**Using the Community General Ecology Model – Zero Dimension**

**Introduction**

**CGEM-0d is currently available only for the Microsoft Windows environment and has been tested in Windows 7 Enterprise and Windows 10.**

**Need to add an introduction here …**

* **what is this model, when was it created and by whom, what’s it for, who should be interested**
* **What is R Shiny and why are we using it**
* **what is this document and what is its purpose**
* **what’s contained in this document.**

**Table of Contents (ultimately set up with hyperlinks to sections)**

**Obtaining the Software**

**What’s contained in the install? Perhaps a table with 2 columns … filename, purpose of file, unless that would be unwieldy.**

**Preparing to Run the Model**

**FishTank Tutorial Exercise 1.0**: ***An introduction with emphasis on phytoplankton.***

**Foreward:**

Thank you for beta testing the FishTank tutorial. It is a work in progress and we need your feedback on all issues including the instructions, the interface, and the ecology. You may email [lowe.lisa@epa.gov](mailto:lowe.lisa@epa.gov) with suggestions, comments, feature requests, and bug reports if you do not have access to the GitHub repository. For those with GitHub access, please post issues, comments, and results on the **Issues** and **Wiki** sections of the GitHub.

**Obtaining the Software**

**The model source code, executables, and documentation can be obtained from EPA’s GitHub account in the CGEM repository, which is located at https://github.com/USEPA/CGEM. To download and run CGEM, users will need to install several free computer software packages that support the model. These include the GitHub Desktop Application, R and R Studio, which can be obtained as below.**

|  |  |
| --- | --- |
| **Software Package** | **URL for Installation** |
| **GitHub Desktop Application** | <https://help.github.com/desktop/guides/getting-started/installing-github-desktop/> |
| **R** | **https://cran.r-project.org/bin/windows/base/** |
| **R Studio** | **<https://www.rstudio.com/products/rstudio/download/>** |

**This section includes instructions for obtaining CGEM from the Github repository.**

**General Instructions**: ***Only for Windows, and instructions were tested on Windows 10***:

1. The GitHub repository is here:

Go to: <https://github.com/USEPA/Predicting-Ecosystem-Response-and-Recovery-to-Nutrients>

If clicking on the above doesn’t work, copy and paste it instead.

* **Instructions to get the code using GitHub:**

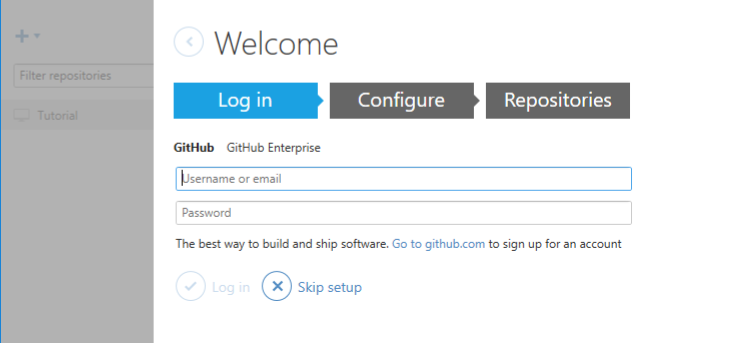
Follow the installation instructions for GitHub Desktop:

<https://help.github.com/desktop/guides/getting-started/installing-github-desktop/>

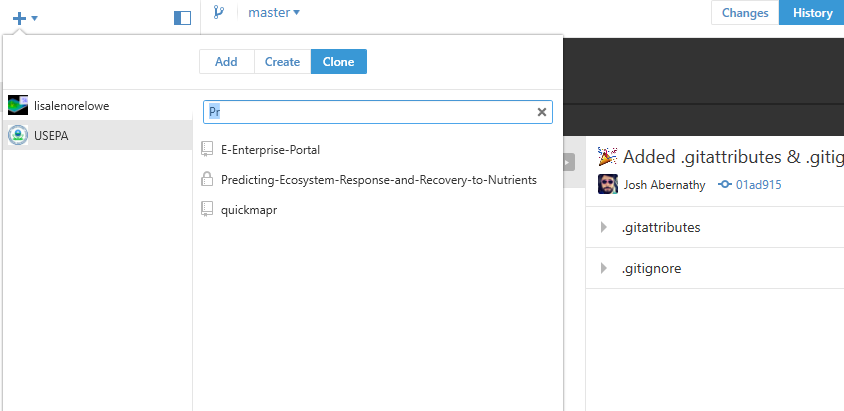
After installation, it should prompt you to run. If not, click this icon that the installer put on your desktop:



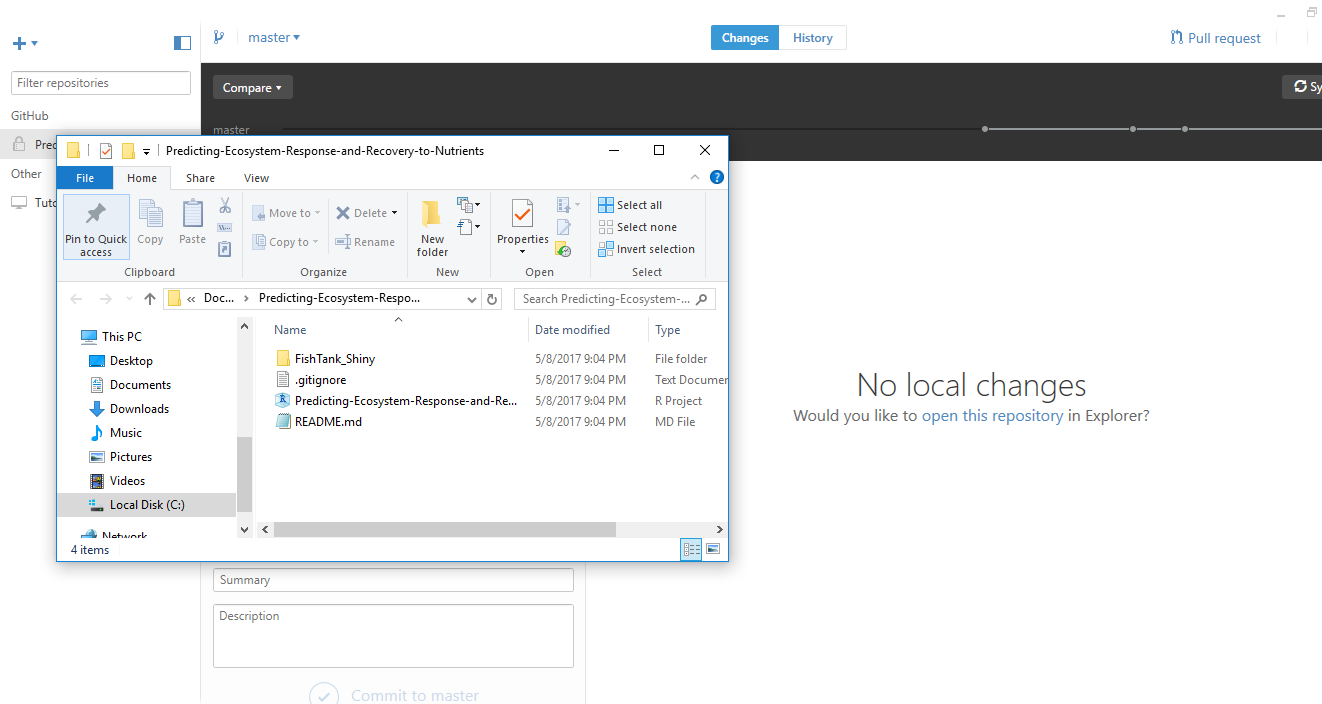
Set up looks like this:



After setup, click “US EPA” on the left, and find “Predicting…” and click on it.

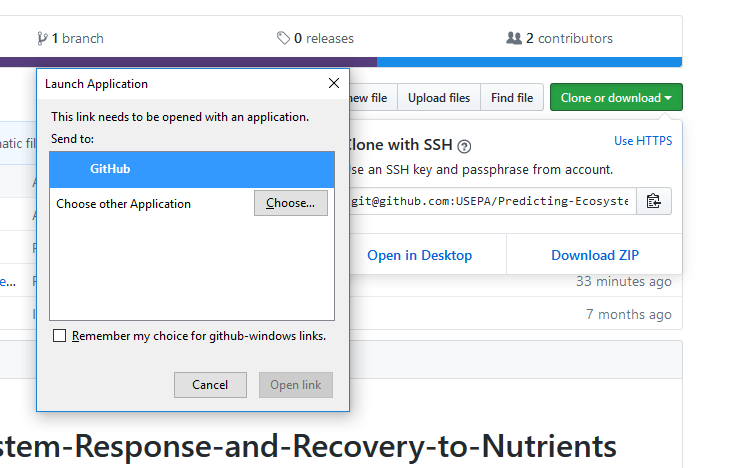


It will say “No local changes” and if you click on “open this repository” underneath that, it will take you to the location of the files downloaded to your computer.



When making this tutorial, I forget\*\* whether I had to do the following before getting the previous screen…go back to the original page: <https://github.com/USEPA/Predicting-Ecosystem-Response-and-Recovery-to-Nutrients>

Now you can click “Clone or download”, then “Open in Desktop”. If your GitHub Desktop is properly installed, it will open the following Launch Application window instead of taking you to the download page:

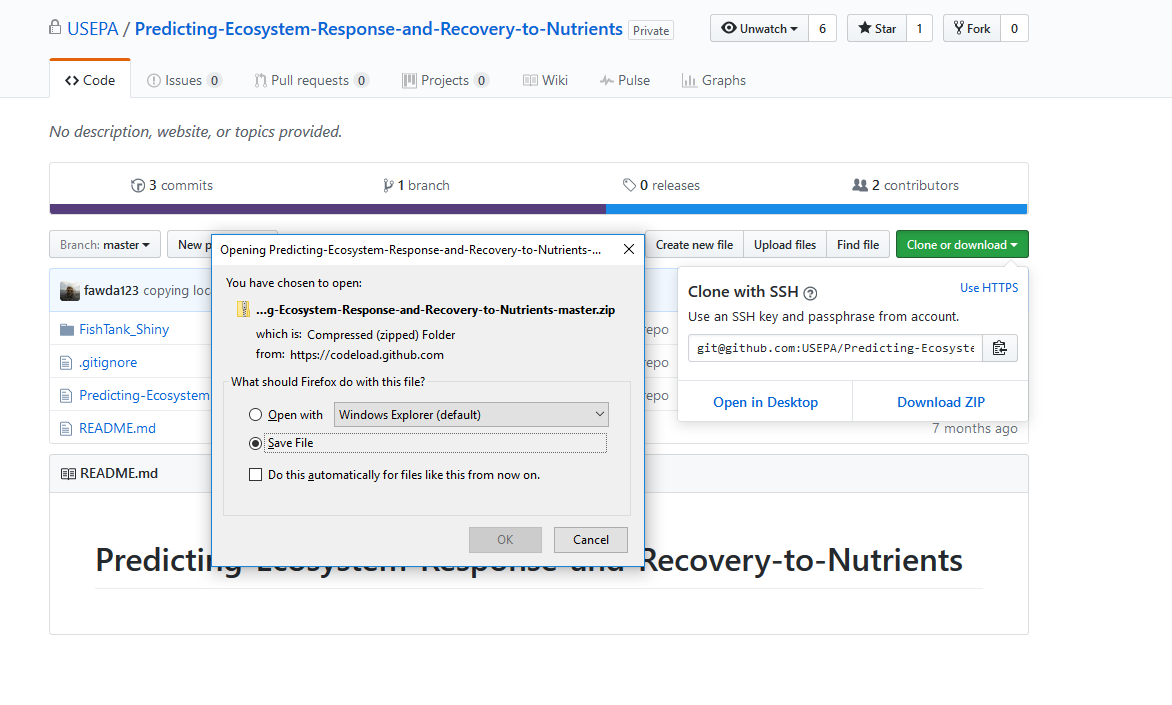


\*\*If you do this and are reading this and can tell what I forgot, please edit the tutorial with what you needed to do.

In any case, do the ‘open this repository’ and note where the **Predicting-Ecosystem-Response-and-Recovery-to-Nutrients.Rproj** file is located, you will click on that to start the tutorial.

* **Instructions to get the code via download:**

**If having problems with GitHub:** For now, just get the zip file. Click the green **Clone or download** button and **Download ZIP**.

Save to your computer. Usually it will go to your Downloads folder. Move it somewhere else (e.g. Documents), right click on the zipped file, and **Extract All**.

Preparing to Run CGEM

Open rstudio and execute the following commands to install the necessary packages:

install.packages("shiny")

install.packages("shinyjs")

install.packages("ggplot2")

install.packages("dygraphs")

install.packages("dplyr")

install.packages("tidyr")

install.packages("xts")

install.packages("ncdf4")

install.packages("htmltools")

The packages may also be installed using pull-down menus by selecting “Tools/Install Packages …” After installation has completed, close R Studio.

1. Have R Studio installed, open it, then:

Install necessary packages, click “Packages, Install”:

-shiny

-shinyjs

-ggplot2

-dygraphs

-dplyr

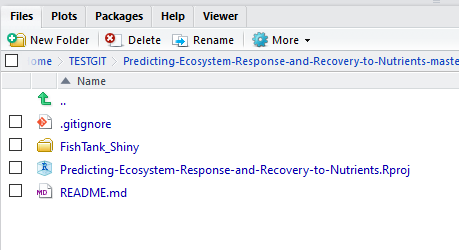
-tidyr

-xts

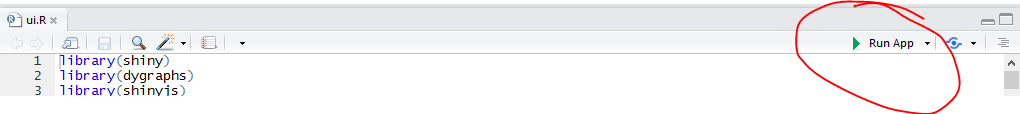
-ncdf4

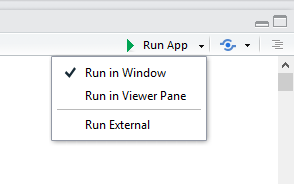
-htmltools

Close R Studio.

1. Go to where you downloaded the code from GitHub, or where you unzipped the directory from GitHub. In that directory, double click Predicting-Ecosystem-Response-and-Recovery-to-Nutrients.Rproj.  This will launch R Studio, and you should be in that directory within R. Verify by typing “getwd()” in the Console.
2. Under the **Files** tab, go to the FishTank\_Shiny subdirectory. 

* In FishTank\_Shiny, click on the file ui.R within R Studio in the files tab, and the file ui.R will appear in the left top panel. Click on **RunApp** (green arrow on the right of ui.R). This tutorial was tested with **Run in Window** selected.

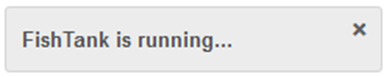




-          Clicking on **Run App** should open a window with the FishTank application.  If you have an error and need to install something else, first click on the red stop sign  to stop R Shiny, then install the package, then click **Run App** again.

-          FishTank for R Shiny will open in a browser and the first tab will be **Select parameters**.  Click on the **Model output** tab.  Now click on the **FishTank GEM** logo button in the top left hand corner. 

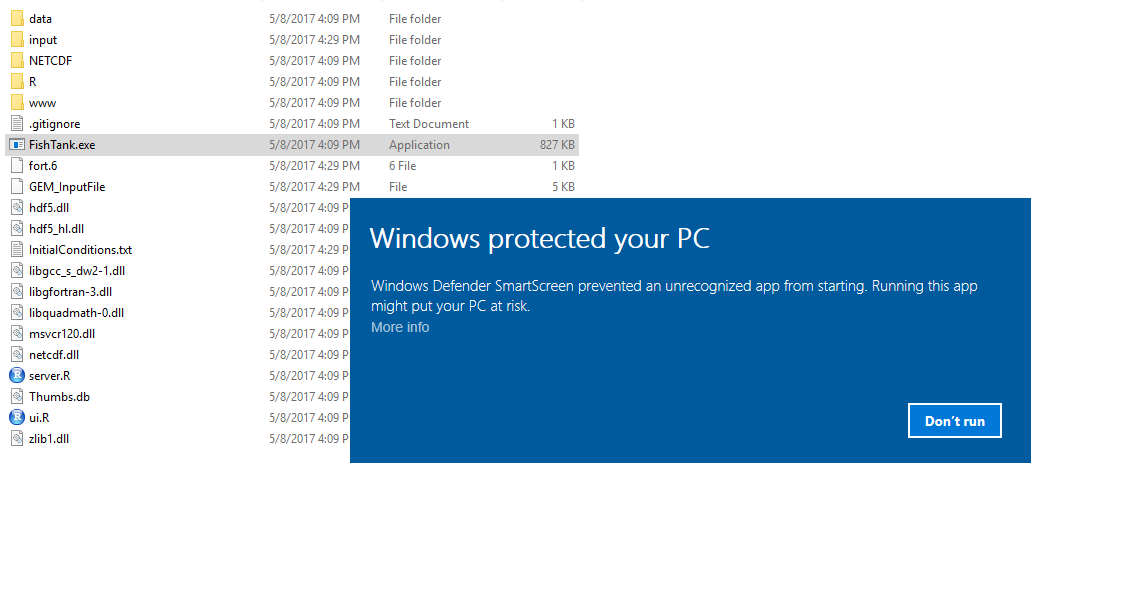
“FishTank is running…" appears in the bottom right corner



and the **Model output** tab displays plots for default O2 and PO4.

**Troubleshooting:**

If you try to run R Shiny and nothing seems to happen, check to see if you can run the executable by itself by clicking on FishTank.exe in the FishTank\_Shiny directory. You may get a warning because Windows does not recognize the executable. Click “More info”, and “Run Anyway”. You should only have to do this once (or once for each time you download a new executable). We are working to fix this problem.

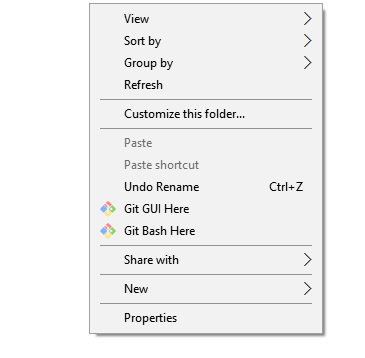


**Also note:**

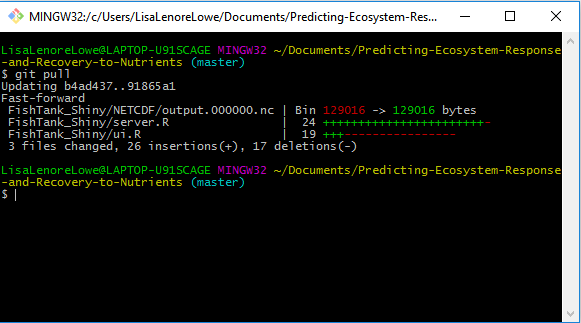
* The code generates a GEM\_InputFile from .RData files and from user input. If you add a GEM\_InputFile, or an InitialConditions.txt file, ***it will be overwritten***. If you are, or want to become, an advanced user and would like to use your own input files, email [lowe.lisa@epa.gov](mailto:lowe.lisa@epa.gov) for some instructions and R Scripts for running outside of R Shiny.
* The file ‘fort.6’ is generated in the FishTank\_Shiny directory on program execution. This will show error messages that do not get sent to R Shiny.
* The files GEM\_InputFile, InitialConditions.txt, and NETCDF/output.000000.nc are generated by the executable. Check the date stamps on these to see if the executable ran.
* The R Shiny code gets the variable names from NETCDF/output.000000.nc upon initialization. It also rewrites it while running. If you get “ERROR: Run model to get labels”, then the output file is probably missing or damaged. Copy the file labels.nc to output.000000.nc and try again.

**Updating the code through GitHub:**

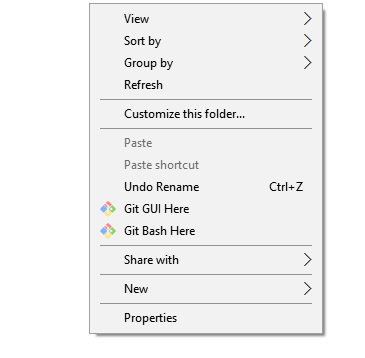
To downoad the latest changes to the repository, right click in your working directory and click **Git Bash Here**:



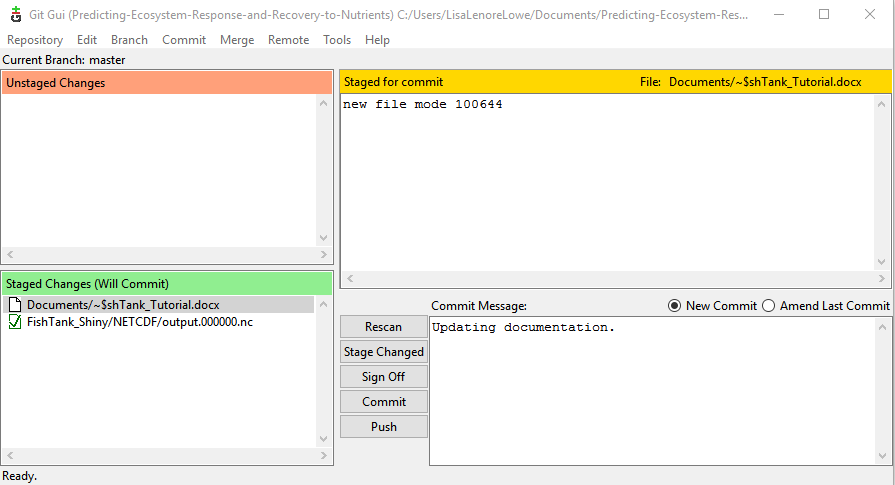
Type “git pull”. (That’s it.)



To upload your latest changes to the repository, right click in your working directory and click **Git GUI Here** (or Bash if you are comfortable with git command line):



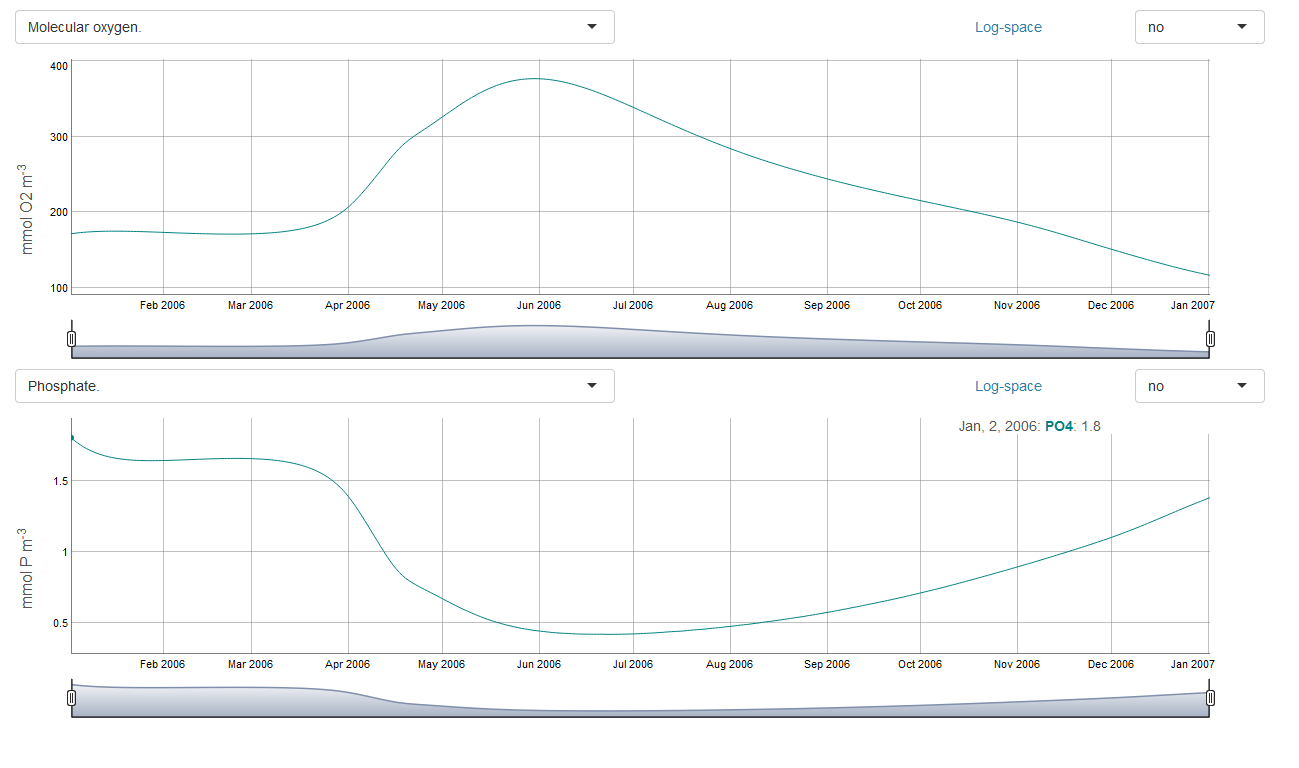
Click **Stage Changed**, type a commit message, and then click **Commit**. Then click **Push** to put your changes to GitHub.



Don’t commit and push code until you are fairly comfortable with what you’re doing. At the same time note that if you make a mistake, it can be undone.

**FishTank Tutorial Exercise 1.0**: ***An introduction with emphasis on phytoplankton.***

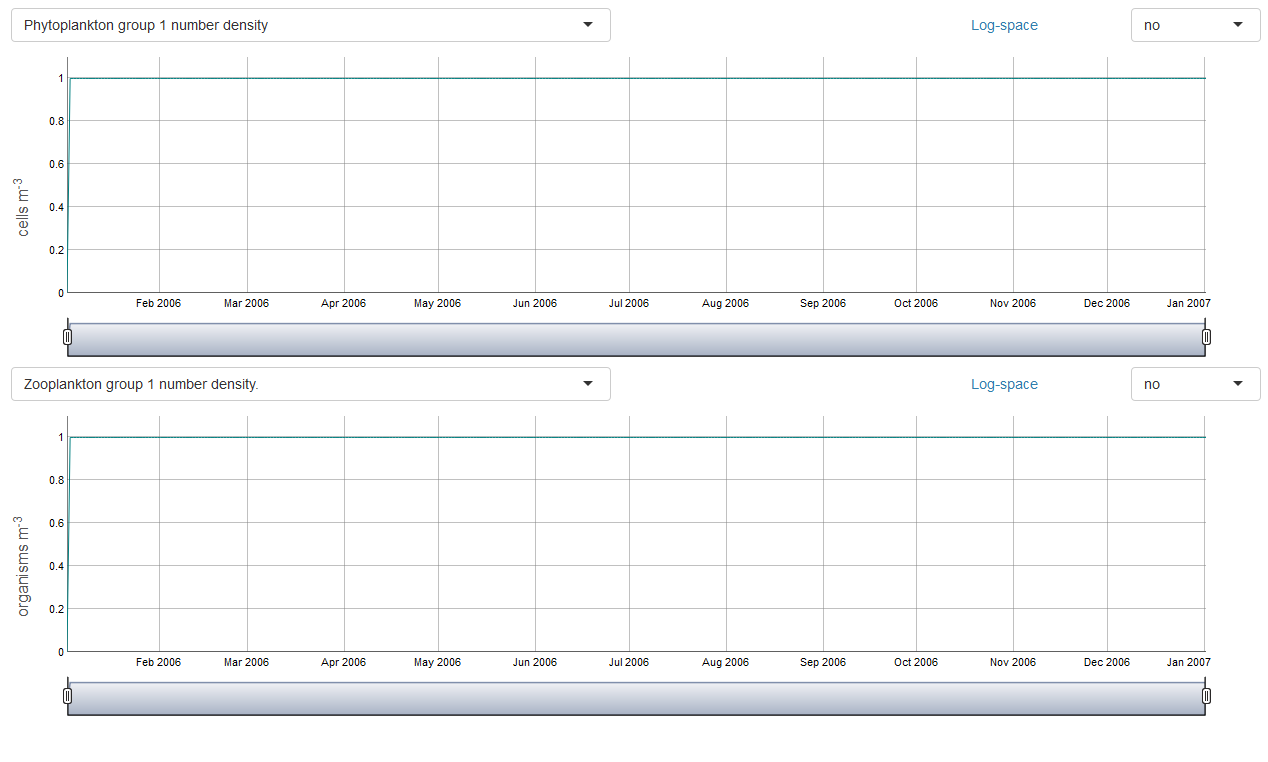
1. Start FishTank as per the general instructions.
2. Run FishTank by clicking the FishTankGEM button. A gray button will appear in the right bottom corner that says “FishTank is running…” The **Model output** tab default shows O2 and PO4:



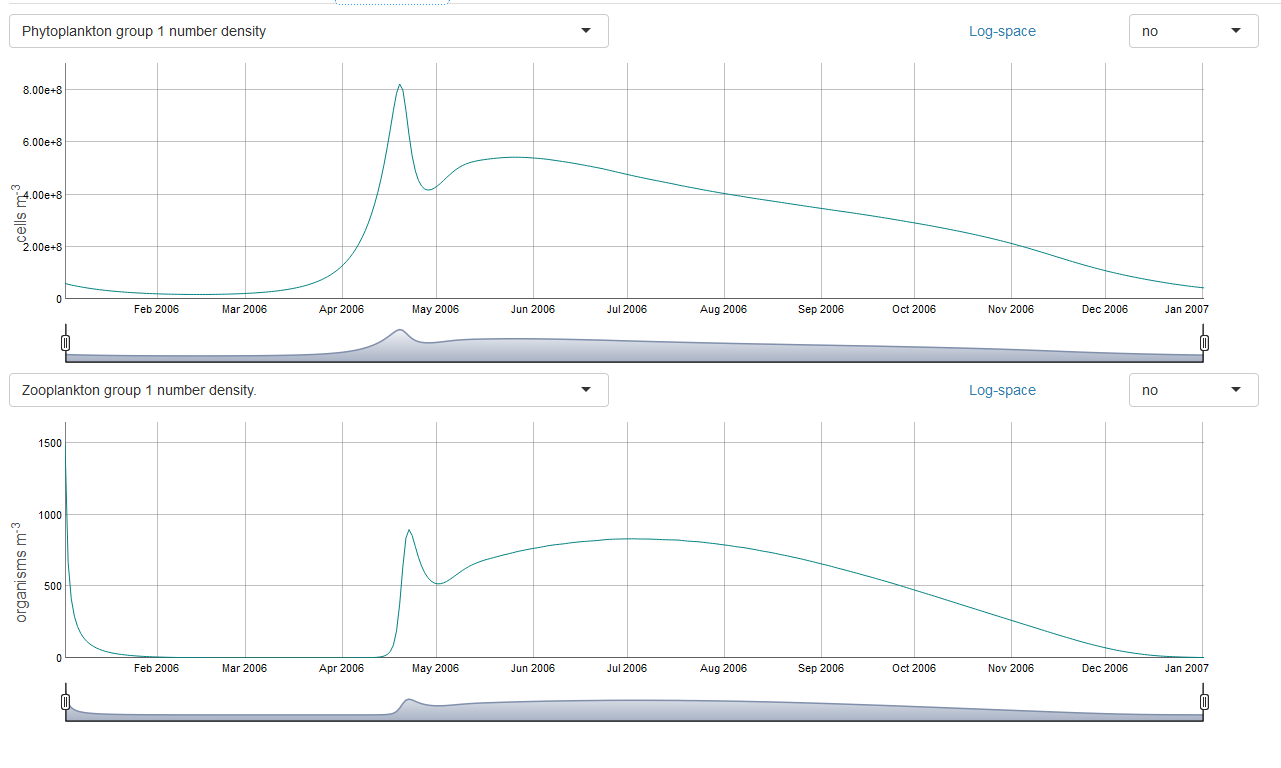
1. Change the plots to show Phytoplankton group 1 number density on the top, and Zooplankton group 1 number density on the bottom.



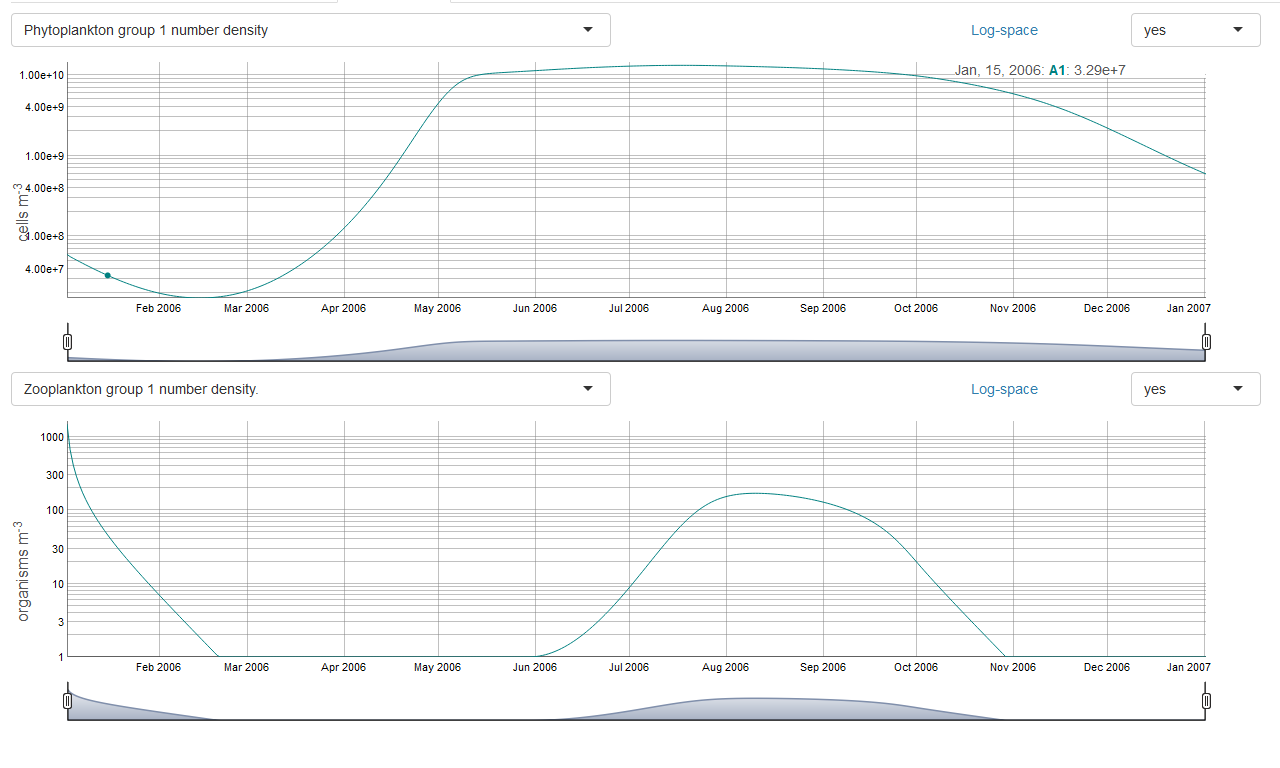
1. Click the **Select initial conditions** tab. Set all Phytoplankton abundance A1:A6 and Zooplankton group 1 and group 2 to **zero** and run again. Look at the results in **Model output**. The code automatically sets a minimum of **1** for these variables to avoid numerical problems. When looking at mass imbalance problems, keep in mind that the code will not allow negative values and it sets them to zero (or 1 for plankton), which can add mass. This usually doesn’t create problems under normal circumstances (physically realistic parameters and reasonable time step values).



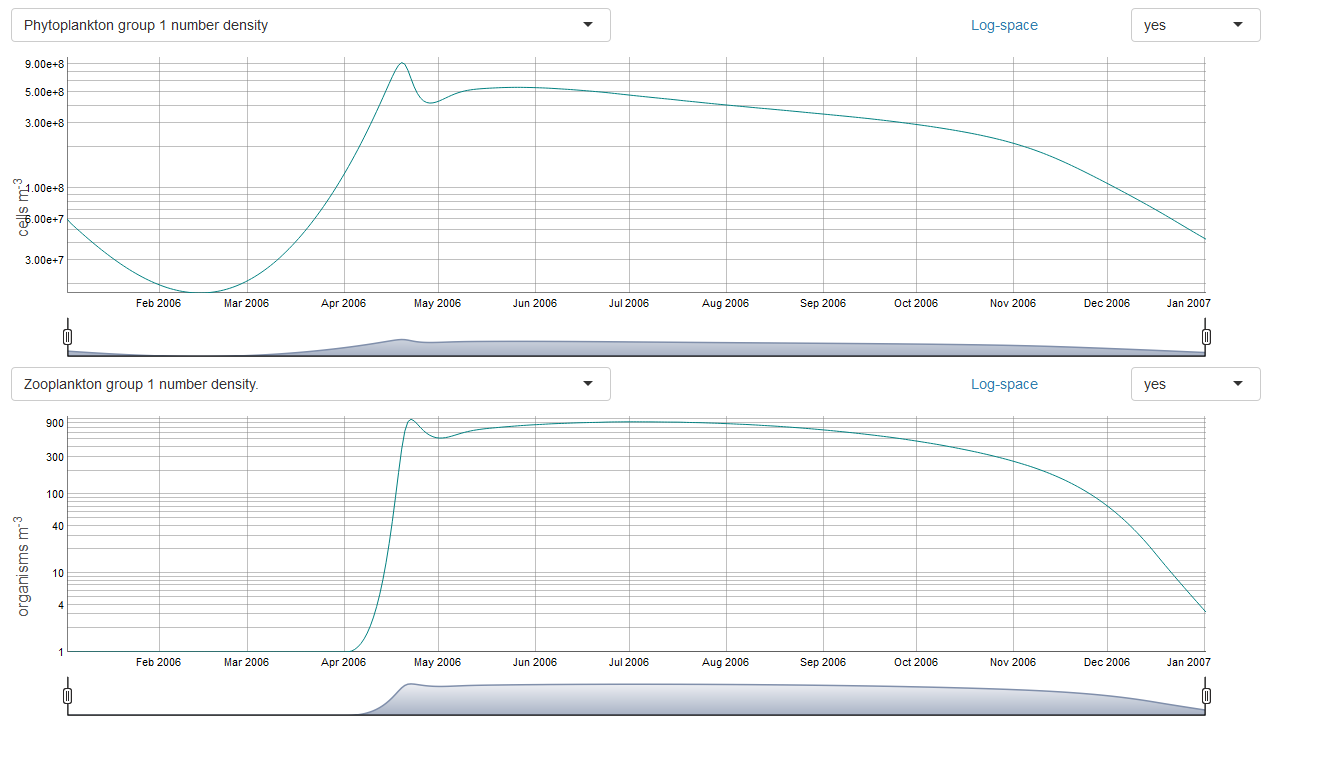
1. The **Select parameters** tab shows 6 phytoplankton and 2 zooplankton species. We can change this to run only one phytoplankton group and one zooplankton group. Click the **Reset all** button under the **FishTank GEM** button to reset the initial conditions, then toggle the switch **One Phytoplankton, One zooplankton** under **Simulation specifics and switches** to 1. Rerun and examine the results.



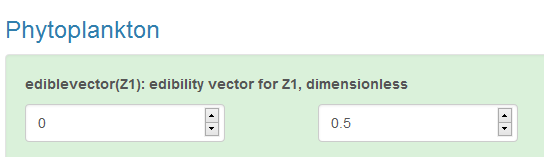
1. Go back to the **Select Parameters** tab. We will go over some of the phytoplankton parameters. Note that while there are 6 places for each phytoplankton parameter, we have selected **One phytoplankton** and FishTank will ignore the last 5 parameters. FishTank will only look at the first of six values for **Phytoplankton**, the first of two values for **Zooplankton**, and the first and seventh value in the **Temperature** functions (phyto+zoo=8). To look at the behavior of the phytoplankton without the effects of zooplankton grazing, we can turn off the zooplankton’s ability to eat phytoplankton by making *Athresh* a very high number. Zooplankton can only graze if the number of phytoplankton is greater than *Athresh*. Set *Athresh* to 1e10. (Only do this for testing, it isn’t physically realistic.) Run, then look at **Model output**. Turn Log-space on for both plots.

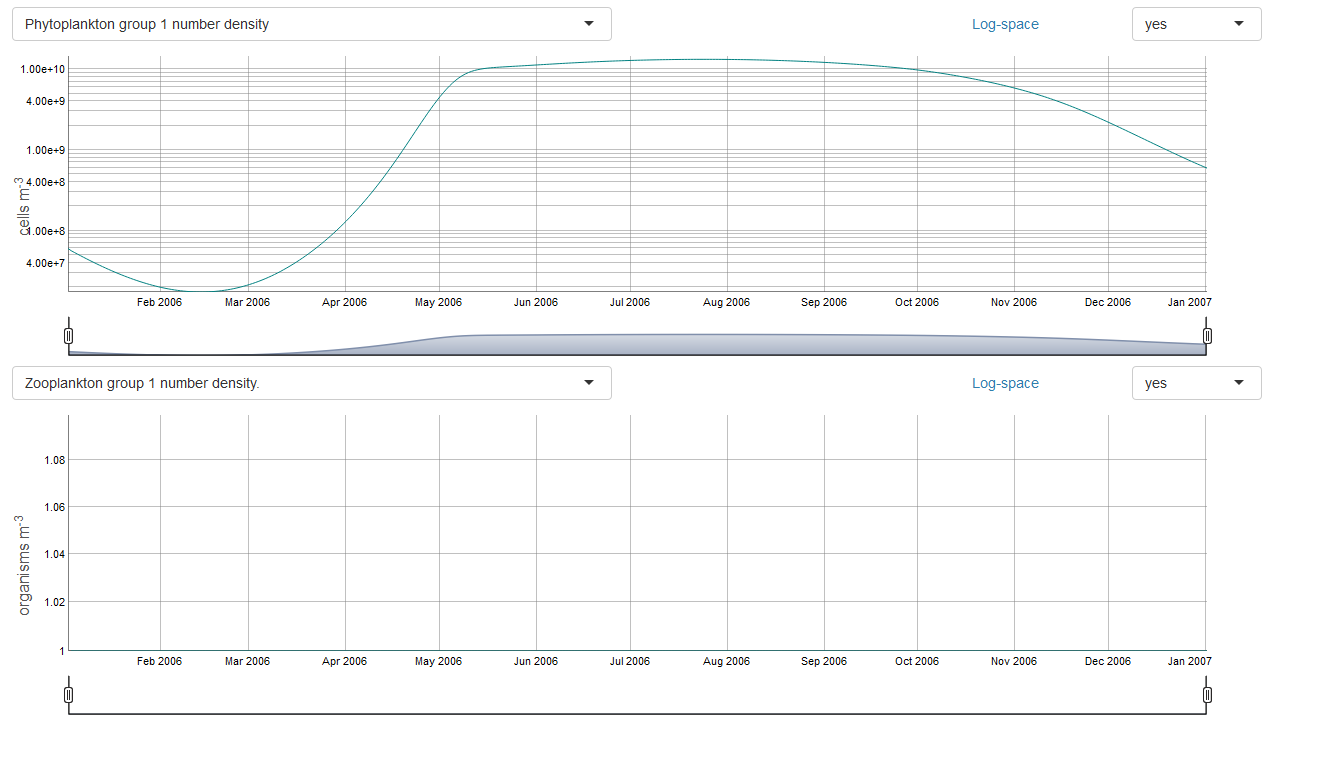


Setting *Athresh* to 1e10 worked until phytoplankton grew higher than 1e10 cells/m3. Set *Athresh* back to 7e7, but change the zooplankton **Initial conditions** to zero. Again, since we don’t let zooplankton go to identically zero, they ramp up when phytoplankton is greater than *Athresh*:

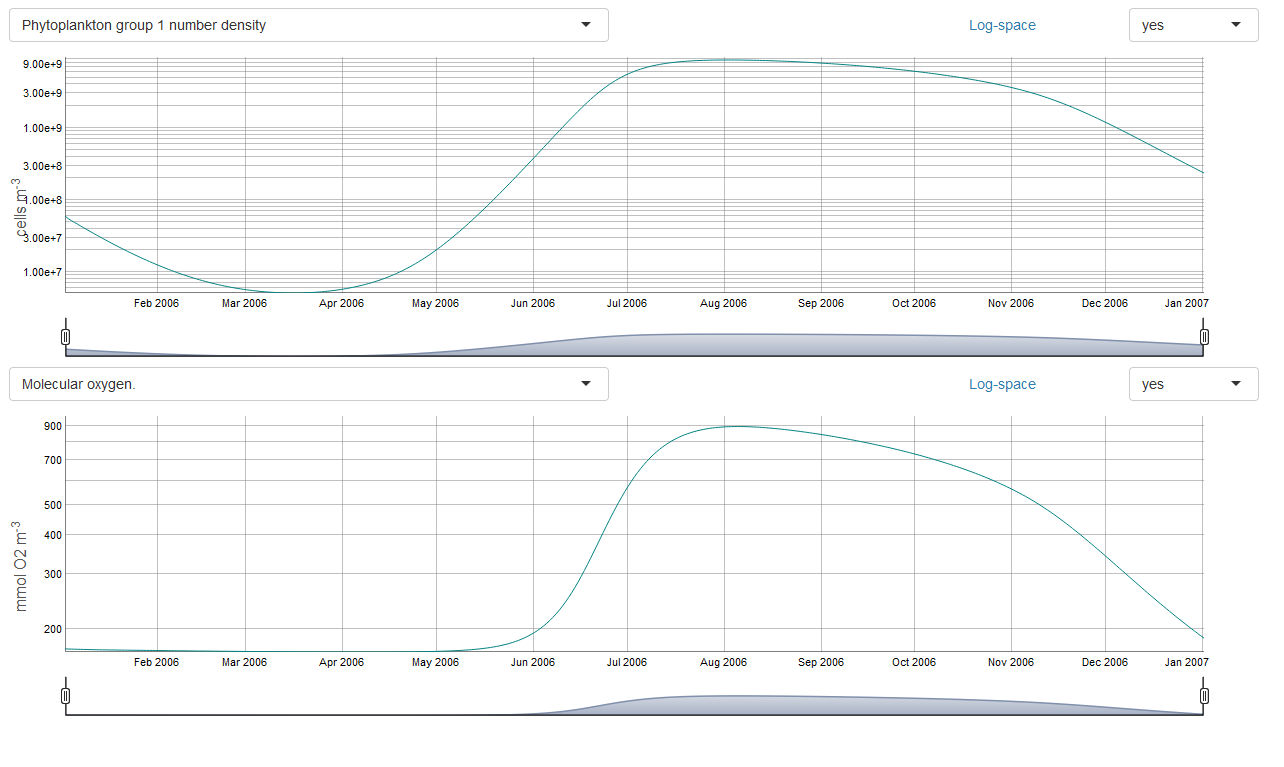


We will use another technique to get rid of those pesky critters…*edible\_vector* is the percent of the phytoplankton that is ‘edible’ to the zooplankton. If we set it to zero, the zooplankton cannot grow from eating phytoplankton.

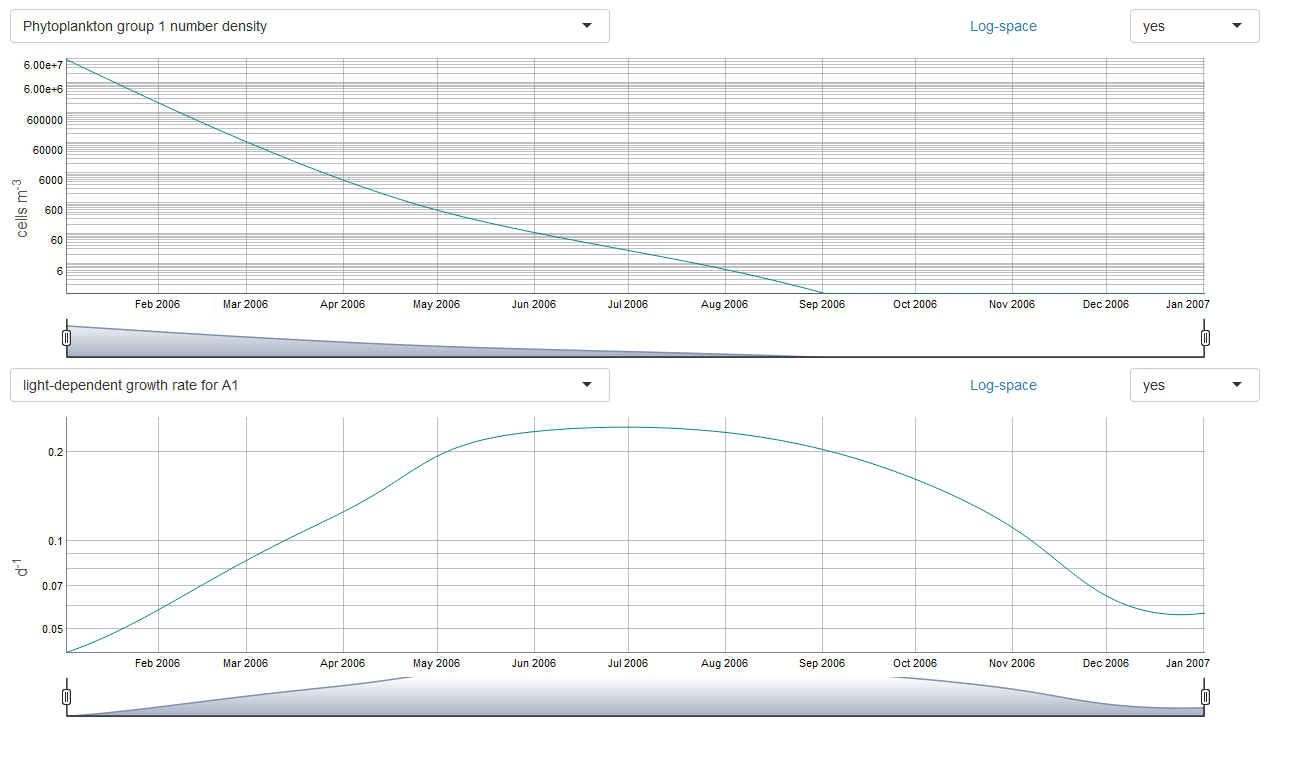




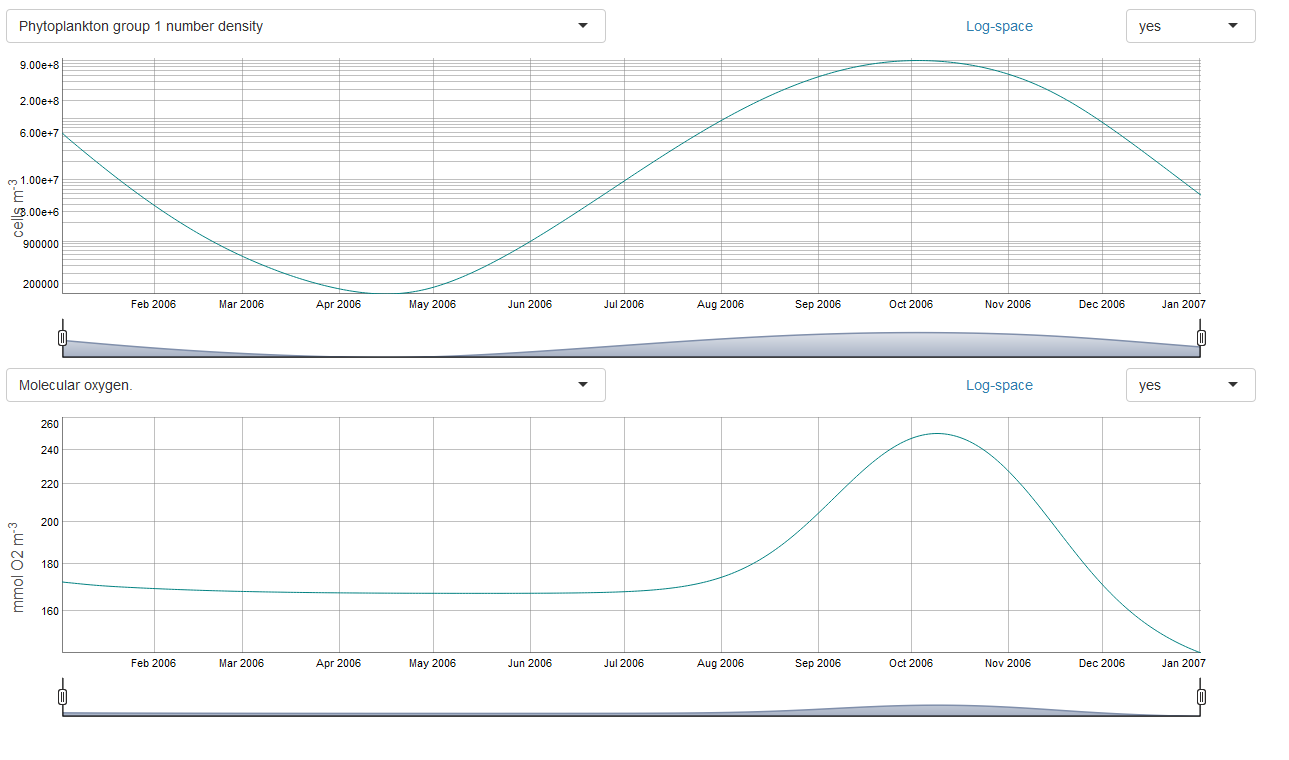
1. Now let’s change some other phytoplankton parameters. Select **Molecular oxygen** in the second plot so we can look at something more interesting than no zooplankton. O2 is ridiculously high from too much phytoplankton and no surface oxygen flux model- we can worry about that later. The parameter *umax* is the maximum growth rate for phytoplankton. Let’s cut it in half, set *umax*=.56 on the **Select parameters** tab under **Phytoplankton**, click run, and look at **Model output**. Note that we did stunt the growth a bit, the growth is not linear with *umax* (did not decrease by half).



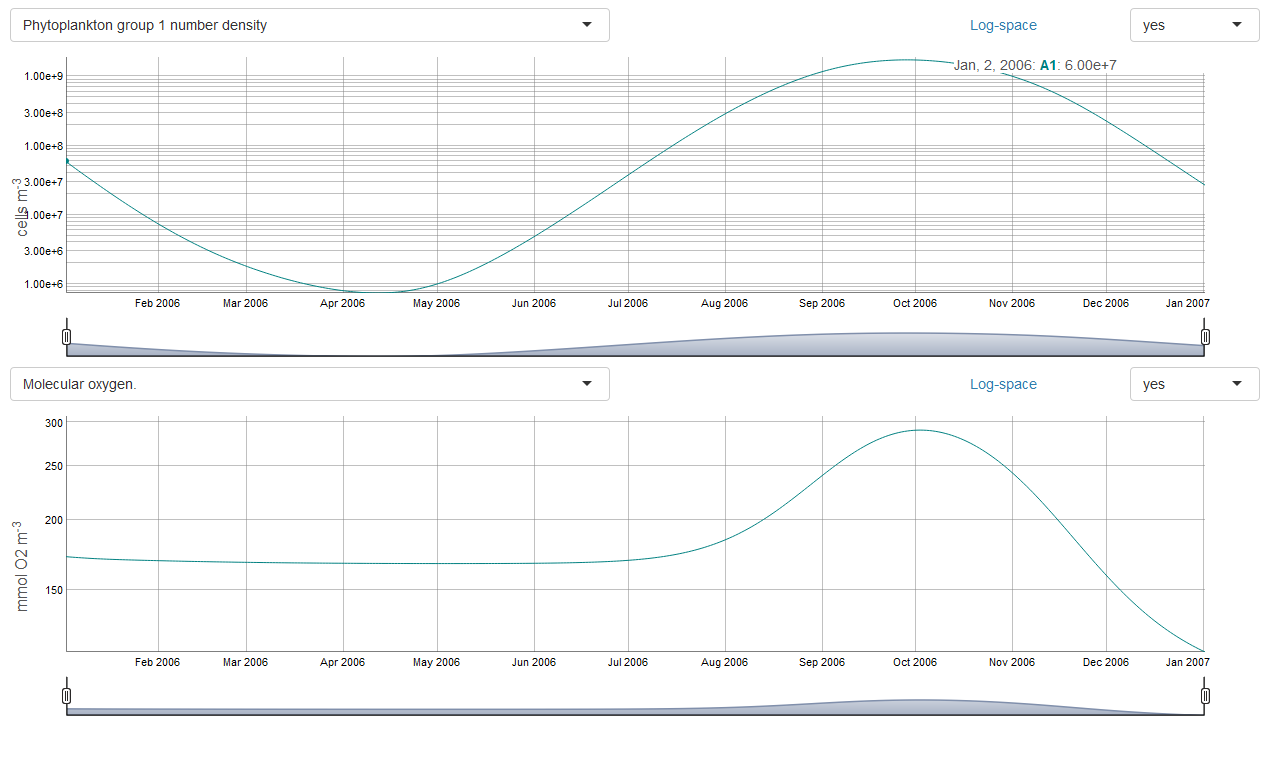
1. The next parameter is *alpha*, a factor that multiplies the irradiance at the cell center (in an exponentially decaying function). Add another zero before the three (multiply *alpha* by .1). This will make the phytoplankton almost always light limited, not nutrient limited, and inhibits growth. In the plot below, phytoplankton and *uE*, the light-dependent growth rate for phytoplankton, are shown.

J. Lehrter set this parameter according to a paper [add citation], and I don’t recommend changing it unless you know why it might be changed…so let’s change it back now (get rid of a zero in *alpha*)!

1. Let’s skip down to mortality. The mortality (parameter *mA*) is the percent of phytoplankton that die off in a timestep. This is currently 11%. Double this (set *mA* = .22) and look at the results. So- maybe that was overkill. Try *mA* = 0.15 to get the following:



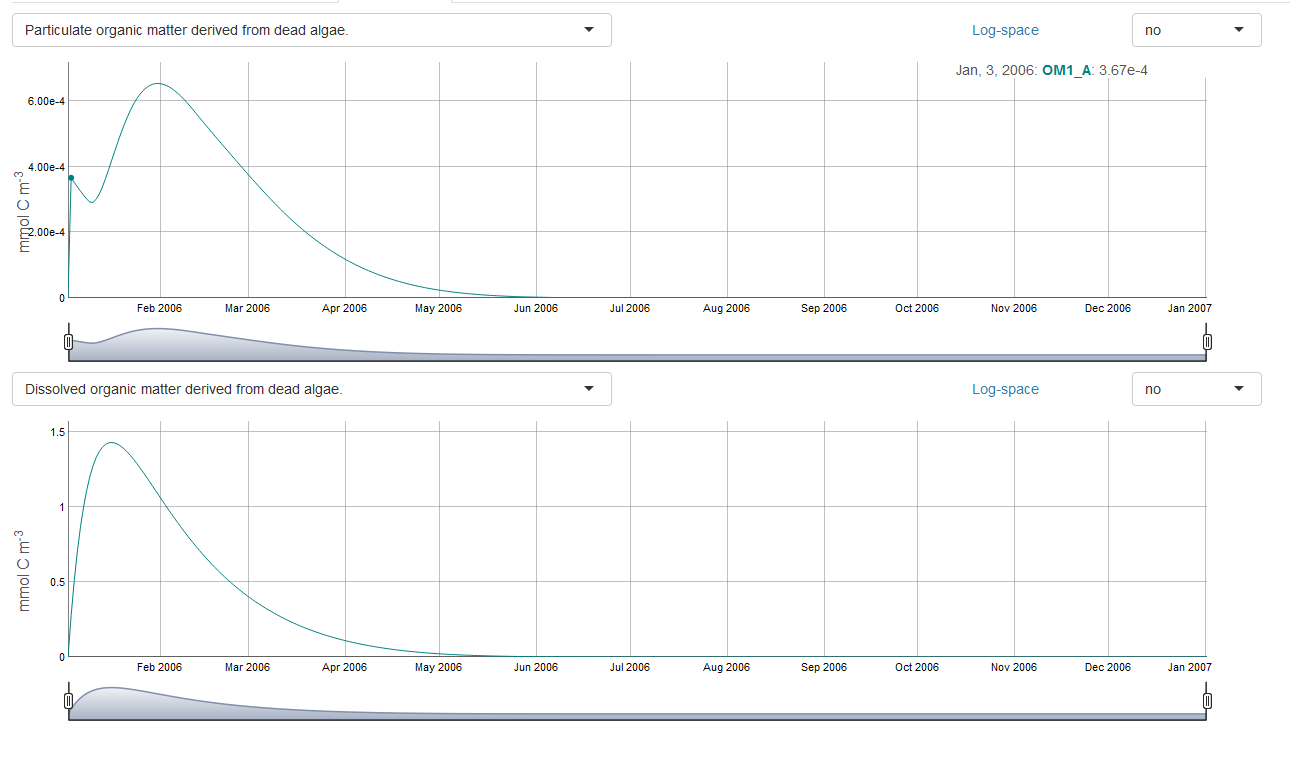
1. Phytoplankton are governed by adding growth, less respiration, less mortality (and usually less grazing, but we turned that off). ‘respg’ is dependent on the growth rate, and ‘respb’ is directly related to the phytoplankton number. Double each one (one at a time) and look at the results. Now everything is pretty low. Put the mortality back to the original (mA=.11) to counteract what you added to respiration. Here is the plot after those three changes:



1. One last test example for phytoplankton. Hit **Reset all**, then toggle on **One Phytoplankton, One zooplankton**, and Run. In **Model output**, choose Phytoplankton in the first plot and Phosphate in the second plot. Turn off Log-space. Now set the initial conditions of PO4 to 1e-8. I want to test a theory that the phytoplankton should die off because they don’t have enough PO4. The phytoplankton do die off, but the test is confusing because PO4 increases. Why does PO4 increase?



When phytoplankton die, they become organic matter, so look at the output **Particulate organic matter derived from dead algae** and **Dissolved organic matter derived from dead algae**. It increases until phytoplankton go to zero. It has a dip because it is both increasing and being decomposed at the same time. When it decomposes, it puts PO4 back into the water column through remineralization, and that is why PO4 increases until the OM has completely decomposed. Mass balance tests will show that the amount of PO4 contained in the initial phytoplankton will be the amount of OM released to the water column. (OM derived from phytoplankton is partitioned into ‘particulate’ and ‘dissolved’ as a function of the Nutrient Quotients (Qn, Qp, Qc) [see Lehrter, et. Al. in preparation].)

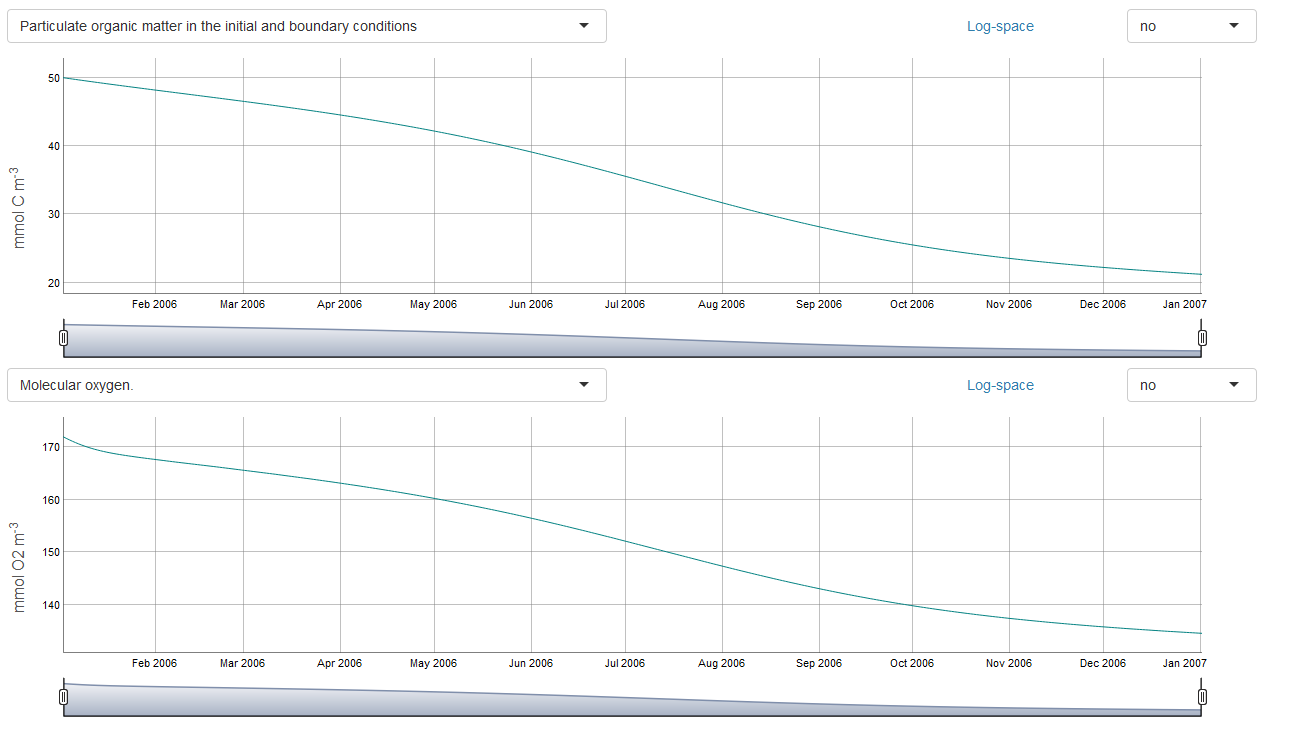


1. One last test example for part one of the tutorial. Reset all and toggle on **One phytoplankton, One zooplankton**, and we will examine how organic matter breaks down, regardless of plankton. If you do this by setting the initial A and Z to zero, you will find a bug! Go ahead and set A to zero in the initial conditions, and look at plots of Nitrate, Ammonia, and Phosphate. I already warned you that it might mess up mass balance, and this is very bad. Don’t do that again (and I’ll look into it later. Short term, or perhaps permanent, fix will be to set initial A to a minimum of 1 during initial conditions error checking).

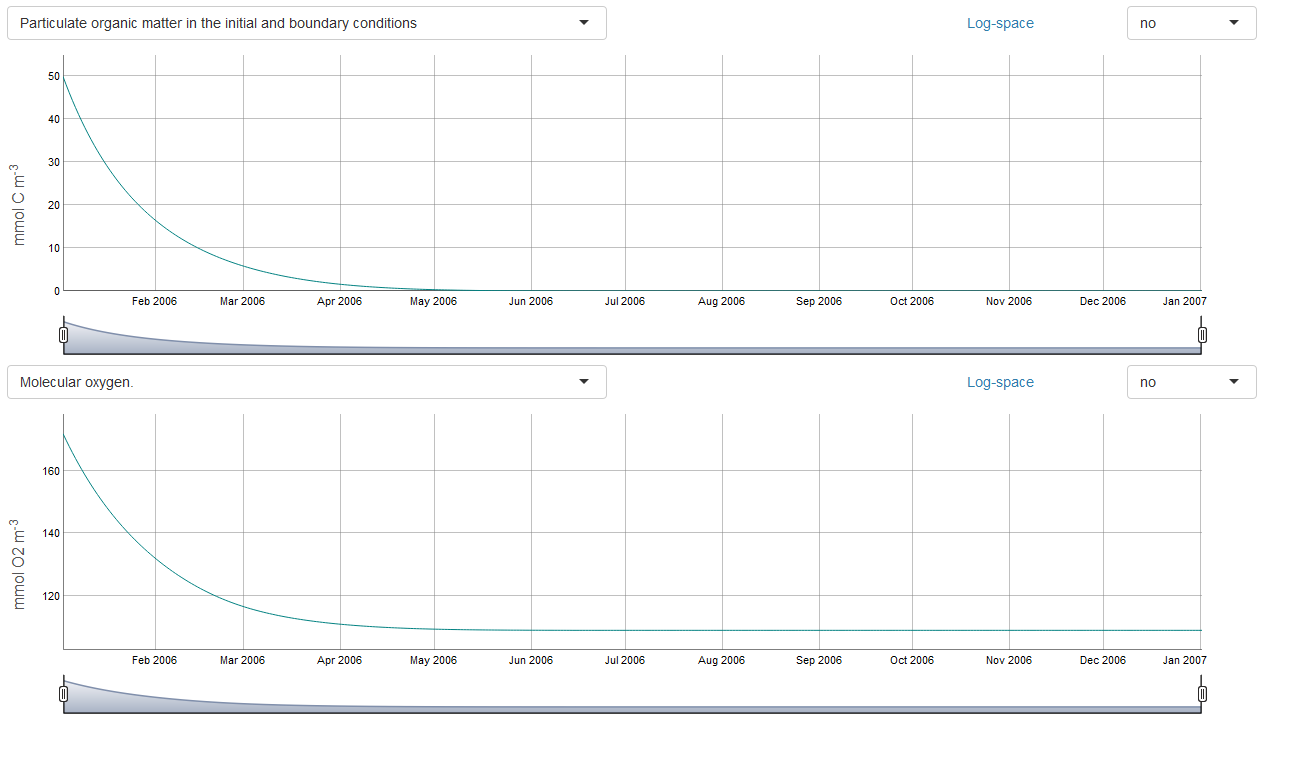
With the initial conditions reset to the original values, turn off phytoplankton and zooplankton activity by setting *edible\_vector*=0, phytoplankton growth and decay parameters to zero (*umax*, *respg*, *respb*, *mA*) and zooplankton growth and decay parameters to zero (*Zrespg*, *Zrespb*, *Zm*), and confirm you that have a steady state for plankton.



Add a big chunk of particulate organic matter by setting OM1\_bc=50 in the initial conditions.



Make it decay faster by changing *KG1\_BC*=30 in the **Organic Matter** section of the **Select parameters** tab.



The organic matter parameters (nitrification, half saturation, etc.) can be calibrated or explored by keeping the plankton terms constant as shown and changing the OM parameter values and the initial values of O2, NO3, NH4, and PO4.