So, B., Smythies, C. A., & Tribble, C. M. (2026). An integrated user environment for RevBayes using RevSyntax and VS Code. *Bulletin of the Society of Systematic Biologists*. https://doi.org/10.18061/bssb.6970

Miscellany

An Integrated User Environment for RevBayes Using RevSyntax and VS Code

Benjamin So[1], Chiara A. Smythies[1], Carrie M. Tribble[1,2]

[1] Department of Biology, University of Washington, [2] Burke Museum of Natural History and Culture, University of Washington

Keywords: RevBayes, phylogenetics, VS Code, IDE

https://doi.org/10.18061/bssb.6970

Bulletin of the Society of Systematic Biologists

Abstract

RevBayes, a phylogenetic probabilistic graphical modeling software, presents prospective users with a steep learning curve. RevBayes lacks a custom integrated development environment (IDE) to facilitate writing and executing code, and as a resultmany users end up copying and pasting individual lines of code from a text editorinto a terminal window. This inefficient and error-prone process limits adoption of RevBayes for statistical phylogenetic analyses, impedes its use as a teaching tool, and creates unnecessary friction for existing users of the software. Here, we describe a user-friendly and efficient workflow for RevBayes analyses by leveraging VS Code, existing VS Code extensions, and our new development tool: RevSyntax. RevSyntax is a custom Visual Studio Code (VS Code) extension, developed to assistresearchers in statistical phylogenetic probabilistic modeling using RevBayes software. Our workflow allows researchers to use VS Code as an IDE for drafting, edit-ing, and executing Rev code, as well as for figure visualization and version control. By introducing simple additional features, RevSyntax smoothens the workflow forRevBayes users, demonstrating the value of customizing existing software for an improved user experience.

Phylogenies are branching diagrams that represent relationships in the Tree of Life as edges (branches) and nodes (leaves) connecting extant (or extinct) taxa and their hypothesized ancestors. The processes that mediate macroevolutionary dynamics (how organisms have evolved, migrated, speciated, and gone extinct) are most often modeled as evolving on a phylogeny (but see also recent advances in network-based methods, *e.g.*, Kong et al., 2025; Teo et al., 2025). `RevBayes` is a probabilistic graphical modeling software that allows users to program custom phylogenetic models via the `Rev` language (Höhna et al., 2016). `RevBayes` includes commands to specify Bayesian probabilistic graphical models for phylogenetics and to estimate parameter values through Markov chain Monte Carlo (MCMC). Some of the most popular `RevBayes` functionalities are phylogenetic estimation for extant and/or extinct organisms (*e.g.*, Ramírez et al., 2025; Rosario Petrucci et al., 2025), estimation of evolutionary rates such as diversification rates (*e.g.*, Ghanavi et al., 2025; Leong et al., 2025), custom specification of special-case ancestral state estimation models for discrete characters (*e.g.*, Augustijnen et al., 2024), and flexible implementation of continuous-trait evolution models (*e.g.*, Castro et al., 2024; May & Moore, 2020). One of the great strengths of `RevBayes` is its customization potential: users are able to rearrange existing model components to create novel methods that address previously unanswerable questions in systematic biology (*e.g.*, Mulvey et al., 2025; Pol et al., 2024; Swiston & Landis, 2025). However, this customization potential also creates a steep learning curve for new users and challenges during drafting, editing, and executing code (Charpentier & Wright, 2022; Tribble et al., 2022).

`RevBayes` does not have an integrated development environment (IDE)—software that serves as a combined interface to facilitate code development and execution (*e.g.*,`RStudio` for `R`). Without an IDE, most users edit `Rev` code as .Rev plain text files in a text editor program such as Visual Studio Code (VS Code), BBedit, or Sublime Text. These editors have built-in syntax highlighting for common coding languages like R, Python, Java, among others, but the niche nature of `Rev` has prevented development of `Rev`-specific tools for these programs. Thus, .Rev files appear as plain text, offering no visual structure. This makes spotting errors and reviewing code difficult and prone to error. While advanced users can modify their coding environment to highlight .Rev files using an `R` syntax highlighter, most `Rev` users do not know how to do this, and the result is often inconsistent with `RevBayes` specific syntax, which can lead to misleading syntax highlighting. These challenges reduce readability of complex scripts and can cause users to fre-

This is an open-access article distributed under the terms of the Creative Commons Attribution 4.0 International License (CCBY-4.0). View this license's legal deed at http://creativecommons.org/licenses/by/4.0 and legal code at http://creativecommons.org/licenses/by/4.0/legalcode for more information.

quently switch between application tabs, which can slow debugging.

After code has been written and edited, commands in .Rev files must still be executed in a command line shell such as bash, either by executing the entire .Rev file—useful for well-established code written by users familiar with the program, or by pasting individual lines into the shell, common for `RevBayes` learners and during troubleshooting. This approach is quite error prone, for example if a user fails to select the entirety of a desired section of code or accidentally skips a section in the script.

The output of `RevBayes` analyses are trace files that contain samples of parameter values, often with tens of thousands of samples or more. These trace files are then typically processed in external software, commonly the `R` package `RevGadgets` (Tribble et al., 2022). Thus, after the MCMC has completed in `RevBayes`, users often switch to `R` for processing and graphing results, requiring the use of another IDE such as RStudio.

Alternatively, users may turn to RStudio for drafting and executing `RevBayes` analyses via the R-Package `Revticulate` (Charpentier & Wright, 2022). However, `Revticulate` is limited in its ability to run longer analyses because the interface between `R` and `RevBayes` will time out (Charpentier & Wright, 2022), curtailing its use for many desired applications. Other users may implement `Rev` via Jupyter notebooks. A web-based tool to generate Rev scripts for basic analyses (`RevScripter`) is under development, but a full-scale graphical user interaface (GUI) for `RevBayes` poses a significant development challenge because of the broad range of possible models.

Version control is essential to open science. It allows users to track changes to files and work collaboratively with other researchers. Version control allows users to avoid accidentally deleting or overwriting hours (or days) of progress, document changes to improve transparency in code development, fork repositories and manage alternatives via branches, and enhances reproducibility. An effective workflow for computational research includes version control built-in to the development interface. As `RevBayes` analyses are typically implemented in a mix of various software, version control decisions rely on the user taking the initiative to add another step to an already complicated process. This barrier is enough to stop some users from implementing version control at the beginning of a project, or from implementing it at all.

The challenges in the status quo for executing `RevBayes` analyses, described above, motivated the creation of `RevSyntax`, a lightweight VS Code extension that improves the `RevBayes` scripting experience through two core features: syntax highlighting and “Send to Terminal.” VS Code is currently used by `RevBayes` developers for backend development (RevBayes Developers, 2021), so `RevSyntax` extends the value of software already used by the `RevBayes` community. Furthermore, VS Code can serve as an IDE for many different languages, which allows for easy integration between `RevBayes` and functionality in `R` for processing and plotting results. With `RevSyntax` complementing existing VS Code extensions, users can rely on VS Code as an IDE for all stages of the `RevBayes` workflow, from designing models, executing analyses, processing results, producing publication ready figures, and even eventually writing their manuscripts in `LaTeX`.

`RevSyntax` development

The `RevSyntax` extension was created using the VS Code Extension API to provide two core features: syntax highlighting and send to terminal features.

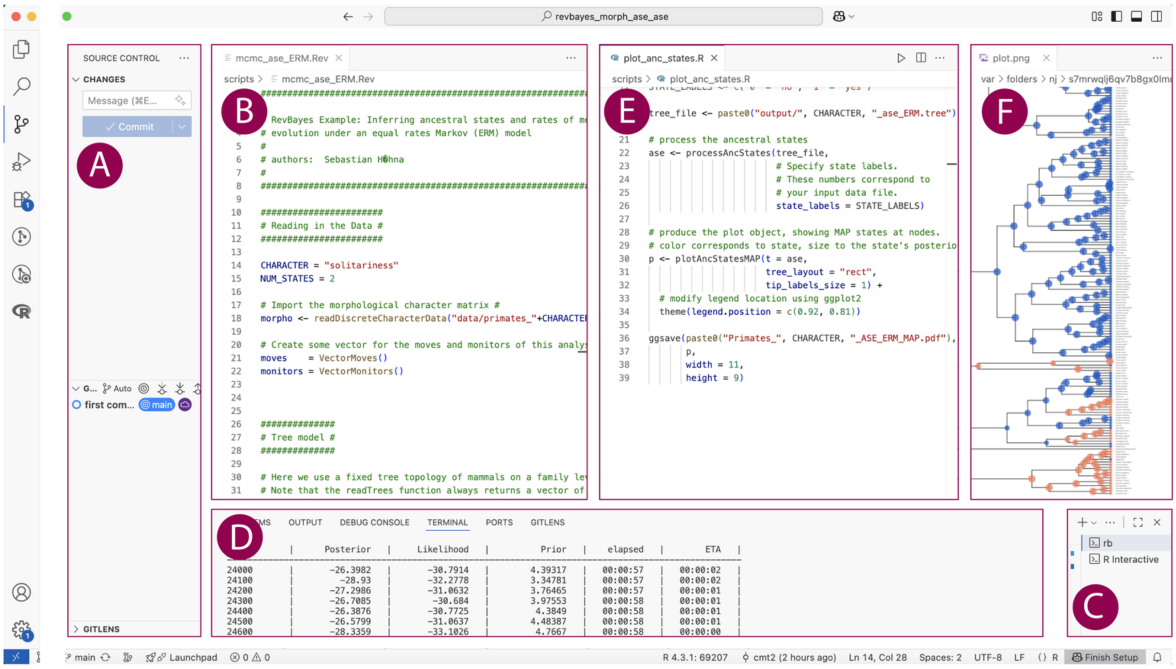
Within the extension, we implemented syntax highlighting through a custom TextMate Grammar. The grammar was defined in a single file (syntaxes/rev.tmLanguage.json) containing `Rev` language patterns that map keywords, operators, and comments to VS Code token scopes. The ability to recognize syntax grammar enables colorized highlighting of commands, arguments, and comments in .Rev files, improving readability and differentiation of code patterns. The color theme of `RevSyntax` follows that of the user’s selection of VS Code theme, and thus can be changed based on user preference.

We enabled “Send to Terminal” capability through a JavaScript function that uses the VS Code API. The API provides access to VS Code-specific functionality, including the location of the cursor, the selected text, and terminal access. This function is included in the extension in src/extension.js.

The final extension was packaged and published to the VS Code Marketplace for easy installation. `RevSyntax` is available via the extensions tab in Visual Studio Code and may be installed with the install button for immediate access. `RevSyntax` source code is available on GitHub (https://github.com/tribblelab/revsyntax).

Configuration

`RevSyntax` enables a user-friendly and efficient workflow for `RevBayes` analyses in VS Code. To implement this workflow, we first install `RevBayes`. We point readers to the `RevBayes` website for information on installing and configuring `RevBayes` for use: https://revbayes.github.io/. Then we configure VS Code for version control via git, `R` code syntax highlighting and execution for plotting results in the `RevGadgets` `R` package, and `Rev` code syntax highlighting and integration. These tasks are accomplished via specific extensions, which must be installed separately after installing VS Code: GitLens, `R`, `R` Syntax, and our newly developed `RevSyntax`. Extensions may be installed via the Extensions Marketplace, accessible online or via the Extensions sidebar of VS Code. We recommend opening an entire project directory in VS Code (rather than individual code files) via the drop-down menu `File > Open Folder`. VS Code contains an Explorer panel, which allows easy navigation within the directory, especially when toggling back and forth between multiple files.



```
###########################################################################

 RevBayes Example: Inferring ancestral states and rates of mo
# evolution under an equal rates Markov (ERM) model
#
# authors:  Sebastian H�hna
#
###########################################################################

#######################
# Reading in the Data #
#######################

CHARACTER = "solitariness"
NUM_STATES = 2

# Import the morphological character matrix #
morpho <- readDiscreteCharacterData("data/primates_"+CHARACTER

# Create some vector for the moves and monitors of this analys
moves    = VectorMoves()
monitors = VectorMonitors()



##############
# Tree model #
##############

# Here we use a fixed tree topology of mammals on a family lev
# Note that the readTrees function always returns a vector of
```

```
tree_file <- paste0("output/", CHARACTER, "_ase_ERM.tree")

# process the ancestral states
ase <- processAncStates(tree_file,
                        # Specify state labels.
                        # These numbers correspond to
                        # your input data file.
                        state_labels = STATE_LABELS)

# produce the plot object, showing MAP states at nodes.
# color corresponds to state, size to the state's posterio
p <- plotAncStatesMAP(t = ase,
                      tree_layout = "rect",
                      tip_labels_size = 1) +
  # modify legend location using ggplot2
  theme(legend.position = c(0.92, 0.81))

ggsave(paste0("Primates_", CHARACTER, "_ASE_ERM_MAP.pdf"),
       p,
       width = 11,
       height = 9)
```

```
          |   Posterior   |   Likelihood   |      Prior   |   elapsed   |      ETA   |
-----------------------------------------------------------------------------------------
24000     |    -26.3982   |    -30.7914    |    4.39317   |  00:00:57   |  00:00:02  |
24100     |      -28.93   |    -32.2778    |    3.34781   |  00:00:57   |  00:00:02  |
24200     |    -27.2986   |    -31.0632    |    3.76465   |  00:00:57   |  00:00:01  |
24300     |    -26.7085   |     -30.684    |    3.97553   |  00:00:58   |  00:00:01  |
24400     |    -26.3876   |    -30.7725    |     4.3849   |  00:00:58   |  00:00:01  |
24500     |    -26.5799   |    -31.0637    |    4.48387   |  00:00:58   |  00:00:01  |
24600     |    -28.3359   |    -33.1026    |     4.7667   |  00:00:58   |  00:00:00  |
```

Figure 1. VS Code configured for a start-to-finish RevBayes analysis.

The version control panel (A) can be used to track changes and work collaboratively using `git` for code development. Next, users can draft and edit Rev code (B). Users can open a terminal window via panel (C), then open RevBayes and send code to run an analysis, visualized in panel (D). After the analysis is complete, users can execute R code in panel (E) and visualize the results of the analysis using `RevGadgets` in panel (F).

Use Case

We demonstrate our VS Code workflow for RevBayes by working through a tutorial from the RevBayes website: "Discrete morphology—Ancestral State Estimation (ASE)" (Höhna, 2022). We also illustrate this workflow in Figure 1, and refer to the lettered panels throughout the text.

After configuring VS Code as described in the above Configuration section, we open the tutorial directory in VS Code using: `File > Open Folder > REVBAYES_MORPH_ASE`.

VS Code integrates version control via GitHub in the source control panel (Fig. 1A). To enable version control, we select "Publish to GitHub" in this control panel. Users who have not previously set up version control via GitHub in VS Code will need to sign into their GitHub account when prompted. This will create a copy of all files on GitHub, and all future changes will be tracked using `git`.

Our new `RevSyntax` extension permits proper syntax highlighting for `Rev` code, as visualized in Figure 1B. VS Code's explorer panel visualizes directory structure and files, which facilitates navigating between files. To proceed with the `RevBayes` tutorial, we open the analysis script `mcmc_ase_ERM.Rev` using the explorer panel. If `RevSyntax` is correctly installed, VS Code will recognize the .Rev file extension and apply appropriate syntax highlighting. Users may add and edit code in this editing panel (B).

`RevSyntax`'s "Send to Terminal" feature allows users to execute code without copying and pasting into a separate program. VS Code includes integrated Terminal(s) for running code in various languages (Fig. 1C). To execute the tutorial Rev code, we first open a Terminal panel running bash and then start RevBayes (most often by typing `> rb`). For this to work, as described above in the Configuration section, the user must initiate RevBayes before executing code. If users do not initiate RevBayes before, VS Code will automatically attempt to execute the code in bash, leading to errors. VS Code will set the working directory to the project directory (REVBAYES_MORPH_ASE) by default. In our example workflow, this is the base directory of the ASE tutorial.

Users may run code line-by-line or execute highlighted chunks at a time using `Cmd + Return` or `Ctrl + Enter` depending on the operating system. The selected code will be executed in the Terminal session running Rev (Fig. 1D).

VS Code can be used to write, edit, and execute `R` code similarly to Rev code, allowing users to accomplish all steps of the analysis process in one program. `R` functionality is enabled via the extensions installed in the Configuration section above. `RevGadgets` is an `R` package for visualizing the results of RevBayes analyses. After running the RevBayes analysis, we use the explorer panel to open the `R` visualization code via `scripts > plot_anc_states.R` (Fig. 1E). Users who wish to view multiple files alongside each other (for example, the analysis .Rev file and the visualization .R file, as seen in Figure 1B,E) can open a secondary side bar using the layout buttons in the upper right corner. VS Code will automatically initiate an `R` session in the lower terminal panel when code is executed (also using `Cmd + Return` or `Ctrl + Enter`), unlike for Rev code. Much like RStudio, VS Code can visualize figures integrated into the interface before saving final versions to the project directory (Fig. 1F).

All modifications to files in directory are tracked automatically through version control (as long as you are working within a local git repository). These changes may then be committed and pushed to the remote repository on GitHub using the source control panel.

Conclusions

`RevBayes` facilitates innovation in statistical phylogenetic analysis, including phylogenetic reconstruction, trait evolution, and their joint inference. Users must specify model components (rates, prior distributions, MCMC moves, among others) using `Rev` code, which requires a solid understanding of Bayesian statistics and the fundamentals of macroevolutionary models. As a result, this requirement imposes a steep learning curve on new users, which is steepened by the lack of a cohesive IDE to guide users through all stages of the analysis process. `RevBayes` also has great potential as a teaching tool for students interested in the nitty-gritty of Bayesian phylogenetic modeling, but without an IDE students are left struggling with an error-prone workflow that requires switching across multiple programs during a single analysis. To address these concerns, we developed `RevSyntax`, which enables a user-friendly workflow for `RevBayes` analyses in VS Code. Our workflow capitalizes on VS Code as the existing IDE of choice for `RevBayes` back-end development and the flexibility of VS Code extensions in setting up custom features such as `Rev` coding. Proper syntax highlighting via `RevSyntax` also expands `RevBayes`' utility as a teaching tool, because the syntax highlights emphasize `Rev` grammar, one of the important considerations in learning a new coding language.

In the era of technological advancements, it is often assumed that the performance of technology is directly proportional to the complexity of the design and length of the code. However, this paper shows how the usage a helpful software extension can drastically improve the progression of scientific research. `RevSyntax` and our associated workflow showcase how small tools can produce meaningful improvements in scientific workflows. By adding basic IDE features for `RevBayes` scripting, the extension simplifies the process of refining and executing phylogenetic probabilistic graphical models.

……

Author Contributions

CMT and CS conceived of the project. BS wrote and implemented the code. CS and CMT provided feedback on user experience. All authors contributed to writing and editing the manuscript.

Conflict of Interest Statement

The authors declare no conflict of interest.

Data availability

All code is available on GitHub at https://github.com/tribblelab/revsyntax.

Acknowledgments

CMT received funding from the UW College of Arts and Sciences. The authors thank Bryan Carstens, an anonymous reviewer, Lauren Frankel, Michael R. May, and Ixchel González Ramírez for feedback on early versions of `RevSyntax` and the manuscript. Basanta Khakurel contributed to RevSyntax code and bug fixes. Jenna T. B. Ekwealor provided feedback on figures.

Submitted: October 24, 2025 EDT. Accepted: February 02, 2026 EDT. Published: March 09, 2026 EDT.

References

Augustijnen, H., Bätscher, L., Cesanek, M., Chkhartishvili, T., Dincă, V., Iankoshvili, G., Ogawa, K., Vila, R., Klopfstein, S., Vos, J. M. de, & Lucek, K. (2024). A macroevolutionary role for chromosomal fusion and fission in *Erebia* butterflies. *Science Advances*, *10*(16), eadl0989. https://doi.org/10.1126/sciadv.adl0989

Castro, M. G., Amado, T. F., & Olalla-Tárraga, M. Á. (2024). Correlated evolution between body size and echolocation in bats (order Chiroptera). *BMC Ecology and Evolution*, *24*(1), 44. https://doi.org/10.1186/s12862-024-02231-4

Charpentier, C. P., & Wright, A. M. (2022). Revticulate: An R framework for interaction with RevBayes. *Methods in Ecology and Evolution*, *13*(6), 1177–1184. https://doi.org/10.1111/2041-210X.13852

Ghanavi, H. R., Chazot, N., Sanmartín, I., Murillo-Ramos, L., Duchêne, S., Sihvonen, P., Brehm, G., & Wahlberg, N. (2025). Region-specific diversification dynamics and biogeographic history of one of the most diverse families of insects. *Systematic Entomology*, *50*(1), 206–220. https://doi.org/10.1111/syen.12651

Höhna, S. (2022). *Discrete morphology — ancestral state estimation*. RevBayes Tutorials. https://revbayes.github.io/tutorials/morph_ase/ase.html

Höhna, S., Landis, M. J., Heath, T. A., Boussau, B., Lartillot, N., Moore, B. R., Huelsenbeck, J. P., & Ronquist, F. (2016). RevBayes: Bayesian Phylogenetic Inference Using Graphical Models and an Interactive Model-Specification Language. *Systematic Biology*, *65*(4), 726–736. https://doi.org/10.1093/sysbio/syw021

Kong, S., Solís-Lemus, C., & Tiley, G. P. (2025). Phylogenetic networks empower biodiversity research. *Proceedings of the National Academy of Sciences*, *122*(31), e2410934122. https://doi.org/10.1073/pnas.2410934122

Leong, J. V., Matos-Maraví, P., Núñez, R., Nunes, R., Liang, W., Braby, M. F., Doleck, T., Aduse-Poku, K., Inayoshi, Y., Hsu, Y.-F., Wahlberg, N., Peggie, D., Mohagan, A. B., Mohagan, D. P., Genaro, J. A., Perez-Asso, A. R., Kunte, K., Martins, D. J., Sáfián, S., … Lohman, D. J. (2025). Around the World in 26 Million Years: Diversification and Biogeography of Pantropical Grass-Yellow *Eurema* Butterflies (Pieridae: Coliadinae). *Journal of Biogeography*, *52*(5), e15107. https://doi.org/10.1111/jbi.15107

May, M. R., & Moore, B. R. (2020). A Bayesian Approach for Inferring the Impact of a Discrete Character on Rates of Continuous-Character Evolution in the Presence of Background-Rate Variation. *Systematic Biology*, *69*(3), 530–544. https://doi.org/10.1093/sysbio/syz069

Mulvey, L. P. A., May, M. R., Brown, J. M., Höhna, S., Wright, A. M., & Warnock, R. C. M. (2025). Assessing the Adequacy of Morphological Models Using Posterior Predictive Simulations. *Systematic Biology*, *74*(1), 34–52. https://doi.org/10.1093/sysbio/syae055

Pol, D., Baiano, M. A., Černý, D., Novas, F. E., Cerda, I. A., & Pittman, M. (2024). A new abelisaurid dinosaur from the end Cretaceous of Patagonia and evolutionary rates among the Ceratosauria. *Cladistics*, *40*(3), 307–356. https://doi.org/10.1111/cla.12583

Ramírez, I. S. G., Deanna, R., & Smith, S. D. (2025). *Late Cretaceous origins for major nightshade lineages from total evidence timetree analysis*. bioRxiv. https://doi.org/10.1101/2025.07.22.666174

RevBayes Developers. (2021). *Setting up an IDE for RevBayes*. https://revbayes.github.io/developer/setup/

Rosario Petrucci, B. do, May, M. R., & Heath, T. A. (2025). Fossils improve extinction-rate estimates under state-dependent diversification models. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *380*(1919), 20230313. https://doi.org/10.1098/rstb.2023.0313

Swiston, S. K., & Landis, M. J. (2025). Testing Relationships between Multiple Regional Features and Biogeographic Processes of Speciation, Extinction, and Dispersal. *Systematic Biology*, *74*(2), 282–300. https://doi.org/10.1093/sysbio/syae062

Teo, B., Bastide, P., & Ané, C. (2025). Leveraging graphical model techniques to study evolution on phylogenetic networks. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *380*(1919), 20230310. https://doi.org/10.1098/rstb.2023.0310

Tribble, C. M., Freyman, W. A., Landis, M. J., Lim, J. Y., Barido-Sottani, J., Kopperud, B. T., Höhna, S., & May, M. R. (2022). RevGadgets: An R package for visualizing Bayesian phylogenetic analyses from RevBayes. *Methods in Ecology and Evolution*, *13*(2), 314–323. https://doi.org/10.1111/2041-210X.13750