- **ANNOTATIONS AND CODE STRUCTURE**:

It is necessary that you include more annotations in the code to facilitate its review by referees or potential users of your model. Ait is better you do it as you develop the model, not at the end. You should place a description of each variable and parameter next to it. It would be good to describe the scheduling (go procedure; what does each action do?) Also explain how each procedure (submodel) works. Take a look, for example, at my model to see what I mean. You should also place together all submodels after the go procedure, and better in the same order as they’re there.

- **BREEDS**:

Consider getting rid of males and females breeds and just leave one breed called “trout”. Instead of using “ask males…”, you should use “ask trout with [sex = “male”], for example. Or is there a good reason to keep these breeds? (Economy of language –i.e., “males” being shorter than trout with [sex = “male”] – is not a reason ;))

In general, collectives should be included only when belonging to the collective affect s the agents conforming it, and viceversa, the agents affect the collective, which is not the case. This design element (collectives) is described in the ODD protocol as:

“In general, ABMs represent how systems are affected by the agents that make them up and how those agents are affected by each other and the system; but many ABMs also include collectives as an intermediate level of organization between the agents and the system. Collectives are defined in ODD as groups of agents that strongly affect both the agents that belong to them and the system, and that are strongly affected by the agents that belong to them. Collectives are often a consequence and means of cooperation among agents. Examples include groups of animals that live cooperatively (e.g., dog packs) and networks of people or businesses that cooperate.”

If you decide to keep the males/females breeds due to technical reasons, they shouldn’t be considered as collectives or kinds of agents different than “trout” (turtles in this case).

[AK done]

- **STATE VARIABLES/GLOBAL PARAMETERS**:

Some of the trout state variables should be modelled as global parameters because there’s no individual variability. If the value is the same for all individuals, then it’s a global parameter. If the difference is between sexes, then you can just have one value for each sex. From a pragmatic point of view, it’s computationally inefficient (takes up more memory, and thus slows down the model) to store hundreds or thousands of times the same value instead of only once. Examples here:

- Vp ,Va, Ve - mu\_cond V,\_cond (Do you plan to introduce some kind of variability in these parameters?)

- The matrix WM. In this case, you’ve to create two parameters: one matrix for the case when there’s no conflict (e.g., WM) and another for the case when there’s conflict (e.g., WMc). [Changed it]

- **HARD CODED VALUES FOR PARAMETERS**:

Parameters in equations or algorithms should be treated as such. However, there are still some values in the code that appear in equations and algorithms that are not defined as global parameters. In that way, you’d have to go to the piece of code of the procedure to change them instead of doing it at the beginning of the code; further, you cannot include them in sensitivity analyses, for instance. Examples:

- L, K, mass0 in the fecundity equation [AK done, these are now turtle variables]

- ifelse prop\_rivals > **0.8** [set quality start\_quality + **100** ] in the “sneaker” procedure [AK done, these are now part of the GUI as sliders]

- **SETUP**:

- I split up the code into different procedures to make it more readable.

- You have to initialize the trout state variables “habitat” and “mates” [Done]

- It’s better to include the initialization of all trout variables within the “sprout” command.

- **SUBMODELS**:

**Go**:

- I can’t really remember the reason why, but I was taught that it was better to place the “tick” command at the beginning of the go procedure, not at the end. [AK done]

- In the “ask turtles ...” command, consider moving the “mortality” procedure to the first action so that the proportion of fish that are going to die don’t have to perform the previous actions. [AK done]

- It’s better to group all actions taking place during reproduction. [Done]

- I think it’s more intuitive to use the state variable “habitat” than using the colour of the patch to create subsets of trout. It happens the same with the rest of procedures (e.g., “migrate”, “mortality”) [Changed]

- I understand that “days-since-child” represents the time since the female spawned, right? (It’s set to 0 in the “reproduce” procedure. Otherwise, what’s the difference with “age”?) The name it’s confusing, consider changing it. Besides, why all females update this variable every tick. Only females that have already spawned should do it, right? I think, it would be more convenient to use a Boolean variable indicating whether the female has already spawn or not that spawning season. At any case, is the condition that more than 365 day has to pass by to spawn again realistic? Or the important fact is that they have spawned or not that season?

- Why conditions for females to choose mates and to reproduce differ??? Why females that have already spawned (age > 365) choose mates???

- What’s “mate-count” used for? (In the “reproduce procedure” you use “if count mates...”.) There’s no need of it then. [Changed it]

**Grim-reaper**:

The procedure is not well formulated. Right now, the probability of dying is not updated when a trout dies. But, if a trout dies then the probability of dying of the next trout should be lower, not the same. In the current implementation, if the number of trout is 2 times the carrying capacity, then the probability of dying of any trout is 100%, so all trout would die. To solve it, you just have to make the procedure being in a turtle context instead of an observer context; in that way, every trout would calculate its own probability of dying (a function of how many trout have died before). [Done it in the go procedure]

**Mortality:**

- Mortality rules are the same for resident and anadromous trout contrary to that stated in the code. It differs between fish in the freshwaters and in the ocean; once, an anadromous fish comes back to the freshwaters, the rule is the same as for resident trout.

I want this to be habitat dependent rather than strategy dependent. Anadromous fish and resident should have the same mortality rate if they are both in freshwater.

- So far, there’s no age limit, so some trout could live for decades or centuries just by chance. Many models include a maximum age, over which the fish dies.

**Migrate:**

The order in which actions are performed is problematic: right now, the quality of already parasitized trout change every day (because they don’t change their patch in the ocean, they’re always in a patch with parasites). If it must be so, it’s ok. If the quality has to remain constant then include this code within the “if age > 365 and my-month = 1 and my-day = 1...” brackets.

**Choose mates:**

- You have omitted the interplay between resident and anadromous males here, right? Is the “sneaker” procedure the substitute?

- According to this formulation, males can mate indefinite times during each spawning season; even, several times the same day. Is that realistic? In my model, males can mate several times during the spawning season as long as they recover their body condition. In other models, they can spawn only once.

- The state variables of females “availa-males” and “max-mate-count” should be local variables. Do they use them for any specific action? [Changed it]

- Why do males track the identity of their female mates?

**Reproduction:**

- Remember that with the “hatch” command the new turtle inherits all its variables from the parent, so variables that should have a different value have to be initialized (e.g., mates). There’s no need of initializing those that must have the same value (e.g., size, habitat).

- So far, you’ve not included mortality linked to spawning. The probability of dying by poor condition typically increases after spawning.

**set-migratory-behaviour:**

- Better move this piece of code with the rest of submodels, as this procedure is used not only during setup but also every time step.

- WMT, GM\_WT should be local variables and not state variables. There’s no need of keeping track of them, right? Again, it takes up memory. Also, it takes time if every fish have to calculate the transpose of WM, so it’s better to make it a global parameter: in the setup you can directly defined WM as the transpose of the weight matrix. Each fish just have to choose the right matrix. [Changed it]

- The statement “ask males with [ anadromous = false] [set start\_quality quality]” is incorrect because you’re already in a trout context, so every trout is asking all anadromous males to set that variable. Instead it should read:

“if sex = "male" and anadromous = false [set start\_quality quality]”

[Changed it]

**Sneaker**:

- Variable “prop\_rivals” should be a local variable. [Changed it]

- Why do you need “start\_quality”? Will you make use of it for any output, graph, calculation, etc? I mean, you can just increase the “quality” variable. Unless you need to track the initial value of “quality”, you can get rid of that variable (start\_quality).

- If “quality” has to differ from “start\_quality” only when “prop\_rivals” is over 0.8 then there’s no need of using an “ifelse” statement.

**val-change reporter**:

I was not totally sure that it was working ok. In the end, it was reporting the value of “val”, which was the original value in the matrix. I preferred to change it.