OMSens Example: Lotka-Volterra

Introduction

OMSens is an OpenModelica addon that offers 3 flavors for parameter sensitivity analysis. This document presents a detailed worked example that uses the 3 OMSens tools in an integrated way.

Recalling the goals of each tool:

- Individual Sensitivity Analysis
  - Used to analyse how a parameter affects a variable when perturbed on its own
- Multiparameter Sweep
  - Exploratory experimentation that sweeps the space of a set of parameters
- Vectorial Sensitivity Analysis
  - Used to find the combination of parameters that maximizes/minimizes a state variable

All these tools focus in a specific simulation time specified by the user, namely the final simulation time.

Model

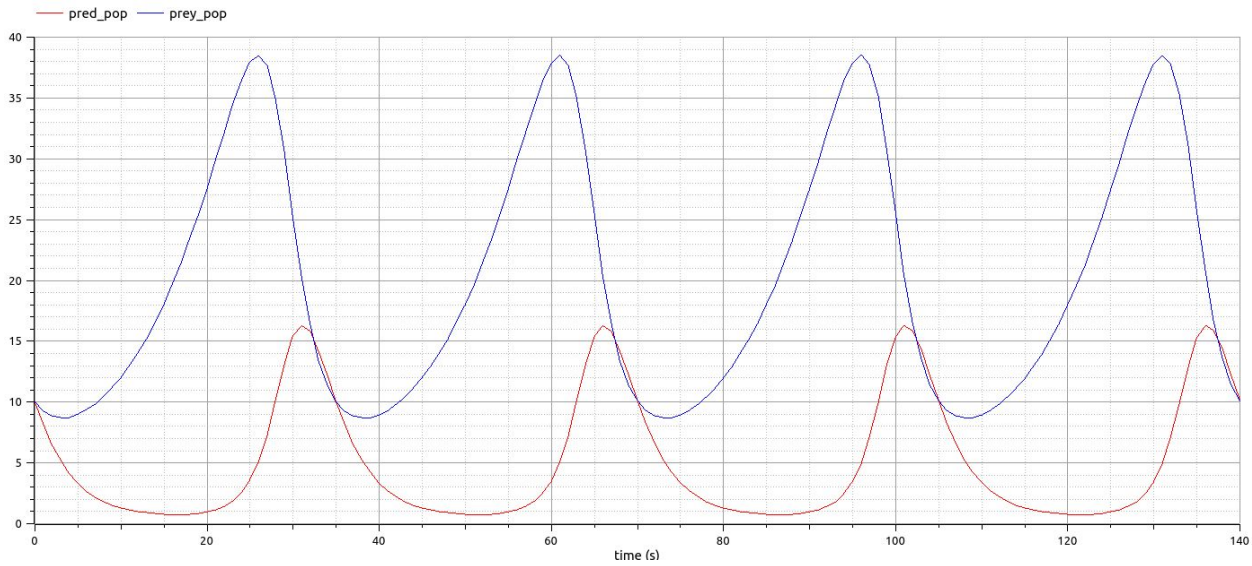
We choose the Lotka-Volterra model that consists of a second-order nonlinear set of ordinary differential equations. The system models the relationship between the populations of predators and preys in a closed ecosystem. More information about this model and a Modelica implementation can be found in the book Principles of Object-Oriented Modeling and Simulation with Modelica 3.3: A Cyber-Physical Approach, section 15.4.1.1.

Implementation

```
model LotkaVolterra "This is the typical equation-oriented model"
  parameter Real alpha=0.1 "Reproduction rate of prey";
  parameter Real beta=0.02 "Mortality rate of predator per
prey";
  parameter Real gamma=0.4 "Mortality rate of predator";
  parameter Real delta=0.02 "Reproduction rate of predator per
prey";
```

```
  parameter Real prey_pop_init=10 "Initial prey population";
  parameter Real pred_pop_init=10 "Initial predator
population";
  Real prey_pop(start=prey_pop_init) "Prey population";
  Real pred_pop(start=pred_pop_init) "Predator population";
initial equation
  prey_pop = prey_pop_init;
  pred_pop = pred_pop_init;
equation
  der(prey_pop) = prey_pop*(alpha-beta*pred_pop);
  der(pred_pop) = pred_pop*(delta*prey_pop-gamma);
end LotkaVolterra;
```

Expected simulation results

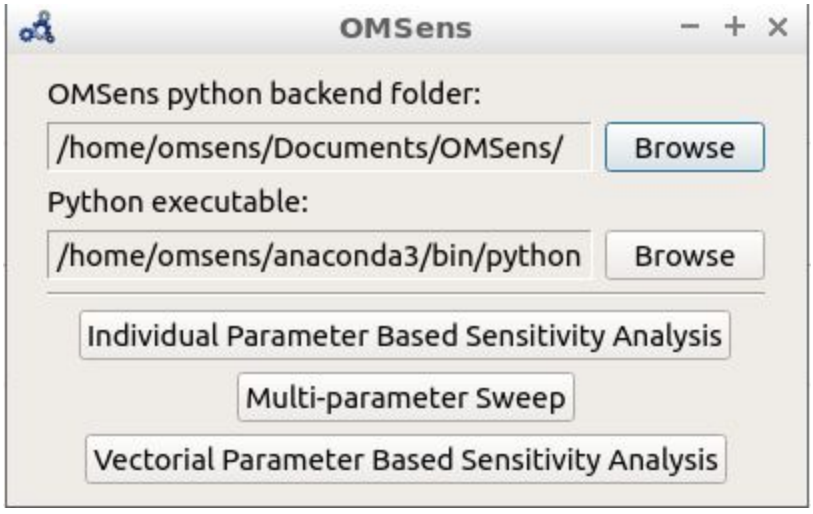


OMSens use example

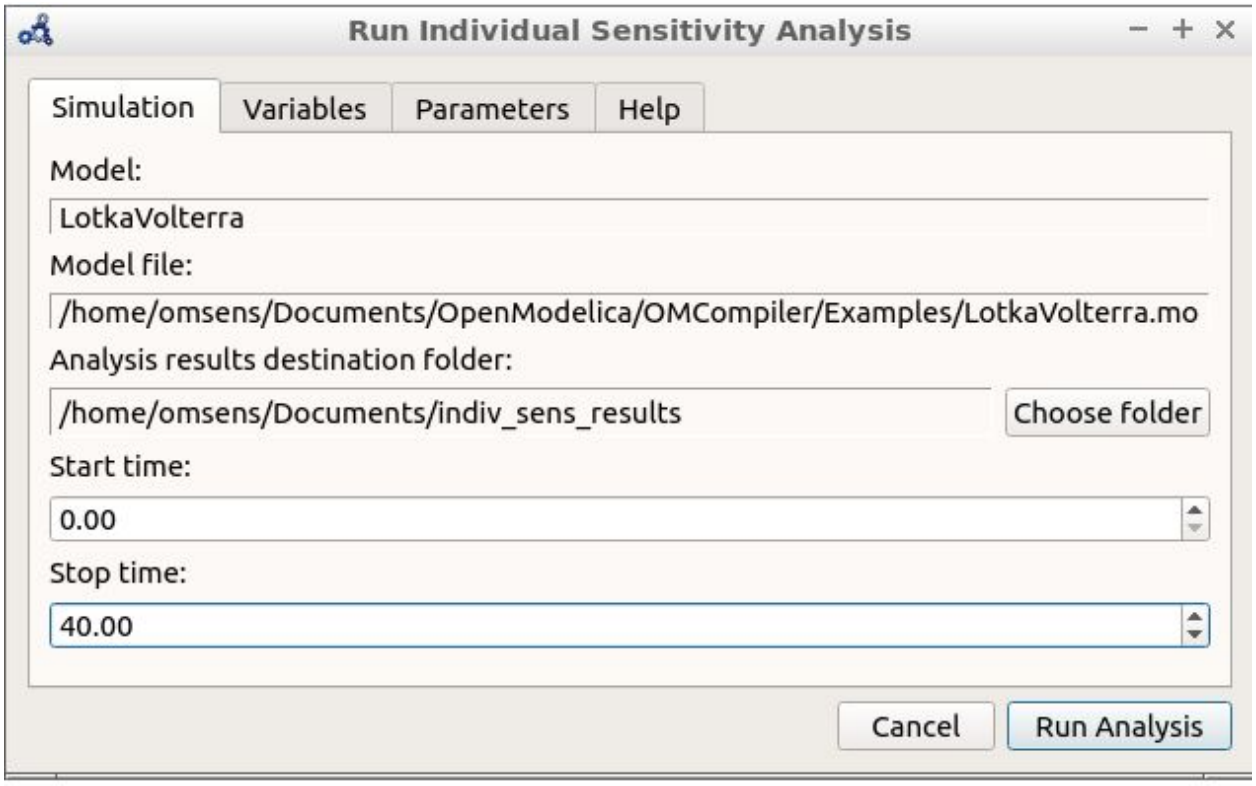
Let’s say we need to investigate the influence of model parameters on the predator population at 40 units of time. We assume a +/-5% uncertainty on model parameters.

We can use OMSens to study the sensitivity model to each parameter, one at a time.

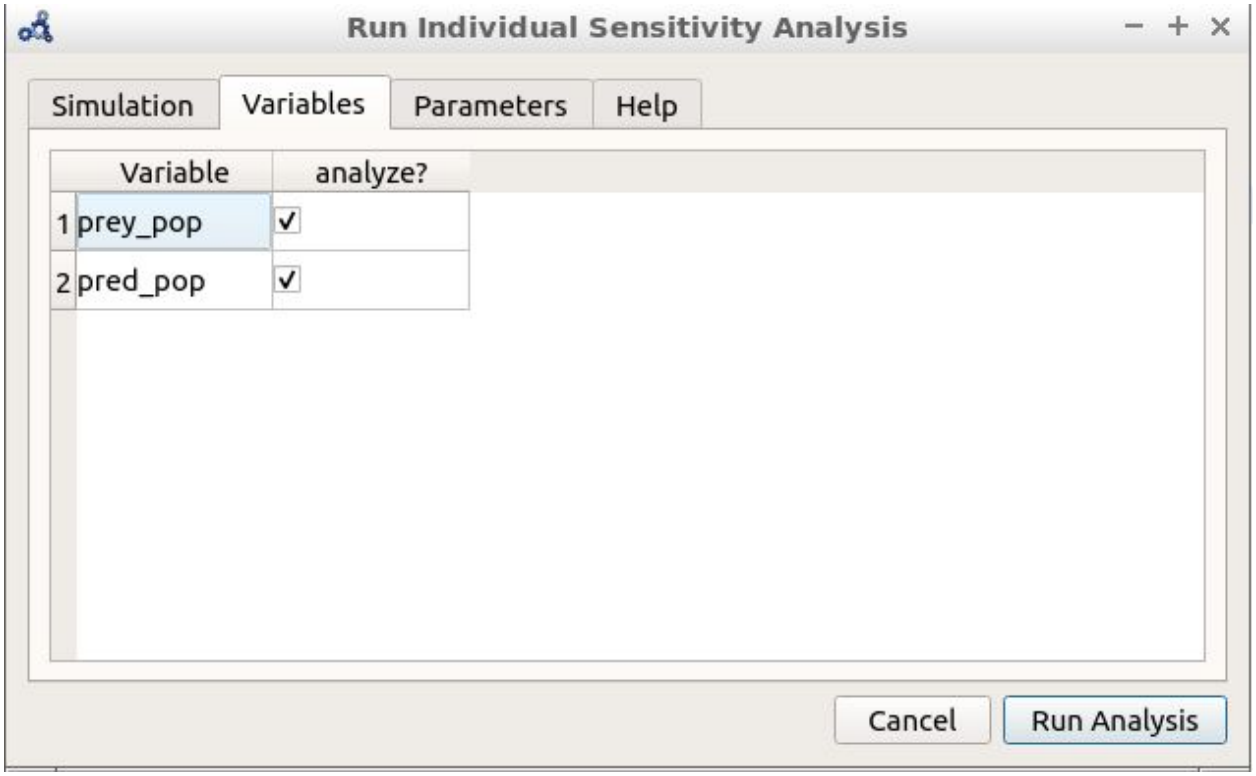
1. Open the Lotka-Volterra model using OMEdit and start OMSens. A window like the one below should appear.



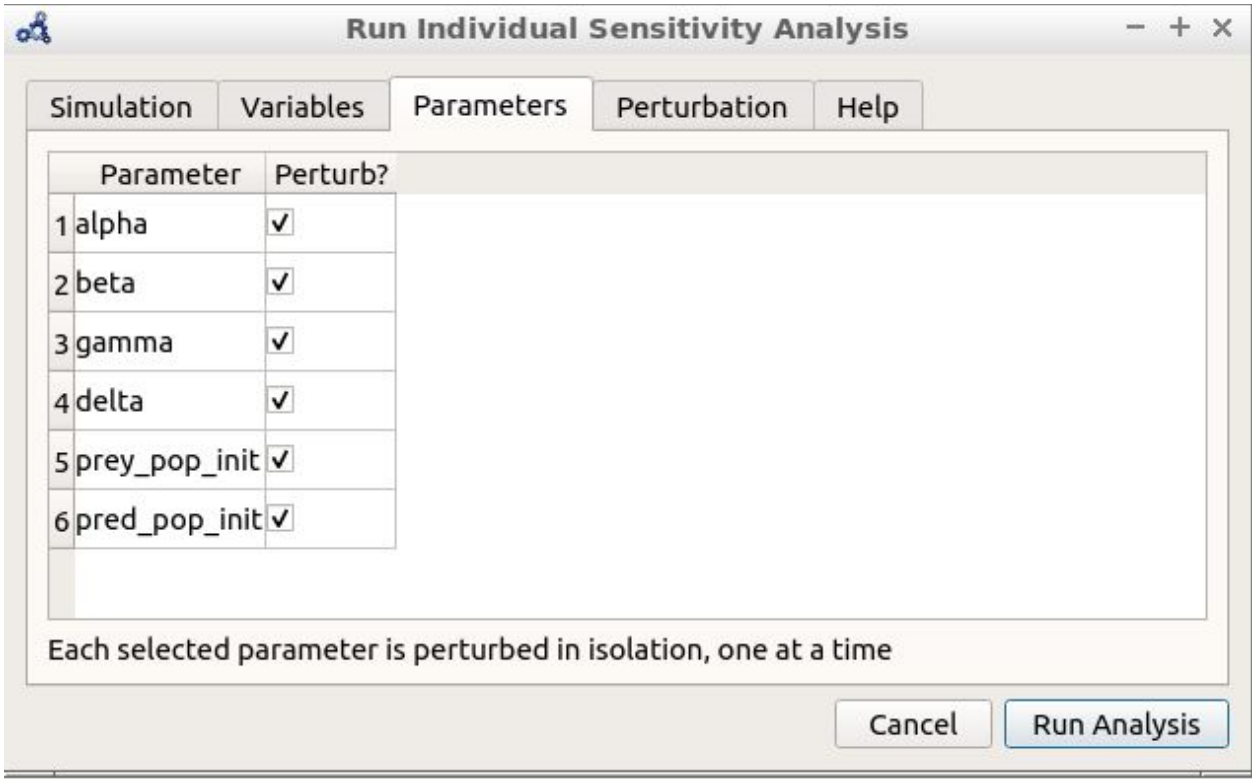
2. Choose “Individual Parameter Based Sensitivity Analysis” and set up the simulation settings.



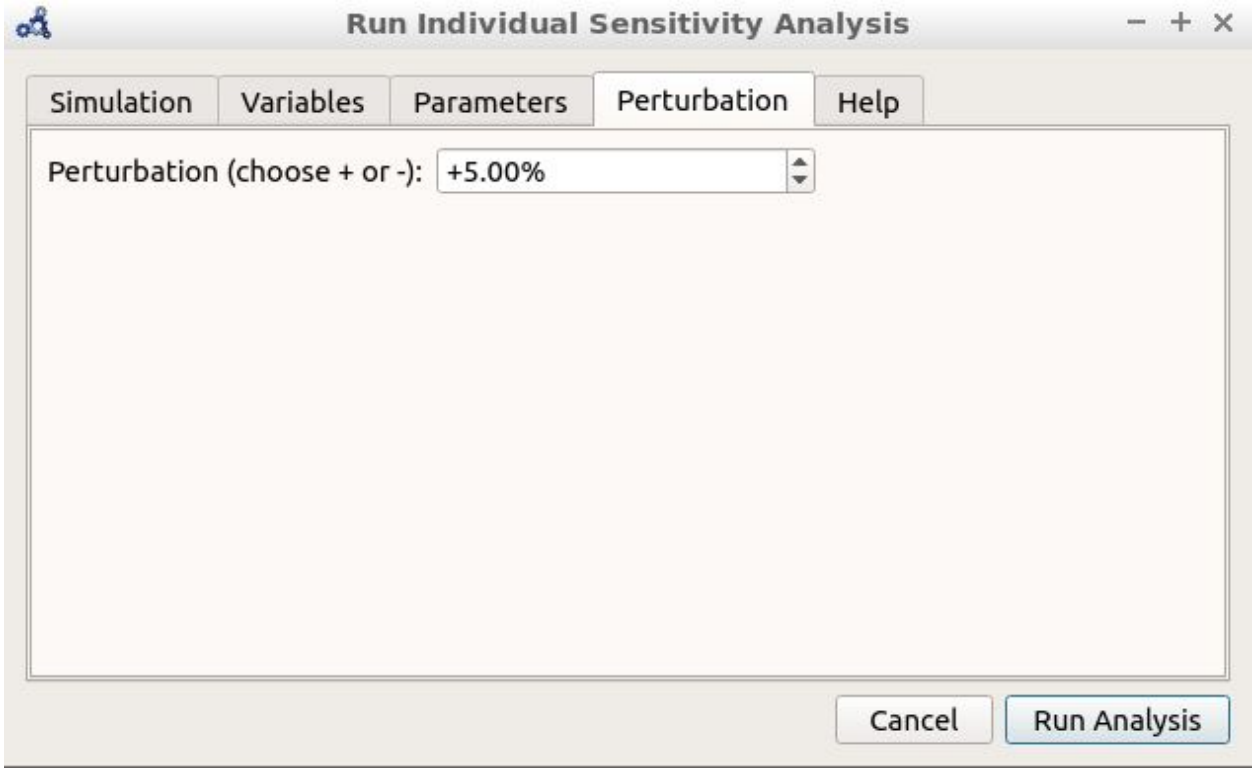
3. Select both variables



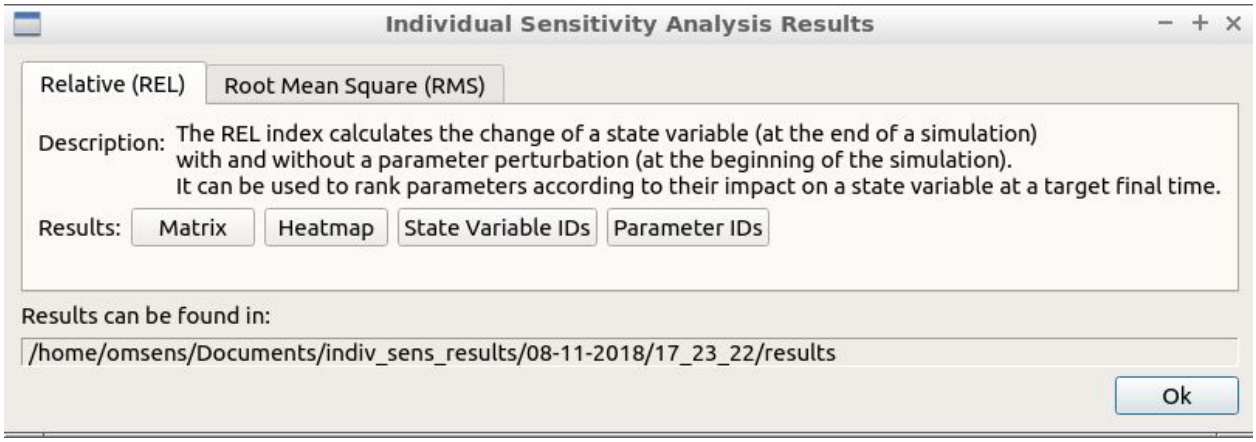
4. Select all of the parameters



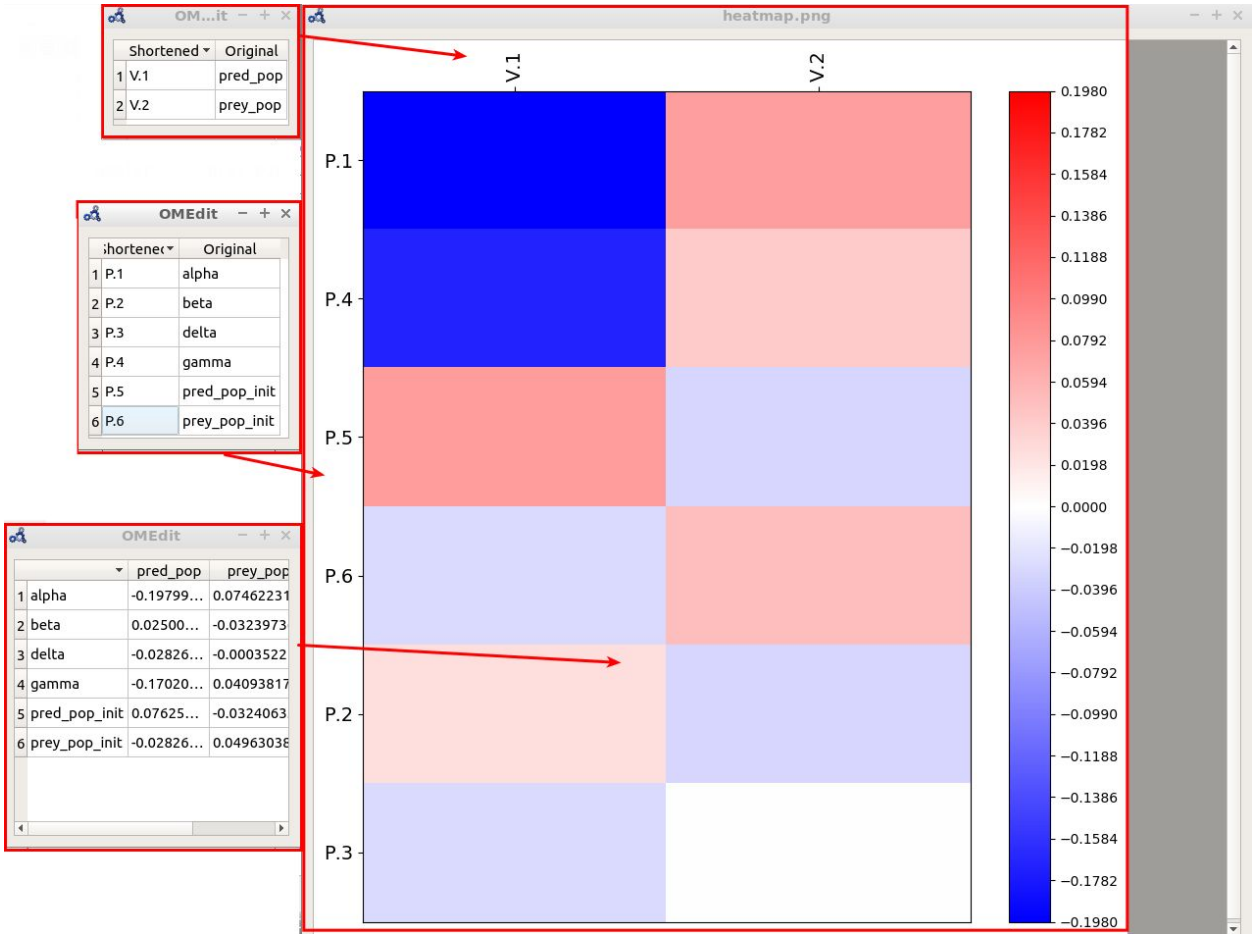
6. Choose the perturbation percentage and direction. Run the simulation



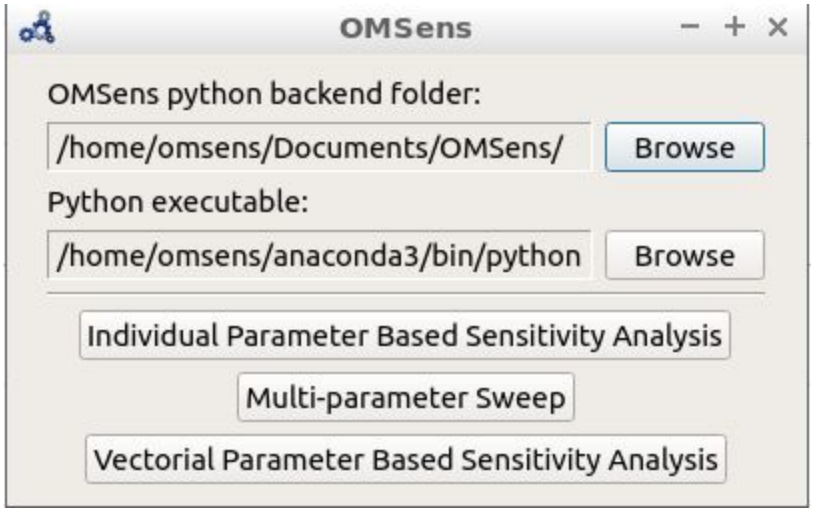
5. The OMSens backend is invoked (python code) and after the analysis a dialog with results is shown. Open the heatmap corresponding to the Relative sensitivity index.



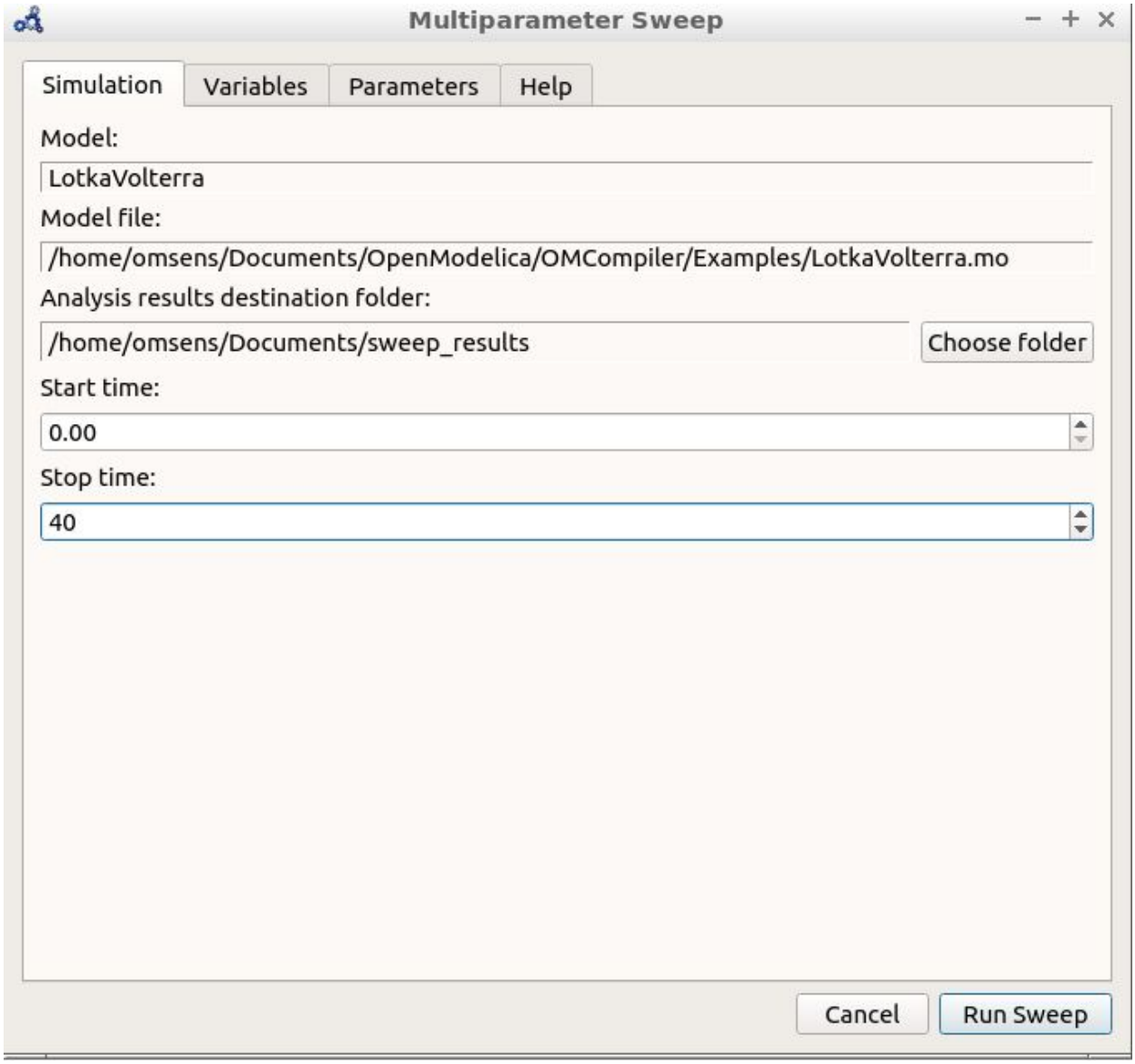
6. This heatmap shows the effect of each parameter on each variable in the form of (parameter,variable) cells. As we can see, *pred_pop* was affected by the perturbation on every parameter but *prey_pop* presents a negligible sensitivity to *delta* (P.3). Recall that this heatmap shows the effect on the variables at time 40 for each perturbation imposed at time 0.



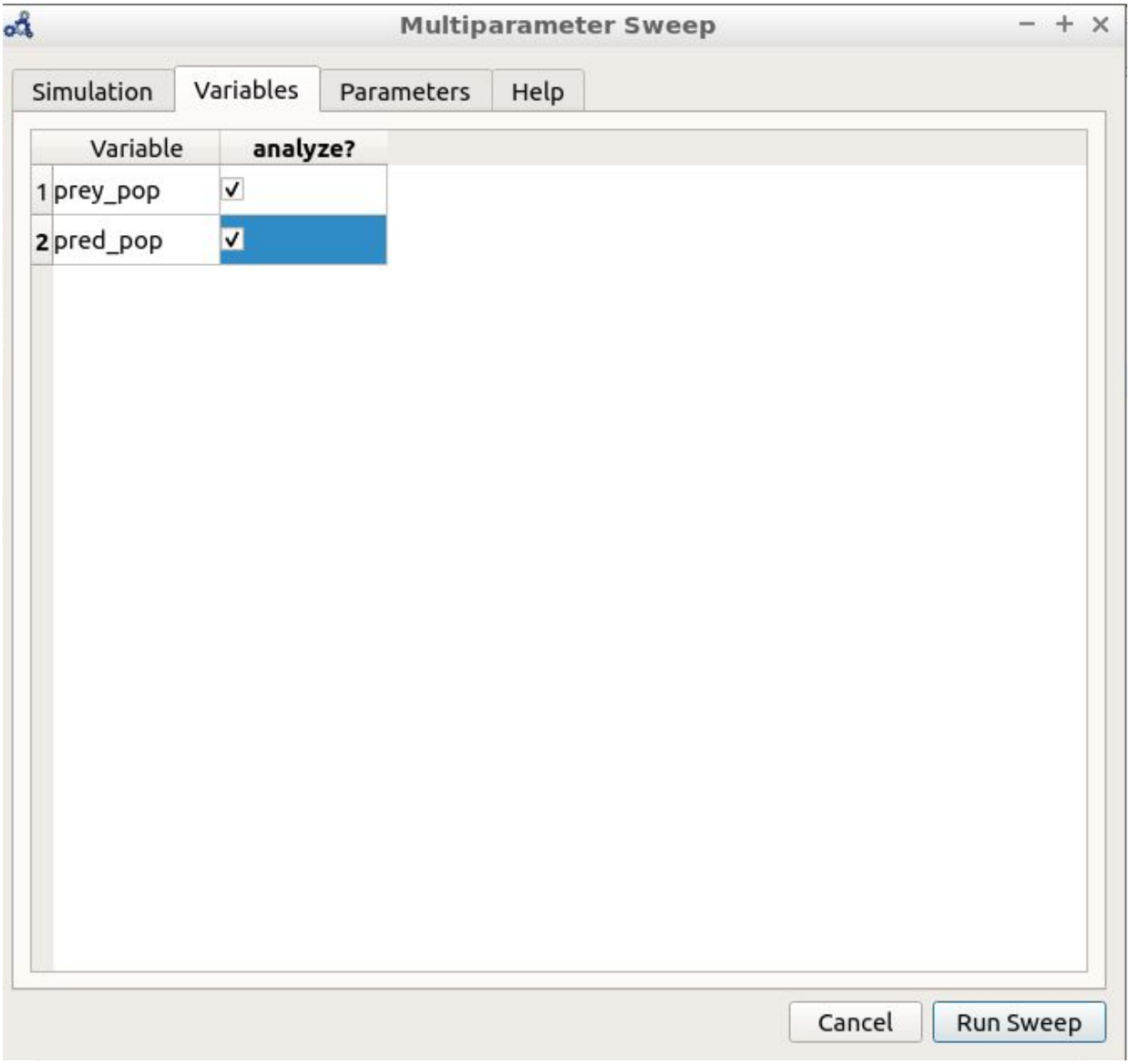
7. Now we would like to see what happens to *pred_pop* when the top 3 most influencing parameters are perturbed **at the same time**.
Open OMSens again and now choose “Multi-parameter Sweep”



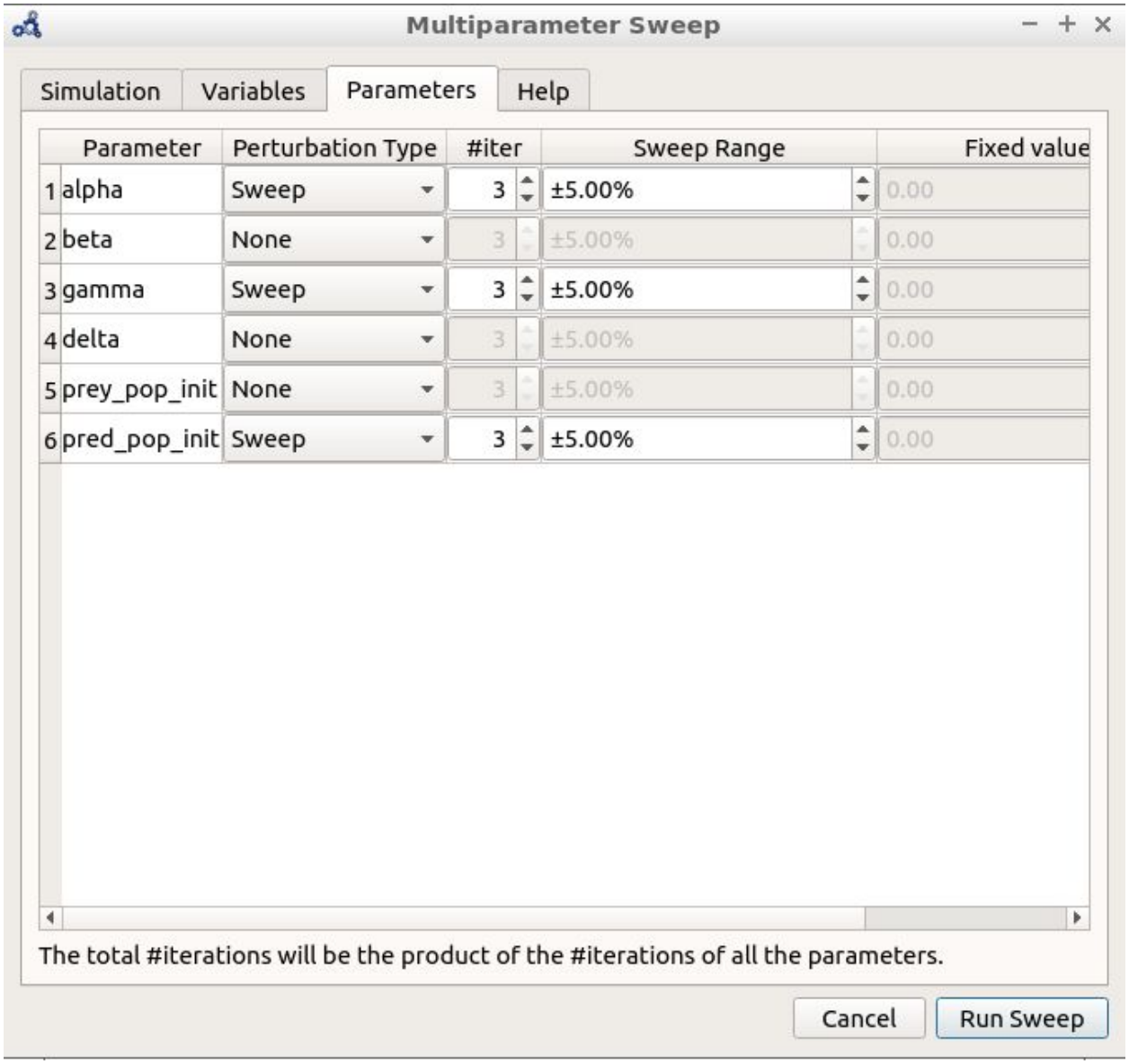
8. Set up the simulation settings.



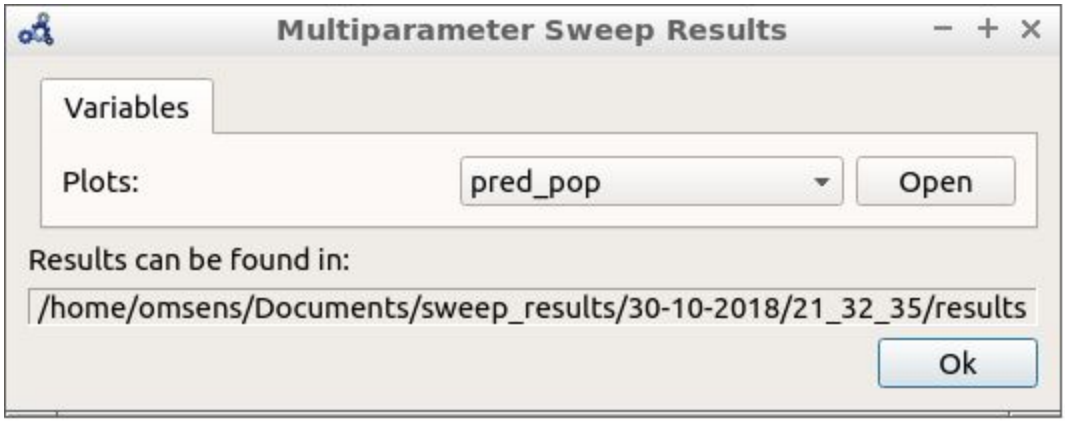
9. Select both variables



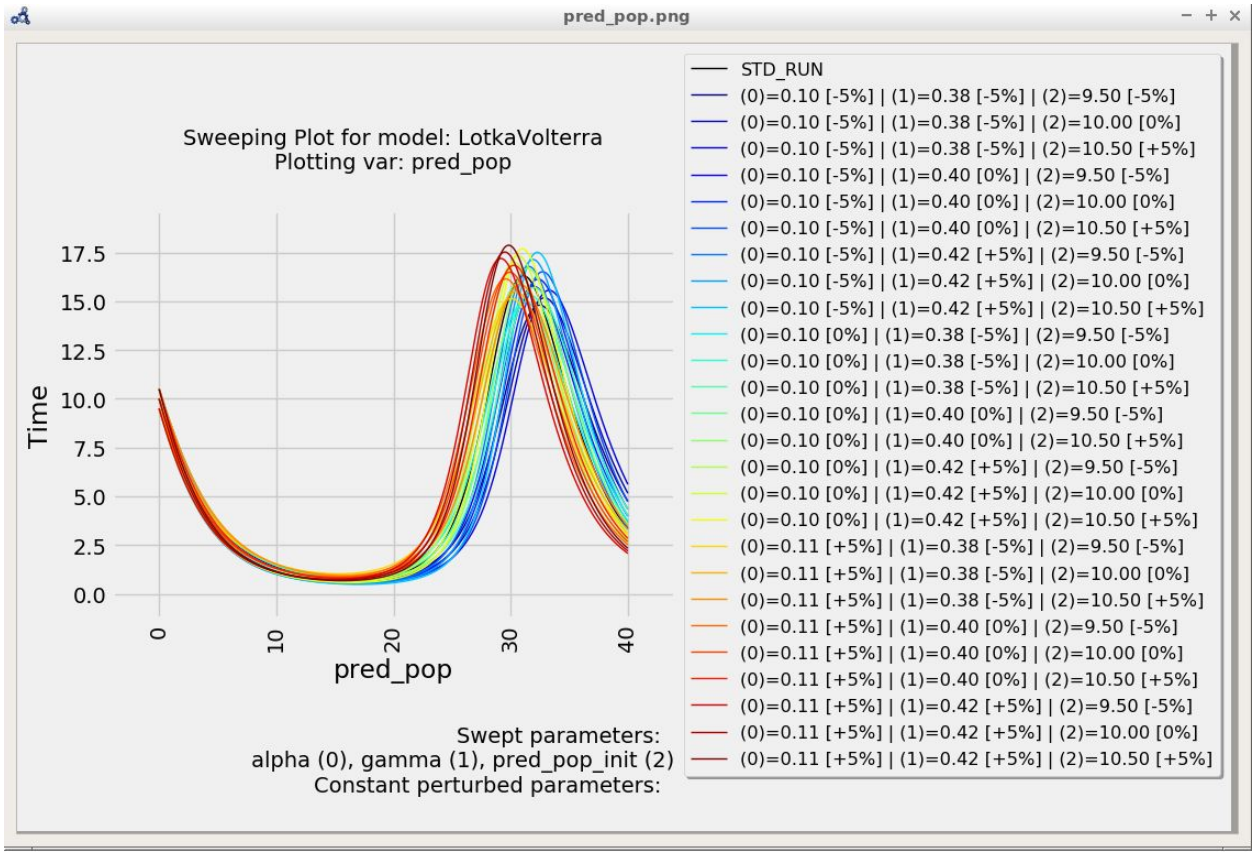
10. Choose to sweep *alpha*, *gamma* and *pred_pop_init* in a range of ±5% from its default value and with 3 iterations (#iter) distributed equidistantly within that range. Run the sweep analysis.



11. The backend is invoked and when it completes the analysis the following results dialog is shown. Open the plot for *pred_pop*



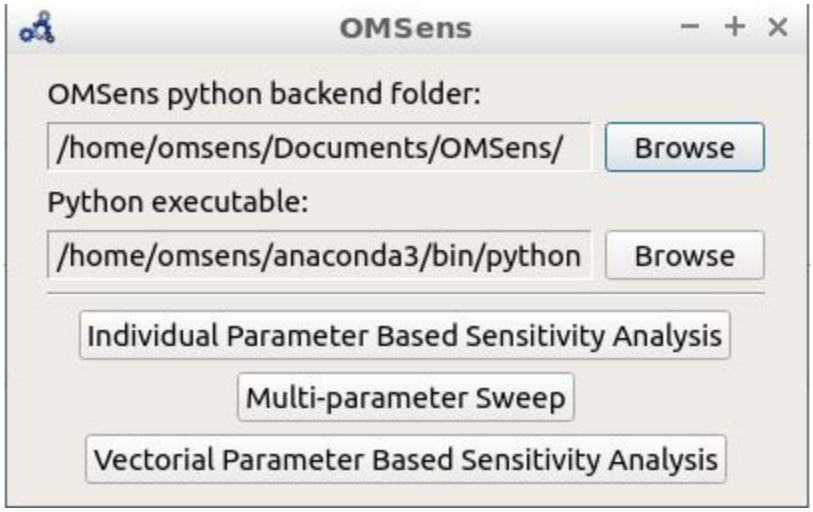
12. At time 40 the parameters perturbations with a higher predator population are all blue, but it's not clear which one. We need something more precise.



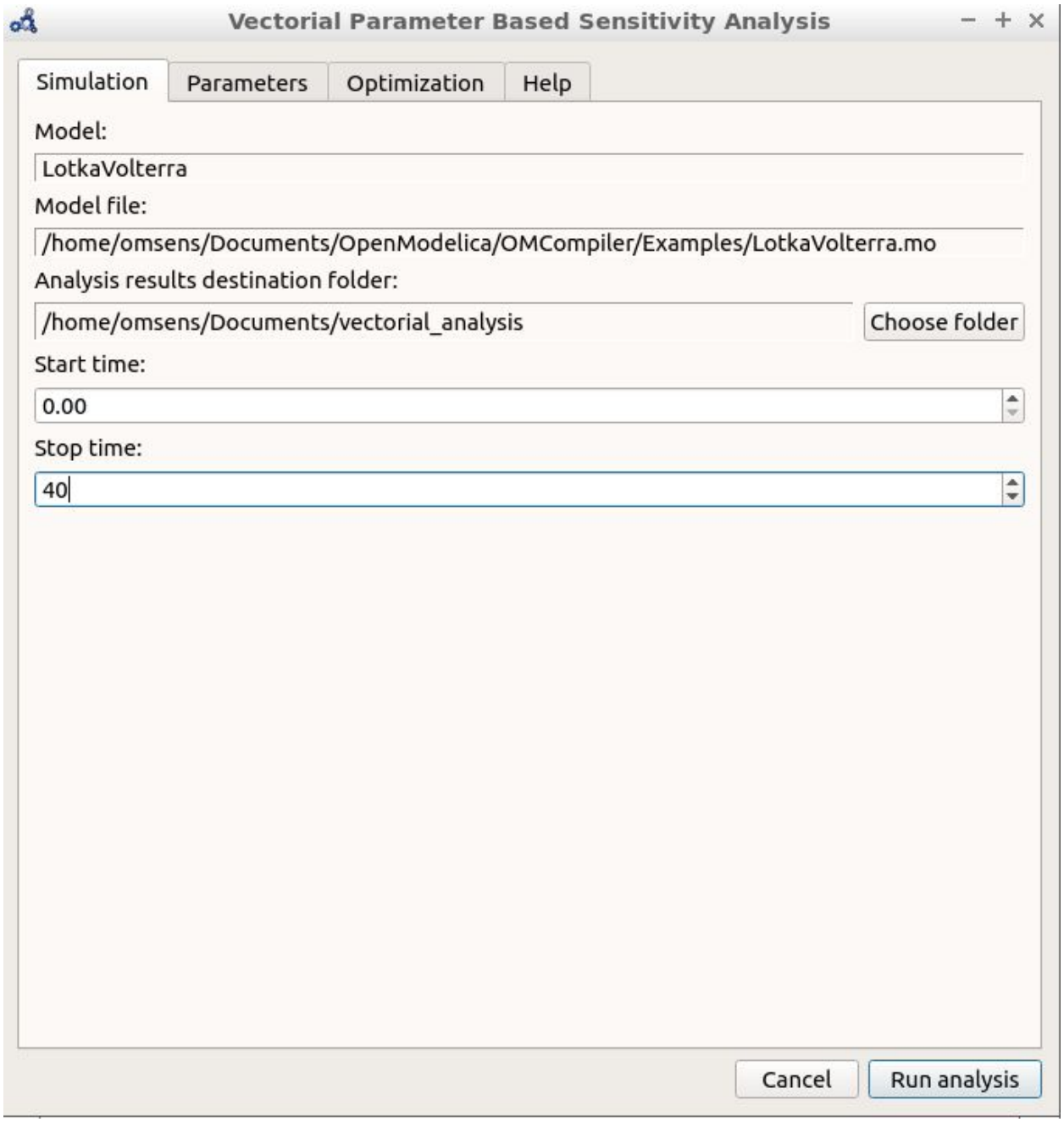
These results can be very informative but clearly the exhaustive exploration approach doesn't scale for more parameters (#p) and more perturbation values (#v) (#v^#p simulations required).

Using the Vectorial optimization-based analysis (see below) we can request OMSens to find a combination of parameters that perturbs the most (i.e. minimize or maximize) the value of the target variable at a desired simulation time.

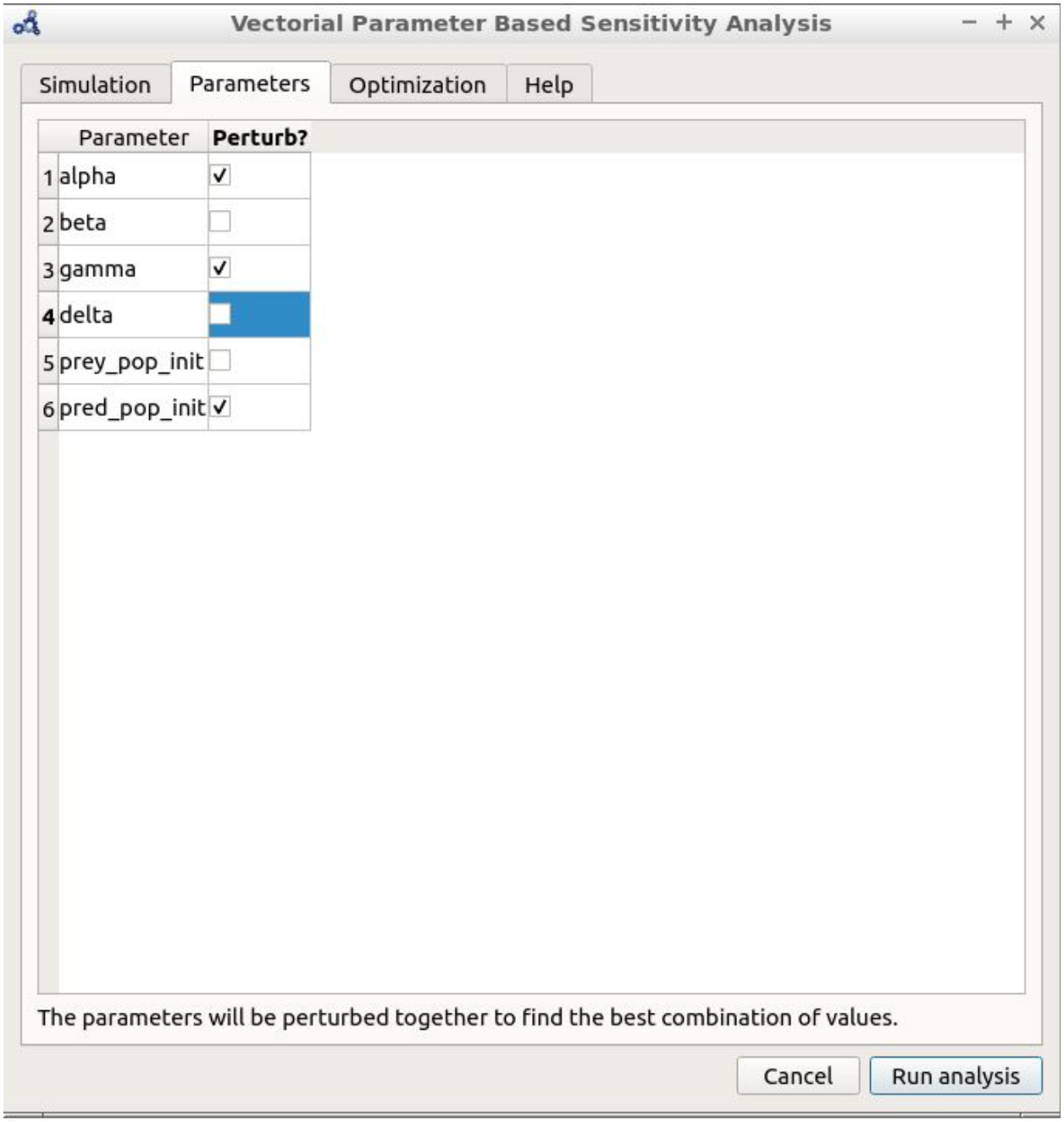
13. Open OMSens again and now choose Vectorial Parameter Based Sensitivity Analysis



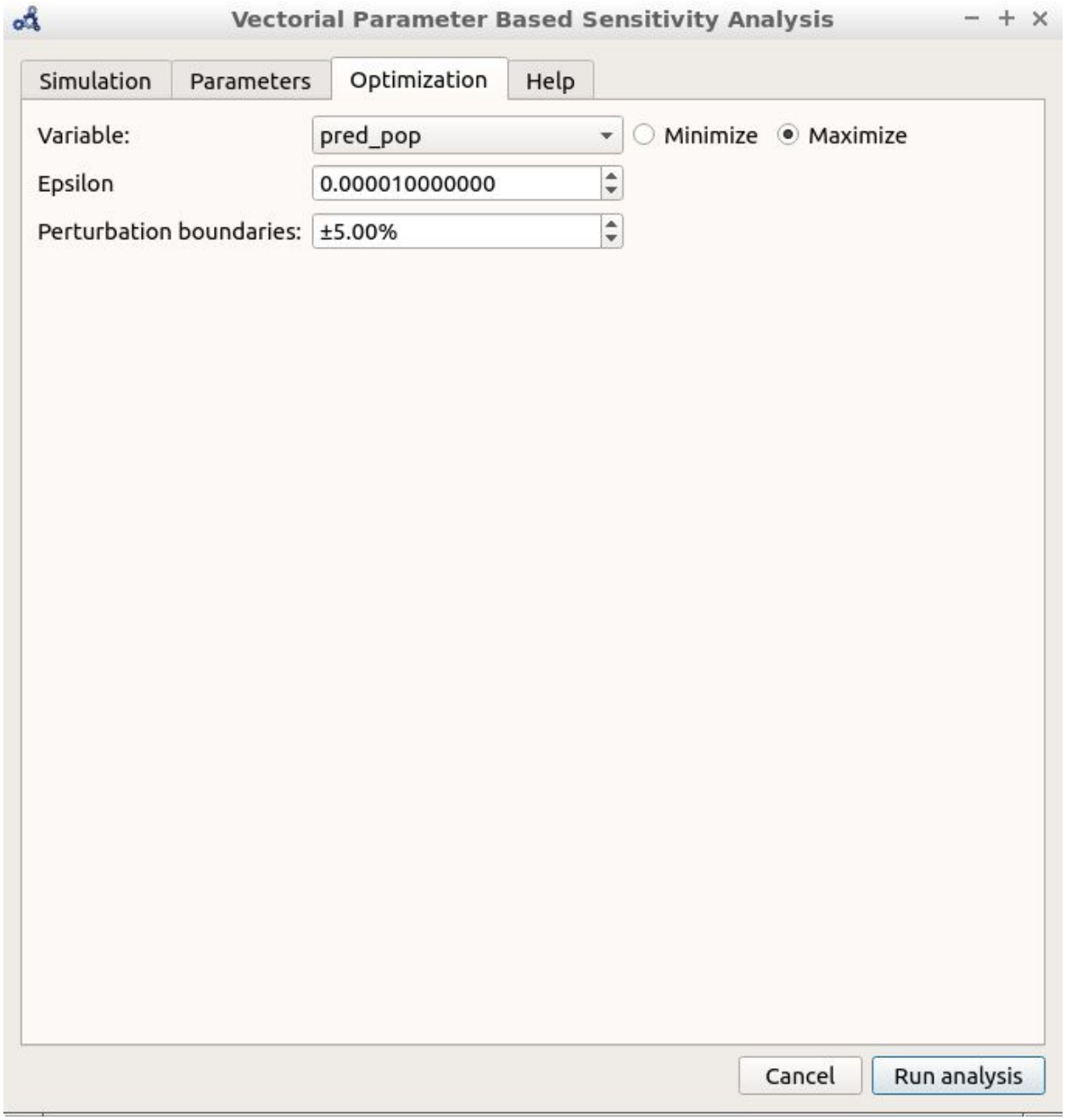
14. Setup the simulation settings



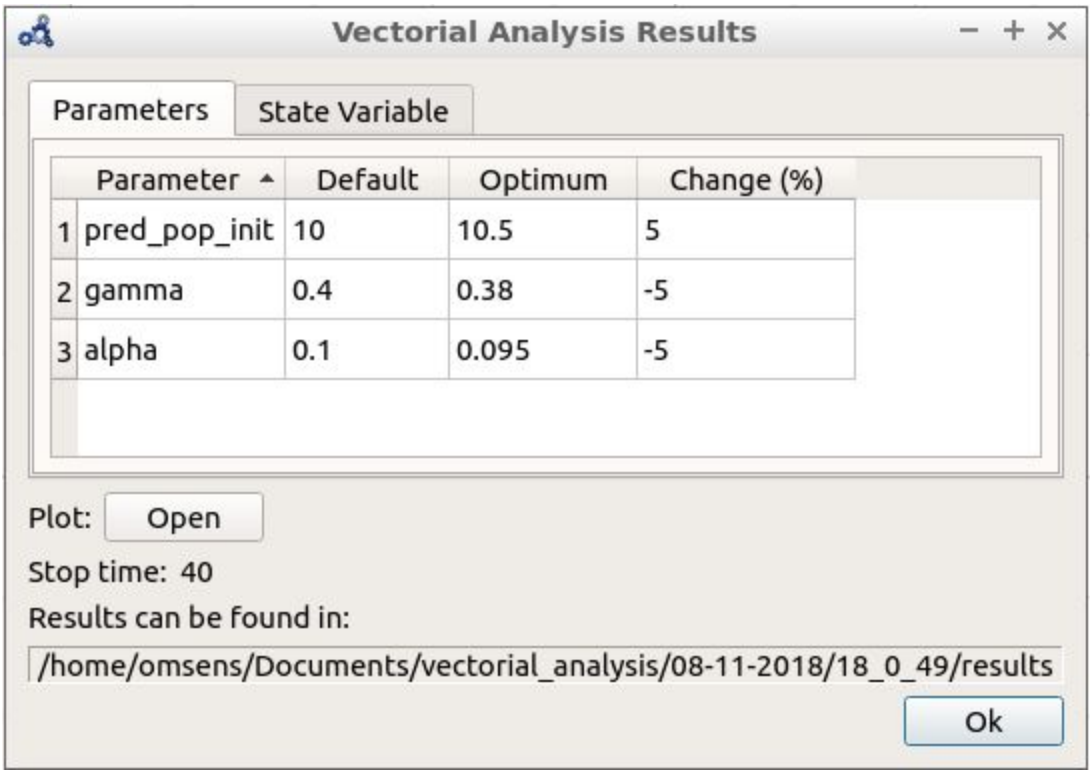
15. Choose only *alpha*, *delta* and *pred_pop_init* to perturb



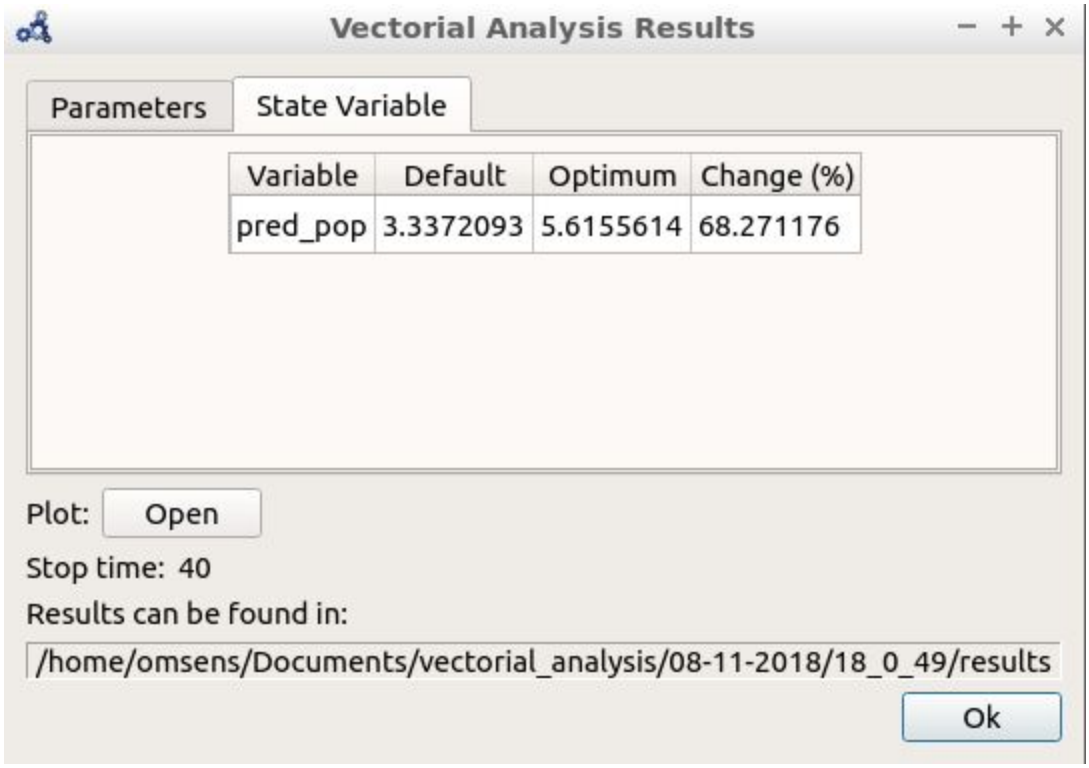
16. Setup the optimization settings and run the analysis.



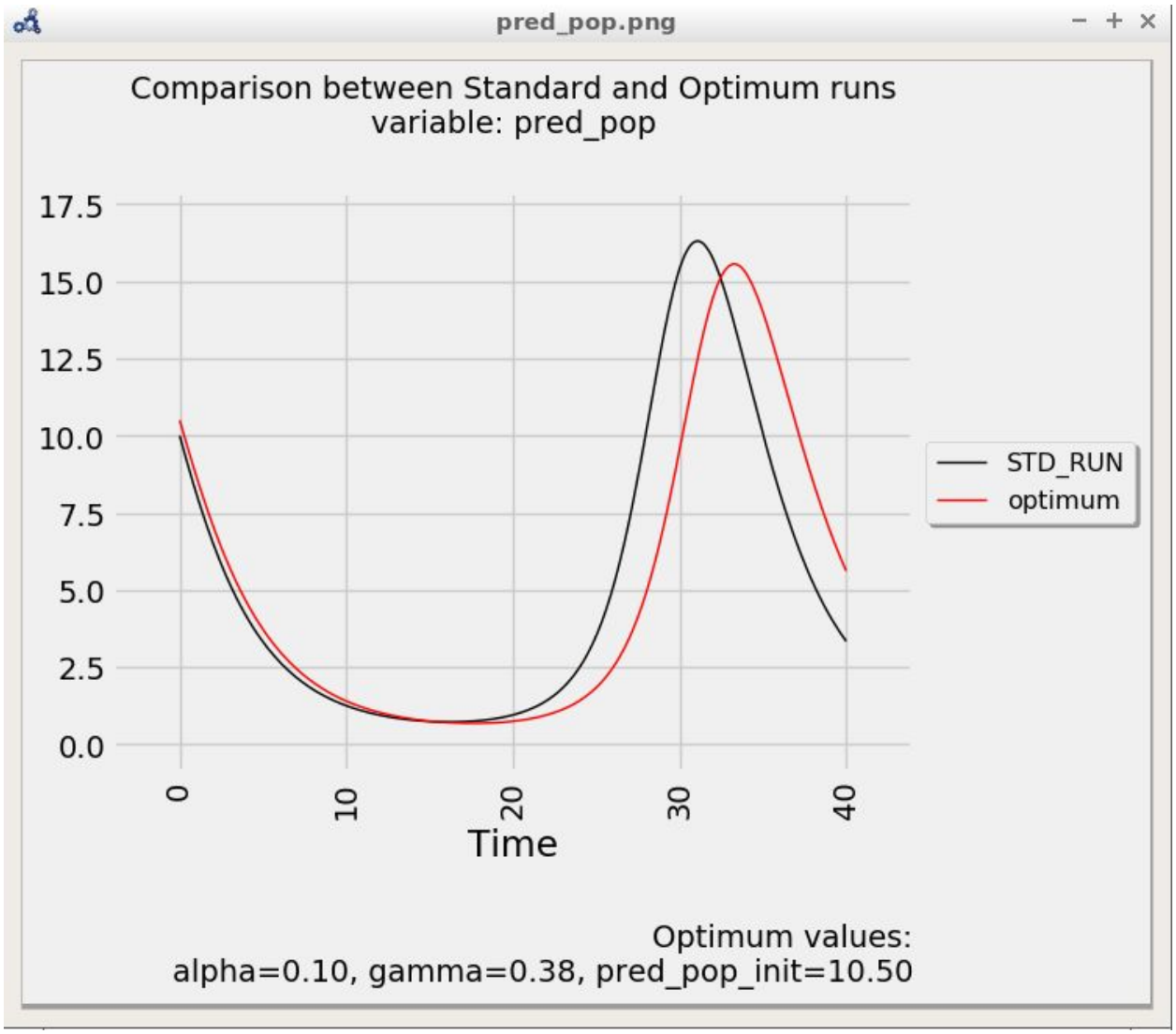
17. The backend is invoked and when it finishes a window with the results is shown. The “Parameters” tab shows the values found by the optimization routine that maximize *pred_pop* at t=40 s.



18. The “State Variable” tab shows the comparison between the values of the variable in the standard run vs the perturbed run at simulation time 40.

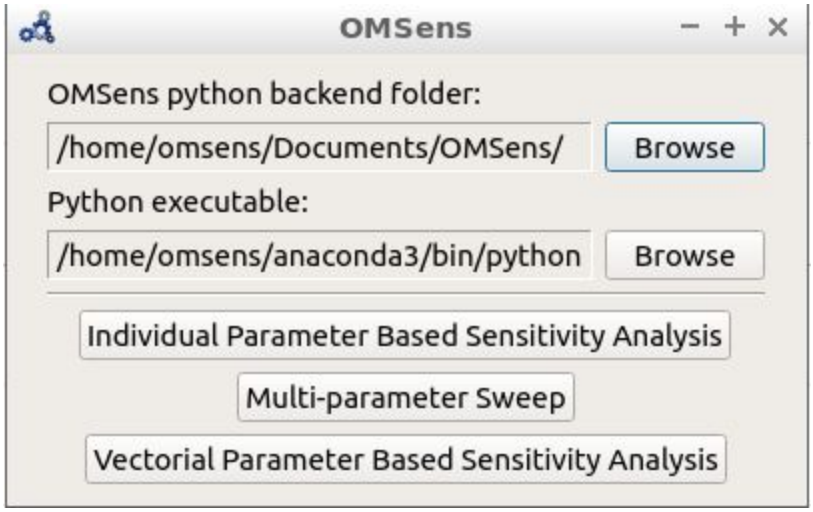


19. If we simulate using the optimum values and compare it to the standard (unperturbed) run, we see that it “delays the bell” described by the variable.



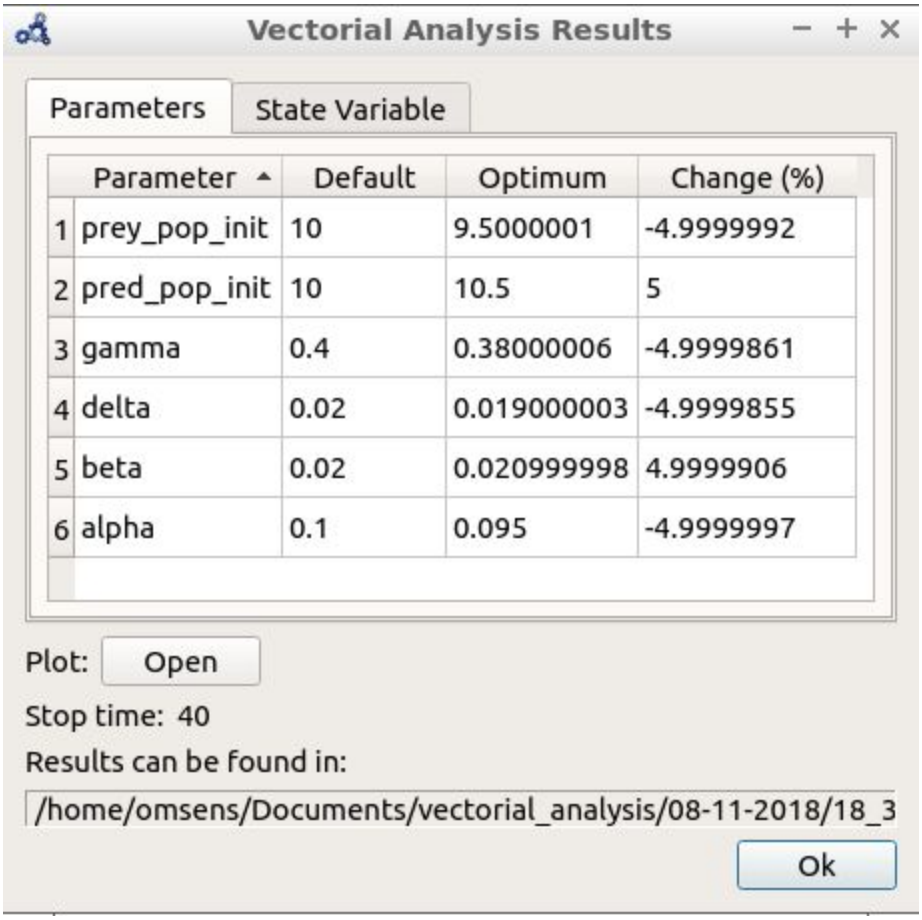
20. So far, we have only perturbed the top 3 parameters detected by the “Individual Sensitivity” method. Maybe we can find a greater effect on the variable if we perturb all 6 parameters. Running a Sweep is not an option as perturbing 6 parameters with 3 iterations each results in 3^6=729 simulations. We run another Vectorial Sensitivity Analysis instead.

Launch OMSens and choose Vectorial Parameter Based Sensitivity Analysis once again

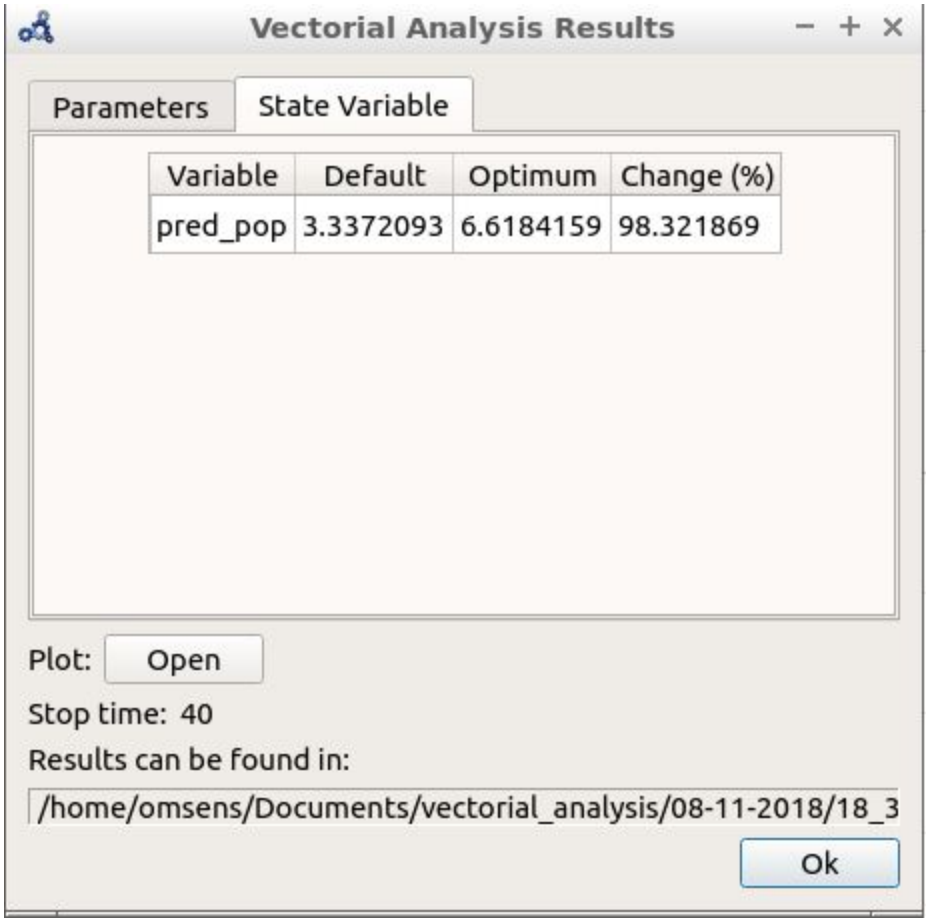


21. Setup Simulation and Optimization as before, but now choose to perturb all 6 parameters.

22. The parameters tab shows that the optimum value is found by perturbing all of the parameters to their boundaries.



23. The “Variable” tab shows that *pred_pop* can be increased by 98% when perturbing the 6 parameters as opposed to 68% when perturbing the “top 3” influencing parameters.



19. The plot shows again that the parameters found delay the bell-shaped curve, but with a stronger impact than before.