Providing color in math

Frank Mittelbach
June 1, 2024

1 Introduction

It is possible to use the \color command in math formulas but doing that has a number of restrictions and often leads to rather unreadable input. For example, to color the summation sign in red in the following formula

\[ X = \sum_{i=1}^{n} x_i \]

A total of four color commands and a number of seemingly unnecessary extra braces in the sub and superscript are needed:

\[
\begin{align*}
X &= \color{red} \sum_{\color{black} i=1}^{\color{black} n} \color{black} x_i \\
&= \{1\} \quad (\text{error})
\end{align*}
\]

While this is ugly but at least works, other things are simply impossible, e.g.,

\[
\begin{aligned}
\{1\} \\
\{2\}
\end{aligned}
\]

are not achievable at all because of the fact that \left and \right form a group. Thus, a \color command immediately before the \right has no effect and \} remains black. Putting the color change before the \left and then resetting the color inside would work if you want to color both braces, but the above combination or the attempt to color only the left brace fails always.

Using \textcolor in formulas appears to work on first glance, but it produces spacing problems that are not easy to overcome either, because it generates a single \mathord and no longer reflects its input, thus \[ a = b \textcolor{red}{\neq} c \] produces

\[ a = b \neq c \]

This is a case one could mend by wrapping the \textcolor and its arguments in a \mathrel, but even when that is possible, the resulting formula source is difficult to maintain and in other situations the solutions are even more complicated or there is no solution at all.

\*This file has version v1.0c dated 2022/07/25, © L\TeX\ Project.
We therefore offer now a dedicated math coloring command named \texttt{mathcolor}.

\begin{verbatim}
\mathcolor [⟨model⟩] {⟨color-spec⟩} {⟨math material⟩}
\end{verbatim}

It has the same arguments as \texttt{textcolor} but is intended for use in formulas. The command does not generate a group and the ⟨math material⟩ retains its math atom states and it correctly handles sub and superscripts that follow.

The command can also be used to color a single opening or closing symbol, e.g., the correct input to our earlier example is

\[
\left\{ \frac{1}{2} \mathcolor{red}{\right}\}
\]

which is how it was produced.

If you attempt to color large operators that use explicit \texttt{\limits}, \texttt{\nolimits}, or \texttt{\displaylimits}, then these limit controls need to immediately follow the operator. However, \texttt{mathcolor} is written in a way that it is possible write

\[
\left[ \mathcolor{red}{\int}\limits_0^1 \textsf{ or } \mathcolor{red}{\int\limits}_0^1 \right]
\]

and achieve the same effect:

\[
\int_0^1 \quad \text{or} \quad \int_0^1
\]

\section{The Implementation}

The code is called inside of the \texttt{color} or \texttt{xcolor} packages, so \$@\$ is already a letter, but the coding here is done in the L3 programming layer, so we need to activate that. But first we check if this file was loaded before, e.g., when both \texttt{color} and \texttt{xcolor} are in the preamble and in that case we stop immediately.

\begin{verbatim}
\ifcsname mathcolor\endcsname \endinput \fi
\end{verbatim}

\g__mathcolor_seq  We need to keep our own stack of color commands, so we allocate a sequence for that.

\begin{verbatim}
\seq_new:N \g__mathcolor_seq
\end{verbatim}

(End of definition for \texttt{\g__mathcolor_seq}.)

\texttt{mathcolor}  The document-level command which expects a color (if unnamed then the optional argument is the model) and colors the content of the second argument (like \texttt{textcolor}, but without spacing problems in math).

\begin{verbatim}
\DeclareDocumentCommand \mathcolor { o m m }
\end{verbatim}

The \texttt{mathcolor} is only supported in math mode because in text mode it has problems scanning away a space after it, for example. We therefore raise an error if it executes anywhere else. The \texttt{\LaTeX} error command is a bit strangely named, because in the kernel it is only used for math alphabets, but the message it gives is fine.

\begin{verbatim}
\mode_if_math:F { \non@alpherr \mathcolor\space } \}
\end{verbatim}

2
First real action is to save the current color value on a stack (needed if the command is
nested or contains some further color changes with \color inside).

\seq_gpush: No \g__mathcolor_seq \current@color

Then we switch to the new color, but we do not want to reset the color after the group
(which is done by \color using \aftergroup\reset@color). The best solution here
would be if the color packages would provide a command doing just the color switch-
ing, but for now we simply undo that part by pushing \use_none:n which gobbles the
\reset@color added by \color with \aftergroup.

\group_insert_after:N \use_none:n

Switching the color is also slightly suboptimal, because depending on whether or not we
have a \langle \langle \text{model} \rangle \rangle argument, we have to call \color with or without the optional argument.
But going low-level here is not an option as we need to support different color packages
and their internals are not identical.

\IfValueTF {#1} { \color[#1]{#2} } { \color{#2} }

Then comes the math material we want to see colored:

#3

After that we need to reset the color ourselves (without a group that does it for us), i.e.,
popping the saved color from our stack, but there are some twists to that so we do this
in a separate command (which in fact needs to be called several times, so inlining the
code wouldn’t be possible.

\__mathcolor_scan_for_scripts:w

(End of definition for \mathcolor. This function is documented on page 2.)

\__mathcolor_scan_for_scripts:w

The complication when changing the color back is due to the fact that the \mathcolor
may be followed by ^ or _ or the hidden superscript ‘ and its argument may end in a
\mathop in which case the sub and superscripts may be attached as \limits instead of
after the material. All cases need separate treatment. And we need to watch out for an
upcoming & and avoid prematurely triggering the end of an alignment cell.

\cs_new_protected:Npn \__mathcolor_scan_for_scripts:w

We need to look at what follows \mathcolor, and we need to do that ignoring (and dropping)
any spaces and \relax as \TeX would do in normal math processing (for example
before a subscript token). We do this with expansion so that hidden sub or superscripts
in macros are still found as long as the macros are expandable.

\peek_remove_filler:n

To avoid problems hitting on & we start with a \group_align_safe_begin:. That has to be ended with \group_align_safe_end: when the danger (aka scanning) is over, which, due to the branching below, happens at four different points, i.e., when the \mathcolor is

1. followed by a “normal” token;
2. followed by a braced sub/superscript;
3. followed by an unbraced sub/superscript;
4. followed by one of the \limits primitives.
In each case we have to end the align safe group and we mark the points below in the code for easy reference.

\group_align_safe_begin:

We first parse for \c_math_subscript_token or \c_math_superscript_token. After \peek_remove_filler:n is done, it sets \l_peek_token equal to the next non-filler token, so we can avoid unnecessary work and just compare that. If either of the tokens is found, call \__mathcolor_handle_scripts:Nw:

\token_case_catcode:NnTF \l_peek_token
{ \c_math_subscript_token {} \c_math_superscript_token {} }
{ \__mathcolor_handle_scripts:Nw }

Otherwise we check if it was any of the limit operation primitives. If that is the case, e.g., if we have a situation such as

\mathcolor{red}{\int}\limits_1

we have to move it directly after the \int to ensure there is no color reset between the operator and the \limits command.

\token_case_meaning:NnTF \l_peek_token
{ \limits { } \nolimits { } \displaylimits { } }

Once that is done, we have to get rid of the token we peeked at and then restart scanning for sub or superscripts. Given that \__mathcolor_scan_for_scripts:w expands while scanning the simplest solution is to add \use_none:n in front of the peeked at token.

Here we end the align safe group and \__mathcolor_scan_for_scripts:w will start a new one.

{ \group_align_safe_end: % case 4 \__mathcolor_scan_for_scripts:w \use_none:n }

If it was not one of these we look for a ‘ ’ and if found remove it and replace it by its expansion. The reason we have to do this (and not rely on the earlier peeking to expand for us is the fact that ‘ ’ is only “math active” and that doesn’t expand under \expanded or \expandafter.

\token_if_eq_meaning:NNTF \l_peek_token ‘
{ \__mathcolor_handle_scripts:Nw ^ \c_group_begin_token \exp_after:wN \prim@s \use_none:n }

If it is anything else we finish off which means we reset the color (because we prevented that before to happen automatically after the next group) and pop the color stack setting \current@color.
The tricky part of handling sub and superscripts is that we have to reset color to the one that is on the stack but reset it back to what it was before to allow for cases like
\[ \textcolor{red}{a+\sum}_{i=1}^n \]
Here, \TeX{} constructs a \verb|vbox| stacking subscript, summation sign, and superscript. So technically the superscript comes first and the \verb|\sum| that should get colored red is the middle.

53 \texttt{\textbackslash cs\_new\_protected:Npn \texttt{\_\_mathcolor\_handle\_scripts:Nw #1}}

The argument is either \^{} or \_{}, so we execute it and explicitly open two { groups. We need two because color resets are always done after a group, so the first group is for script material (in case it was just something like \_\_i) and the second is needed for the color reset to keep it within the super or subscript. If that reset would happen after it then the color \verb|\special| would interfere with \TeX{} math spacing.

55 \texttt{#1 \textbackslash c\_group\_begin\_token \textbackslash c\_group\_begin\_token}

Now it is time to change the color back to the one on the stack.

56 \texttt{\textbackslash seq\_get:NN \texttt{\_\_mathcolor\_seq \texttt{\_\_current\_color}}}

\verb|\set@color| adds \verb|\aftergroup\reset@color|. We now add bit more so that the code executed after the current (inner) group looks like this:

\verb|\reset@color \textbackslash \_\_mathcolor\_scan\_for\_scripts:w|

The \verb|\_\_mathcolor\_scan\_for\_scripts:w| then retakes control and initiates parsing for another sub or superscript.

58 \texttt{\textbackslash group\_insert\_after:N \texttt{\_\_\_\_mathcolor\_scan\_for\_scripts:w}}

Before we give control to \TeX{} to process the sub or superscript some final adjustment is necessary: if the input was \^{}\ldots{} then we have one { too many, because we already supplied the outer one already. In that case we drop it. Otherwise we have an unbraced single token sub or superscript which means we are missing a closing } at the end and need to account for that: this is done in false branch by \verb|\use_ii_i:n|.

After scanning for a brace all scanning is done, so here are the other two points where we have to end the align safe group (in the true and false case).

60 \texttt{\textbackslash peek\_remove\_filler:n}

62 \texttt{\textbackslash token\_if\_eq\_meaning:NNTF \texttt{l\_peek\_token \texttt{\_\_\_c\_group\_begin\_token}}

{
\group_align_safe_end: \peek_catcode_remove:NT \c_group_begin_token \{
\exp_after:wN \group_align_safe_end: \use_i_i:nn \c_group_end_token
\}
\}
\}
(End of definition for \__mathcolor_handle_scripts:Nw.)
\ExplSyntaxOff
\ExplSyntaxOn
Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

\aftergroup .................. 3, 5
\c_group_end_token ........... 58, 69
\group_insert_after:N ....... 10, 58, 59
\color ........................ 1, 3, 11
\cs_new_protected:Npn ........ 15, 53
\DeclareDocumentCommand ........ 6
\displaylimits ................ 2, 31
\endcsname .................. 2
\endinput ................... 2
\exp_after:wN ................ 42, 68
\expandafter ................ 4
\group_insert_after:N ....... 3, 19
\group_align_safe_end: ........ 3, 19
\group_align_safe_begin: ...... 3, 19
\group_insert_after:N .......... 3, 45, 64, 68
\c_group_begin_token ..... 41, 55, 62, 65
\mathop ........................ 3
\mathrel ........................ 1
\mathord ........................ 1
\mathord ......................... 1
\mode_if_math:TF ............. 8
\nolimits ........................ 2, 30
\peek_catcode_remove:NTF ........ 65
\nolimits ........................ 2, 30
\peek_catcode_remove:NTF ........ 65
\nolimits ........................ 2, 30
\peek_catcode_remove:NTF ........ 65
\peek_remove_filler:n 4, 17, 60
\lpeek_token 4, 20, 27, 38, 62
\relax 3
\right 1
\seq commands:
\seq_get:NN 56
\seq_gpop:NN 47
\seq_gpush:Nn 9
\seq_new:N 5
\space 8
\special 5
\sum 5
\token commands:
\c_math_subscript_token 4, 22
\c_math_superscript_token 4, 23
\token_case_catcode:NnTF 20
\token_case_meaning:NnTF 27
\token_if_eq_meaning:NNTF 38, 62
\use commands:
\use_ii_i:nn 5, 69
\use_none:n 3, 4, 10, 35, 42
\vbox 5