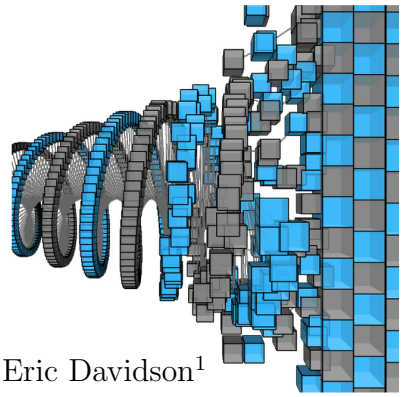
GeNeTool
Tutorial



Emmanuel Faure[1,2], Isabelle Peter[1], and Eric Davidson[1]

[1]Division of Biology, California Institute of Technology, Pasadena, CA 91125, USA
[2]Centre de Recherche en pistémologie Appliquée, Ecole Polytechnique, Centre National de la Recherche Scientifique, 75015 Paris, France

Abstract

GeNeTool is designed to compute Boolean gene expression in time and space as an output of gene regulatory interactions, including under conditions in which these interactions are perturbed, either experimentally or purely in silico. Your starting point for use of **GeNeTool** will be a gene regulatory network (GRN) consisting of a set of regulatory genes plus the regulatory interactions between them. If this GRN is based on experimental data, then **GeNeTool** will allow you to compare the gene expression that it computes using information derived from your GRN model to the observed expression of the same genes. **GeNeTool** can also be used to test purely theoretical network architectures for their spatial and temporal expression outputs. Following is a verbal account of the way users can enter into the program the many different types of genomic, regulatory, and developmental information that **GeNeTool** accommodates, and of the diverse operations included in its repertoire. Information and instructions are entered into the program through an interactive graphics interface, as illustrated in detail in an accompanying video.

Contents

List of Figures

1 Downloading and Opening GeNeTool; Handling Model and Menus; Exporting Results

GeNeTool operates on any ordinary contemporary laptop or desktop computer, PC or Apple. The program can be downloaded at **GeNeTool**

Specific Instructions

1.1 Installing GeNeTool

1.1.1 Requirement

You first need to be sure that Java is already installed on your computer (as is usually the case). If not, download Java.

1.1.2 The application

To download and install the application :

- click on the logo of your operating system
- accept downloading the file
- decompress it (with a double-click on Mac or Linux, or in Windows you can use 7-Zip)
- run the software by double-clicking on the application

If you have any troubles for the installation or running the software , email Emmanuel Faure emmanuel.faure@polytechnique.edu

1.2 New and existing models

Create : The program opens with an empty model template, in place, so you can immediately start building your own model.

Open : If you wish to open an existing model, go on the main bar *File* menu, click on *Open Model*[1] , and select the desired model ; it will exist as an XML file.You can also just drag your model file in the **GeNeTool** window.
→ *Remark:* You can download and open this example.

Save : As in many programs, in the *File* menu you can save your model (click on *Save Model*) or save it under another name (click on *Save Model as*) if you wish to create another version. It will automatically be saved in a XML file extension.

1.3 Menus

GeNeTool contains several menu windows inside the application itself.

Open : When you start the application with an empty model, all the menus are closed. To open them, click on the blue box of the name of the menu you desire. For example, a simple click on *Genes* opens the menu "Genes". By default many menus are hidden before they become useful, but once you start to enter different kinds of information, more menus will become accessible.

[1] All the buttons of **GeNeTool** are described in *italic* in this tutorial

Close : It may become desirable to close some windows, and this can be done in the same way as the window is opened, by a simple click name of the window.

Drag : If you need a specific visual configuration, you can drag each window by maintaining a click on the name of the desired window, moving the mouse, and releasing it at the desired position.

Configuration : In addition, on the main bar is a *Windows* menu with the list of all possible menus. A click on the desired menu can alternatively open or close it. If you wish to open all menus click on *Cascade*. Clicking on *Best* will open them in the best configuration.

1.4 Exporting results

The XML file contains all the information needed by **GeNeTool** to run your model. Do not try to modify elements inside the XML file itself, as this will destroy it. On the main bar is an *Export* menu :

Export Genes You can export the list of Genes with their vector equations in a PDF format (click on *Rules in PDF*) , or a common Text format (click on *Rules in TXT*)

Export Model You can export the matrix of model results in PDF format (click on *Model in PDF*) , or in a reusable CSV format (click on *Model in CSV*)

Export Data You can export the matrix of your observed data in PDF format (click on *Data in PDF*)

Export Comparison You can export your personalized comparison in PDF format (click on *Compare in PDF*)

2 Entering Explicit and Implicit Data from your GRN Model

The GRN models here considered as the starting point for use of **GeNeTool** are static gene interaction models for given developmental stages and temporal periods built in **BioTapestry**, or an equivalent platform. They explicitly display GRN circuitry, and at each node they indicate the regulatory inputs responsible for the state of activity (ON or OFF) of the gene represented at that node. They explicitly or implicitly include information on the ways the inputs at each node are handled by the respective *cis*-regulatory control systems, that is, the *cis*-regulatory logic operations at that node. In building a **GeNeTool** model, the total informational content at each node of the GRN is captured in a Boolean vector equation, or if there are (or may be) alternatively utilized *cis*-regulatory modules controlling the gene, by more than one vector equation.

2.1 Entering Genes, (video 1)

The first step in building a **GeNeTool** model is to enter by name all of the genes included in the starting GRN(s) (see video illustrating this operation). Each gene has its own field and each is accompanied by a field wherein the vector equation(s) for that gene will be built.

Specific Instructions

2.1.1 Create a Gene

- Open the *Genes* Menu, and click on the blue '+' to create a new gene.
- Enter the name of the gene, and click *OK*.
- You can create as many genes as you want.

2.1.2 *Gene* Menu (fig. 1)

- On the left part of the window, all the genes created are listed in alphabetic orders. You can move the elevator on the left part of the list (with a simple drag on it or using the wheel of you mouse) to go up and down of this list.

- A click on the name of the gene in in this list will put it in the center of the window as the active gene, where you can see more details on the gene.

You have 3 check boxes:

- The first one on the left (checked by default) is for defining whether the active entity is a gene or something else such as a maternal input, or a signaling interaction

- The second box (not checked by default) is to define whether the active gene is known or unknown.

- The third box (not checked by default) is to define whether the active gene is expressed ubiquitously.

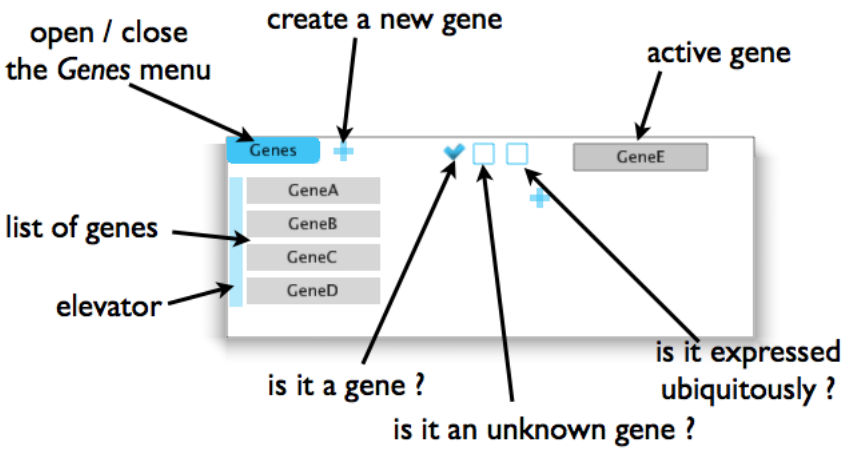


Figure 1: *Genes* Menu

2.2 Entering vector equation objects and operations derived from the GRN

The vector equation includes all the regulatory information required to control expression of the gene, according to the GRN. This consists essentially of (i) the identities of the multiple transcription factor inputs that control the gene; (ii) the logic operations by which the multiple inputs are processed; (iii) temporal data indicating real or abstract intervals between successive regulatory steps Of these (i) and (ii) are extracted from the GRN, and (iii) is supplied from additional sources (see below).
Since the input factors are products of other genes in the system they are already present in the gene array with which we began. The way the inputs and operations are combined in constructing a vector equation is illustrated in words in the following example, which pertains to gene X which is ON if inputs generated from Gene A AND from Gene B, are both present, i.e., because Gene A AND Gene B were both ON; but this is the result only when the product of Gene C, which encodes a repressor, is NOT present. Using 1 to represent ON and 0 OFF, the statement would read *"if [A=1 AND B=1] AND NOT C=1 then X=1, else 0"*. **GeNeTool** uses such instructions to evaluate the expression states of Genes A, B, and C at each time step (and in all relevant spatial domains if the system has multiple domains) and then to compute the output of Gene X, 0 or 1, at every time and place.
The total repertoire of logic functions used in **GeNeTool** and their definitions can be seen in Table 1.
For an illustration of the means by which the appropriate input genes and operator logic functions are moved into a vector equation template, see video.

Entering regulatory inputs and logic operators that act on them

2.2.1 Operators

Open the *Operators* menu. You have the list of all possible operators implemented in **GeNeTool**. They are organized by type:

- Bracket: [ ] , in order to force the software to focus on a specific sequence order, expressions in brackets are processed first
- Unary : **NOT**, this is the negative boolean operator, it can be in front of a Gene , brackets, or a temporal operator.
- Binary : **AND**, **OR** , both are common AND and OR boolean operators. They can be placed between 2 expressions.
- Temporal : **AT**, **AFTER**, **PERM** → *See Section 3*
- Spatial : **IN**, **CC**, **NCC_n**, **NCC_d** → *See Section 4*
- Constraint : <, > → *See Section 6.1*

2.2.2 Create a Vector Equation (fig. 2, video 2)

- Open the *Genes* Menu and pick one name on the genes list to set as the active gene.
- Under the name of the active gene, click on the blue '+' to create a new Vector Equation. By default, the expression inside the vector equation is empty, and the statement will read *"if <empty box> then 1 else 0"*.
- To add a regulatory input inside the vector equation, drag it from the list, maintaining a click on the name of the desired input, moving the mouse, and releasing it inside the empty box of the vector equation.
- To add a logic operator inside the vector equation, drag it from the operators list to within the vector equation box.
- You may use as many inputs and operators as you wish, but the vector equation must be appropriately written for it to be computed by **GeNeTool**. To the right of the Vector Equation, a blue check will tell you if this the case or not. If it is not, you can alter the vector equation by dragging new elements inside the Vector Equation, or moving them to another position.
- You can also remove elements from the Vequation Equation by dragging them to the trash (located on the bottom left corner of the **GeNeTool** window)
- By default the end of the statement is *"then 1 else 0"*, but if you click on *'1'* or/and *'0'*, this statement can be reversed.

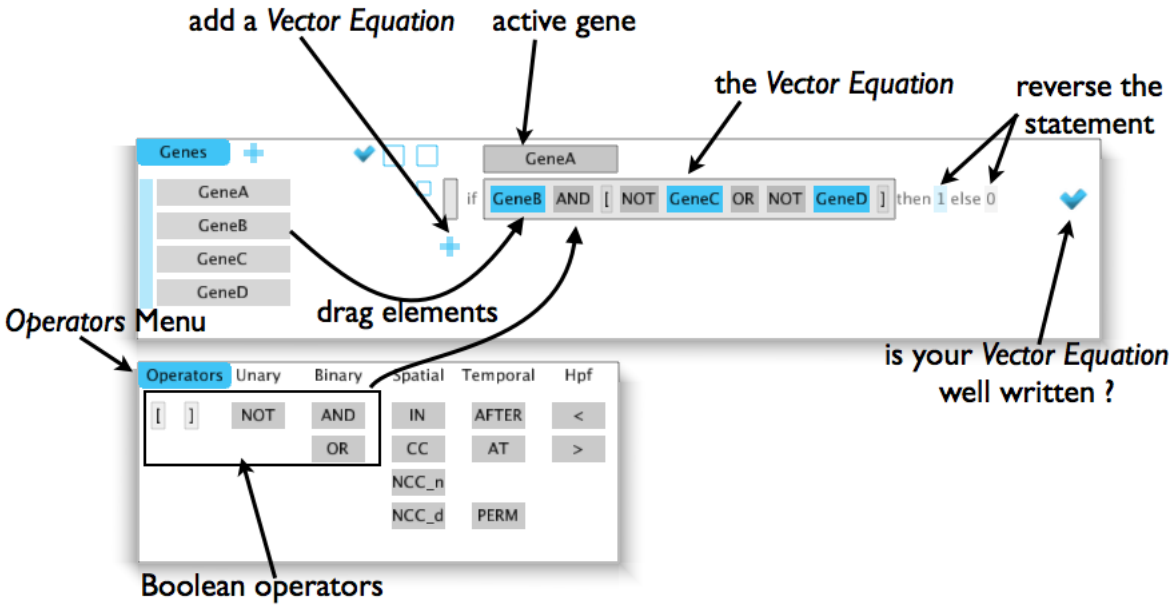


Figure 2: Create a Vector Equation

2.3 Alternative *cis*-regulatory modules

Many regulatory genes operate at different times or places under control of alternative *cis*-regulatory modules, which respond to alternative regulatory states. Alternative *cis*-regulatory modules may be indicated by experimental data e.g., *cis*-regulatory analyses, or they may be implied by responses of a gene to different inputs in different contexts, or they may be assumed in building a synthetic model. In such cases the vector equation for a given gene must contain a single vector equation for each of the alternative *cis*-regulatory modules, which states the inputs to which that module responds and the logic operations it performs on these inputs. In addition a module choice vector equation must be included which states, for example, that if the output of Module A = 1 OR the output of Module B = 1 the output of the gene is 1, else 0. Where alternative modules are in play, the number of vector equations governing the computation of expression of a given gene in **GeNeTool** is thus the number of modules + 1.

Specific Instructions

Alternative vector equations for a given gene

2.3.1 Entering alternative vector equations

- You can add as many Vector Equations as you want for each Gene by simply continue to click on the blue '+' under the Gene Name (same as 2.2.2).
- Double-click on the box just on the left of the *if* to specify the name of the VE[2]. (If you want to change the name, just repeat a double-click on the name).
- Click on the small blue box on the left of the VE name to specify if this specific VE is unknown or not (by default un-click means known)

2.3.2 Entering module choice OR statements (fig. 3)

- Click under the small blue box to specify a default Vector Equation (gray box).

[2] Vector Equation

- Drag the name of the other Vector Equations and the operator OR to the default box.

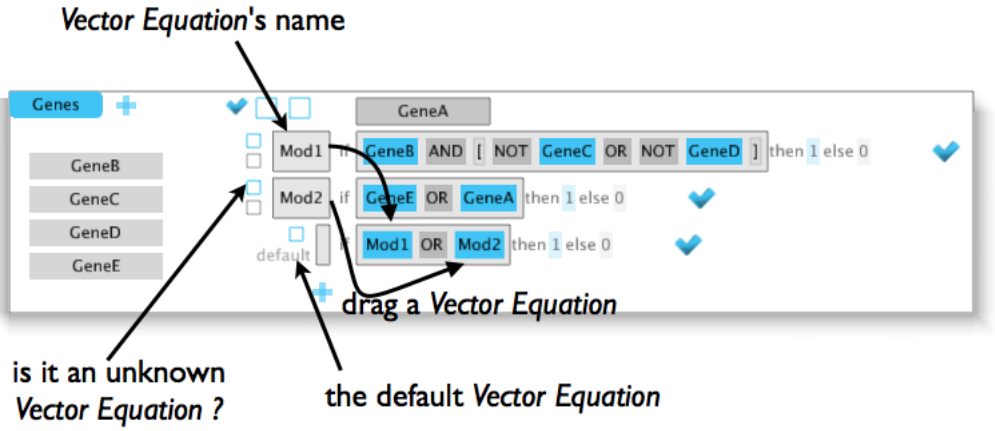


Figure 3: Module Choice

3 Temporal Steps in the Boolean Model

The Boolean model progresses by successive steps at each of which the state of expression of each gene is newly computed. The pace, in real time or arbitrary units, is set by the interval between the steps, the *step time*. This is the interval between activation of a regulatory gene, say Gene A, and activation of its direct target gene, Gene B. Thus it includes the time for transcription of A, processing and export of the mRNA, translation of the encoded transcription factor, import of the transcription factor back into the nucleus, accumulation of enough of it there to significantly occupy the *cis*-regulatory target sites of Gene B, and initiation of transcription of B. The step time used in **GeNeTool** can come from any external source. In sea urchin embryos at 15 °C a single, constant 3-hr step time which was separately computed earlier from measured molecular process rate constants turns out to fit the data very well. Other systems living at higher temperatures will have shorter step times. The step time does not have to be constant, and could be computed from measurements for each step by the usual ODEs, or could be chosen arbitrarily for certain synthetic applications. The principle by which the step time is introduced into **GeNeTool** is by inserting it as a lag in each vector equation. For this we utilize a function called *AT*, as follows. Suppose the step time is 3 hr, and we use the same vector equation logic as in the above example, i.e., *"if [A=1 AND B=1] AND NOT C=1 then X=1, else 0"*. We would write *"if [AT-3 A=1 AND AT-3 B=1] AND [NOT AT-3 C=1], then X=1, else 0"*. In running **GeNeTool** the intervals when the computation is assessed should be a fraction of the step time, chosen in reasonable conformity to the resolution of the available data. There are two other (more rarely used) temporal functions in **GeNeTool**. A feature of repression often encountered is its multistep nature, in which the repressor enters the system and binds to its *cis*-regulatory target site, and this in turn triggers a sequence of subsequent events that results in permanent (at least with respect to the process considered) transcriptional quiescence. These events begin with the action of co-repressors or chromatin modifiers, initially mobilized to the gene by binding of the sequence specific repressor. Ultimately however the repressed state is maintained by specific chromatin states or protein complexes, even when the repressor is no longer present. For this situation **GeNeTool** utilizes the function *PERM*. For example, *if [PERM-3 A=1] then X=1, else 0"*, means that 3h after A turns ON, X turns OFF and stays OFF forever. Similarly, **GeNeTool** allows a gene to be activated by a positive input but remain ON thereafter irrespective of the continued input from the activator. For this the function *AFTER* is used. For example, *"if [AFTER-3 A=1] then X=1, else 0"*, means 3h hour after A turns ON, X turns ON but it then remains ON forever (unless it is later specifically repressed).

3.1 Entering *AT* statements in vector equations (fig. 4)

- The *AT* operator is a in the Temporal column in the *Operators* Menu. It can be placed in front of a Gene.
- The same as in 2.2.2, drag the *AT* operator into the Vector Equation.
- When the *AT* operator is in the Vector Equation box, it is combined with a hyphen and a number. The number specifies the specific step time for this operator. This step time is by default set at 0; to change the value, move the mouse over this number (it appears in blue), and a simple click will increase the value by 1. While your mouse is over this number, you can also use your keyboard up and down arrows to modify this value.

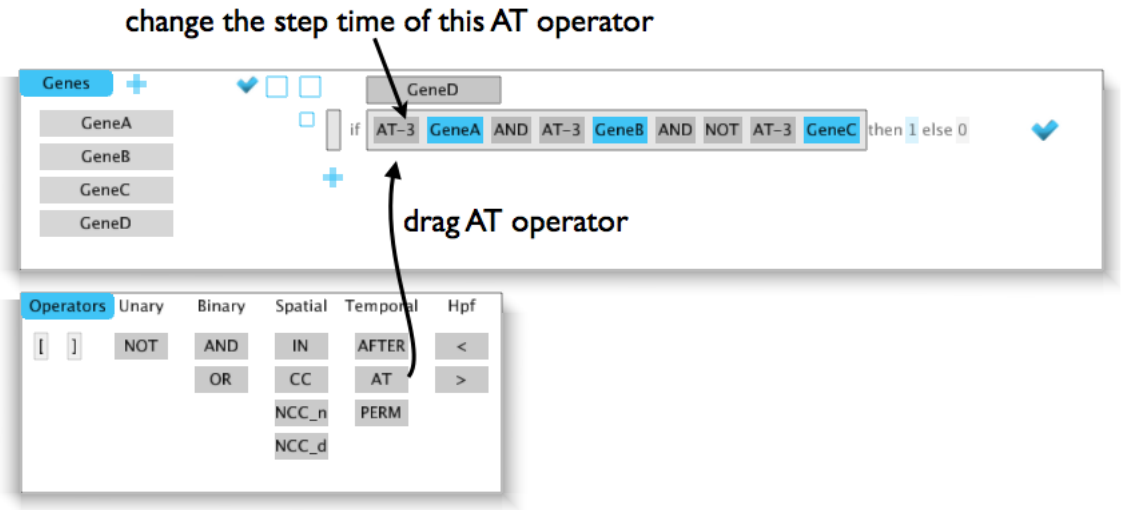


Figure 4: Drag the *AT* operator to the Vector Equation

3.2 Entering *PERM* statements in vector equations

- The *PERM* operator is a in the Temporal column in the *Operators* Menu. It can be placed in front of a Gene.
- As in 2.2.2, drag the *PERM* operator into the Vector Equation.
- When the *PERM* operator is in the Vector Equation box, it is combined with a hyphen and a number. The number specifies the specific step time for this operator. This step time is by default set at 0; to change the value, move the mouse over this number (it appears in blue), and a simple click will increase the value by 1. While your mouse is over this number, you can also use your keyboard up and down arrows to modify this value.

3.3 Entering *AFTER* statements in vector equations

- The *AFTER* operator is a in the Temporal column in the *Operators* Menu. It can be placed in front of a Gene.
- As in 2.2.2, drag the *AFTER* operator into the Vector Equation.

- When the *AFTER* operator is in the Vector Equation box, it is combined with a hyphen and a number. The number specifies the specific step time for this operator. This step time is by default set at 0; to change the value, move the mouse over this number (it appears in blue), and a simple click will increase the value by 1. While your mouse is over this number, you can also use your keyboard up and down arrows to modify this value.

4 Spatial Domains of the Developmental System to be Modeled

A primary though not the only use of **GeNeTool** is to model GRNs which progressively establish diverse regulatory states in the different spatial domains of the developing embryo or body part. In general, after very early stages, spatial subdivision in development is accomplished by inter-domain signaling, where a domain or territory is defined as a set of cells which express a common regulatory state at a given time, which mechanistically defines the developmental fate(s) of their descendants. Thus to model a typical developmental process through time, is necessary to keep track of the relative positions of the various regulatory state domains as these progressively change, through subdivision, migration or growth. This becomes immediately obvious if we consider the requirements for the program to be able to compute the effects of signaling; a short range signal can only affect regulatory states in closely adjacent cells, for example. Furthermore the ancestry of the cells of a given domain must be known since its regulatory state at any given time is computed from the regulatory state of the ancestral cells at the previous step time. Thus both the longitudinal history of domain descent and the mutual positions of regulatory state domains as time progresses must be input for the **GeNeTool** computation to be effected.
In **GeNeTool** three types of geometrical relationship are used to define relative domain position through time. If two domains are immediately contiguous, their relation (at a given time) is symbolized *CC*, i.e., accessible to signals that require immediate cell contact (e.g., Notch signaling); if not contiguous but nearby, the domains are *NCC_n* (non-contiguous, near), i.e., these domains could be accessible to interaction by means of short range signals which extend only a few cell diameters; if distant the domains are *NCC_d* and could interact only with long range signals.

Specific Instructions

4.1 Entering the number and identities of the ultimate regulatory state domains that the model will compute

4.1.1 Create a domain

- Open the *Domains* menu, and click on the blue '+' to create a new domain.
- Enter the name of the gene, and click *OK*.
- You can create as many spatial domains as you want.

4.1.2 *Domains* Menu (fig. 5)

- On the left part of the window, all the domains created are listed in alphabetic order. You can move the elevator on the left part of the list (with a simple drag on it or using the wheel of your mouse) to go up and down this list.
- A click on the name of the domain in this list will place it in the center of the window as the active domain, where you can see more details.

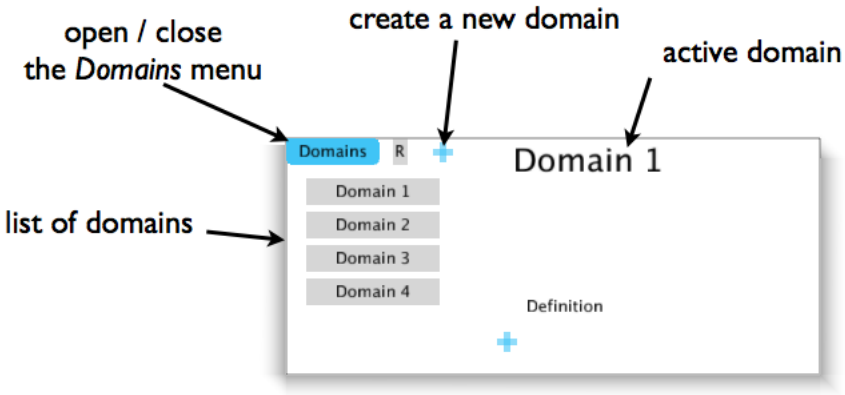


Figure 5: *Domains* Menu functionalities

4.2 Entering the early to late tree of domain descent that explicitly provides the progenitors of each domain through time

4.2.1 Create progenitors (video 3)

- Open the *Progenitors* menu,
- Specify your time unit (by default *hpf*, hours post fertilzation): click on *'hpf'* and enter the desired unit time name
- Specify the last step time (in your time unit) of your model: move your mouse over *"Max 0"*, and use your up and down arrows to change the value.
- Click on '+' to add a new progenitor
- Enter its name, and click *OK* (after this, you can still modify the name with a double-click)
- Move your mouse over the number after *"from"* and use your up and down arrow to modify the first time step of this progenitor
- Move your mouse over the number after *"to"* and use your up and down arrow to modify the last time step of this progenitor
- you may add as many progenitors there are.

4.2.2 Create the lineage tree for each domain (fig. 6, video 4)

- in the *Domain* menu, there is an empty box under *"Domain Ancestors"*
- Drag, in order, the list of progenitors through time into this box, by clicking on the name of each specific progenitor and dropping them in the box.
- In case of a mistake, you can remove progenitors from this box by dragging them to the trash

4.3 Entering the relative spatial relations (*CC* or *NCC*) of the domains with respect to one another (fig. 7, video 5)

- In the *Domain* menu, click on the '+' under Definition to add a new spatial relation for this active domain. This will create an empty box
- First, drag the *CC* or *NCC* operator from the *Operators* menu inside this box.

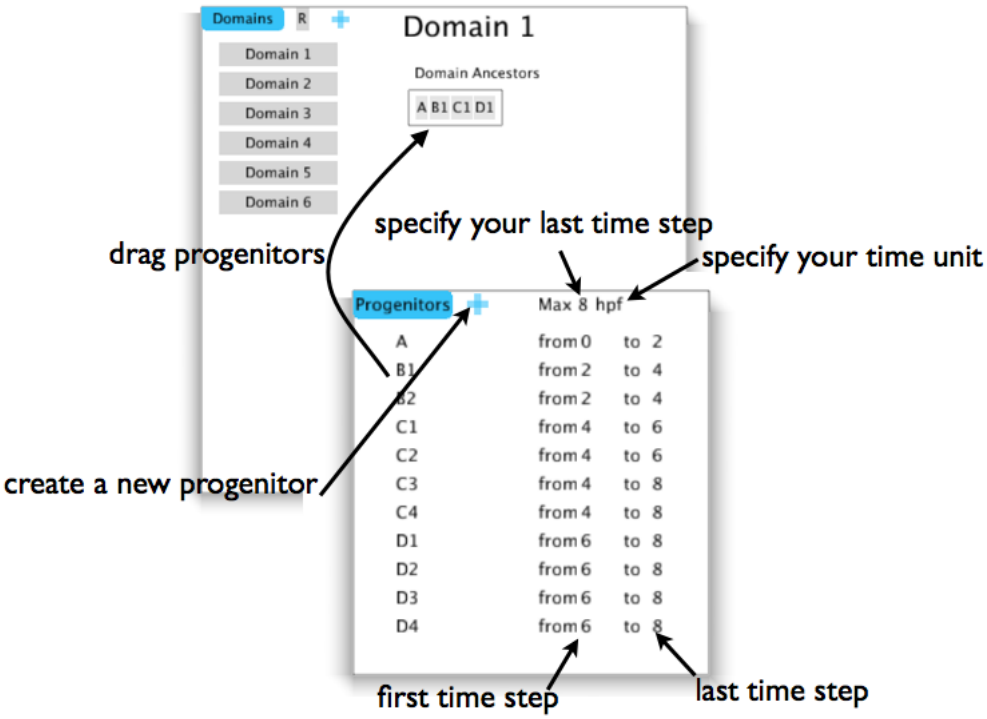


Figure 6: Create a lineage tree

- Specify the temporal beginning and end of this spatial relation: move your mouse on the first number after *CC/NCC* and use your up and down arrows for the first time step, and do the same on the last number to to denote he end. Of this spatial relationship.
- Drag from the Domains list, the corresponding domain of this spatial relation.
- You can add as many spatial relations as you want by continuing click on '+'
- You can delete a spatial relation by clicking on '-'

5 Signaling Interactions

In **GeNeTool** a canonical feature of many types of developmentally significant signal is built into the program. This is that whatever the particular biochemical pathway of signal transduction, there is always in the end an already present, specific transcription factor the state of activity of which depends on whether it is in a cell receiving the particular signal for which that factor mediates the transcriptional response (immediate early response factor). The signal transduction system works by affecting the activity of the immediate early response factor. The identity of the target genes ultimately affected by signal reception is encoded in the regulatory genome, according to presence of *cis*-regulatory target sites for the immediate early response transcription factor. Furthermore, the immediate early response factors behave in a Janus-like way: In a cell receiving the signal they act alone or with other transcription factors to activate their target genes; in a cell inaccessible to that signal, the same factors instead bind an obligate repressor and specifically prevent target gene expression. In **GeNeTool** signaling functions are encoded in terms of the state of these Janus immediate early response factors (J factors), one for each type of signal. Thus a vector equation determines the expression of the signal ligand gene in the domain producing the signal; in the receiving cells (required to be *CC*, *NCC_n*, or *NCC_d*, depending on the extracellular range of that type of signal) a vector equation gives the J-factor for that signal ligand the value of 1 if it is in range of the signal, else 0; and this value is then used in the vector equation determining the expression of the target gene(s). For example if Gene

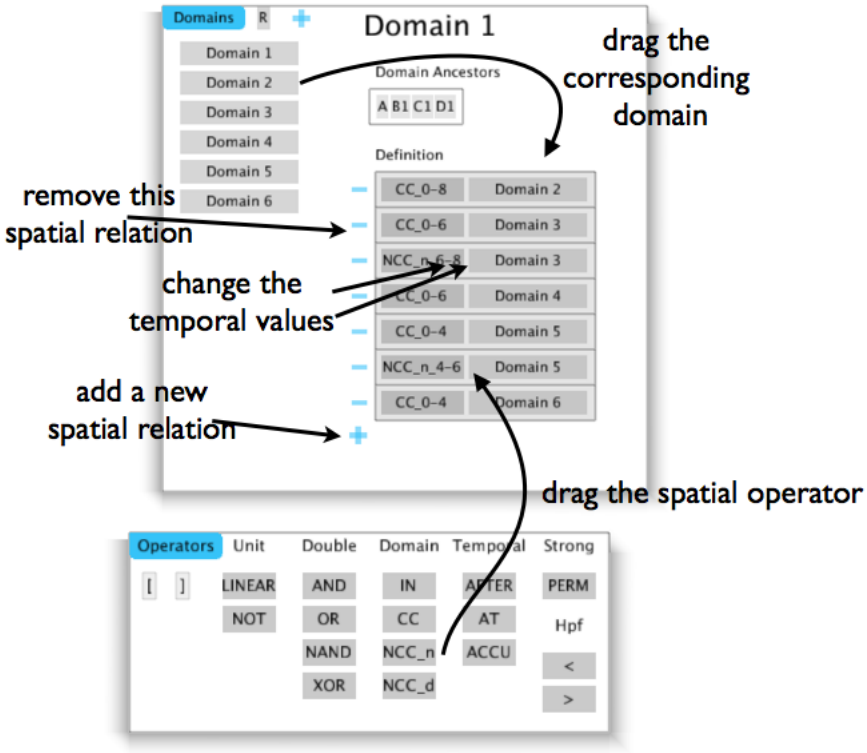


Figure 7: Relative spatial relations

X is turned on by signal Y, we might write *"if J(Y) =1 AND AT-3 A=1 then X=1, else 0.* Note that this type of algorithm would also lend itself easily to representation of different transcriptional effects from given signals at different positions with respect to the signal ligand producing domain.

Specific Instructions

5.1 Entering the state of the J-factor conditional on signal reception in a vector equation

5.1.1 Create the J-factor (fig. 8)

- As in 2.1.1, click on '+' in the *Genes* menu and enter the name of your J-factor. Specify that it is not a gene by clicking on the first box (see Figure 1)
- Activate your Gene X with a click on its name in the list of Genes.
- As in 2.2.2 create the Vector Equation for Gene X, and drag the J-factor from the list

5.2 Defining the range of a given signal (fig. 9)

- In the *Genes* menu, activate the J-factor
- Add a Vector Equation (see 2.2.2)
- Drag the signal, spatial operators and ***R*** to the generic domain in the Vector Equation box.

R is a generic domain (placed upper right the list of domains). It is a surrogate symbol for the domain where the vector equation is computed. For example, suppose the gene B is defined by the vector equation *"if [A=1 in R], then 1 else 0"*. When the state of B is computed in different domains, if, in one of these domains A is ON, B will also be ON.

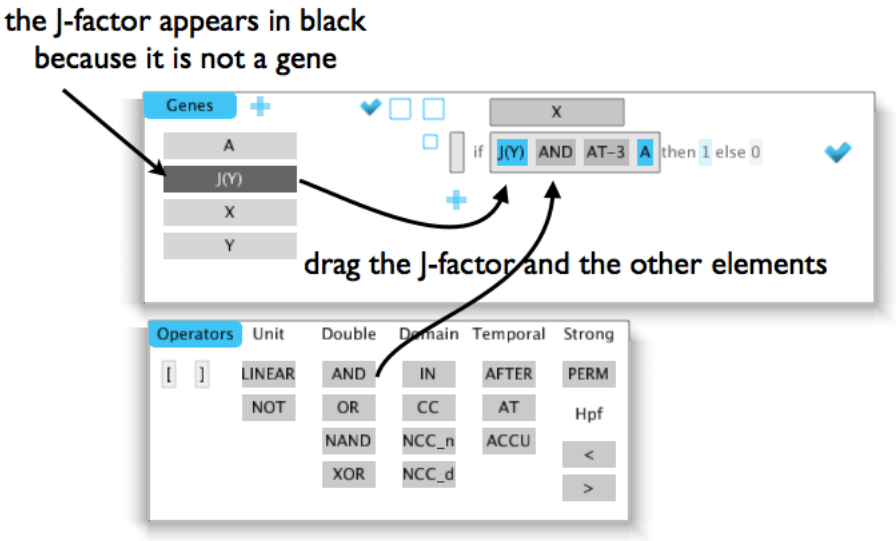


Figure 8: Using the J-Factor

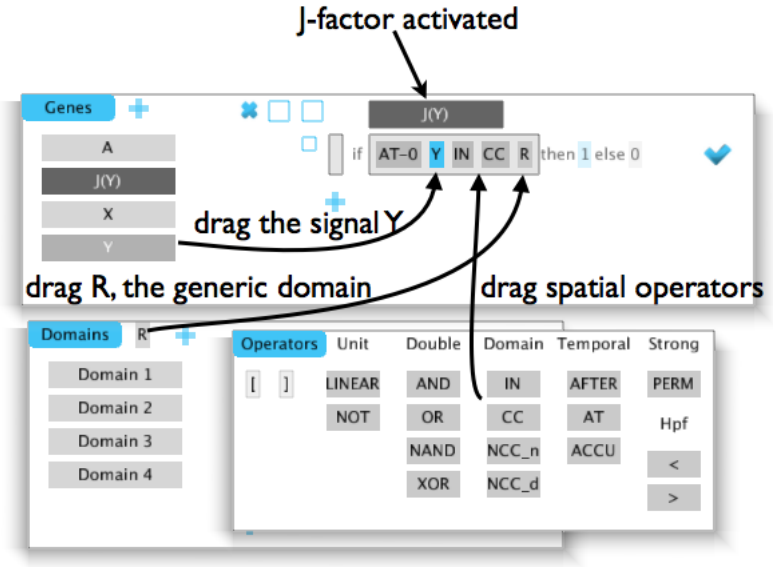


Figure 9: Define the J-Factor

6 Initial Conditions and Other Manually Set Inputs

The operation of the model begins with a set of initial conditions set manually by the operator. For example, if the model begins with zygotic activity in an early embryo, the initial conditions would include any maternal inputs localized in the egg that are of regulatory significance in that they endow given early blastomeres with distinct regulatory properties. If it were to pertain to development of a body part or organ the initial conditions might include the initial regulatory state of the progenitor field and any axially oriented signal inputs. In addition to initial conditions there arise occasions when it is desirable to set other inputs going at given times and places. For example, it might be known that such inputs exist but not what causes them to be activated or repressed when and where they are observed to be, requiring manual intervention. Or inputs might be imposed as an in silico experiment to determine the effects on the rest of the system.

Specific Instructions

6.1 Setting initial conditions in GeNeTool (fig. 10, video 8)

- In the *Genes* menu , create a new element (click on '+' , see 2.1.1 and Figure 1)
- Add a Vector Equation (see 2.2.2)
- **Time specifcation** : Drag the elements '>' and '<' from the *Operators* menu to specify the time boundaries of this initial condition. When you release these elements in the Vector Equation box, they appear with a number. As usual, put your mouse over this number and use your up and down keyboard arrows to change its value. *Remark*: '>' and '<' mean greater than and less than. Example: *">2 and <5"* is equivalent to *"is ON at 3 and 4"*
- **Space specification** : Drag the AND operator and the spatial condition (IN a *specific Domain* ..) where the initial input will be active.

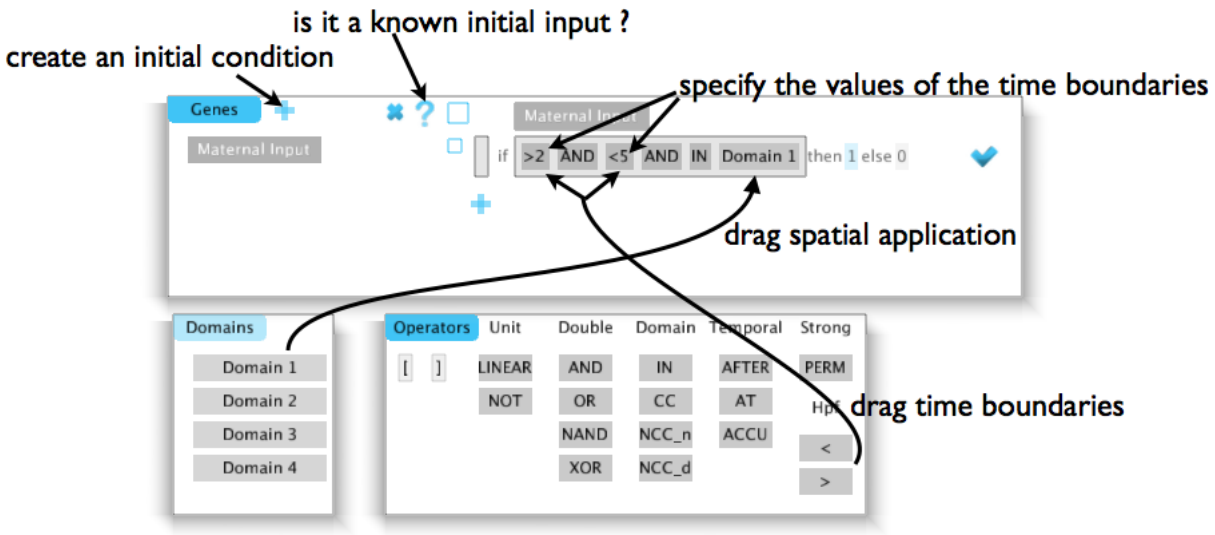


Figure 10: Initial Conditions

6.2 Other manually imposed inputs

- As in 6.1, you can create as many manually imposed inputs as you wish
- If this manually set input is unknown, click on the second box left to the name. (Figure 2.1.1 and section 6.1)

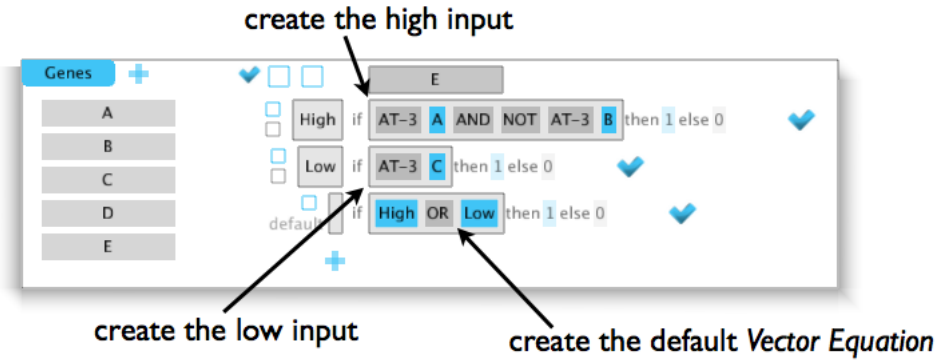
7 Responses Dependent on Input Levels

Occasionally given genes respond in a opposite fashion to diverse levels of the same input. For example it is not uncommon to encounter genes activated by low levels of an input but repressed by high levels. For such special cases of dependence on input levels the response to high input vs. low input can be coded in the vector equation. For the upstream gene producing the high or low inputs into its downstream target genes, the problem is to define the conditions producing these different levels of product. For example if these differences in output were due to the operation of separate *cis*-regulatory modules, only the output of the appropriate module would be used to compute the expression state of the responding gene.

Specific Instructions

7.1 Encoding opposite responses to high vs. low levels of a given input (fig. 11)

- In the *Genes* menu , create a new element (click on '+' , see 2.1.1 and Figure 1)
- Add a Vector Equation (see 2.2.2) for the high level, name it (see 2.3.1), and drag its corresponding elements.
- Add another Vector Equation for the low level, name it, and drag its corresponding elements.
- Add a default Vector Equation, and as in 2.3.2, drag the high and low level module names into the box.
- **Remark:** to specify an input level (or a module) in the vector equation of on an other gene, double click on the gene input.



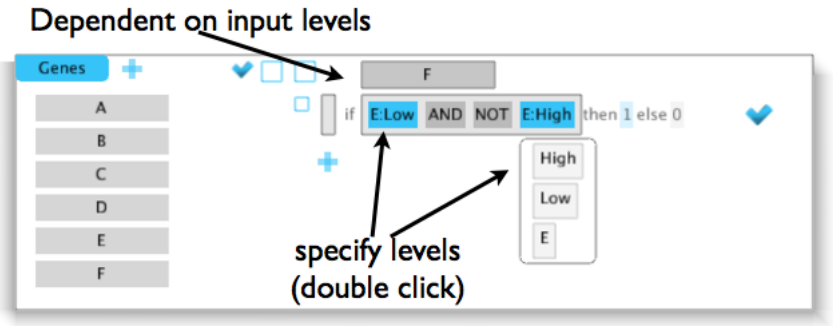


Figure 11: Responses dependent on input levels, where gene E produces the inputs to gene F, and gene F is activated by low levels of products of gene E and repressed by high levels.

8 Computational Functions of GeNeTool

The basic output of **GeNeTool** is a matrix of computed spatial gene expression results at successive time intervals. But this output can be used for many different purposes. **GeNeTool** is designed to generate a specific comparison between observed spatial expression patterns over time and the computed patterns, in such a way as to indicate explicitly both temporal and/or spatial disagreements and perfect convergence. The program can also provide explicit comparisons between two different computational runs. This is particularly useful for analysis of the results of in silico perturbations, by comparison with the control or unperturbed system. A very large range of perturbations is available for testing the significance of any aspect of the system. Vector equations can be changed, mimicking site-specific mutations or evolutionary alterations; inputs can be cancelled, mimicking morpholino or iRNA treatment; inputs can be inserted, mimicking ectopic over expression experiments; spatial relations can be changed by altering *CC*/*NCC* statements; temporal relations can be changed by altering *AT* statements.

Specific Instructions

8.1 Running a GeNeTool computation (fig. 12, video 9)

- **Requirement:** You need to have at least one Gene and one Domain to use the *Model* menu window
- *Remark:* If you have several domains, each row in each domain is computed in parallel (to enable interactions between domains in real time). But only one domain can be visualize at a time. To change the active domain on the model, click on its name in the domain list in the *Domains* menu (as in 4.1.2 when you change the active domain in the *Domains* menu)
- Open the *Model* menu.
    * The model initializes with all the genes created in columns, and a first empty row as the first computed step time.
    * Under the menu name there are 4 buttons:
        . '+' to **add** a new computed step.
        . '-' to **delete** the last computed step
        . 'reset' to **reset** the model at the first step
        . 'reload' to **recomputed** all the steps (see section 8.3)

Manage computed step time: the first line starts at the step 0, and each time you click on '+' a new line is computed, with +1 added to the step time on the previous line. You can modify this value by putting your mouse on the step time number and using the up and down keyboard arrows. If you don't want to visualize some computed step time, just click on the little '-' on the left of the corresponding step number to hide the specific line. Click on the '+' on the left of the time unit to reveal all rows.

8.2 Comparing observed expression data with computation results

8.2.1 Entering observed expression data in format for comparison

8.2.1.a Entering observed expression data (same representation as in BioTapestry) (fig. 13, video 6)

- Once you have already provided the progenitors of each domain through time (see 4.2), and at least one gene, the menu *Expression* appears. Open it.
- Click on a gene name in the list of gene in the *Gene* menu to activate a gene (same as in 2.1.2). The expression matrix from the **BioTapestry** data file can be displayed for this active gene as follows:

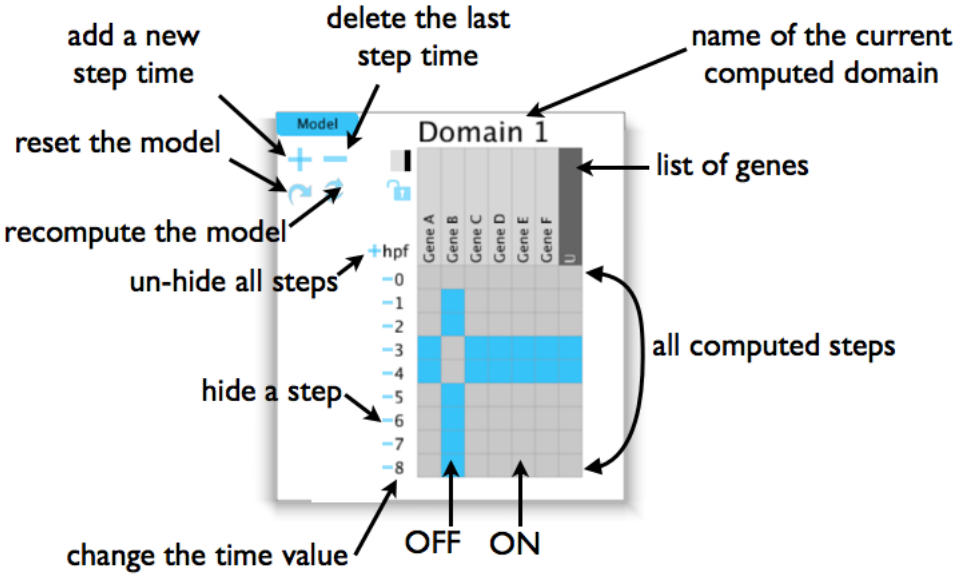


Figure 12: Running a **GeNeTool** computation

- All the boxes of this matrix are by default set as 'no data' *(white)*. To specify your observed data through time and space, click on each box to assign its value.
- 1 click to specify 'no expression' *(grey)*
- 2 clicks for 'weak expression' *(light blue)*
- 3 clicks for 'expression' *(blue)*
- 4 clicks for 'maternal expression' *(green)*
- 5 clicks back to no data.
- 6 clicks back as 1 click
- ...

The color legend is on the top of this menu.

8.2.1.b Visualize observed expression data in format for comparison to computed data (fig. 14, video 7)

- Open the *Data* menu. In the same format as the *Model* menu all the genes are represented in columns and steps time in rows.
- For an active domain, (see section 4.1 to activate a domain) , the observed matrix of all gene expression is represented in order through time. On the left of this matrix, the tree of this domain, containing all its progenitors, is shown.

8.2.2 Executing and interpreting the comparison (fig. 15, video 10)

- In the *Model* menu, click on the grey box with a black bar on the left (which transforms to a blue box with a black bar when it is activated) to visualize the comparison between the observed data and the model on the active domain.

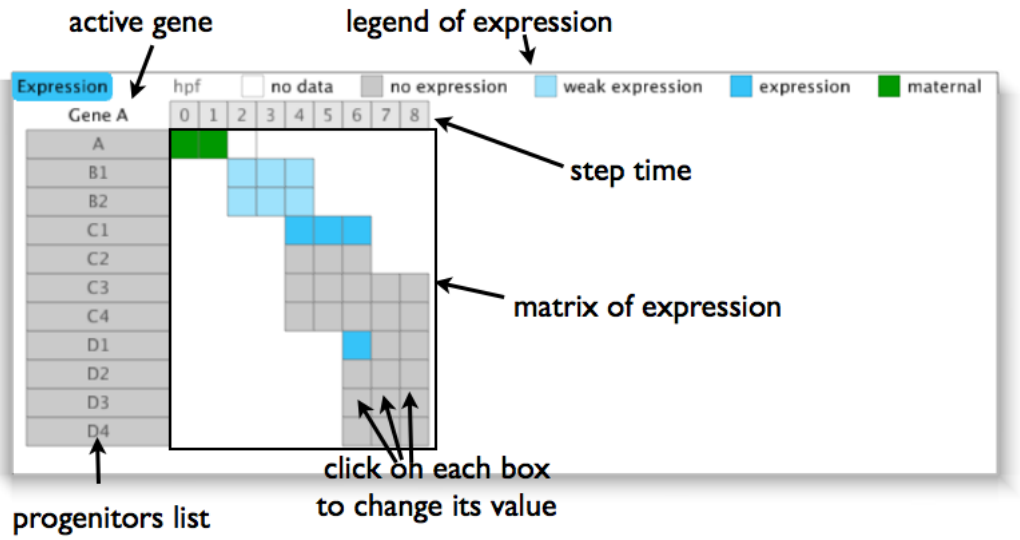


Figure 13: Entering observed expression data

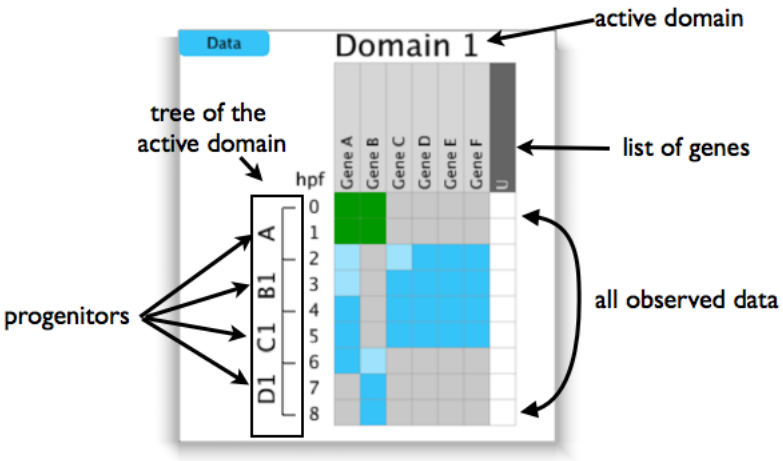


Figure 14: Visualize observed expression data

- When the observed data and the model are exactly similar, nothing is shown.
- When there is a discrepancy between the observed data and the model, a bar on the left of the box of the matrix is shown.
- Three different types of discrepancies are indicated:
  * Maternal discrepancy *(light green bar)*
  * Insignificant timing discrepancy, less than or equal to the step times *(light black bar)*
  * Significant discrepancy *(filled black bar)*
- When there are no observed data, there are no comparisons shown.

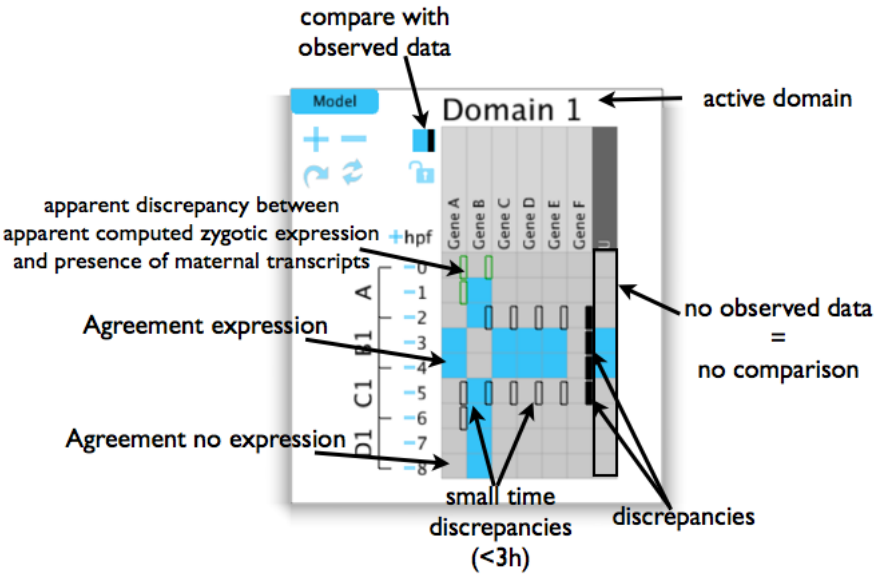


Figure 15: Executing and interpreting the comparison

8.3 Perturbations

8.3.1 Perturbations by manual alteration of vector equations

- Open the *Gene* menu, and click on a gene name to activate it.
- In its Vector Equation box, changes can be made by :
  * changing the value of the temporal operator
  * moving elements inside the vector equation box
  * dragging elements to trash.
  * adding another part of the vector equation

8.3.2 Perturbations by manual alteration of inputs (fig. 16, video 11)

- You can simply alter the value of any elements at any time by clicking on its specific time box in the model.
- If the value of the box was previously computed ON *(blue)*, and you turn it OFF, it appears grey with a white border to specify that it was manually altered. Similarly if the value of the box was previously computed OFF *(grey)*, and you turn it ON, it appears OFF blue with a white border.

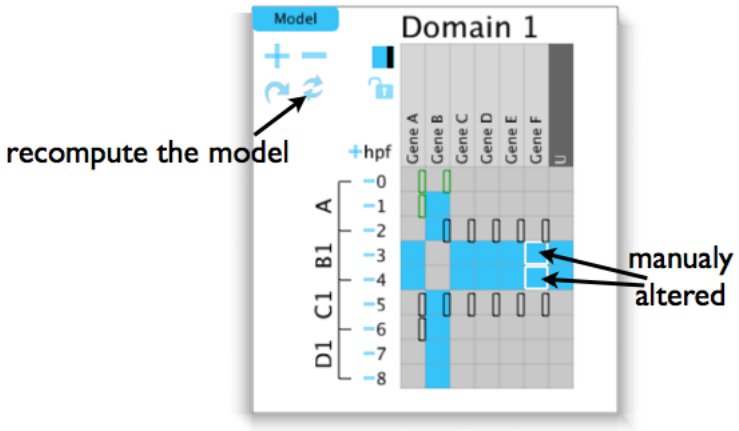


Figure 16: Perturbations by manual alteration of inputs

8.3.3 Visualizing the effect of a perturbation

Once you have modified your vector equation or altered an input, you can simply click on the 're-compute' button in the *Model* menu (see 8a) to visualize the impact of the modification of this vector equation in the model.

8.3.4 Create different versions of the model (fig. 17,video 12)

For each perturbation, you can create different versions of your model:

- Click on the lock to save the current version and avoid any possibilities of modification.
- Click on the blue double-box to create a new version of the model.
- Make your perturbations (by manually altering inputs or changing the Vector Equations)
- To switch between different registered versions, leave your mouse on the current version name , and a sub-menu will give you all the other versions available, click on any one to switch.
- A click on the blue cross left of the version's name will delete the current version.
- When you save the model (see section 8.3.4), all versions are saved.

8.4 Comparisons (video 13)

The menu *Compare* offers the possibilities of several types of visualization and comparison.
Main characteristics :

- **Specify the time boundaries of the comparison**: place your mouse on the number and use your up-down keyboard arrows to modify the values.
- **Visualization format of expression patterns**:
  * *ON/OFF*, on when there was an expression at least at some point in a given time frame
  * *heat-map* format representing exactly the percentage of expression during the time frame.

  → click on the first rows of the visualized matrix to adjust this format
- **To visualize the comparison in different format :** (fig. 18)

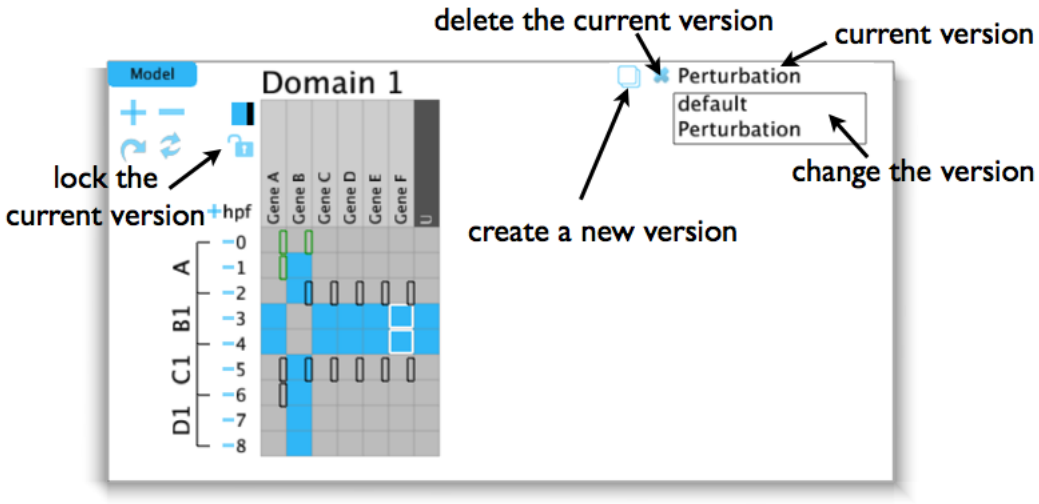


Figure 17: Create different versions

* *ON/OFF*, on when at least once, both model / data (or version/version) were similar
* *heat-map* format compute the exact percentage of similitude between model/data (or version/version).

→ click on the second row of the visualized matrix to adjust this format

- **Different kinds of comparison :** **(fig. 19)**

* between different versions
* between a specific version with observed data
* between domains

→ click on each element before and after the '/' to visualize the different possibilities of comparison

- **Matrix of comparisons :** **(fig. 20)**

* move your mouse over the column name to visualize all types of possible comparison in column and click on your choice.
* do the same for the row.

→ click on each '+' to visualize all columns and lines, and on a specific '-' to hide it

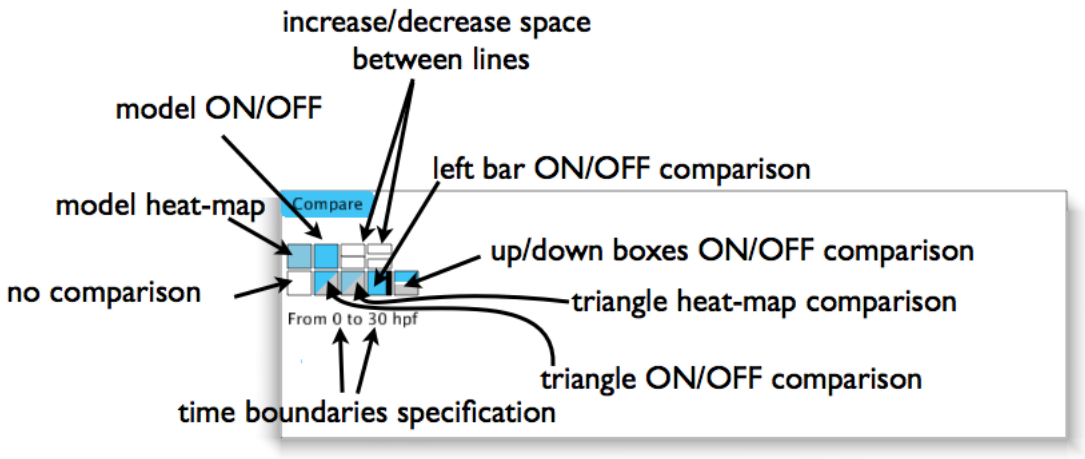


Figure 18: *Comparison* menu

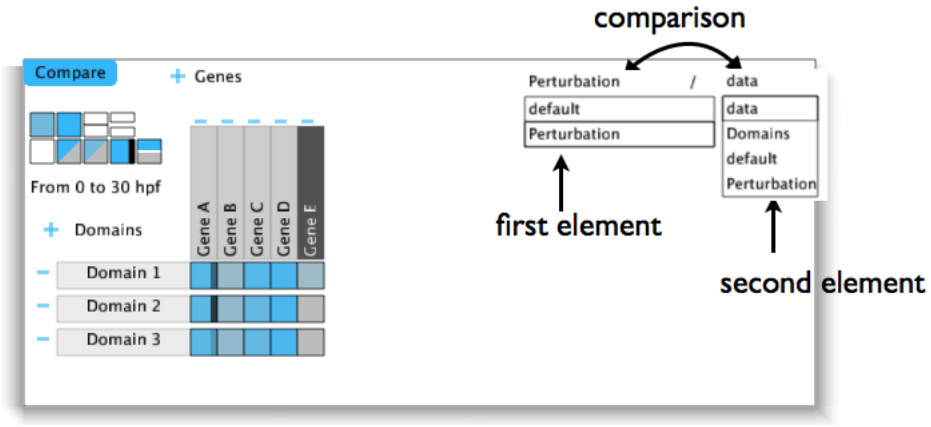


Figure 19: Comparison Possibilities

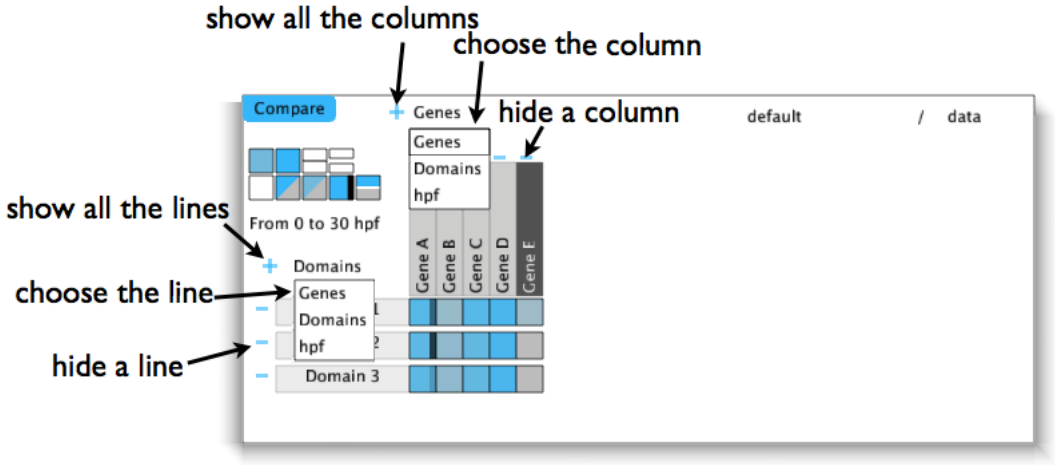


Figure 20: Matrix of Comparisons