Macro Packages for Typesetting Commutative Diagrams

Gabriel Valiente Feruglio
Universitat de les Illes Balears
Departament de Ciències Matemàtiques i Informàtica
E-07071 Palma de Mallorca (Spain)
Internet: `valiente@ipc4.uib.es`

Abstract

There have been several efforts aimed at providing TEX and its derivatives with a suitable mechanism for typesetting commutative diagrams, with the consequent availability of several macro packages of widespread use in the category theory community, and a long debate about the best syntax to adopt for commutative diagrams in LATEX3 has taken place last year in the `CATEGORIES` discussion list. From the user's point of view, however, there is not much guidance when it comes to choosing a macro package, and even after a decision is made, the conversion of diagrams from the particular conventions of a macro package to another macro package's conventions may prove to be rather hard.

Typesetting commutative diagrams is a surprisingly difficult problem, in comparison with TEX macro packages for other purposes, as judged by the amount of code needed and years of development invested. The existing macro packages for typesetting commutative diagrams are reviewed in this paper and they are compared according to several criteria, among them the capability to produce complex diagrams, quality of the output diagrams, ease of use, quality of documentation, installation procedures, resource requirements, availability, and portability. The compatibility of the different macro packages is also analyzed.

Introduction

Commutative diagrams are a kind of graphs that are widely used in category theory, not only as a concise and convenient notation but also as a powerful tool for mathematical thought.

A diagram in a certain category is a collection of nodes and directed arcs, consistently labeled with objects and morphisms of the category, where "consistently" means that if an arc in the diagram is labeled with a morphism f and f has domain A and codomain B, then the source and target nodes of this arc must be labeled with A and B respectively.

A diagram in a certain category is said to *commute* if, for every pair of nodes X and Y, all the paths in the diagram from X to Y are equal, in the sense that each path in the diagram determines through composition a morphism and these morphisms are equal in the given category. For instance, saying that the diagram[1]

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd}$$

commutes is exactly the same as saying that

$$g' \circ f = f' \circ g.$$

As a notation, the graphic style of presentation inherent to commutative diagrams makes statements and descriptions involving categories more clear and manageable than textual presentations. For instance, consider the definition of an equalizer. A morphism $e : X \to A$ is an *equalizer* of a pair of morphisms $f : A \to B$ and $g : A \to B$ is $f \circ e = g \circ e$ and for every morphism $e' : X' \to A$ satisfying $f \circ e' = g \circ e'$ there exists a unique morphism $k : X' \to X$ such that $e \circ k = e'$.

An equivalent definition is that e is an equalizer if the upper part of the diagram

[1] All the diagrams in this paper have been typeset using the XY-pic macro package, unless otherwise stated. The reader should not infer any preference by the author for that particular macro package, but should understand that some macro package is needed for the examples in the paper. Sample diagrams typeset with the other macro packages are given in Appendix I.

$$\begin{tikzcd} X \arrow[r, "e"] & A \arrow[r, shift left, "f"] \arrow[r, shift right, "g"'] & B \\ X' \arrow[u, dotted, "k"] \arrow[ur, "e'"'] & & \end{tikzcd}$$

commutes and, whenever the lower part of the diagram also commutes, there is a unique k such that the whole diagram commutes.

As a tool for thought, proofs involving properties that are stated in terms of commutative diagrams can often be given in a "visual" way, in what has been called *diagram chasing*. For instance, the proposition that if both inner squares of the following diagram commute, then also the outer rectangle commutes,

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "a"'] & B \arrow[r, "g"] \arrow[d, "b"] & C \arrow[d, "c"] \\ A' \arrow[r, "f'"'] & B' \arrow[r, "g'"'] & C' \end{tikzcd}$$

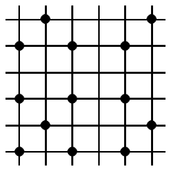
can be proven as follows:

$$\begin{array}{rcll} (g' \circ f') \circ a & = & g' \circ (f' \circ a) & \text{(associativity)} \\ & = & g' \circ (b \circ f) & \text{(commutativity of left square)} \\ & = & (g' \circ b) \circ f & \text{(associativity)} \\ & = & (c \circ g) \circ f & \text{(commutativity of right square)} \\ & = & c \circ (g \circ f) & \text{(associativity).} \end{array}$$

Commutative diagrams[2] range from simple, rectangular matrices of formulae and arrows to complex, non-planar diagrams with curved and diagonal arrows of different shapes.

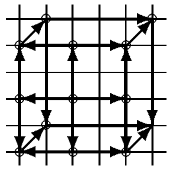
Constructing commutative diagrams

Commutative diagrams are constructed in most cases as rectangular arrays, as Donald Knuth does in Exercise 18.46 of *The TEXbook*. The objects or vertices are set much like a `\matrix` in TEX or an `array` environment in LATEX,

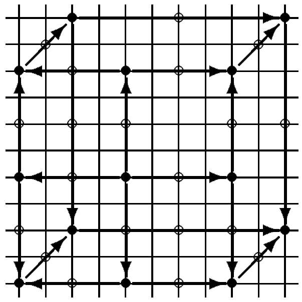


[2] The epithet "commutative" is traditional and it is originated in the fact that diagrams may be used to display equations such as the commutative and associative laws. Although not all such diagrams which people draw commute in the formal sense given, this paper adheres to tradition and all such diagrams are called commutative diagrams herein.

and the morphisms or arrows are set either right after the vertex where they start,



or in a cell on their own,



depending on the macro package being used, where the grids correspond to the sample diagram presented in Appendix I. (Sketching a commutative diagram on such a grid on paper may prove to be a mandatory step before typing the actual diagram, at least for all but the simplest diagrams.) This gives a first distinction,

- either one object or one or more morphisms in each non-empty cell, or
- one object and all departing morphisms in each non-empty cell.

Whether they belong together with their source object in a cell or they use a cell on their own, morphisms are specified by the address of their target cell. Such addresses can be implicit, absolute or relative to the source cell, and they can be either arbitrary or limited by the available diagonal slops.

Moreover, some macro packages even support symbolic addresses, by which nodes are tagged with identifier names and arrows are specified by making reference to the names of their source and target nodes. This is a step forward in the sense of the LATEX principle of emphasizing structural descriptions, and in fact it is of great help for designing complex diagrams because it divides the task into two separate subtasks, the one of producing a correct and elegant arrangement of nodes and the other one of laying out the correct arrows and positioning their labels.

Evaluation guidelines

The following aspects are considered in the next section for each of the macro packages in turn. The spirit of these guidelines is to give the potential user a feeling of what to expect from a macro package for typesetting commutative diagrams, and they are based on the experience of the author during the last two or three years, as user of some of the macro packages.

Arrow styles. The arrows used in commutative diagrams often are of different shapes, in order to distinguish different kinds of morphisms such as monomorphisms, epimorphisms, isomorphisms, and inclusions, to name just a few, and sometimes they have a shaft other than a solid line, for instance dashed or dotted, to indicate that it is the existence of the corresponding morphisms what is being characterized.

A collection of built-in arrow shapes and shafts is included in every macro package, and some macro packages even provide facilities for defining new arrow styles, for instance by defining a new control sequence name and choosing a particular combination of tail (the piece that appears at the source end), head (the piece that appears at the target end), and shaft, from a predefined pallette of possible heads, tails, and shafts.

Automatic stretching. Most of the macro packages provide for the automatic stretching of arrows to meet their source and target nodes, where meeting a node means to get as close to the (rectangular) box enclosing the node as dictated by some predefined parameter.

While it may be appropriate for most horizontal and vertical arrows, in the case of diagonal arrows it may leave the arrow too far from the node, and extra diagram fine-tuning (see below) is needed in such cases in order to get the arrow closer to the node. The macro package by John Reynolds, however, incorporates basic facilities for associating an hexagon, octagon, or diamond to a node, instead of the usual rectangle, although it does not exploit them in the macros for commutative diagrams.

Diagram fine-tuning. Given a correct description of the structure, a macro package has the task of choosing the best possible arrows to produce the commutative diagram. Sometimes the best choice may not seem good enough, because only a limited number of slopes may be available for the arrows, because arrows may cross, and because arrow labels may superimpose. Manual fine-tuning belongs therefore to producing complex commutative diagrams.

Arrow stretching can be regarded as automatic fine-tuning. Manual fine-tuning facilities, on the other hand, include moving labels around, moving arrows around, modifying their size, changing the distance from the source node to the beginning of the arrow, as well as from the end of the arrow to the target node, and setting spacing parameters such as the gap between columns and between rows. Some macro packages provide the facility to adjust these gaps to different values between specific rows or columns, which is essential in order to get the proper perspective of a three-dimensional diagram. Otherwise, empty rows and columns have to be added to the diagram to get the desired perspective. Appendix III shows the degree of automatic stretching provided by each of the macro packages.

Installation. None of the macro packages requires a complex installation procedure, and in most cases the only requirement in order to get the package running is to drop a single macro or style file somewhere in the TEX search path. Some macro packages, however, have accompanying special fonts to get better diagonal lines and arrows, that is, they provide more diagonal slopes and a wider variety of arrow heads and tails to choose from.

In such a case, installation can get more complicated. METAFONT is not as easy to drive or as familiar to the user as TEX or LATEX; many implementations do not make it available, and on others only the system administrator is able to install fonts. A ready-to-use collection of the additional fonts at standard magnifications is distributed, however, with some macro packages.

Documentation. Ranges from small text files to comprehensive user guides, and even to book chapters.

User support. The authors of the different macro packages have been receptive to comments and willing to provide user support. Almost all of the macro packages remain under development and are open to suggestions from users. Moreover, further development of the XY-pic macro package by Kristoffer Rose and Ross Moore is being funded by three different sources.

Ease of use. The relative ease of use of a macro package is a subjective matter, depending to a large extent on previous experiences in using similar macro packages. Nevertheless, there are at least two characteristics of a macro package for typesetting commutative diagrams that are worth mentioning. The way in which the array of cells underlying a commutative diagram has to be conceived is of most importance. The requirement, found in some macro packages, of extra cells for morphisms makes the macro package much more difficult to use, because the user has to add many spurious rows and columns only to hold these morphisms and to get proper spacing, and the code for the diagrams gets bigger and more obscure (compare the last two grids in the previous section).

Orthogonal to the conception of the array of cells is the way in which coordinates for the source and target nodes of the arrows have to be specified. While such addresses are implicit in the name of the arrow in some macro packages, they are absolute coordinates, coordinates relative to the cell where they are declared, or even symbolic coordinates in other macro packages.

The other aspect is the degree of manual fine-tuning needed to achieve a readable commuting diagram. Even when the macro package provides enough facilities, fine-tuning a complex commutative diagram may take more time and effort than conceiving, designing, and coding the whole diagram. Some of the macro packages require visual or measured adjustment by the user of the size and position of every node, arrow, and label, whereas for others most diagrams may be input as easily as any other mathematical formula in TEX and they are typeset nicely without any manual adjustment at all.

Resource requeriments. It is well known that TEX has been designed to support high-quality typesetting of mathematical text, and that it does not offer much built-in support when it comes to drawing and performing arbitrary computations. Because most of the macro packages are built on top of TEX, they are forced to resort to indirect ways of performing computations and to produce large diagrams by juxtaposition of small line and arrow segments. Therefore, a complex diagram may take up lots of computations, line segments, words of TEX memory, and time to typeset. Appendix II compares resource requirements for the different macro packages, showing the main file size together with statistics for the various TEX memory categories. The statistics are based on sample runs to typeset the sample diagrams presented in Appendix I with the different macro packages.

Availability. All the macro packages reviewed in this paper can be found in the CTAN archives, and either are in the public domain or are free software, subject to the terms of the GNU General Public Licence as published by the Free Software Foundation. They are listed in Appendix IV.

Compatibility. Converting a commutative diagram among different macro packages is no straightforward task, not only because of the different approaches to constructing a diagram mentioned in the previous section, but also because of differences in naming conventions and in the available arrow styles and slopes. Converting the sample diagram in Appendix I has taken the author many hours of careful work, and in some cases building the diagram again from scratch for another macro package has proven to be the most efficient solution.

The macro packages are therefore highly incompatible. Nevertheless, the macro package by Paul Taylor provides some initial facilities for emulating other macro packages. Maybe a common, agreed-upon syntax for commutative diagrams (see the last section below) would provide a suitable framework for solving these incompatibilities. Moreover, although it may seem rather natural that the macro packages are not compatible with each other, because the idioms are under development and none of the authors is, in principle, under any obligation to the users of the other macro packages, the adoption of a common standard would have the advantage to the whole user community that the diagrams which have already been drawn with one macro package could be pasted into a document using another macro package.

TEX format requirements. While it would be desirable to be able to typeset a commutative diagram under any derivative of TEX, some macro packages can only run on LATEX because they borrow the `picture` environment and one or more of the special fonts `line10`, `linew10`, `circle10`, and `circlew10`. Other macro packages require AMS-TEX or the `amstex` package in LATEX. The other way round, some macro packages run on TEX but do not run when used in a LATEX document.

Output quality. This is perhaps the most subjective aspect in these guidelines, and therefore it is left for the reader to evaluate. See the sample diagrams in Appendix I, and make a guess at which of the macro packages has been used in Valiente (1994).

Macro packages

The different macro packages are listed in turn in the following, under the name of the respective author, and they are analyzed according to the evaluation guidelines presented in the previous section. No attempt has been made to put them in chronological order of development, and the list is sorted by author name.

American Mathematical Society. AMS-TEX includes some commands for typesetting commutative diagrams, which are also available in AMS-LATEX as a separate option. Only horizontal and vertical arrows are supported, and therefore AMS-TEX can only handle "rectangular" commutative diagrams. Moreover, only "plain" arrows can be used within commutative diagrams, although AMS-TEX provides about 30 different arrow shapes, and arrows do not automatically stretch to their source and target vertices. Commutative diagrams are specified as an array of cells, with either one object or one or more morphisms in each non-empty cell, although unlike matrices, no column separator is needed (a special delimiter has to be used, however, in place of missing arrows). Arrow coordinates are implicit in the name of the arrow and only the four basic directions are available, where arrows can only extend to the adjacent row and/or column in the array. The only fine-tuning facilities provided are a stretching command to force arrows in the same column to be set to the same length (actually, to the width of the longest label in that column), which does not suffice in order to achieve appropriate arrow stretch when the vertices have different width (this manual stretching facility requires the whole `amstex` package to be loaded in AMS-LATEX), and a command to change the minimum arrow width in a diagram, for instance to get it to fit on a page. Documentation is as scarce as the facilities the package provides, only four pages in *The Joy of TEX* and one page in the AMS-LATEX User's Guide.

Barr. Instead of using a matrix notation, commutative diagrams are specified in the macro package developed by Michael Barr by composing more elementary diagrams, using primitive shapes such as squares and triangles. Arrow coordinates are implicit within these shape macros. Additional arrows can be specified by giving the absolute address, within an implicit `picture` environment, of their source node, together with the relative address of their target node as a slope and a length, but stretching is not automatic in these cases. It supports diagonal arrows only in the usual LATEX slopes, and only a few different arrow shapes are available. There are no facilities for diagram fine-tuning. It only runs on LATEX. Documentation consists of a 10-page document, called *The diagram macros*, which explains the principles and gives detailed examples.

Borceux. In the macro package developed by Francis Borceux, commutative diagrams are specified as an array of cells, with one object and all departing morphisms in each non-empty cell. There are facilities for introducing one object and one morphism, or two *crossing* morphisms, in each non-empty cell, but at most two items may belong to the same cell. The delimiter for columns is, unlike the `&` character used in *all* the other macro packages, the special character § that is not even available in many keyboard layouts. It supports diagonal arrows of different shapes and in many different, although not arbitrary, slopes, and it also supports parallel and adjoint (counter-parallel) arrows, some curved arrows, and automatic stretching. Arrow coordinates are implicit in the name of the arrow for the 32 principal directions. Different facilities for diagram fine-tuning are provided. It only runs on LATEX. Documentation consists of a detailed 12-page document, called *User's guide for Diagram 3*. Two restricted macro files are distributed for small TEX implementations, one that only allows for plain arrows and another one that also provides parallel and adjoint (counter-parallel) plain arrows. A further macro file is distributed with the package that provides additional triple, quadruple, and quintuple arrows, parallel and disjoint.

Gurari. Unlike the case of most of the other macro packages, Eitan Gurari has developed a general drawing package on top of TEX. It supports diagonal arrows of different shapes and arbitrary slopes, curved arrows and loops, automatic stretching, and symbolic addressing. Arrow coordinates can be simbolic, because of the possibility of naming any location within a drawing, but they are relative in the sample diagrams presented in the appendices because the macros used are the ones given in page 160 of the book *TEX & LATEX—Drawing and Literate Programming*. It runs on both TEX and LATEX. The macros are well documented in the book, with several basic chapters and one chapter devoted to general grid diagrams, but there is only one page describing commutative diagrams and there is only one sample diagram in the whole book.

Reynolds. John Reynolds has developed a macro package consisting of a collection of general macros for producing a wide variety of diagrams and another collection of macros, which depend on the general macros, for producing commutative diagrams. It supports diagonal arrows only in the usual LATEX shapes and slopes, because the macros depend on the LATEX picture facilities to draw lines, arrows, and circles, although it also supports parallel and adjoint (counter-parallel) arrows, loops, and it provides automatic stretching. Commutative diagrams as specified by giving the absolute coordinates for each node and for the source and target node of each arrow, and approach close to symbolic addressing. Excellent facilities for diagram fine-tuning are provided. It only runs on LaTeX. Documentation consists of a rather cryptic 12-page ASCII file, called *User's Manual for Diagram Macros*, describing the macro package, together with a LaTeX input file that produces a 7-page document of sample diagrams.

Rose. A macro package has been developed by Kristoffer Rose on top of a more general drawing language, called the *XY-pic kernel*. It supports diagonal arrows of different shapes and in many different, although not arbitrary, slopes, and it also supports parallel and adjoint (counter-parallel) arrows, curved arrows, and loops. Arrows stretch automatically, and there are ample facilities for defining additional arrow styles. Commutative diagrams are specified as an array of cells, with one object and all departing morphisms in each non-empty cell. Arrow coordinates for the target node are implicit in the name of the arrow for the 16 principal directions, and they can be absolute or relative for all other directions. Different facilities for diagram fine-tuning are provided. It runs on both TeX and LaTeX. Documentation is excellent, both a comprehensive *XY-pic User's Guide* and a more technical document called *XY-pic Reference Manual* are provided with the package. The latter also describes the XY-pic kernel.

Smith. The *Expanded Plain TeX* macro package includes macros for typesetting commutative diagrams, written by Steven Smith, in a file named `arrow.tex`. It supports diagonal arrows only in the usual LaTeX slopes, because the macros depend on the LaTeX font `line10`, and only a "plain" arrow shape is available, besides pairs of parallel and adjoint (counter-parallel) arrows. Commutative diagrams are specified as an array of cells, with either one object or one or more morphisms in each non-empty cell. There is not any automatic stretching of arrows. Arrow coordinates are implicit in the name of the arrow for the four basic directions, and they are relative addresses for all other directions. Designing a complex diagram using this macro package is as difficult as fine-tuning a simple diagram, even requiring manual computations of horizontal and vertical dimensions to get a desired arrow size and slope. It runs on both TeX and LaTeX. Documentation is enough to cover the facilities provided by the macros, seven pages in *Expanded Plain TeX* and a two-page source document named `commdiags.tex`, reproducing eleven textbook commutative diagrams.

Spivak. LAMS-TeX includes an environment for producing commutative diagrams that supports diagonal arrows of different shapes and in many different, although not arbitrary, slopes. Arrows stretch automatically, and there are ample facilities for defining additional arrow styles. Commutative diagrams are specified as an array of cells, with one object and all departing morphisms in each non-empty cell. Arrow coordinates are relative addresses, and mnemonics can be easily defined for the most common arrow coordinates. Superb facilities for diagram fine-tuning are provided. It only runs on TeX. Documentation is excellent, two chapters of the book *LAMS-TeX—The Synthesis* describing every detail from diagram design to coding and fine-tuning.

Taylor. A macro package has been developed by Paul Taylor that supports diagonal arrows of different shapes and slopes, and even at arbitrary slopes (implemented by rotating horizontal arrows through PostScript `\special` commands). Arrows stretch automatically, and there are ample facilities for defining additional arrow styles. Commutative diagrams are specified as an array of cells, with either one object or one or more morphisms in each non-empty cell. Arrow coordinates are implicit in the name of the arrow, and they are complemented with explicit slope and length parameters. There are plenty of options for diagram fine-tuning, either global to the whole document of local to a single diagram. It runs on both TeX and LaTeX. Documentation is excellent, a quite comprehensive document calles *Commutative Diagrams in TeX (Version 4)* that is even provided typeset in booklet format.

Van Zandt. As in the case of the macro packages by Eitan Gurari, PSTricks is a general drawing package built on top of TeX. Instead of extending TeX by defining graphics primitives, however, it is a collection of PostScript-based TeX macros, and it can be seen in fact as a high-level TeX-like interface to the PostScript language. It supports diagonal arrows of different shapes and arbitrary slopes, curved arrows and loops, automatic stretching, and symbolic addressing for both node and arrow coordinates. It runs on both TeX and LaTeX. The macros are well documented in the *PSTricks User's Guide*, although there are only two pages describing commutative diagrams and there are only two sample diagrams in the whole document.

Discussion

Syntactic issues. Syntactic issues are so fundamental to user acceptance of a macro package for typesetting commutative diagrams, that a volunteer task within the LaTeX3 project has been founded in October 1992 under the name *Research on Syntax for Commutative Diagrams*, with Paul Taylor as coordinator and Michael Barr and Kristoffer Rose as members.

After an initiative by Michael Barr, who started a discussion within the categorical community about the best syntax to adopt for commutative diagrams in LaTeX3, a rather heated debate has taken place in the `CATEGORIES` discussion list. There have been many contributions between June and August 1993, although the discussion list has been silent in these matters ever since.

Curved arrows. The need for curved arrows arises when "parallel" morphisms have to be distinguished from each other, for instance when it is not known if the morphism $h : A \to C$ is equal to the composition of the morphisms $g : B \to C$ and $f : A \to B$,

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[rr, bend right, "h"'] & B \arrow[r, "g"] & C \end{tikzcd}$$

because otherwise the composite morphism would not need to be made explicit.

The need for curved arrows also arises when there are loops in a diagram. For instance, consider the definition of an isomorphism.

A morphism $f : A \to B$ in a given category is an *isomorphism* if there exist a morphism $g : B \to A$ in that category such that $g \circ f = id_A$ and $f \circ g = id_B$. That is, if the diagram

$$\begin{tikzcd} A \arrow[loop left, "id_A"] \arrow[r, shift left, "f"] & B \arrow[l, shift left, "g"] \arrow[loop right, "id_B"] \end{tikzcd}$$

commutes. One possible trick to eliminate the need for such curved arrows is to "straighten up" the diagram by appropriately duplicating some nodes. For the previous example, a morphism $f : A \to B$ is an isomorphism if the following two diagrams commute.

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[rr, "id_A"', shift right] & B \arrow[r, "g"] & A \end{tikzcd} \qquad \begin{tikzcd} B \arrow[r, "g"] \arrow[rr, "id_B"', shift right] & A \arrow[r, "f"] & B \end{tikzcd}$$

These diagrams, however, look much better with a curved arrow,

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[rr, bend right, "id_A"'] & B \arrow[r, "g"] & A \end{tikzcd} \qquad \begin{tikzcd} B \arrow[r, "g"] \arrow[rr, bend right, "id_B"'] & A \arrow[r, "f"] & B \end{tikzcd}$$

and therefore the need for curved arrows cannot always be eliminated without sacrificing diagram clarity and, perhaps arguably, esthetics. While some authors of category theory textbooks seem to prefer to duplicate nodes, others make a thorough use of curved arrows.

Design issues. Diagrams are essentially a communication medium, and therefore good design means a design for readability. Although readability issues can be as subjective as esthetics issues, however, some basic principles may help in the design of readable diagrams. The first principle is to follow the natural order of writing, which at least within occidental writing conventions means left to right, top to bottom, and foreground to background. A second principle is to appropriately give depth to three-dimensional diagrams, in such a way that the foreground lies a little below the background. This principle finds no easy justification, because it may seem to contradict the top-to-bottom order of writing by imposing a bottom-to-top order from foreground to background, but it is true of all kinds of pictorial representations.

User interface. Most of the macro packages provide a simple user interface, consisting of a certain matrix notation. While it adheres to the LaTeX principle of emphasizing structural descriptions, such a specification may become much too obscure for a complex diagram. Some authors have argued against the use of alternative technologies (if you want WYSIWYG, use a pen and paper) but maybe the time has arrived to have a state-of-the-art drawing program with specific facilities for designing commutative diagrams. One possible scenario would be to sketch the arrangement of nodes and arcs on the computer screen using a mouse, and to let the drawing program translate the design into the language of (any of) the macro packages, taking care of all the time-consuming details of computing coordinates, choosing appropriate slops for the arrows, placing arrow labels, fine-tuning, etc. Further facilities could include, for instance, trying different layouts based both on the structural description of the diagram as a graph and on knowledge of the kind of graphs that commutative diagrams are, and performing specific operations on descriptions such as, for instance, obtaining the *dual* of a commutative diagram.

Open issues. Although the conceptual framework used for evaluating the different macro packages resulted from the experience of the author using them and converting diagrams between them, it is precisely because of the evaluation having been carried out by only one person that the resulting data may be somewhat biased. A more general investigation would involve mathematicians and computer scientists writing their own diagrams, as well as (LA)TEX-competent secretaries typing their work, and would produce quantitative measures of learning times for the different macro packages and, once they are fluent in each macro package, measures of the time it takes them to transcribe a diagram drawn on paper.

Further additional investigations include evaluating the degree of help given by each macro package towards improving the quality of the output diagrams, for instance by means of informative messages; quantifying the degree of fine-tuning needed with each macro package in order to produce a complex diagram; evaluating the robustness of the different macro packages when the user makes common errors, such as omitting brackets or mistyping command names; and, last but not least, designing a standard library of common diagrams against which the different macro packages could be evaluated and compared.

Acknowledgement

In order to avoid name clashes among the control sequences defined in the different macro packages, all the diagrams have been typeset separately and included in the final document as encapsulated PostScript files. Thanks to Michel Goossens and Sebastian Rahtz for their advice. Ricardo Alberich Martí provided guidance during the design of the experiment to obtain time statistics. Special thanks are also debt to the anonymous reviewer for many detailed comments and suggestions.

Bibliography

American Mathematical Society. *AMS-LATEX Version 1.1 User's Guide*. Electronic document distributed with the package, August 1991.

Michael Barr. *The diagram macros*. Electronic document distributed with the package.

Francis Borceux. *User's guide for Diagram 3*. Electronic document distributed with the package.

Eitan M. Gurari. *TEX & LATEX—Drawing and Literate Programming*. McGraw-Hill, New York, 1994.

Donald E. Knuth. *The TEXbook*. Addison-Wesley, 1984.

John Reynolds. *User's Manual for Diagram Macros*. Electronic document distributed with the package, December 1987.

Kristoffer H. Rose. *XY-pic User's Guide*. Electronic document distributed with the package, March 1994.

Kristoffer H. Rose and Ross Moore. *XY-pic Reference Manual*. Electronic document distributed with the package, March 1994.

Steven Smith. *Arrow-theoretic diagrams*. Chapter 5 in Karl Berry and Steven Smith, *Expanded Plain TEX*. Electronic document distributed with the package, March 1993.

Michael D. Spivak. *The Joy of TEX—A Gourmet Guide to Typesetting with the AMS-TEX Macro Package*. American Mathematical Society, second edition, 1990.

Michael D. Spivak. *LAMS-TEX—The Synthesis*. The TEXplorators Corporation, Houston, Texas, 1989.

Paul Taylor. *Commutative Diagrams in TEX (Version 4)*. Electronic document distributed with the package, January 1993.

Gabriel Valiente Feruglio. *Knowledge Base Verification using Algebraic Graph Transformations*. PhD thesis, University of the Balearic Islands, June 1994.

Timothy Van Zandt. *PSTricks User's Guide*. Electronic document distributed with the package, March 1993.

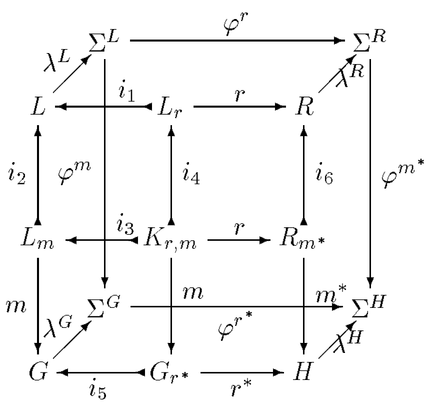
Appendix I: Sample diagrams

The following diagrams reproduce a fairly complex commutative diagram, taken from Valiente (1994), using all the macro packages reviewed in this paper. The diagram consists of a pushout construction of partial closed morphisms of total unary algebras in the foreground, together with a corresponding pushout construction of total morphisms of total signature algebras in the background.

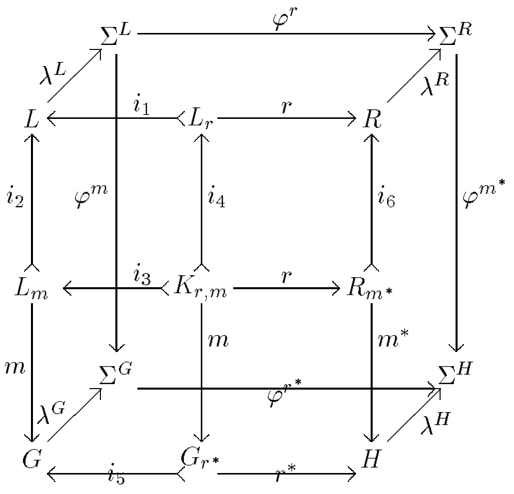
American Mathematical Society (no diagonal arrows available)

$$\begin{tikzcd}
L & L_r \arrow[l, "i_1"'] \arrow[r, "r"] & R \\
L_m \arrow[u, "i_2"] \arrow[d, "m"'] & K_{r,m} \arrow[l, "i_3"'] \arrow[r, "r"] \arrow[u, "i_4"'] \arrow[d, "m"] & R_{m^*} \arrow[u, "i_6"'] \arrow[d, "m^*"] \\
G & G_{r^*} \arrow[l, "i_5"] \arrow[r, "r^*"'] & H
\end{tikzcd}$$

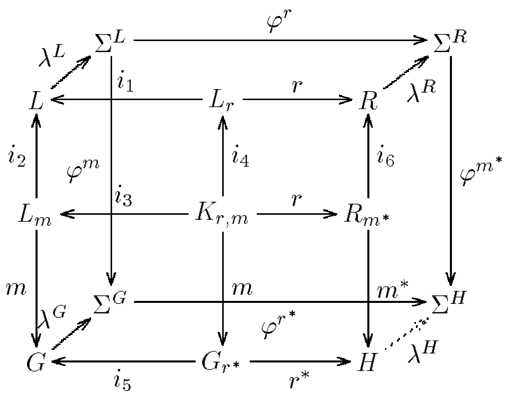
Michael Barr `diagram.tex`



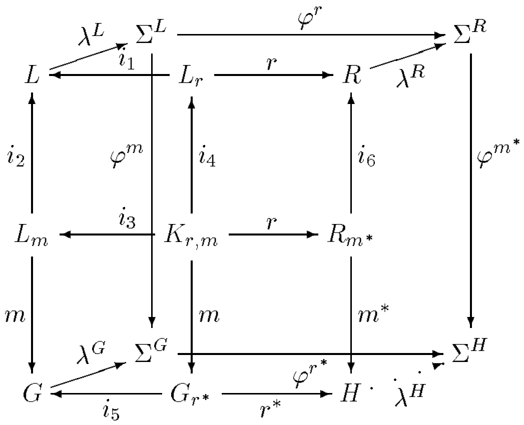
Francis Borceux `Diagram`



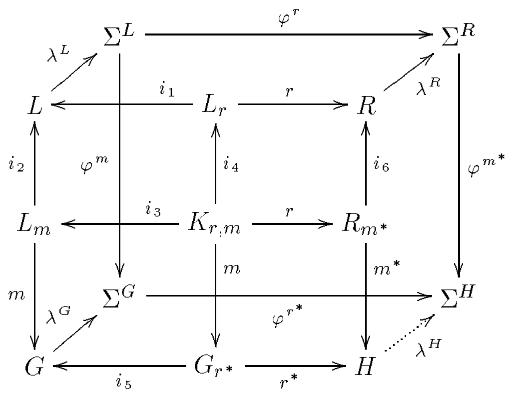
Eitan Gurari `AlDraTeX.sty`



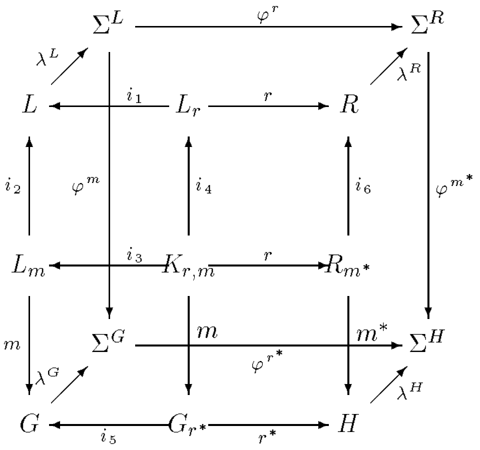
John Reynolds `diagmac.sty`



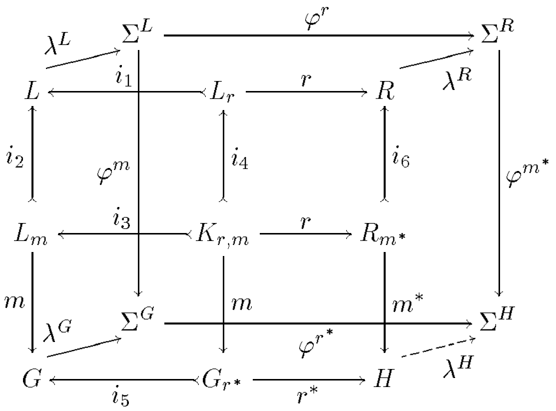
Kristoffer Rose XY-pic



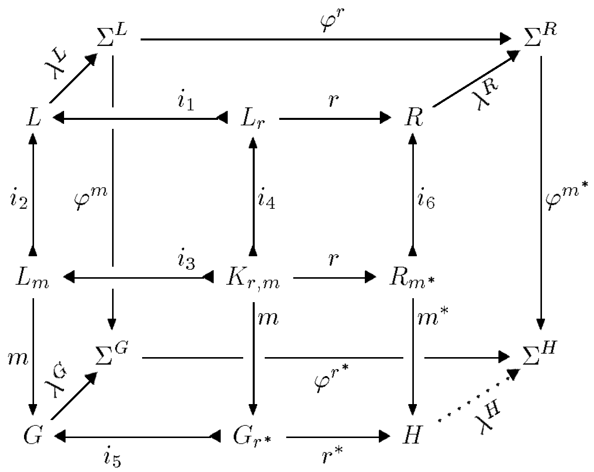
Steven Smith `arrow.tex`



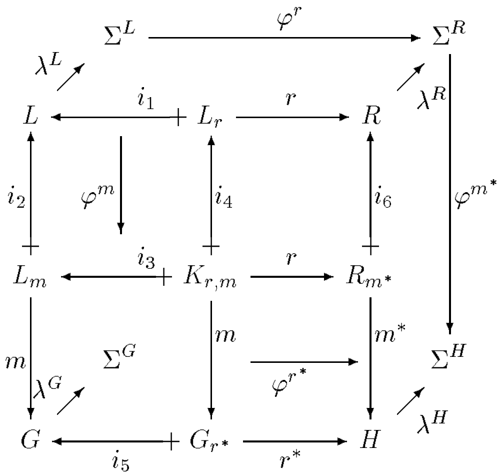
Michael Spivak $\mathcal{L}\mathcal{A}\mathcal{M}\mathcal{S}$-TEX



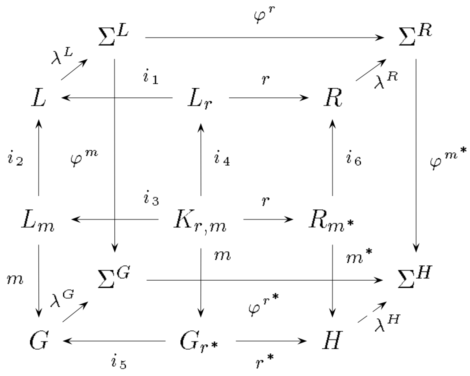
Paul Taylor `diagrams.tex`



Paul Taylor (emulating Francis Borceux)



Timothy Van Zandt `PSTricks`



Appendix II: Source code for the sample diagrams

```
\newcommand{\up}[1]{\raisebox{1em}{$#1$}}
\newcommand{\down}[1]{\raisebox{-1em}{$#1$}}
\newcommand{\Left}[1]{\makebox[5pt][r]{$#1$}}
\newcommand{\Right}[1]{\makebox[5pt][l]{$#1$}}
```

American Mathematical Society

```
$$\begin{CD}
L       @<i_1<< L_r     @>r>>   R       \\
@Ai_2AA         @AAi_4A         @AAi_6A \\
L_m     @<i_3<< K_{r,m} @>r>>   R_{m^*} \\
@VmVV           @VVmV           @VVm^*V \\
G       @<<i_5< G_{r^*} @>>r^*> H
\end{CD}$$
```

Michael Barr

```
$$\bfig
\putsquare<-2`-2`-2`-2;500`500>(0,500)[L`L_r`L_m`K_{r,m};\qquad i_1`i_2`i_4`]
\putsquare<1`0`-2`1;500`500>(500,500)[\phantom{L_r}`R`\phantom{K_{r,m}}`R_{m^*};r``i_6`]
\putsquare<0`1`1`-2;500`500>(0,0)[\phantom{L_m}`\phantom{K_{r,m}}`G`G_{r^*};\qquad i_3`m`\up m`i_5]
\putsquare<0`0`1`1;500`500>(500,0)%
  [\phantom{K_{r,m}}`\phantom{R_{m^*}}`\phantom{G_{r^*}}`H;r``\up{m^*}`r^*]
\putsquare<1`1`1`1;1000`1000>(250,250)%
  [\Sigma^L`\Sigma^R`\Sigma^G`\Sigma^H;\varphi^r`\varphi^m`\varphi^{m^*}`\varphi^{r^*}]
\putmorphism(125,1125)(1,1)%
  [\phantom L`\phantom{\Sigma^L}`{\up{\Right{\lambda^L}}}]{0}{1}{l}
\putmorphism(1125,1125)(1,1)%
  [\phantom R`\phantom{\Sigma^R}`{\down{\Left{\lambda^R}}}]{0}{1}{r}
\putmorphism(125,125)(1,1)
  [\phantom G`\phantom{\Sigma^G}`{\up{\Right{\lambda^G}}}]{0}{1}{l}
\putmorphism(1125,125)(1,1)%
  [\phantom H`\phantom{\Sigma^H}`{\down{\Left{\lambda^H}}}]{0}{1}{r}
\efig$$
```

Francis Borceux

```
\setdefaultscale{40}
\begin{diagram}
? ? \Sigma^L ? ? ? ? \Ear[280] {\varphi^r} ? ? ? ? \Sigma^R ??
? \Near[50] {\lambda^L} ? ? ? ? ? ? ? ? \neaR[50] {\lambda^R} ??
L ? ? \Wmono[130] {\qquad i_1} ? ? L_r ? ? \Ear[130] r ? ? R ?? ??
\Nmono[130] {i_2} ? ? \Sar[280] {\varphi^m} ? ? \nmonO[130] {i_4} ? ? ? ? \nmonO[130] {i_6} ? ?
  \saR[280] {\varphi^{m^*}} ?? ??
L_m ? ? \Wmono[100] {\qquad i_3} ? ? K_{r,m} ? ? \Ear[100] r ? ? R_{m^*} ?? ??
\Sar[130] m ? ? \Sigma^G ? ? \saR[130] {\up{m}} ? ? \eaR[280] {\varphi^{r^*}} ? ?
  \saR[130] {\up{m^*}} ? ? \Sigma^H ??
? \Near[50] {\lambda^G} ? ? ? ? ? ? ? ? \neaR[50] {\lambda^H} ??
G ? ? \wmonO[130] {i_5} ? ? G_{r^*} ? ? \eaR[130] {r^*} ? ? H ??
\end{diagram}
```

Eitan Gurari

```
\Draw
\PenSize(0.25pt)
\ArrowSpec(V,5,3,2)
\ArrowHeads(1)
\GridSpace(10,10)
\GridDiagramSpec()(\MyEdge)
\Define\L(4){,+#1..+#2\,L\,#3\,#4}
\Define\D(4){,+#1..+#2\,D\,#3\,#4}
```

```
\Define\MyEdge(5){
   \IF \EqText(#3,D) \THEN
      \EdgeSpec(D)
   \ELSE
      \EdgeSpec(L)
   \FI
   \IF \EqText(#1,#2) \THEN
      \RotateTo(#4)
      \CycleEdge(#1)
      \EdgeLabel(--$#5$--)
   \ELSE
      \Edge(#1,#2)
      \IF \EqText(,#4) \THEN
         \EdgeLabel(--$#5$--)
      \ELSE
         \EdgeLabel[#4](--$#5$--)
      \FI
   \FI}
\GridDiagram(8,8)()()({
& $\Sigma^L$ \L(6,0,+,\mbox{$\varphi^m$}) \L(0,6,,\mbox{$\varphi^r$}) & & & & & $\Sigma^R$
  \L(6,0,,\mbox{$\varphi^{m^*}$}) //
$L$ \L(-1,1,,\mbox{$\lambda^L$}) & & & $L_r$ \L(0,-3,+,\mbox{$i_1$}) \L(0,3,,\mbox{$r$}) & & &
  $R$ \L(-1,1,+,\mbox{$\lambda^R$}) & //
& & & & & & & //
& & & & & & & //
$L_m$ \L(3,0,+,\mbox{$m$}) \L(-3,0,,\mbox{$i_2$}) & & & $K_{r,m}$ \L(-3,0,+,\mbox{$i_4$})
  \L(3,0,,\mbox{$m$}) \L(0,-3,+,\mbox{$i_3$}) \L(0,3,,\mbox{$r$}) & & & $R_{m^*}$
  \L(3,0,,\mbox{$m^*$}) \L(-3,0,+,\mbox{$i_6$}) & //
& & & & & & & //
& $\Sigma^G$ \L(0,6,+,\mbox{$\varphi^{r^*}$}) & & & & & $\Sigma^H$ //
$G$ \L(-1,1,,\mbox{$\lambda^G$}) & & & $G_{r^*}$ \L(0,-3,,\mbox{$i_5$}) \L(0,3,+,\mbox{$r^*$})
  & & & $H$ \D(-1,1,+,\mbox{$\lambda^H$}) & //})
\EndDraw
```

John Reynolds

```
\def\diagramunit{0.6pt}
$$\ctdiagram{
\ctv 0,0:{G}
\ctv 100,0:{G_{r^*}}
\ctv 200,0:{H}
\ctv 0,100:{L_m}
\ctv 100,100:{K_{r,m}}
\ctv 200,100:{R_{m^*}}
\ctv 0,200:{L}
\ctv 100,200:{L_r}
\ctv 200,200:{R}
\ctel 0,100,0,200:{i_2}
\cter 100,100,100,200:{i_4}
\cter 200,100,200,200:{i_6}
\ctel 0,100,0,0:{m}
\cter 100,100,100,0:{m}
\cter 200,100,200,0:{m^*}
\ctetg 100,200,0,200;60:{i_1}
\ctetg 100,100,0,100;60:{i_3}
\cteb 100,0,0,0:{i_5}
\ctet 100,200,200,200:{r}
\ctet 100,100,200,100:{r}
\cteb 100,0,200,0:{r^*}
\ctv 75,25:{\Sigma^G}
```

```
\ctv 275,25:{\Sigma^H}
\ctv 75,225:{\Sigma^L}
\ctv 275,225:{\Sigma^R}
\ctet 0,0,75,25:{\lambda^G}
\ctdot
\cteb 200,0,275,25:{\lambda^H}
\ctsolid
\ctet 0,200,75,225:{\lambda^L}
\cteb 200,200,275,225:{\lambda^R}
\ctelg 75,225,75,25;150:{\varphi^m}
\cterg 275,225,275,25;150:{\varphi^{m^*}}
\ctet 75,225,275,225:{\varphi^r}
\cteb 75,25,275,25:{\varphi^{r^*}}
}$$
```

Kristoffer Rose

```
\definemorphism{unique}\dotted\tip\notip
\spreaddiagramrows{-1pc}
\spreaddiagramcolumns{-1pc}
\diagram
& \Sigma^L \xto[dddd]_{\varphi^m} \xto[rrrr]^{\varphi^r} & & & & \Sigma^R
  \xto[dddd]^{\varphi^{m^*}} \\
L \urto^{\lambda^L} & & \llto_<<<<{i_1} L_r \rrto^r & & R \urto_{\lambda^R} \\ \\
L_m \uuto^{i_2} \ddto_m & & \llto_<<<<{i_3} \uuto_{i_4} K_{r,m} \ddto^<<<<m \rrto^r & &
  \uuto_{i_6} R_{m^*} \ddto^<<<<{m^*} \\
& \Sigma^G \xto[rrrr]_{\varphi^{r^*}} & & & & \Sigma^H \\
G \urto^{\lambda^G} & & \llto^{i_5} G_{r^*} \rrto_{r^*} & & H \urunique_{\lambda^H}
\enddiagram
```

Steven Smith

```
\harrowlength=45pt
\sarrowlength=.30\harrowlength
$$\gridcommdiag{
& & \Sigma^L & & & & {\harrowlength=100pt\mapright^{\varphi^r}}
  & & & & \Sigma^R \cr
& \arrow(1,1)\lft{\lambda^L} & & & & & & & & \arrow(1,1)\rt{\lambda^R} \cr
L & & \mapleft^{\qquad i_1} & & L_r & & \mapright^r & & R \cr \cr
\mapup^{i_2} & & {\varrowlength=100pt\mapdown^{\varphi^m}}
  & & \mapup_{i_4} & & & & \mapup_{i_6} & &
  {\varrowlength=100pt\mapdown_{\varphi^{m^*}}}
  \cr \cr
L_m & & \mapleft^{\qquad i_3} & & K_{r,m} & & \mapright^r & & R_{m^*} \cr \cr
\mapdown^m & & \Sigma^G & & \mapdown_{\up{m}}
  & & {\harrowlength=100pt\mapright_{\varphi^{r^*}}}
  & & \mapdown_{\up{m^*}} & & \Sigma^H \cr
& \arrow(1,1)\lft{\lambda^G} & & & & & & & & \arrow(1,1)\rt{\lambda^H} \cr
G & & \mapleft_{i_5} & & G_{r^*} & & \mapright_{r^*} & & H
}$$
```

Michael Spivak

```
$$\Cgaps{0.5}
\Rgaps{0.5}
\cgaps{1.3;0.7;1;1;1.3}
\rgaps{0.7;1;1;1.3;0.7}
\CD
  & \Sigma^L @() \L{\varphi^m} @(0,-4) @() \L{\varphi^r} @(4,0)
  & & &
  & \Sigma^R @() \l{\varphi^{m^*}} @(0,-4)
\\
```

```
L @() \L{\lambda^L} @(1,1)
  & &
  L_r @() \L{i_1} \0t @(-2,0) @() \L{r} @(2,0)
  & & R @() \l{\lambda^R} @(1,1)
  &
\\
\\
L_m @() \L{i_2} \0t @(0,2) @() \L{m} @(0,-2)
  & & K_{r,m} @() \L{i_3} \0t @(-2,0) @() \L{r} @(2,0) @() \l{i_4} \0t @(0,2) @() \l{m} @(0,-2)
  & & R_{m^*} @() \l{i_6} \0t @(0,2) @() \l{m^*} @(0,-2)
  &
\\
  & \Sigma^G @() \l{\varphi^{r^*}} @(4,0)
  & & & & \Sigma^H
\\
  G @() \L{\lambda^G} @(1,1)
  & & G_{r^*} @() \l{i_5} \0t @(-2,0) @() \l{r^*} @(2,0)
  & & H @() \l{\lambda^H} \a- @(1,1)
& \\
\endCD$$
```

Paul Taylor

```
\diagramstyle[heads=littleblack,size=1.5em,PS]
\begin{diagram}
& & \Sigma^L & & & & \rTo^{\varphi^r} & & & & \Sigma^R \\
& \ruTo^{\lambda^L} & \vLine & & & & & & & \ruTo_{\lambda^R} & \\
L & & \HonV & \lEmbed^{i_1} & L_r & & \rTo^r & & R & & \\
& & & & & & & & & & \\
\uEmbed^{i_2} & & \vLine^{\varphi^m} & & \uEmbed_{i_4} & & & & \uEmbed_{i_6} & &
  \dTo_{\varphi^{m^*}} \\
& & & & & & & & & & \\
L_m & & \HonV & \lEmbed^{i_3} & K_{r,m} & & \rTo^r & & R_{m^*} & & \\
& & \dTo & & \dTo_m & & & & \dTo_{m^*} & & \\
\dTo^m & & \Sigma^G & \hLine & \VonH & & \hLine_{\varphi^{r^*}} & & \VonH & \rTo & \Sigma^H \\
& \ruTo^{\lambda^G} & & & & & & & & \ruDotsto_{\lambda^H} & \\
G & & \lEmbed_{i_5} & & G_{r^*} & & \rTo_{r^*} & & H & & \\
\end{diagram}
```

Paul Taylor emulating Francis Borceux

```
\diagramstyle[size=1.5em]
\begin{diagram}
& & \Sigma^L & & & & \Ear {\varphi^r} & & & & \Sigma^R \\
& \Near {\lambda^L} & & & & & & & & \neaR {\lambda^R} \\
L & & \Wmono {\qquad i_1} & & L_r & & \Ear r & & R \\ \\
\Nmono {i_2} & & \Sar {\varphi^m} & & \nmonO {i_4} & & & & \nmonO {i_6} & &
  \saR {\varphi^{m^*}} \\ \\
L_m & & \Wmono {\qquad i_3} & & K_{r,m} & & \Ear r & & R_{m^*} \\ \\
\Sar m & & \Sigma^G & & \saR {\up{m}} & & \eaR {\varphi^{r^*}} & & \saR {\up{m^*}} & & \Sigma^H \\
& \Near {\lambda^G} & & & & & & & & \neaR {\lambda^H} \\
G & & \wmonO {i_5} & & G_{r^*} & & \eaR {r^*} & & H \\
\end{diagram}
```

Timothy Van Zandt

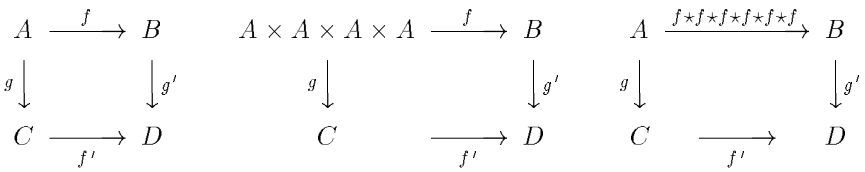
```
$$\setlength{\arraycolsep}{0.1in}
\begin{array}{cccccc}
& \Rnode{SL}{\Sigma^L} & & & & \Rnode{SR}{\Sigma^R} \\ [0.15in]
\Rnode{L}{L} & & \Rnode{Lr}{L_r} & & \Rnode{R}{R} & \\ [0.15in] \\ [0.15in]
\Rnode{Lm}{L_m} & & \Rnode{Krm}{K_{r,m}} & & \Rnode{Rm}{R_{m^*}} & \\ [0.15in]
& \Rnode{SG}{\Sigma^G} & & & & \Rnode{SH}{\Sigma^H} \\ [0.15in]
```

```
\Rnode{G}{G} & & \Rnode{Gr}{G_{r^*}} & & \Rnode{H}{H} & \\ [0.15in]
\end{array}
\psset{nodesep=5pt,arrows=->}
\everypsbox{\scriptstyle}
\ncLine{Lr}{R} \Aput{r}
\ncLine{Krm}{Rm} \Aput{r}
\ncLine{Gr}{H} \Bput{r^*}
\ncLine{Lr}{L} \bput{0}(0.3){i_1}
\ncLine{Krm}{Lm} \bput{0}(0.3){i_3}
\ncLine{Gr}{G} \Aput{i_5}
\ncLine{SL}{SR} \Aput{\varphi^r}
\ncLine{SG}{SH} \Bput{\varphi^{r^*}}
\ncLine{SR}{SH} \Aput{\varphi^{m^*}}
\ncLine{SL}{SG} \Bput{\varphi^m}
\ncLine{Lm}{G} \Bput{m}
\ncLine{Krm}{Gr} \aput{0}(0.3){m}
\ncLine{Rm}{H} \aput{0}(0.3){m^*}
\ncLine{Lm}{L} \Aput{i_2}
\ncLine{Krm}{Lr} \Bput{i_4}
\ncLine{Rm}{R} \Bput{i_6}
\ncLine{L}{SL} \Aput[1pt]{\lambda^L}
\ncLine{R}{SR} \Bput[1pt]{\lambda^R}
\ncLine{G}{SG} \Aput[1pt]{\lambda^G}
\ncLine[linestyle=dashed]{H}{SH} \Bput[1pt]{\lambda^H}$$
```

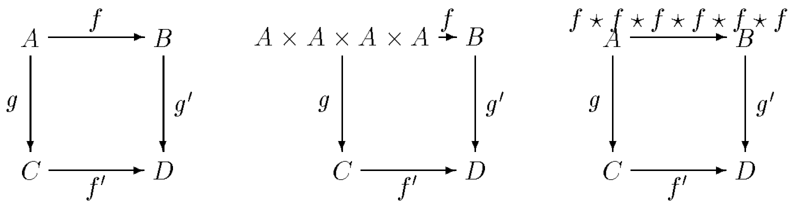
Appendix III: Automatic stretching

The following diagrams illustrate the degree of automatic stretching of arrows provided by each of the macro packages. A simple square diagram is typeset with a long label for the top-leftmost node in order to determine if the bottom horizontal arrow stretches to meet its source node, and it is also typeset with a long label for the top horizontal arrow in order to determine if it stretches long enough to fit the label.

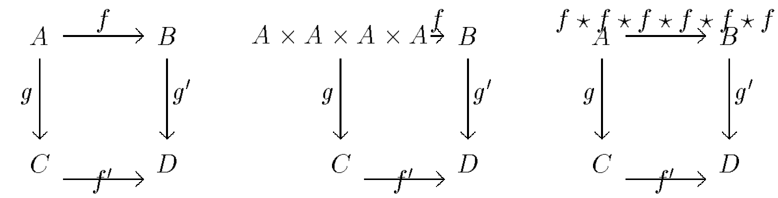
American Mathematical Society Arrows do not stretch to meet their source and target nodes, but they stretch to fit their labels, although only the arrow carrying the long label stretches. Manual fine-tuning is needed in order to get the same stretch in all the other arrows lying in the same column of the array.



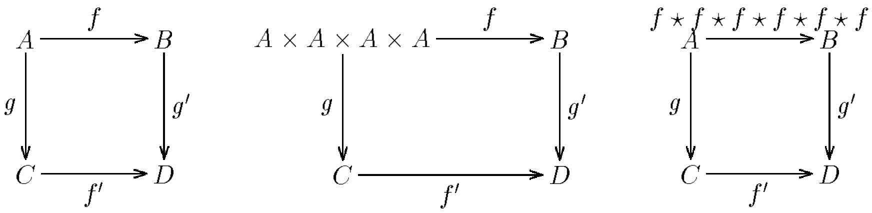
Michael Barr Arrows within the shape macros stretch to meet their source and target arrows, but individual arrows obtained with `\putmorphism` do not. In both cases, arrows do not stretch to fit their labels and the required dimensions have to be given explicitly.



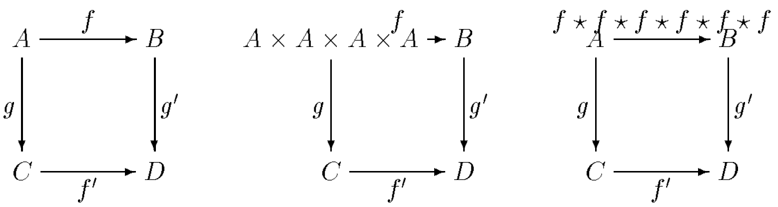
Francis Borceux Arrows stretch to meet their source and target nodes, but they do not stretch to fit their labels.



Eitan Gurari Arrows stretch to meet their source and target nodes, but they do not stretch to fit their labels.



John Reynolds Arrows stretch to meet their source and target nodes, although the labels do not get centered on the stretched arrows. They do not stretch to fit their labels.



Kristoffer Rose Arrows stretch to meet their source and target nodes, although the labels do not get centered on the stretched arrrows. They do not stretch to fit their labels.

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd} \qquad \begin{tikzcd} A \times A \times A \times A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd} \qquad \begin{tikzcd} A \arrow[r, "f\star f\star f\star f\star f\star f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd}$$

Steven Smith Arrows do not stretch to meet their source and target nodes, but they stretch to fit their labels.

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd} \qquad \begin{tikzcd} A \times A \times A \times A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd} \qquad \begin{tikzcd} A \arrow[r, "f\star f\star f\star f\star f\star f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd}$$

Michael Spivak Arrows stretch to meet their source and target nodes, but they do not stretch to fit their labels, even producing overfull `\hbox`es.

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd} \qquad \begin{tikzcd} A \times A \times A \times A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd} \qquad \begin{tikzcd} A \arrow[r, "f\star f\star f\star f\star f\star f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd}$$

Paul Taylor Arrows stretch to meet their source and target nodes, and they also stretch to fit their labels.

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd} \qquad \begin{tikzcd} A \times A \times A \times A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd} \qquad \begin{tikzcd} A \arrow[r, "f\star f\star f\star f\star f\star f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd}$$

Timothy Van Zandt Arrows stretch to meet their source and target nodes, although the labels do not get centered on the stretched arrows. They do not stretch to fit their labels, and the required dimensions have to be given explicitly.

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd} \qquad \begin{tikzcd} A \times A \times A \times A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd} \qquad \begin{tikzcd} A \arrow[r, "f\star f\star f\star f\star f\star f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd}$$

Appendix IV: Resource requeriments

Package size. The following table lists the size (in kilobytes) of the main macro files that have to be loaded into TEX or LATEX in order to use the respective packages.

package	main files	size
AMS-LATEX	`amscd.sty`	10
Barr	`diagram.tex`	40
Borceux	`Diagram`	270
Gurari	`DraTex.sty` and `AlDraTex.sty`	136
Reynolds	`diagmac.sty`	42
Rose	`xypic.tex` and `xy.tex`	68
Smith	`arrow.tex`	24
Spivak[a]	`amstexl.tex` and `lamstex.tex`	200
Taylor	`diagrams.tex`	86
Van Zandt	`pstricks.tex`, `pst-node.tex` and `pstricks.con`	84

[a] Although LAMS-TEX offers much more than the macros for commutative diagrams, it has to be loaded as a whole in order to use the macros. Most such macros can be removed from TEX's memory by loading the file `cd.tox` (4 kilobytes), freeing up about 5800 words of memory, and can be later added again by loading the file `cd.tex` (36 kilobytes), but the whole LAMS-TEX has to be loaded before.

Time statistics. The following table lists statistics for the time (in seconds) needed to typeset the sample diagrams presented in Appendix I, using TEX and LATEX 2_ε on a Macintosh SE/30, with the different macro packages. The mean time and the confidence interval at a significance level of 95% is given for the total time needed to typeset a diagram and for the marginal time, computed as the difference between the time needed to typeset two copies of the sample diagram using a macro package and the time needed to typeset one copy of the same diagram using the same macro package, where these two random variables are assumed to have a normal distribution and to be independent, and where the mean and the variance have been estimated from a sample of 30 observations.

package	total time		marginal time	
	mean	95% confidence interval	mean	95% confidence interval
AMS-LATEX	18.1367	18.0317– 18.2416	1.6600	1.5544– 1.7660
Barr	48.8033	48.7731– 48.8335	29.9334	29.8800– 29.9870
Borceux	127.5630	127.5060–127.6210	28.3170	28.1730– 28.4600
Gurari	388.4630	388.4320–388.4950	638.8270	638.5000–639.1490
Reynolds	46.7000	46.6357– 46.7643	26.8200	26.7520– 26.8880
Rose	242.7400	242.3810–243.0990	210.0730	209.2900–210.8500
Smith	22.9600	22.9031– 23.0169	5.2400	5.1817– 5.2980
Spivak	37.3000	37.2445– 37.3555	11.9833	11.9263– 12.0400
Taylor	66.8400	66.7553– 66.9247	14.3133	14.1420– 14.4850
Taylor emulating Borceux	67.3533	67.3243– 67.3823	11.4767	11.4360– 11.5170
Van Zandt	37.8233	37.7809– 37.8657	14.2100	14.1520– 14.2680

Appendix V: Availability

Availability. The following table lists the CTAN directories where the different macro packages are stored, together with the authoritative FTP addresses they are mirrored from.

package	CTAN **directory**	**mirrored from**
$\mathcal{AMS}$-LaTeX	`macros/ams/amslatex/`	`e-math.ams.org:/ams`
Barr	`macros/latex/contrib/misc/diagram.tex`	(not mirrored)
Borceux	`macros/generic/diagrams/borceux/`	`theory.doc.ic.ac.uk:` `/tex/contrib/Borceux/diagram-3`
Gurari	`macros/generic/dratex/`	(not mirrored)
Reynolds	`macros/latex/contrib/misc/diagmac.sty`	(not mirrored)
Rose	`macros/generic/diagrams/xypic/`	`ftp.diku.dk:/diku/users/kris/TeX`
Smith	`macros/eplain/arrow.tex`	`ftp.cs.umb.edu:/pub/tex/eplain`
Spivak	`macros/lamstex`	(not mirrored)
Taylor	`macros/generic/diagrams/taylor/`	`theory.doc.ic.ac.uk:` `/tex/contrib/Taylor/tex`
Van Zandt	`graphics/pstricks/`	`princeton.edu:/pub/tvz/pstricks`