ERDC/TN EMRRP-EBA-26
September 2022

Ecological Model Development: Evaluation of System Quality

by S. Kyle McKay[1], Nate Richards[2], and Todd M. Swannack[3]

PURPOSE: Ecological models are used throughout the US Army Corps of Engineers (USACE) to inform decisions related to ecosystem restoration, water operations, environmental impact assessment, environmental mitigation, and other topics. Ecological models are typically developed in phases of conceptualization, quantification, evaluation, application, and communication. Evaluation is a process for assessing the technical quality, reliability, and ecological basis of a model and includes techniques such as calibration, verification, validation, and review. In this technical note (TN), we describe an approach for evaluating system quality, which generally includes the computational integrity, numerical accuracy, and programming of a model or modeling system. Methods are presented for avoiding computational errors during development, detecting errors through model testing, and updating models based on review and use. A formal structure is proposed for model test plans and subsequently demonstrated for a hypothetical habitat suitability model. Overall, this TN provides ecological modeling practitioners with a rapid guide for evaluating system quality.

INTRODUCTION: Ecological models serve a valuable role in informing complex water resource and environmental decisions, particularly when they facilitate collaboration and increase the transparency of decisions (Langsdale et al. 2013). The USACE uses ecological models to inform many aspects of project planning, design, and operations, and the model development process efficiently parallels and complements USACE planning processes (McKay et al. 2019). Ecological model development typically proceeds through a repeatable series of steps such as conceptualizing the processes to be captured, quantifying model structure in the language of mathematics, evaluating models, applying models to answer a given question, and transparently communicating each step (Grant and Swannack 2008).

Ecological models are defined as "a representation of a system for a purpose" (USACE 2011, EC-1105-2-412), and model evaluation focuses on assessing how well a model represents its stated objectives. Model evaluation includes a broad suite of methods for examining the quality and appropriateness of an ecological model for a given application. Due to differences in underlying philosophies (Eker et al. 2018), terminology is often inconsistently used in model evaluation (Refsgaard and Henriksen 2004; Augusiak et al. 2014). Comprehensive reviews of model evaluation methods are addressed elsewhere (e.g., Rykiel 1996; Grant and Swannack 2008; Swannack et al. 2012; Bennett et al. 2013; Augusiak et al. 2014), and generally include both

1 U.S. Army Engineer Research and Development Center (ERDC), Environmental Laboratory (EL), New York, NY, Phone: 917-790-8717, Email: kyle.mckay@usace.army.mil
2 US Army Corps of Engineers Rock Island District, Rock Island, IL
3 ERDC-EL, Austin, TX

quantitative and qualitative assessments of a given model and its application. Briefly, evaluation techniques can be coarsely divided into six major subjects based on the scope of what is being evaluated (combined from Bennett et al. 2013 and Augusiak et al. 2014).

1. *Conceptual model evaluation* assesses simplifying assumptions, epistemological biases, assumed temporal and spatial scale of interacting processes, and other parallel issues.

2. *Data evaluation* refers to the underlying numerical and qualitative data used in the calibration and parameterization of the model along with the limitations of those data (e.g., incomplete period of record, representativeness of sites, data quality).

3. *Implementation verification* refers to whether a model numerically performs as intended and described what we refer to as system quality (the focus of this TN).

4. *Model output verification and corroboration* compares model predictions to data (either involved in or independent from model building), which includes dozens of methods including not only visual and statistical performance, but also correcting for pattern recreation and parameterization issues (Bennett et al. 2013).

5. *Model analysis* examines model behavior and emergent properties with sensitivity analyses, which address the relative effect of parameters on outcomes (Pianosi et al. 2016).

6. *Qualitative evaluation* addresses topics such as model purpose, credibility of models and modelers, transparency and communication, value beyond prediction (e.g., learning, collaboration), and buy-in from stakeholders (Bennett et al. 2013; Merritt et al. 2017).

The USACE has developed a set of model review procedures to ensure the quality of models used in project decision-making (USACE 2011). The *model certification* process evaluates three major dimensions of models. First, *technical quality* is reviewed relative to underlying theory supporting an analysis and use of best available science. Second, the *system quality* is tested by examining the computational integrity of a model or modeling system (i.e., implementation verification, Augusiak et al. 2014). Third, model *usability* assesses the capacity of agency practitioners to understand and reliably apply tools with minimal error. Here, we describe best practices for evaluating the system quality of ecological models by avoiding errors (quality assurance), detecting errors through formal testing (quality control), and updating models based on review and use (model update). We use the term *model* in a general sense to include all numerical tools used in USACE projects, which range in complexity from simple spreadsheets with a few cells to complex models with thousands of lines of computational code.

Quality Assurance Best Practices
- Plan model development.
- Develop good modeling habits.
- Documentation is key.

QUALITY ASSURANCE TO AVOID ERRORS:

The importance of both quality assurance and quality control is well-documented in other fields of science, manufacturing, construction, and technology development (e.g., ANSI 9000). However, peer-reviewed literature and best practice documents on ecological model evaluation emphasize identification of problems *post hoc* (i.e., quality control of a model), while avoidance of problems by developers is often as or more crucial (i.e., quality assurance in the modeling process). While

these techniques are common in other fields (e.g., rigorous laboratory procedures for chemical analyses), model development often begins in a more *ad hoc* manner. Following good process does not guarantee a good product but managing system quality of models begins with error prevention and avoidance. An abbreviated list of error avoidance best practices includes the following.

- *Plan workflow*: Model development benefits from a planning step, where workflow of the tool is outlined or structured around the needed inputs, functions, intermediate outcomes, outputs, and visualizations (McKay 2009).

- *Work in teams*: Dedicated modeling time and team-based coding are rarely applied in a modern work environment that values multi-tasking and spurns duplicative effort. However, these procedures catch errors as they occur; for instance, Panko and Halverson (2001) showed that spreadsheet development in groups of three led to 78% fewer errors.

- *Maintain consistency*: Consistency in style, documentation, and archiving facilitate use and future development of models (O'Beirne 2005). "Good coding style is like using correct punctuation. You can manage without it, but it sure makes things easier to read." (Wickham 2019). Some organizations establish style guides or templates for consistency (e.g., Google's Style Guide, https://google.github.io/styleguide/Rguide.html). However, a corporate- or agency-style is not required to maintain a set of personal best practices for descriptive variable naming, input-output indication, punctuation, or other stylistic elements (e.g., simple input-output color coding in spreadsheets; McKay 2009).

- *Use error messages*: Errors occur in ecological models, and developers, reviewers, and users all benefit from knowing the location and type of error. Error messages are a key mechanism for communicating with developers, users, and reviewers, and they do not need to be programmed in complex or comprehensive ways to provide utility. For instance, a few points of input and output checking often improve the rigor of a tool. Simple error messaging might include techniques for catching out-of-range inputs (e.g., conditional formatting in Microsoft Excel for a percentage-based input) or ending a script if an output is inappropriate (e.g., a code break for a negative depth value).

- *Document as you go*: Real-time model documentation leads to better outcomes because tools are more understandable, important features are not forgotten between development and reporting, and key aspects or complex segments of code are clearly articulated. Documentation can take many forms such as a modeling notebook, in-line code documentation, or parallel report writing. Open science tools have emerged to facilitate documentation with code such as R Markdown (https://rmarkdown.rstudio.com/), Jupyter Notebooks (https://jupyter.org/), and the review toolbar in Microsoft Excel.

QUALITY CONTROL TO FIND ERRORS: Quality control of models includes troubleshooting, stress-testing (i.e., breaking the model), and verifying numerical outcomes of a tool. Many developers conduct these activities informally, but a formal set of procedures increases transparency and trust with non-developers that tools have been appropriately developed and checked (Eker et al. 2018).

Quality Control Best Practices

- Check as you go.
- Formally document testing.

Interim code checking. Model testing need not be confined to final products. Individual scraps or lines should be tested and verified during coding, and larger loops, functions, and modules should also be independently examined. Furthermore, colleagues can provide a valuable check on intermediate model versions to ensure the basic structure of the model and provide in-process feedback. Open-science methods facilitate this process by enhancing transparency in code sharing and documentation. Interim testing and error-checking can also be facilitated with a growing family of developer-oriented tools. For instance, RStudio (https://rstudio.com/) is a common platform for accessing the R Statistical Software that color-codes and automatically spaces code for easier viewing and reading, and Microsoft Excel has tools for tracking precedent and dependent cells, showing formulas, and other auditing capability (McKay 2009).

Formal model testing. Formal testing procedures provide a powerful mechanism for both ensuring the system quality of tools and transparently communicating the rigor of the model to reviewers and users. Test plans provide a structure to test and to track a series of challenges posed to the tool. A test plan can be developed prior to development or after completion of models. Test plans rarely provide a comprehensive examination of every aspect of a model (e.g., dozens of tests for every line of code in a 10,000-line software) but instead transparently communicate how and under what conditions a model was tested and the associated numerical accuracy of a tool. Test plans vary with model scale and scope, but the basic elements are as follows.

- *Introduction*: Test plans should begin with an overview of the plan and clear objectives for developing and implementing tests for a particular model.

- *Model Component*: Ecological models typically contain many nested components, which can be tested separately or collectively (e.g., a single line, a function, or an entire model). The model components to be tested (and those not being tested) should be clearly identified and should provide sufficient error checking to give both the developer and users confidence in the tools. Some model components may be informally tested during development (see above) and may not be included in the test plan.

- *Test Approach*: Each model component may undergo specific tests and documentation. What model components were tested? How were components tested and answers checked (e.g., against manual calculations)? Who conducted tests (developer, co-author, third party)? When were tests conducted (beta vs. final version)? Some common tests include use of expected input ranges, use of boundary conditions (e.g., depth = 0), comparison with prior tools (e.g., existing software), and input of out-of-range values expected to lead to errors or logical inconsistencies (e.g., negative depth, 110% ground cover). Grouping tests condenses and simplifies communication (e.g., 25 random inputs, all passed). We suggest the final test plan should include only successful outcomes, and thus, any test failures should lead to repair of the model, recompiling code, and complete retesting of the model.

- *Outcome*: The criteria used to determine whether a test item has passed or failed needs to be specified along with a simple outcome (e.g., pass/fail).

- *Deliverables*: Test results should be documented clearly through narrative, tabular, or graphical outputs. Developers should be able to clearly describe outcomes and key model updates made because of testing, as these are often of interest to reviewers.

MODEL UPDATE: The procedures described here have largely focused on quality assurance and control as executed by model developers. However, peer review and model use provide two other mechanisms for identifying errors and updating models.

Model Update Best Practices

- Plan ahead for review.
- Establish mechanisms for archival, update, and version control.

Peer review. With a growing emphasis on simulation and analytics, models are increasingly scrutinized and reviewed as a standard practice in ecological applications (e.g., publication of modeling code by refereed journals). The USACE's model certification process (USACE 2011) is a formal model review procedure, where reviewers are consistently charged to address system quality issues such as: Are model computations presented in sufficient detail? Has the model been tested for errors? Does documentation sufficiently describe testing? Does the model inform users of erroneous or inappropriate inputs or outputs? Because of the consistency of system quality issues, developers can preempt reviewer comments and needs, and a model evaluation plan could be written prior to model development (as is the case with other reviews of USACE reports).

Model use. System quality errors are often identified during the application of models, and reporting errors, revising code, and archiving code are complex challenges for developers and organizations alike. These issues are particularly important for complex models or software developed over long time periods but are also crucial to models used infrequently over long periods (e.g., a spreadsheet used on two projects a decade apart). Software versioning methods are well-described in computer science such as sematic versioning of models in Major.Minor.Patch format (e.g., Version 1.1.4). Long-term maintenance also includes archival of models for both internal use (e.g., a USACE District server or with the USACE Ecosystem Restoration Planning Center of Expertise) and external communication (e.g., a public facing model library).

HYPOTHETICAL CASE STUDY: Habitat suitability models are common in ecosystem restoration practice (Swannack et al. 2012), and here, we use a hypothetical model for the purpose of demonstrating quality assurance and control procedures. Habitat suitability models assess habitat quantity for a particular taxa (e.g., acres of potential slough darter habitat) and scale the area by metrics of habitat quality (i.e., suitability indices indicating perfect habitat as one and inhospitable habitat as zero). The details of this model are immaterial due to its demonstrative purpose, but a model was developed for a generic forest bird including a continuous variable for forest cover and a categorical variable for canopy age and complexity. These two suitability metrics are combined via a geometric mean, which assumes either variable can serve as a limiting factor for overall habitat suitability.

$$SI_{cover} = Forest_{per} / 100$$

$$SI_{age} = if\ (AgeInput=1,\ 1.0,\ 0.5)$$

$$HSI_{total} = (SI_{cover} * SI_{age})^{0.5}$$

where SI_{cover} is a suitability index for forest cover, $Forest_{per}$ is the percent forested land cover for a patch (0 to 100), SI_{age} is a categorical variable for forest age indicating appropriate canopy structure, *AgeInput* is 1 for "yes" or 0 for "no" (i.e., *AgeInput* = 1 then SI_{age} = 1, *AgeInput* = 0 then SI_{age} = 0.5), and HSI_{total} is cumulative patch suitability based on a geometric mean.

Avoiding errors. The model described above was programmed in the R Statistical Software language. More specifically, RStudio was used to interface with R, which allowed developers to see color-coded comments, variables, inputs, and text (Figure 1). The model drew from existing habitat suitability index models (the ecorest R package, McKay and Hernández-Abrams 2020), and the model was written in the standard style of the lead author (SKM). This style denotes functions clearly with capitalization and specifies the type and structure of input data immediately below a function. In-line documentation facilitated communication among the authors, and interim ad hoc testing was conducted prior to formal testing, which identified preliminary errors in error messages. Code was subsequently reviewed by co-authors with expertise in index-models (NR, TMS) and R (TMS), and all code and outputs were shared via R Markdown.

```
#FUNCTION - SIcover - Percent forest cover of patch
#x = Percent forested land cover for a patch (0 to 100%)
SIcover <- function(x){
 #If input variable is 0 to 100, then compute SI.
 if(x>=0 & x<=100){
 SIcover <- x / 100
 #Send error message if inputs are not 0-100.
 } else {
 SIcover <- "Input percent cover is outside of 0 to 100 range."
 }
 SIcover #Send output
}

#FUNCTION - SIage - Does the forest have sufficient canopy structure
#x = Binary 1 or 0 (Yes or No, respectively)
SIage <- function(x){
 #If input variable is 0 or 1, then compute SI as 0.5 or 1, respectively.
 if(x==1 || x==0){
 SIage <- ifelse(x==1, 1, 0.5)
 #Send error message if inputs are not 0 or 1.
 } else {
 SIage <- "Input should be a categorical YES (input = 1) or NO (input = 0)."
 }
 SIage #Send output
}

#FUNCTION - HSItotal - Compute cumulative HSI with a geometric mean.
#x is a vector of two SI values (range 0 to 1).
HSItotal <- function(x){
 #Check input variable is within range.
 if(all(x<=1) & all(x>=0)){
 HSItotal <- prod(x, na.rm=TRUE)^(1/length(x))
 #Send error message if inputs are not 0 or 1.
 } else {
 HSItotal <- "Inputs are outside of 0 to 1 range."
 }
 HSItotal #Send output
}
```

Figure 1. R code for simple, hypothetical habitat suitability model.

Finding errors. A test plan was developed and executed for this simple model (Table 1). All tests were conducted on the final code by the lead author (SKM). The basic design of the test plan was to check each function independently with expected ranges of inputs as well as values intended to break the model. All values were verified against manual calculations; these quality control procedures identified no known errors.

Table 1. Model test plan for a hypothetical habitat suitability model coded in R.

Component	Test	Expected Value	Outcome
SI_{cover}	Inputs for expected values (0, 25, 50, 75, 100)	0.0, 0.25, 0.50, 0.75, 1.0	Pass
SI_{cover}	Out of range inputs (-50, -1, 101, 150)	Error message	Pass
SI_{age}	Inputs for expected values (1,0)	1.0, 0.5	Pass
SI_{age}	Out of range inputs (5, -1)	Error message	Pass
SI_{age}	Incorrect format inputs ("Yes", "No")	Error message	Pass
HSI_{total}	Pairs of expected input values (SI_{cover}, SI_{age}): (1,1), (0,1), (1,0), (0.5,0.5), (1,0.5), (0,0.5), (0.01,1)	1, 0, 0, 0.5 0.71, 0, 0.01	Pass
HSI_{total}	Pairs of out of range inputs (SI_{cover}, SI_{age}): (-1,1), (1,-1), (1,2), (0,"Yes"), (1,"No")	Error message	Pass

SUMMARY: The objective of model evaluation is to determine if the model is useful for addressing the specific goals of the project (Rykiel 1996). Errors can be introduced at dozens of steps in the modeling process. For instance, an incorrect conceptualization of the system or invalid theory could doom an analysis before a single line of code is programmed. Likewise, a well-validated and acceptable model could be misapplied outside of its intended geographic range or input errors could be introduced. A broader view of model evaluation addresses all these aspects of the model and more, and readers are encouraged to examine more thorough treatments of evaluation by Rykiel (1996), Refsgaard and Henriksen (2004), Swannack et al. (2012), Bennett et al. (2013), and Augusiak et al. (2014). For this TN, we have focused narrowly on the computational integrity of a tool (i.e., its system quality) with a specific focus on techniques for avoiding and detecting errors in ecological models. While many modelers conduct these activities informally, we encourage the transparent and explicit documentation of these steps. Our assumption is that a model must first be computationally accurate before pursuing other forms of validation, and techniques are offered here to achieve this intermediate and important outcome.

ADDITIONAL INFORMATION: The authors are grateful for thoughtful input from the ERDC Integrated Ecological Modeling Team. Additional reviews by Dr. Chuck Theiling and Ms. Darixa Hernández-Abrams significantly improved this TN. The use of products does not represent an endorsement of these products by either the author or the Department of the Army. The study was conducted with support from the Ecosystem Management and Restoration Research Program (EMRRP). For information on EMRRP, please contact the Program Manager, Dr. Trudy Estes (Trudy.J.Estes@erdc.usace.army.mil) or consult https://emrrp.el.erdc.dren.mil/. This TN should be cited as follows:

> McKay, S. K., N. Richards, and T. M. Swannack. 2022. *Ecological model development: Evaluating system quality*. ERDC/TN EMRRP-EBA-26. Vicksburg, MS: US Army Engineer Research and Development Center. http://el.erdc.usace.army.mil/emrrp/emrrp.html.

REFERENCES

American National Standards Institute (ANSI). 2015. *Quality management systems: Fundamentals and vocabulary*. ANSI 9000. American National Standards Institute, Washington, D.C.

Augusiak, J., P. J. Van den Brink, and V. Grimm. 2014. "Merging validation and evaluation of ecological models to 'evaludation': a review of terminology and a practical approach." *Ecological Modelling* 280:117-128.

Bennett, N. D., B. F. Croke, G. Guariso, J. H. Guillaume, S. H. Hamilton, A. J. Jakeman, S. Marsili-Libelli, L. T. Newham, J. P. Norton, C. Perrin, and S. A. Pierce. 2013. "Characterising performance of environmental models." *Environmental Modelling & Software* 40: 1-20.

Eker, S., E. Rovenskaya, M. Obersteiner, and S. Langan. 2018. "Practice and perspectives in the validation of resource management models." *Nature Communications* 9:5359, https://doi.org/10.1038/s41467-018-07811-9.

Grant, W. E., and T. M. Swannack. 2008. *Ecological modeling: A common-sense approach to theory and practice*. Malden, MA: Blackwell Publishing.

Langsdale, S., A. Beall, E. Bourget, E. Hagen, S. Kudlas, R. Palmer, D. Tate, and W. Werick. 2013. "Collaborative modeling for decision support in water resources: Principles and best practices." *Journal of the American Water Resources Association* 49 (3): 629-638.

McKay, S. K. 2009. *Reducing spreadsheet errors*. ERDC/TN EMRRP-EBA-03. Vicksburg, MS: US Army Engineer Research and Development Center.

McKay, S. K., N. Richards, and T. Swannack. 2019. *Aligning ecological model development with restoration project planning*. ERDC/TN EMRRP-SR-89. Vicksburg, MS: US Army Engineer Research and Development Center.

Merritt, W. S., B. Fu, J. L. Ticehurst, S. El Sawah, O. Vigiak, A. M. Roberts, F. Dyer, C. A. Pollino, J. H. Guillaume, B. F. Croke, and A. J. Jakeman. 2017. "Realizing modelling outcomes: A synthesis of success factors and their use in a retrospective analysis of 15 Australian water resource projects." *Environmental Modelling & Software* 94: 63-72.

O'Beirne, P. 2005. Spreadsheet Check and Control: 47 Key Practices to Detect and Prevent Errors. Wexford, Ireland: Systems Publishing.

Panko, R. R., and R. P. Halverson. 2001. "An experiment in collaborative development to reduce spreadsheet errors." *Journal of the Association for Information Systems*, 2 (4).

Pianosi, F., K. Beven, J. Freer, J. W. Hall, J. Rougier, D. B. Stephenson, and T. Wagener. 2016. "Sensitivity analysis of environmental models: A systematic review with practical workflow." *Environmental Modelling & Software* 79:214-232.

Refsgaard, J. C., and H. J. Henriksen. 2004. "Modelling guidelines: Terminology and guiding principles." *Advances in Water Resources* 27: 71-82.

Rykiel, E. J. 1996. "Testing ecological models: the meaning of validation." *Ecological modelling* 90 (3): 229-244.

Swannack, T. M., J. C. Fischenich, and D. J. Tazik. 2012. *Ecological modeling guide for ecosystem restoration and management*. ERDC/EL TR-12-18. Vicksburg, MS: US Army Engineer Research and Development Center.

US Army Corps of Engineers (USACE). 2011. Assuring quality of planning models. EC-1105-2-412. Washington, DC.

Wickham, H. 2019. *Advanced r*. CRC press. http://adv-r.had.co.nz/.