# Responses to the Executive Editor and Reviewers on the Manuscript 2022-41

Haixu Wang and Jiguo Cao

We thank the executive editor and reviewers for their critical assessments of our work. Their feedback greatly helps us to improve the quality of the manuscript. In the following, we address their concerns point by point.

---

## Responses to the Executive Editor's Comments on the Manuscript 2022-41

*The paper and code is much improved. You did not submit a "point-by-point response to the reviewers comments" with your revision. This is needed.*

**Response:** Thank you very much for the positive feedbacks and suggestion. When we submitted our previous revision, we put the "point-by-point response to the reviewers comments" at the end of the revised manuscript in the same pdf file. In this revision, we have now put the responses to reviewers' comments in this separate document.

*You can remove the blue text in the paper.*

**Response:** Thank you. We have changed the blue text to black.

*Some of your code is VERY slow to run, for example, line 273. You need to include progress bars so the user can know how much more time is needed for it to complete.*

**Response:** Thank you very much for pointing this out. We have addressed the verbal option in functions as commented by Reviewer 1's 3rd comment. The functions can now to show or suppress iteration information by setting `verbal=1` or `verbal=0` respectively.

## Responses to Reviewer #1's Comments on the Manuscript 2022-41

*The authors present a new package which is designed to estimate parameters from ODE models using the parameter cascade method. The manuscript itself provides good background on this methodology, as well as alternatives (of which there aren't many). There is good documentation of the package itself and technical backgrounds on the methods.*

**Response:** Thank you very much for the summary and positive comments. In the following, you can find our point-by-point responses to your comments and suggestions. The blue color highlights changes in the manuscript.

## Responses to comments

1. The paper shows numerical solutions to ODE systems which appears to perform well. For these systems, if there is an analytic Jacobian and Hessian matrix, a comparison in performance between pCODE and CollocInfer would be very useful. If there are no known forms for these, then that should be stated as it is the main advantage of this package, but in that case, I would like to see an additional example where these expressions are known so a comparison can be seen.

**Response:** Thank you very much for the suggestion. We have included the applications of the CollocInfer package. In the following, you can find the detailed results and comparisons between pCODE and CollocInfer. We have also updated the R code to include the comparisons.

We will focus our comparisons on the Fitz-Hugh Nagumo model which is also tested by the CollocInfer package document. The later package requires the input of Jacobian and Hessian matrices. For comparisons, we will simulate 100 data sets based on the true ODE model with the same set of parameter values. Each simulated data set will be added by some random noises and passed to both our pCODE and CollocInfer package for parameter estimations. The following Figure 1 illustrates the estimates of the parameters (a,b,c) over 100 replications from both methods:

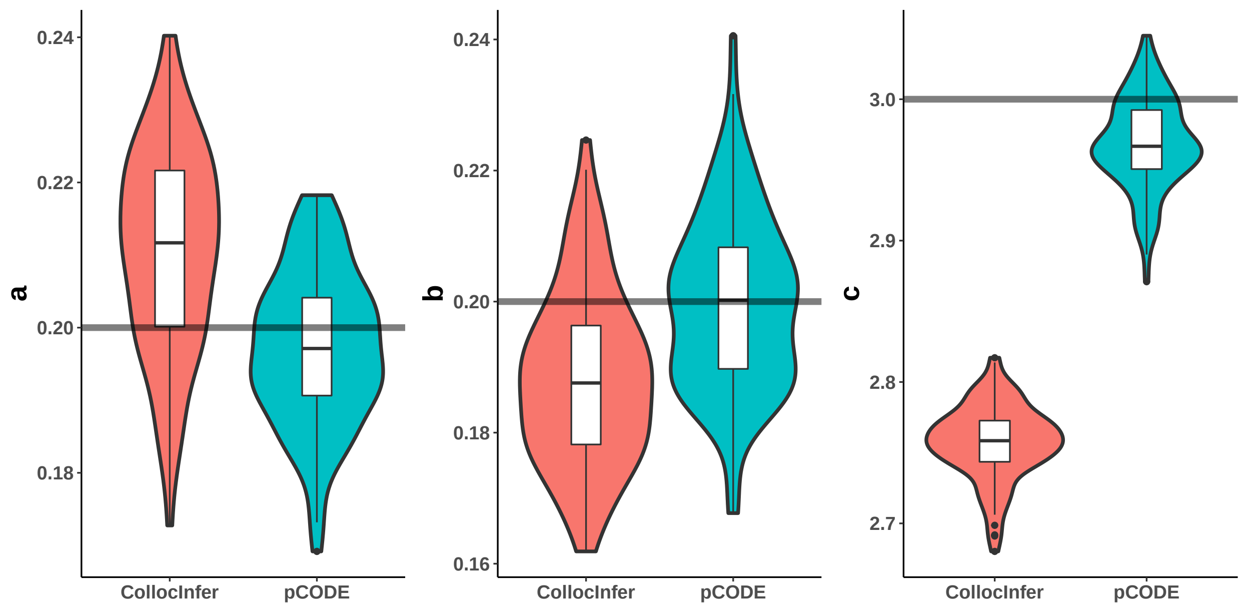

Figure 1: Parameter estimates of (a,b,c) over 100 simulated data sets for both `pCODE` and `CollocInfer` packages. The blue violion plot contains estimates from `pCODE` whereas the red one contains those from `CollocInfer`. The black horizontal lines correspond to the true model parameters used for simulating data sets.

Package `CollocInfer` requires the Jacobian and Hessian matrices for estimating the parameters of the ODE models, whereas the proposed `pCODE` package is a derivative-free method. Moreover, we can use Figure 1 to compare the two packages on the parameter estimation of the Fitz-Hugh Nagumo model. Both packages are estimating parameters a and b with satisfaction, and the estimates from both methods cover the true values. However, we can see that `pCODE` produced better estimations for the third parameter c in the model. Our package does not depend on users to provide tedious details but still performs equally well in parameter estimations.

2. The code provided with the paper begins with rm(list=ls()) which I would not recommend. It is better to trust the author to run a fresh R process, as is becoming the standard.

**Response:** Thank you very much for the suggestion. We have removed this line of code in the R codes for the manuscript.

3. In both the pcode , bootsvar and deltavar functions, an argument to supress output as the code is running would be a great addition. The output is nice to know the function is running (and optimising) but it is rather a lot of output, and when running a lot of repeats,

is largely superfluous information. The internal function nls_optimize does indeed have this argument, but it is hardcoded in the outer wrappers to be set as verbal=1.

**Response:** Thank you very much for the suggestion. In the newest version of the package (v0.9.4), we are able to use `verbal=1` or `verbal=0` to show or suppress iteration information from the optimization processes. One example is the following:

```
pcode(
  data = observation, time = times, ode.model = ode.model,
  par.initial = 0.3, par.names = 'theta',state.names = 'X',
  basis.list = basis, lambda = 1e2,controls = list(verbal = 0)
)
```

4. The authors don't link to the GitHub page in the paper itself, so I've assumed it to be at https://github.com/alex-haixuw/PCODE. If this is the correct source URL, it should be added to the DESCRIPTION file.

**Response:** Thank you very much for the suggestion. We have added the GitHub page of the package in the DESCRIPTION file. The source URL is available at `https://cran.r-project.org/web/packages/pCODE/index.html`.

5. There are very few automated tests (in fact only one, which just appears to test the raw functionality). I'd like to see greatly expanded test coverage. Testing coverage with covr reveals substantial chunks of code that are untested, which will make future development risky

**Response:** Thank you very much for the comment. We have created a few more automated tests for models with likelihood functions, bootstrap and delta variance estimators, and ODE models with several dimensions. We have also employed `covr` to check code coverages of tests. The updated package tests are available on the GitHub page at `https://github.com/alex-haixuw/PCODE/tree/master/tests/testthat`.

6. Each function is well documented with sufficient detail to run the package.

**Response:** Thank you very much for this positive comment.

7. I note that all the functional code is in a single file ( PCODE.R ). There isn't strictly speaking anything wrong with this, although convention is moving toward functions being in their own files for increased readability / ease of tracking in version control. I don't think it's worth the authors splitting these functions out, but it is something perhaps to keep in mind for future development.

**Response:** Thank you very much for the suggestion. We will split the single R file into individual components during future developments.

8. Line 155 of pcode_rjournal.R : I'd like to see a programmatic solution for this. Indeed when I ran the code from top to bottom, some of the estimation repetitions did fail, but I can't easily tell which ones (without scrolling through the output). I imagine this could be solved with a try/catch block and saving certain iterations as NULL

**Response:** Thank you very much for the suggestion. We have modified the codes to save iterations as `NULL`, where `ode`, the ODE solver, does not work. Using the suggested `tryCatch`, we are able to run codes for the full iterations and print the iteration where `ode` fails.

```
tryCatch({
  desolve.mod <- ode(y=est.state, times=times, func=ode.model, parms=est.par)
  est.curve[,jj] <- desolve.mod[,2]
}, error=function(e){cat("ERROR at iteration ", jj, "\n")})
```

## Responses to Reviewer #2's Comments on the Manuscript 2022-41

*The authors provide the R-package 'pCODE' which employs a parameter cascade method to estimating the parameters of the ODE models, similar to 'CollocInfer' package, however using a derivative-free method. This provides an advantage that users do not need to supply the Jacobian or Hessian matrix. The problem is significant and authors' contribution may be valuable but I recommend major revision is required to make this paper significantly better. Some specific recommendations are provided below.*

**Response:** Thank you for the summary and comments. We address your concerns and suggestions point by point in the following paragraphs. The blue color indicates changes in the manuscript.

## Major points

1. *Figure 3: Shouldn't the plots of the bootstrap and delta method variance estimator be presented in the same scale so that it is easier to compare across the two methods?*

   **Response:** Indeed, thank you very much for the suggestion. We have improved the graphs so the scales are consistent between two sets of estimates. Figure 2 is the updated plot.

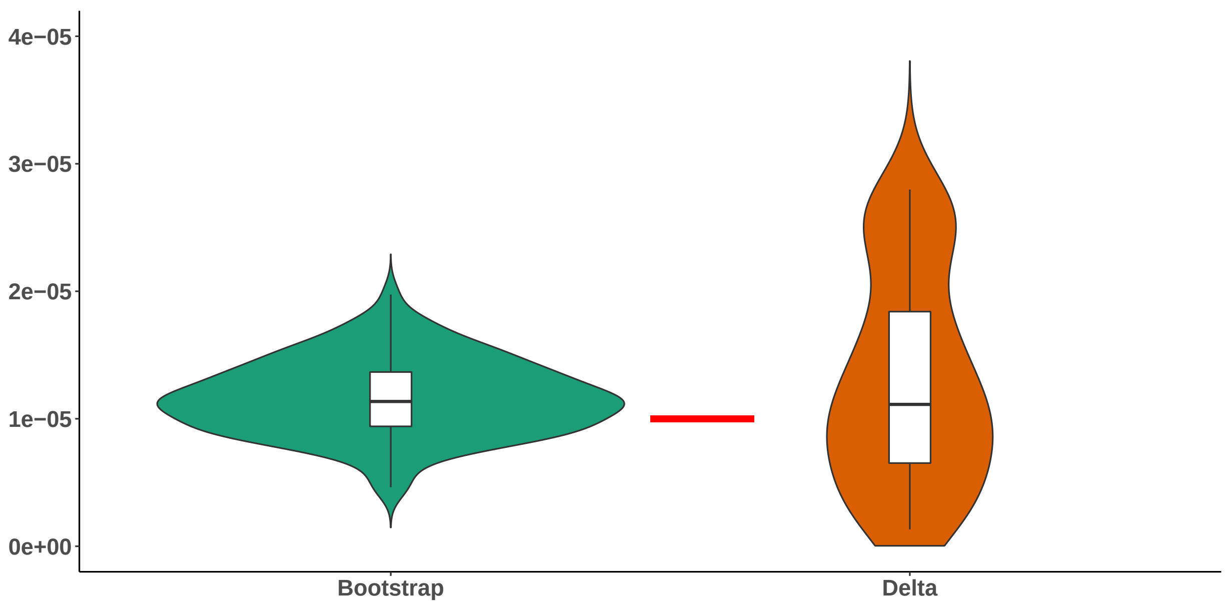

Figure 2: Violin and boxplots of the variance estimates from both the bootstrap and Delta variance method in 100 replications. The red line segment indicates the true variance. Green plot includes the estimates from the bootstrap method, whereas the origin plot includes those from the Delta method.

2. *I would like to see the violin plot overlaid on top of the boxplots in Figure 2 and Figure 3.*

**Response:** Thank you very much for the suggestions. We have added the violin plots overlaid on top of the boxplots in both figures. Figure 3 replaces the original Figure 2. Figure 2 replaces the original Figure 3.

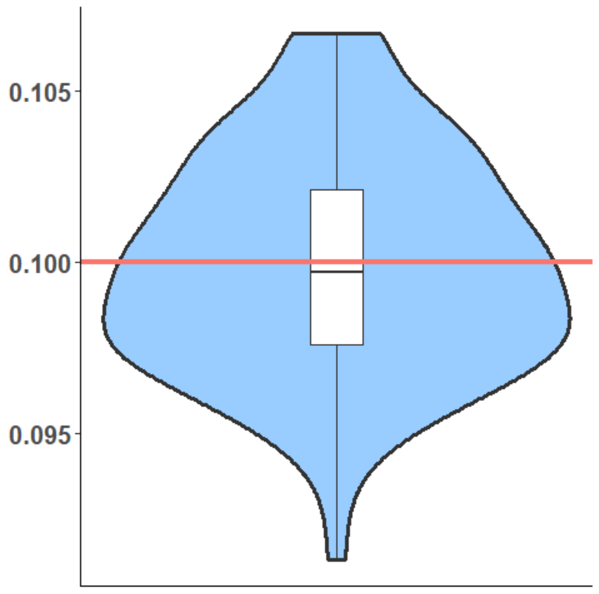

Figure 3: Violin and boxplots of the parameter estimates over 100 replications. The red line segment indicates the true parameter.

3. *Please stylise code so that it is consistent. E.g. include spaces after comma. See https://style.tidyverse.org/ or use the 'styler' package.*

**Response:** Thank you very much for the suggestion. We have stylized our example codes with the `styler` package.

4. *I ask the authors to expand their Summary section to a Discussion. E.g. what are potential limitations with their package and possibility of future work? The two ODE models presented assuming i.i.d errors. What is the performance if this is not the case? I also note that there is no presentation on the accuracy of the parameters estimates for the FitzHugh-Nagumo ODE model?*

**Response:** Thank you for the comments. First, we have included the presentation of the accuracy of parameter estimation for the FitzHugh-Nagumo ODE model as well as a comparison to the existing `CollocInfer` package. The detailed result is updated in Figure 4.

In addition, we can also compare the estimation performance between the proposed package `pCODE` and the existing one `CollocInfer`. Figure 4 summarizes the comparison between two packages on the parameter estimation of the Fitz-Hugh Nagumo model.

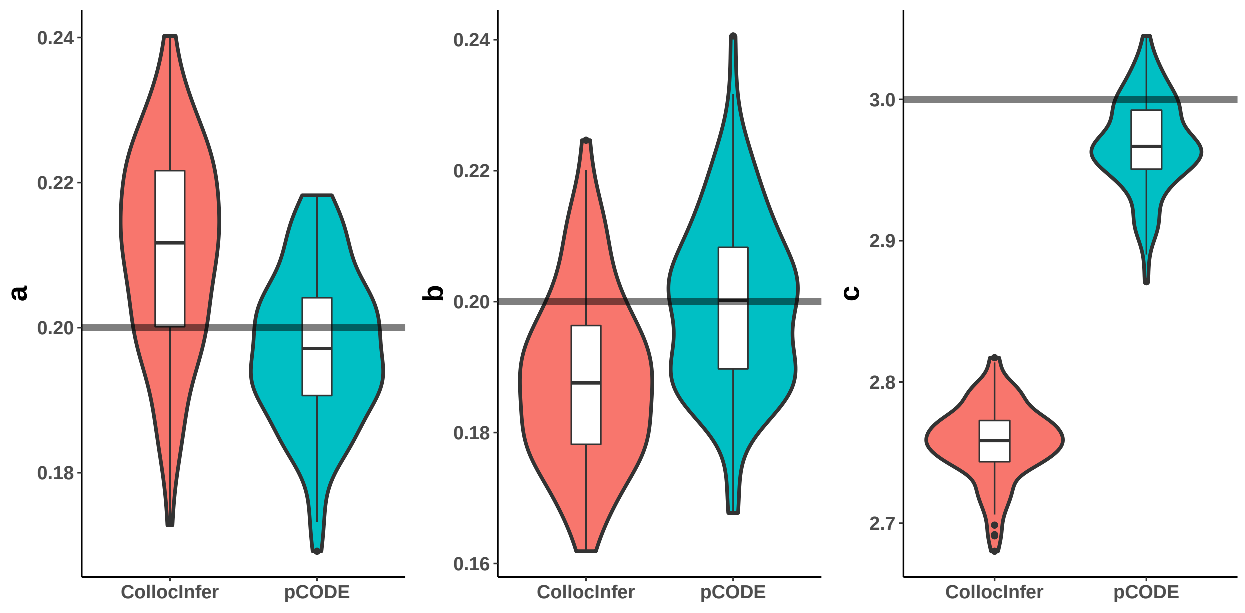

Figure 4: Parameter estimates of (a,b,c) over 100 simulated data sets for both `pCODE` and `CollocInfer` packages. The blue violion plot contains estimates from `pCODE` whereas the red one contains those from `CollocInfer`. The black horizontal lines correspond to the true model parameters used for simulating data sets.

Package `CollocInfer` requires the Jacobian and Hessian matrices for estimating the parameters of the ODE models, whereas the proposed `pCODE` package is a derivative-free method. Both packages are estimating parameters a and b with satisfaction, and the estimates from both methods cover the true values. However, we can see that `pCODE` produced better estimations for the third parameter c in the model. Our package does not depend on users to provide tedious details but still performs equally well in parameter estimations.

Second, we have expanded the Summary section to a Discussion. In the following, you can find our discussions about the limitations and future work of this package.

One of the future works is to expand the flexibility of this package. Even though `pCODE` can provide satisfactory parameter estimates, current functions do not allow users to input the Jacobian and Hessian matrices to speed up the optimization process. In future package developments, we will implement the functionality to allow the input of these matrices. One limitation of the package is the time performance of procedures, especially for the calculation of the bootstrap variance estimator. For that matter, we are implementing this function with parallel computation ability to speed up the calculation. Moreover, we plan to expand the functionalities of the package to include data visualizations and generations of diagnostic plots and summaries..

5. *Could the authors comment on time performance? In particular, the bootstrap approach I imagine would be time consuming?*

**Response:** Thank you very much for pointing out this. We have addressed this comment in the Discussion section about the limitations and future works of this package. The time performance of profiled estimation (with nested optimizations) is often an issue. One major factor impacting the speed of optimization is the number of basis functions used for smoothing the observations. In the case of the Fitz-Hugh Nagumo model example, each dimension requires 101 basis functions. Choosing between the speed and quality of estimation depends on the number of data points, and more basis functions usually lead to better parameter estimates at the cost of computation speed. One particular solution to deal with the bootstrap approach is to employ parallel computation which will be much faster than the current version. We will include this functionality in the future version of the package.

6. *Could the authors include comparison of their package with CollocInfer? While I understand that CollocInfer will have the benefits of providing extra information by providing derivative information, I think it is beneficial for readers to know how much they have to gain/lose in using this approach.*

**Response:** Thank you very much for the suggestion. We have carried out the comparison and presented the results in Figure 4 per the earlier comment.

7. *There are a number of improvements needed for scholarly writing as written under Minor revisions. Could the authors please be sure to check these through thoroughly for future submission?*

**Response:** Thank you very much for pointing this out. We have gone through the minor revisions listed below.

## Responses to questions of the reviewer

- *I didn't fully understand Figure 2. Why is there multiple gray lines in Figure 2(b)? The black one is the true data generating process and the gray lines are based on best estimate but why is there multiple?*

**Response:** Given the true ODE model with its parameters and initial value, we can add noises to the data generating process for getting simulated data. Each gray line corresponds to one estimated model from a set of simulated observations with random noises. We have replicated the simulation and estimation process 100 times, hence we can obtain 100 estimated ODE models with the estimated parameter and initial value. In the meantime, we can plot each estimated ODE model as a gray line in Figure 2(b) and compare it to the true data generating process.

- *Can authors explain why the variance estimate is biased to a higher value in Figure 3?*

**Response:** The bootstrap variance estimator is better than the Delta one as shown in Figure 3. This is as expected since we are able to estimate parameters very well, hence we can to estimate the variance with more confidence. On the other hand, the Delta variance estimator is sensitive to both the simulated noisy observations and the current parameter estimate. Therefore, we can see that the variance estimates fluctuate a lot even though we are using the same data-generating process. In applications, we would recommend users use the bootstrap variance estimator when time is not an issue.

## Minor points

- *"We show that pCODE offers a derivative-free procedure to estimate any ODE models where its functions are easily understood and to apply." -¿ "We show that pCODE offers a derivative-free procedure to estimate any ODE models where its functions are **easy to understand and apply.**"*

**Response:** Thank you for pointing out this. We have corrected the sentence.

- *"analytic solution" -¿ "analytical solution"*

**Response:** Thank you for pointing out this. We have corrected the typo.

- *"pakcage pCODE along with their usages and syntaxes" -¿ typo for "package"*

**Response:** Thank you for pointing out this. We have corrected the typo.

- *Consistency required in the headings, e.g. why is the parameter in the "The Parameter cascade method" capitalised?*

**Response:** We have corrected the headings to be consistent.

- *Italicise mathematical notations, e.g. i-th, j-th, t etc.*

**Response:** Thank you for pointing out this. We have italicized the mathematical notations for better consistency.

- *"Our goal is to estimate $\theta$" -¿ missing a space*

**Response:** We have added the missing space.

- *Missing full stop at equation (3).*

**Response:** We have added the full stop.

- *Use $N \times N$ instead of $N$ * $N$.*

**Response:** We have corrected the typo.

- *Figure 5 has a typo on the x-axis label: "Tine" should be "Time"*

**Response:** Thank you very much. We have corrected the typo in the Figure.

- *The code provided for 'likfun' is inefficient. 'dnorm' function is vectorised so you can just simply use 'sum(dnorm(x, 0, 0.1, log = TRUE))'*

**Response:** Thank you very much. We have replaced the original codes with the suggested method.