For quite some time, I hardly have the need to draw commutative diagrams. Therefore I did not update this document. It has been long on my ToDo-List to create a package to be able to draw commutative diagrams using a convenient syntax. Florêncio Neves now did this, his package *tikz-cd* is available on CTAN[1]. I strongly suggest that you have a look there and will provide this document only for reference.

[1]http://ctan.org/pkg/tikz-cd

Commutative Diagrams using Ti*k*Z

Felix Lenders

April 12, 2012

When typesetting mathematics with LaTeX, one often has the wish to insert commutative diagrams. Looking for suitable packages, one finds quickly *xypic*–package. Trying the first example with it, I have been disappointed about the poor output quality, also the documentation is from 1999. So I decided to use my favorite graphics package Ti*k*Z. This results in a little bit more work but a high quality result.

1 A first example

Load Ti*k*Z by including the following code into your header:

```
\usepackage{tikz}
\usetikzlibrary{matrix,arrows}
```

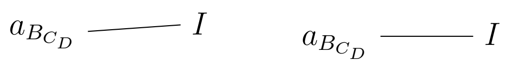
One remark important in Ti*k*Z-Usage: Every Ti*k*Z-Command ends with an `;`.

A first example can be obtained with the following code:

$$\begin{tikzcd}
A \arrow[rr, equal, "\varphi"] \arrow[dr, dash, "\Psi" description] & & B \arrow[dl, dash, "\Phi"] \\
& C &
\end{tikzcd}$$

```
\begin{tikzpicture}[description/.style={fill=white,inner sep=2pt}]
    \matrix (m) [matrix of math nodes, row sep=3em,
    column sep=2.5em, text height=1.5ex, text depth=0.25ex]
    { A & & B \\
        & C & \\ };
    %\draw[double,double distance=5pt] (m-1-1) - (m-1-3);
       \path[-,font=\scriptsize]
    (m-1-1) edge[double,thick,double distance=5pt] node[auto] {$ \varphi $} (m-1-3)
            edge node[description] {$ \Psi $} (m-2-2)
    (m-1-3) edge node[auto] {$ \Phi $} (m-2-2);
\end{tikzpicture}
```

At first we open environment `tikzpicture`. In `[]` some styles are defined which will be later discussed. Then a matrix named `m` is created, in `[]` also some additional style parameters are set. In `{}` the matrix elements are given, columns separated by `&`, rows separated by `\\` as known from `tabular`-environment. The style parameter for the matrix are `matrix of math nodes` which makes LaTeX automatically switching to math mode so you can write `A` instead of `$ A $`. With `row sep` and `column sep`, the row and column widths are defined. The parameters are `text height=1.5ex, text depth=0.25ex` to fix following problem: If you have objects with different heights in one row and connect them by a line, Ti*k*Z draw the line from the middle of one object to the middle of the other and therefore it gets slanted. Adding the parameters above avoids the problem. You can compare the results:



With `\path` the connectors between the matrix-nodes are drawn. A matrix-element is here adressed by (*matrix-name*-*row*-*column*), as you would normally do in linear algebra with a_{ij}, where a_{ij} is the entry of matrix A in row i and column j. Note that this is different from usual usage of cartesian coordinates, where you would write $(j, -i)$ for the same result. So with

```
\path (m-1-1) edge (m-2-3);
```

one would draw a straight line from the node of matrix `m` in first row, first column to the node of matrix `m` in second row, third column. In most cases, only a line is not enough, you also want to put some description on it. This can be done with:

```
\path (m-1-1) edge node { $ \scriptstyle \varphi $ } (m-2-3);
```

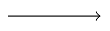
Of course, there is no need to write the complete command command for every line, with

```
\path
(m-1-1) edge (m-1-3)
        edge node {$ \scriptstyle \Psi $} (m-2-2)
(m-1-3) edge (m-2-2);
```

we draw three lines, two of them starting at `(m-1-1)` and ending at `(m-1-3)` (`(m-2-2)`), the third starts at `(m-1-3)` and ends at `(m-2-2)`.

2 Defining styles

In most cases, the Ti*k*Z standard output is not quite what you want. You can change it in many ways as discussed in 3. It would be tedious to make the same changes on every single line, so Ti*k*Z offers the feature to define styles. For example with



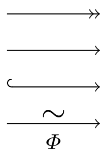
```
\begin{tikzpicture}[normal line/.style={->},font=\scriptsize]
\path[normal line]
(m-1-1) edge (m-1-2);
\end{tikzpicture}
```

the style `normal line` is defined and applied to the `\path` using `\path[normal line]`.

3 Fine Tuning

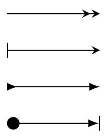
3.1 Arrows

Arrows are defined by *start-kind*-*end-kind*, where *kind* ∈ {`>`, `»`, `|`, `>|`, `)`, `o`, `*`, `left hook`, `right hook`, `<`, ... }. To get a nice arrow for bijective mappings, a little hand-work has to bone that the ∼-sign is not too much above the arrow. A few examples are shown below:



```
\begin{tikzpicture}[
  bij/.style={above,sloped,inner sep=0.5pt}]
  \matrix (a) [matrix of math nodes,row sep=.5em,
  column sep=3em, nodes in empty cells]
  { & \\ & \\ & \\ & \\ };
  \path[-»] (a-1-1) edge (a-1-2);
  \path[->] (a-2-1) edge (a-2-2);
  \path[right hook->] (a-3-1) edge (a-3-2);
  \path[->] (a-4-1) edge node[bij] {$ \sim $}
                         node[below] {$ \scriptstyle \Phi $} (a-4-2);
\end{tikzpicture}
```

You can also use `stealth`- or `latex`-style, if you prefer somehow "thicker" arrows:



```
\begin{tikzpicture}
  \matrix (a) [matrix of math nodes,row sep=.5em,
  column sep=3em, nodes in empty cells]
  { & \\ & \\ & \\ & \\ };
  \path[>=stealth,-»] (a-1-1) edge (a-1-2);
  \path[>=stealth,|->] (a-2-1) edge (a-2-2);
  \path[>=latex,>->] (a-3-1) edge (a-3-2);
  \path[>=latex,*->|] (a-4-1) edge (a-4-2);
\end{tikzpicture}
```

3.2 Curved Lines

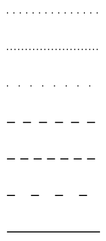
Sometimes it ist nice to have curved lines. You can get them by setting `bend left=`*angle* or setting `bend right=`*angle*.

$$\begin{tikzcd}
A & B & & C & D \\
& & E \arrow[ull, bend left] \arrow[ul] \arrow[ur] \arrow[urr, bend right]
\end{tikzcd}$$

```
\begin{tikzpicture}
  \matrix (m) [matrix of math nodes, row sep=3em,
  column sep=3em, text height=1.5ex, text depth=0.25ex]
  { A & B &   & C & D \\
      &   & E &   &   \\ };
  \path[->]
  (m-2-3) edge [bend left=15] (m-1-1)
          edge [bend right=30] (m-1-5)
          edge (m-1-2)
          edge (m-1-4);
\end{tikzpicture}
```

3.3 Line styles

Different predefined line-styles are `dotted`, `densely dotted`, `loosely dotted`, `dashed`, `densely dashed`, `loosesly dashed`, `solid`.



```
\begin{tikzpicture}
  \matrix (a) [matrix of math nodes,row sep=.5em,
  column sep=3em, nodes in empty cells]
  { & \\ & \\ & \\ & \\ & \\ & \\ & \\ };
  \path[dotted] (a-1-1) edge (a-1-2);
  \path[densely dotted] (a-2-1) edge (a-2-2);
  \path[loosely dotted] (a-3-1) edge (a-3-2);
  \path[dashed] (a-4-1) edge (a-4-2);
  \path[densely dashed] (a-5-1) edge (a-5-2);
  \path[loosely dashed] (a-6-1) edge (a-6-2);
  \path[solid] (a-7-1) edge (a-7-2);
\end{tikzpicture}
```

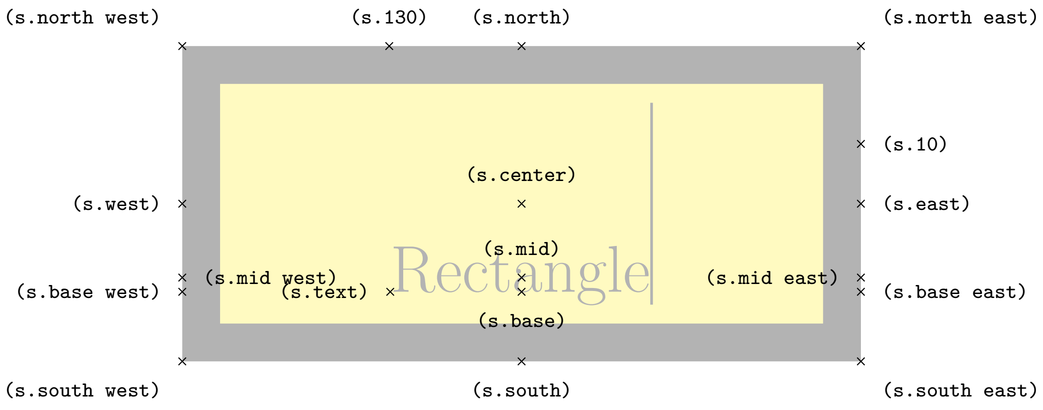
3.4 Long Matrix Nodes

If you have long matrix nodes and are not satisfied with Ti*k*Z-standard suggestion, you can decide at which anchor the arrow should begin `(`*name*`-`*i*`-`*j*`.`*anchor*`)`. Here *anchor* can be an angle or a direction as `south east`. You can align single matrix elements by putting `|[`*align*`]|` before the element and more general, you can adjust by this every style parameter of a matrix element.

$$\begin{tikzcd}
\bigoplus A' \otimes_R A'' \arrow[rr] \arrow[dr] & & B \arrow[d] \\
& \bigoplus C' \otimes_R C'' &
\end{tikzcd}$$

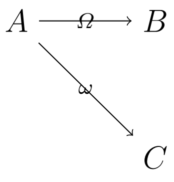
```
\begin{tikzpicture}
  \matrix (m) [matrix of math nodes, row sep=3em,
  column sep=3em, text height=1.5ex, text depth=0.25ex]
  { \bigoplus A' \otimes_R A" & |[red]| B \\
                              & |[left]| \bigoplus C' \otimes_R C"\\ };
  \path[->]
  (m-1-1) edge (m-1-2)
  (m-1-1.205) edge (m-2-2.173)
  (m-1-2) edge (m-2-2.north east);
\end{tikzpicture}
```

The possible anchors can bee seen in the following sketch:



3.5 Positioning descriptions

The position of arrow descriptions can be controlled with parameters for `node`-command. Without option, the description is always placed in the middle of the arrow direct onto the arrow:



$$\begin{tikzcd}
A \arrow[r, "\Omega" description] \arrow[dr, "\omega" description] & B \\
 & C
\end{tikzcd}$$

```
\begin{tikzpicture}
  \matrix (m) [matrix of math nodes, row sep=3em,
  column sep=3em]
  { A & B \\
      & C \\ };
  \path[->,font=\scriptsize]
  (m-1-1) edge node {$ \Omega $} (m-1-2)
          edge node {$ \omega $} (m-2-2);
\end{tikzpicture}
```

This is not nice. If you want to position the description direct onto the arrow, you have to make the background white:

$$\begin{tikzcd}
A \arrow[r, "\Omega" description] \arrow[dr, "\omega" description] & B \\
& C
\end{tikzcd}$$

```
\begin{tikzpicture}[descr/.style={fill=white,inner sep=2.5pt}]
  \matrix (m) [matrix of math nodes, row sep=3em,
  column sep=3em]
  { A & B \\
      & C \\ };
  \path[->,font=\scriptsize]
  (m-1-1) edge node[descr] {$ \Omega $} (m-1-2)
          edge node[descr] {$ \omega $} (m-2-2);
\end{tikzpicture}
```

If you don't what to have it direct on the arrow, you can try the `auto`-option, which often provides good results. If you prever the description on the opposite site of the arrow as chosen by Ti*k*Z, you can use `auto,swap`-option.

$$\begin{tikzcd}
A \arrow[r, "\Omega"] \arrow[dr, "1"] \arrow[dr, "1'"'] & B \\
& C
\end{tikzcd}$$

```
\begin{tikzpicture}[descr/.style={fill=white,inner sep=2.5pt}]
  \matrix (m) [matrix of math nodes, row sep=3em,
  column sep=3em]
  { A & B \\
      & C \\ };
  \path[->,font=\scriptsize]
  (m-1-1) edge node[auto] {$ \Omega $} (m-1-2)
          edge node[auto] {$ 1 $} (m-2-2)
          edge node[auto,swap] {$ 1' $} (m-2-2);
\end{tikzpicture}
```

You can also place descriptions by hand, using `\node[`*dir*`]`, with either *dir* $\in$ {`above`, `below`, `left`, `right`, `above left`, `above right`, `below left`, `below right`} or more concrete for example `above=2pt,below=7pt`.

$$\begin{tikzcd}
A \arrow[r, "\Omega"] \arrow[dr, "1"] \arrow[dr, "1'"'] & B \\
& C
\end{tikzcd}$$

```
\begin{tikzpicture}[descr/.style={fill=white,inner sep=2.5pt}]
  \matrix (m) [matrix of math nodes, row sep=3em,
  column sep=3em]
  { A & B \\
      & C \\ };
  \path[->,font=\scriptsize]
  (m-1-1) edge node[above] {$ \Omega $} (m-1-2)
          edge node[above right] {$ 1 $} (m-2-2)
          edge node[left=13pt] {$ 1' $} (m-2-2);
\end{tikzpicture}
```

It is also possible to adjust the description to the arrow direction with `sloped`-parameter:

$$\begin{tikzcd}
A \arrow[r, "\Omega"] \arrow[dr, "A(x)"' sloped] & B \\
& C
\end{tikzcd}$$

```
\begin{tikzpicture}[descr/.style={fill=white,inner sep=2.5pt}]
  \matrix (m) [matrix of math nodes, row sep=3em,
  column sep=3em]
  { A & B \\
      & C \\ };
  \path[->,font=\scriptsize]
  (m-1-1) edge node[auto] {$ \Omega $} (m-1-2)
          edge node[below,sloped] {$ A(x) $} (m-2-2);
\end{tikzpicture}
```

3.6 Equation Environments

If one uses a Ti*k*Z-Diagram inside an equation environment, one has to use the option `baseline=(current bounding box.base)` so that the equation label is placed correctly.

3.7 Crossing Arrows

If there are crossing edges, you can use the following workaround the get white spaces at the crossing parts: First draw the lines in background, then draw “invisible”, thick white lines for every line, which is later of the background lines and after this, draw the normal foreground lines. Ti*k*Z can do this in one step with the `\preaction`-command. For example a cube(taken from `http://texblog.net/latex-archive/maths/tikz-commutative-diagram-edges-over-under/`):

$$\begin{tikzcd}
 & f^*E_V \arrow[rr] \arrow[dl] \arrow[dd, dotted] & & E_V \arrow[dl] \arrow[dd] \\
f^*E \arrow[rr, crossing over] \arrow[dd] & & E & \\
 & U \arrow[rr, dotted] \arrow[dl, dotted] & & V \arrow[dl] \\
M \arrow[rr] & & N \arrow[from=uu, crossing over] &
\end{tikzcd}$$

```
\begin{tikzpicture}[
        back line/.style={densely dotted},
        cross line/.style={preaction={draw=white, -,
           line width=6pt}}]
    \matrix (m) [matrix of math nodes,
         row sep=3em, column sep=3em,
         text height=1.5ex,
         text depth=0.25ex]{
        & f^\ast E_V & & E_V \\
          f^\ast E   & & E   \\
        & U          & & V   \\
          M          & & N   \\
    };
    \path[->]
        (m-1-2) edge (m-1-4)
                edge (m-2-1)
                edge [back line] (m-3-2)
        (m-1-4) edge (m-3-4)
                edge (m-2-3)
        (m-2-1) edge [cross line] (m-2-3)
                edge (m-4-1)
        (m-3-2) edge [back line] (m-3-4)
                edge [back line] (m-4-1)
        (m-4-1) edge (m-4-3)
        (m-3-4) edge (m-4-3)
        (m-2-3) edge [cross line] (m-4-3);
\end{tikzpicture}
```

4 Further Information

At first, you should take a look in the excellent pgf-manual, which you can find for example at `http://www.ctan.org/tex-archive/graphics/pgf/base/doc/generic/pgf/pgfmanual.pdf`.

Also helpful for me has been two TEXblog.net entries:
`http://texblog.net/latex-archive/maths/pgf-tikz-commutative-diagram/`
`http://texblog.net/latex-archive/maths/tikz-commutative-diagram-edges-over-under/`

If you have any feedback, what can be done better or what could be extended to this tutorial, I would be glad if you send me a mail to `f.lenders/bei/stud.uni-heidelberg.de`.