The **scalerel** Package

Routines for constrained scaling and stretching of objects,
relative to a reference object or in absolute terms

Steven B. Segletes
steven.b.segletes.civ@mail.mil

December 29, 2016
v1.8

1   Commands and Description

The scalerel package is used to scale and vertically stretch objects, either relative to other objects, or in absolute terms. Its commands may be invoked in either math mode or text mode (if there is no math in the objects to manipulate); however, the objects it manipulates will, by default, be processed in math mode. Nonetheless, one may process the objects in text mode (assuming they do not contain math-specific characters or commands) by [non-intuitively] specifying arguments as `{$object$}`, instead of the usual `{object}`.

There are four basic commands with the scalerel package, two of which have star variants:

```
\scalerel[*][max_width]{object}{reference}
\stretchrel[*][narrowest_aspect]{object}{reference}
\scaleto[max_width]{object}{height}
\stretchto[narrowest_aspect]{object}{height}
```

In addition, there are a number of added commands that have been implemented as composites of the four basic commands, they are

```
\scaleleftright[max_width]{left-obj.}{reference}{right-obj.}
\stretchleftright[narrowest_aspect]{left-obj.}{reference}{right-obj.}
\hstretch{scale}{object}
\vstretch{scale}{object}
\scaleobj{scale}{object}
```

Unless delimited for text mode, `object` is assumed to be a math object and will be processed in the *current* math style.[1] On the other hand, `reference`

---

[1] By LaTeX math styles, we mean `\displaystyle`, `\textstyle`, `\scriptstyle`, and `\scriptscriptstyle`.

1

will take on the current mode (text or math). If in math mode, `reference` will likewise be taken in the current math style.

There are also several auxiliary commands, useful in their own right:

```
\ThisStyle{...\SavedStyle...}
\LMex
\LMpt
\def\scriptstyleScaleFactor{0.7}
\def\scriptscriptstyleScaleFactor{0.5}
\Isnextbyte[[v] OR q]{test byte}{string}
\ignoremathstyle[D]
\discernmathstyle
```

In the case of all scalerel macros, `object` is the object to be scaled or stretched. It can be as simple as a symbol, like a summation sign ($\sum$), or it can be an object of complex description. Likewise, in cases of *relative* scaling and stretching, `reference` is the reference object in relation to which the manipulated `object` is scaled or vertically stretched. The `reference` may likewise be of complex description (such as a mathematical expression). When an `object` is scaled or stretched *relative* to a `reference` object, it is also vertically shifted, if necessary, so that its vertical extent conforms to that of the `reference` object.

In cases of scaling or stretching *to* a specified size, `height` will be the *total* vertical height + depth of the `object` following a scale or stretch manipulation. Since scaling or stretching to an absolute size provides no `reference` object, the baseline of the manipulated `object` remains unchanged.

In cases of scaling (relative or absolute), the constraint `max_width` is optionally specified as the maximum width allowable for the manipulated object. If the manipulated width would otherwise exceed this limit, the `object` width is reduced to meet this limit. If the `object` width is reduced, its aspect ratio will change with respect to its original shape.

By definition, vertical stretching will increase the (height-to-width) aspect ratio of an object. In the case of stretching (relative or absolute), the constraint `narrowest_aspect` is the narrowest aspect ratio allowed by the stretch.[2] Its value is given in %, such that a parameter value of `100` indicates 100% or an aspect ratio of 1. If the stretch would otherwise cause the manipulated `object`'s aspect ratio to become more narrow than (*i.e.*, larger than) this value, the width of the manipulated `object` is increased to meet this threshold. The value of `narrowest_aspect` must be an integer.

Because *relative* scaling or stretching is done *relative* to a reference object, the

---

[2]Prior to V1.8 of the package, the limiting aspect ratio was erroneously based on that portion of the stretched object *above* the baseline, rather than the object's full vertical extent.

\scalerel \stretchrel	\scalerel and \stretchrel commands will, by default, print out the manipulated object followed immediately by the reference object. Because this may not always be desired, the star (*) version of these commands suppresses the output of the reference object, so that only the object that was manipulated is output.
\scaleleftright \stretchleftright	The \scaleleftright and \stretchleftright commands are intended to provide the functional capability of the \left and \right commands in math mode, for delimiting symbols not otherwise supported by those commands.
\hstretch \vstretch \scaleobj	The \hstretch and \vstretch commands merely take the second argument and either horizontally or vertically stretch it by a scale factor given by the first argument. The scale is not given in percent, like some aforementioned commands. Remember also, that these commands are, by default processed in math mode. In a like manner, \scaleobj performs a scale of the object. It is like the \scalebox command of the graphicx package, except that its argument (like others in this package) defaults to math mode, and is displayed in the current math display style.
\ThisStyle \SavedStyle	A useful auxiliary command \ThisStyle{} has been introduced for remembering the current math style. When invoked, this command will remember the math style present at the time of its invocation and then execute its argument. Within the extent of its argument, an invocation of the macro \SavedStyle will reinstate the math style that was active upon invocation. This command is very useful when LaTeX boxes are being set within the math environment, since the prevailing math style does not carry into the box, without the use of \SavedStyle.
\LMex \LMpt \scriptstyleScaleFactor \scriptscriptstyleScaleFactor	The lengths \LMex and \LMpt work in conjunction with \ThisStyle and are what I am calling the "local-mathstyle 'ex'" and "local-mathstyle 'pt'," respectively. They are lengths, available for use within the argument of \ThisStyle, that scale with the current mathstyle, equalling 1ex and 1pt in \displaystyle and \textstyle math, and scaled by a factor of 0.7 in \scriptstyle and 0.5 in \scriptscriptstyle. Such a scalable length is useful when scalerel is used to place objects relative to each other across various mathstyles. The 0.7 and 0.5 relative height factors apply to the default Computer Modern font. For other fonts, these scale factors may be reset with a \def on the macros \scriptstyleScaleFactor and \scriptscriptstyleScaleFactor.
\Isnextbyte	A service routine \Isnextbyte was employed by this package. Because of its more general utility, it is made accessible to the user. It is an improved version of the command \isnextbyte from the stringstrings package. It will determine if the first byte from a passed string argument matches a specified *match*-character. It produces a T or F result, which is printed out by default. With the use of the q optional argument, the output can be suppressed, with the result instead being stored in a macro \theresult.

1.1 Efficiency of nested **scalerel** macros

As of version 1.4, scalerel commands will autodetect the current math display style (so-called display-style, text-style, script-style, and/or scriptscript-style) and process its arguments in the same mode. The overhead associated with this capability is an invocation of `\mathchoice`, which must create 4 boxes (one in each math style) before selecting one to typeset. While a small overhead in itself, an efficiency issue can arise when scalerel macros are nested, since if 3 scalerel macros are nested, `\mathchoice` will build $4^3 = 64$ boxes in order to select the proper one to typeset (and a nesting level of 4 yields 256 boxes).

`\ignoremathstyle`
With the use of `\ignoremathstyle`, the math style of the scalerel processing can be fixed in a single math style, thus eliminating the construction of `\mathchoice` boxes. The increase in processing efficiency can be noticeable when operating on deeply nested scalerel macros.

The invocation of `\ignoremathstyle` will force all subsequent scalerel calls to process its arguments in `\textstyle` math. Alternately, `\ignoremathstyle[D]` will force the processing of scalerel macros in `\displaystyle` math.

`\discernmathstyle`
This streamlining can be undone, reinstating the more general math-style preserving approach, with the invocation of `\discernmathstyle`. These macros can be used anywhere in a document by a user, to turn off and on the `\mathchoice` feature of scalerel, or they may be used within user macro definitions, to fix the math style by which scalerel processes its macros within the confines of the user macro.

2 Usage Examples

Now for a few examples. Let us define

```
\def\preblob{\displaystyle\sum_{i=0}^3}
\def\blob{\displaystyle\frac{\displaystyle\frac{x^3}{z+r^3}}%
        {\displaystyle\frac{y}{x^2}}%
}
```

Here are the raw definitions of `$\preblob\blob$`, unscaled:

$$\sum_{i=0}^{3} \frac{\dfrac{x^3}{z+r^3}}{\dfrac{y}{x^2}}$$

2.1   The \scalerel Command

Now we employ $\scalerel{\preblob}{\blob}$ to scale the summation to match the size of the acted-upon term:

$$\sum_{i=0}^{3} \frac{\dfrac{x^3}{z+r^3}}{\dfrac{y}{x^2}}$$

If we wish constrain the width of the summation to 3ex, we employ $\scalerel[3ex]{\preblob}{\blob}$

$$\sum_{i=0}^{3} \frac{\dfrac{x^3}{z+r^3}}{\dfrac{y}{x^2}}$$

Of course, if the manipulated object contains text symbols (as in this case), a width constraint will change their aspect ratio, which is generally not desirable.

Now let's say you wanted to introduce notation to bound mathematical expressions by triangles. After defining \blob as before, you could use $\scalerel[3ex]{\triangleleft}{\blob}$ \scalerel*[3ex]{\triangleright}{\blob}$

$$\left\langle \frac{\dfrac{x^3}{z+r^3}}{\dfrac{y}{x^2}} \right\rangle$$

Here, the second call to **\scalerel** was with the star (∗) option, indicating that **\blob** should not be printed out following the right-hand delimiter. A less-tall expression would appear in those same delimiters as

$$\vartriangleleft Q \vartriangleright$$

Because the width limit had not been reached, no horizontal compression of the object was required. Note: see the **\scaleleftright** and **\stretchleftright** commands for a more direct way to employ these sort of transformations.

2.2   The \stretchrel Command

In its most simple application **\stretchrel** can be used in the fashion of a **\left\{** in math mode. While there is no reason to replace that more efficient usage supplied by LaTeX, it is nonetheless instructional to see a comparison of

$\left\{\blob\right.$ and $\stretchrel[550]{\{}{\blob}$, to see how the aspect-ratio limiting option can be employed to avoid an overly stretched (narrow) manipulation:

$$\left\{\begin{array}{c}\dfrac{x^3}{z+r^3}\\[1ex]\dfrac{y}{x^2}\end{array}\right. \qquad \left\{\begin{array}{c}\dfrac{x^3}{z+r^3}\\[1ex]\dfrac{y}{x^2}\end{array}\right.$$

The expression on the right uses a standard { character, which has been vertically stretched, but limited to an aspect ratio no narrower than 5.5. To use symbols for which the \left nomenclature will not work, \stretchrel provides a viable alternative, as shown in this stick-figure example:

```
3m $\stretchrel[600]{\updownarrow\,}
{\fbox{\rule[-1.8em]{0ex}{4em}Work Table}}$
```

3m $\updownarrow$ $\boxed{\text{Work Table}}$
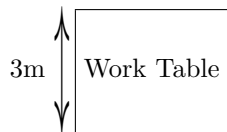
or in this use, $\stretchrel[350]{\int}{\blob} dx$, of the stretched integral sign (aspect limited to 3.5):

$$\int \dfrac{\dfrac{x^3}{z+r^3}}{\dfrac{y}{x^2}}\,dx$$

2.3 Baseline Shifts of *Relative* Scales and Stretches

It was mentioned that when \scalerel and \stretchrel are employed, the manipulated object gets vertically shifted to match the extent of the reference object. To see how this works, we provide the following example, that employs some inline math and scaling.

The baseline is the line that runs under this text $\boxed{\dot\imath \; \dfrac{\dfrac{x^3}{z+r^3}}{\dfrac{y}{x^2}}}\dot\imath$
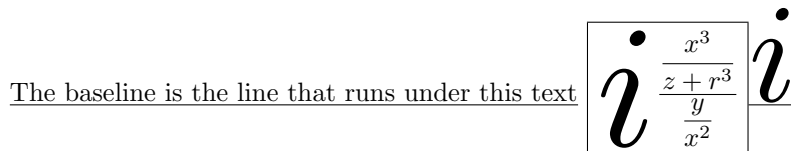
The first case of the large letter "i" was generated with \scalerel, and so the baseline of the "i", normally not a descending letter, was dropped to conform to the descended reference equation. All of this was set in an \fbox to show the extent of the objects. For comparison, an "i", scaled to 35pt, was then placed

using a \scaleto command, which does not change the baseline of the original object.

2.4 The \scaleto and \stretchto Commands

The \scaleto and \stretchto commands are comparable to the \scalerel and \stretchrel commands, except they do not accept a reference object as their second mandatory argument. Rather, they take an absolute height (*i.e.*, height + depth), specified with units. These commands use the same optional arguments as \scalerel and \stretchrel to constrain the width and/or the aspect ratio, respectively, of the manipulated object.

As was mentioned in the prior section, a difference between these and the *relative* scale/stretch commands is that an object's baseline is not altered with the use of \scaleto and \stretchto. Because there is no reference object employed with these commands, there is no need for starred (*) versions of these commands, which would otherwise suppress the printing of the reference object.

Examples follow the established pattern established in prior sections. First, we scale the equation blob used as an example in this documentation to a vertical extent of 80pt, preserving the original aspect ratio,

$\scaleto{\blob}{80pt}$
$$\frac{\dfrac{x^3}{z+r^3}}{\dfrac{y}{x^2}}$$

Below, we stretch the Greek letter, capital phi, to 60pt, but constrain the aspect ratio to no less than 1.75.

$$\Phi$$

$\stretchto[175]{\Phi}{60pt}$

2.5 The \scaleleftright and \stretchleftright Commands

These commands are composites of several `\scalerel` and `\stretchrel` commands, intended to provide the functionality of the `\left ...\right` syntax of mathmode, using symbols for which the `\left ...\right` syntax does not work. It uses the same optional arguments as `\scalerel` and `\stretchrel` in order to limit the width or aspect ratio of the respective manipulated objects. If one of the symbols on the left or right end of the enclosed reference block is to be blank, these commands support the use of the period argument `{.}` as a blank placeholder, in a fashion analogous to the `\left ...\right` syntax of mathmode.

$\scaleleftright[3ex]{\prod}{\blob}{\coprod}$

$$\prod \frac{\frac{x^3}{z+r^3}}{\frac{y}{x^2}} \coprod$$

$\stretchleftright[450]{.}{\blob}{\in}$

$$\frac{\frac{x^3}{z+r^3}}{\frac{y}{x^2}} \in$$

2.6 The \hstretch, \vstretch, and \scaleobj Commands

These commands simply perform horizontal or vertical stretching, respectively. They use a scale factor, rather than an absolute length. If the stretch factor is greater than 1, the stretched length is larger than the original. If less than one, it is compressed with respect to the original.

$\otimes\hstretch{3}{\otimes}\hstretch{0.5}{\otimes}$

$\otimes\vstretch{3}{\otimes}\vstretch{0.5}{\otimes}$

The `\scaleobj` command performs scaling in the fashion of a `\scalebox` command, except that its argument is processed in math mode of the current math display style, by default. Thus, the difference between the following two invocations:

```
\(\scaleobj{2}{\sum_{i=0}^{n}}\)
\[\scaleobj{2}{\sum_{i=0}^{n}}\]
```

is given as $\displaystyle\sum_{i=0}^{n}$ versus

$$\sum_{i=0}^{n}$$

2.7 The \ThisStyle, \SavedStyle, \LMex, and \LMpt Commands

These are very powerful macros that provide the ability to save the current math style and to import it to various places (for example, inside boxes and other macros) where the math style would otherwise be lost. These commands are part of this package because the various scalerel macros employ them to automatically import the current math style into their arguments.

The easiest way to show how these macros are used is to provide a working example, such as the one given below.

```
\documentclass{article}
\usepackage{stackengine}
\usepackage{graphicx}
\usepackage{scalerel}
\stackMath
\def\ccdot{\scalebox{1.15}{$\SavedStyle\cdot$}}
\def\altdiv{\mathbin{\ThisStyle{%
  \stackunder[-.6\LMex]{%
    \stackon[-.45\LMex]{\SavedStyle\sim}{\ccdot}%
  }{\ccdot}}}}
\begin{document}
$A \altdiv B ~~A \div B$\par
$\scriptstyle A \altdiv B ~~A \div B$\par
$\scriptscriptstyle A \altdiv B ~~A \div B$\par
\end{document}
```

The output of this example shows a fabricated glyph (\altdiv) across the math styles, comparing it to \div:

$$A \stackrel{.}{\approx} B \quad A \div B$$

$$A\stackrel{.}{\approx}B \quad A\div B$$

$$A\stackrel{.}{\approx}B \quad A\div B$$

The difficulties that must be overcome are several: the composite glyph is created by way of a stack, which processes its arguments in either text mode or `\textstyle` math mode; and the stacking distances between the components of the composite glyph need to vary with the mathstyle. To overcome the first issue, the `\altdiv` macro is wrapped in a `\ThisStyle{...}` wrapper. Whenever one of the glyph components (*e.g.*, `\sim` or `\cdot`) is employed within the stackengine where the mathstyle is otherwise lost, it is prepended with a `\SavedStyle` to reinstate the current math style. Thus, the glyph components will present in the proper mathstyle. Likewise, the vertical stacking gaps between the glyph components, given as optional arguments to the stacking commands, are here specified in multiples of `\LMex`. In this way, the stacking gaps will shrink with the smaller math styles, keeping the spacings on the same relative scale as the size of the smaller glyphs.

3 Real World Application

3.1 White Curly Brackets

Ref: http://tex.stackexchange.com/questions/100966/defining- scalable-white-curly-brackets-and-and

A user defined two new symbols made by combining the font glyphs of a brace and a vertical strut, calling them `\llbrace` and `\rrbrace`, to look as follows around an argument (in this case x):

$$\{\!|x|\!\}$$

But they found that when represented in reduced size fonts, the symbol glyphs of the brace and the strut separated (note: `\tiny` font has been scaled up to match prior figure's height):

$$\{\!| \; x \; |\!\}$$
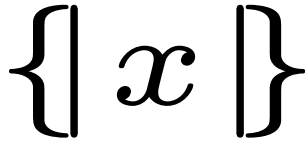
This happens because font designers don't merely scale their fonts when shrunk down in size, but they do things (like thickening up lines and changing aspect ratios) in an attempt to retain legibility. In this case, the font's auto-adjustment at smaller scale caused this composite representation to change in unacceptable ways.

The scalerel solution was to use the image from the normal-size representation and to shrink it down as needed. So, in this case, a normal-sized copy of the braces were stored in boxes called `\lXbrace` and `\rXbrace`, resepctively. Then the following terms were defined:

```
\def\lxbrace{\scalerel*{\usebox{\lXbrace}}{\llbrace}}
\def\rxbrace{\scalerel*{\usebox{\rXbrace}}{\rrbrace}}
```

Now, in lieu of using `\llbrace` directly, the use of `\lxbrace` would take a copy of the normal-sized `\lXbrace` and scale it down to the vertical height of the malformed tiny `\llbrace`, and substitute for it. Here, we show the result at both normal size and tiny size:

$$\{\!| \; x \; |\!\} \quad \{\!| x |\!\}$$
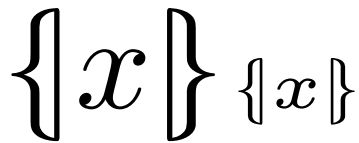
Note that the brace is the properly formed scale of the normal-sized brace in both cases, even as the x adjusts to the fontsize reduction.

If one wanted the default braces narrower (to match the look of the comparable symbols in the literature), one could use `\hstretch` in the definitions to achieve that quickly as

```
\def\lxbrace{%
    \hstretch{0.6}{\scalerel*{\usebox{\lXbrace}}{\llbrace}}}
\def\rxbrace{%
    \hstretch{0.6}{\scalerel*{\usebox{\rXbrace}}{\rrbrace}}}
```

to give the following

$$\left\{\!\!\left\{ x \right\}\!\!\right\} \{\!\!\{x\}\!\!\}$$

Likewise, the `\left` and `\right` features of equation mode cannot be used with the white braces,

$$\{\!\!\left[\!\!\left[\left\{ \left[(a) \cdot bc^{2^3} \right]^4 \right\}^5 \right]\!\!\right]\!\!\}$$

but scalerel can fix that, too:

$$\left\{\!\!\left[\!\!\left[\left\{ \left[(a) \cdot bc^{2^3} \right]^4 \right\}^5 \right]\!\!\right]\!\!\right\}$$

with

```
\(\scaleleftright[1.5ex]{\lxbrace}{\core}{\rxbrace}\)
```

where `\core` is the core inner equation to which the white braces need to be scaled. In this case, the scaled white braces were width-limited to 1.5ex.

3.2   Extra-Wide Oversymbols

Ref:
http://tex.stackexchange.com/questions/100574/really-wide-hat-symbol/101136#101136

There are many oversymbols defined to stand over a single-letter variable. There are a smaller number of extra-wide oversymbols to extend over small groups of symbols. Occasionally, however, a need arises for an oversymbol to stretch over a larger group of symbols.

With these definitions,

```
\usepackage{scalerel,stackengine}

\newcommand\reallywidehat[1]{\ensurestackMath{%
\savestack{\tmpbox}{\stretchto{%
  \scaleto{%
    \scalerel*[\widthof{\ensuremath{#1}}]{\kern-.6pt\bigwedge\kern-.6pt}%
    {\rule[-\textheight/2]{1ex}{\textheight}}}%WIDTH-LIMITED BIG WEDGE
  }{\textheight}%
```

```
}{0.5ex}}%
\stackon[1pt]{#1}{\tmpbox}%
}}

\newcommand\reallywideparen[1]{\ensurestackMath{%
\savestack{\tmpbox}{\stretchto{%
  \scaleto{%
    \scalerel*[\widthof{\ensuremath{#1}}]{\kern-.6pt\frown\kern-.6pt}%
    {\rule[-\textheight/2]{1ex}{\textheight}}}%WIDTH-LIMITED BIG FROWN
  }{\textheight}%
}{0.9ex}}%
\stackon[-.5pt]{#1}{\tmpbox}%
}}
```

arbitrarily wide oversymbols may be constructed:

$$\widehat{POQ} + \widehat{QOR} + \widehat{ROP} = 2\pi r$$

$$\overparen{zbcdefghijklm} \neq \widehat{zbcdefghijklm}$$

4 Future Development

It would be relatively straightforward to extend this approach to horizontal scaling problems. However, it is not exactly clear to the author what format is best suited for user needs. If he gets feedback in that regard, it will inform him how best to proceed.

5 Code Listing

```
\ProvidesPackage{scalerel}
[2016/12/29 v1.8
 Routines for constrained scaling and stretching of objects,
 relative to a reference object or in absolute terms]
%
% This work may be distributed and/or modified under the
% conditions of the LaTeX Project Public License, either version 1.3
% of this license or (at your option) any later version.
% The latest version of this license is in
%   http://www.latex-project.org/lppl.txt
% and version 1.3c or later is part of all distributions of LaTeX
% version 2005/12/01 or later.
%
```

```
% This work has the LPPL maintenance status 'maintained'.
%
% The Current Maintainer of this work is Steven B. Segletes.

% V1.01-Correct typos in documentation
% V1.1 -Corrected usepackage dependencies; Significant doc rewrite.
% V1.2 -Added \scaleleftright and \stretchleftright; revised docs.
% V1.3 -Corrected missing % in stretch routines, added \hstretch and
%        \vstretch commands; fixed version number in .sty file
% V1.4 -Auto-detect mathmode and style; use the proper math display
%        style; Allow \ignoremathstyle and \discernmathstyle to
%        revert to former approach in rare cases and return to new
%        approach.
%       -Added \Isnextbyte as a supplemental routine
%       -Added \scaleobj (like \scalebox, but defaults to math mode
%        using current display style).
% V1.5 -Eliminated mathstyle package because of incompatibilites with
%        many other packages.  Implemented \mathchoice approach instead.
% V1.6 -Made available \LMex and \LMpt lengths (Local Math ex and
%        Local Math pt), usable inside \ThisStyle arguments (1ex/pt in
%        \textstyle and \displaystyle, 0.7ex/pt in \scriptstyle, and
%        0.5ex/pt in \scriptscriptstyle).
%       -Revised/shortened/improved \Isnextbyte.
%       -Replaced ifthen package calls with etoolbox calls.
% V1.7 -Note that the V1.5 implementation with \mathchoice seemed to
%        have made the use of \ignoremathstyle and \discernmathstyle
%        vestigial.  Thus, \ignoremathstyle was revised to be used
%        for the purpose of streamlining package efficiency when
%        only textstyle (or displaystyle with \ignoremathstyle[D])
%        processing is required.
%       -Fixed use of dot {.} as null argument in \scaleleftright{}{}{}
%        and \stretchleftright{}{}{}, which had been causing overflow.
%       -Made \hstretch and \vstretch routines more efficient, using
%        optional arguments of \scalebox, instead of \scalerel.
% V1.8 -Fixed divide by 0 bugs that would arise when object to be
%        scaled fell completely below the baseline (zero height).
%        Required revisions to \scalerelplain, \stretchrelplain,
%        \scaleto, and \stretchto.
%       -Fixed aspect-ratio bug, that calculated aspect for stretched object
%        based only upon portion above baseline

\usepackage{calc}
\usepackage{graphicx}
\usepackage{etoolbox}
\global\newlength\thesrwidth
\global\newlength\thesrheight
\global\newlength\srblobheight
\global\newlength\srblobdepth
\global\newlength\mnxsrwidth
\newlength\LMex
```

14

```
\newlength\LMpt
\def\scriptstyleScaleFactor{.7}
\def\scriptscriptstyleScaleFactor{.5}

\newcommand\scalerel{\@ifstar{\scalerelplain}{\scalerelplus}}

\newcommand\scalerelplain[3][99in]{\ThisStyle{%
  \setbox0\hbox{\@obj{#3}}%
  \setlength\srblobheight{\ht0+\dp0}%
  \setlength\srblobdepth{\dp0}%
  \setbox2\hbox{\@obj{#2}}%
  \setbox0\hbox{\raisebox{\dp2}{\@obj{#2}}}%
  \setlength\thesrwidth{\wd0*\ratio{\srblobheight}{\ht0}}%
  \setlength\thesrheight{\ht0*\ratio{\srblobheight}{\ht0}}%
  \setlength\mnxsrwidth{#1}%
  \ifdim\thesrwidth>\mnxsrwidth\setlength\thesrwidth{\mnxsrwidth}\fi%
  \raisebox{-\srblobdepth}{\resizebox{\thesrwidth}{\thesrheight}{\copy0}}%
  }%
}

\newcommand\scalerelplus[3][99in]{\scalerelplain[#1]{#2}{#3}#3}

\newcommand\stretchrel{\@ifstar{\stretchrelplain}{\stretchrelplus}}

\newcommand\stretchrelplain[3][10000]{\ThisStyle{%
  \setbox0\hbox{\@obj{#3}}%
  \setlength\srblobheight{\ht0+\dp0}%
  \setlength\srblobdepth{\dp0}%
  \setbox2\hbox{\@obj{#2}}%
  \setbox0\hbox{\raisebox{\dp2}{\@obj{#2}}}%
  \setlength\thesrwidth{\wd0}%
  \setlength\thesrheight{\ht0*\ratio{\srblobheight}{\ht0}}%
  \setlength\mnxsrwidth{\thesrheight*100/#1}%
  \ifdim\thesrwidth<\mnxsrwidth\setlength\thesrwidth{\mnxsrwidth}\fi%
  \raisebox{-\srblobdepth}{\resizebox{\thesrwidth}{\thesrheight}{\copy0}}%
  }%
}

\newcommand\stretchrelplus[3][10000]{\stretchrelplain[#1]{#2}{#3}#3}

\newcommand\scaleto[3][99in]{\ThisStyle{%
  \setlength\srblobheight{#3}%
  \setbox2\hbox{\@obj{#2}}%
  \setbox0\hbox{\raisebox{\dp2}{\@obj{#2}}}%
  \setlength\thesrwidth{\wd0*\ratio{\srblobheight}{\ht0}}%
  \setlength\thesrheight{\ht0*\ratio{\srblobheight}{\ht0}}%
  \setlength\mnxsrwidth{#1}%
  \ifdim\thesrwidth>\mnxsrwidth\setlength\thesrwidth{\mnxsrwidth}\fi%
  \setlength\srblobdepth{\dp2*\ratio{\srblobheight}{\ht0}}%
  \raisebox{-\srblobdepth}{\resizebox{\thesrwidth}{\thesrheight}{\copy0}}%
```

```
    }%
}

\newcommand\stretchto[3][10000]{\ThisStyle{%
  \setlength\srblobheight{#3}%
  \setbox2\hbox{\@obj{#2}}%
  \setbox0\hbox{\raisebox{\dp2}{\@obj{#2}}}%
  \setlength\thesrwidth{\wd0}%
  \setlength\thesrheight{\ht0*\ratio{\srblobheight}{\ht0}}%
  \setlength\mnxsrwidth{\thesrheight*100/#1}%
  \ifdim\thesrwidth<\mnxsrwidth\setlength\thesrwidth{\mnxsrwidth}\fi%
  \setlength\srblobdepth{\dp2*\ratio{\srblobheight}{\ht0}}%
  \raisebox{-\srblobdepth}{\resizebox{\thesrwidth}{\thesrheight}{\copy0}}%
  }%
}

\newcommand\scaleleftright[4][99in]{%
  \ifx.#2#3\else\scalerel[#1]{#2}{#3}\fi%
  \ifx.#4\else\scalerel*[#1]{#4}{#3}\fi%
}
\newcommand\stretchleftright[4][10000]{%
  \ifx.#2#3\else\stretchrel[#1]{#2}{#3}\fi%
  \ifx.#4\else\stretchrel*[#1]{#4}{#3}\fi%
}

\newcommand\hstretch[2]{%
  \ThisStyle{\scalebox{#1}[1]{\@obj{#2}}}}

\newcommand\vstretch[2]{%
  \ThisStyle{\scalebox{1}[#1]{\@obj{#2}}}}

\newcommand\scaleobj[2]{%
  \ThisStyle{\scalebox{#1}{\@obj{#2}}}}

% DETERMINE IF THE SCALEREL ROUTINE WAS CALLED IN MATH MODE.
% IF SO, USE THE PROPER MATH DISPLAY STYLE FOR THE CALL,
% *UNLESS* THE FIRST CHARACTER OF THE ARGUMENT IS A $ SIGN,
% SIGNIFYING A DESIRED ESCAPE FROM MATH MODE FOR THE ARGUMENT.
\newcommand\@obj[1]{%
  \if T\@mmode%
    \Isnextbyte[q]{$}{#1}%
    \if F\theresult%
      $\SavedStyle#1$%
    \else%
      $#1$%
    \fi%
  \else%
    $#1$%
  \fi%
}
```

```
\def\@mstyleD{\displaystyle}
\def\@mstyleT{\textstyle}
\def\@mstyleS{\scriptstyle}
\def\@mstyles{\scriptscriptstyle}
%
\def\SavedStyle{\csname @mstyle\m@switch\endcsname}
%
\newcommand\ThisStyle[1]{%
  \ifmmode%
    \def\@mmode{T}\mathchoice%
      {\edef\m@switch{D}\LMex=1ex\relax\LMpt=1pt\relax#1}%
      {\edef\m@switch{T}\LMex=1ex\relax\LMpt=1pt\relax#1}%
      {\edef\m@switch{S}\LMex=\scriptstyleScaleFactor ex\relax%
                        \LMpt=\scriptstyleScaleFactor pt\relax#1}%
      {\edef\m@switch{s}\LMex=\scriptscriptstyleScaleFactor ex\relax%
                        \LMpt=\scriptscriptstyleScaleFactor pt\relax#1}%
  \else%
    \def\@mmode{F}%
    \edef\m@switch{T}\LMex=1ex\relax\LMpt=1pt\relax#1%
  \fi%
}
%

\let\sv@ThisStyle\ThisStyle

\newcommand\discernmathstyle{\let\ThisStyle\sv@ThisStyle}

% \ignoremathstyle WOULD BE USED TO SPEED UP DEEPLY NESTED
% ROUTINES OF scalerel PACKAGE, WHEN LIMITED TO textstyle MATH
\newcommand\ignoremathstyle[1][T]{%
  \renewcommand\ThisStyle[1]{%
    \edef\m@switch{#1}\LMex=1ex\relax\LMpt=1pt\relax%
    \ifmmode\def\@mmode{T}\else\def\@mmode{F}\fi##1}%
}

% THIS CAN REPLACE \isnextbyte IN STRINGSTRINGS PACKAGE
% BECAUSE IT WORKS ON CHARS THAT CAN'T BE \edef'ED (\dag, ETC.)
\newcommand\Isnextbyte[3][v]{%
  \def\SR@Letter@A{#2}%
  \SR@nextbyte{\SR@Letter@B}#3\relax\relax\relax%
  \ifdefstrequal{\SR@Letter@A}{\SR@Letter@B}%
    {\edef\theresult{T}}{\edef\theresult{F}}%
  \if q#1\else\theresult\fi%
}
\def\SR@nextbyte#1#2#3\relax{\def#1{#2}}

\endinput
```