

Developing an automated iterative near-term forecasting system for an ecological study

*Ethan P. White^{1,2,3}, Glenda M. Yenni¹, Shawn D. Taylor⁴, Erica M.
Christensen¹, Ellen K. Bledsoe⁴, Juniper L. Simonis¹, S. K. Morgan
Ernest^{1,3}*

¹ Department of Wildlife Ecology and Conservation, University of Florida, Gainesville,
FL, United States

² Informatics Institute, University of Florida, Gainesville, FL, United States

³ Biodiversity Institute, University of Florida, Gainesville, FL, United States

⁴ School of Natural Resources and Environment, University of Florida Gainesville, FL,
United States

Abstract

1. Most forecasts for the future state of ecological systems are conducted once and never updated or assessed. As a result, many available ecological forecasts are not based on the most up-to-date data, and the scientific progress of ecological forecasting models is slowed by a lack of feedback on how well the forecasts perform.
2. Iterative near-term ecological forecasting involves repeated daily to annual scale

forecasts of an ecological system as new data becomes available and regular assessment of the resulting forecasts. We demonstrate how automated iterative near-term forecasting systems for ecology can be constructed by building one to conduct monthly forecasts of rodent abundances at the Portal Project, a long-term study with over 40 years of monthly data. This system automates most aspects of the six stages of converting raw data into new forecasts: data collection, data sharing, data manipulation, modeling and forecasting, archiving, and presentation of the forecasts.

3. The forecasting system uses R code for working with data, fitting models, making forecasts, and archiving and presenting these forecasts. The resulting pipeline is automated using continuous integration (a software development tool) to run the entire pipeline once a week. The cyberinfrastructure is designed for long-term maintainability and to allow the easy addition of new models. Constructing this forecasting system required a team with expertise ranging from field site experience to software development.
4. Automated near-term iterative forecasting systems will allow the science of ecological forecasting to advance more rapidly and provide the most up-to-date forecasts possible for conservation and management. These forecasting systems will also accelerate basic science by allowing new models of natural systems to be quickly implemented and compared to existing models. Using existing technology, and teams with diverse skill sets, it is possible for ecologists to build automated forecasting systems and use them to advance our understanding of natural systems.

Key-words: forecasting, prediction, mammals, iterative forecasting, Portal Project

44 **Introduction**

45 Forecasting the future state of ecological systems is important for management,
46 conservation, and evaluation of how well models capture the processes governing
47 ecological systems (Clark et al., 2001; Tallis & Kareiva, 2006; Díaz et al., 2015; Dietze,
48 2017). In 2001, Clark et al. (2001) called for a more central role of forecasting in
49 ecology. Since then, an increasing number of ecological forecasts are being published
50 that focus on societally important questions from daily to decadal time scales (Dietze et
51 al., 2018). At daily scales, ecological forecasts predict the occurrence of environmental
52 issues like toxic algal blooms (Stumpf et al., 2009) and pollen (Prank et al., 2013). At
53 monthly scales, forecasts are used to predict the stocks of fisheries (NOAA, 2016) and
54 the probability of coral bleaching events (Liu et al., 2018). At decadal time scales,
55 ecological forecasts are used to predict how biodiversity will change as it responds to
56 anthropogenic influences (Harris et al., 2018). These forecasting examples highlight the
57 important role that ecological forecasts play in recasting ecological knowledge in
58 societally relevant ways and also improve our understanding of ecological systems by
59 testing the ability of our models to predict how systems will change in the future
60 (Dietze et al., 2018; Harris et al., 2018).

61 While some of the examples given above (e.g., fisheries stock estimates) are regularly
62 repeated, most ecological forecasts are made once, published, and never assessed or
63 updated (Dietze et al., 2018). This lack of both regular assessment and active updating
64 has limited the progress of ecological forecasting and hindered our ability to make
65 useful and reliable predictions. The lack of active assessment results in limited
66 information on how much confidence to place in forecasts and makes it difficult to
67 determine on which forecasting methods to build. Without regular updates, forecasts
68 lack the most current data, and the longer a forecast remains out of date, the less
69 accurate it becomes (Petchey et al., 2015; Dietze et al., 2018). More regular updating
70 and assessment will advance ecological forecasting as a field by accelerating the

71 identification of the best models for individual forecasts and improving our
72 understanding of how to best design forecasting approaches for ecology in general. This
73 approach has helped accelerate forecasting ability in other fields such as meteorology
74 (Kalnay, 2003; McGill, 2012; Bauer et al., 2015). For ecological forecasting to mature
75 as a field, we need to change how we produce and interact with forecasts, creating a
76 more dynamic interplay between model development, prediction generation, and
77 incorporation of new data and information (Dietze et al., 2018).

78 With the goal of making ecological forecasting more dynamic and responsive, Dietze et
79 al. (2018) recently called for an increase in iterative near-term forecasting. Iterative
80 near-term forecasting is defined as making predictions for the near future and repeatedly
81 updating those predictions through a cycle of evaluation, integration of new data, and
82 generation of new forecasts. Because forecasts are made ‘near-term’—daily to annual
83 time scales instead of multi-decadal—predictions can be assessed more quickly and
84 frequently, leading to more rapid model improvements (Tredennick et al., 2016; Dietze
85 et al., 2018). Since forecasts are made repeatedly through time, new data can be
86 continuously integrated with each iteration (Dietze et al., 2018). By quickly identifying
87 how models are failing, facilitating rapid testing of improved models, and incorporating
88 the most up-to-date data available, iterative near-term forecasting has the potential to
89 promote rapid improvement in the state of ecological forecasting. In addition to
90 yielding improved information for guiding policy and management (Clark et al., 2001;
91 Luo et al., 2011; Petchey et al., 2015), this iterative approach will help improve our
92 basic understanding of ecological systems (Dietze et al., 2018). For example, alternative
93 mechanistic models can be compared to determine which model provides the best
94 forecasts, thus providing insights into the importance of different ecological processes
95 (Dietze et al., 2018). Iterative near-term forecasting provides the more dynamic
96 interplay between models, predictions, and data that has been identified as necessary for
97 improving ecological forecasting and our understanding of ecological systems more

98 broadly.

99 Because iterative near-term forecasting requires a dynamic integration of models,
100 predictions, and data, Dietze et al. (2018) highlight approaches to data management,
101 model construction and evaluation, and cyberinfrastructure that are necessary to
102 effectively implement this type of forecasting (Box 1). Data needs to be released
103 quickly under open licenses (Vargas et al., 2017; Dietze et al., 2018) and structured so
104 that it can be used easily by a variety of researchers and in multiple modeling
105 approaches (Borer et al., 2009; Strasser et al., 2011). Models need to be able to deal
106 with uncertainty, in both the predictors and the predictions, to properly convey
107 uncertainty in the resulting forecasts (Diniz-Filho et al., 2009). Multiple models should
108 be developed, both to assess which models are performing best (Dietze et al., 2018) and
109 to facilitate combining models to form ensemble predictions which tend to perform
110 better than single models (Araujo & New, 2007; Diniz-Filho et al., 2009; Dormann et
111 al., 2018). Ensuring that data and models are regularly updated and new forecasts are
112 made requires cyberinfrastructure to automate data processing, model fitting, prediction,
113 model evaluation, forecast visualization, and archiving. In combination, these
114 approaches should allow forecasts to be easily rerun and evaluated as new data becomes
115 available (Box 1; Dietze et al., 2018).

116 While iterative near-term forecasting is an important next step in the evolution of
117 ecological forecasting, the requirements outlined by Dietze et al. (Box 1) are not trivial
118 to implement (e.g., making quality data available in near real-time and automatically
119 rerunning forecasts in reproducible ways), and few of their recommendations are in
120 widespread use in ecology today (e.g., Wilson et al., 2014; Stodden & Miguez, 2014;
121 Yenni et al., 2018). We explored what it would entail to operationalize Dietze et al.'s
122 recommendations by constructing our own iterative near-term forecasting pipeline for
123 an on-going, long-term ecological study that collects high-frequency data on desert
124 rodent abundances (Brown, 1998; Ernest et al., 2008). We constructed an automated

125 forecasting pipeline with the goal of being able to forecast rodent abundances and
126 evaluate our predictions on a monthly basis. In this paper, we discuss our approach for
127 creating this iterative near-term forecasting pipeline, the challenges we encountered, the
128 tools we used, and the lessons we learned so that others can create their own iterative
129 forecasting systems. For those interested in implementing iterative forecasting, either on
130 their own or as part of a team, this paper will provide a roadmap for how to build such a
131 system and what skills will be helpful to do so. For readers looking for an introduction
132 to automation and continuous integration in an ecological context, we recommend our
133 paper on data management for continuously collected data, which includes a tutorial on
134 how to set up some of the aspects of automation described in this paper (Yenni et al.,
135 2018).

136 **System Background**

137 Iterative forecasting is most effective with frequently collected data, since it provides
138 more opportunities for updating model results and assessing (and potentially improving)
139 model performance (Box 1; Dietze et al., 2018). The Portal Project is a long-term
140 ecological study situated in the Chihuahuan Desert (2 km north and 6.5 km east of
141 Portal, Arizona, US). Researchers have been continuously collecting data at the site
142 since 1977, including data on the abundance of rodent and plant species (monthly and
143 twice yearly, respectively) and climatic factors such as air temperature and precipitation
144 (daily) (Brown, 1998; Ernest et al., 2009, 2016, 2018). The site consists of 24 50m x
145 50m experimental plots. Each plot contains 49 permanently marked trapping stations
146 laid out in a 7 x 7 grid, and all plots are trapped with Sherman live traps for one night
147 each month. For all rodents caught during a trapping session, information on species
148 identity, size, and reproductive condition is collected, and new individuals are given
149 identification tags. This information on rodent populations is high-frequency, uses

150 consistent trapping methodology, and has an extended time-series (475 monthly
151 samples and counting), making this study an ideal case for near-term iterative
152 forecasting. Forecasting of rodent population dynamics in the southwest (and more
153 broadly) is important because of their link to zoonotic diseases such as hantavirus and
154 plague (Parmenter et al., 1993; Gage & Kosoy, 2005; Springer et al., 2016). In addition,
155 this forecasting system is being used to improve population dynamic modeling for this
156 community and to explore the utility of incorporating experimental data into ecological
157 forecasting models.

158 **Implementing an automated iterative forecasting system**

159 Implementation of iterative forecasting requires the regular updating of models with new
160 raw data as it becomes available and the presentation of those forecasts in usable forms;
161 in our case, this occurs monthly. Updating models in an efficient and maintainable way
162 relies on developing an automated pipeline to handle the six stages of converting raw
163 data into new forecasts: data collection, data sharing, data manipulation, modeling and
164 forecasting, archiving, and presentation of the forecasts (Figure 1a). To implement the
165 pipeline outlined in Figure 1a, we used a “continuous analysis” framework (*sensu*
166 Beaulieu-Jones & Greene, 2017) that automatically processes the most up-to-date data,
167 updates the models, makes new forecasts, archives the forecasts, and updates a website
168 with analysis of current and previous forecasts. In this section we describe our approach
169 to streamlining and automating the multiple components of the forecasting pipeline and
170 the tools and infrastructure we employed to execute each component.

171 **Continuous Analysis Framework**

172 A core aspect of iterative near-term forecasting is the regular rerunning of the
173 forecasting pipeline. We employed “continuous analysis” (*sensu* Beaulieu-Jones &

174 Greene, 2017) to drive the automation of both the full pipeline and a number of its
175 individual components. Continuous analysis uses a set of tools originally designed for
176 software development called “continuous integration” (CI). CI combines computing
177 environments for running code with monitoring systems to identify changes in data or
178 code. Essentially, CI is a computer helper who watches the pipeline and, when it sees a
179 change in the code or data, runs all the computer scripts needed to ensure that the
180 forecasting pipeline runs from beginning to end. This is useful for iterative near-term
181 forecasting because it does not rely on humans to create new forecasts whenever new
182 models or data are added. These tools are common in the area of software development,
183 where they are used to automate software testing and integrate work by multiple
184 developers working on the same code base. However, these tools can be used for any
185 computational task that needs to be regularly repeated or run after changes to code or
186 data (Beaulieu-Jones & Greene, 2017). Our forecasting pipeline currently runs on a
187 publicly available continuous integration service (Travis CI; <https://travis-ci.org/>) that is
188 free for open source projects (up to a limited amount of computing time). This
189 continuous integration integrates directly with GitHub (<https://github.com>), the online
190 repository where we store the associated code and data. Because of the widespread use
191 of CI in software development, alternative services that can run code on local or
192 cloud-based computational infrastructure also exist (Beaulieu-Jones & Greene, 2017).
193 We use CI to quality check data, test code using “unit tests” (Wilson et al., 2014), build
194 models, make forecasts, and publicly present and archive the results (Figure 1b).

195 To ensure that software pipelines continue to run automatically as software
196 dependencies change, a key component of “continuous analysis” is the use of a
197 reproducible computational environment (Beaulieu-Jones & Greene, 2017). We
198 followed Beaulieu and Greene’s (2017) recommendation to use software containers.
199 Software containers are standalone packages that contain copies of everything needed to
200 run a given piece of software, including the operating system (Boettiger, 2015). Once

201 created, a software container is basically a time capsule, containing all the software
202 dependencies in the exact state used to develop and run the software (Boettiger, 2015).
203 If those dependencies change (or disappear) in the wider world, they still exist,
204 unchanged, in the container. We use an existing platform, Docker (Merkel, 2014), to
205 store an exact image of our complete software environment by adding our project
206 specific code to a container created by the Rocker project, which is a Docker image with
207 many important R packages (i.e., the tidyverse packages; Wickham, 2017) pre-installed
208 (Boettiger & Eddelbuettel, 2017). We implemented this system because we experienced
209 issues with external dependencies breaking our pipeline (e.g., when the `tscount`
210 package (Liboschik et al., 2015), was temporarily removed from CRAN and could not
211 be installed in the usual way). In combination, the automated running of the pipeline
212 (continuous integration) and the guarantee it will not stop working unexpectedly due to
213 software dependencies (via a software container) allows continuous analysis to serve as
214 the glue that connects all stages of the forecasting pipeline.

215 **Data Collection, Entry, and Processing**

216 Iterative forecasting benefits from frequently updated data so that state changes can be
217 quickly incorporated into new forecasts (Dietze et al., 2018). Both frequent data
218 collection and rapid processing are important for providing timely forecasts. Since we
219 collect data monthly, ensuring that the models have access to the newest data requires a
220 data latency period of less than 1 month from collection to availability for modeling. To
221 accomplish this, we automated components of the data processing and quality
222 assurance/quality control (QA/QC) process to reduce the time needed to add new data
223 to the database [Yenni et al. (2018); Figure 1].

224 New data is double-entered into Microsoft Excel using the “data validation” feature.
225 The two versions are then compared using an R script to control for errors in data entry.
226 Quality control (QC) checks using the `testthat` R package (Wickham, 2011) are run

227 on the data to test for validity and consistency both within the new data and between the
228 new and archived data. The local use of the QC scripts to flag problematic data greatly
229 reduces the time spent error-checking and ensures that the quality of data is consistent.
230 The cleaned data is then uploaded to the GitHub-based PortalData repository
231 (<https://github.com/weecology/PortalData>). GitHub (<https://github.com/>) is a software
232 development tool for managing computer code development, but we have also found it
233 useful for data management. On GitHub, changes to data can be tracked through the Git
234 version control system which logs all changes made to any files in the repository, giving
235 us a record of exactly of when specific lines of data were changed or added. All updates
236 to data are processed through “pull requests,” which are notifications that someone has a
237 modified version of the data to contribute. QA/QC checks are automatically run on the
238 submitted data using continuous integration to ensure that no avoidable errors reach the
239 official version of the dataset (Yenni et al., 2018).

240 We also automated the updating of supplementary data tables, including information on
241 weather and trapping history, that were previously updated manually. As soon as new
242 field data is merged into the repository, continuous integration updates all
243 supplementary files. Weather data is automatically fetched from our cellular-connected
244 weather station, cleaned, and appended to the weather data table. Supplementary data
245 tables related to trapping history are updated based on the data added to the main data
246 tables. Using CI for this ensures that all supplementary data tables are always
247 up-to-date with the core data (Yenni et al., 2018).

248 **Data Sharing**

249 The Portal Project has a long history of making its data publicly available so that anyone
250 can use it for forecasting or other projects. Historically, the publication of the data was
251 conducted through data papers (Ernest et al., 2009, 2016), the most common approach
252 in ecology; this approach, however, caused years of data latency. With the recent switch

253 to posting data directly to a public GitHub repository (Figure 1) with a CC0 waiver
254 (i.e. no restrictions on data use; <https://creativecommons.org/publicdomain/zero/1.0/>),
255 data latency for everyone has been reduced to less than one month, making meaningful
256 iterative near-term forecasting possible for not only our group but other interested
257 parties, as well (Ernest et al., 2018; Yenni et al., 2018).

258 **Data Manipulation**

259 Once data is available, it must be processed into a form appropriate for modeling
260 (Figure 1). For many ecological datasets, this requires not only simple data
261 manipulation but also a good understanding of the data to facilitate appropriate
262 aggregation. Data manipulation steps are often conducted using custom one-off code to
263 convert the raw data into the desired form (Morris & White, 2013), but this approach
264 has several limitations. First, each researcher must develop and maintain their own data
265 manipulation code, which is inefficient and can result in different researchers producing
266 different versions of the data for the same task. Subtle differences in data processing
267 decisions have led to confusion when reproducing results for the Portal data in the past.
268 Second, this kind of code is rarely robust to changes in data structure and location.
269 Based on our experience developing and maintaining the Data Retriever (Morris &
270 White, 2013; Senyondo et al., 2017), these kinds of changes are common. Finally, this
271 kind of code is generally poorly tested, which can lead to errors based on mistakes in
272 data manipulation. To avoid these issues for the Portal Project data, the Portal team has
273 been developing an R package (portalr; <http://github.com/weecology/portalr>) for
274 acquiring the data and handling common data cleaning and aggregation tasks. As a
275 result, our modeling and forecasting code only needs to install this package and run the
276 data manipulation and summary functions to get the appropriate data (Figure 1b). The
277 package undergoes thorough automated unit testing to ensure that data manipulations
278 are achieving the desired results. Having data manipulation code maintained in a

279 separate package that focuses on consistently providing properly summarized forms of
280 the most recent data has made maintaining the forecasting code itself much more
281 straightforward.

282 **Modeling and Forecasting**

283 Iterative near-term forecasting involves regularly updating a variety of different models
284 (Figure 1). Ideally, new models should be easy to incorporate to allow for iterative
285 improvements to the general modeling structure and approach. We use CI to update the
286 models and make new forecasts each time the modeling code changes and when new
287 data becomes available (Figure 1b). We use a modular plugin infrastructure to allow
288 new models to be easily added to the system. This approach treats each model as an
289 interchangeable black box; all models have access to the same input data and generate
290 the same structure for model outputs (Figure 2). Details of how to add a new model to
291 the system are provided in the core GitHub repository
292 (<https://github.com/weecology/portalPredictions/wiki/Adding-a-new-model>). During
293 each run of the forecasting code, all existing models are run and the standardized
294 outputs are combined into a single file to store the results of the different models'
295 forecasts. A weighted ensemble model is then added with weights based on how well
296 individual models fit the training data. This plugin infrastructure makes it easy to add
297 and compare very different types of models, from the basic time-series approaches
298 currently implemented to the more complex state-space and machine learning models
299 we hope to implement in the future. As long as a model script can load the provided
300 data and produce the appropriate output, it will be run and its results incorporated into
301 the rest of the forecasting system. This means that anyone can add a new model to the
302 existing system by: 1) creating their own copy of the project (typically by forking the
303 project on GitHub); 2) developing a new model; and 3) submitting a pull request to our
304 repository.

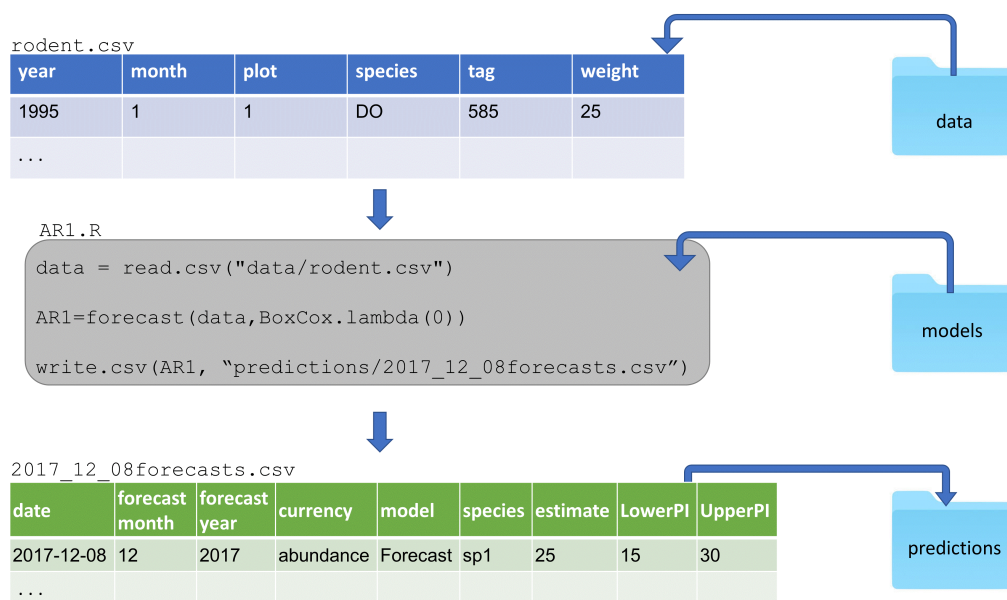


Figure 2: Demonstration of plugin infrastructure. All model scripts (represented here by the example AR1.R) are housed in a single folder. Each model script uses data provided by the core forecasting code (represented here by rodent.csv) and returns its forecast outputs in a predefined structure that is consistent across models (represented here by the example 2017_12_08forecasts.csv). Outputs from all models run on a particular date are combined into the same file (i.e. 2017_12_08forecasts.csv) to allow cross-model evaluations. Model output files are housed in a folder containing all forecast outputs from all previous dates to facilitate archiving and forecast assessment.

305 In addition to flexibility in what model structures can be supported, we also wanted to
306 support flexibility in what the models predict. Allowing models to make forecasts for
307 system properties ranging from individual species' population abundances to total
308 community biomass facilitates exploration of differences in forecastability across
309 different aspects of ecological systems. We designed a forecast output format to support
310 this. Each forecast output file contains the date being forecast, the collection date of the
311 data used for fitting the models, the model name, the date the forecast was made, the
312 state variable being forecast (e.g., rodent biomass, the abundance of a species), and the
313 forecast value and associated uncertainty of that forecast (Figure 2). This allows us to
314 store a variety of different forecasts in a common format and may serve as a useful
315 starting point for developing a standard for storing ecological forecasts more generally.

316 Forecasts are currently evaluated using root mean square error (RMSE) to evaluate
317 point forecasts and coverage to evaluate uncertainty. We plan to add additional metrics,
318 like deviance, that incorporate both accuracy and uncertainty and better match the
319 calibration method (Hooten & Hobbs, 2015; Dietze et al., 2018). In addition to
320 evaluating the actual forecasts, we also use hindcasting (forecasting on already collected
321 data; Jolliffe & Stephenson, 2003) to gain additional insight into the methods that work
322 best for forecasting this system. For example, a model is fit using rodent observations
323 up to June 2005, then used to make a forecast 12 months out to May 2006. The
324 observations of that 12-month period can immediately be used to evaluate the model.

325 Since hindcasting is conducted using data that has already been collected, it allows
326 model comparisons to be conducted on large numbers of hindcasts and provides insight
327 into which models make the best forecasts without needing to wait for new data to be
328 collected (Harris et al., 2018). It can also be used to quickly evaluate new models
329 instead of waiting for an adequate amount of data to accumulate. As the performance of
330 different models is understood through evaluation of forecasts and hindcasts, models
331 can be refined or removed from the system or ensemble to iteratively improve the

332 resulting forecasts.

333 **Archiving**

334 Publicly archiving forecasts before new data is collected allows the field to assess,
335 compare, and build on forecasts made by different groups (McGill, 2012; Tredennick et
336 al., 2016; Dietze et al., 2018; Harris et al., 2018) (Figure 1). Archiving serves as a form
337 of pre-registration for model predictions because the forecasts cannot be modified once
338 the data to assess them has been collected. This helps facilitate an unbiased
339 interpretation of model performance. To serve this role, archives should be publicly
340 accessible and be a permanent record that cannot be changed or deleted. This second
341 criterion means that GitHub is not sufficient for archival purposes because repositories
342 can be changed or deleted (Bergman, 2012; White, 2015). We explored three major
343 repositories for archiving forecasts: FigShare (<https://figshare.com/>), Zenodo
344 (<https://zenodo.org/>), and Open Science Framework (<https://osf.io/>). While all three
345 repositories allowed for easy manual submissions (i.e., a human uploading files after
346 each forecast), automating this process was substantially more difficult. Various
347 combinations of repositories, APIs (i.e., interfaces for automatically interacting with the
348 archiving websites), and associated R packages had issues with: 1) integrating
349 authorization with continuous integration; 2) automatically making archived files public;
350 3) adding new files to an existing location; or 4) automatically permanently archiving
351 the files. Our eventual solution was to leverage the GitHub-Zenodo integration
352 (<https://guides.github.com/activities/citable-code/>) and automatically push forecasts to a
353 GitHub repository from the CI server and release them via the GitHub API. The
354 GitHub-Zenodo integration is designed to automatically create versioned archives of
355 GitHub repositories. We created a repository for storing forecasts
356 (<https://github.com/weecology/forecasts>) and linked this repository with Zenodo (a
357 one-time manual process). Each time a new forecast is created, our pipeline adds the

new forecasts to the GitHub repository and uses the GitHub API to create a new “release” for that repository. This triggers the GitHub-Zenodo integration, which automatically archives the resulting forecasts under a top-level DOI that refers to all archived forecasts (<https://doi.org/10.5281/zenodo.839580>). Through this process, we automatically archive every forecast made with a documented time-stamp. In addition, we also archive the full state of the modeling and forecasting repository (<https://doi.org/10.5281/zenodo.833438>). Through a similar process, the raw data in the data repository is also archived on a Zenodo whenever data is added or changed (Yenni et al., 2018), allowing retrieval of older versions of the data used for forecasting (<https://doi.org/10.5281/zenodo.1219752>). This ensures that every forecast is fully reproducible since the exact code and data used to generate every forecast is preserved. Early forecasts from this system are archived in the modeling and forecasting code archive, not in the newer repository ‘forecasts’.

Presentation

Each month, we present our forecasts on a website that displays monthly rodent forecasts, model evaluation metrics, monthly reports, and information about the study site (Figure 3; <http://portal.naturecast.org>). The website includes a graphical presentation of the most recent month’s forecasts (including uncertainty) and compares the latest data to the previous forecasts. Information on the species and the field site are also included. The site is built using Rmarkdown (Allaire et al., 2017), which naturally integrates into the pipeline and is automatically updated after each forecast. The `knitr` R package (Xie, 2015) compiles the code into HTML, which is then published using Github Pages (<https://pages.github.com/>). The files for the website are stored in a subdirectory of the forecasting repository. As a result, the website is also archived automatically as part of archiving the forecast results.

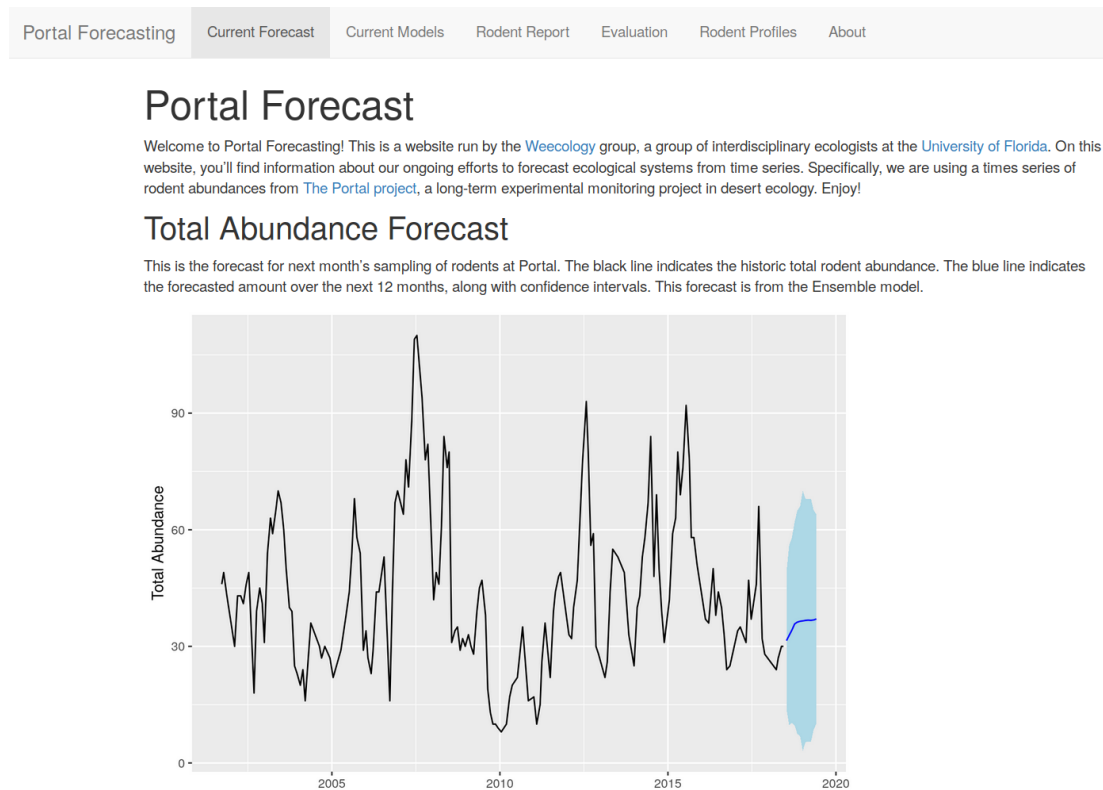


Figure 3: Screen capture of the homepage of the Portal Forecasting website (<http://portal.naturecast.org>). This site contains information on the most current forecasts, evaluation of forecast performance, and general information about the species being forecast.

Discussion

Following the recommendations of Dietze et al (2018), we developed an automated iterative forecasting system (Figure 1) to support repeated forecasting of an ecological system. Our forecasting system automatically acquires and processes the newest data, updates the models, makes new forecasts, publicly archives those forecasts, and presents both the current forecast and information on how previous forecasts performed. Every week, the forecasting system generates a new set of forecasts with no human intervention, except for the entry of new field data. Our forecasting system ensures that forecasts based on the most recent data are always available and is designed to allow rapid assessment of the performance of multiple forecasting models for a number of different states of the system, including the abundances of individual species and community-level variables such as total abundance. To create this iterative near-term forecasting system, we used R to process data and conduct analyses and leveraged existing tools and services (i.e. GitHub, Travis, Docker) for more complicated cyberinfrastructure tasks. Thus, our approach to developing iterative near-term forecasting infrastructure provides an example for how short-term ecological forecasting systems can be developed.

We designed this forecasting system with the goal of making it relatively easy to build, maintain, and extend. We used existing technology for both running the pipeline and building individual components, which allowed us to build the system relatively cheaply in terms of both time and money. This included the use of tools like Docker for reproducibility, Travis CI continuous integration for automatically running the pipeline, Rmarkdown and `knitr` for generating the website, and the already existing integration between Github and Zenodo to archive the forecasts. By using this “continuous analysis” approach (Beaulieu-Jones & Greene, 2017), where analyses are automatically rerun when changes are made to data, models, or associated code, we have reduced the time required by scientists to run and maintain the forecasting pipeline. To make the system

410 extensible so that new models could be easily incorporated, we used a plugin-based
411 infrastructure so that adding a new model to the system is as easy as adding a single file
412 to the ‘models’ folder in our repository (Figure 2). This should substantially lower the
413 barriers to other scientists contributing models to this forecasting effort. We also
414 automatically archive the resulting forecasts publicly so that the performance of these
415 forecasts can be assessed by both us and other researchers as new data is collected. This
416 serves as a form of pre-registration by providing a quantitative record of the forecast
417 before the data being predicted was collected.

418 While building this system was facilitated by the use of existing technological solutions,
419 there were still a number of challenges in making existing tools work for automated
420 iterative forecasting. Continuous integration is designed primarily for running
421 automated tests on software, not for running a coordinated forecasting pipeline. As a
422 result, extra effort was sometimes necessary to figure out how to get these systems to
423 work properly in non-standard situations, like running code that was not part of a
424 software package. In addition, hosted continuous integration solutions, like Travis,
425 provide only limited computational resources. As the number and complexity of the
426 models we fit has grown, we have had to continually invest effort in reducing our total
427 compute time so we can stay within these limits. Finally, we found no satisfactory
428 existing solution for archiving our results. All approaches we tried had limitations when
429 it came to automatically generating publicly-versioned archives of forecasts on a
430 repeated basis, and our eventual solution was difficult to configure to such a degree that
431 it will remain an impediment for most researchers. Overall, we found existing
432 technology to be sufficient to the task of creating an iterative forecasting pipeline, but it
433 required greater expertise and a greater investment of time than is ideal. Additional tool
434 development to reduce the effort required for scientists to set up their own short-term
435 forecasting systems would clearly be useful. Our efforts, however, show that it is
436 possible to use existing tools to develop initial iterative systems as a method for both

437 advancing scientific understanding and developing proof of concept forecasting systems.

438 Expanding the community of ecological forecasters using continuous analysis

439 approaches will require both an expansion of the current toolkit and the development of

440 standards to facilitate interoperability of forecasts and forecasting systems. One of the

441 major challenges for our current forecasting system is supporting computationally

442 intensive forecasts. Projects involving larger datasets and/or complex modelling

443 approaches will require either hosted solutions that provide infrastructure for running

444 continuous integration and allow long-running distributed jobs or solutions that involve

445 the user setting up their own continuous analysis system on cloud infrastructure or high

446 performance computing centers. Event-driven serverless cloud platforms like

447 OpenWhisk (<https://openwhisk.apache.org/>) and AWS Lambda

448 (<https://aws.amazon.com/lambda/>) provide potential as hosted solutions, and open

449 source continuous integrations systems like Jenkins (<https://jenkins.io/>) can be

450 integrated with either cloud or high performance computing centers. However, both

451 solutions are currently more complicated to set up than the hosted continuous

452 integration approach we have employed using Travis. In addition to scalability issue for

453 more computationally intensive projects, the toolkit for continuous analysis needs be

454 made more researcher friendly. To broaden the user-base that can use continuous

455 analysis for forecasting, we recommend the development of tools that make setting up

456 continuous analysis easier by automating configuration steps. We also recommend the

457 development of tools or data repository infrastructure to support the easy automated

458 archiving of regularly generated data and forecasts (see Yenni et al., 2018). Finally, the

459 development of standards for ecological forecasting to allow interoperability among

460 forecasting systems will be essential for the growth of this field (see discussions of data

461 standards, meta-data, and ontologies in ecology more broadly Jones et al., 2006; Madin

462 et al., 2008; Michener & Jones, 2012). Now is an opportune time for developing these

463 standards while the community of ecological forecasters is still small. While we have

464 developed an initial format for storing and sharing forecasts, it is still lacking in several
465 areas. Most notably, our approach to storing information on models and their associated
466 uncertainty is insufficient for all but the simplest models. Improving this framework
467 will require capturing both covariances between state variables and the full uncertainty
468 in the models, by either storing full model objects or additional information like full
469 ensembles of predictions (e.g., from Monte Carlo based approaches). This is
470 challenging due to a lack of general standards for reporting uncertainty (Dietze et al.,
471 2018).

472 Because of the breadth of expertise needed to set up our forecasting pipeline, our effort
473 required a team with diverse skills and perspectives, ranging from software
474 development to field site expertise. It is rare to find such breadth within a single
475 individual, and our system was developed as a collaboration between the lab collecting
476 and managing the data and a computational ecology lab. When teams have a breadth of
477 expertise, communication can be challenging (Winowiecki et al., 2011). We found a
478 shared base of knowledge related to both the field research and computational skills was
479 important for the success of the group. The two labs are part of a joint interdisciplinary
480 ecology group that has a mission of breaking down barriers between field and
481 computational/theoretical ecologists (<http://weecology.org>). Everyone on the team had
482 received training in fundamental data management and computing skills through a
483 combination of university courses, Software and Data Carpentry workshops (Teal et al.,
484 2015), and lab training efforts. In addition, everyone was broadly familiar with the
485 study site and methods of data collection, and most team members had participated in
486 field work at the site on multiple occasions. This provided a shared set of knowledge
487 and vocabulary that actively facilitated interdisciplinary interactions. All members of
488 the team actively participated in the development of the forecasting pipeline. Given the
489 current state of tools for automated iterative forecasting, forecasting teams require some
490 experience in working with continuous integration and APIs. This means either

interdisciplinary teams or additional training will often be required for creating these pipelines until tool development improves. To improve the success of these diverse groups, we believe efforts at providing ‘team science’ training to scientists interested in forecasting will be beneficial for the success of iterative forecasting attempts for the foreseeable future (Read et al., 2016).

We developed infrastructure for automatically making iterative forecasts with the goals of making accurate forecasts for this well-studied system, learning what methods work well for ecological forecasting more generally, and improving our understanding of the processes driving ecological dynamics. The most obvious application of automated iterative ecological forecasting is for speeding up development of forecasting models by using the most recent data available and by quickly iterating to improve the models used for forecasting. By learning what works best for forecasting in this and other ecological systems, we will better understand what the best approaches are for ecological forecasting more generally. By designing the pipeline so that it can forecast many different aspects of the ecological community, we also hope to learn about what aspects of ecology are more forecastable. Finally, automated forecasting infrastructures like this one also provide a core foundation for faster scientific inquiry because new models can quickly be applied to data and compared to existing models. The forecasting infrastructure does the time-consuming work of data processing, data integration, and model assessment, allowing new research to focus on the models being developed and the inferences about the system that can be drawn from them (Dietze et al., 2018). We plan to use this pipeline to drive future research into understanding the processes that govern the dynamics of individual populations and the community as a whole. By regularly running different models for population and community dynamics, a near-term iterative pipeline such as ours should also make it possible to rapidly detect changes in how the system is operating, which should allow the rapid identification of ecological transitions or even possibly allow them to be prevented (Pace et al., 2017). By building

518 an automated iterative near-term forecasting infrastructure, we can improve our ability
519 to forecast natural systems, understand the biology driving ecological dynamics, and
520 detect or even predict changes in system state that are important for conservation and
521 management.

522 **Acknowledgements**

523 We thank Henry Senyondo for help with continuous integration and Hao Ye for
524 discussions and feedback on the manuscript. We thank all of the graduate students,
525 postdocs, and volunteers who have collected the Portal Project over the last 40 years
526 and the developers of the software and tools that made this project possible. We thank
527 Heather Bradley for all of her logistical support that made this research possible. This
528 research was supported by the National Science Foundation through grant 1622425 to
529 S.K.M. Ernest and by the Gordon and Betty Moore Foundation's Data-Driven
530 Discovery Initiative through grant GBMF4563 to E.P. White.

531 **Data Accessibility**

532 The data used in this study is from the Portal Project and is openly available (CC0) on
533 GitHub (<https://github.com/weecology/PortalData>) and archived on Zenodo (Ernest et
534 al. (n.d.)). Code for reproducing all analyses is available on GitHub
535 (<https://github.com/weecology/portalPredictions>) and archived on Zenodo (White,
536 Yenni, et al., 2018). Forecasts made by this system are all archived to Zenodo (White,
537 Bledsoe, et al., 2018).

538 **Author Contributions**

539 All authors conceived the ideas and designed methodology; All authors developed the
540 automated forecasting system; EPW and SKME led the writing of the manuscript. All
541 authors contributed critically to the drafts and gave final approval for publication.

542 **Box 1. Key practices for automated iterative near-term** 543 **ecological forecasting**

544 A list of some of the key practices developed by Dietze et al (2018) for facilitating
545 iterative near-term ecological forecasting and discussion of why these practices are
546 important.

547 **Data**

548 **1. Frequent data collection**

549 Frequent data collection allows models to be regularly updated and forecasts to be
550 frequently evaluated (Dietze et al., 2018). Depending on the system being studied, this
551 frequency could range from sub-daily to annual, but typically the more frequently the
552 data is collected the better.

553 **2. Rapid data release under open licenses**

554 Data should be released as quickly as possible (low latency) under open licenses so that
555 forecasts can be made frequently and data can be accessed by a community of
556 forecasters (Vargas et al., 2017; Dietze et al., 2018).

557 **3. Best practices in data structure**

558 To reduce the time and effort needed to incorporate data into models, best practices in
559 data structure should be employed for managing and storing collected data to ensure it
560 is easy to integrate into other systems (interoperability) (Borer et al., 2009; Strasser et
561 al., 2011; White et al., 2013).

562 **Models**

563 **4. Focus on uncertainty**

564 Understanding the uncertainty of forecasts is crucial to interpreting and understanding
565 their utility. Models used for forecasting should be probabilistic to properly quantify
566 uncertainty and to convey how this uncertainty increases through time. Evaluation of
567 forecast models should include assessment of how accurately they quantify uncertainty
568 as well as point estimates (Hooten & Hobbs, 2015; Harris et al., 2018).

569 **5. Compare forecasts to simple baselines**

570 Understanding how much information is present in a forecast requires comparing its
571 accuracy to simple baselines to see if the models yield improvements over the naive
572 expectation that the system is static (Harris et al., 2018).

573 **6. Compare and combine multiple modeling approaches**

574 To quickly learn about the best approaches to forecasting different aspects of ecology,
575 multiple modeling approaches should be compared (Harris et al., 2018). Different
576 modeling approaches should also be combined into ensemble models, which often
577 outperform single models for prediction (Weigel et al., 2008).

578 **Cyberinfrastructure**

579 In addition to improvements in data and models, iterative near-term forecasting requires
580 improved infrastructure and approaches to support continuous model development and

581 iterative forecasting (Dietze et al., 2018).

582 **7. Best practices in software development**

583 Best practices should be followed in the development of scientific software and
584 modeling to make it easier to maintain, integrate into pipelines, and build on by other
585 researchers. Key best practices include open licenses, good documentation, version
586 control, and cross-platform support (Wilson et al., 2014; Hampton et al., 2015).

587 **8. Support easy inclusion of new models**

588 To facilitate the comparison and ensembling of different modeling approaches, code for
589 fitting models and making forecasts should be easily extensible, to allow models
590 developed by different groups to be integrated into a single framework (Dietze et al.,
591 2018).

592 **9. Automated end-to-end reproducibility**

593 Each forecast iteration involves acquiring new data, updating the models, and making
594 new forecasts. This should be done automatically without requiring human intervention.
595 Therefore, the process of making forecasts should emphasize end-to-end reproducibility,
596 including data, models, and evaluation (Stodden & Miguez, 2014), to allow the
597 forecasts to be easily rerun as new data becomes available (Dietze et al., 2018).

598 **10. Publicly archive forecasts**

599 Forecasts should be openly archived to demonstrate that the forecasts were made
600 without knowledge of the outcomes and to allow the community to assess and compare
601 the performance of different forecasting approaches both now and in the future (McGill,
602 2012; Tredennick et al., 2016; Dietze et al., 2018; Harris et al., 2018). Ideally, the
603 forecasts and evaluation of their performance should be automatically posted publicly in
604 a manner that is understandable by both scientists and the broader stakeholder
605 community.

606 **Box 2. Glossary of terms**

607 **CI.** ‘Continuous Integration.’ The practice of continuously building and testing a code
608 base as it is developed. **Data latency.** The time it takes for data to be available for use.
609 **Docker.** An open-source Linux program for containerization (see software container).
610 **git.** An open-source version control system. **GitHub.** A web-based host for git projects.
611 Other options for a similar service include GitLab or Bitbucket. **PortalData.** The git
612 repository for the Portal data, found on GitHub. **portalPredictions.** The git repository
613 for the forecasts made using Portal data, found on GitHub. **portalr.** An R package for
614 using the Portal data. **QA/QC.** ‘Quality Assurance.’ Testing the quality of a product.
615 ‘Quality Control.’ The process of ensuring the quality of a product. **Rocker.** A project
616 making it easy to use Docker containers in the R environment. **Software container.**
617 Allows a developer to package up an application with all of the parts it needs to run
618 reliably. **testthat.** R package used to set up automated testing for QA/QC. **Travis.** A
619 continuous integration service that integrates easily with GitHub and R. Examples of
620 similar programs are Jenkins or CodeShip. **Unit test.** A component of quality control in
621 which each smallest testable part of software is formally tested. **Zenodo.** An open data
622 archive that integrates easily with GitHub.

623 **References**

- 624 Allaire, J., Cheng, J., Xie, Y., McPherson, J., Chang, W., Allen, J., . . . Arslan, R. (2017).
625 *Rmarkdown: Dynamic documents for r*. Retrieved from
626 <https://CRAN.R-project.org/package=rmarkdown>
- 627 Araujo, M. B., & New, M. (2007). Ensemble forecasts of species distributions. *Trends*
628 *in Ecology and Evolution*, 22(1), 42–47.
- 629 Bauer, P., Thorpe, A., & Brunet, G. (2015). The quiet revolution of numerical weather

630 prediction. *Nature*, 525(7567), 47.

631 Beaulieu-Jones, B. K., & Greene, C. S. (2017). Reproducibility of computational
632 workflows is automated using continuous analysis. *Nature Biotechnology*, 35(4),
633 342–346.

634 Bergman, C. (2012). On the preservation of published bioinformatics code on github.
635 Retrieved from [https://caseybergman.wordpress.com/2012/11/08/](https://caseybergman.wordpress.com/2012/11/08/on-the-preservation-of-published-bioinformatics-code-on-github/)
636 on-the-preservation-of-published-bioinformatics-code-on-github/

637 Boettiger, C. (2015). An introduction to docker for reproducible research. *ACM*
638 *SIGOPS Operating Systems Review*, 49(1), 71–79.

639 Boettiger, C., & Eddelbuettel, D. (2017). An introduction to rocker: Docker containers
640 for r. *arXiv Preprint arXiv:1710.03675*.

641 Borer, E. T., Seabloom, E. W., Jones, M. B., & Schildhauer, M. (2009). Some simple
642 guidelines for effective data management. *The Bulletin of the Ecological Society of*
643 *America*, 90(2), 205–214.

644 Brown, J. H. (1998). The desert granivory experiments at portal. *Experimental Ecology*.
645 *Oxford University Press, Oxford, UK*, 71–95.

646 Clark, J. S., Carpenter, S. R., Barber, M., Collins, S., Dobson, A., Foley, J. A., . . . others.
647 (2001). Ecological forecasts: An emerging imperative. *Science*, 293(5530), 657–660.

648 Dietze, M. C. (2017). *Ecological forecasting*. Princeton University Press.

649 Dietze, M. C., Fox, A., Beck-Johnson, L. M., Betancourt, J. L., Hooten, M. B.,
650 Jarnevich, C. S., . . . White, E. P. (2018). Iterative near-term ecological forecasting:
651 Needs, opportunities, and challenges. *Proceedings of the National Academy of Sciences*.
652 doi:10.1073/pnas.1710231115

653 Diniz-Filho, J. A. F., Bini, L. M., Rangel, T. F., Loyola, R. D., Hof, C., Nogues-Bravo,
654 D., & Araujo, M. B. (2009). Partitioning and mapping uncertainties in ensembles of

655 forecasts of species turnover under climate change. *Ecography*, 32(6), 897–906.

656 Díaz, S., Demissew, S., Carabias, J., Joly, C., Lonsdale, M., Ash, N., . . . others. (2015).
 657 The ipbes conceptual framework—connecting nature and people. *Current Opinion in*
 658 *Environmental Sustainability*, 14, 1–16.

659 Dormann, C. F., Calabrese, J. M., Guillerá-Arroita, G., Matechou, E., Bahn, V., Bartoń,
 660 K., . . . others. (2018). Model averaging in ecology: A review of bayesian,
 661 information-theoretic, and tactical approaches for predictive inference. *Ecological*
 662 *Monographs*.

663 Ernest, S. K. M., Brown, J. H., Thibault, K. M., White, E. P., & Goheen, J. R. (2008).
 664 Zero sum, the niche, and metacommunities: Long-term dynamics of community
 665 assembly. *The American Naturalist*, 172(6), E257–E269.

666 Ernest, S. K. M., Valone, T. J., & Brown, J. H. (2009). Long-term monitoring and
 667 experimental manipulation of a chihuahuan desert ecosystem near portal, arizona, usa.
 668 *Ecology*, 90(6), 1708–1708.

669 Ernest, S. K. M., Yenni, G. M., Allington, G., Christensen, E. M., Geluso, K., Goheen, J.
 670 R., . . . Valone, T. J. (2016). Long-term monitoring and experimental manipulation of a
 671 chihuahuan desert ecosystem near portal, arizona (1977–2013). *Ecology*, 97(4),
 672 1082–1082.

673 Ernest, S. M., Yenni, G. M., Allington, G., Bledsoe, E., Christensen, E., Diaz, R., . . .
 674 Valone, T. J. (2018). The portal project: A long-term study of a chihuahuan desert
 675 ecosystem. *bioRxiv*. doi:10.1101/332783

676 Ernest, S. M., Yenni, G. M., Allington, G., Bledsoe, E. K., Christensen, E. M., Diaz, R.,
 677 . . . Brown, J. H. (n.d.). Weecology/portaldata. doi:10.5281/zenodo.zenodo.1215988

678 Gage, K. L., & Kosoy, M. Y. (2005). Natural history of plague: Perspectives from more
 679 than a century of research. *Annu. Rev. Entomol.*, 50, 505–528.

680 Hampton, S. E., Anderson, S. S., Bagby, S. C., Gries, C., Han, X., Hart, E. M., ...
 681 others. (2015). The tao of open science for ecology. *Ecosphere*, 6(7), 1–13.

682 Harris, D. J., Taylor, S. D., & White, E. P. (2018). Forecasting biodiversity in breeding
 683 birds using best practices. *PeerJ*.

684 Hooten, M. B., & Hobbs, N. (2015). A guide to bayesian model selection for ecologists.
 685 *Ecological Monographs*, 85(1), 3–28.

686 Jolliffe, I. T., & Stephenson, D. B. (Eds.). (2003). *Forecast verification: a practitioner's*
 687 *guide in atmospheric science*. John Wiley; Sons, Ltd. Retrieved from
 688 <http://linkinghub.elsevier.com/retrieve/pii/S0169207005001214>

689 Jones, M. B., Schildhauer, M. P., Reichman, O., & Bowers, S. (2006). The new
 690 bioinformatics: Integrating ecological data from the gene to the biosphere. *Annu. Rev.*
 691 *Ecol. Evol. Syst*, 37, 519–44.

692 Kalnay, E. (2003). *Atmospheric modeling, data assimilation and predictability*.
 693 Cambridge university press.

694 Liboschik, T., Fokianos, K., & Fried, R. (2015). *Tscount: An r package for analysis of*
 695 *count time series following generalized linear models*. Universitätsbibliothek
 696 Dortmund.

697 Liu, G., Eakin, C. M., Chen, M., Kumar, A., De La Cour, J. L., Heron, S. F., ... Strong,
 698 A. E. (2018). Predicting heat stress to inform reef management: NOAA coral reef
 699 watch's 4-month coral bleaching outlook. *Frontiers in Marine Science*, 5, 57.

700 Luo, Y., Ogle, K., Tucker, C., Fei, S., Gao, C., LaDeau, S., ... Schimel, D. S. (2011).
 701 Ecological forecasting and data assimilation in a data-rich era. *Ecological Applications*,
 702 21(5), 1429–1442.

703 Madin, J. S., Bowers, S., Schildhauer, M. P., & Jones, M. B. (2008). Advancing
 704 ecological research with ontologies. *Trends in Ecology & Evolution*, 23(3), 159–168.

705 McGill, B. J. (2012). Ecologists need to do a better job of prediction – part ii – partly
 706 cloudy and a 20% chance of extinction (or the 6 p’s of good prediction). Retrieved from
 707 [https://dynamicecology.wordpress.com/2013/01/09/](https://dynamicecology.wordpress.com/2013/01/09/ecologists-need-to-do-a-better-job-of-prediction-part-ii-mechanism-vs-pattern/)
 708 [ecologists-need-to-do-a-better-job-of-prediction-part-ii-mechanism-vs-pattern/](https://dynamicecology.wordpress.com/2013/01/09/ecologists-need-to-do-a-better-job-of-prediction-part-ii-mechanism-vs-pattern/)

709 Merkel, D. (2014). Docker: Lightweight linux containers for consistent development
 710 and deployment. *Linux J.*, 2014(239). Retrieved from
 711 <http://dl.acm.org/citation.cfm?id=2600239.2600241>

712 Michener, W. K., & Jones, M. B. (2012). Ecoinformatics: Supporting ecology as a
 713 data-intensive science. *Trends in Ecology & Evolution*, 27(2), 85–93.

714 Morris, B. D., & White, E. P. (2013). The ecodata retriever: Improving access to
 715 existing ecological data. *PloS One*, 8(6), e65848.

716 NOAA. (2016). NOAA fisheries (2016) status of stocks 2016: Annual report to
 717 congress on the status of u.s. Fisheries (noaa fisheries, washington, dc). Retrieved from
 718 <https://doi.org/10.5281/zenodo.833438>

719 Pace, M. L., Batt, R. D., Buelo, C. D., Carpenter, S. R., Cole, J. J., Kurtzweil, J. T., &
 720 Wilkinson, G. M. (2017). Reversal of a cyanobacterial bloom in response to early
 721 warnings. *Proceedings of the National Academy of Sciences*, 114(2), 352–357.

722 Parmenter, R. R., Brunt, J. W., Moore, D. I., & Ernest, S. (1993). The hantavirus
 723 epidemic in the southwest: Rodent population dynamics and the implications for
 724 transmission of hantavirus-associated adult respiratory distress syndrome (hards) in the
 725 four corners region. *A Report for the Federal Centers for Disease Control and*
 726 *Prevention*.

727 Petchey, O. L., Pontarp, M., Massie, T. M., Kéfi, S., Ozgul, A., Weilenmann, M., ...
 728 others. (2015). The ecological forecast horizon, and examples of its uses and
 729 determinants. *Ecology Letters*, 18(7), 597–611.

730 Prank, M., Chapman, D. S., Bullock, J. M., Belmonte, J., Berger, U., Dahl, A., ...
 731 others. (2013). An operational model for forecasting ragweed pollen release and
 732 dispersion in europe. *Agricultural and Forest Meteorology*, 182, 43–53.

733 Read, E. K., O'Rourke, M., Hong, G., Hanson, P., Winslow, L. A., Crowley, S., ...
 734 Weathers, K. (2016). Building the team for team science. *Ecosphere*, 7(3).

735 Senyondo, H., Morris, B. D., Goel, A., Zhang, A., Narasimha, A., Negi, S., ... White,
 736 E. P. (2017). Retriever: Data retrieval tool. *The Journal of Open Source Software*, 2(19),
 737 451. doi:10.21105/joss.00451

738 Springer, Y. P., Hoekman, D., Johnson, P. T., Duffy, P. A., Hufft, R. A., Barnett, D. T.,
 739 ... others. (2016). Tick-, mosquito-, and rodent-borne parasite sampling designs for the
 740 national ecological observatory network. *Ecosphere*, 7(5), e01271.

741 Stodden, V., & Miguez, S. (2014). Best practices for computational science: Software
 742 infrastructure and environments for reproducible and extensible research. *Journal of*
 743 *Open Research Software*, 2(1).

744 Strasser, C., Cook, R., Michener, W., Budden, A., & Koskela, R. (2011). Promoting data
 745 stewardship through best practices. In *Proceedings of the environmental information*
 746 *management conference 2011 (eim 2011)*. Oak Ridge National Laboratory (ORNL).

747 Stumpf, R. P., Tomlinson, M. C., Calkins, J. A., Kirkpatrick, B., Fisher, K., Nierenberg,
 748 K., ... Wynne, T. T. (2009). Skill assessment for an operational algal bloom forecast
 749 system. *Journal of Marine Systems*, 76(1-2), 151–161.

750 Tallis, H. M., & Kareiva, P. (2006). Shaping global environmental decisions using
 751 socio-ecological models. *Trends in Ecology & Evolution*, 21(10), 562–568.

752 Teal, T. K., Cranston, K. A., Lapp, H., White, E., Wilson, G., Ram, K., & Pawlik, A.
 753 (2015). Data carpentry: Workshops to increase data literacy for researchers.
 754 *International Journal of Digital Curation*, 10(1), 135–143.

755 Tredennick, A. T., Hooten, M. B., Aldridge, C. L., Homer, C. G., Kleinhesselink, A. R.,
 756 & Adler, P. B. (2016). Forecasting climate change impacts on plant populations over
 757 large spatial extents. *Ecosphere*, 7(10).

758 Vargas, R., Alcaraz-Segura, D., Birdsey, R., Brunsell, N. A., Cruz-Gaistardo, C. O.,
 759 Jong, B. de, ... others. (2017). Enhancing interoperability to facilitate implementation
 760 of redd+: Case study of mexico. *Carbon Management*, 8(1), 57–65.

761 Weigel, A. P., Liniger, M., & Appenzeller, C. (2008). Can multi-model combination
 762 really enhance the prediction skill of probabilistic ensemble forecasts? *Quarterly*
 763 *Journal of the Royal Meteorological Society*, 134(630), 241–260.

764 White, E. P. (2015). Some thoughts on best publishing practices for scientific software.
 765 *Ideas in Ecology and Evolution*, 8(1).

766 White, E. P., Baldridge, E., Brym, Z. T., Locey, K. J., McGlinn, D. J., & Supp, S. R.
 767 (2013). Nine simple ways to make it easier to (re) use your data. *Ideas in Ecology and*
 768 *Evolution*, 6(2).

769 White, E. P., Bledsoe, E. K., Christensen, E. M., Harris, D. J., Simonis, J. L., Taylor, S.
 770 D., ... Ernest, S. K. M. (2018, February). Weecology/forecasts: 2018-02-17.
 771 doi:10.5281/zenodo.839580

772 White, E. P., Yenni, G. M., Taylor, C., Shawn D and, Bledsoe, E. K., Simonis, J. L., &
 773 Ernest, S. K. M. (2018, February). Weecology/portalPredictions 2018-02-16.
 774 doi:10.5281/zenodo.833438

775 Wickham, H. (2011). Testthat: Get started with testing. *The R Journal*, 3, 5–10.
 776 Retrieved from
 777 http://journal.r-project.org/archive/2011-1/RJournal_2011-1_Wickham.pdf

778 Wickham, H. (2017). *Tidyverse: Easily install and load the 'tidyverse'*. Retrieved from
 779 <https://CRAN.R-project.org/package=tidyverse>

780 Wilson, G., Aruliah, D. A., Brown, C. T., Hong, N. P. C., Davis, M., Guy, R. T., ...
781 others. (2014). Best practices for scientific computing. *PLoS Biology*, 12(1), e1001745.

782 Winowiecki, L., Smukler, S., Shirley, K., Remans, R., Peltier, G., Lothes, E., ...
783 Alkema, L. (2011). Tools for enhancing interdisciplinary communication.
784 *Sustainability: Science, Practice and Policy*, 7(1), 74–80.

785 Xie, Y. (2015). *Dynamic documents with r and knitr* (Vol. 29). CRC Press.

786 Yenni, G. M., Christensen, E. M., Bledsoe, E. K., Supp, S. R., Diaz, R. M., White, E. P.,
787 & Ernest, S. M. (2018). Developing a modern data workflow for living data. *bioRxiv*,
788 344804.