We would like to thank the community member for their comments. We included the original comment in black font and **our response in bold violet font**.

Any planned changes or additions to the text are in violet font with boxes around them.

Response CC1 – John Ding

Figure 2. Distribution of the six evaluated performance metrics

In Figure 2, the histograms of the NSE (Nash-Sutcliffe efficiency) and R (Pearson correlation coefficient) highlight calibration results of four lake models on a daily step. The former is a variance-based, and the latter, a correlation-based metric. The simulated data counts cluster around a median value of 0.96 (an exact value not shown in the text) and of 0.98 (shown in Line 220), respectively, both at the upper end of their performance scale.

In a different context of rainfall-runoff modelling, for an observed hydrograph, Duc and Sawada (2023, Equation 25, Figure 2) show that the upper end/bound of the NSE is related to the correlation coefficient, R, as follows: NSEu=2-1/(RxR).

The median NSE and R values for simulated lake water temperatures appear to follow the equality.

Reference

Duc, L. and Sawada, Y.: A signal-processing-based interpretation of the Nash–Sutcliffe efficiency, Hydrol. Earth Syst. Sci., 27, 1827–1839, https://doi.org/10.5194/hess-27-1827-2023, 2023.

Thank you for this interesting comment. It is reassuring to see that the model performance for the single best model is following the relationship. If you or anyone else is interested in further exploring the relationship between different performance metrics we would like to point out that the data containing all evaluated model performance metrics for the 2000 parameter sets for each lake and model are available in the Zenodo repository linked in the "Code and data availability" section of the manuscript (https://zenodo.org/doi/10.5281/zenodo.13165427). Below, we attached R code snippets to visualize some of these data. Please note that during the calibration we calculated NSE and Equation 25 in Duc and Sawada (2023) is using NSEu, the upper limit for NSE when there is no bias in the simulation, which might explain the difference. As these details are not strongly related to the main topic of the manuscript, we will not adapt any changes regarding this in the manuscript.

```
################################################################################
##                                                                            ##
##  A short script to investigate the relationships between different model  ##
##  performance metrics used in the calibration of 4 1D hydrodynamic lake    ##
##  models. The data can be found in the results_lhc.zip file here:          ##
##  https://github.com/aemon-j/isimip-sensitivity-analysis/blob/main/data/   ##
##  Info on the calibration setup and the used models can be found in the    ##
##  manuscript: https://doi.org/10.5194/egusphere-2024-2447                  ##
##  Author: J. Feldbauer, date: 2024-10-17                                   ##
##                                                                            ##
################################################################################


# load necessary libraries
library(tidyverse)


# read in data from the calibration
res <- read.csv("results_lhc.csv", sep = ",", header = TRUE)


# filter the model runs to runs with OK performance metrics
res <- res |> filter(abs(bias) <= 1, nse >= -2, r >= 0.6)


# data.frame with relationship (NSEu=2-1/(RxR)) from  Duc and Sawada (2023)
dat_das <- data.frame(R = seq(-1, 1, by = 0.05),
                      NSEu = 2 - 1/seq(-1, 1, by = 0.05)^2)


# plot NSE against R, color code the points according to the bias and facet the
# plot to the four used models. As for some parametrizations the NSE is very low
# the y axis is limited from -2 to 1. Add line with NSEu=2-1/(RxR) relationship
p1 <- res |> ggplot() +
  geom_point(aes(x = r, y = nse, col = bias)) + facet_wrap(~model) +
```

```
  geom_line(data = dat_das, aes(x = R, y = NSEu), lwd = 1.3) + xlim(0.6, 1) +

  ylim(-2, 1) + scale_color_gradient2(low = "red4", mid = "cyan", high = "red4")

# save plot as png figure

ggsave("r_nse_rel1.png" ,p1, width = 10, height = 6)
```