

Using $\text{\texttt{Xy-pic}}$ in $\text{\texttt{LyX}}$

H. Peter Gumm

July 24, 2008

With the current version of $\text{\texttt{LyX}}$ and with the `preview`-style installed in the $\text{\texttt{L}^{\text{\texttt{A}}}\text{\texttt{T}}_{\text{\texttt{E}}}\text{\texttt{X}}$ -System, the graph drawing package $\text{\texttt{Xy-Pic}}$ can now be conveniently used inside $\text{\texttt{LyX}}$. Diagrams can be edited and displayed inside the main $\text{\texttt{LyX}}$ editing window. Here, we shall describe how to use the `\xymatrix` command from $\text{\texttt{xy-pic}}$ inside $\text{\texttt{LyX}}$ in order to draw, to edit and to preview diagrams as typically used in category theory, algebra, and related fields.

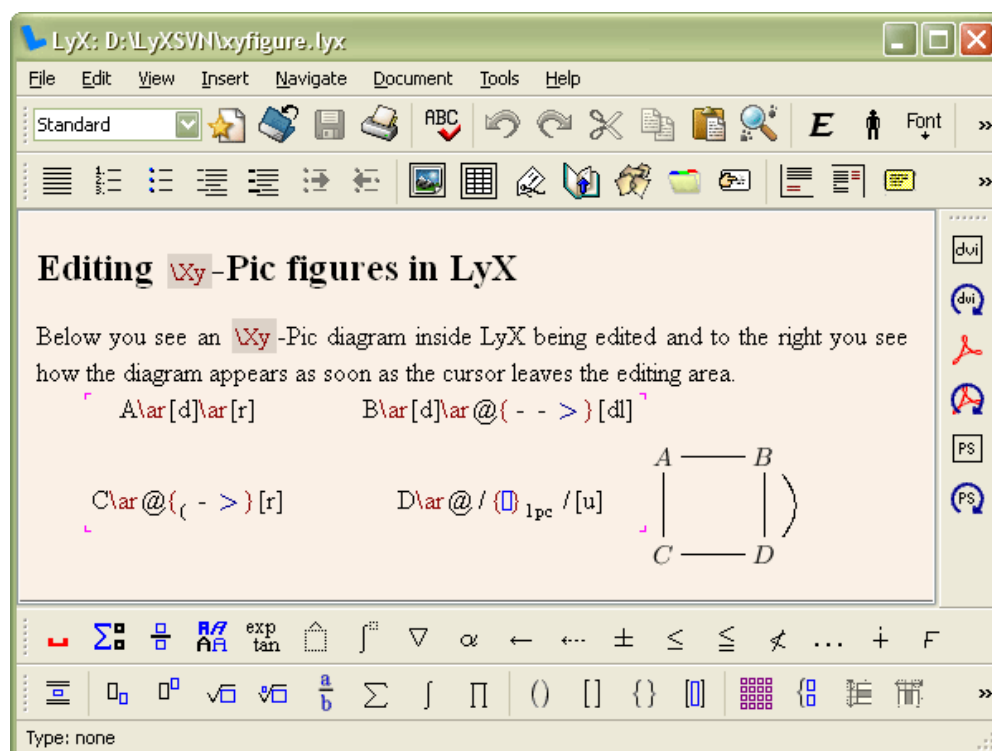
Contents

1	Introduction	2
2	Preparation	3
3	Commutative diagrams	4
3.1	The matrix layout of diagrams	4
3.2	Arrows	5
3.3	Labels	5
3.4	Arrow modification	6
3.4.1	Arrow design	6
3.4.2	Designing your own arrows	7
3.5	Arrow positioning	7
3.5.1	Inline or centered diagrams	8
3.6	Bending arrows	8
3.6.1	Raising the shaft	8
3.6.2	Specifying exit- and entrance directions.	8
3.7	Modifying vertices	9
3.7.1	Framing objects	9

4 Using LyX's math editor	9
4.1 Caveat - how to enter braces	10
4.2 Setting up the matrix	10
4.3 Entering nodes, arrows and labels	10
4.4 Modifying arrows	11
4.5 What if something goes wrong	11
5 Hacks	11
5.1 Horizontal and vertical scaling	11
5.2 Invisible arrows	12
References	13

1 Introduction

The `xypic`-package has long served as a convenient tool for easily constructing graphs and diagrams in \LaTeX . Unfortunately, its use in LyX had long been restricted to the infamous ERT-boxes, meaning that the LyX editor could only display the \LaTeX -source and not the finished diagram. The new `preview`-style of \LaTeX which is part of the AUCT \LaTeX project[4], finally enables the editing and displaying of `xypic`-diagrams, constructed, displayed and interactively edited inside LyX.



In this note, we describe how `Xy-pic` can be used from inside LyX, how diagrams can

be created and edited. We have tested the following using LyX version 1.3.7, running under WindowsXP.

There are two modes of operations: For a start, and for some first tests, it may be easiest to first enter the Xy-Pic code inside the LyX-window, select it all and convert it to a graphical representation by pressing Ctrl-m or Ctrl-M. If you use Xy-Pic more frequently, or if you want to modify your initial figure, you will want to assemble and modify your figures using LyX's math editor. This is shown in blue in the above figure: Once the cursor is moved over a diagram, this is displayed as an array of nodes and arrow-commands. These can be changed interactively. When the cursor leaves the editing area, the diagram reappears.

In the first two sections of this documentation, we explain how to use LyX in the first mentioned mode and introduces all Xy-Pic features that might be of use for drawing commutative diagrams, graphs or automata. Section 3 explains how to use the Xy-Pic commands inside a math-editing area.

It is not our intention to write another introduction to Xy-Pic, rather our motivation is to give an introduction how the most important commands work inside LyX, since the keystrokes as explained in the Xy-Pic manual[?] will not always function correctly inside LyX.

2 Preparation

The following requires that the L^AT_EX-packages `xypic` and `preview` are installed in the L^AT_EX system. They are available from CTAN, see at [3], resp. at [2]. After freshly installing them, it may be necessary, to run **Edit▷Reconfigure** from the main LyX menu. The steps to a first diagram output in LyX are then:

1. Activate and test `preview`

- a) Open LyX, choose **Edit▷Preferences▷Look and Feel▷Graphics** and place a check-mark at *Instant Preview*.
- b) Test, if `instant-preview` works by opening a LyX-document and entering any math-formula, e.g. $a + b = c$.
- c) Move the cursor out of the formula, and watch it change its appearance to look just like in the finished dvi- or postscript document.

2. Activate and test `xypic`

- a) Inside your LyX-Documnet, enter the text `\xymatrix{A \ar[r] & B} .`
- b) Select the whole text and choose **Insert▷Math▷Display Formula**, or use the corresponding keyboard shortcut **Ctrl-M**.

- c) Move the mouse cursor out of the editing box and wait for a split second to see an arrow appear: $A \longrightarrow B$.

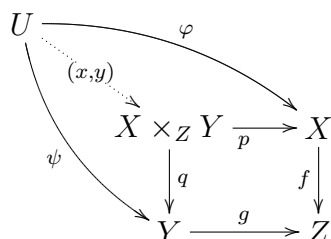
Note: Your document will not be compilable when it has the same name as one of the commands defined by `xypic`.

3 Commutative diagrams

The following diagram, which is taken from the documentation of `Xy-Pic[?]` by its creator Kristoffer H. Rose, will provide an example for many of the features available with that package. Its source code is:

```
\xymatrix{
  U      \ar@/_{1pc}/[ddr]_{\psi}\ar@/{1pc}/[drr]^{\varphi} \\
  \ar@{.}>[dr]|-{(x,y)}\\
  & X \times_Z Y \ar[d]_q \ar[r]_p & X \ar[d]_f \\
  & Y \ar[r]_g & Z }
```

Again, to turn this code into a graphical output, select it all at once starting from the `\xymatrix{ ...` up to the closing brace `... }` and turn it into display-math as explained above. A moment after the cursor leaves the math-area, you should see the diagram in its full graphical glory as shown below.



3.1 The matrix layout of diagrams

`xymatrix` uses a matrix to define the layout of the vertices of a diagram. For the above example, we need a 3×3 -matrix of which 5 entries are used for the vertices U , $X \times_Z Y$, X , Y , Z , the other positions remaining empty. In this case, the following matrix determines the layout:

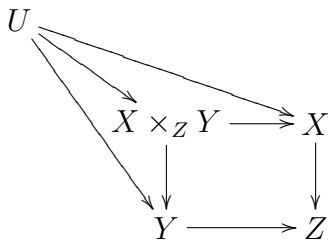
```
\xymatrix{
  U      & & \\
  & X \times_Z Y & X \\
  & Y & Z }
```

The pattern should be familiar from \LaTeX : We see three rows, each row being terminated by the end-of-line-marker `\`. Each line consists of entries, separated by the ampersand `&`.

3.2 Arrows

Having entered the vertices, we add arrows between them. The basic `xypic`-command to produce an arrow is `\ar`, it is entered into the cell of the matrix where the arrow is to start. The target of the arrow is defined by direction commands `u` (up) `d` (down) `l` (left), or `r` (right). These can be combined to a path and enclosed in square brackets. As an example, the arrows from the vertex U in the upper left corner down and right to the vertices $X \times_Z Y$, Y , and X are, respectively, defined as `\ar[dr]`, `\ar[ddr]` and `\ar[ddr]`. Thus the above diagram with all arrows added becomes:

```
\xymatrix{
  U \ar[ddr] \ar[ddr] \ar[dr] \\
    & X \times_Z Y \ar[d] \ar[r] & X \ar[d] \\
    & Y \ar[r] & Z }
```

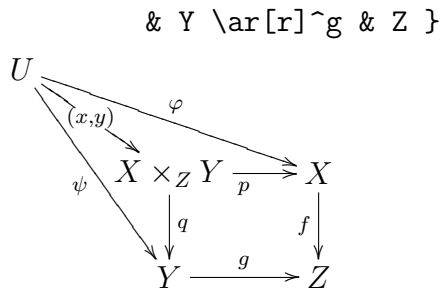


3.3 Labels

Labels are attached to arrows by affixing them as upper or lower indices to the `\ar`-command. Thus, `\ar[ddr]^{\varphi}` defines an arrow going one cell down, two to the right and having the label φ attached above. To attach a label below the arrow, make it a lower index as in `\ar[ddr]_{\psi}`. This explanation is correct only for arrows pointing to the right. More precisely, imagine looking along the arrow in the direction it is pointing. Then an upper index places a label to the left and a lower index places it to the right. Consequently, an arrow pointing from right to left, such as `\ar[l]^{\alpha}_{\beta}` will have label α below and label β above the arrow, i.e. $\xleftarrow[\alpha]{\beta}$. Using the character `|` instead of `^` or `_`, it is even possible to place the label right onto the arrow, obscuring part of its shaft.

Normally, a label is placed halfway between an arrow's start and target objects. In the first diagram, the central arrow starting in U has the label (x, y) in the middle of the arrow's shaft, rather than in the middle between the two objects it connects. This is achieved by prefixing the label with a minus sign, here: `\ar[dr]|-{(x,y)}`.

```
\xymatrix{
  U \ar[ddr]_{\psi} \ar[ddr]^{\varphi} \ar[dr]|-{(x,y)} \\
    & X \times_Z Y \ar[d]_q \ar[r]_p & X \ar[d]_f }
```



Xy-pic normally permits labels to be shifted towards the tip or towards the end of an arrow by prefixing the label with a ratio, such as e.g. `(.3)`. In LyX this works only for labels which are placed on top of the arrow, such as `\ar[r](0.3){\phi}`. The form `\ar[r]!-(x,y)` used above guarantees that the label is placed at the midpoint of the arrow rather than at the midpoint between the nodes connected by the arrow.

For labels placed to the left or to the right of the arrow this does not work. The corresponding Xy-pic code such as e.g. `\ar[r]^(.3)p` is not correctly interpreted by LyX's math editor. A workaround is suggested in the last section of this note.








3.4 Arrow modification

Modification of the design, the form or the positioning of arrows are introduced by the `@`-character. This is followed by a pair of matching brackets, where the form of the bracket pair, `{ }` or `< >` or `/ /` indicates, whether we want to modify the design, the shift or the curvature of the arrow. Various modifications can be applied to an arrow at the same time.







3.4.1 Arrow design

Various designs such as *solid*, *dotted*, *dashed* or *double* are possible for the shaft of an arrow. These can be combined with various ends and various tips. In general, the design of an arrow is described by following the command `\ar` immediately by an `@`-sign and a pair of braces `{...}` containing characters describing the end, the shaft and the tip of the arrow. These characters are chosen to give some form of ASCII-rendering of the real thing. For instance `\ar@>...>>` produces an arrow with split end, a dotted shaft and double head. A number of other arrow designs is given in the table below. Note that the ends of embedding arrows $A^{\hookrightarrow} B$ are described by raising or lowering opening parentheses, such as in `\ar@{^(->}[r]`.

Result	Source code in LyX
\longrightarrow	<code>\ar</code>
$-->$	<code>\ar@{-->}</code>
$\cdots\cdots\cdots\rightarrow$	<code>\ar@{...>}</code>

	<code>\ar@{~>}</code>
	<code>\ar@{->>}</code>
	<code>\ar@{-->>}</code>
	<code>\ar@{>->>}</code>
	<code>\ar@{_(->}</code>
	<code>\ar@{^(->}</code>
	<code>\ar@{ - }</code>

Following the @-character by either a 2, 3, _, or a ^, we can produce arrows with double, triple shaft or arrows showing only the lower or upper half of their tips and ends. Arrows need not have tips nor ends, as the last example shows :

Result	Source code for \LaTeX
	<code>\ar@2</code>
	<code>\ar@3</code>
	<code>\ar@_(->}</code>
	<code>\ar@^(->}</code>
	<code>\ar@^(>>->>}</code>
	<code>\ar@{^<-_>}</code>
<code>= = =</code>	<code>\ar@2{--}</code>

3.4.2 Designing your own arrows

Within certain limits there is even a way to design your own arrows. Using some the characters `><|ox+()` [one can even design one's own arrow tips using the `\newdir` command in the preamble. For explanations, we refer to the ~~X~~ manual, from which we take the example:

```
\newdir{|>}{!/4.5pt/@{|}*:(1,-.2)@^{>}*:(1,+.2)@_>}}.
```

This defines a new arrow tip, referred to as `|>` in `\ar@{-|>}[r]` and which displays correctly in \LaTeX as:

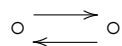
$$A \longrightarrow B$$

3.5 Arrow positioning

Arrows are shifted sideways with the modifier `@<...>` where the ellipsis is replaced by a positive or negative measure. For instance, to design a pair of mutually opposing arrows between two nodes, we shift them to see them apart. Note that the direction of the shift (positive) is to the left if one looks along the arrow. Thus

```
\[\xymatrix{\circ \ar@<1ex>[r] \& \circ \ar@<1ex>[l]}
```

produces



3.5.1 Inline or centered diagrams

Arrows and diagrams can be used inline, as this one: $\circ \rightleftarrows \circ$. For this they should be placed inside an inline math-environment `$...$` or `\(...\)`. To center a diagram, put it inside `display-math` parentheses `\[` and `\]`. Diagrams constructed inline can later be centered, or, conversely, centered diagrams can be changed to inline formulas with **Edit** \triangleright **Math** \triangleright **Alignment**.

3.6 Bending arrows

There are two simple methods to make arrows bend. The first is giving an explicit value by which the midpoint of the arrow's shaft is raised or depressed, the other is by forcing the arrow to leave its origin in a prescribed compass direction and to make him enter the target at another direction. The necessary bending of the arrow is determined automatically. We describe both methods.

3.6.1 Raising the shaft

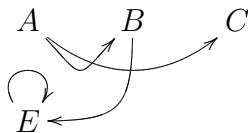
For bending arrows we use the modifier `@/.../`. The ellipsis stands for a TeX-measure which needs to be entered as a lower or upper index. Whereas in `xypic`, we could simply write, e.g. `\ar@/_1pc/` for an arrow bending 1pc downwards, this cannot directly be done in LyX. It is necessary, to enclose the measure in a pair of braces, such as e.g. `\ar@/_{1pc}/`. As an example, here are two opposing arrows between *A* and *B*, each bending by .5 pica, given by the following source code:

```
\[ \xymatrix{A \ar@/_{.5pc}/[r] & B \ar@/_{.5pc}/[l]} \]
```



3.6.2 Specifying exit- and entrance directions.

An alternative for making arrows bend is by specifying their compass direction as they are leaving their source and their direction from which they enter their target. Instead of north, north-east, east, etc., the directions are named `u`, `ur`, `r`, `dr`, `d`, `dl`, `l`, `ul`, standing for up, up-right, right, down-right, etc.. A direction is specified as `@(out,in)` where *out* stands for the direction the first object is left and *in* stands for the direction from which the target is entered. As an example, we show some bending arrows and a loop, which arises when we do not specify a target for an arrow, just its incoming and outgoing direction:



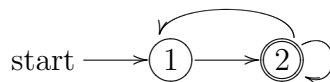
```
\[\xymatrix{A \ar@(dr,dl)[r]\ar@(dr,dl)[rr]
&B\ar@(d,r)[dl] &C \\
E \ar@(ul,ur)}\]
```

3.7 Modifying vertices

The above example is reminiscent of an automata diagram, except that in such a diagram states would be enclosed in small circles, with double circles denoting final states.

3.7.1 Framing objects

With \Xy-Pic , entries can obtain a single or a double frame, such as \boxed{A} or \boxed{B} by prefixing an entry with $*[F-]$ or $*[F=]$ and enclosing the portion of the entry to be framed in braces. Normally, the frame will be very tight so that it must be widened by prefixing with $+$ or with $++$. Round frames, such as \circledA and \circledB are obtained by specifying the shape as $[o]$. So the latter figure was constructed as $*++[o][F=]\{B\}$. This way, the following automaton



can be typeset as

```
\[\xymatrix{\text{\txt{start}}\ar[r]
&*++[o][F]{1}\ar[r]
&*++[o][F=]{2}\ar@(ur,dr)\ar@(ur,ul)[l]
}\]
```

The \LaTeX command $\text{\entrymodifiers}=\{\dots\}$ will make a certain entry style the default, that can, of course be overridden for individual entries. Thus, after $\text{\entrymodifiers}=\{++[o][F=]\}$ all following entries inside \Xy -matrices would be encircled.

4 Using \LyX 's math editor

As an alternative to writing the \xypic code, then transforming it into a math-editing environment by marking it and applying \Ctrl-m , or \Ctrl-M , one may construct and modify the whole \xypic -diagram inside \LyX 's math editor. We describe the editing steps for a figure just like the one above.

4.1 Caveat - how to enter braces

Recall that in L^AT_EX's math-editor any pair of braces { and } that are to enclose a macro-parameter must be entered by typing just \{ . The closing brace is automatically supplied and in between a box into which to the parameter is entered. In connection with X_Y-diagrams, this applies in particular to arrow modifications that are normally given in the form @{ ... } with the ellipsis standing for the description of end, shaft and tip of the arrow. Inside the math-editor, enter just @\{ and let L^AT_EX provide the closing brace and the box into which to enter the description of the arrow.

Braces that are entered without the backslash \ will just appear as typed, but cannot be used to receive a macro parameter. They are useful, for instance to denote sets, e.g. $\{x \in X \mid x \notin x\}$ will display as $\{x \in X \mid x \notin x\}$.

4.2 Setting up the matrix

With Ctrl-m or Ctrl-M open a formula environment and enter: \xymatrix. This produces a 1×1- X_Y-matrix. Add extra rows by typing Ctrl-Enter and add columns by typing Alt-m c i.

At any time, further rows or columns can be entered or deleted using commands available from Edit▷Math, resp. their shortcuts, beginning with Alt-m c for the column commands or Alt-m w for the row commands.¹

4.3 Entering nodes, arrows and labels

Type the nodes into the correct positions of the matrix. If you move the cursor out of the matrix, you should see a first rendering of the node layout. Next, add the arrows at the nodes from where they should emanate by typing \ar[p], where p can be any path made up from the characters u, d, l, r. Make sure that the path indeed leads to a node available within the matrix. Otherwise, the figure will not display as the cursor leaves the editing area.

Next, label the arrows by attaching them as upper or lower indices to the end of their path. As always, in L^AT_EX's math editor, an underscore _ opens a box for a lower index and a ^ followed by a space opens a box for an upper index. You can enter any L^AT_EX-code as a label.

¹Note that in the keyboard shortcuts starting with M- the letter M stands for a “Meta”-key, which in Windows translates to the Alt-key. Similarly in shortcuts listed as starting with C-, the C stands for Ctrl.

4.4 Modifying arrows

Finally, modify the appearance of the arrows by entering @-modifiers $@\{\dots\}$, $@<\dots>$, $@(\dots,\dots)$ or $@/\dots/$. The above caveat applies to the first form only. It must be entered as $@\{$ with the arrow description entered inside the L^AT_EX-supplied box. If this box remains empty, you have specified an empty arrow. This is a useful construction, too, as you will see in the next section.

The other modifiers, $@<\dots>$, $@(\dots,\dots)$ and $@/\dots/$ are typed as shown with the arrow description replacing the ellipsis. The code for bending arrows, which in xypic is $@/_measure/$ or $@/^measure/$ where *measure* is any valid T_EX-measure, should be entered as upper or lower index to the first slash $/$. Make sure that the ending slash does not end up with the upper or lower index.

4.5 What if something goes wrong

When constructing a diagram, you should at times check it by just moving the cursor out of the editing area to see whether instant preview can successfully convert it into graphical output. If this does not happen, it may either be that instant preview for some reason is not aware that it should modify/retranslate the graphics. Moving the cursor into the editing area and out again should wake up instant preview.

A more serious reason could be a syntactical error in your input. If necessary, redo the last editing steps, using Ctrl-z, or try to translate the L^AT_EX-file into dvi using Ctrl-d or View▷Dvi. There should be some error generated, which hopefully gives you a hint as to the source of the mistake.

5 Hacks

Certain things do not work correctly inside L^AT_EX. The ones that we (used to) miss most are the horizontal and vertical scaling of diagrams, and the correct positioning of arrows. After some experimentation, we have found workarounds that we are explaining here.

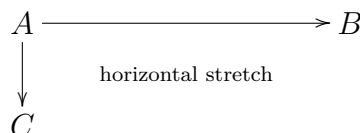
5.1 Horizontal and vertical scaling

It is often convenient to stretch the horizontal or vertical dimensions of a diagram by using spacing commands for rows and/or columns. According to the X_Y-manual, for instance, $\backslash\mathrm{xymatrix@R=1pc}\{\dots\}$ defines an X_Y-matrix with row spacing of 1 pica. Similarly, $\backslash\mathrm{xymatrix@C=\dots}\{\dots\}$ allows to modify the space between columns. Unfortunately, these commands do currently not work inside L^AT_EX, as the @-character is interpreted by L^AT_EX as ending the X_Y-matrix-macro.

Knowing that `Xy` stores the values for row-spacing and column-spacing in the variables `\xymatrixrowsep@` and `\xymatrixcolsep@`, add the following macro to the preamble (Layout▷Document▷Preamble)

```
\newcommand{\xyR}[1]{%
\xydef@\xymatrixrowsep@{#1}
} % end of \xyR
```

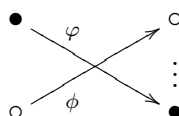
A macro `\xyC` can be defined correspondingly by replacing `\xymatrixrowsep@` with `\xymatrixcolsep@`. Now, a figure can be scaled by entering `\xyR{...}` into the `Xy`-matrix. Place the cursor inside the matrix, just before the first entry. Then enter `\xyR{}` or `\xyC{}` or both. Don't forget the backslashes and remember, that the closing brace is automatically supplied by `LATEX`. Inside the braces enter the dimensions. The default is 2pt. Here you see a diagram which is squashed vertically and stretched horizontally with `\xyR{9pc}\xyC{.5pc}`:



5.2 Invisible arrows

Another `Xy`-feature not functioning in `LATEX` is the correct positioning of labels along the shaft of arrows. The code `\ar[r]|(0.3)\varphi`, for instance will place the label ϕ not in the middle of the arrow, but rather at about 30% of the way from its origin. The same ought to work with labels above or below the arrow, e.g. `\ar[r]^(0.3)\phi` or `\ar[r]_(0.3)\phi`. Unfortunately, this simple does not function in `LATEX`.

Still, this feature is sometimes necessary, when the default position of the label would otherwise clutter the picture, or even coincide with other items, such as the intersection of the arrows in the figure below.



For strictly horizontal arrows, a workaround is easy: just pad the label itself with enough spaces to make the visible part shift enough to the right or to the left, as e.g. in `\ar[r]^{ \phi \ \ \ }`. For vertical or diagonal arrows, this does not work, so you might wish to use the following trick:

Produce an invisible second arrow, shorten (or prolong) it past its goal by adding a decimal stretching ratio, e.g. (0.6) or (1.4) to its path. Attach the label to this invisible arrow. In the above diagram, the downward pointing arrow with its label

was produced by `\ar[dr] \ar@{}[dr(0.6)]^{\varphi}`, where the second arrow is the invisible one, reaching only 0.6 of the way. Hence the label will appear at 0.3 of the way of the original visible arrow.

This workaround has two minor drawbacks: First, it does not work with bending arrows. Secondly, prolonging an invisible arrow beyond the normal dimension of the figure will invisibly extend the figure box, and thereby cause too much vertical space between the figure and the preceding or the following paragraph.

Invisible arrows are an important tool, though, since they can, in principle, be used to place information at any chosen place in a diagram. Above, for instance, we have used an invisible arrow to carry the `\vdots` as label and at the preceding picture we had used an invisible arrow to carry some text into the center of the figure.

References

- [1] Kristoffer H. Rose: *Xy-Pic User's Guide*. Version 3.7, Feb. 16, 1999. Available as part of the xypic L^AT_EX package. <http://www.ctan.org/tex-archive/macros/generic/diagrams/xypic/xy/doc/xyguide.pdf>
- [2] <http://www.ctan.org/tex-archive/help/Catalogue/entries/xypic.html>
- [3] <http://www.ctan.org/tex-archive/help/Catalogue/entries/preview-latex.html>
- [4] <http://www.gnu.org/software/auctex/>