GSTDMB 2012
Dynamical Modelling for Biology and Medicine:

COPASI / Parameter estimation
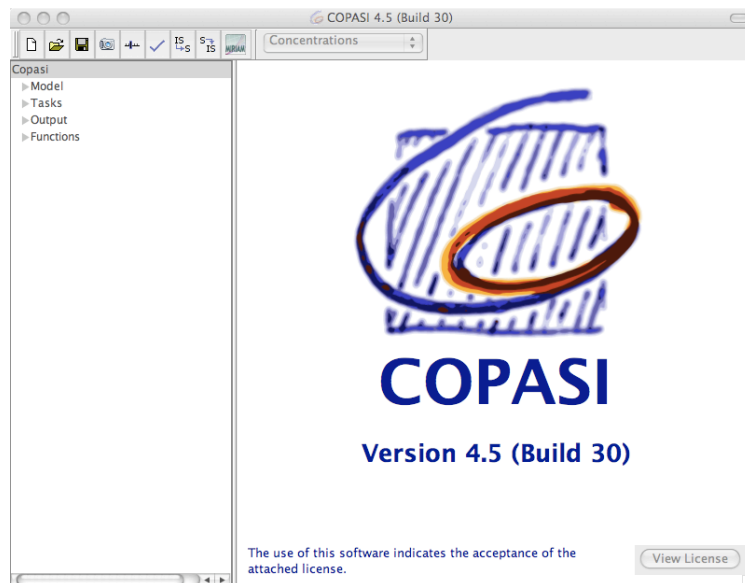
Markus Owen

1   COPASI

COPASI stands for *COmplex PAthway SImulator*. It is for the simulation and analysis of bio-chemical networks, although it can be used for more general models. COPASI can be used to develop models, to run simulations of different types (for example differential equations and stochastic simulations), and to carry out various forms of analysis, including metabolic control analysis, parameter estimation, and sensitivity analysis. COPASI is free for non-commercial use, and can be downloaded from: `http://www.copasi.org/tiki-index.php`

COPASI has more model analysis and simulation capabilities than CellDesigner, and will import SBML models, but does not currently support graphical design and layout of models. We'll use it because of its parameter estimation and stochastic simulation capabilities.

COPASI is available in University of Nottingham computer labs via:

Start > All Programs > UoN Software > School of Mathematics > Copasi > CopasiUI

Running it should give the following initial window:

Online documentation can be found at:

`http://www.copasi.org/tiki-index.php?page=od&structure=od`

Files for this practical can be found at:

`http://www.maths.nottingham.ac.uk/minisites/mathsforlife/gstdmb2012/`

2 Parameter estimation for a positive feedback model

Consider the following positive feedback model for mRNA x encoding a transcription factor y:

$$\frac{dx}{dt} = \frac{\lambda y^m}{\theta^m + y^m} - \mu_1 x, \qquad \frac{dy}{dt} = \delta x - \mu_2 y. \tag{1}$$

We will use the parameters $\lambda = 1, m = 3, \theta = 0.5, \mu_1 = 1, \delta = 1, \mu_2 = 1$. The phase plane for these parameters shows that the system is bistable (try sketching the nullclines, or see lecture 3, slide 20).

Rather than working through how to construct this model in COPASI, we provide our version at http://www.maths.nottingham.ac.uk/minisites/mathsforlife/gstdmb2012/ as "`positivefeedback.cps`"), and focus in the next section on parameter estimation.

For your information, Appendix A shows how to create the model in COPASI, and Appendix A.1 shows how to run simulations. For the above parameters, and for initial conditions $x(0) = 0.6; y(0) = 0.5$, the solution goes to a steady state with high x and y (see Figure 4a). Changing the initial condition for x to $x(0) = 0.4$, the system goes to a low steady state (Figure 4b).

2.1 Parameter Estimation outline

Suppose we had an experimental system for which we thought equation (1) was a good model, and we sought to parameterise the model. It is often a good idea to test ideas about parameter estimation, such as what data to collect, how many and which time points to use, etc, by generating simulated data from the model with known parameters, and then attempting to recover those parameter values by fitting the model to that *pseudo-experimental data.*

Appendix A.2 shows how to use COPASI to create text files containing pseudo-experimental data from simulations, suitable for subsequent parameter estimation. This process was used with the two sets of initial conditions described above (`ic1`: $x(0) = 0.6, y(0) = 0.5$; and `ic2`: $x(0) = 0.4, y(0) = 0.5$), and with 5 and 20 output intervals, to generate the four data files:
"`positivefeedback_timecourse_ic1_i05.txt`",
"`positivefeedback_timecourse_ic2_i05.txt`",
"`positivefeedback_timecourse_ic1_i20.txt`",
"`positivefeedback_timecourse_ic2_i20.txt`".

The pseudo-experimental data files contain a header line and three columns of data points—the first column is time, the second and third columns are the x and y data values at those times. This generated data can be read in by COPASI and used for parameter estimation for the same model.

The following steps show how to set up parameter estimation, based on the model file `positivefeedback.cps`, and the data with 20 intervals. Ensure `positivefeedback.cps` and the above data files are in the same folder. To load the file in COPASI, click "`File->Open`", navigate to the appropriate folder and select `positivefeedback.cps`. Then save the file with a new name—we used "`positivefeedback_parest_i20.cps`" to indicate that we will fit to the data with 20 intervals.

1. Click "`Tasks`", then "`Parameter Estimation`". This should bring up the parameter estimation window, as illustrated in Figure 1.

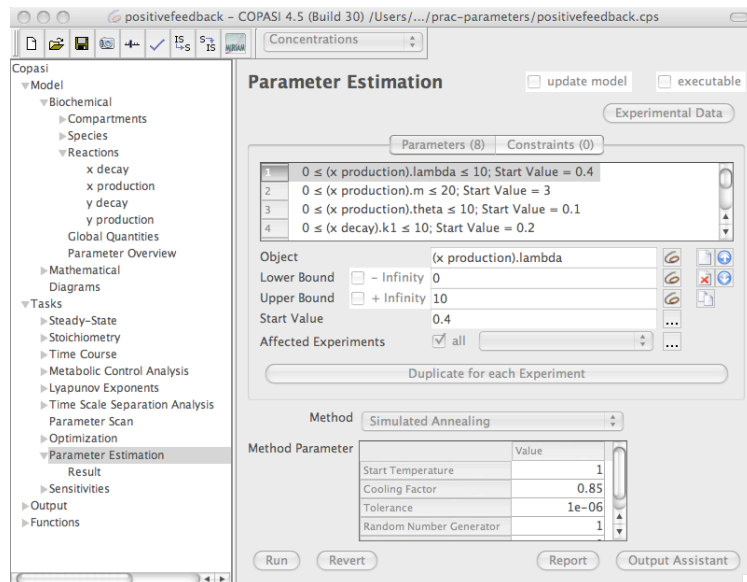2. Next we need to tell Copasi which experimental data to use.

Figure 1: The COPASI Parameter estimation panel.

Click "`Experimental Data`", and then click the blank paper to choose the first file "`positivefeedback_timecourse_ic1_i20.txt`".
Set the "`Experiment Type`" as "`Time Course`", make sure column 1 type is "`Time`", set column 2 to "`dependent`", then click the COPASI swirl and choose "`Species`", "`Transient Concentrations`", "`[x](t)`". Similarly, set column 3 to "`dependent`", then choose "`Species`", "`Transient Concentrations`", "`[y](t)`".

Then click the blank paper again to choose the second file "`positivefeedback_timecourse_ic2_i20.txt`" (for the second set of initial conditions). Click on the first file again in the "`File`" box, and select "`Copy settings below`" "`to next`", which should duplicate the column information for the second data file (i.e. so that the columns are "`Time`", "`x`" and "`y`").

Leave "`Weight Method`" as "`Mean Square`", and click "`OK`".

3. Now we need to specify which parameters to estimate.

Click for a new parameter to estimate (click the blank sheet of paper to the right of the "`Object`" line). Then click the Copasi swirl to select an object. Choose "`Reactions`" > "`Reaction Parameters`", then pick a reaction (e.g. x production) and a parameter. Repeat this until you have selected all 6 parameters in the model.

We should also fit the initial values of x and y for each experiment (such values would often not be known). Click the blank sheet of paper to the right of the "`Object`" line. Then click the Copasi swirl, choose "`Species`" > "`Initial Concentrations`", then pick "`[x](t=0)`". Uncheck "`Affected Experiments`" > "`all`", and deselect "`Experiment 1`" (so only "`Experiment`" is selected). Then repeat the addition of the parameter "`[x](t=0)`", but this time select only "`Experiment 1`". This allows two different initial values of x, one for each experiment.

Repeat the above for "`[y](t=0)`". You should now have 10 parameters to fit in total.

4. Click "`Output Assistant`" and select "`Parameter Estimation Result`" – this will plot the data and best fit solution at the end of the estimation procedure. Then click "`Output

Assistant" again, and select "`Progress of Fit`" – this will show the cost function as the estimation proceeds (as in Figure 2 below).

You can play with the "Output Assistant" to get different plots; go to the Output->Plots to delete any that you don't want anymore.

5. Then, in the "`Parameter Estimation`" panel, select a "`Method`" ("`Particle Swarm`" seems to be quite good for this problem), and click on "`Run`". It should start the fitting process, and once it is finished (or you have stopped the process), you can click on "`Result`" below "`Parameter Estimation`" (click on the triangle if you cannot see it) to see the parameter values that have been estimated. The "`Parameter Estimation Result`" window should show the final simulated values corresponding to the estimated parameters, and the experimental values, something like Figure 2 below.
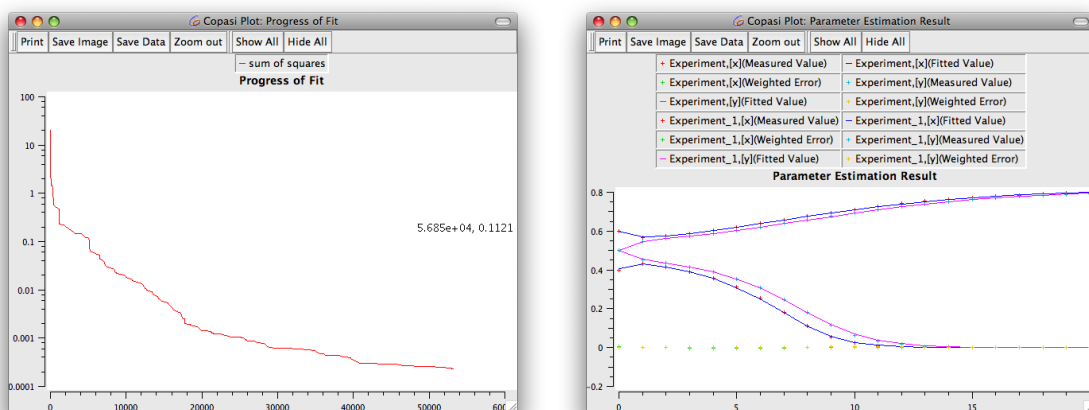

Figure 2: Parameter estimation "`Progress of Fit`" and "`Parameter Estimation Result`", for the positive feedback model in equation (1).

The "`Correlation`" tab indicates correlations between parameters. For our problem, you should see that "`lambda`" and "`(x decay).k1`" (which is μ_1) are strongly positively correlated, as are "`delta`" and "`(y decay).k1`" (which is μ_2). This is because it is only the ratio of λ to μ_1 that determines the steady states of the model (and similarly for δ and μ_2). So as long as these parameters vary with the same ratio, only the subtle details of the timecourse will change.

One way to better constrain the parameters for such a system could be to obtain separate data on the decay rates μ_1 and μ_2, e.g. by separately blocking transcription and translation.

6. You could explore the effect of choosing different fitting algorithms (there are many under the "`Method`" menu), different initial parameter guesses, or different parameter ranges. You could also change other details such as the number of data files (what if you only had data for one set of initial conditions), or the number of intervals in the data — e.g. by using 5 intervals, as in the data files:
"`positivefeedback_timecourse_ic1_i05.txt`",
"`positivefeedback_timecourse_ic2_i05.txt`".

4

You could also consider what would happen if you only had data on one variable. For example, it is often common to only have data on mRNA concentrations and not protein concentrations. You can mimic this by going to the "Experimental Data" window from the "Parameter Estimation" panel, and changing the "Type" entry for Column 3 (y) to "ignored", clicking "OK" and then re-running the parameter estimation. This should try to estimate the parameters only using the x (mRNA) data. Looking at the results, how well are the parameter estimated compared to with the full data available?

When you change such details, it is sensible to save your COPASI file under another (informative) name, so that you can compare the alternatives and return to previous versions if necessary.

3 Conclusions

You should now be able to use COPASI to estimate parameters for a model by fitting to experimental data (possibly using multiple data sets — we used two). This practical also illustrates the difficulties that can arise in parameter estimation, including correlations between parameters and the importance of having sufficient data to constrain the parameters.

For further information about constructing and simulating models in COPASI, see the Appendices to this practical.

A Creating the positive feedback model in COPASI

In the following I have tried to make the steps as simple as possible, but I cannot guarantee that there are no shortcuts possible. The easiest way to get a model is to load it from some other source. For example, this model can be found as a COPASI file "positivefeedback.cps" at
http://www.maths.nottingham.ac.uk/minisites/mathsforlife/gstdmb2012/positivefeedback.cps
ready for simulation. When following these steps, take care to save your work periodically.

1. *Create a new model:*
   Click the blank sheet of paper, which will open a panel where you can enter the name of your model. Leave the units alone and leave the "Rate Law Interpretation" pull-down menu as "deterministic model". Click "Commit".

2. *Add the model variables:*
   In the left panel are various lists which can be expanded/contracted by clicking on the small triangles. Use this to expand the "Model" list, and then expand the "Biochemical" sublist. Click "Species", then click in the "Name" box and enter "x". Then click "Initial Concentration" and enter 0.05 (this is just part of the initial condition for x and can be changed later if you like).

   Click in the next "Name" box and enter "y", and enter an initial value of 0.01.

   If you find yourself deeper within the "Species" hierarchy (for example if you double click on a species this will open a panel about that species), you can return to the species overview by clicking "Species" in the left panel.

3. *Add the x transcription term:*
   Click "Reactions" in the left panel, which will bring up a list of reactions in your model (empty so far). Click "Name" and enter "x production" (or another name if you like). Double clicking the row for this reaction will take you to a panel where the rate can be specified (or you can "Commit", then expand the "Reactions" sublist and click on "x production" to alter the details of that reaction).

   Enter " -> x; y" in the "Chemical Equation" box, indicating that "x" is produced, and "y" is a modifier (transcription factor x is not used up in transcription).

   Click "New Rate Law", which takes you to the "Rate law for x production" panel. Enter the formula

   ```
   lambda*y^m/(y^m+theta^m)
   ```

   and select "y" as a "Modifier" from the pull-down menu within the "Parameters" box. This states that the variable "y" in the above expression modifies the rate of x-production as defined above. Then "Commit" and return to the "x production" entry in the "Reactions" sublist. Choose "Rate Law for x production" in the "Rate Law" pull down menu.

   If the "Chemical Equation" is incompatible with the rate law (e.g. if you alter it by mistake), the text background should appear red.

   At this stage you might want to see what the corresponding equations for a deterministic ODE model would be. You can open the "Mathematical" sublist and then click "Differential equations" to display the ODEs. The "local parameters" pull-down menu allows you to choose between displaying the parameter values or their names. You

should check that at this stage the equations look as you expect. Compartment volumes appear all over the place as the system is set up to cope with multiple compartments with different volumes.

4. *Add the translation term* δx:
   Click "Reactions" in the left panel and enter "y production" as a new reaction name. Double click on the name and enter " -> y; x" in the "Chemical Equation" box, indicating that "y" is produced, and "x" is a modifier (mRNA is not used up in translation).

   Click "New Rate Law", which takes you to the "Rate law for y production" panel. Enter the formula

   ```
   delta*x
   ```

   and select "x" as a "Modifier". Then "Commit" and return to the "y production" entry in the "Reactions" sublist. Choose "Rate Law for y production" in the "Rate Law" pull down menu. "Commit", and then take another look at the "Differential equations".

5. *Add the decay terms:*
   Enter "x decay" as a new reaction name. Read the following two possibilities before deciding which to use to specify the rate of decay.

   (a) You can represent decay of x using the chemical equation "x -> ", selecting "Mass action (irreversible)". This will create a parameter "k1" which is a property of the x decay reaction, referred to as "(x decay).k1". If you do y in the same way, this will also have a parameter "(y decay).k1", so that so far the list of parameters would be:
   ```
   (x production).lambda
   (x production).theta
   (x production).m
   (x decay).k1
   (y production).delta
   (y decay).k1
   [x]_0
   [y]_0
   ```
   "Commit" this, and take another look at the "Differential equations", which should be finished now.

   (b) Alternatively, you can define your own rate laws for decay of x and y and give the parameters the names you choose. To do this, click "Reactions", enter a new reaction "x decay" with the chemical equation " x -> ". Click "New Rate Law" and enter the formula
   ```
   mu1*x
   ```

   with "x" set to be the "Substrate" from the pull-down menu within the "Parameters" box. "Commit" this, then return to the "x decay" reaction and choose "Rate Law for x decay" in the "Rate Law" pull down menu. "Commit", and then return to the "Reactions" table. Now add "y decay" to the list, with the chemical equation " y -> ". Click "New Rate Law" and enter the formula
   ```
   mu2*y
   ```

with "y" set to be the "Substrate" from the pull-down menu within the "Parameters" box. "Commit" this, then return to the "y decay" reaction and choose "Rate Law for y decay" in the "Rate Law" pull down menu. "Commit" and take another look at the "Differential equations" (see Figure 3).

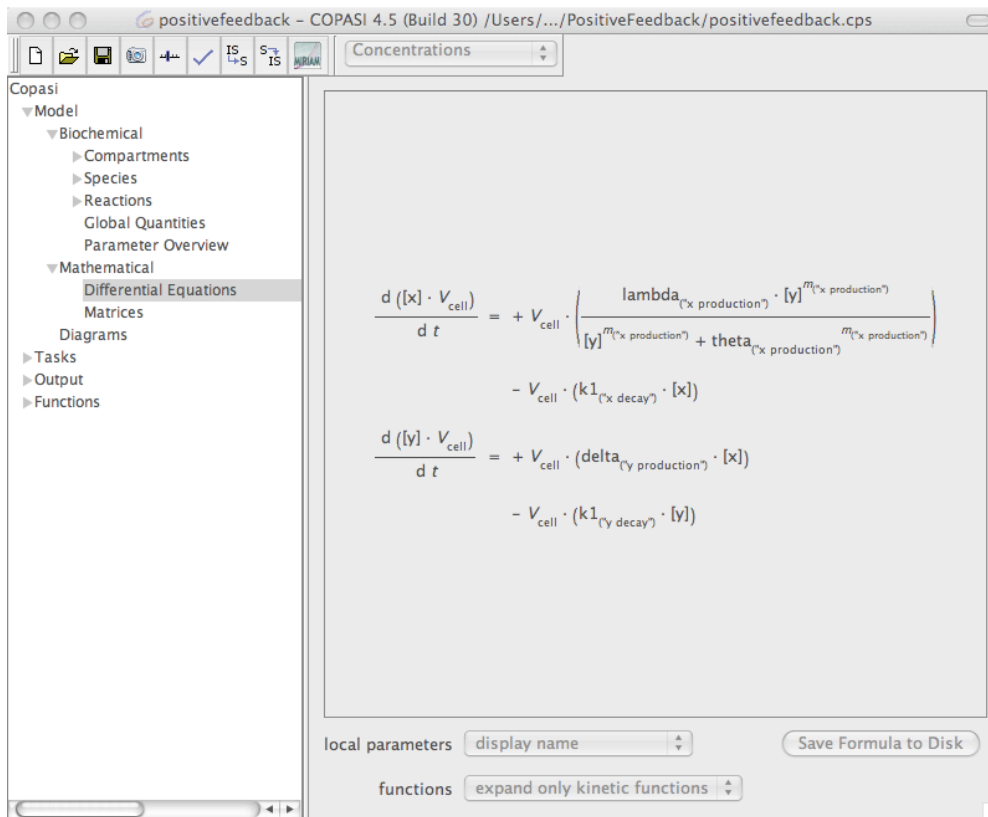Figure 3: The equations for the positive feedback model in equation (1), as constructed in CO-PASI.

6. *Set the parameter values:*
   Click "Model", then "Biochemical", then "Parameter Overview", and set the initial conditions to $x(0) = 0.6$; $y(0) = 0.5$ and the parameters to $\lambda = 1$; $m = 3$; $\theta = 0.5$; $\mu_1 = 1$; $\delta = 1$; $\mu_2 = 1$. Recall that, for example, μ_1 is the decay rate of x, and, depending on your choice above, this may have the same name, or use the notation "x decay, k1".

A.1 Simulating the positive feedback model in COPASI

Now let's run a simulation. Click "Tasks" in the left panel and then "Time Course". Check that the "Duration" is 20, and that the method is "deterministic".

Click on the "Output Assistant", and select "Concentrations, Volumes, and Global Quantity Values". You can define your own plots under "Output" in the left panel, but for our purposes this built-in plot is ideal – it shows the concentrations of both variables. Once you have chosen a form for the output, click "Run" to run the simulation. You should see a window with two curves, as in Figure 4a, where x and y go to a high value. For these parameters the system is bistable (try sketching the nullclines, or see lecture 3, slide 20). If you change the
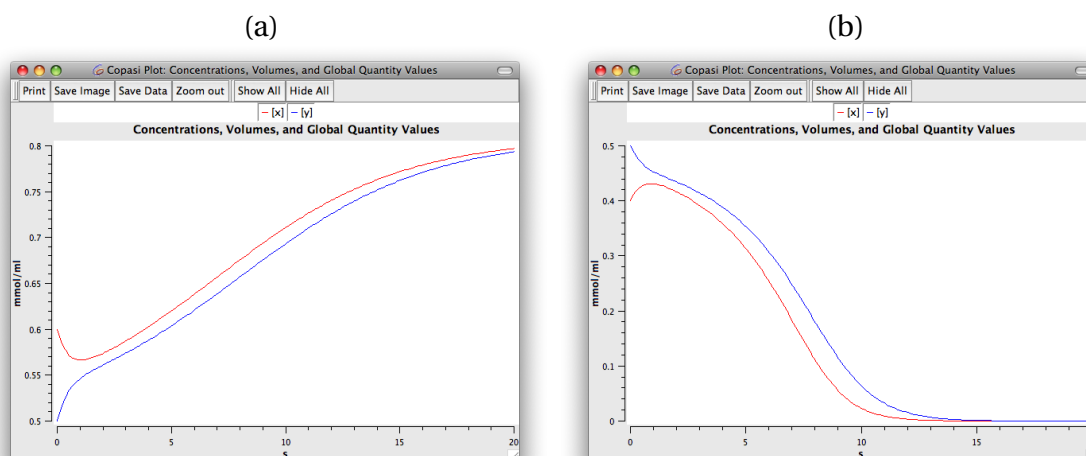
Figure 4: Simulations of the model in equation (1), using COPASI. **(a)** Initial conditions, $x(0) = 0.6$, $y(0) = 0.5$, solutions go to the upper steady state. **(b)** Initial conditions, $x(0) = 0.4$, $y(0) = 0.5$, solutions go to the lower steady state.

initial value of x to 0.4 (use the "`Parameter Overview`") and run the simulation again, you will see that the solution goes to low values for x and y (Figure 4b).

You can also run a parameter scan that generates simulations for a range of different initial x values. Go to the "`Parameter Scan`" entry in the "`Tasks`" menu. Select "`Parameter Scan`" as the "`New Scan Item`" and click "`... Create!`".

Make sure the "`Subtask`" is "`Time course`" and the "`always use initial conditions`" and "`output during subtask execution`" checkboxes are ticked. Click the COPASI swirl to the right to select "`Species`", "`Initial Concentrations`", "`[x](t=0)`". Set "`intervals`" to 10, "`min`" to 0 and "`max`" to 1 — this will give initial x values from 0 to 1 in steps of 0.1. Then click "`Run`", and you should see 11 sets of curves appear in the plot window, as in Figure 5, illustrating the bistability in the system.

A.2   Generating pseudo-experimental data

In order to generate pseudo-experimental data, we will output simulation results at selected time points. To do this, we set the interval size / number of intervals accordingly (e.g. set "`Intervals`"=20), and use "`Report`" to specify an output file. Click on "`Output Assistant`", scroll down to "`-- Reports`", select "`Time and all Variable Values (Concentration Units)`" and click "`Create!`". Then, in the "`Time Course`" panel, click "`Report`" to choose a file to output data to (e.g. for the first set of initial conditions and 20 intervals, we used "`positivefeedback_timecourse_ic1_i20.txt`").

Use this process with the two sets of initial conditions described above (`ic1`: $x = 0.6$, $y = 0.5$; and `ic2`: $x = 0.4$, $y = 0.5$), and with 5 and 20 output intervals, to generate the four data files:
"`positivefeedback_timecourse_ic1_i05.txt`",
"`positivefeedback_timecourse_ic2_i05.txt`",
"`positivefeedback_timecourse_ic1_i20.txt`",
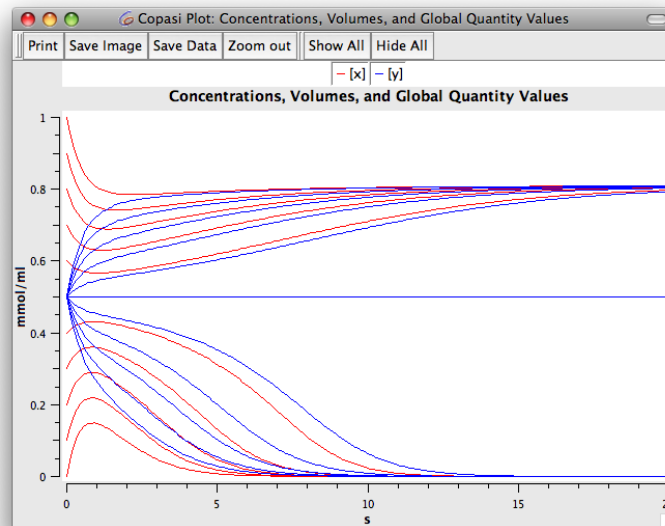"`positivefeedback_timecourse_ic2_i20.txt`".

Figure 5: Simulations of the model in equation (1), using COPASI, with the initial x value ranging from 0 to 1 in steps of 0.1, and with $y(0) = 0.5$. For $x(0) < 0.5$, solutions go to the lower steady state, and for $x(0) > 0.5$ solutions go to the upper steady state.

B   Other features of COPASI

To explore other features of COPASI, you could load a number of example models, by clicking "Examples" in the "Choose a file" dialog (which appears when you click File->Open, or when you click on the "Open" icon). These examples include the Brusselator model that was discussed in the lectures. Try loading this one, then click "Tasks"->"Time course", and click "Output"->"Plots" and choose the "ConcentrationPlot". Make sure the "active?" checkbox is ticked. Then if you click "Run" in the "Time Course" task, you will see plots of the Brusselator state variables oscillating. You can try changing parameter values, etc, via the "Model" group.

COPASI can load and save models in the Systems Biology Markup Language (SBML), a common format for sharing models. There are lots of other tools that can handle SBML, including CellDesigner and "toolboxes" for MATLAB. There is also a repository of biological models at http://www.ebi.ac.uk/biomodels-main/. Most models in the repository are maintained in multiple formats including SBML. In principle you should be able to download a model and load it into COPASI and other SBML compliant packages.