# FreeForm: Flexibly Augmenting Formulas with Synchronized Markup and Graphical Edits


Jeffrey Tao
jefftao@seas.upenn.edu
University of Pennsylvania
Philadelphia, PA, USA

Litao Yan
ltyan@seas.upenn.edu
University of Pennsylvania
Philadelphia, PA, USA

Jessica Shi
jwshi@seas.upenn.edu
University of Pennsylvania
Philadelphia, PA, USA

Mia Ginsberg
gmia@seas.upenn.edu
University of Pennsylvania
Philadelphia, PA, USA

Andrew Head
head@seas.upenn.edu
University of Pennsylvania
Philadelphia, PA, USA

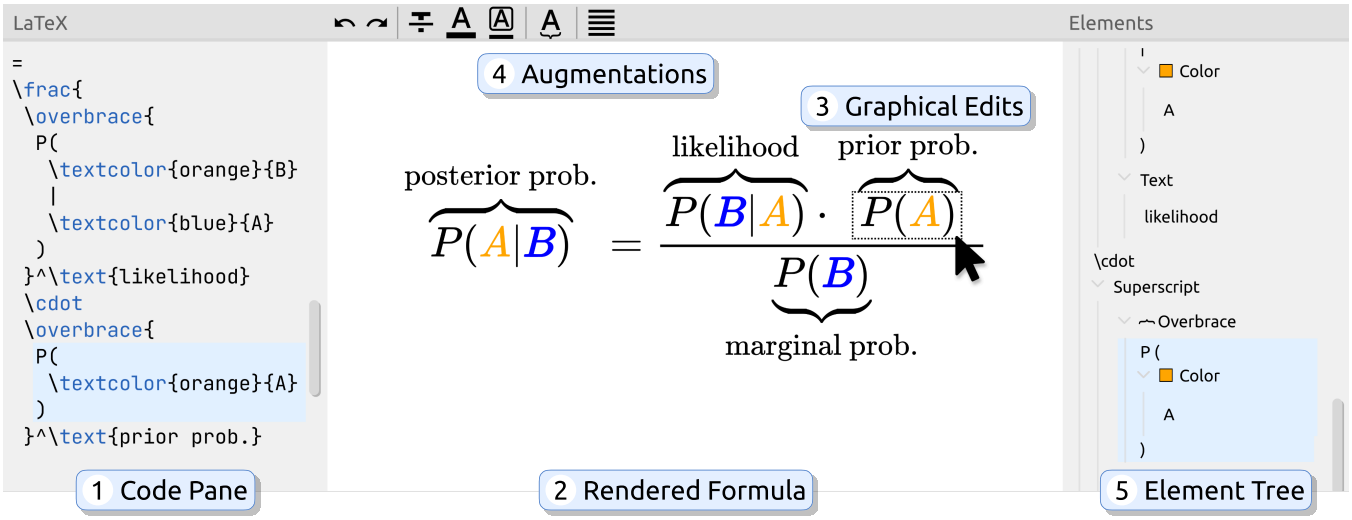




**Figure 1: An example of augmenting a formula with FreeForm.** Authors can quickly add and edit LaTeX-based formula code in the left pane (❶) and preview the rendered result in the center pane (❷). Graphical edits allow authors to select elements of the formula directly (❸) and apply augmentations such as colors and labels via a menu (❹). Edits are synchronized in both directions between the code and graphics editors, backed by a syntax-tree-based representation. The right pane shows this tree hierarchy and facilitates navigation and editing (❺).


## Abstract

Authors of typeset formulas augment those formulas to make them easier to understand. When they do so, they trade off between using markup tools like LaTeX and formula-unaware graphical editors. In this paper, we explore how editing tools could combine the best affordances of both kinds of tools. We develop FreeForm, a projectional editor wherein authors can augment formulas—with color, labels, spacing, and more—across multiple synchronized representations. Augmentations are created graphically using direct selections and compact menus. Those augmentations propagate to LaTeX markup, which can itself be edited and easily exported. In two lab studies, we observe the value of our editor versus baselines of a widely-used LaTeX document editor and a state-of-the-art formula augmentation tool. Finally, we make recommendations for the design of projectional markup augmentation editors.



## CCS Concepts

• **Human-centered computing** → **Interactive systems and tools**.

## Keywords

formulas, annotation, projectional editing, LaTeX



*CHI '25, April 26-May 1, 2025, Yokohama, Japan*




**ACM Reference Format:**
Jeffrey Tao, Litao Yan, Jessica Shi, Mia Ginsberg, and Andrew Head. 2025. FreeForm: Flexibly Augmenting Formulas with Synchronized Markup and Graphical Edits. In *CHI Conference on Human Factors in Computing Systems (CHI '25), April 26-May 1, 2025, Yokohama, Japan.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3706598.3714288

## 1 Introduction

Mathematical concepts are often presented using mathematical notation. The advantage of notation its ability to precisely convey rich semantics. The downside of notation is that it can be hard for readers to understand, particularly if the notation is unfamiliar or complex. Consider the formula in Figure 1 representing Bayes' theorem. To use this formula, a reader needs to understand its constituent terms (i.e., "$A$," "$B$," and "$P(\cdots)$") and map its larger expressions to concrete quantities (e.g., "$P(A)$" representing the prior probability of an event). Because formulas are rarely self-explanatory to the uninitiated, they often benefit from being augmented with descriptive elements, such as labels describing the meaning of an expression or colors linking an identifier to its description in the text.

Given the visual simplicity of such augmentations, one might expect that they are similarly simple to produce, but this can be far from the reality. Authors have found doing this work tedious and clunky with existing tools [30].

Among the two kinds of tools that authors have at their disposal to augment formulas, both have their downsides. On the one hand, authors can augment formulas in the markup languages they use to write the formulas, like LaTeX [42] and its macros for coloring and annotating. However, these markup languages can require cumbersome and error-prone editing, arising from the intermixing of annotation markup with the underlying formula. Participants in a study by Wu et al. [71] identified difficulty with debugging nested braces and locating markup to edit.

Alternatively, authors can use graphical editors such as PowerPoint or Illustrator, augmenting formulas using familiar affordances for adding shapes, text, and styles. The challenge of using these tools is that annotations are unmoored from the structure of the formula and must be redone whenever the formula changes. Authors must perform precision positioning and sizing operations that could be inferred from the coordinates of the augmented expressions.

In the HCI community, one solution to this problem is projectional editing [21], where users can access multiple representations of the same artifact, each with complementary affordances. In this work, we develop FreeForm as a projectional editor for notation augmentation. We define the key projections as markup (in this case, LaTeX), an annotatable render, and a structure hierarchy view. Augmentations are made easy to invoke, and projections are kept synchronized and co-present so that authors can shift between representations as is expedient to them.

In two studies, we evaluate the impact of these design choices on the authoring of augmentations. In a controlled study (Section 5), we compared FreeForm to a state-of-the-art DSL for augmentation (FFL [71]) and a standard LaTeX (Overleaf) baseline. Both FreeForm and FFL reduced task time for coloring and labeling tasks, FreeForm most of all, though FreeForm increased task time and difficulty on an additional formula alignment task. In a follow-up observation study (Section 6), we document design successes and gaps. Perhaps most significantly, we describe how participants interleaved editing activities across synchronized, co-present editors viewing the formula as LaTeX, graphics, and trees. Taken together, this paper contributes:

- FreeForm, a projectional editor that accelerates augmentation of formulas with synchronized affordances for augmenting formulas using markup and graphics editing.
- Evidence from two studies of how and when FreeForm helps authors perform augmentations.
- Recommendations for designing markup-based projectional editors, informed by observations from our two studies.

## 2 Related Work

While notation plays a central role in many scientific fields (e.g., as described in Grainger et al. [24]), it is also often difficult to understand. Studies have shown notation leading to worse comprehension of texts [54], greater cognitive load [44], and greater difficulty following along with proofs [16]. Sweller [61] provides an interpretation via cognitive load theory: to work with a formula, one must keep its many interrelated parts in mind all at once. Small graphical changes to spatial grouping and symbol design can influence perception of a formula's meaning, such as operator precedence and semantics [27, 41, 67]. Segmenting [4] and annotating [33] formulas have resulted in observed learning gains. There has been recent interest among HCI researchers in developing document formats that support formula augmentation [15, 29, 43]. The purpose of this paper is to develop better tools for authors to change ways that formulas are consumed.

### 2.1 Tooling For Augmenting Math

In this section, we review production and prototype tools that can help authors augment formulas.

*LaTeX.* LaTeX [56] is a widely used language for document typesetting and provides support for augmentations such as colors [6], labels [48], alignment, slashes, borders, and backgrounds. Community developed packages such as `annotate-equations` [36] provide yet more sophisticated and aesthetic augmentation options. LaTeX formulas are supported in many web-based authoring tools from Notion [51] to Jupyter Book [10] and MathOverflow [60]. However, a limitation of using LaTeX to augment formulas is the clutter and fussiness of expressing and editing augmentations as macros inline with the markup [30].

*Other language-based approaches.* To overcome the messiness of LaTeX-based augmentation specifications, Wu et al. [71] proposed *FFL*, a language where the augmentation markup is concise and separable from the formula LaTeX. FFL provides targeting rules so that augmentations can be applied to many instances of a formula across an entire document. We compare to FFL as a strong baseline in Section 5 and discuss tradeoffs in Section 7.

Other document authoring DSLs have been developed with support for augmenting formulas. These include Heartdown [45] and Nota [15], which let authors annotate symbols with definitions, link formulas to executable code, and link symbols to definitions in the text. They also include *manim* [49], a DSL for authoring math animations, which allows the construction of step-by-step builds of formulas and accompanying augmentations. We anticipate that interactions from FreeForm could transfer to helping authors augment augmentations with these DSLs as well.

*Structured editors.* Many tools [3, 14, 20, 32, 47, 50, 52, 53, 65] support structured editing of formulas. In general, these tools provide menu-based interactions for inserting symbols and composing them into structures, though some [3, 14, 20, 47, 50, 65] support augmentations to some degree. A few [14, 47, 50, 65] allow selection and manipulation of symbols in the rendered formula. A less common choice is to make the synced LaTeX continuously available (e.g., as in [3]). We lend evidence to support this design choice, and situate it in a design that additionally foregrounds augmentation operations and incorporates tree-aware graphical selection mechanisms and tree structure views.

*Graphical editors.* Tools like PowerPoint [13] and Adobe Illustrator [34] offer graphical editing affordances that authors sometimes use to augment formulas [30]. In this work, we aim to develop editing interactions inspired by these tools where, upon application, augmentations are linked into the formula structure.

*Additional modalities.* Notation can also be authored and annotated with formula-aware drag and drop gestures [17, 68, 73] and handwriting input [37, 40, 62], as well general purpose documentation modalities like sketching [57, 72], touch gestures [63], and voice input [72]. These capabilities can be quite advanced; for instance, some tools allow annotations to make space around themselves [57], and others allow symbolic computation with selected expressions [68, 73]. Where our current design centers simple menu-based augmentation, it could likely benefit from the larger space of HCI affordances for flexible and direct annotation.

## 2.2 Projectional Editors

FreeForm is a kind of projectional editor. Projectional editors are editors that support the manipulation of some model (e.g., program, data) via multiple projections where each projection offers its own affordances [21]. Conventionally, these editors have been proposed to help users edit in ways that align with preferred domain representations [59]. Toolkits for creating projectional editors (e.g., [19, 39, 66]) often assume program ASTs as models, and offer textual, structured, and graphical views that map to AST changes. Projectional editors are frequently being devised by the HCI field to map authoring tasks to the right integrated abstractions, including for authoring graphics [9], hardware schematics [46], data visualizations [70], and data analysis pipelines [38].

FreeForm is a projectional editor optimized for notation augmentation. This paper defines the key projections for the text: textual LaTeX, a formula render with tree-aware selections, and a property/hierarchy view. It provides evidence of the utility of each, both on its own and in coordination with others. It is not a requirement of projectional editors that projections are co-present, though they are for FreeForm, and we show the importance of this decision later in this paper.

There are other projectional editors for LaTeX. In Section 2.1, we name (and differentiate from) structured editors that have projectional aspects, allow some editing of formula renders and access to LaTeX. Gliimpse [18] uses animation effects to show correspondences between projections, in this case ASCII source for symbols in LaTeX source and renders in compiled documents. i-LaTeX [23] introduces projections for several kinds of visuals, including formulas and tables. Its support for formulas is limited to mapping graphical selections to positions in the source.

# 3 System

## 3.1 Design Considerations

Our design was motivated by two major goals for notation authoring. These goals followed from recent studies of notation augmentation [30, 71] and conversations with scientists who had experience writing notation in instructional materials and research communications (4 professors, 2 graduate students, $R1$–$6$).

*Constant availability of LaTeX.* In Head et al. [30]'s study of math augmentation practices, nearly all authors augmented formulas that they had initially written in LaTeX. This included many authors who eventually turned to graphical editors to perform the actual augmentations. This strong preference for LaTeX-based notation authoring was shared by our informants. For 3 of the informants ($R3$–$5$), LaTeX entry was so important that they opted to use Beamer to create their presentations, a LaTeX-based framework for creating slide decks. We note additionally that LaTeX is a formula entry modality in many general audience authoring tools, like Word, PowerPoint, and Keynote; we envision our interactions eventually plugging into environments like these.

*Formula-aware graphical editing.* While LaTeX is a preferred modality for formula input, it creates friction when adding augmentations. Authors in Head et al. [30] described that "code gets horrible looking" as macros are added to it to specify augmentations. This downside was replicated in Wu et al. [71]'s study, where lab study participants frequently made errors related to incorrectly matched braces when using a LaTeX baseline to augment formulas.

Thus, authors sometimes fall back on graphical editors like Google Slides, PowerPoint, Adobe Illustrator, and Mathcha to augment formulas [30]. The downside of these tools is that they treat formulas as collections of naïve shapes. This leads to tedious work positioning and arranging marks [30]. In the words of an author from Head et al.'s study, "What [would it] look like in Google Slides if you could attach pointing labels to things, and now when I move the thing, the label moves?"

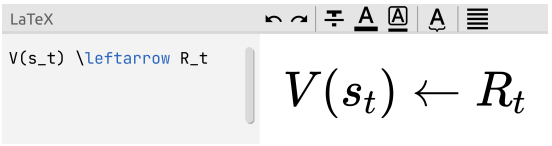
## 3.2 Illustrative Scenario

We designed FreeForm to combine the formula-aware augmentation capabilities of LaTeX with the ease of graphical edits. We demonstrate the most essential features of FreeForm with a walkthrough of its usage on an example formula.[1]

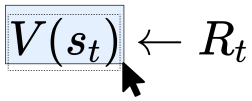
In this walkthrough, the author is trying to add labels to the formula $V(s_t) \leftarrow R_t$ to describe the meaning of its terms in an article they are writing.[2] Here is how they go about that task.

*LaTeX-based formula entry.* The author enters the formula using LaTeX, which they are using for typesetting math in the rest of their document. The formula renders instantly in the graphics pane.

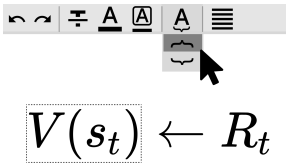
[1] See also the accompanying video figure.
[2] This example was selected from Greydanus and Olah [26].



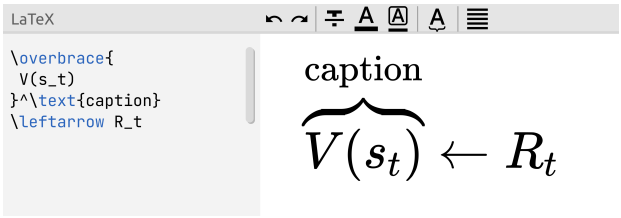

*Selection.* The author wants to augment the formula to explain the meaning of the terms on either side of the arrow—first "$V(s_t)$" and then "$R_t$". Rather than entering LaTeX's macros for labels, the author adds the label directly. To do this, they first directly select "$V(s_t)$" by dragging the mouse over it. Note that the selected term actually contains multiple symbols, all of which should be under the same label.



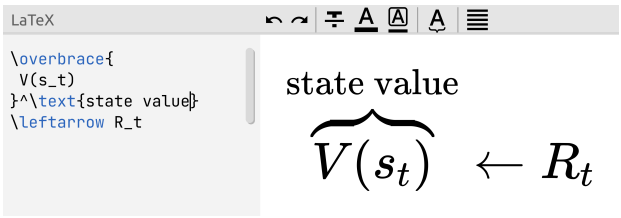

*Augmentation.* They then augment the expression by selecting the menu icon for a label and selecting from the dropdown that they wish to add a label above the selected term.




A label with the placeholder text "caption" is instantly added to the formula.

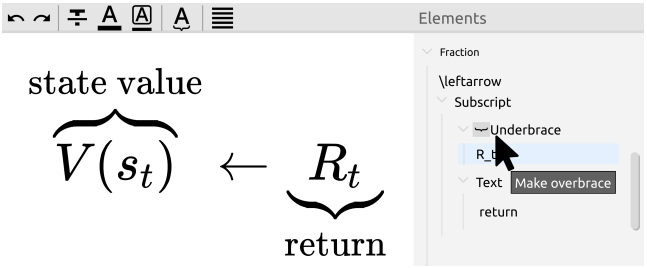



*Markup-grounded edits.* The label augmentation is propagated to the LaTeX markup, wrapping "`V(s_t)`" with the labeling macro ("`\overbrace{...}`"). The author can then change the label by editing the placeholder "caption" text in the markup pane (e.g., to "state value"). Note that the brace automatically resizes, since the label is grounded in LaTeX markup.




*Cross-representation editing.* Since all of FreeForm's representations are grounded in the same markup, the author is free to make changes in any of them. For instance, the author can write the markup for an underbrace label underneath "$R_t$" reading "return".

As soon as they do the new label can be further edited with the graphical editing functionality.




*Reconfiguring with the element pane.* Editing or removing these brace annotations in LaTeX would normally require precise detection of curly braces and editing spanning multiple regions of the markup. If the author wants to flip the brace for this new "return" label from below to above the formula, they can navigate to the augmentation in the "Elements" side bar and click on the brace orientation toggle.

### 3.3 Key Features

The last section depicts how the core ideas of FreeForm enable better editing. The key ideas are that authors can interact with different projections of the formula at will, and graphical edits are ultimately connected back to the formula via markup. Here, we further describe specific important features that FreeForm provides.

*Tree-aware selection.* To augment a formula in FreeForm, the author first has to select math expressions. Basic selections are made by clicking or dragging with the mouse over the rendered elements of a formula. Selections in FreeForm can correspond to glyphs and structures rendered from any set of subtrees of the LaTeX AST, enabling users to apply augmentations with the same level of specificity that they could achieve by writing LaTeX. To allow users to specify such selections, FreeForm considers selecting all children of a subtree to imply selection of the root of the subtree. Figure 2 illustrates an example of selecting different parts of fraction.

Importantly, augmentations apply around the *contiguous coherent subsequences* within a selection. That is, if an author has selected the composite symbol $x_i$ and chooses to add a border box augmentation, the border is drawn around the whole symbol, rather than the component glyphs $x$ and $i$ individually. We also manifest the tree structure of the LaTeX AST in the element pane to enable precise selection when graphical selections may be ambiguous.

*Augmentations.* FreeForm supports applying colors, boxes, strike marks, and brace labels via simple menus, obviating memorization of macro names and fiddling with braces and syntax. Figure 3 shows the different augmentations that can be applied using FreeForm.

*Alignment.* FreeForm provides an interaction mode for manipulating the alignment of a multi-line formula (e.g. for systems of equations). Rather than place "&" signs in markup, authors can click and drag column boundaries to move them. As they do, they receive feedback as a line moving with their cursor showing what the new column will divide. Figure 4 illustrates an example of aligning a system of equations.

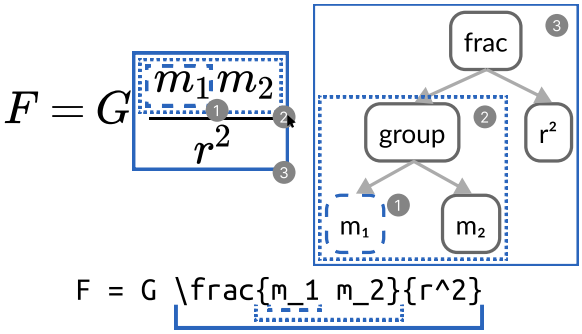



**Figure 2: Selection mechanics.** The user interactively selects formula regions by dragging their mouse (left). This corresponds to selecting subtrees of the LaTeX AST (right), which are just regions of the LaTeX markup (bottom). Border styles correspond to the subtrees and ranges of markup that correspond to a particular dragged selection. Here, as the user drags their cursor over $m_1$ (❶), then the numerator group (❷), then the entire fraction (❸), the selection grows to surround the highest coalesced selection.

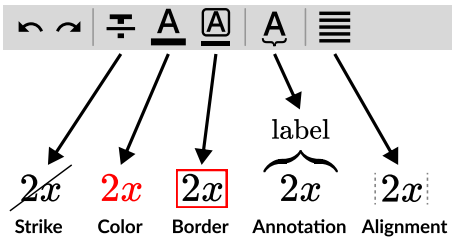



**Figure 3: Supported augmentations.** FreeForm supports adding strike marks, color, border boxes, and brace annotations.

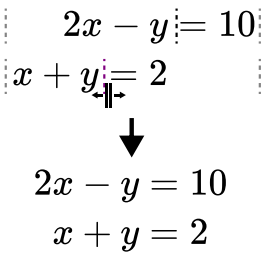


**Figure 4: Alignment.** Alignment can be adjusted interactively by dragging alignment separators between elements in the formula.

*Modification.* The element pane on the right of FreeForm (Figure 1, ❺) manifests the tree structure of the LaTeX AST as a hierarchical tree view. Augmentations appear with widgets that enable rapid reconfiguration (changing colors, flipping label directions) and deletion. Authors can quickly undo and redo changes using buttons in the center pane.

## 4 Implementation

FreeForm is implemented as a client-only, single-page React.js application, enabling its potential use as a component within other web-based authoring environments. We use MathJax [8] to render LaTeX markup as HTML and associate rendered elements back to AST nodes. [3] The markup pane is an instance of CodeMirror [28].

FreeForm needs to propagate edits between LaTeX markup and its graphical representation. We follow the common approach of projectional editors, which ground projections and edits made with them as views and edits of the underlying AST. Graphical interactions are modeled as transformations of the LaTeX AST, and the markup is kept in sync by serializing the AST. Pasted and edited markup is incorporated into the model by simply parsing the markup. Rapid feedback on edits is provided by serializing the LaTeX AST and rerendering on every change.

*Augmentations.* In the most straightforward case, augmentations are added by inserting a parent of the selected AST node which represents the augmentation, which is then serialized to LaTeX and rendered to update the markup and graphics panes. Similarly, deletions and reconfigurations (e.g. changing color) in the element pane are implemented by removing or changing the parameters of an augmentation node. In some cases, FreeForm performs smarter reconciliation of edits. For example, selecting the exact same elements and applying colors twice will modify the existing color augmentation instead of introducing a new one.

*Alignment.* The alignment augmentation requires a separate implementation from the other augmentations. Different cells of the alignment are forests of AST subtrees stored as a 2D array. Adding an alignment marker divides the corresponding cell node in the AST into two cell nodes separated at the subtree implied by the mouse position. Removing an alignment marker merges the adjacent cells.

## 5 Controlled Study

We designed a controlled study in which participants completed a number of augmentation tasks, each with FreeForm and two baselines. We sought to answer two research questions:

RQ1. Does FreeForm speed up formula augmentation?
RQ2. Does FreeForm make formula augmentation easier?

### 5.1 Methodology

*Participants.* We recruited 30 participants from mailing lists for Master's and senior-level Bachelor's students in the computer science department at a private university. Participants were required to have prior experience writing LaTeX formulas. Participants rated their comfort with LaTeX as a median of 4 on a 5-point Likert scale ($\sigma = 0.97$). 4 reported using LaTeX daily, 4 weekly, 12 monthly, and 10 less than monthly.[4] We refer to these participants as $C1$–$30$.

*Baselines.* We compared FreeForm to two baseline tools. The first is Overleaf, a de facto environment for authoring LaTeX documents with an estimated 18M users [55]. The Overleaf environment was set up with all necessary LaTeX packages for augmenting notation. The second is FFL [71] (see Section 2.1). FFL was our stronger

[3] IDs of AST nodes are propagated to HTML elements by wrapping the LaTeX for each node in MathJax's "`\cssId`" macro.
[4] We did not ask participants for their familiarity with FFL. We assumed no prior knowledge, given FFL's status as research software.

baseline given that it was shown in a recent study to reduce the difficulty of performing some augmentation tasks.

*Tasks.* Three tasks were assigned, each focusing on a different kind of augmentation: color ($T_C$), labels ($T_L$), and alignment ($T_A$). In each task, the participant was given an unaugmented formula, and asked to augment it to match a reference image.[5] To prevent sessions running over time, we cut participants after 8 minutes spent on a task,[6] though this was rarely needed in practice.

The study was within-subjects: each participant completed all tasks with all 3 interfaces.[7] They used one interface at a time, completing all three tasks with an interface before moving on. Interface order was counterbalanced to mitigate learning effects; there were 6 potential interface orders, and each was assigned equally often across participants.[8]

Prior to the first task with any interface, the participant watched a 5-minute demo on doing the tested augmentations with the interface and practiced on an example formula.[9] Participants were allowed to refer back to written tutorial materials during the tasks.

*Setting.* Study sessions took place in a dedicated study room. Most participants used a study computer we provided. 4 used their own laptops due to a strong personal preference to have access to their familiar keyboard configurations.

*Measurements and analysis.* All tasks were timed. The study facilitator checked for task completion by comparing the participant's screen to a reference result. After each task, the participant reported the difficulty of the task on a 7-point Likert scale.

To compare task time across interfaces, we fit linear mixed-effects models. In these models, tools, tool order, tasks, and the interaction between tool and task were fixed effects, and participant was a random effect. Significance was assessed with an $F$-test with Satterthwaite's estimate of effective degrees of freedom [58]. $F$-values were adjust using the Holm–Bonferroni method [31]. We assessed pairwise differences using Tukey's HSD test [64]. To compare difficulty across interfaces, we conducted Friedman tests within tasks [22], and assessed pairwise differences using Conover [11] tests. For $T_A$, we assessed significant with the Wilcoxon signed-rank test [69], as it was completed with only two interfaces.

## 5.2 Results

Below, section 5.2.1 answers RQ1, and section 5.2.2 answers RQ2.

*5.2.1 Effect on task speed.* Nearly all tasks (98%) were completed within the 8-minute time limit. For all tasks done with FreeForm, the task was completed within time limits. 3 tasks were not completed with Overleaf, and 3 were not completed with FFL. Our analysis showed a significant effect of tool on task speed ($F = 29.5$, $p < 0.001$). The effect varied by task, with a significant interaction between tool and task ($F = 30.5$, $p < 0.001$, see Figure 5).

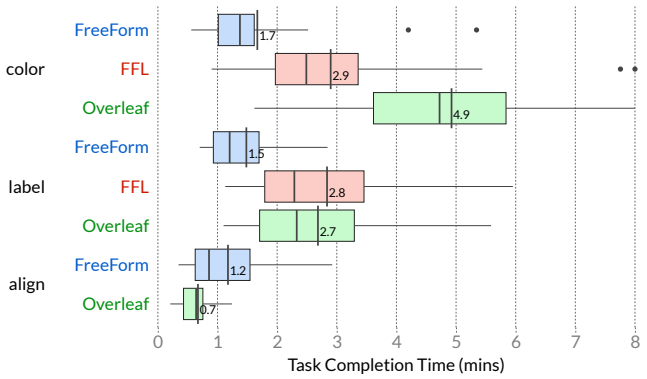



**Figure 5: Task completion times by tool.** Box plots illustrate the distribution of completion times for each tool across the coloring, labeling, and alignment tasks. Mean completion times are indicated by gray vertical lines within each box, with corresponding values labeled. FreeForm led to significantly faster task completion times than the baselines for the coloring and labeling tasks, and significantly slower times for the alignment task.

Post-hoc comparisons clarified the direction of the difference across tasks. For coloring task $T_C$, FreeForm led to the fastest completion times ($\mu = 1.7$ min $(m)$, $\sigma = 1.1m$), followed by FFL ($\mu = 2.9m$, $\sigma = 1.7m$) and Overleaf ($\mu = 4.9m$, $\sigma = 2.0m$). FreeForm was significantly faster than both FFL ($p = 0.011$) and Overleaf ($p < 0.001$). FFL was significantly faster than Overleaf ($p < 0.001$). For labeling task $T_L$, FreeForm also led to the fastest completion times ($\mu = 1.5m$, $\sigma = 0.87m$), followed by Overleaf ($\mu = 2.7m$, $\sigma = 1.2m$) and FFL ($\mu = 2.8m$), $\sigma = 1.7m$). FreeForm was significantly faster than both Overleaf ($p = 0.002$) and FFL ($p < 0.001$).

The direction of effect was different for the alignment task, where participants performed the task more quickly with Overleaf ($\mu = 0.67m$, $\sigma = 0.39m$) than with FreeForm ($\mu = 1.2m$, $\sigma = 0.78m$). This difference was statistically significant ($p = 0.002$).

*5.2.2 Effect on difficulty.* Within each task, there was a significant effect of tool on difficulty (Figure 6). For coloring task $T_C$, there was a significant difference between tools ($F = 29$, $p < 0.001$). Post-hoc Conover tests revealed that FreeForm and FFL were both significantly less difficult to use than Overleaf ($p < 0.001$ for both comparisons). There was no significant difference between FreeForm and FFL. For labeling task $T_L$, there was also a significant effect ($F = 11$, $p = 0.004$), once again with FreeForm and FFL reported significantly less difficult than Overleaf ($p < 0.001$) and no significant difference between FreeForm and FFL. For alignment task $T_A$, there was also a significant effect of tool on difficulty ($W = 28.5$, $p = 0.012$), in this case with Overleaf reported less difficult to use than FreeForm.

In an end-of-study questionnaire, participants rated the ease of using each tool overall, on a 7-point Likert scale where a high score indicates high agreement. There was a significant difference between the ease for the three tools (Friedman's test, $F = 15$, $p < 0.001$). Post-hoc tests showed that FreeForm ($\mu = 5.2$, $\sigma = 1.7$) was rated significantly easier to use than Overleaf ($\mu = 3.4$, $\sigma = 1.5$, $p = 0.001$), and FFL was also rated significantly easier to use than

[5] Details for individual tasks appear in Appendix A and the supplemental material.
[6] Based on upper limits of task completion times observed in pilot studies.
[7] With the exception of $T_A$, which could not be performed with FFL.
[8] Task order was not counterbalanced to simplify the set of assignable conditions. The design embeds an untested assumption that if learning effects are present across tasks, the effects would be similarly felt across interfaces.
[9] See tutorial materials in the supplemental material.

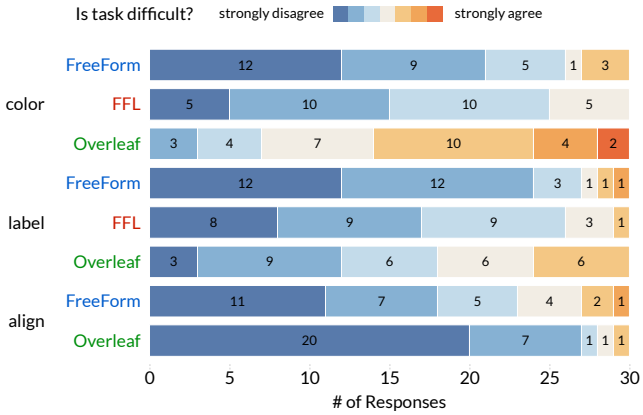



**Figure 6: Self-reported difficulty in performing tasks with each tool.** Participants were asked to indicate their agreement with the statement "It was difficult to complete the task" on a 7-point Likert scale, from "strongly disagree" to "strongly agree". FreeForm and FFL were reported significantly easier to use than Overleaf for the coloring and labeling tasks, with no significant difference between FreeForm and FFL. Overleaf was reported as significantly easier to use than FreeForm for the alignment task.

Overleaf ($p = 0.001$). However, no significant difference was found between FreeForm and FFL ($p = 0.880$).

## 6 Observation Study

We performed an observation study to assess the value and usability of FreeForm when used for more substantial, realistic document writing tasks. This study answered two research questions:

RQ3. How do authors leverage FreeForm with their document editor during a longer-form authoring task?
RQ4. Which features of FreeForm are useful and why?

### 6.1 Methodology

*Participants.* We recruited 17 participants from mailing lists for Ph.D. students at a private U.S. university. All had at least intermediate proficiency in writing math notation with LaTeX (0 beginner, 10 intermediate, 6 advanced, 1 expert). The majority had at least intermediate proficiency adding annotations to notation with LaTeX (6 beginner, 7 intermediate, 4 advanced, 0 expert). All had some experience teaching, measured as the number of semesters of TA experience, with a median of 4 semesters ($\sigma = 2$). All were comfortable with the math involved in the task, reporting a median of 7 on a 7-point Likert scale ($\sigma = 0.7$).

*Task.* Each participant was asked to author a handout for an imagined high school-level physics class. We provided participants with a hand-written derivation and imagined walkthrough explanation script. We asked participants to deprioritize writing prose or creating graphics to ensure they had time to complete the derivation, though most participants wrote some text. To promote mathematical engagement, we encouraged them to modify the math content however they saw fit.

*Authoring environment.* Participants wrote up the handout in Overleaf. Of the 17 participants, 11 were given access to FreeForm (referred to as $O1$–$11$). Participants were not required to use FreeForm but were told to use it as they wished. The remaining 6 participants ($L1$–$6$) helped us ground what augmentation looks like without FreeForm. To decide on how many participants to recruit, we followed a rough theoretical sampling approach [12], recruiting for each condition for as long as we were seeing new behaviors.

*Training.* Prior to the task, the participant was given training in FreeForm, or, if they were assigned to the Overleaf-only condition, LaTeX-based augmentation. The training consisted of a written tutorial document. The document showed examples of how to apply all 5 kinds of augmentations, and required participants to apply each augmentation to an example formula.

*Setting.* All sessions took place on the Zoom remote video conferencing software. FreeForm was hosted on a public website accessible over the internet. All study documents (tutorials, instructions) were sent to the participant over Zoom chat.

*Duration.* The participant worked on the task either for 30 minutes, or less time if they felt they had finished earlier.

*Analysis.* During the task, participants were told that they should focus on authoring and that we might interrupt them to ask questions about their process. We also asked them to reflect on their experience in a post-task questionnaire and a brief post-task interview. For 8 of the 11 authors using FreeForm, we asked about the usefulness of each of FreeForm's main features.[10] Participants were able to report, for each feature, whether it was "not useful," "somewhat useful," "very useful," or they "did not use" it. One author conducted all sessions and took detailed notes on participant behaviors and utterances, paying close attention to whether they made a particular edit in Overleaf or FreeForm, what and how features were invoked, mouse and keyboard activity, and time spent. This author then inductively coded these notes and the questionnaire responses in their entirety and grouped them into common *observed workflows*, *design successes* ($S1$–$6$), and *design gaps* ($G1$–$3$) in a process inspired by thematic analysis [5]. The set of codes was determined through conversation between the first and last author after the latter reviewed preliminary coding results. The codes that were retained were those that were seen as providing actionable guidance for future editor design.

### 6.2 Results

Our observations from this study help answer RQ3 by enumerating workflows that participants followed (Section 6.2.1). They answer RQ4 by finding which aspects of the design were successful (Section 6.2.2) as well as pitfalls that future tools for markup augmentation should avoid (Section 6.2.3).

*6.2.1 Observed workflows.* FreeForm was used to support a number of different authoring workflows.

[10] It was not until after the first few sessions that we realized the benefits of collecting this information consistently from participants.

*Goal-directed augmentation.* Most authors at some point seemed to have an augmentation already planned when they opened FreeForm. In these situations, interactions with FreeForm were brief, lasting less than a minute. We observed this pattern of usage for nearly all authors. It was most common when authors applied strike marks ($O3$–6, $O10$) or when applying a previously-designed color scheme to instances of identifiers in a new formula ($O2$, $O4$).

*Exploratory augmentation.* 4 authors ($O1$, $O2$, $O5$, $O10$) engaged in deeper, "bursty" interactions with FreeForm where they spent 1–2 minutes trying out different variants of augmentations before arriving at a variant they liked. We most frequently observed this behavior when an author was experimenting with different colors. $O1$ and $O2$ also switched between kinds of augmentation, swapping out colors with boxes or vice versa.

*Cleaning.* 2 authors ($O4$, $O5$), while following a copy-and-paste-driven workflow for authoring a derivation, used the element pane to completely remove all augmentations from a pasted line before modifying the notation. $O4$ did so between FreeForm's markup and element panes, whereas $O5$ did so between Overleaf and FreeForm. $O5$ voluntarily remarked that being able to do so "is actually really useful", as the alternative would have involved precisely editing a lot of LaTeX markup.

*Lookup/tutorial usage.* 2 authors ($O3$, $O9$) used FreeForm as a reference for looking up syntax for augmentation macros. These authors entered placeholder formulas that they were not planning to use in their document, augmented them, and then copied the macros that appeared in the LaTeX. These authors described the value of FreeForm as akin to other tools that allow users to look up LaTeX macros for symbols (e.g. Greek letters).

*Bootstrapping augmentation.* Not all augmentation work took place in FreeForm. Rather, authors sometimes wrote or tweaked augmentation markup in Overleaf. One variant of this pattern is that 3 authors ($O4$–6) used FreeForm to "bootstrap" the creation of augmentation markup across their document. After introducing a great deal of augmentations to a formula using FreeForm, they would augment later formulas by reusing pieces of the augmented formula LaTeX rather than opening FreeForm. $O5$ described their preference for this workflow, telling us, "I'm currently more familiar with editing on Overleaf and I can keep track of the `textcolor` for [an identifier] on Overleaf…once I have the code for the correct colors for these elements, I find it much faster for me to just copy and paste that code in other places within the code editor."

*Copy & paste authoring.* We observed that participants, when editing both Overleaf and FreeForm, authored both notation and augmentations by copying and pasting. They did so in two ways: fully replicating a previous line of markup before modifying it ($O1$, $O3$–5, $O8$, $O9$) and copying fragments of markup to compose a new line incrementally ($O1$, $O4$–6, $O10$). Though this workflow is not a direct consequence of our design, we mention it here as it may be relevant to future editors. Only 2 authors ($O2$, $O7$) never performed this copy-and-paste authoring workflow.

*6.2.2 Design successes.* What features should future projectional markup editors have to best support authors in making annotations? Here, we review the aspects from FreeForm that seemed to accelerate authors in our study.

*S1: Tree-aware selections.* Tree-aware selections were at times crucial for authors to apply augmentations in the way they wanted. With boxes ($O1$, $O2$, $O10$), brace annotations ($O7$, $O10$), and strike marks ($O4$), participants interactively applied augmentations to groups of elements such as fractions that would not have been possible to achieve graphically without our tree-aware selection mechanisms.

*S2: Element pane.* 5 participants ($O1$, $O2$, $O4$, $O5$, $O10$) used the element pane to modify or remove an augmentation. They did not strictly need to do so: participants also seemed comfortable undoing and reapplying augmentations ($O4$, $O10$) or modifying augmentation markup by hand ($O1$, $O5$, $O6$, $O10$). The element pane was also used to make ($O1$) or refine ($O2$, $O5$, $O10$) selections.

*S3: LaTeX editing.* Many participants took advantage of the ability to edit LaTeX within FreeForm. 5 participants ($O1$, $O4$, $O8$, $O10$, $O11$) authored all of their notation exclusively within the FreeForm markup pane, and all rated the ability to edit LaTeX as "very useful". $O5$ and $O7$ sometimes authored or edited notation in FreeForm while in the middle of performing augmentations.

The constant presence of editable LaTeX also provided an alternative to graphical editing, allowing authors to adjust the markup of augmented formulas if graphical edits did not match their expectations. For example, the interactive alignment features in FreeForm do not currently permit adding empty columns; when $O4$ encountered this, they switched to the markup pane and manually added the "&" where appropriate to resolve their issue.

Another example of adjusting augmentation markup is when $O5$ encountered the scenario shown below, where applying a cancellation augmentation to notation that was already colored caused the strike mark to be colored when they actually wanted it to be black.

| | | |
|---|---|---|
| Expected | $\not{v_0}$ | `\cancel{\textcolor{purple}{2}}` `\textcolor{purple}{v_0}` |
| Actual | $\not{v_0}$ | `\textcolor{purple}{\cancel{2}v_0}` |

Recognizing what had happened by examining the LaTeX in the markup pane, and perhaps deciding that the equivalent graphical edit would be harder (removing both augmentations, reapplying color to 2 and $v_0$ separately, reapplying the cancellation to 2), $O5$ instead edited the markup directly, breaking up the notation to be colored separately and wrapping the first in the `\cancel` macro.

*S4: Graphical editing.* The strongest indication of the value of graphical editing was its voluntary (and for many participants, sustained) usage. About half of the authors ($O2$, $O7$, $O9$, $O10$, $O11$) exclusively used graphical editing to augment formulas. All authors created at least one augmentation using the graphical editor.

An additional signal of the value of graphical editing is an observation about the authors who were not given access to FreeForm. Of those 6 authors, 4 spent at least some time debugging syntax errors (usually mismatched braces) related to adding augmentations, sometimes for minutes at a time. For $L3$, who self-identified as an intermediate LaTeX user, the syntax errors precluded their finishing the task in the allotted time.

Interactive alignment, specifically, was a graphical editing feature on which authors' editing tendencies diverged. Three participants (*O*7, *O*9, *O*10) exclusively performed alignment interactively using FreeForm, and 2 more (*O*1, *O*4) edited alignment both in markup and interactively. *O*7 remarked that "I find it difficult to picture how the equations will look when I add the ampersands so I found it easier to use the align [in FreeForm] to make sure that everything is aligned on the actual character that I want" and that "[In Overleaf] I would have had to make the change over all the rows, compile again, see whether it makes sense. In [FreeForm] I can kind of do it line-by-line." However, the remaining participants (*O*2, *O*3, *O*5, *O*6, *O*8, *O*11) exclusively edited alignment in markup.

*S5: Co-present projections.* Different projections had different affordances, and authors were able to use them in concert since they were available at the same time. Authors who wrote their math notation in FreeForm sometimes transitioned directly to adding augmentations graphically (*O*1, *O*4, *O*5).

Authors also went in the other direction. *O*1 at one point had colored a symbol and wanted to extend that color to a neighboring symbol. The graphical edit would have been quite involved—remove the color, select both symbols, reapply the color. Instead, *O*1 edited the LaTeX to move the neighboring symbol inside of the coloring macro. And as discussed in S3, other participants (*O*4, *O*5) edited the LaTeX markup when graphical editing affordances were not sufficient to achieve specific augmentation goals.

Authors transitioned between other representations as well. For instance, they commonly edited (*O*1, *O*2, *O*10) or deleted (*O*1, *O*2, *O*10) augmentations they had just created in the graphical pane by interacting with the element pane. *O*4 once propagated unwanted augmentations within copied and pasted notation, recognized the issue by looking at the rendered formula in the graphical pane, selected the elements graphically to highlight them in the element pane, and removed the augmentations using the element pane.

Connections between co-present projections allowed authors to leverage the strengths of one representation to aid in using another. *O*4 graphically selected augmented notation to locate the corresponding augmentations in the element pane. *O*2 clicked on elements in the element pane, then looked in the graphical pane to understand where in the formula they were. *O*5 and *O*10 both accidentally selected too many terms with a drag selection in the graphics pane and used the element pane to refine their selections.

It would be better to have even more of these cross-projection selection affordances. 3 participants (*O*2, *O*8, *O*9) wished for graphical selections to extend to the markup pane, as well as the reverse. After *O*8 unsuccessfully attempted to highlight some markup, expecting it to select the corresponding graphical elements, they remarked "I think it was sort of a reflex…it would be interesting if I could hover over any part and then I could see the piece of code that is corresponding to that element."

*S6: Rapid feedback.* The ability to preview formulas live appeared to be very valuable. All 8 participants that we asked to rate feature usage (*O*4–*O*11) used the live previews, and all rated it "very useful". *O*10, when asked about why they authored all of their notation within FreeForm, responded "because it has instant preview."

*6.2.3 Design gaps.* Our study revealed multiple ways in which this workflow did not match participants' actual behavior and gaps in workflow that tools like FreeForm should fill.

*G1: Integration.* Needing to go between FreeForm and Overleaf likely inhibited some participants' usage of FreeForm. *O*8 stated that "I wish I could do this all inside [FreeForm]. I think it's hard to go back and forth…because there are limitations in [FreeForm]…but I also can't ask Overleaf to do the click and color stuff that [FreeForm] does."

We hypothesize that when authors did switch to FreeForm, they felt that the editing task in front of them would sufficiently benefit from graphical editing. The types and sizes of tasks that merited switching varied from author to author. For example, *O*7 was the only participant who always switched to FreeForm to use its interactive alignment, whereas the rest were often content to write ampersands by hand. Another example is that 3 participants (*O*2, *O*10, *O*11) always added colors graphically, while the rest sometimes added colors in code.

The way that participants authored inline math also exemplifies this switching cost. We observed that participants *never* reached for FreeForm when writing inline math, even when it contained colors. We hypothesize that this is because inline math is typically short, and even if it includes augmentations, participants would rather stay in-context with their writing rather than switch tools.

4 participants (*O*2, *O*4, *O*9, *O*11) explicitly asked for deeper integration of FreeForm with their document editor in their post-task survey responses and indicated that the existence of such an integration would be important to their potential future usage of FreeForm. These authors envisioned being able to load the formula under the cursor into FreeForm, modify its augmentations, and then jump right back into the editor with the changes integrated.

*G2: Propagating changes.* We observed that several authors (*O*2–4, *O*8, *L*1–3) committed to a choice of applying the same color to every instance of a symbol in their document. 4 later regretted this decision, and *O*2 went to the effort of completely removing the colors. *O*4 wished for a way "to have propagated that choice more automatically. I'd like to be able to choose a color scheme for those variables and then think less about it."

Authors expended a fair bit of effort to perform these mass colorings. *O*2 brought each formula block into FreeForm and performed large multi-select and coloring operations. Others (*O*4, *L*1, *L*3) did so by authoring the notation with the colors in the first place, following a copy-and-paste workflow. 2 authors (*O*3, *O*8) attempted to define new macros, but they did not integrate well with FreeForm. *L*3 did successfully define new macros and later used them to undo their coloring decision.

*G3: Aggressive LaTeX synchronization.* FreeForm, in its prioritization of synchronization between representations, preserves the last syntactically valid formula when changing markup, and restores this markup if the markup pane focus is lost. This sometimes caused frustration for 3 participants (*O*4, *O*7, *O*10) when they authored partial (not syntactically valid) LaTeX markup and subsequently lost it. For these participants, reverting markup to match the currently displayed formula did not match their expectations and caused them to lose their work. These 3 participants voiced frustration or adjusted their authoring workflow to compensate, making sure

that their markup was syntactically valid before switching focus. We believe that FreeForm should instead allow the markup pane to diverge from the graphical representation and give authors a means to restore the previous markup. More broadly, this raises a concern for all markup-grounded projection editors of how projections should handle invalid markup.

## 7 Discussion

In this section, we consider the research questions posed in Section 5 and Section 6. We frame the projectional, graphical editing math augmentation paradigm shown in FreeForm against the existing paradigms of inline markup and external styling languages. We put our results in context by considering the extent to which they can generalize beyond our study. Finally, we chart out next steps in this line of research that build on the insights from this paper.

### 7.1 Answers To Research Questions

*RQ1: Speed gains.* We observed in our controlled study that FreeForm significantly improved author speed over both baselines on the coloring and labeling tasks. However, on the alignment task, authors were actually slower. With these results, our answer to whether FreeForm sped up the process of augmenting formulas is *yes, but it depended on the task.*

*RQ2: Ease of augmentation.* Across our two studies, we observed that FreeForm can make tasks easier, but it varies by author and exact task. Participants in our controlled study rated both FreeForm and FFL as easier than Overleaf on the coloring and labeling tasks, but there was no significant difference between FreeForm and FFL on difficulty. However, in our observation study, authors sometimes needed to propagate color changes across their documents (G2), a task that is cumbersome in FreeForm but easy with FFL.

On the controlled alignment task, participants rated Overleaf as significantly less difficult than FreeForm. In our observation study results, many participants ignored interactive alignment but a few strongly preferred it to adding ampersands in markup (S4).

*RQ3: Interesting workflows.* In our observation study, we observed participants engaging in a variety of workflows such as edits spanning multiple representations, goal-directed and exploratory augmentation, and macro lookup (Section 6.2.1). These were enabled by FreeForm's multiple representations, which gave authors access to relevant tools at each step (S2–S4), and its grounding in markup, which allowed authors to freely switch between them (S5).

*RQ4: Design successes and opportunities.* Our key design point of LaTeX-grounded graphical editing (S1, S4) was validated by participants' continuous and successful usage of it in our observation study. Participants took advantage of the ability to write LaTeX (S3) and cross-representational affordances (S5). However, we also observed gaps in integration (G1), propagating changes across an entire document (G2), and over-aggressive synchronization of markup with graphics (G3).

### 7.2 Augmentation Modalities–Better Together

Through our controlled and observation studies, we observed benefits to all three of the editing paradigms for augmenting formulas that we considered. As a result, we now believe that successful future tooling for math augmentation (and potentially other markup-grounded, projectional editors) should incorporate the best of all three paradigms so that authors have ready access to the appropriate tool in any editing situation. Below, we use *Cn* to refer to responses gathered from our controlled study participants when they were asked to explain which editing tools they found easier or harder to use.

#### 7.2.1 *Inline markup: LaTeX.*

*Aspects to borrow.* As discussed in section 3.1, authors are used to writing math notation in LaTeX and work with many tools that can take LaTeX markup as input. In our observation study, in situations where authors were already writing LaTeX, some authors tended to continue editing in LaTeX, even for applying augmentations (G1). LaTeX is also the most flexible representation, as we observed with participants adjusting markup when graphical edits weren't sufficient (S3).

*Why it's not enough.* The tradeoff is that LaTeX can be hard to write, requiring authors to remember macros, find relevant segments of markup to augment, and fuss over syntax and braces. On the color and labeling tasks in our controlled study, participants considered LaTeX significantly harder to use.

#### 7.2.2 *External styling languages: FFL.*

*Aspects to borrow.* FFL and FreeForm showed similar levels of ease in the controlled study (Section 5.2). Some participants praised how with FFL, "it is faster to change many things at once" (*C*16) and "selecting all instances of a string is very handy" (*C*23). In our observation study, authors using LaTeX and FreeForm sometimes needed to be able to propagate edits across their documents and wanted an easier way (G2). This is precisely what FFL excels at.

*Why it's not enough.* FFL carries a high learning burden. The FFL authors [71] note its issues with lacking "closeness of mapping" [25] with LaTeX, requiring authors to learn a new syntax. FFL also presents challenges when precisely targeting augmentations, requiring authors to write complex intersection selectors or count number of occurrences of a symbol. As one of our controlled study participants wrote, it "will sometimes cause issues from overlapping selection domains" (*C*13).

#### 7.2.3 *Graphical editing.*

*Aspects to borrow.* Graphical editing with FreeForm allowed participants to precisely select (S1) and apply augmentations without needing to remember macros and worry about syntax (S4). For some augmentation edits, graphical edits were faster or easier to perform (S2, Section 5.2).

*Why it's not enough.* Graphical edits are not necessarily faster. In our controlled study, alignment was both faster and easier in LaTeX markup than it was using FreeForm (Section 5.2). Some specific augmentation edits could not be easily expressed using graphics, causing participants to edit the markup to compensate (S3).

## 7.3 Limitations

The generalizability of our findings are limited by several aspects of the study design. Our controlled and observation study involved primarily students (Ph.D. students for the observation study, primarily Master's students for the controlled study). While participants reflected the ability of somewhat experienced LaTeX users, we expect that performance and behaviors would vary if we were to recruit those who had extensive prior experience augmenting formulas.

Differences in augmentation performance were assessed with respect to just one task for each kind of augmentation. Differences could wash out for smaller augmentation tasks, where LaTeX is less tangled or the FFL is less voluminous.

Though we provided participants in our controlled study with training materials for all three conditions, it is possible that the statistically significant gap in augmentation speed between FreeForm and FFL on the coloring and labeling tasks is explained by FFL's higher learning curve and participants' relative inexperience.

One gap in the findings of the controlled study is the absence of a comparison to a graphical editing baseline. Without this comparison, we cannot characterize the impact of our syntax-informed graphical editing compared to a conventional graphics editor. Some of our features seem to strictly reduce the work required (e.g. since FreeForm is based in LaTeX, authors do not need to reposition labels after changing formula content), but it is experimentally unproven whether our editing paradigm has any drawbacks compared to conventional graphics editors.

## 7.4 Future Work

*Integrations.* The results of our observation study (G1) and our analysis of the necessary features from each of the augmentation paradigms (Section 7.2) strongly motivate integration of an markup-grounded, projectional editor like FreeForm into a full document editor such as Overleaf or VS Code. This would allow authors to access graphical editing affordances quickly with their formula content already loaded within broader document authoring workflows. If combined with a tool like FFL, an editor tool could support a powerful combination of rapid localized augmentations and propagation of those edits across an entire document.

*Tighter synchronization between markup and graphics.* Some participants desired linked selections, wherein graphical selections would highlight the corresponding markup in the LaTeX, or vice versa. Other quality of life improvements such as editing captions in the graphical editor might additionally prove useful, requiring text editing functionality within the graphical editor itself.

*Expanding the space of augmentations.* Font brightness, background colors, and font styles all fit within the existing interaction mechanisms and require small changes to add support. Other kinds of augmentations, such as positioning (`\vspace` and `\hspace`) and more flexible labels (`annotate-equations`) will require new patterns of interaction design and more involved implementations.

# 8 Conclusion

In this work, we introduced FreeForm, an interface for augmenting formulas with co-present projections and graphical edits grounded in the underlying markup. In a controlled study, we found that participants were able to apply color and labeling augmentations significantly faster than with a conventional LaTeX editor and a state-of-the-art formula augmentation tool and with less difficulty than a conventional LaTeX editor. In an observation study, we found that participants used graphical editing to augment formulas quickly and that they took advantage of the simultaneous availability of graphical and markup-based editing. We also discovered that users needed tighter integration between their document-editing and augmentation workflows, with the ability to make both precise and document-wide changes. From these observations, we proposed how future editors should bring together the availability of inline markup, the ability to make broad changes with styling DSLs, and the precision and expressive ease of graphical editing.

## Acknowledgments

We thank the scientists who spoke to us about their authoring workflows and helped inform the motivations in Section 3.1.

## References

[1] Andrew M.H. Alexander. [n. d.]. *Exponential Improvements.* http://www.andrusia.com/math/complex-numbers/eulers-identity.pdf Last accessed September 12, 2024.
[2] David Attwood. [n. d.]. *Radiation by an Accelerated Charge: Scattering by Free and Bound Electrons.* https://people.eecs.berkeley.edu/~attwood/srms/2007/04_new_2007.pdf Last accessed September 12, 2024.
[3] AxMath [n. d.]. https://www.axsoft.co/.
[4] Paul Ayres. 2006. Impact of Reducing Intrinsic Cognitive Load on Learning in a Mathematical Domain. *Applied Cognitive Psychology* 20, 3 (2006), 287–298.
[5] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (2006), 77–101.
[6] David P. Carlisle. 1997. *The* `color` *package.*
[7] Andrew N Carr. 2021. *Everyday data science.*
[8] Davide Cervone. 2012. MathJax: A Platform for Mathematics on the Web. *Notices of the American Mathematical Society* 59, 02 (Feb. 2012), 1.
[9] Ravi Chugh, Brian Hempel, Mitchell Spradlin, and Jacob Albers. 2016. Programmatic and direct manipulation, together at last. In *Proceedings of the 37th ACM SIGPLAN Conference on Programming Language Design and Implementation.* ACM, 341–354.
[10] The Jupyter Book Community. [n. d.]. Jupyter Book.
[11] WJ Conover. 1999. *Practical nonparametric statistics.* John Wiley & Sons, Inc.
[12] Juliet Corbin and Anselm Strauss. 2008. *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory* (3rd ed.). SAGE Publications.
[13] Microsoft Corporation. 2024. Microsoft Powerpoint.
[14] Microsoft Corporation. 2024. Microsoft Word.
[15] Will Crichton. [n. d.]. A New Medium for Communicating Research on Programming Languages. https://willcrichton.net/nota/. Last accessed September 12, 2024.
[16] Diana Dee-Lucas and Jill H Larkin. 1991. Equations in scientific proofs: Effects on comprehension. *American Educational Research Journal* 28, 3 (1991), 661–682.
[17] Yancarlos Diaz, Gavin Nishizawa, Behrooz Mansouri, Kenny Davila, and Richard Zanibbi. 2021. The MathDeck Formula Editor: Interactive Formula Entry Combining LaTeX , Structure Editing, and Search. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems.* ACM, 1–5.
[18] Pierre Dragicevic, Stéphane Huot, and Fanny Chevalier. 2011. Gliimpse: Animating from markup code to rendered documents and vice versa. In *Proceedings of the 24th annual ACM symposium on User interface software and technology.* ACM, 257–262.
[19] Andrew D. Eisenberg and Gregor Kiczales. 2007. Expressive programs through presentation extension. In *Proceedings of the 6th international conference on Aspect-oriented software development.* ACM, 73–84.
[20] Equation Editor for online mathematics - create, integrate and download [n. d.]. https://editor.codecogs.com/. Last accessed September 12, 2024.
[21] Martin Fowler. 2008. *Projectional Editing.* https://martinfowler.com/bliki/ProjectionalEditing.html
[22] Milton Friedman. 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the american statistical association* 32, 200 (1937), 675–701.
[23] Camille Gobert and Michel Beaudouin-Lafon. 2022. i-LaTeX : Manipulating Transitional Representations between LaTeX Code and Generated Documents. In *CHI Conference on Human Factors in Computing Systems.* ACM, 1–16.

[24] Tess N. Grainger, Athmanathan Senthilnathan, Po-Ju Ke, Matthew A. Barbour, Natalie T. Jones, John P. DeLong, Sarah P. Otto, Mary I. O'connor, Kyle E. Coblentz, Nikunj Goel, et al. 2022. An empiricist's guide to using ecological theory. *The American Naturalist* 199, 1 (2022), 1–20.
[25] T.R.G. Green and M. Petre. 1996. Usability Analysis of Visual Programming Environments: A 'Cognitive Dimensions' Framework. *Journal of Visual Languages & Computing* 7, 2 (1996), 131–174.
[26] Sam Greydanus and Chris Olah. 2019. The Paths Perspective on Value Learning. https://distill.pub/2019/paths-perspective-on-value-learning. *Distill* (2019).
[27] Avery Harrison, Hannah Smith, Taylyn Hulse, and Erin R. Ottmar. 2020. Spacing Out! Manipulating Spatial Features in Mathematical Expressions Affects Performance. *Journal of Numerical Cognition* 6, 2 (Sept. 2020), 186–203.
[28] Marijn Haverbeke, Adrian Heine, et al. [n. d.]. *CodeMirror*. https://codemirror.net/
[29] Andrew Head, Kyle Lo, Dongyeop Kang, Raymond Fok, Sam Skjonsberg, Daniel S. Weld, and Marti A. Hearst. 2021. Augmenting Scientific Papers with Just-in-Time, Position-Sensitive Definitions of Terms and Symbols. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, 1–18.
[30] Andrew Head, Amber Xie, and Marti A. Hearst. 2022. Math Augmentation: How Authors Enhance the Readability of Formulas using Novel Visual Design Practices. In *CHI Conference on Human Factors in Computing Systems*. ACM, 1–18.
[31] Sture Holm. 1979. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics* (1979), 65–70.
[32] HostMath - Online LaTeX formula editor and browser based math equation editor [n. d.]. https://www.hostmath.com/. Last accessed September 12, 2024.
[33] Yung Hsin I. and Fred Paas. 2015. Effects of Computer-Based Visual Representation on Mathematics Learning and Cognitive Load. *Journal of Educational Technology & Society* 18, 4 (2015), 70–77.
[34] Adobe Inc. [n. d.]. Adobe Illustrator.
[35] Ted Jacobson. [n. d.]. *Supplement Phy374 Spring 2006: Maxwell's equations*. https://www.physics.umd.edu/grt/taj/374a/maxwell374a.pdf Last accessed September 12, 2024.
[36] ST John and other contributors. 2023. *annotate-equations – Easily annotate math equations using TikZ*.
[37] Wenhui Kang, Jin Huang, Qingshan Tong, Qiang Fu, Feng Tian, and Guozhong Dai. 2024. MathAssist: A Handwritten Mathematical Expression Autocomplete Technique. *Proceedings of the 29th International Conference on Intelligent User Interfaces* (2024).
[38] Mary Beth Kery, Donghao Ren, Fred Hohman, Dominik Moritz, Kanit Wongsuphasawat, and Kayur Patel. 2020. mage: Fluid Moves Between Code and Graphical Work in Computational Notebooks. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. ACM, 140–151.
[39] Amy J. Ko and Brad A. Myers. 2006. Barista: An implementation framework for enabling new tools, interaction techniques and views in code editors. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 387–396.
[40] Philipp Kühl and Daniel Kirsch. [n. d.]. Detexify LaTeX handwritten symbol recognition. https://detexify.kirelabs.org/classify.html. Last accessed September 12, 2024.
[41] David Landy and Robert L. Goldstone. 2007. Formal notations are diagrams: Evidence from a production task. *Memory & Cognition* 35 (2007), 2033–2040.
[42] LaTeX - A document preparation system [n. d.]. https://www.latex-project.org/.
[43] Min-Jui Lee and Bing-Yu Chen. 2024. MaugVLink: Augmenting Mathematical Formulas with Visual Links. In *2024 IEEE 17th Pacific Visualization Conference (PacificVis)*. 337–342.
[44] Martin Leung, Renae Low, and John Sweller. 1997. Learning from equations or words. *Instructional Science* 25 (1997), 37–70.
[45] Yong Li, Shoaib Kamil, Alec Jacobson, and Yotam Gingold. 2022. HeartDown: Document Processor for Executable Linear Algebra Papers. In *SIGGRAPH Asia 2022 Conference Papers*. ACM, 1–8.
[46] Richard Lin, Rohit Ramesh, Nikhil Jain, Josephine Koe, Ryan Nuqui, Prabal Dutta, and Bjoern Hartmann. 2021. Weaving Schematics and Code: Interactive Visual Editing for Hardware Description Languages. In *The 34th Annual ACM Symposium on User Interface Software and Technology*. ACM, 1039–1049.
[47] LyX | LyX - The Document Processor [n. d.]. https://www.lyx.org/.
[48] Lars Madsen, Will Robertson, and Joseph Wright. 2014. *The mathtools package*.
[49] Manim Community [n. d.]. https://www.manim.community/.
[50] Mathcha - Online Math Editor [n. d.]. https://www.mathcha.io/. Last accessed September 12, 2024.
[51] Inc. Notion Labs. [n. d.]. Notion.
[52] Online Latex Editor | Tutorialspoint [n. d.]. https://www.tutorialspoint.com/online_latex_editor.php. Last accessed September 12, 2024.
[53] Online LaTeX Equation Editor [n. d.]. https://latexeditor.lagrida.com/. Last accessed September 12, 2024.
[54] Magnus Österholm. 2008. Do students need to learn how to use their mathematics textbooks? The case of reading comprehension. *NOMAD Nordic Studies in Mathematics Education* 13, 3 (2008), 53–73.
[55] Overleaf. [n. d.]. *About Us*. https://www.overleaf.com/about Last accessed 12 September 2024.
[56] The LaTeX Project. [n. d.]. LaTeX. https://www.latex-project.org
[57] Hugo Romat, Emmanuel Pietriga, Nathalie Henry-Riche, Ken Hinckley, and Caroline Appert. 2019. SpaceInk: Making Space for In-Context Annotations. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. ACM, 871–882.
[58] Franklin E. Satterthwaite. 1946. An approximate distribution of estimates of variance components. *Biometrics bulletin* 2, 6 (1946), 110–114.
[59] Charles Simonyi, Magnus Christerson, and Shane Clifford. 2006. Intentional software. In *Proceedings of the 21st annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications*. 451–464.
[60] StackExchange. [n. d.]. *MathOverflow*. https://mathoverflow.net/
[61] John Sweller. 2010. Element interactivity and intrinsic, extraneous, and germane cognitive load. *Educational psychology review* 22 (2010), 123–138.
[62] Eugene M. Taranta and Joseph J. LaViola. 2015. Math Boxes: A Pen-Based User Interface for Writing Difficult Mathematical Expressions. In *Proceedings of the 20th International Conference on Intelligent User Interfaces (IUI '15)*. Association for Computing Machinery, 87–96.
[63] Craig S. Tashman and W. Keith Edwards. 2011. LiquidText: a flexible, multitouch environment to support active reading. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. Association for Computing Machinery, 3285–3294.
[64] John W Tukey. 1949. Comparing individual means in the analysis of variance. *Biometrics* (1949), 99–114.
[65] Joris van der Hoeven. [n. d.]. TeXmacs.
[66] Markus Völter and Sascha Lisson. 2014. Supporting Diverse Notations in MPS' Projectional Editor. In *GEMOC@MoDELS*.
[67] Theresa E. Wege, Sophie Batchelor, Matthew Inglis, Honali Mistry, and Dirk Schlimm. 2020. Iconicity in Mathematical Notation: Commutativity and Symmetry. *Journal of Numerical Cognition* 6, 3 (2020), 378–392.
[68] Erik Weitnauer, David Landy, and Erin Ottmar. 2016. Graspable math: Towards dynamic algebra notations that support learners better than paper. In *2016 Future Technologies Conference (FTC)*. IEEE, 406–414.
[69] Frank Wilcoxon. 1992. Individual comparisons by ranking methods. In *Breakthroughs in Statistics: Methodology and Distribution*. Springer, 196–202.
[70] Yifan Wu, Joseph M. Hellerstein, and Arvind Satyanarayan. 2020. B2: Bridging Code and Interactive Visualization in Computational Notebooks. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. ACM, 152–165.
[71] Zhiyuan Wu, Jiening Li, Kevin Ma, Hita Kambhamettu, and Andrew Head. 2023. FFL: A Language and Live Runtime for Styling and Labeling Typeset Math Formulas. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. ACM, 1–16.
[72] Dongwook Yoon, Nicholas Chen, François Guimbretière, and Abigail Sellen. 2014. RichReview: blending ink, speech, and gesture to support collaborative document review. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. ACM, 481–490.
[73] Robert Zeleznik, Andrew Bragdon, Ferdi Adeputra, and Hsu-Sheng Ko. 2010. Hands-on math: a page-based multi-touch and pen desktop for technical work and problem solving. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. ACM, 17–26.

# A Controlled Study Tasks

Participants were asked to replicate formulas from existing scientific texts and teaching resources (some of which came from documents analyzed in Head et al. [30]). The tasks focused on three kinds of augmentations:

- *Color* ($T_C$): The participant colorized a formula to match a reference formula for cosine similarity from Carr [7].
- *Label* ($T_L$): The participant labeled a formula to match a reference formula describing the conversion of points between coordinate systems from Alexander [1].
- *Alignment* ($T_A$): The participant aligned a set of formulas corresponding to Maxwell's equations, inspired by a typical layout of the formulas (e.g. from Attwood [2], Jacobson [35]).

Prompts for all tasks appear in the supplemental material.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009