We thank the reviewers for their insightful and invaluable comments about our work. We have made the advised revisions as outlined below.

Overall the revision is longer by just a few sentences.

Changes suggested by the reviewers to our package `vivid` have been addressed and an updated version of `vivid` is now available on CRAN.

Comments from Reviewer 1

General:

> The paper presents a new R package called vivid that provides tools for computing and visualizing variable importance and variable interaction measures for machine learning models. The functionality of the package allows users to easily compute importance and interaction measures for different types of models. These measures can then be compared using the provided visualizations. Functions are also provided for creating individual conditional expectation curves and partial dependence plots for sets of predictor variables from a model.
>
> A strength of the package is the suite of tools provided for easily creating visualizations that provide different perspectives of variable importance and variable interactions. The package provides beneficial tools for visualizing variable relationships in machine learning models such as the generalized pairs partial dependence plots and graphical representations of variable importances/interactions. The package allows the user access to various methods for computing variable importance that are commonly used.
>
> A shortcoming of the package is a lack of methods for variable importance and variable interactions that account for correlation between variables. It has become well accepted that correlation between variables can lead to biased variable importance measures including partial dependence plots that rely on a permute-and-predict technique. The authors briefly mention this in the discussion and the fact that this is an ongoing area for research. However, adding warnings about the negative effects of correlated variable and/or incorporating additional tools such as accumulated local effects plots or conditional random forest variable importance that attempt to address the correlation concern would greatly strengthen the package.

Thank you for your kind words about our paper. We appreciate the positive feedback. In relation to accounting for correlation between variables, we agree that there is a deficit for variable importance that accounts for correlations. This is true of the literature in general. In `vivid` we provide a GPDP, where correlations are evident from the plot of the data and the PDP with convex hull. As the reviewer mentioned, incorporating accumulated local effects into our GDPD and ZPDP plots is a useful avenue for future updates of our package. We have added more details and warnings about any potential correlations leading to biased variable importance/interaction measures to the discussion section.

> Page 1, Paragraph 4: In addition to discussing model agnostic and model specific VIVI methods, I would define local versus global methods (i.e., one prediction versus entire model). For example, ICE curves and LIME are a local methods while PDP curves and permutation importance are global methods.

We have added some commentary to this section (page 1, paragraph 4) discussing global and local methods as well as adding an extra column to Table 1 stating which packages incorporate global or local methods. We have also added a comment (page 3, paragraph 3) stating how `vivid` can be used on subsets of data to provide a more local-based variable importance measure.

> Page 1, Paragraph 4: For the packages that compute the same metrics (i.e., partial dependence plots), do they implement the computation in the same way? I think it could be valuable to discuss if there are differences (and possibly what the differences are).

Additional commentary has been added to Section 1 (Introduction) page 3, paragraphs 1-2 discussing implementations of variable importance and partial dependence in different R packages.

> Page 2, Table 1: I think this table is very valuable. It is useful to have all these packages listed in one place. To improve viewing, I would consider organizing the table further by breaking the provided information into additional columns such as: variable importance, variable interactions, or both; visualization types provided; model-agnostic methods, model-specific methods, or both; etc. It could also be nice to use these additional groupings to order the rows of the table based on similar attributes in some manner.

To improve reading we have amended Table 1 to include variable importance/interactions, visualization types, and global/local methods. As suggested, we have also grouped together packages by similar attributes. In the pdf version of our work, we group packages by their ability to compute either model specific or model agnostic variable importance/interaction measures. In the html version, we allow a reader to sort alphabetically (by clicking on a column header) or to search for a specific entry using the search box (for example, searching for 'VInt' will filter the table to only display packages that provide variable interaction measures).

> Page 3, Paragraph 3: I think this paragraph could be made stronger by highlighting the contributions of vivid that make it different from the previously developed packages for visualizing variable importance.

We have added some commentary to page 3, paragraph 3 highlighting the contributions of vivid that differ from the previously developed packages. These include mentioning seriation, filtering, and clustering techniques.

> Page 4, Paragraph 2: The randomForest package will return a variable importance measure if the 'importance' measure is set to FALSE, but only one type of variable importance is returned instead of multiple measures. Can you clarify the need for setting 'importance' to TRUE when using vivid? Also, when the 'vivi' function is applied later on the page, the parameter of 'importanceType' is set to 'agnostic'. This also raises the question of why 'importance = TRUE' needs to be specified when training the random forest since seemingly vivid computes a separate measure using permutation importance.

We agree that the discussion regarding random forest importance was not fully stated. We have now explained this point in manuscript (on page 4, paragraph 1) and have removed the `importance = TRUE` argument from the code on page 4. We have also added details clarifying the `importanceType` argument on page 5 paragraph 3.

> Page 4, Paragraph 2: Random forest variable importance is typically computed on out-of-bag observations. Is this true for xgboost as well? What about model types that do not have an out-of-bag observations? How do these compare?

`xgboost` has a number of variable importance measures that do not use OOB observations. However, comparing different methods of variable importance across multiple different fits is outside the scope of our paper. Instead, we are using the agnostic importance measure to to better compare fits and not the different importance measures. A comment has been added to page 6 paragraph 1 to clarify that the agnostic importance measure is used for the `xgboost` fit.

> Page 4-5, Section 'Calculating VIVI': It is a bit unclear from this section which model types can be used with the vivi function. I think I was able to figure it out by piecing information together from multiple locations (e.g., paragraph 2 on page 5 and the explanation of the 'CVpredictfun' function). However, I think it would help readers if you could state explicitly which model types can be used with vivi, which can be used out-of-the-box and which require the use of the 'CVpredictfun'.

In `vivid`, we use the predict function from the `condvis2` package [Hurley et al., 2022] for the agnostic importance and interaction scores. Consequently, `vivid` can be used out-of-the-box with any model type that works with `condvis2` predict. We have amended this section (page 5, paragraph 2) to explicitly state which packages work out-of-the-box with `vivid` and which require the use of `CVpredictfun`.

Page 4, Paragraph 5: It is stated that variables not used by the model but included in the input dataset to the 'vivi' function will return a variable importance of 0. Since a variable used by the model could have a variable importance of 0, I would recommend adjusting this, so that NAs are returned for variables not used in the model.

This is a reasonable suggestion, and the sentence referred to is now deleted. However the agnostic variable importance and interaction (from the `flashlight` package) still return zero for variables included in the dataset and not in the model. This is the behaviour inherited from `flashlight` and we have added a sentence to page 5, paragraph 6 to clarify this. Changing this behaviour would require us to access the variables used in the fit from the fit, which is not always possible, as many fits do not provide this information.

Page 5, Paragraph 1: Permutation variable importance typically involves multiple replications of permutations to account for variability. How is this accounted for in vivid? Is the user able to specify the number of permutation replications to implement, and can the user view the variation across replicates?

By default we permute 4 times and the average is taken. We have added an option to our package (which is now available on CRAN) to allow a user to select the desired number of permutations. A comment has been added to page 5 paragraph 1 to reflect this. Allowing users to view the variation across replicates is a great suggestion and will be implemented in the next version of our package. We have added a comment addressing this to the discussion section (page 15, paragraph 3).

Page 5, Paragraph 1: It is mentioned that root mean square error is used for the model error score. Typically, this measure is only used with regression. What metric is used for computation of permutation importance in classification? Can the users specify which metric they would like to use (e.g., AUC or accuracy)?

In `vivid`, when `importanceType = 'agnostic'`, importance works for numeric and numeric binary response only. In the case of binary classification, the response is converted to numeric and the root mean square error is calculated. To obtain different metrics, embedded variable importance methods would have to be used. Beyond this, users can not specify different metrics, however providing these options is a great avenue for future work. A comment reflecting this has been added to the discussion section.

Page 5, Paragraph 2: A possible concern I have with the computation and comparison of model-specific variable importances is whether it makes sense to directly compare them. In particular, I am thinking about how random forest variable importance is often computed on out-of-bag observations (as is the case with the randomForest R package permutation based variable importances). What data are the model-specific variable importances computed on with the other packages (i.e., ranger, mar, mlr3, and tidymodels)? If the variable importances are computed on training data instead of out-of-bag data, the values would not be directly comparable. Is this addressed in the package? Additionally, model-specific variable importances measure different quantities (e.g., gain with xgboost and %IncMSE with randomForest). The scale of the importances would not be directly comparable. I would recommend that the authors address this at least in the form of caution for users.

We agree that using model-specific variable importance will result in importance measures that are not directly comparable for different model fits (a warning about which we have added to page 5, paragraph 3). However, comparing model specific scores could be useful when comparing the same fitting procedure evaluated using different parameters. In our work we use the agnostic VImp method for both model fits to allow for a direct comparison. A comment clarifying this has been added to page 4 paragraph 1.

Page 6, Paragraph 1: Would it be possible to parallelize the code for computing Friedman's H-statistic to increase computation time?

We could in future explore this option. However, currently `vivid` uses the `flashlight` package to compute Friedman's H-statistic.

Page 6, Paragraph 1: In the sentence, "Each of the models were built using their default settings and, for each fit, the agnostic VImp was measured." Should it say "the agnostic VInt" or "VImp and VInt" since the paragraph starts out by describing the computation burden of Friedman's H-Statistic for interactions?

Thank you for pointing out this lack of clarity. The sentence should read 'the agnostic VImp and VInt'. We have fixed this sentence.

Page 6, Paragraph 3: What is an example of a case where the vivid matrix would not be symmetric?

`vivid` can be used to display interaction measures that are not symmetric. For example, the interaction measures from the `randomForestExplainer` package are asymmetric and could be visualised using the heamap from `vivid`. A comment has been added to page 7, paragraph 1.

Page 7, Paragraph 3: What is the ‘angle‘ input parameter in ‘viviHeatmap‘?

The angle argument controls the angle of the x-axis labels on the top of the heatmap. We have added a sentence reflecting this to page 8 paragraph 1.

Page 7, Paragraph 3: I was not able to find a definition for ‘lstat’. What is ‘lstat’?

We have added the definition for ‘lstat’ (which is the percentage of lower status of the population) to this section.

Page 7, Paragraph 4: It would be useful to describe the benefits or reasons for using different graph layouts. Do certain layouts help with cognitive interpretation of the results? Does the layout recommendation vary across scenarios?

We have added some commentary to the final paragraph of page 8 and continuing onto page 9 discussing the cognitive interpretation of using different graph layouts.

Page 8, Paragraph 1: Can you also filter a graph based on a variable importance threshold in addition to the ‘intThreshold‘? I see that a variable of ‘impThresh‘ is defined later on the page.

Currently we only provide a function argument in `vivid` to filter interactions below a certain selected threshold. Providing an option to filter variable importance from the network plot could result in a visualisation with edges not being connected to any nodes. In our work, the variable ‘impThresh’ is used in conjunction with ‘intThreshold’ to select variables with importance and interaction values in the top 10% and filter out the remaining variables.

Page 9, Figure 3: Would it be better to not fill the clusters will color since it affects the colors of the nodes and edges in the graph?

Thank you for this great suggestion. We have amended the order of how the clustered network is drawn in our package (new version is now available on CRAN). Now, the cluster is drawn first and the nodes/edges are drawn above the cluster, which eliminates any discolouration effect.

Page 9, Figure 3: How is the number of cluster selected in the hierarchical clustering? Can users adjust the number of clusters?

If the `igraph` package is used, then the number of clusters is controlled by the relevant `igraph` function. However, in the example shown (code on page 9 with corresponding

Figure 3(b) on page 10), we perform hierarchical clustering using the `hclust` function. Then, the number of clusters can be selected via the `k` argument from the `cutree` function. This function cuts trees into groups of data, where `k` is the desired number of groups. In our example in Figure 3, we select three groups. A comment clarifying this has been added to page 9, paragraph 2.

> Page 10, Figure 4: It would be nice if the plots were ordered by variable importance

In our work we state that the ordering of the PDPs is taken from the ordering of variables in the data set, or may be specified via the vars argument. However, we fail to mention that the ordering of Figure 4 is taken directly from our seriated `vivid`-matrix. This ordering allows us to plot the variables that have the greatest influence on the response. A comment clarifying this point has been added to page 11 paragraph 1.

> Page 10, Paragraph 3: Are the scatterplots on the lower diagonal of the observed data? A quick clarification would be helpful.

Yes, the scatterplots are of the observed data. A comment reflecting this has been added on page 11, paragraph 3.

> Page 11, Figure 5: Again, could you order the plots by variable importance?

As with the previous point regarding the ordering, we order the GPDP by selecting the first five variables from our `vivid` matrix. This returns the top five most influential variables. A comment clarifying this has been added to page 11, paragraph 4.

Page 11, Paragraph 2: Reference for "graph Eulerian"?

We have added a reference for the classical algorithm for constructing Eulerian paths to page 12, paragraph 4.

Page 11, Paragraph 2: In the sentence, "In our adaption...", insert "which is" before "layout".

The relevant sentence has been fixed.

Page 11, Paragraph 2: A small semantic comment – since importance has a specific meaning in this paper, adjust the phrase "...the most important interacting variables..." to use a word different than 'important' such as 'the variables with the largest interaction values'.

We have changed the sentence in question (now on page 12, paragraph 4) to remove any ambiguity concerning the word 'important'.

Page 12, Paragraph 2: It might be beneficial to define Eulerians for readers who are not familiar with the term.

We have added a brief definition of Eulerians to page 12, paragraph 4.

Page 13, Figure 7b: On page 12, it says that the second zen-path consists of four unconnected paths. However, at first glance, there only appears to be three unconnected paths. I'm guessing that the bottom path is two separate paths based on the axis labels, but it would be nice for the path with ptratio, lstat, and tax to be moved to the right, so that it is visually separated from the path above.

Our `pdpZen` function calls a function named `zenplot` from the `zenplots` package, which is responsible for determining the plot's layout. The `pdpZen` function also enables the passing of arguments to the `zenplots` function, which encompasses various layout options. However, to change the layout would involve some manipulation of the `zenplot` function arguments. To maintain clarity in the paper's code, the layout has been kept unchanged. Nevertheless, a remark has been added to page 14, paragraph 1, stating that alternative layouts can be achieved using `zenplots` function arguments.

> Page 14, Paragraph 1: Another semantic comment – Change the phrase in the last sentence "...the most important aspects" to use a word other than important.

We have changed the sentence to use another word other than 'important'

> Page 14, Paragraph 2: Some other areas to consider addressing in the future research section: Computations and visualizations of higher order interactions; interactivity with visualizations; multivariate response variables; scaling to larger data (sample size and predictions) - how to best account for these in terms of computation time and meaningful visualizations?

Thank you for these suggestions. we have incorporated your ideas into our discussion about future work.

Code:

> Overall, I found the code easy to use and understand.

Thank you.

> When running the code provided with the paper, I had to install several packages (e.g., zenplots and sp) that were required to generate some of the visualizations from vivid. I saw in the description file that these packages were listed under 'Suggests' instead of 'Imports'. I would recommend moving these to 'Imports' since they are required to implement functions in vivid. 'Suggests' is good for packages that are only called in vignettes and examples.

To install the `zenplots` package, you must first install the `graph` package, which is found on BioConductor and not CRAN. As a result, `zenplots` will not install automatically when installing `vivid` and has to remain in 'Suggests'. The `sp` package has been moved to 'Imports'.

In the documentation for the ‘vivi‘ function, there is some ambiguity with the input parameter of importanceType. The definition makes it seem like model-specific variable importance measures can be used with random forests. However, the paper alludes to model-specific variable importance measures for other model types. Can vivid return model-specific variable importances for other models? Furthermore, the description for importanceType says "One of either "%IncMSE" or "IncNodePurity" for use with randomForest." These are values for randomForest regression models. The metrics used with the randomForest classification models are different. How does a user specify an embedded variable importance with a randomForest classification model? Also, what importance measure is calculated if ‘importanceType‘ is not specified? The function shows 'NULL' as the default. I would recommend clarifying this in the documentation.

We agree that there was some ambiguity related to this function argument in our documentation. This has been clarified in our current version of our package on CRAN. Commentary clarifying the `importanceType` argument has been added to page 5 paragraph 3. Additionally, we have added examples of using embedded variable importance with the `vivi` function to our package documentation and to our vignette, available at: https://alaninglis.github.io/vivid/articles/vividVignette.html

I'd recommend including more details (or references to the details) in the documentation of how the variable importance values and variable interaction measures are being computed. I think it is important to be clear about how these computations are done, so that users can better understand the capabilities of these techniques.

We have included more details (including references to the used methods) in our package documentation and is now available on CRAN.

1 Comments from Reviewer 2:

General:

> Dear authors of vivid, thank you for your great contribution to the R community. I really enjoyed reading through your thoughtful paper. Below, I've noted down a couple of small issues I've found in the paper and a suggestion for potential future improvement. When looking at the code, I'm afraid there is more room for improvement there. Before we get to this though, let's first look at the paper, and what I like so much about it.
>
> vivid introduces a unified way of visualizing variable importance and variable interactions for predictive models. To this end, a standard way of calculating those metrics is defined and visualizations based on 'ggplot2' are offered. This is completed with a set of utility functions. In the paper, the usage of the package is introduced using a couple of models for a well-known demo data set. In particular, I've enjoyed the following points:
>
> 1. Great idea to describe motivation behind package and to explain design choices.
> 2. Well done to introduce terms you are using, even if they might be well known, so that you make sure everybody understands the same thing.
> 3. Fantastic overview of other packages; suggestion: some vertical padding the table might make it easier on the eyes (or gridlines).
> 4. Really thought of everything, even reordering variables.
>
> For future work, I would suggest to consider the following: It would seem that 'vivi()' calculates its values from the training data. Yet https://christophm.github.io/interpretable-ml-book/feature-importance.html#feature-importance-data makes a case for feature importance being meaningful for both training and test data. Perhaps vivid should offer that as a possibility?

Thank you for your kind words about our paper. We appreciate the positive feedback.

We have added some vertical padding to Table 1 to improve reading.

In relation to computing the importance on the training or test sets. As the reviewer notes, Molnar makes an argument for both cases in Section 8.5.2 of his book *Interpretable Machine Learning*. As we use relatively small data sets in our examples, the test data itself is small. Since the goal of our visualisations is to understand the model fit rather than explicitly perform variable selection, we decided to use the training data to calculate the VImps in our examples. Though, as all vivid functions have the dataset as input, calculations and visualisations could easily be swapped for the test data.

> Table 2 has the value for the Description field for ‘CVpredictfun‘ missing. This led to all other Description field values up to ‘pdpVars‘ to be wrong (shifted by one) and the Description value of ‘pdpVars‘ to occur twice.

Table 2 has now been fixed to display the correct entries.

> On page 7 where the heatmap is discussed, you give explanations for ‘crim’ and ‘nox’ but not for their interaction ‘lstat’. (It is then given on page 11 in the discussion of the ‘pdpPairs’ plot matrix.)

A description of ‘lstat’ has been added to this section.

Code:

> Codebase not clean. In general, it would be expected that (at least) the master branch of a repository is in pristine order. This means there should be no commented out code, no superfluous files or directories, no build artefacts. This is not the case. In particular, there is code, rendered figures, cached files and the like, not relevant for the package. Mostly I think those are journal articles. They do not belong into the package repository, they should all live in one or more separate repositories. Right now, cloning the ‘vivid‘ repository would introduce a lot of binary files and data files. In my company, container security would stop any container trying to install vivid because of this, making it unusable in an enterprise context.

We have cleaned up the package repository and removed all files not related to the package. Any files related to journal articles are now located in a separate repository.

> Tests. The purpose of unit tests or even integration tests is to ensure that a piece of software fulfils its purpose or plys nicely with other parts of software. Your tests only check that the correct class is returned, and not the results of the operations carried out. Again, this would make your package unusable in an enterprise context.

We have added more tests to our current iteration of our package, which is now available on CRAN.

> Code style. Code should be linted to ensure a uniform code style; most linters will also catch common mistakes like non-speaking variable names (e.g. what’s the difference between ‘data‘ and ‘data1‘?)

Thank you for this suggestion. We have used the `styler` package throughout the latest iteration of our package (now available on CRAN) to uniform the code style.

> Meaningful commit messages. I highly recommend putting the effort in and writing out meaningful commit messages that describe what the commit changes (and perhaps why). Right now, having messages like ‘package update‘ or ‘tiny changes‘ don’t really help anyone.

We will improve the quality and meaningfulness of the commit messages in the future to better reflect the changes.

References

Catherine Hurley, Mark OConnell, and Katarina Domijan. *condvis2: Interactive Conditional Visualization for Supervised and Unsupervised Models in Shiny*, 2022. URL https://CRAN.R-project.org/package=condvis2. R package version 0.1.2.