2mm make the gluon appear tighter



```
\begin{fmfgraph}(30,3)
\fmfset{curly_len}{2mm}
\fmfleft{i} \fmfright{o} \fmf{gluon}{i,o}
\end{fmfgraph}
```

while a value of 4mm results in a more loose shape:



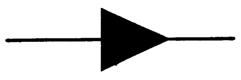
Currently, it is to impossible to switch `curly_len` for individual propagators by using multiple `\fmfset`s. Most propagators are drawn at the very end and the most recent `\fmfset` will be in effect for *all* propagators.

Until this a future version `feynMF` will allow for these parameter as options similar to `tension`, there is a convenient way of achieving the same effect using the extension mechanism. We just define a new style that will locally redefine the appropriate parameters:

```
\fmfcmd{%
style_def ir_photon expr p =
 save wiggly_len, wiggly_slope;
 wiggly_len = 6mm; wiggly_slope = 45;
 draw_photon p
enddef;
style_def uv_photon expr p =
 save wiggly_len, wiggly_slope;
 wiggly_len = 2mm; wiggly_slope = 90;
 draw_photon p
enddef;}
```

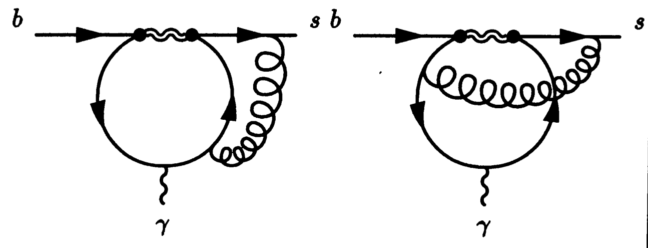
In the case of photons, there are two parameters `wiggly_len` and `wiggly_slope`. The latter should be increased when the wiggles are shortened. In Figure 6.6, we use both kinds of photons just like any other line style.

The shape of the arrow heads is governed by `arrow_len` and `arrow_ang`, with the latter denoting the half opening angle of the arrow head:

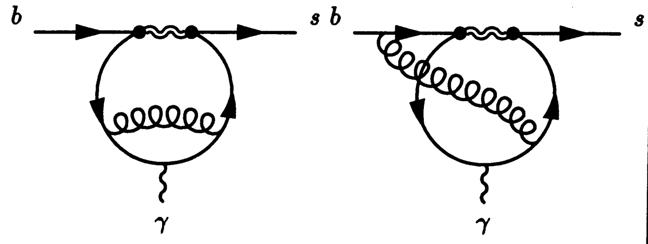


```
\begin{fmfgraph}(30,3)
```

```
\global\def\NLO#1#2#3#4#5#6#7#8{%
\begin{fmfgraph*}(35,25)
 \fmftop{b,s} \fmfbottom{g}
 \fmflabel{$b$}{b} \fmflabel{$s$}{s}
 \fmflabel{$\gamma$}{g}
 \fmf{fermion}{b,v1}\fmf{fermion}{v2,s}
 \fmf{phantom,right=5,tag=1}{v1,v2}
 \fmf{dbl_wiggly}{v1,v2}
 \fmffreeze
 \fmfi{fermion}{%
   subpath (0,1) of vpath1(__v1,__v2)}
 \fmfi{fermion}{%
   subpath (1,2) of vpath1(__v1,__v2)}
 \fmfi{photon}{vloc(__g)
   .. point 1 of vpath1(__v1,__v2)}
 \fmfdotn{v}{2}
 \fmfset{curly_len}{2.5mm}
 \fmfi{gluon}{%
  begingroup;
   clearxy; save p, t;
   path p[]; numeric t[];
   p1 = vpath#4(#3); p2 = vpath#8(#7);
   t1 = #1*length(p1); z1 = point t1 of p1;
   t2 = #5*length(p2); z2 = point t2 of p2;
   z1{direction t1 of p1 rotated #2}
    ..{direction t2 of p2 rotated #6}z2
  endgroup}
\end{fmfgraph*}}
```

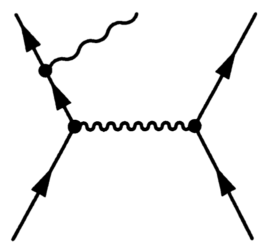


```
\NLO{.65}{-90}{__v1,__v2}{1}%
    {.7}{90}{__v2,__s}{} \qquad
\NLO{.1}{90}{__v1,__v2}{1}
    {.7}{90}{__v2,__s}{}
```



```
\NLO{.3}{90}{__v1,__v2}{1}%
    {.7}{-90}{__v1,__v2}{1} \qquad
\NLO{.3}{-60}{__b,__v1}{}%
    {.65}{-120}{__v1,__v2}{1}
```

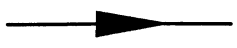
Figure 6.5: Next-to-Leading-Order $b \longrightarrow s\gamma$-diagrams.



```
\begin{fmfgraph}(40,30)
 \fmfleftn{i}{2}\fmfrightn{o}{2}\fmftop{g2}
 \fmf{phantom}{i1,v1,i2}
 \fmf{fermion}{o1,v2,o2}
 \fmf{uv_photon}{v1,v2}\fmffreeze
 \fmf{fermion}{i1,v1,g1,i2}
 \fmf{ir_photon,tension=0}{g1,g2}
 \fmfdot{g1,v1,v2}
\end{fmfgraph}
```

Figure 6.6: Different kinds of photons in the same diagram.

```
 \fmfset{arrow_len}{cm}\fmfset{arrow_ang}{25}
 \fmfleft{i}\fmfright{o}\fmf{fermion}{i,o}
\end{fmfgraph}
```



```
\begin{fmfgraph}(30,3)
 \fmfset{arrow_len}{cm}\fmfset{arrow_ang}{10}
 \fmfleft{i}\fmfright{o}\fmf{fermion}{i,o}
\end{fmfgraph}
```

How can I use Makefiles efficiently?

The most convenient way of using `feynMF` is to let the `make` utility figure out which commands have to be run. Since the `.mf`-file is created from the `.tex`-file every time LaTeX is run, even if nothing has changed, we need to aid `make` a little in the decision making process.

Below is a `Makefile` rule that supports `feynMF`. In lines 1–6, we set up a few variables to make site specific customizations easier: `MODE` is the printer type (can be found from the file `modes.mf`, the output of the `MakeTeXPK` script or from your local wizard). `MAG` is the magnification which is usually 1, but some LaTeX packages need other values:

the most prominent example is `seminar.sty`, which wants `magstep(4)`. `DPI` is the product of the printer resolution and `MAG`. Finally, `SUF` is a suffix that is used to build the METAFONT file name from the LaTeX file name (they can't share the same name, because the names of the `.log` files would clash).

```
 1 MODE = laserjet
 2 MAG = 1
 3 DPI = 300
 4 # MAG = magstep(4)
 5 # DPI = 622
 6 SUF = pics
 7 .tex:.dvi
 8   -cp $*$(SUF).mf $*$(SUF).mf~
 9   -latex $*
10   if cmp -s $*$(SUF).mf $*$(SUF).mf~; then\
11     :;\
12   else\
13     mf '\mode:=$(MODE);mag:=$(MAG);input '\
14       $*$(SUF);\
15     gftopk $*$(SUF).$(DPI)gf;\
16     latex $*;\
17   fi
18   while grep -s\
19     'Rerun to get cross-references right.'\
20     $*.log;\
21   do\
22     latex $*;\
23   done
```

Lines 8–23 define the sequence of commands that will produce a `.dvi`-file from the `.tex`-file (in the actual `Makefile`, the leading blanks have to be TABs, of course). Lines 8–9 create a copy of the `.mf`-file and run LaTeX (ignoring errors). Line 10 checks if the `.mf`-file has changed: if so, it is processed by METAFONT, the resulting `.gf`-file is turned into a `.pk`-file and LaTeX is run again (lines 13–16). The loop in lines 18–23 is independent from `feynMF` and runs LaTeX until all cross-references are resolved. It should be obvoius how to extend this rule to support BibTeX, `makeindex`, etc.

How can I make sure that LaTeX will find the correct fonts?

The answer is system dependent, unfortunately. However, if you use a variation on the above `Makefile` and make sure that your font search path includes the current directory, you should have no problems. Your TeX administrator knows how to set up which environment variables, if necessary.

Never, ever rely on the automatic font generation methods to run METAFONT for you on the `feynMF` files. They will install the output in a system directory (where it doesn't belong) and it might be hard to remove it again. Even worse, some installations will search the system directories first and you might be forever stuck with an obsolete version if you make changes to the diagrams. The same applies to running `gftopk`.

How can I make sure that a recipient will be able to format my preprint that uses `feynMF`?

`feynMF` has a compatible companion `feynmp.sty` that uses METAPOST [6] in place of METAFONT. Shipping the PostScript files produced by METAPOST and `feynmp.sty` with the preprint, makes it portable to all installations that support PostScript and doesn't require the recipient to run METAFONT. METAPOST can be installed easily by any competent TeX administrator and will be a default part of the next Unix TeX distribution.

However, the preferred option is to spread the gospel of `feynMF`, of course ...

6.1.5 Conclusions

I hope that this tutorial has eased the allegedly steep learning curve of `feynMF` a bit by providing a cookbook of examples from the "Real World". At the same time, I hope to have shown that the system is sufficiently general and can be extended to do things I have not thought about yet. *It is your turn now, go out, draw beautiful Feyman diagrams and calculate them or measure the corresponding cross sections.*

Bibliography

[1] T. Ohl, CERN Computer Newsletter **220**, pp. 22-28, June 1995.

[2] T. Ohl, CERN Computer Newsletter **221**, pp. 46-51, December 1995.

[3] T. Ohl, *feynMF user's manual*, 1995, TeX file accompanying the distribution.

[4] T. Ohl, Comp. Phys. Comm. **90**, 340 (1995).

[5] D. E. Knuth, *The METAFONTbook* (Addison-Wesley, Reading, MA, 1986).

[6] J. D. Hobby, Computer Science Report 162, AT&T Bell Laboratories (unpublished).