Contents

1 Introduction

2 Tutorial
  2.1 Loading the Package
  2.2 A First Diagram
  2.3 Adding Styles

3 Documentation
  3.1 Commands \\& Environments
  3.2 Keys \\& Styles

4 Examples

2.4 When the Algorithm Isn’t Enough

Index
1 Introduction

TikZ-Feynman provides a new way to draw Feynman diagrams in \( \LaTeX \) that does not rely on external programs and uses a clear extensible syntax.

If you use TikZ-Feynman in an academic setting, please cite:


Feynman diagrams provide a description of interactions of subatomic particles in a form that is clearer and more succinct than the corresponding mathematical description. They were introduced by Feynman and first appear in his paper ‘Space-Time Approach to Quantum Electrodynamics’ [2]. Since then, \( \LaTeX \) has become widely used to type-set scientific papers and currently, two leading methods of drawing Feynman diagrams in \( \LaTeX \) are feynMF/feynMP [3] and AxoDraw [4], with the latter also featuring a GUI front-end called JaxoDraw [5, 6].

Both feynMF/feynMP and AxoDraw have quite complicated syntax. As an example, the code to generate an \( \ell \ell \to \ell \ell \) scattering Feynman diagram in each package is:

AxoDraw
\begin{picture}
\ArrowLine(190,270)(160,300)
\ArrowLine(160,240)(190,270)
\ArrowLine(270,300)(240,270)
\ArrowLine(240,270)(270,240)
\Photon(190,270)(240,270){4}{4.5}
\Vertex(190,270){1.5}
\Vertex(240,270){1.5}
\end{picture}

feynMF/feynMP
\begin{fmfchar*}(40,30)
\fmfpen{thick}
\fmfleft{i1,i2}
\fmfright{o1,o2}
\fmf{fermion}{i1,v1,o1}
\fmf{fermion}{i2,v2,o2}
\fmf{photon,label=$q$}{v1,v2}
\fmfdot{v1,v2}
\end{fmfchar*}

\feynmandiagram \\
\hspace{1cm} [horizontal=a to b] \\
\hspace{1cm} {i1 -- [fermion] a -- [fermion] i2, a -- [photon] b, f1 -- [fermion] b -- [fermion] f2, ;}

The learning curves for both AxoDraw and feynMF/feynMP are quite steep, and although this is partly resolved by using JaxoDraw, this requires an external program. In addition, the creation of Feynman diagrams in feynMF/feynMP requires further processing outside of \( \LaTeX \).

TikZ-Feynman on the other hand uses a clear syntax and delegates the positioning of vertices to algorithms originally developed by Hu [7], and Peter and Kozo [8], both of which were implemented into TikZ by Pohlmann [9]. Since TikZ-Feynman is built on TikZ, users can harness the power and extensibility of TikZ [10] making it easy to extend to accommodate individual needs. In order to produce more complicated diagrams, relative or absolute positioning of vertices can also be used in TikZ-Feynman so that any diagram can be generated with relative ease.

TikZ-Feynman is made available through the Comprehensive \( \LaTeX \) Archive Network (ctan)\(^1\) and can also be downloaded from the project page\(^2\). The project is open source and contributions are welcome. The management of bugs and feature requests is done at Github\(^3\).

TikZ-Feynman’s versioning will approximately follow semantic versioning. This means that changes in the third number (1.0.0 to 1.0.1) will consist of bug fixes and very minor changes but they should not change the output otherwise\(^4\). Changes in the second number (1.0.0 to 1.1.0) will consist of new features but everything should be backwards compatible. Finally, changes in the first number (1.0.0 to 2.0.0) indicates a major change in the package and code written for 1.0.0 is not guaranteed to work on 2.0.0. The intended version of this package to use should be indicated when the package is loaded with \usepackage[compat=x.y.z]{tikz-feynman} so the user may be informed of any discrepancies. If needed, earlier versions may be downloaded from the project page\(^5\).

\(^1\)https://ctan.org/pkg/tikz-feynman
\(^2\)http://www.jpellis.me/projects/tikz-feynman
\(^3\)https://github.com/JP-Ellis/tikz-feynman
\(^4\)That is, with the exception of the bug that they are fixing.
\(^5\)http://www.jpellis.me/projects/tikz-feynman
Licence

This documentation may be redistributed and/or modified under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

The code of this package may be distributed and/or modified under the conditions of the LaTeX Project Public License, either version 1.3 of this license or (at your option) any later version.

This work has the LPPL maintenance status ‘maintained’.

The Current Maintainer of this work is Joshua Ellis.

This package is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgements

The original proof-of-concept for using Ti\kZ to draw Feynman diagrams was done by the user ‘Jake’ on the \TeX\ StackExchange. His original answer can be viewed at: http://tex.stackexchange.com/a/87395/26980.

I must also thank all the people who have used the development versions of Ti\kZ-Feynman and offered suggestions to improve it and found bugs for me to fix.
# Tutorial

## 2.1 Loading the Package

After installing the package, the TikZ-Feynman package can be loaded with `\usepackage{tikz-feynman}` in the preamble. It is recommended that you also specify the version of TikZ-Feynman to use with the optional package argument `compat`:

```
\usepackage[compat=1.1.0]{tikz-feynman}
```

This ensures that any new versions of TikZ-Feynman do not produce any undesirable changes without warning.

## 2.2 A First Diagram

Feynman diagrams can be declared with the `\feynmandiagram` command. It is analogous to the \tikz command from TikZ and requires a final semi-colon (\;\;) to finish the environment. For example, a simple s-channel diagram is:

```
\feynmandiagram [horizontal=a to b] {
    i1 -- [fermion] a -- [fermion] i2,
    a -- [photon] b,
    f1 -- [fermion] b -- [fermion] f2,
};
```

Let’s go through this example line by line:

**Line 1** `\feynmandiagram` introduces the Feynman diagram and allows for optional arguments to be given in the brackets `\{\langle \text{options} \rangle \}`. In this instance, `horizontal=a to b` orients the algorithm outputs such that the line through vertices `a` and `b` is horizontal.

**Line 2** The left fermion line is drawn by declaring three vertices (`i1`, `a`, and `i2`) and connecting them with edges `--`. Just like the \feynmandiagram command above, each edge also take optional arguments specified in brackets `\{\langle \text{options} \rangle \}`. In this instance, we want these edges to have arrows to indicate that they are fermion lines, so we add the `fermion` style to them.

As you will see later on, optional arguments can also be given to the vertices in exactly the same way.

**Line 3** This edge connects vertices `a` and `b` with an edge styled as a photon. Since there is already a vertex labelled `a`, the algorithm will connect it to a new vertex labeled `b`.

**Line 4** This line is analogous to line 2 and introduces two new vertices, `f1` and `f2`. It re-uses the previously labelled `b` vertex.

**Line 5** Finish the declaration of the Feynman diagram. The final semi-colon (`\;\;)` is important.

The name given to each vertex in the graph does not matter. So in this example, `i1`, `i2` denote the initial particles; `f1`, `f2` denotes the final particles; and `a`, `b` are the end points of the propagator. The only important aspect is that what we called `a` in line 2 is also `a` in line 3 so that the underlying algorithm treats them as the same vertex.

The order in which vertices are declared does not matter as the default algorithm re-arranges everything\(^1\). For example, one might prefer to draw the fermion lines all at once, as with the following example (note also that the way we named vertices is completely different):

\(^1\)It is possible for the algorithm to get a little confused in some circumstances, but these cases should be rather rare. For some algorithms (such as the layered layout), the order in which vertices are introduces does matter. This is documented in section 3.2.2.
As a final remark, the calculation of where vertices should be placed is usually done through an algorithm written in Lua. As a result, LuaTeX is required in order to make use of these algorithms. If LuaTeX is not used, TikZ-Feynman will default to a more rudimentary algorithm and will warn the user instead.

2.3 Adding Styles

So far, the examples have only used the photon and fermion styles. The TikZ-Feynman package comes with quite a few extra styles for edges and vertices which are all documented over in section 3. As an example, it is possible to add momentum arrows with momentum=<text>, and in the case of end vertices, the particle can be labelled with particle=<text>. As an example, we take the generic s-channel diagram from section 2.2 and make it a $e^+ e^- \rightarrow \mu^+ \mu^-$ diagram:

\begin{feynmandiagram} [horizontal=a to b] {
  i1 [particle=$e^-$] -- [fermion] a -- [fermion] i2 [particle=$e^+$],
  a -- [photon, edge label=$\gamma$, momentum'={$k$}] b,
  f1 [particle=$\mu^+$] -- [fermion] b -- [fermion] f2 [particle=$\mu^-$],
};

In addition to the style keys documented below, style keys from TikZ can be used as well:

\begin{feynmandiagram} [horizontal=a to b] {
  i1 [particle=$e^-$, very thick] -- [fermion] a -- [fermion] i2 [particle=$e^+$],
  a -- [red, photon, edge label=$\gamma$, momentum'={$k$}] b,
  f1 [particle=$\mu^+$, opacity=0.2] -- [fermion] b -- [fermion] f2 [particle=$\mu^-$],
};
For a list of all the various styles that TikZ provides, have a look at the TikZ manual; it is extremely thorough and provides many usage examples.

TikZ-Feynman also supports combining styles together which can be useful in certain new models such as supersymmetry:

\begin{feynmandiagram} [horizontal=a to b]
\vertex [particle=\(\tilde W\)] (i1)
\vertex [anti fermion] (i2)
\vertex [charged scalar, edge label=\(\tilde q\)] (b)
\vertex [fermion, particle=\(q\)] (f1)
\diagram {i1 -- [boson] a -- [anti fermion] i2 -- [\(q\)], a -- [gluon] b -- [ fermion] [\(q\)], f1 -- [gluon] b}.
\end{feynmandiagram}

2.4 When the Algorithm Isn’t Enough

By default, the \feynmandiagram and \diagram commands use the spring layout algorithm to place all the edges. The spring layout algorithm attempts to ‘spread out’ the diagram as much as possible which—for most simpler diagrams—gives a satisfactory result; however in some cases, this does not produce the best diagram and this section will look at alternatives. There are three main alternatives:

**Add invisible edges** While still using the default algorithm, it is possible to force certain vertices to be closer together by adding extra edges and making them invisible through draw=none. The algorithm will treat these extra edges in the same way, but they are simply not drawn at the end;

**Use a different algorithm** In some circumstances, other algorithms may be better suited. Some of the other graph layout algorithms are listed in section 3.2.2, and an exhaustive list of all algorithms and their parameters is given in the TikZ manual;

**Manual placement** As a last resort, very complicated or unusual diagrams will require each vertex to be manually placed.

2.4.1 Invisible Edges

The underlying algorithm treats all edges in exactly the same way when calculating where to place all the vertices, and the actual drawing of the diagram (after the placements have been calculated) is done separately. Consequently, it is possible to add edges to the algorithm, but prevent them from being drawn by adding draw=none to the edge style.

This is particularly useful if you want to ensure that the initial or final states remain closer together than they would have otherwise as illustrated in the following example (note that opacity=0.2 is used instead of draw=none to illustrate where exactly the edge is located).

\begin{feynmandiagram} [horizontal=a to b]
\vertex [particle=\(\tilde W\)] (i1)
\vertex [anti fermion] (i2)
\vertex [charged scalar, edge label=\(\tilde q\)] (b)
\vertex [fermion, particle=\(q\)] (f1)
\diagram {i1 -- [boson] a -- [anti fermion] i2 -- [\(q\)], a -- [gluon] b -- [ fermion] [\(q\)], f1 -- [gluon] b}.
\end{feynmandiagram}

\footnote{For more details on this layout and any other graph layouts available, see section 3.2.2}
2.4.2 Alternative Algorithms

The graph drawing library from TikZ has several different algorithms to position the vertices. By default, \texttt{\textbackslash{diagram}} and \texttt{\textbackslash{feynmandiagram}} use the spring layout algorithm to place the vertices. The spring layout attempts to spread everything out as much as possible which, in most cases, gives a nice diagram; however, there are certain cases where this does not work. A good example where the spring layout doesn’t work are decays where we have the decaying particle on the left and all the daughter particles on the right.

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{example.png}
\caption{Example diagram with alternative algorithms.}
\end{figure}

\footnote{See section 3.2.2 for some alternative algorithms.}

\begin{verbatim}
% No invisible to keep the two photons together
\feynmandiagram [small, horizontal=a to t1] {
  a [particle=\pi^0] -- [scalar] t1 -- t2 -- t3 -- t1,
  t2 -- [photon] p1 [particle=\gamma],
  t3 -- [photon] p2 [particle=\gamma],
};

% Invisible edge ensures photons are parallel
\feynmandiagram [small, horizontal=a to t1] {
  a [particle=\pi^0] -- [scalar] t1 -- t2 -- t3 -- t1,
  t2 -- [photon] p1 [particle=\gamma],
  t3 -- [photon] p2 [particle=\gamma],
  p1 -- [opacity=0.2] p2,
};
\end{verbatim}
You may notice that in addition to adding the \texttt{layered layout} style to \texttt{feynmandiagram}, we also changed the order in which we specify the vertices. This is because the \texttt{layered layout} algorithm does pay attention to the order in which vertices are declared (unlike the default \texttt{spring layout}); as a result, \texttt{c--f2, c--f3} has a different meaning to \texttt{f2--c--f3}. In the former case, \texttt{f2} and \texttt{f3} are both on the layer below \texttt{c} as desired; whilst the latter case places \texttt{f2} on the layer above \texttt{c} (that, the same layer as where the $W^-$ originates).

2.4.3 Manual Placement

In more complicated diagrams, it is quite likely that none of the algorithms work, no matter how many invisible edges are added. In such cases, the vertices have to be placed manually. \texttt{TikZ-Feynman} allows for vertices to be manually placed by using the \texttt{vertex} command.

The \texttt{vertex} command is available only within the \texttt{feynman} environment (which itself is only available inside a \texttt{tikzpicture}). The \texttt{feynman} environment loads all the relevant styles from \texttt{TikZ-Feynman} and declares additional \texttt{TikZ-Feynman}-specific commands such as \texttt{vertex} and \texttt{diagram}. This is inspired from PGFPlots and its use of the \texttt{axis} environment.

The \texttt{vertex} command is very much analogous to the \texttt{node} command from \texttt{TikZ}, with the notable exception that the vertex contents are optional; that is, you need not have \texttt{\{⟨text⟩\}} at the end. In the case where \texttt{(}) is specified, the vertex automatically is given the \texttt{particle} style, and otherwise it is a usual (zero-sized) vertex.

To specify where the vertices go, it is possible to give explicit coordinates though it is probably easiest to use the \texttt{positioning} library from \texttt{TikZ} which allows vertices to be placed relative to existing vertices\footnote{The \texttt{TikZ manual} has some extensive documentation explaining how to use the \texttt{positioning library}.}. By using relative placements, it is possible to easily tweak one part of the graph and everything will adjust accordingly—the alternative being to manually adjust the coordinates of every affected vertex.

Finally, once all the vertices have been specified, the \texttt{diagram*} command is used to specify all the edges. This works in much the same way as \texttt{diagram} (and also \texttt{feynmandiagram}), except that it uses an very basic algorithm to place new nodes and allows existing (named) nodes to be included. In order to refer to an existing node, the node must be given in parentheses.

This whole process of specifying the nodes and then drawing the edges between them is shown below for the muon decay:
\begin{tikzpicture}
\begin{feynman}
\vertex (a) {\(\mu^{-}\)};
\vertex [right=of a] (b);
\vertex [above right =of b] (f1) {\(\nu_{\mu}\)};
\vertex [below right =of b] (c);
\vertex [above right =of c] (f2) {\(\overline{\nu}_{e}\)};
\vertex [below right =of c] (f3) {\(e^{-}\)};
\diagram* {[fermion] (a) -- (b) -- (f1),
[boson, edge label' =\(W^{-}\)] (b) -- (c),
[anti fermion] (c) -- (f2),
[fermion] (c) -- (f3),
};
\end{feynman}
\end{tikzpicture}
3 Documentation

3.1 Commands & Environments

\tikzfeynmanset{⟨options⟩}

This command will process ⟨options⟩ using \pgfkeys with the default path set to /tikzfeynman. Typically, ⟨options⟩ will be a comma-separated list of the form ⟨key⟩=⟨value⟩, though the full power of the mechanism behind \pgfkeys can be used (see the TikZ manual for a complete description).

Typically, this is used in the preamble of the document to add or change certain keys for the whole document.

\feynmandiagram{⟨TikZ options⟩}{⟨diagram options⟩}{⟨diagram instructions⟩}

This command creates a \{tikzpicture\} and \{feynman\} environment, and places a \{diagram\} inside with the provided ⟨diagram instruction⟩. Please refer to the documentation for \{diagram\} for the ⟨diagram instruction⟩ syntax.

The optional arguments specified in ⟨tikz options⟩ are passed on to the \{tikzpicture\}, and the ⟨diagram options⟩ are passed on to \{diagram\}. If only one optional argument is given, then the optional arguments are given to both. A single optional argument will usually suffice as most keys are recognized by both commands; however, in the event that a key is not recognized, both options are provided.

\begin{feynman}{⟨options⟩}
⟨environment contents⟩
\end{feynman}

The \{feynman\} environment is where all the drawing of Feynman diagrams takes place. It makes all the TikZ-Feynman styles available and defines commands such as \vertex and \diagram which are otherwise unavailable outside of this environment. The \{feynman\} environment is only accessible within the \{tikzpicture\} environment.

Options which are passed in ⟨options⟩ apply for the whole environment in the same way that the \{scope\} environment work in TikZ.

\vertex{⟨options⟩}{⟨name⟩}{at}{⟨coordinate⟩}{⟨contents⟩};

Defines a new vertex with the provided ⟨name⟩. If ⟨contents⟩ is not provided, the resulting vertex will have zero size. On the other hand, if ⟨contents⟩ is provided, the \texttt{particle}={⟨contents⟩} style is applied. Additional styles can be applied to the vertex through ⟨options⟩.

The final semicolon (;) is vital for this command since without it, the \LaTeX{} engine will not know when the \vertex command ends. Additionally, this command \texttt{cannot} be chained like one can do with the inbuilt TikZ commands.

This command is only available with the \{feynman\} environment.

\begin{feynman}{⟨options⟩}{⟩diagram instructions\}
\end{feynman}

Begins a new diagram using the \texttt{spring} layout. Keys passed through ⟨options⟩ can include general TikZ keys, graph-specific keys and any applicable TikZ-Feynman keys too. Other algorithms (such as \texttt{tree layout}) can be passed through ⟨options⟩ and that will override the \texttt{spring layout}.

The syntax for the ⟨diagram instructions⟩ is thoroughly described in the TikZ manual, but in the context of this package, it will usually suffice to know the following:

• Vertices within the graph are specified with no delimiters (i.e. no parenthesis, no brackets) and only require spaces around either side. In order to refer to a vertex defined outside of the \{diagram\} command, its name must be given in parenthesis: ⟨⟨name⟩⟩. Note that in order to refer to external
vertices, one must use \texttt{\textbackslash diagram*} as most algorithms (including the default \texttt{spring layout}) are incompatible with vertices defined outside of the algorithm. When a vertex name is used multiple times, the underlying algorithm will consider them to be the same vertex and introduces additional edges.

Options can be given to the vertex in brackets after the name: \texttt{\textbackslash name \{\texttt{options}\}}. For vertices defined outside of the \texttt{\textbackslash diagram} command, these options should be specified when the vertex is first declared.

- The edges between each pair of vertices is specified with \texttt{--}, and these can be chained together: \texttt{a -- b -- c}. In order to pass a style to the edge, it is specified in brackets after the dashed: \texttt{-- \{\texttt{options}\}}. For example, to make an edge red, one would use \texttt{--[red]}.
- A comma (,)—or equivalently a semicolon (;)—specifies the end of a sequence of edges and vertices and allows for another sequence to be started. So \texttt{a -- b, c -- d} will create two disconnected edges.
- Subgroups (akin to scopes in Ti\textit{k}Z) are specified with braces: \texttt{\{\texttt{options}\}\texttt{\{diagram instructions\}}}.

This can be quite useful when a lot of edges or nodes share a common style. For example, one could use \texttt{\{\texttt{edges=fermion\}}\ a -- b -- c, x -- y -- z\}} and every edge will have the \texttt{fermion} style applied automatically.

Another useful feature of subgroups is that an edge to a group will create an edge to each vertex in that subgroup as shown below. The example also shows how they can be nested which in some cases (such as with a \textit{layered layout}) can be extremely useful.

\begin{center}
\begin{tikzpicture}
\begin{feynmandiagram}[\texttt{nodes=circle, horizontal=a1 to b3}]
\diagram* \{
\feynmandiagram*\{\texttt{nodes=circle, horizontal=a1 to b3}\} {
 a1 -- (b1, b2, b3 -- (c1, c2 -- d1))
};
\end{feynmandiagram}
\end{tikzpicture}
\end{center}

\texttt{\textbackslash diagram*\{\texttt{options}\}\{\texttt{diagram instructions}\}}

Same as \texttt{\textbackslash diagram}, but instead of using the \texttt{spring layout} algorithm to place the vertices, this uses the most basic algorithm. This basic algorithm in most cases will not produce a satisfactory diagram, but is intended to be used with vertices are declared and positioned outside of the \texttt{\textbackslash diagram*} command. Essentially, \texttt{\textbackslash diagram*} should be used only to connect existing vertices.

### 3.2 Keys & Styles

The various styles and options that allow the Feynman diagrams to be customized are defined in what Ti\textit{k}Z calls keys. The documentation includes all keys which are defined within Ti\textit{k}Z-Feynman which all begin with the prefix \texttt{/tikz\_feynman}. In addition, a few of the keys from Ti\textit{k}Z itself which are particularly useful to Ti\textit{k}Z-Feynman are documented and these are prefixed with \texttt{/tikz} or \texttt{/graph\_drawing}. Please refer to the Ti\textit{k}Z manual for a more in thorough documentation of the Ti\textit{k}Z keys.

Ti\textit{k}Z-Feynman provides many \texttt{every } (key) which allow every instance of (key) to be modified. For example, to make every diagram red except for small diagrams which should be blue, then one would add to the preamble:
If you are completely unhappy with a particular inbuilt style, you can define your own style with 
\(\langle\text{key}\rangle/.\text{style}=\langle\ldots\rangle\) as shown in the following example:

\[
\text{\tikzfeynmanset{}
  myblob/.style={
    shape=circle,
    draw=blue,
    fill=red,
  }
}
\]

All the keys defined here are made available inside the \{feynman\} environment and inside \feynmandiagram, but if you wish to access them outside of this (say, in a regular \{tikzpicture\} environment), you will need to specify the full path with the leading /tikzfeynman.

### 3.2.1 Feynman Keys

- `/tikzfeynman/execute at begin feynman={⟨TEX code⟩}` (no default)
  - Allows for custom code to be executed at the start or end of each \{feynman\} environment.

- `/tikzfeynman/execute at end feynman={⟨TEX code⟩}` (no default)
  - Allows for custom code to be executed at the start or end of each \{feynman\} environment.

- `/tikzfeynman/every feynman={⟨styles⟩}` (no default)
  - Set of styles which are applied to every \{feynman\} environments (and consequently, every apply inside all \feynmandiagram too). The style also applies to regular TikZ commands used inside the \{feynman\} environment.

\[
\text{\tikzfeynmanset{every feynman={red}}}
\begin{tikzpicture}
  \node at (0, 0.5) {This is not red};
  \begin{feynman}
    \node at (0, -0.5) {This is red};
  \end{feynman}
\end{tikzpicture}
\]

- `/tikzfeynman/inline=⟨node⟩` (style, no default)
  - A style used to display a Feynman diagram inline (typically in an equation), and aligning such that its vertical placement is at the node specified. The node specification must enclosed in parentheses. For nodes which contain text (such as when the particle style is applied), it is possible to use the baseline of the text inside the node to line up with the baseline of the equation by using \((⟨node⟩).\text{base}) as demonstrated in the following example. Note that this key applies additional styles to make the diagram fit in an equation more nicely; if you do not wish to have these additional styles, use the `baseline` key.

\[
\gamma 
\begin{tikzpicture}
  \node at (0, 0.5) {This is not red};
  \begin{feynman}
    \node at (0, -0.5) {This is red};
  \end{feynman}
\end{tikzpicture}
\]

\[
\gamma = ig_c \gamma^\mu
\]
\begin{equation}
\feynmandiagram [inline=(d.base), horizontal=d to b] {
    a -- [fermion] b -- [fermion] c,
    b -- [boson] d [particle=$\gamma$],
};
\end{equation}

$= i g_e \gamma^\mu$

/tikz/baseline=⟨node⟩

Changes the vertical alignment of the Feynman diagram such that it diagram’s baseline is at the node specified. This works in the same was as inline=⟨node⟩, but it does not apply any additional styles (notice how the following example is larger than the one above).

\begin{equation}
\feynmandiagram [baseline=(d.base), horizontal=d to b] {
    a -- [fermion] b -- [fermion] c,
    b -- [boson] d [particle=$\gamma$],
};
\end{equation}

= i g_e \gamma^\mu

/ggraph drawing/horizontal=⟨node⟩ to ⟨node⟩

The underlying algorithm will arrange all the nodes relative to each other, but beyond that it has no idea how the overall graph should be oriented. By using one of the above keys, the final output of the algorithm is oriented and/or mirrored such that the two nodes specified are on the same horizontal (or vertical) line. The two nodes need not actually be connected by an edge for this to work.

The two ⟨node⟩ specifications should not be enclosed in parentheses, unlike the inline and baseline keys. The horizontal' and vertical' keys work in the same was as horizontal and vertical, but with a flip.
3.2.2 Diagram Keys

/\texttt{tikzfeynman/every\ diagram=}(\texttt{styles}) \hspace{1cm} (no\ default)

Set of styles which are applied to every diagram; that is, to everything inside the $\texttt{feynmandiagram}$, $\texttt{diagram}$ and $\texttt{diagram*}$ commands but not the general $\texttt{feynman}$ environment (see section 3.2.1 for that).

\begin{verbatim}
/\texttt{tikzfeynmanset}{every\ diagram=red}
\feynmandiagram[\texttt{small}, horizontal=d to b] { 
a -- [fermion] b -- [fermion] c, 
b -- [boson] d, 
};
\end{verbatim}

/\texttt{tikzfeynman/small} \hspace{1cm} (no\ value)
/\texttt{tikzfeynman/medium} \hspace{1cm} (no\ value)
/\texttt{tikzfeynman/large} \hspace{1cm} (no\ value)

Changes the default separation between the vertices and changes the size of arrows, blobs, and other shapes to fit different context. The \texttt{small} size is best used with when the diagram is quite simple and doesn’t have too many annotations (such as momentum arrows and particle labels). The \texttt{medium} size is the default and is usually large enough that even diagrams with many labels and momentum arrows do not become too cluttered. Finally the \texttt{large} key is best for large illustrations as used on the title page of this document.
There are several algorithms which are available to place the vertices which are all provided within the graph drawing library from TikZ. Below are listed a few of these layouts which are more relevant for drawing Feynman diagrams. For a more complete description of how these algorithm work, please refer to the TikZ manual. Please note that most graph drawing algorithms are implemented in Lua and thus require LuaTEX in order to work. If LuaTEX is not used, TikZ-Feynman will default to a more rudimentary algorithm and will warn the user instead.

\texttt{/graph drawing/spring layout=(string)} \hfill (no default)

Uses Hu’s spring layout [7] as implemented by Pohlmann [9]. This models each edge as springs and attempts to spread everything out as much as possible. This is the default layout.

\[\feynmandiagram [baseline=(b), small, horizontal=d to b, red] {\node (a) at (0,0) \{a\}; \node (b) at (1,1) \{b\}; \node (c) at (2,0) \{c\}; \node (d) at (3,1) \{d\}; \node (e) at (4,0) \{e\}; \node (f) at (5,1) \{f\}; a -- [fermion] b -- [fermion] c, b -- [boson] d,};\]

\texttt{/graph drawing/spring electrical layout=(string)} \hfill (no default)

Uses Hu’s spring electrical layout [7] as implemented by Pohlmann [9]. This models each edge as springs and gives each vertex a charge. This algorithm allows for the charge of a particular vertex to be adjusted using the \texttt{electric charge} key (the default is 1).

\[\feynmandiagram [nodes=circle, small, horizontal=c to d] {\node (a) at (0,0) \{a\}; \node (b) at (1,1) \{b\}; \node (c) at (2,0) \{c\}; \node (d) at (3,1) \{d\}; \node (e) at (4,0) \{e\}; \node (f) at (5,1) \{f\}; a -- [fermion] b -- [fermion] c, b -- [boson] d,}\]

\texttt{/graph drawing/layered layout=(string)} \hfill (no default)

Uses the Sugiyama layout algorithm [8] as implemented by Pohlmann [9] in order to place the node.
When an edge is specified, the first vertex is always located on the layer above the second vertex. This creates a hierarchy of vertices which is particularly useful for decays.

Two vertices can be forced to be on the same layer with the `\graphdrawing/same layer` key.

\begin{feynmandiagram}[nodes=\text{circle}, \text{small}, horizontal=\text{a to b}, \text{layered layout}] (a -- b -- (c, d -- (\text{e, f})),
\end{feynmandiagram}

\begin{feynmandiagram}[nodes=\text{circle}, \text{small}, horizontal=\text{a to b}, \text{layered layout}] (a -- b -- (c -- \{c1, c2\}, d -- \{d1, d2\}), \text{[same layer]} \text{c1, d},)
\end{feynmandiagram}

\text{\texttt{\graphdrawing/tree layout=\textit{(string)}}} \quad \text{(no default)}

Uses the Reingold–Tilford algorithm in order to place the node. This works in a similar way to the layered layout, but has quite a lot of additional options to handle missing children in the tree. Please refer to the TikZ manual for a thorough description of these additional features.

When an edge is specified, the first vertex is always located on the layer above the second vertex. This creates a hierarchy of vertices which is particularly useful for decays.

\begin{feynmandiagram}[nodes=\text{circle}, \text{small}, horizontal=\text{a to b}, \text{tree layout}] (a -- b -- (c, d -- (\text{e, f})),
\end{feynmandiagram}
Just like it is possible to change the shape of every vertex or edge in the whole document, it is also possible to change the shape of every vertex or edge in a single diagram by modifying these keys and adding the desired styles.

3.2.3 Vertex Keys

/tikzfeynman/vertex

The default, base style applied to every vertex initially. Other styles are subsequently added. This sets the vertex shape to be a coordinate, that is, a null shape with no size or width.

/tikzfeynman/every (vertex shape)=⟨styles⟩

The style of specific vertices can be modified by changing the appropriate every ⟨vertex shape⟩ key. For example, in order to change the style of every dot-styled vertex, you can modify the every dot key, or to modify every vertex globally, the every vertex key can be modified.

/tikzfeynman/dot

Modifies the vertex so that it has a small filled circle.
/tikzfeynman/square dot (no value)
Modifies the vertex so that it has a small filled square.

\feynmandiagram [small] {
  a -- b [square dot] -- (c, d)
};

/tikzfeynman/empty dot (no value)
Modifies the vertex so that it has a small empty circle.

\feynmandiagram [small] {
  a -- b [empty dot] -- (c, d)
};

/tikzfeynman/crossed dot (no value)
Modifies the vertex so that it has a small circle with a cross inside.

\feynmandiagram [small] {
  a -- b [crossed dot] -- (c, d)
};

/tikzfeynman/blob (no value)
Modifies the vertex so that it is a large blob, usually used to denote an effective operator.

\feynmandiagram [small] {
  a -- b [blob] -- (c, d)
};

/tikzfeynman/particle=(name) (no default)
Modifies the vertex so that it shows ⟨name⟩. This is intended to label initial and final particles, but it should not be used on internal vertices as it will result in the lines at the vertex having a gap. For propagators (the edge label key is much more appropriate).

Note that if ⟨name⟩ contains characters such as brackets ([]) or commas (,), the whole ⟨name⟩ has to be enclosed in braces ⟨{⟩}; otherwise, the parser will interpret the comma as the end of the ⟨name⟩ and the start of the next key, or the closing bracket as the end of all optional arguments.
3.2.4 Edge Keys

Just like with the various vertex keys, each edge type has a corresponding every (edge type); however, due to the existence of very similar keys such as scalar, charged scalar and anti charged scalar, more specific keys inherit styles from less specific ones. For example, styles in every scalar will apply to charged scalar and anti charged scalar.

The edge styles do not override each other and instead stack. This means that an edge styled with both boson and fermion will consist of a straight line with a wavy line on top and an error. If not styles are provided, then a simple solid line is drawn.

\feyndiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};

\feyndiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};

\tikzfeynman/every edge={green},
\tikzfeynman/every boson={red},
\tikzfeynman/every photon={blue},
\}
\feynmandiagram [small] {
    a [particle={\(a\})] -- [boson] o -- [photon] b [particle={\(b\)}],
    f1 [particle={\(c\})] -- [fermion] o -- [scalar] f2 [particle={\(d\})],
};

\feyndiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};

\feyndiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};

\tikzfeynman/plain
\}
\feynmandiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};

\feyndiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};

\feyndiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};

\feyndiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};

\tikzfeynman/boson
\}
\feynmandiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};

\feyndiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};

\feyndiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};

\feyndiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};

\feyndiagram [small, horizontal=a to b] {
    a [particles={\gamma, Z}] -- [boson] b -- (c, d),
};
/tikzfeynman/charged boson

Draws a sinusoidal line with an arrow to denote a charged boson.

```
feynmandiagram [horizontal=a to b] { a -- [charged boson] b};
```

/tikzfeynman/anti charged boson

Draws a sinusoidal line with an arrow pointing the other way to denote an anti charged boson.

```
feynmandiagram [horizontal=a to b] { a -- [anti charged boson] b};
```

/tikzfeynman/photon

Draws a sinusoidal line to denote a photon.

```
feynmandiagram [horizontal=a to b] { a -- [photon] b};
```

/tikzfeynman/scalar

Draws a dashed line to denote a scalar.

```
feynmandiagram [horizontal=a to b] { a -- [scalar] b};
```

/tikzfeynman/charged scalar

Draws a dashed line with an arrow to denote a charged scalar.

```
feynmandiagram [horizontal=a to b] { a -- [charged scalar] b};
```

/tikzfeynman/anti charged scalar

Draws a dashed line with an arrow pointing the other way to denote a charged scalar antiparticle.

```
feynmandiagram [horizontal=a to b] { a -- [anti charged scalar] b};
```

/tikzfeynman/ghost

Draws a dotted line to denote a ghost.

```
feynmandiagram [horizontal=a to b] { a -- [ghost] b};
```

/tikzfeynman/fermion

Draws a solid line with an arrow to denote a fermion.

```
feynmandiagram [horizontal=a to b] { a -- [fermion] b};
```
/tikzfeynman/anti fermion

Draws a solid line with an arrow pointing the other way to denote an antifermion.

\feynmandiagram [horizontal=a to b] { a -- [anti fermion] b};

/tikzfeynman/majorana

Draws a solid line with two arrows pointing to the center to denote a Majorana particle.

\feynmandiagram [horizontal=a to b] { a -- [majorana] b};

/tikzfeynman/anti majorana

Draws a solid line with two arrows pointing to the ends to denote a Majorana particle.

\feynmandiagram [horizontal=a to b] { a -- [anti majorana] b};

/tikzfeynman/gluon

Draws a coiled line to denote a gluon.

\feynmandiagram [horizontal=a to b] { a -- [gluin] b};

/tikz/edge label =⟨text⟩(no default)
/tikz/edge label' =⟨text⟩(no default)

Places a label halfway along the edge with the given text. The primed key switches which side of the edge the label is placed.

/tikzfeynman/insertion=[⟨options⟩]⟨distance⟩(no default)

Places an insertion (for mass or momentum insertion) along an edge. The distance specifies how far along the edge the insertion should be placed such that 0 and 1 respectively correspond to the start and the end of the edge.

Multiple insertions can be placed along a single edge by repeating the style key. Through the ⟨options⟩ argument, the insertion size and style can be changed.

\feynmandiagram [horizontal=a to b] { a -- [insertion=0.33, insertion=0.67] b};

/tikzfeynman/insertion/size=⟨distance⟩(no default, initially 3pt)

Specifies how big the insertion should be. The length of each edge starting from the center will be $\sqrt{2} \times$ ⟨distance⟩.

\feynmandiagram [horizontal=a to b] { a -- [insertion={size=10pt}0.4] b};

/tikzfeynman/insertion/style=⟨distance⟩(no default, initially empty)

Specifies additional styles to applying to the lines of the insertion.

\feynmandiagram [horizontal=a to b] { a -- [insertion={style=red}0.4] b};
3.2.5 Momentum Keys

\begin{verbatim}
\tikzfeynman/momentum=[[\langle\text{options}\rangle]\langle\text{text}\rangle] (no default)
\tikzfeynman/momentum'=[[\langle\text{options}\rangle]\langle\text{text}\rangle] (no default)
\tikzfeynman/reversed momentum=[[\langle\text{options}\rangle]\langle\text{text}\rangle] (no default)
\tikzfeynman/reversed momentum'=[[\langle\text{options}\rangle]\langle\text{text}\rangle] (no default)
\tikzfeynman/rmomentum=[[\langle\text{options}\rangle]\langle\text{text}\rangle] (no default)
\tikzfeynman/rmomentum'=[[\langle\text{options}\rangle]\langle\text{text}\rangle] (no default)
\end{verbatim}

Places a momentum arrow on the specified edge with label given by \langle text \rangle. The primed (') version place the momentum arrow on the other side of the edge; that is, if the momentum arrow was on the right, it will be placed on the left of the edge. The reversed momentum and reversed momentum' keys are analogous to momentum and momentum' except that the momentum arrow points in the opposite direction. Finally, the rmomentum and rmomentum' are aliases of reversed momentum and reversed momentum'.

Note that due to the way the arrow is drawn, it doesn’t inherit styles of the edge. As a result, they have to be re-specified through \langle options \rangle.

\begin{feynmandiagram}
layered layout, horizontal=a to b {
  a -- [red, fermion, edge label=\langle ab \rangle, momentum=\langle arrow style=red \rangle \langle p_{ab} \rangle] b
  -- [blue, photon, edge label=\langle bc \rangle] c
  -- [green!40!black, scalar, momentum=\langle p_{cd} \rangle] d,
}\end{feynmandiagram}

\begin{feynmandiagram}
layered layout, horizontal=b to c {
  a -- [photon, momentum=\langle p \rangle] b
  -- [fermion, half left, looseness=1.5, momentum=\langle k \rangle] c
  -- [fermion, half left, looseness=1.5, momentum=\langle k-p \rangle] b,
  c -- [photon, momentum=\langle p \rangle] d,
}\end{feynmandiagram}

The \langle options \rangle allows for the following options to customize the appearance of the momentum arrows. These can be either set globally with the usual \texttt{\tikzfeynmanset}, or can be set on a individual basis through the \langle options \rangle argument of the momentum key. In the latter case, only the \langle key \rangle in /tikzfeynman/momentum/\langle key \rangle is required.

\begin{verbatim}
\tikzfeynman/momentum/label distance=\langle distance \rangle (no default, initially 0pt)
\end{verbatim}

Set the separation between the text and the arrow. Note that the text is still surrounded by an inner sep=0.3333em by default so the default distance of 0pt will not result in the momentum label touching the arrow.
3.2.6 Modifier Keys

Modifier keys serve only to slightly modify a small feature of the edge.

\begin{verbatim}
/tikzfeynman/half left
/tikzfeynman/half right
/tikzfeynman/quarter left
/tikzfeynman/quarter right
\end{verbatim}

Modifies the edge so that it bends left or right in such a way that it completes half a circle, or a quarter of a circle.

\begin{verbatim}
\feynmandiagram [horizontal=a to b] {
 a -- [red, fermion, half left] b -- [blue, fermion, half left] a,
};
\end{verbatim}

\begin{verbatim}
\feynmandiagram [horizontal=a to c] {
 a -- [red!0!blue, fermion, quarter left] b
 -- [red!33!blue, fermion, quarter left] c
 -- [red!66!blue, fermion, quarter left] d
 -- [red!100!blue, fermion, quarter left] a,
};
\end{verbatim}

\begin{verbatim}
/tikz/out=(angle) (no default)
/tikz/in=(angle) (no default)
\end{verbatim}

Specifies the angle at which the edge leaves the first vertex in an edge and the angle at which it enters the second vertex in an edge.
/tikz/relative=(true or false)  
If relative is set to false, the angle is relative to the paper whilst when relative is set to true, the angle is relative to the straight line joining the two vertices.

/tikz/looseness=(number)  
As the name suggests, this specifies how 'loose' or 'tight' a curve is connecting two vertices.
4 Examples

Below are a few diagrams which demonstrate how the package can be used in some more practical examples.

**Vertex Rule**

\[
\begin{aligned}
\text{\feynmandiagram & horizontal=a to b & \} \\
\text{a [particle=$Z$]} & \text{-- [photon, momentum=$p_1$]} & b, \\
\text{f1 [particle=$\overline{f}$]} & \text{-- [fermion, momentum=$p_3$]} & b, \\
\text{-- [fermion, momentum=$p_2$]} & \text{f2 [particle=$f$],}
\end{aligned}
\]

**Tree Level Diagrams**

\[
\begin{aligned}
\text{\feynmandiagram & horizontal=a to b & \} \\
\text{i1 [particle=$e^{-}$]} & \text{-- [fermion]} & a -- [fermion] i2 [particle=$e^{+}$], \\
a -- [photon, edge label=$\gamma$] & b, \\
f1 [particle=$\mu^{-}$] & \text{-- [fermion]} & b -- [fermion] f2 [particle=$\mu^{+}$], \\
\end{aligned}
\]

\[
\begin{aligned}
\text{\feynmandiagram & vertical=a to b & \} \\
i1 [particle=$e^{-}$] & \text{\} -- [fermion] a \\
-- [fermion] f1 [particle=$e^{+}$], \\
a -- [photon, edge label=$\gamma$] & b, \\
i2 [particle=$e^{+}$] & \text{\} -- [anti fermion] b \\
-- [anti fermion] f2 [particle=$e^{-}$],
\end{aligned}
\]
Loops

\begin{tikzpicture}
\begin{feynman}
\diagram [vertical'=a to b] {
  i1 [particle=\(e^{-}\)]
  -- [fermion] a
  -- [draw=none] f1 [particle=\(e^{+}\)] b,
  i2 [particle=\(e^{+}\)]
  -- [anti fermion] b
  -- [draw=none] f2 [particle=\(e^{-}\)],
};
\end{feynman}
\end{tikzpicture}

\begin{tikzpicture}
\feynmandiagram [layered layout, horizontal=b to c] {
  a -- [photon, momentum=\(p\)] b
  -- [fermion, half left, momentum=\(k\)] c
  -- [fermion, half left, momentum=\(k-p\)] b,
  c -- [photon, momentum=\(p\)] d,
};
\end{tikzpicture}

\begin{tikzpicture}
\feynmandiagram [layered layout, horizontal=a to b] [edges=gluon] {
  \{i1, i2\} -- a -- [half left] b -- [half left] a,
  b -- [f1, f2],
};
\end{tikzpicture}
Box Diagrams

\textbf{Box Diagrams}

\begin{feynmandiagram}[layered layout, horizontal=a to b] {
    % Draw the top and bottom lines
    i1 [\text{particle} = \langle d \rangle]
    \node {a} [\text{fermion}, \text{edge label} = W^{-}]
    \node {b} \node {f1} [\text{particle} = \langle \mu^{-} \rangle],
    i2 [\text{particle} = \langle \overline{s} \rangle]
    \node {c} [\text{anti fermion}]
    \node {d} [\text{photon}, \text{edge label} = W^{+}],
    \node {f2} [\text{particle} = \langle \mu^{+} \rangle],
    % Draw the two internal fermion lines
    \node {g} [\text{same layer}] a -- \node {q} [\text{fermion}, \text{edge label} = q] c,
    \node {v_\mu} [\text{same layer}] b -- \node {\nu_{\mu}} [\text{anti fermion, edge label} = \nu_{\mu}] d,
};
Meson decay and mixing

\begin{tikzpicture}
\begin{feynman}
\vertex (a1) {$\overline{b}$};
\vertex [right=1.5cm of a1] (a2);
\vertex [right=1cm of a2] (a3);
\vertex [right=1.5cm of a3] (a4) {$\overline{u}$};
\vertex [below=2em of a1] (b1) {$d$};
\vertex [below=2em of a4] (b2) {$d$};
%% See section 13.5 of PGF/TikZ manual
\vertex at ($(a2)!0.5!(a3)!0.5cm!90:(a3)$) (d);
%% Equivalent way to obtain (d):
% \vertex at ($(b2)!0.5!(b3) + (0, -0.5cm)$) (d);
\vertex [above=of a4] (c1) {$\overline{d}$};
\vertex [above=2em of c1] (c3) {$\overline{d}$};
\vertex at ($(c1)!0.5!(c3) - (1cm, 0)$) (c2);

\diagram* {
(a4) -- [fermion] (a3) -- [fermion] (a2) -- [fermion] (a1),
(b1) -- [fermion] (b2),
(c3) -- [fermion, out=180, in=45] (c2) -- [fermion, out=-45, in=180] (c1),
(a2) -- [boson, quarter left] (d) -- [boson, quarter left] (a3),
(d) -- [boson, bend left, edge label={$W^+$}] (c2),
};
\draw [decoration={brace}, decorate] (b1.south west) -- (a1.north west) node [pos=0.5, left] {\(B^0\)};
\draw [decoration={brace}, decorate] (c3.north east) -- (c1.south east) node [pos=0.5, right] {\(\pi^+\)};
\draw [decoration={brace}, decorate] (a4.north east) -- (b2.south east) node [pos=0.5, right] {\(\pi^-\)};
\end{feynman}
\end{tikzpicture}
\begin{tikzpicture}
\begin{feynman}
\vertex (a1) {\overline b};
\vertex [right=1cm of a1] (a2);
\vertex [right=1cm of a2] (a3);
\vertex [right=1cm of a3] (a4) {b};
\vertex [right=1cm of a4] (a5);
\vertex [right=2cm of a5] (a6) {u};
\vertex [below=2em of a1] (b1) {d};
\vertex [right=1cm of b1] (b2);
\vertex [right=1cm of b2] (b3);
\vertex [right=1cm of b3] (b4) {\overline d};
\vertex [below=2em of a6] (b5) {\overline d};
\vertex [above=of a6] (c1) {\overline u};
\vertex [above=2em of c1] (c3) {d};
\vertex at ($(c1)!0.5!(c3) - (1cm, 0)$) (c2);
\diagram* {
{\[\text{edges=fermion}\]}
(b1) -- (b2) -- (a2) -- (a1),
(b5) -- (b4) -- (b3) -- (a3) -- (a4) -- (a5) -- (a6),
},
\vertex [above=of a6] (c1) {\overline u};
\vertex [above=2em of c1] (c3) {d};
\vertex at ($(c1)!0.5!(c3) - (1cm, 0)$) (c2);
\draw [decoration={brace}, decorate] (b1.south west) -- (a1.north west) node [pos=0.5, left] {B^0};
\draw [decoration={brace}, decorate] (c3.north east) -- (c1.south east) node [pos=0.5, right] {\pi^-};
\draw [decoration={brace}, decorate] (a6.north east) -- (b5.south east) node [pos=0.5, right] {\pi^+};
\end{feynman}
\end{tikzpicture}
\begin{tikzpicture}
  \begin{feynman}
  \vertex (a1) (\overline b);
  \vertex[right=2cm of a1] (a2);
  \vertex[right=0.5cm of a2] (a3);
  \vertex[right=0.5cm of a3] (a4);
  \vertex[right=2cm of a4] (a5) (\overline s);
  \vertex[below=2cm of a1] (b1) (d);
  \vertex[below=2cm of a5] (b2) (d);
  \vertex[below=1.5em of a5] (c1) (s);
  \vertex[above=1.5em of b2] (c3) (\overline s);
  \vertex at ($(c1)!0.5!(c3) - (1cm, 0)$) (c2);
  \diagram* {
    \edges=fermion
    (a5) -- (a4) -- (a3) -- (a2) -- (a1),
    (b1) -- [fermion] (b2),
    (c3) -- [fermion, out=180, in=-60] (c2) -- [fermion, out=60, in=180] (c1),
    (a3) -- [gluon, bend right] (c2),
    (a4) -- [boson, out=90, in=90, looseness=2.0, edge label'={\(W^+\)}] (a2),
  };
  \draw [decoration={brace}, decorate] (b1.south west) -- (a1.north west)
    node [pos=0.5, left] (\(B^{0}\));
  \draw [decoration={brace}, decorate] (a5.north east) -- (c1.south east)
    node [pos=0.5, right] (\(\phi\));
  \draw [decoration={brace}, decorate] (c3.north east) -- (b2.south east)
    node [pos=0.5, right] (\(K^{0}\));
  \end{feynman}
\end{tikzpicture}
INDEX

This index only contains automatically generated entries. A good index should also contain carefully selected keywords. This index is not a good index.

anti charged boson key, 19
anti charged scalar key, 19
anti fermion key, 20
anti majorana key, 20
arrow distance key, 22
arrow shorten key, 22
arrow style key, 22
baseline key, 12
blob key, 17
boson key, 18
charged boson key, 19
charged scalar key, 19
crossed dot key, 17
\diagram, 9, 10
dot key, 16
edge label key, 20
edge label’ key, 20
edges key, 16
empty dot key, 17
Environments
   feynman key, 9
every (edge style) key, 18
every (vertex shape) key, 16
every diagram key, 13
every edge key, 18
every feynman key, 11
execute at begin feynman key, 11
execute at end feynman key, 11
fermion key, 19
feynman environment, 9
\feynmandiagram, 9
ghost key, 19
gluon key, 20
/graph drawing/
   horizontal, 12
   horizontal’, 12
   layered layout, 14
   spring electrical layout, 14
   spring layout, 14
   tree layout, 15
   vertical, 12
   vertical’, 12
   half left key, 22
   half right key, 22
   horizontal, 12
   horizontal’ key, 12
   in key, 22
   inline key, 11
   insertion key, 20
   label distance key, 21
   label style key, 22
   large key, 13
   layered layout key, 14
   looseness key, 23
   majorana key, 20
   medium key, 13
   momentum key, 21
   momentum’ key, 21
   nodes key, 16
   out key, 22
   particle key, 17
   photon key, 19
   plain key, 18
   quarter left key, 22
   quarter right key, 22
   relative key, 23
   reversed momentum key, 21
   reversed momentum’ key, 21
   rmomentum key, 21
   rmomentum’ key, 21
   scalar key, 19
   size key, 20
   small key, 13
   spring electrical layout key, 14
   spring layout key, 14
   square dot key, 17
   style key, 20
/tikz/
   baseline, 12
   edge label, 20
   edge label’, 20
References


