The Journal of Open Source Software

hatchR: A toolset to predict hatch and emergence phenology in wild fishes

Morgan M. Sparks[1], **Eli Felts**[2], and **Bryan M. Maitland**[1]

1 US Forest Service, Rocky Mountain Research Station, Boise, ID, USA **2** US Fish and Wildlife Service, Idaho Fish and Wildlife Conservation Office, Orofino, ID, USA

DOI:

Software

- Review
- Repository
- Archive

Submitted:
Published:

Abstract

Understanding the timing of key life history events is necessary for managing and conserving populations. Historically, models to predict hatch and emergence timing for fishes were difficult to employ in wild settings because average incubation temperature was needed as the primary parameter in predictive models. However, recent improvements to these techniques reworked models such that they could be applied in wild environments as long as users had data for when adult fish spawned and a record of average daily temperature over the course of development. Despite these improvements, their application remains limited due to few parameterizations for varying species, being largely limited to salmonids. Here we present `hatchR`, a software ecosystem that allows users to predict hatch and emergence timing for wild fishes, as well as additional tools to aid in those analyses. `hatchR` allows users to leverage popular historic parameterizations for phenological models or to easily implement custom parameterizations using data not included in the package. `hatchR` is also distributed in two forms—an open source R package for maximum customization, as well as an HTML graphical-user-interface web application for individuals not familiar with scripting languages. To demonstrate potential uses, we present two case studies as likely applications for this software. `hatchR` promises to open many exciting avenues in research and management of fishes during their early life history.

Introduction

As primarily poikilothermic organisms, the development and growth of fishes is tightly linked with the temperature of their ambient environment. This close relationship has allowed researchers to generate statistical models that allow the prediction of developmental phenology with high accuracy and precision. These models were typically developed in aquaculture settings and their initial formulations were not applicable to wild populations because they assumed a constant temperature over the course of development (Alderdice & Velsen, 1978; Beacham & Murray, 1990; McPhail & Murray, 1979). However, Sparks et al. (2019) reformulated this approach as an "Effective Value model", in which the input was daily average temperature after a parent spawned and fish would either hatch or emerge when effective values cumulatively summed to one.

The resulting effective value approach has now been widely applied in Salmonids for which parameterizations from aquaculture were readily available—for example Pacific Salmon (*Oncorhynchus spp.*) models developed by Beacham & Murray (1990) have been applied to various species and populations (Adelfio, Wondzell, Mantua, & Reeves, 2019, 2024; Kaylor et al., 2021) while models developed for Bull Trout (*Salvelinus confluentus*) by McPhail & Murray (1979) were extended by Austin, Essington, & Quinn (2019). Despite growing popularity, applications have been largely limited within Salmonids, presumably because parameterizations for such models already existed due to their wide use in aquaculture and their general popularity as sport and commercial fish.

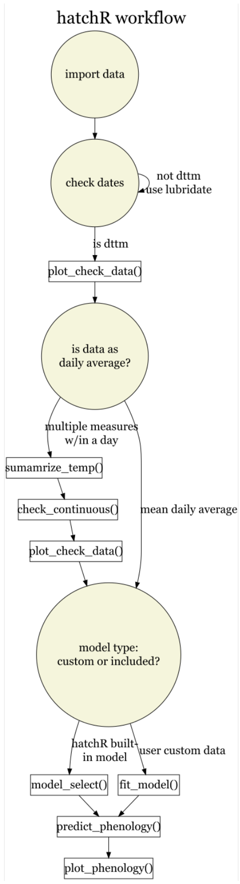
To bridge the gap between the application of one-off effective value model applications within individual studies and the lack of parameterization for other species, we developed the software ecosystem, `hatchR`. Specifically, `hatchR` allows users to input standard raw or summarized temperature datasets that are commonly collected in wild settings, run basic checks on those data, use built-in parameterizations like those from Beacham & Murray (1990) or Sparks, Westley, Falke, & Quinn (2017), develop custom models from their own or published temperature and phenological data, and predict hatch and emergence timing using these models in the effective value framework.

To widen the user application of these methods, we distribute two user-interfaces for `hatchR`. The first is a R package distributed via CRAN that allows users the most customizable application for these methods. The R package is especially powerful as it allows users to automate their analyses over multiple variables such as phenology type, multiple spawn dates, or different habitats with varying thermal regimes. These variable approaches are outlined in the package documentation on `hatchR`'s website. Alternatively, we also distribute a Shiny application in the form of an HTML-based web tool to interact with many of `hatchR`'s functions in a graphical-user-interface. The Shiny form trades-off some of automative power for user simplicity, while still allowing users to leverage much of the functionality of `hatchR`'s R package. Below, we present the basic overview of the software and multiple case studies of how it may be applied.

```
##
## Attaching package: 'ggdist'

## The following objects are masked from 'package:ggridges':
##
##     scale_point_color_continuous, scale_point_color_discrete,
##     scale_point_colour_continuous, scale_point_colour_discrete,
##     scale_point_fill_continuous, scale_point_fill_discrete,
##     scale_point_size_continuous
```

Package Overview

`hatchR` is meant to primarily be a tool for predicting phenology. In this sense, we mostly limit functionality to these applications and provide minimal data checking and plotting help. This decision is in part driven by the diversity of data types that users may import and the difficulty in addressing all those data types with respect to various data checks. In other words, we expect users to know their data better than we do and to check it accordingly. We do provide two basic data check functions discussed in the Checking Data section. Similarly, we provide limited functionality for plotting results, but provide examples of how to build custom visualization from output, specifically in R. For the Shiny application, we provide a base output plot, but the ability to download your results for custom plotting in programs of the user's choice. Finally, we provide brief summaries of general applications of `hatchR` below, but encourage users to visit articles hosted on the software webpage that extensively outline primary functions and applications, especially automating the application of predicting phenology across multiple variables.



Effective value models

Effective value models were created by Sparks et al. (2019) to implement developmental models in wild environments for Sockeye Salmon (*O. nerka*). The need for their development arose because historic models, specifically those in Beacham & Murray (1990), only considered the average incubation temperature during development and, for wild fishes, average incubation temperature was impossible to estimate because it was unknown when fish hatched even if adult spawn timing was known. To address this, Sparks et al. (2019) used the reciprocal of the formulation of model 2 from Beacham & Murray (1990) and assigned an effective value for every day of development using the daily average temperature.

The model follows the general format of:

$$EffectiveValue_i = 1/exp(log_e a - log_e(Temperature_i - b))$$

Where i is the daily value and a fish hatches or emerges when the cumulative sum of effective values reaches one:

$$\sum_{i=1}^{n} EffectiveValue_i = 1$$

The effective value model framework is the basis for the phenological models in `hatchR`, both in the included `model_table` in the package (though `model_table` includes more complex models developed by Beacham & Murray (1990)), as well as for custom models users can fit with `fit_model()`. Specifically, `model_table` has includes parameterizations from Beacham & Murray (1990), Sparks et al. (2017), and Austin et al. (2019) (who extended McPhail & Murray (1979)).

Data format

Water temperature datasets collected for wild environments are often either 1.) already summarized by day (*i.e.*, mean daily temperature) or, 2.) in a raw format from something like a HOBO TidbiT where readings are taken multiple times per day, which can be summarized into a mean daily temperatures. Alternatively, new statistical models like that of Siegel, Fullerton, FitzGerald, Holzer, & Jordan (2023) could be similarly implemented.

Fundamentally, `hatchR` assumes you have input data with two columns: a date column, giving the date (and often time) of a temperature measurement, and a temperature column, giving the associated temperature measurement (in centigrade). Other columns are okay to include, but these two columns (with any column name—just *without* spaces) are required. We expect your data to look something like this:

date	temperature
2000-01-01	2.51
…	…
2000-07-01	16.32
…	…
2000-12-31	3.13

`hatchR` assumes you've checked for missing records or errors in your data as it *will function with gaps*, so it's important to go through the data checks discussed below, as well as your own validity checks. `hatchR` can use values down to freezing (e.g, 0 °C), which returns extremely small effective values, and time to hatch or emerge may be > 1 year. In these cases, we suggest users consider how much of that data type is reasonable with their data.

For users choosing to implement `hatchR` in program R, data can be imported from any format the user chooses, as long as users can eventually coerce their data into a `dataframe` or `tibble` format, in which each row represents a single record. For the Shiny application, users must have their data stored as a .csv (comma separated values) file for upload, which can easily be exported using datasheet software like Microsoft Excel or Google Sheets.

Checking Data

`hatchR` is built assuming data will be analyzed as daily average temperatures. Despite that assumption, raw data (*e.g.*, as outputted by HOBO loggers) can be used and `hatchR` includes functionality to summarize those data into a format that is usable, as well as provides functions for basic visual and programmatic data checks to make sure outliers or missing data are at least brought to user's attention.

We demonstrate the utility of the summary and check functions `summarize_temp()`, `plot_check_temp()`, and `check_continuous()` using a simulated year-long dataset with temperature readings every thirty minutes.

```r
# create date object for a year with 30 min reading intervals
dates <- seq(from = ymd_hms("2000-07-01 00:00:00"),
             to = ymd_hms("2001-06-30 23:59:59"), length.out = 17568)

# create empty dataframe
year_sim <- data.frame(matrix(NA, nrow = length(dates), ncol = 1))

# date column
```

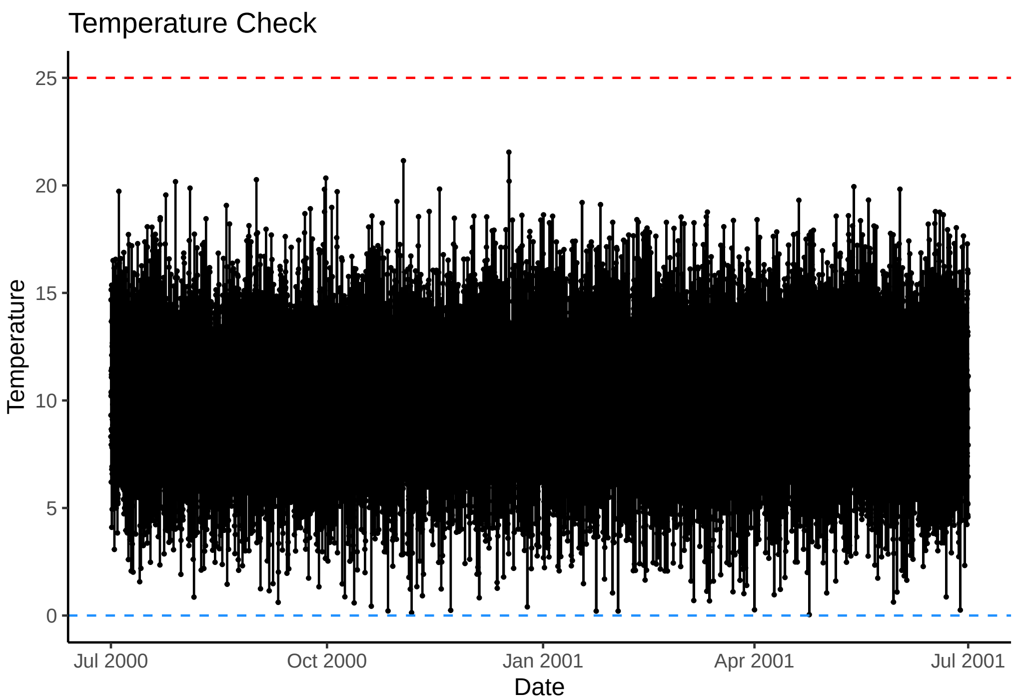
The Journal of Open Source Software

```r
colnames(year_sim)[1] <- "date"

# add dates vector to date column
year_sim[1] <- dates

#random seed
set.seed(123)

# take temps from a random normal dist with mean 10 sd 3
# for every date time combo in dates and append to column (temp) in year_sim
year_sim$temp <- rnorm(n = length(dates), mean = 10, sd = 3) %>%
  abs()

dim(year_sim)
```

```
## [1] 17568     2
```

First, we recommend checking imported data for any outliers or strange inputs using `plot_check_temp()`

```r
plot_check_temp(data = year_sim,
                dates = date,
                temperature = temp,
                temp_min = 0, # temp_min and max lines are
                              # user customizable
                temp_max = 25)
```



There are no obvious outliers but since each day has 48 records, we need to summarize it to daily mean temperature with `summarize_temp()` and then check for missing days with `check_continuous()`. We also recommend using `plot_check_temp()` again on the summarized data (though leave out the resulting plot for space efficiency in this manuscript).

```r
# summarize
year_sim_summ <- summarize_temp(data = year_sim,
                                dates = date,
                                temperature = temp)

# now a year's worth of single-day data
dim(year_sim_summ)
```

```
## [1] 365   2
```

```r
# check continuous (no errors)
check_continuous(data = year_sim_summ,
                 dates = date)
```

```
## i No breaks were found. All clear!
```

```r
# we can demonstrate an error by removing Oct. 8 (100th day)
check_continuous(data = year_sim_summ[-100,],
                 dates = date)
```

```
## Warning: ! Data not continuous
## i Breaks found at rows:
## i 100
```

```r
# it is useful to plot again to check your summarized data
plot_check_temp(data = year_sim_summ,
                dates = date,
                temperature = daily_temp,
                temp_min = 0, # temp_min and max lines are
                              # user customizable
                temp_max = 15)
```

Model Selection

talk about using nls for fit_model

model_table and fit_model

fit_model for three non-salmonid specieslib

Fitting models for other fishes

Below, we demonstrate how the `fit_model()` function may be used to create custom parameterizations for species beyond the Salmonids in the `model_table` included in the package. We include parameterizations from three warm-water species to demonstrate the `fit_model()` utility for fishes beyond the scope of the original effective value approach. We include parameterizations for commonly cultured sportfishes including Smallmouth Bass (*Micropterus dolomieu*), Channel Catfish (*Ictalurus punctatus*) from Small & Bates (2001), and Lake Sturgeon (*Acipenser fulvescens*) from Smith & King (2005).

We demonstrate the utility of this approach by creating a random thermal regime with an ascending thermograph with a mean temperature of 16 °C, parameterizing models for each species, and demonstrating days to hatch and developmental period for each species with the random thermal regime (Figure XXX).

```r
###  make temp regime
set.seed(123)
# create random temps and corresponding dates
temps_sim <- sort(rnorm(n =30, mean = 16, sd = 1), decreasing = FALSE)
```

```r
dates_sim <-  seq(from = ymd("2000-07-01"),
             to = ymd("2000-07-31"), length.out = 30)

data_sim <- matrix(NA, 30, 2) |> data.frame()
data_sim[,1] <- temps_sim
data_sim[,2] <- dates_sim

# change names so they aren't the same as the vector objects
colnames(data_sim) <- c("temp_sim", "date_sim")

### smallmouth mod
smallmouth <- matrix(NA, 10, 2) |> data.frame()
colnames(smallmouth) <- c("hours", "temp_F")
smallmouth$hours <- c(52, 54, 70, 78, 90, 98, 150, 167, 238, 234)
smallmouth$temp_F <- c(77, 75, 71, 70, 67, 65, 60, 59, 55, 55)

# change F to C and hours to days
smallmouth <- smallmouth |>
  mutate(days = ceiling(hours/24),
         temp_C = (temp_F -32) * (5/9))


smb_mod <- fit_model(temp = smallmouth$temp_C,
                     days = smallmouth$days,
                     species = "smb",
                     development_type = "hatch")


### catfish mod
catfish <- matrix(NA, 3, 2) |> data.frame()
colnames(catfish) <- c("days", "temp_C")
catfish$days <- c(16,21,26)
catfish$temp_C <- c(22,10,7)

cat_mod <- fit_model(temp = catfish$temp_C,
                     days = catfish$days,
                     species = "catfish",
                     development_type = "hatch")

### lake sturgeon mod
sturgeon <-  matrix(NA, 7, 2) |> data.frame()
colnames(sturgeon) <- c("days", "CTU")
sturgeon$days <- c(7,5,6,6,5,11,7)
sturgeon$CTU <- c(58.1, 62.2, 61.1, 57.5, 58.1, 71.4, 54.7)

sturgeon <- sturgeon |>
  mutate(temp_C = CTU/days) # change CTUs to average temp and add column

sturgeon_mod <- fit_model(days = sturgeon$days,
                          temp = sturgeon$temp_C,
                          species = "sturgeon",
                          development_type = "hatch")
```

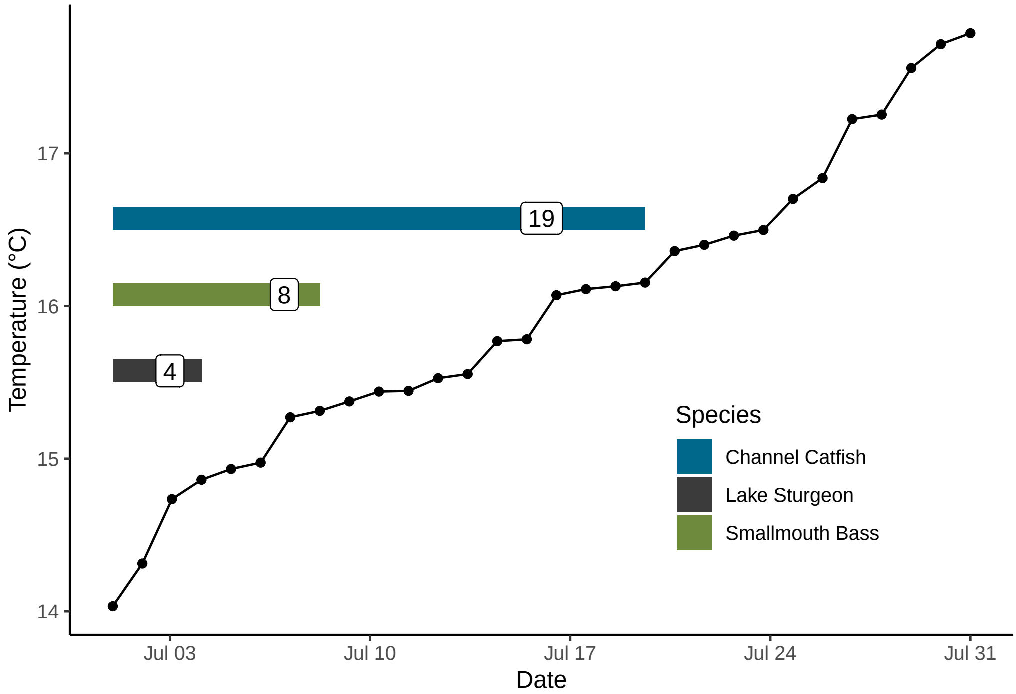
Note the model the R^2 fit from the models below. You can see the generally all preform well and are in line with values from model 2 of Beacham & Murray (1990).

```
#model fits
smb_mod$r_squared; cat_mod$r_squared; sturgeon_mod$r_squared
```

```
## [1] 0.9868067
```

```
## [1] 0.9433598
```

```
## [1] 0.9217358
```



Predicting Phenology and Output

To illustrate model selection and phenology prediction we will recreate a small portion of the analysis done by Sparks et al. (2019) using the `woody_island` dataset included in this package . We will predict both hatch and emergence timing, so we will obtain a model expression for each using `model_select()`.

```
sockeye_hatch_mod <- model_select(
  author = "Beacham and Murray 1990",
  species = "sockeye",
  model = 2,
  development_type = "hatch"
  )

sockeye_emerge_mod <- model_select(
  author = "Beacham and Murray 1990",
  species = "sockeye",
  model = 2,
  development_type = "emerge"
  )
```

We can now use our model expressions to predict when sockeye would hatch and emerge at Woody Island in 1990. First we predict hatch timing using `predict_phenology()`:

The Journal of Open Source Software

```r
WI_hatch <- predict_phenology(
  data = woody_island,
  dates = date,
  temperature = temp_c,
  spawn.date = "1990-08-18",
  model = sockeye_hatch_mod
  )
```

And then look inside the returned object to see days to hatch and development period:

```r
WI_hatch$days2done
```

```
## NULL
```

```r
WI_hatch$dev.period
```

```
##        start       stop
## 1 1990-08-18 1990-10-30
```

We can also do the same with emergence:

```r
WI_emerge <- predict_phenology(
  data = woody_island,
  dates = date,
  temperature = temp_c,
  spawn.date = "1990-08-18",
  model = sockeye_emerge_mod   # notice we're using emerge model expression here
  )

# see days to hatch and development period
WI_emerge$days2done
```

```
## NULL
```

```r
WI_emerge$dev.period
```

```
##        start       stop
## 1 1990-08-18 1991-03-09
```

Understanding your results

The output from `predict_phenology()` includes a lot of information. If we look at our `WI_hatch` object we see there are multiple elements stored in a list which can be accessed using the `$` operator.

```r
str(WI_hatch)
```

`WI_hatch$days2done` outputs the predicted days to hatch or emerge.

`WI_hatch$dev.period` is a 1x2 dataframe with the dates corresponding to when your fish's parent spawned (which you input with `predict_phenology(spawn.date = ...)`) and the date when the fish is predicted to hatch or emerge.

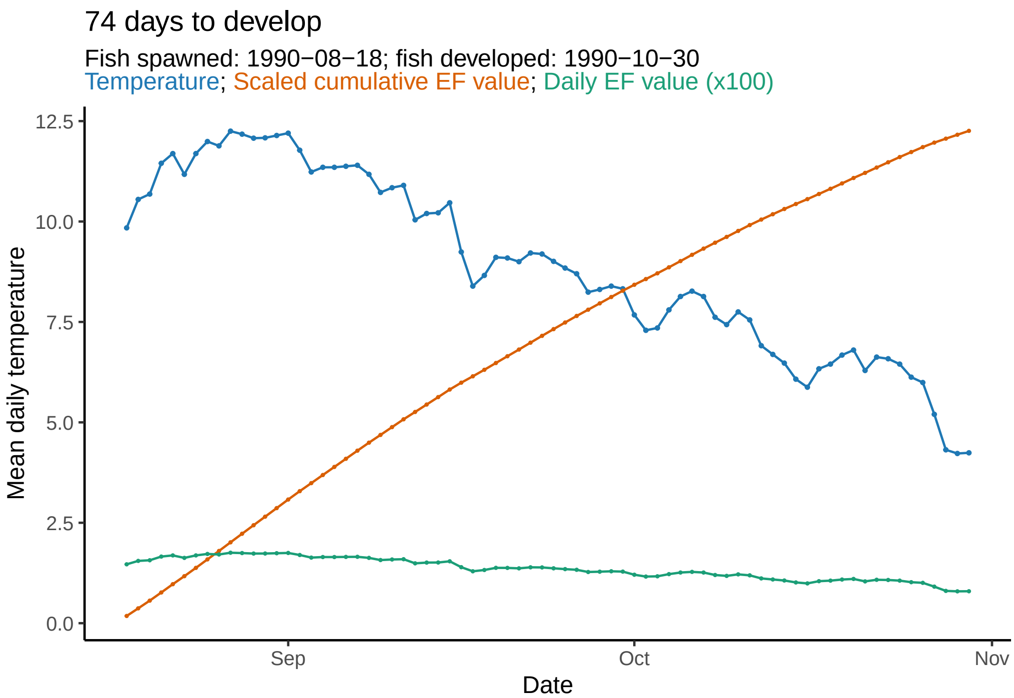
`WI_hatch$ef.vals` is a vector of each day's effective value as evaluated using whatever model is chosen.

`WI_hatch$ef.tibble` is a n x 4 tibble ($n =$ number of days to hatch or emerge) and the columns are the date, each day's temperature and effective value, and the cumulative sum of the effective values. The `ef.tibble` object is meant to serve as the basis for users to make custom figures for their data beyond the functionality we discuss below.

Plotting phenology

hatchR has a built in function, `plot_phenology()`, that allows users to visualize their phenology results. The plot visualizes three specific components: 1.)the temperature regime over which you are predicting, 2.) the cumulative sum of effective values, and 3.) the effective value for each day in your prediction span. The function allows you to output various figures based on your interests, but defaults to a figure with all information and the corresponding labels.

```
plot_phenology(WI_hatch)
```



Case Study 1

A common management scenario where developmental phenology might be useful would be trying to understand if fish might be free-moving before some management action. For instance, will have fish have emerged from redds when a stream section has been opened to grazing or bridge decommissioning will commence?

In this scenario, we will consider the grazing example and Bull Trout, a threatened fish in the United States under the Endangered Species Act (Nolfi, Melbihess, Fisher, & Ellis, 2024), and the East Fork Salmon River, a key Bull Trout population in the upper Salmon River watershed. The fisheries manager there wants to know if fish will likely be out of the gravel and free-swimming by June 1st. In this system, it is expected that Bull Trout will be done spawning by the end of September, so we'll consider the last possible spawn date as September 30th.

We demonstrate this first case study using the graphical user interface portion of the `hatchR` ecosystem found at https://elifelts.shinyapps.io/hatchR/. Users will first upload their data with the `Import Data` window, which requires them to select their file on their personal computer, provide the program with the columns corresponding for dates and temperatures, and then provide the format in which dates are coded (*e.g.*, year-month-day or day-month-year). Once data is uploaded the program automatically plots the user's data using `plot_check_temp()` in the background and provides them the outputted

The Journal of Open Source Software

graphical check. After uploading and checking data, the user switches to the `Model Phenology` window. In this circumstance, we use the preloaded parameterization for bull trout from Austin et al. (2019) with the `Existing` button for model selection, which the user selects with the various drop down options in the menu. After the model is selected, the user can choose multiple spawn dates from the interactive calendar provided. We show results for spawning for September 30th as indicated in the example above. Once dates are chosen, a table entry for each spawn date is outputted in the `Phenology Summaries` tab and corresponding plot with data from each spawn date in the `Timeline Plot` tab. Output from predicting phenology and the resulting figure are downloadable from their respective tabs. The process is demonstrated in full in Figure XXX, but the interface is described more completely in the Articles on `hatchR`'s website https://bmait101.github.io/hatchR/.

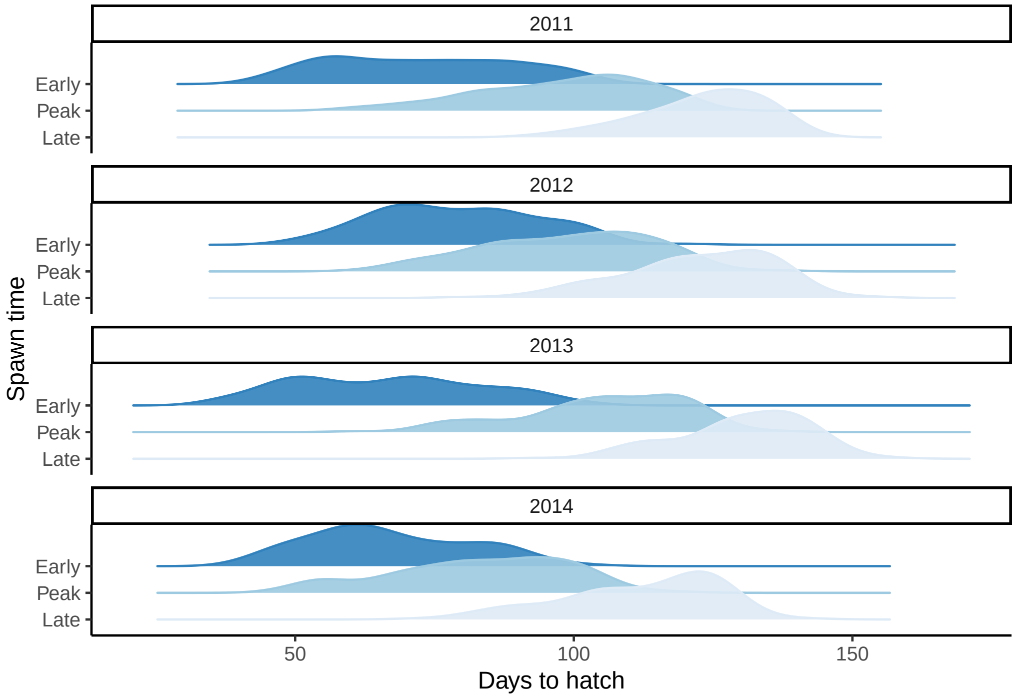
In this example we expect the last fish out of the gravel well before the June 1st date and the manager could allow grazing in this area without worrying about direct mechanical disturbance to fish developing in the gravel.

Case Study 2

For the second example, we will again use bull trout, but demonstrate a much more complex application for the purpose of showing the full flexibility of the programmatic application of **hatchR**. In this scenario we will use data from Isaak, Luce, Chandler, Horan, & Wollrab (2018) (included `idaho` dataset), which includes temperature data from 226 sites across the major upper Columbia River headwater watersheds in Idaho. For this approach we winnow putative bull trout spawning sites by filtering for sites with mean August temperature </= 13 °C in accordance with thresholds from Isaak, Young, Nagel, Horan, & Groce (2015). For the resulting 139 sites we will demonstrate predicting hatch timing in these putative Bull Trout spawning habitats.

We need to setup our models and data for this analysis, which we don't show those steps here for the sake of concision in this manuscript, however they are demonstrated in `paper.Rmd` included in the GitHub repository for **hatchR**. After the setup, we can easily map `predict_phenology()` across all putative spawning sites and three spawn dates (September 1-Early Spawning, September 15-Peak Spawning, and September 31-Late Spawning), the results of which are presented in Figure XXX.

```r
hatch_res <- isaak_summ_bt |>
  mutate(
    dev_period = map2(
      summ_obj, spawn_dates, # map across our site object and spawn dates
      predict_phenology,
      temperature = daily_temp,
      model = bt_hatch,
      dates = date
      ) |>
      map_df("dev.period") |> # pull out just dev.period results
      list()
    ) |>
  select(site, dev_period) |> # just select the columns we want
  unnest(cols = c(dev_period)) |> # unnest everything
  mutate(days_to_hatch = stop - start) # make a new column of days to hatch
```



Discussion

talk about how these models also represent local adaptation and heritable plasticity

talk about how these models represent point estimates and that emergence and hatch will take the form of a distribution

Acknowledgements

We thank Laura Koller for her help designing the `hatchR` logo. Dan Isaak provided useful discussion about model development and temperature datasets.

The views expressed in this manuscript are those of the authors and do not necessarily represent the views or policies of USFS or USFWS. Any mention of trade names, products, or services does not imply an endorsement by the U.S. government, USFS, or USFWS. USFS and USFWS do not endorse any commercial products, services or enterprises.

References

Adelfio, L. A., Wondzell, S. M., Mantua, N. J., & Reeves, G. H. (2019). Warm winters reduce landscape-scale variability in the duration of egg incubation for coho salmon (oncorhynchus kisutch) on the copper river delta, alaska. *Canadian Journal of Fisheries and Aquatic Sciences*, *76*(8), 1362–1375. doi:10.1139/cjfas-2018-0152

Adelfio, L. A., Wondzell, S. M., Mantua, N. J., & Reeves, G. H. (2024). Expanded, compressed, or equal? Interactions between spawning window and stream thermal regime generate three responses in modeled juvenile emergence for pacific salmon. *Canadian Journal of Fisheries and Aquatic Sciences*. doi:10.1139/cjfas-2023-0238

Alderdice, D. F., & Velsen, F. P. J. (1978). Relation between temperature and incubation time for eggs of chinook salmon (oncorhynchus tshawytscha). *Journal of the Fisheries Research Board of Canada*, *35*(1), 69–75. doi:10.1139/f78-010

Austin, C. S., Essington, T. E., & Quinn, T. P. (2019). Spawning and emergence phenology of bull trout Salvelinus confluentus under differing thermal regimes. *Journal of Fish Biology*, *94*(1), 191–195. doi:10.1111/jfb.13864

Beacham, T. D., & Murray, C. B. (1990). Temperature, egg size, and development of embryos and alevins of five species of pacific salmon: A comparative analysis. *Transactions of the American Fisheries Society*, *119*(6), 927–945. doi:10.1577/1548-8659(1990)119<0927:TESADO>2.3.CO;2

Isaak, D. J., Luce, C. H., Chandler, G. L., Horan, D. L., & Wollrab, S. P. (2018). Principal components of thermal regimes in mountain river networks. *Hydrology and Earth System Sciences*, *22*(12), 6225–6240. doi:10.5194/hess-22-6225-2018

Isaak, D. J., Young, M. K., Nagel, D. E., Horan, D. L., & Groce, M. C. (2015). The cold-water climate shield: delineating refugia for preserving salmonid fishes through the 21st century. *Global Change Biology*, *21*(7), 2540–2553. doi:10.1111/gcb.12879

Kaylor, M. J., Justice, C., Armstrong, J. B., Staton, B. A., Burns, L. A., Sedell, E., & White, S. M. (2021). Temperature, emergence phenology and consumption drive seasonal shifts in fish growth and production across riverscapes. *Journal of Animal Ecology*, *90*(7), 1727–1741. doi:10.1111/1365-2656.13491

McPhail, J., & Murray, C. (1979). *The early life-history and ecology of dolly varden (salvelimus malma) in the upper arrow lakes*. Nelson, BC, Canada.

Nolfi, D., Melbihess, T., Fisher, S., & Ellis, L. (2024). 5-Year Statuse Review Coterminous United States Population of Bull Trout (Salvelinus confluentus).

Siegel, J. E., Fullerton, A. H., FitzGerald, A. M., Holzer, D., & Jordan, C. E. (2023). Daily stream temperature predictions for free-flowing streams in the Pacific Northwest, USA. *PLOS Water*, *2*(8), e0000119. doi:10.1371/journal.pwat.0000119

Small, B. C., & Bates, T. D. (2001). Effect of Low-Temperature Incubation of Channel Catfish Ictalurus punctatus Eggs on Development, Survival, and Growth. *Journal of the World Aquaculture Society*, *32*(2), 189–194. doi:10.1111/j.1749-7345.2001.tb01094.x

Smith, K. M., & King, D. K. (2005). Dynamics and extent of larval lake sturgeon Acipenser fulvescens drift in the Upper Black River, Michigan. *Journal of Applied Ichthyology*, *21*(3), 161–168. doi:10.1111/j.1439-0426.2005.00623.x

Sparks, M. M., Falke, J. A., Quinn, T. P., Adkison, M. D., Schindler, D. E., Bartz, K., Young, D., et al. (2019). Influences of spawning timing, water temperature, and climatic warming on early life history phenology in western alaska sockeye salmon. *Canadian Journal of Fisheries and Aquatic Sciences*, *76*(1), 123–135. doi:10.1139/cjfas-2017-0468

Sparks, M. M., Westley, P. A. H., Falke, J. A., & Quinn, T. P. (2017). Thermal adaptation and phenotypic plasticity in a warming world: Insights from common garden experiments on Alaskan sockeye salmon. *Global Change Biology*, *23*(12), 5203–5217. doi:10.1111/gcb.13782