

Modeling Hypocotyl Elongation in *Arabidopsis*

Matthew Schultz

Introduction

When a plant is first beginning to develop, it makes decisions based on environmental conditions to determine how best to grow. These decisions help the plant adapt to surrounding conditions and are necessary due to plants' sedentary nature. The high sensitivity to internal and external growth regulators, the availability of mutants, and the simplicity of the hypocotyl, the portion of a seedling's stem which is below the cotyledons or embryonic leaves, make this portion of the plant an excellent model for studying the interaction of signals that influence a plant's growth (Vandenbussche, Verbelen et al. 2005). By using a computational model, we sought to better understand the relationship between light, an external regulator, the circadian clock, an internal regulator, and hypocotyl growth.

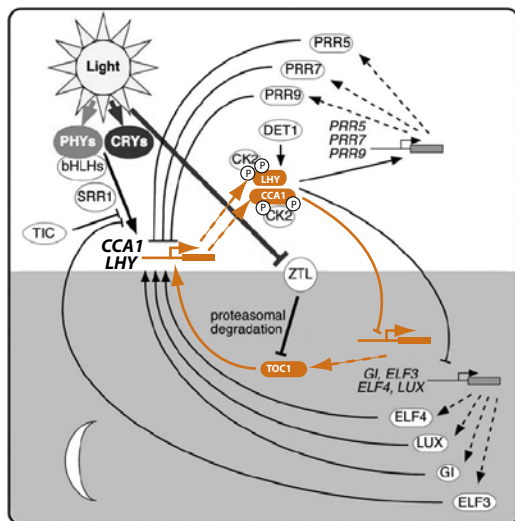


Figure 1. Shown here is a schematic of plant circadian clock components as well as the time of day they are active. Taken with permission from (McClung 2006).

One very important factor that regulates hypocotyl growth is the circadian clock. The circadian clock is an important biological pathway and is observed in a wide variety of organisms (McClung 2006). Both plants and animals use this internal clock to regulate their daily activities. Two sets of components in the clock, the first light-dependent and the second light-independent, allow an organism to maintain its daily rhythms without light input as well as to change the timing of those rhythms when light input varies (McClung 2006).

The current model for the circadian clock is a relatively complex web of interacting components, but at its core are two interlocking negative feedback loops. The basic components of this loop are shown in brown in the figure below. As shown in Figure 1, there are three major players in this loop. Timing of CAB Expression 1 (TOC1), Late Elongated Hypocotyl (LHY), and Circadian Clock Associated 1 (CCA1) all help regulate both of the loops shown below.

Mutants that overexpress the CCA1 protein have been shown to disrupt several circadian rhythms and plant developmental processes, including hypocotyl elongation, and are considered arrhythmic (Wang and Tobin 1998). Additionally, CCA1 expression can be entrained by exposing plants to different time courses of light and dark cycles (Wang and Tobin 1998). These features of CCA1 show it to be important to the circadian clock and the pathways it regulates as well as capable of producing rhythmic output similar to the circadian clock's output. We chose to include only CCA1 because we felt it would appropriately represent the cyclic interactions input by the clock into the hypocotyl development pathway. Simplicity can be a virtue when designing a computational model, and for that reason we excluded the other known clock components.

Hypocotyl growth is also strongly affected by light input. Plants require red and/or blue wavelength light in order to carry out photosynthesis and have consequently developed photoreceptors that can sense whether sufficient light exists in the environment to survive. If a developing plant does not detect correct light conditions for survival it will become etiolated, a condition where the plant's hypocotyl elongates and chloroplast development stops, among other processes, in an effort to find a suitable light source. Once the plant finds the light it needs to perform photosynthesis, it de-etiolates and resumes normal functions. Thus, light inhibits hypocotyl elongation. *Arabidopsis* possesses many light sensing proteins that when exposed to light can create signaling cascades that trigger different effects, including de-etiolation. These photoreceptor proteins fall into three families: phytochromes, cryptochromes, and phototropins.

For our experiment we used two different mutant lines that are defective in their capacity to signal through these photoreceptor pathways. The first was a *hy2* mutant, which is deficient in the ability to produce phytochromobilin, the light-detecting chromophore component of phytochrome. This mutation results in a decreased ability to sense light (Kohchi, Mukougawa et al. 2001). The second mutant line was *hy5*. HY5 is a transcription factor that when present promotes the transcription of certain light induced genes (Chen, Chory et al. 2004). It is degraded by another protein, COP1, which is active in the absence of light. Therefore, *hy5* mutants are unable to correctly express certain light regulated genes and are defective in light inhibition of hypocotyls (Chen, Chory et al. 2004). Although we used data sets from *hy2* and *hy5* mutant lines, we included in our model a light component that directly influenced PIF5 rather than the complex signaling pathway that exists in plants, again for simplicity's sake.

The final component in hypocotyl development that our model considered was the protein PIF5. PIF4 and PIF5 are proteins that appear to function semi-redundantly in regulating hypocotyl growth (Nozue, Covington et al. 2007). It has been shown that hypocotyl growth in *Arabidopsis* is controlled by both the circadian clock and light signals from the environment, and that these inputs mediate the quantities of PIF4 and/or PIF5 proteins. Depending on the time of day, the circadian clock can activate or repress the transcription of PIF4 and PIF5 mRNA. On the other hand, light regulates PIF4 and PIF5 activity through activation of phytochrome, which causes the degradation of PIF4 and PIF5 proteins (Nozue, Covington et al. 2007). Due to the semi-redundant nature of PIF4 and PIF5, we only included PIF5 in our model. Additionally, we simplified the relationship between growth rate and PIF5 concentration by assuming that PIF5 concentration was equivalent to growth rate.

In any analysis of a biological mechanism, an obvious question is how to go about organizing, adding, and modifying elements in a hypothetical pathway. A computational model presents an effective way to do all these things. The development of numerous software packages (many at low cost) that allow researchers to easily visualize and simulate their models, along with the standardized system of storing models known as the System Biology Markup Language (SBML), make a computational model a viable and useful tool.

Computational models are very helpful because they force a researcher to thoroughly analyze what

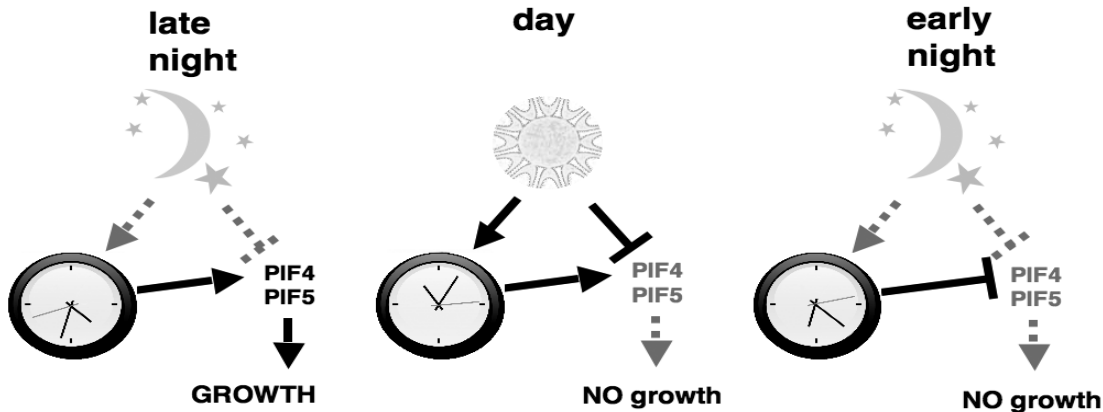


Figure 2. Shown here are the factors believed to affect PIF4 and PIF5. During the day the circadian clock activates the transcription of PIF4 and PIF5 mRNA, while light competes to inhibit it by degrading PIF4 and PIF5 protein. During the night PIF4 and PIF5 mRNA expression is inhibited early and activated later on.

he or she already knows as well as provide a benchmark for how well a proposed pathway approximates experimental results. The act of entering information into a model compels the researcher to examine current data in an organized fashion. This examination can help shed light on components the researcher has yet to discover or reveal that a current component needs to be modified. If a model reproduces experimental data over a wide variety of environmental or genetic perturbations, odds are that the current model represents something close to reality. If it fails to do one or both of these, the experimental conditions under which the model fails provide a starting point for improvements. This iterative method has been used successfully to confirm hypothesized models (see, e.g., Locke, Southern et al. 2005). Although modeling techniques and technologies are not yet advanced enough to replace experiments, hopefully one day they will be. Both time and money would be saved if all one had to do was input the parameters of an experiment into a computer and wait for the output. Given these potential benefits, using a model to help analyze how light, the circadian clock, and PIF5 control hypocotyl growth in *Arabidopsis* seemed a logical choice.

Speaking in these general terms, however, does not really give a sense of the process of computational modeling. The basic steps for using a model to analyze a system are first, creating a visual representation of the pathway; second, obtaining rate constants for the various reactions in the pathway using parameter estimation; and finally, comparing simulated results with experimentally obtained ones.

The final step in analyzing a pathway with a computational model is to run different sets of experimental inputs through the model and compare those results with the experimental results. For example, if the model has been constructed using experimental data from a wild type organism, data on mutants should also be run through the model to see if it still predicts the correct results. As mentioned before, if the model does not produce the correct results, one has to start over from step one and add, modify, or remove components in an attempt to correct the model.

Designing and Using a Model

Creating the visual “map” of a particular pathway involves going through all the information a researcher has about the nature of the pathway being studied. How are all the components connected?

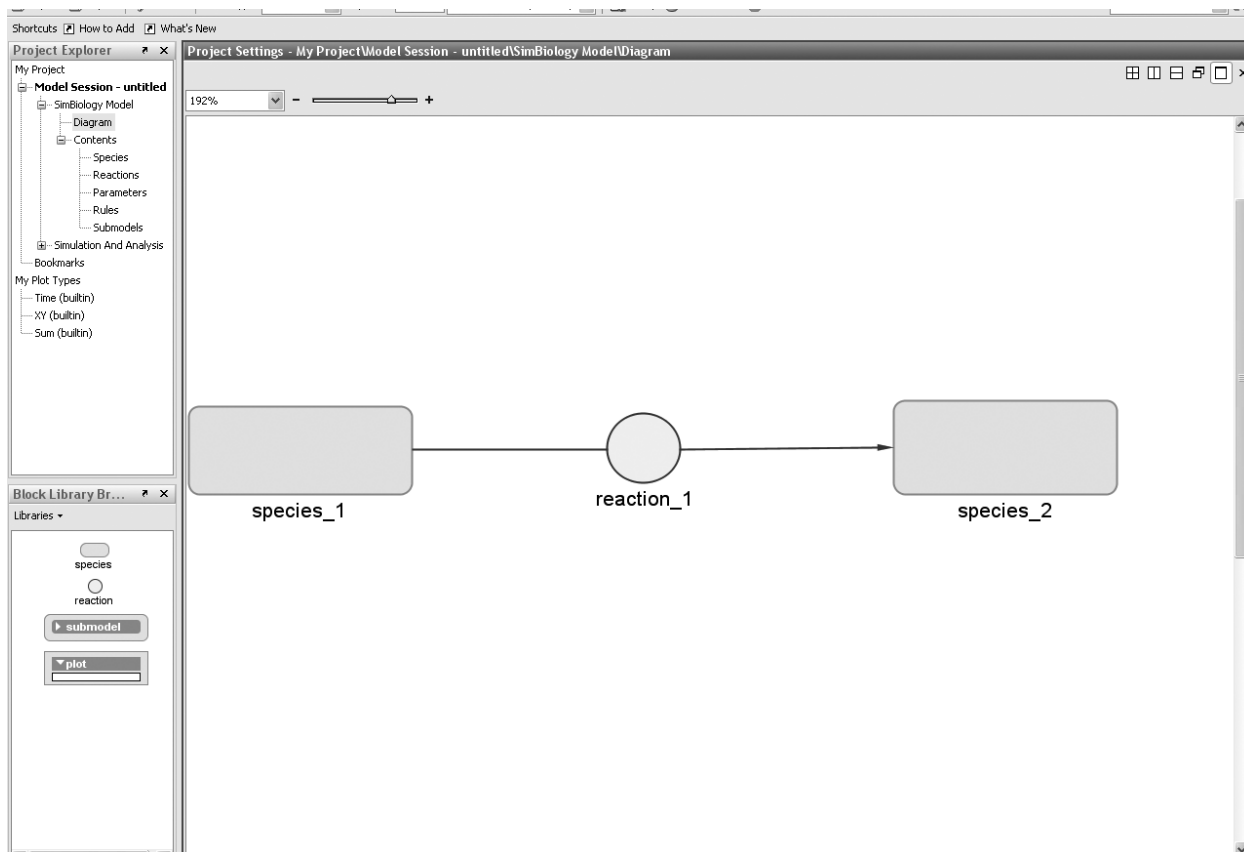


Figure 3. This figure depicts the Simbiology model space after a simple pathway has been entered.

What is the nature of the relationship between each connected pair of elements? Is the inclusion of this component vital or will it merely complicate the question being posed? All of these are valid concerns that must be addressed before proceeding further.

The next choice a researcher must make as a model designer is which software package to use. Although there are many freely available online, support and documentation for such products is not always reliable. For this project we chose to use an application built into MATLAB called Simbiology. Although not as versatile in terms of features as some other modeling packages, Simbiology proved more stable. Additionally the functions it lacks are being released in updates every several months.

Once a modeling program is chosen, inputting a model is usually a simple procedure. In Simbiology, there are three major elements that one can add to the model space: species, reaction circles, and reaction arrows. Shown below is an example of a very simple model:

To obtain the above image, all the user needs to do is click and drag the desired number of species into the model space, place reaction circles in between interacting species, and finally hold CTRL, click, and drag reaction arrows to connect species and reaction circles. Once this is done, each individual reaction circle needs to be defined. By double clicking on a reaction circle, the user can modify attributes such as the kinetic law, parameters, and the name of the reaction, and can even include a description for people who later use the model. A similar procedure can be followed to change attributes of the species blocks as well.

Once the reaction and species attributes are defined, the last two inputs are events and rules. An event, as the name implies, occurs when a given set of conditions, or triggers, are met. These triggers could be based on time, the concentration of another species, or a combination of the two. Once the conditions are met, the event occurs and alters the model accordingly. For our model, this feature would have been very useful for varying light input over time. We could simply have set triggers for time points when the lights would come on and turn off. Unfortunately, the version of Simbiology we used did not include this feature (a newer version does). We managed to overcome this deficiency through the use of another MATLAB function.

Rules, the second input, are like events in that they alter parameters or species amounts during the course of a simulation; however, rules do not respond to triggers like events do. Rules come in three types: algebraic, assignment, and rate. Each type tells Simbiology how to interpret a given rule. Assignment or algebraic rules simply set the value of a species or parameter based on the given equation, whereas a rate rule will increase or decrease the quantity of a given species based on an input equation. While the rate rule may sound superfluous in light of the rate equations that determine how one species changes into another, it allows the user to account for effects like diffusion that the other rate equations can not.

Table 1. Reactions and respective rate equations. Terms such as CCA1_Translation are rate constants or parameters that were estimated.

Reaction	Rate Equation
CCA1Gene -> CCA1RNA	$\cos(\text{time} * 2 * 3.14159 / 24) + 1$
CCA1RNA -> CCA1	CCA1_Translation * CCA1RNA
CCA1 -> Degraded CCA1	PIF5_Transcription * PIF5Gene
PIF5Gene + CCA1 -> PIF5RNA	PIF5_Translation * CCA1 * PIF5RNA
PIF5RNA -> PIF5	PIF5_Degradation * Light * PIF5
PIF5 + Light -> Degraded PIF5	CCA1_Degradation * CCA1

The basic network of our model with its three main components—PIF5, CCA1, and light—is shown below:

This image illustrates why we chose to take so many steps to simplify the model. Although the pathway does not look particularly complex, it is easy to imagine how including all players in the pathway would have resulted in a very cluttered model space. Our thinking is that we can add additional components one by one as their relevance to the model's accuracy is made clear. Building the model from the ground up like this made it far easier to troubleshoot and analyze data than it would have been otherwise.

Obtaining Data

In most cases, experimental data about the quantity of at least one species over time is needed to produce an accurate model. This data is required to estimate the rate parameters in each reaction. The alternative would be to conduct experiments that attempt to directly measure these rate parameters. By utilizing this alternative, one does not have to rely on an estimation program, which usually provides sloppy sets of parameters. *Sloppy* here means that the parameters could vary quite a bit from actual values and still produce models with predictive power. However, it has been shown that any endeavor to obtain parameters directly must be extremely accurate if it is to produce a model that has reliable predictive power (Gutenkunst, Waterfall et al. 2007). Since a model is only as good as its predictive power, we decided to follow the approach using parameter estimation.

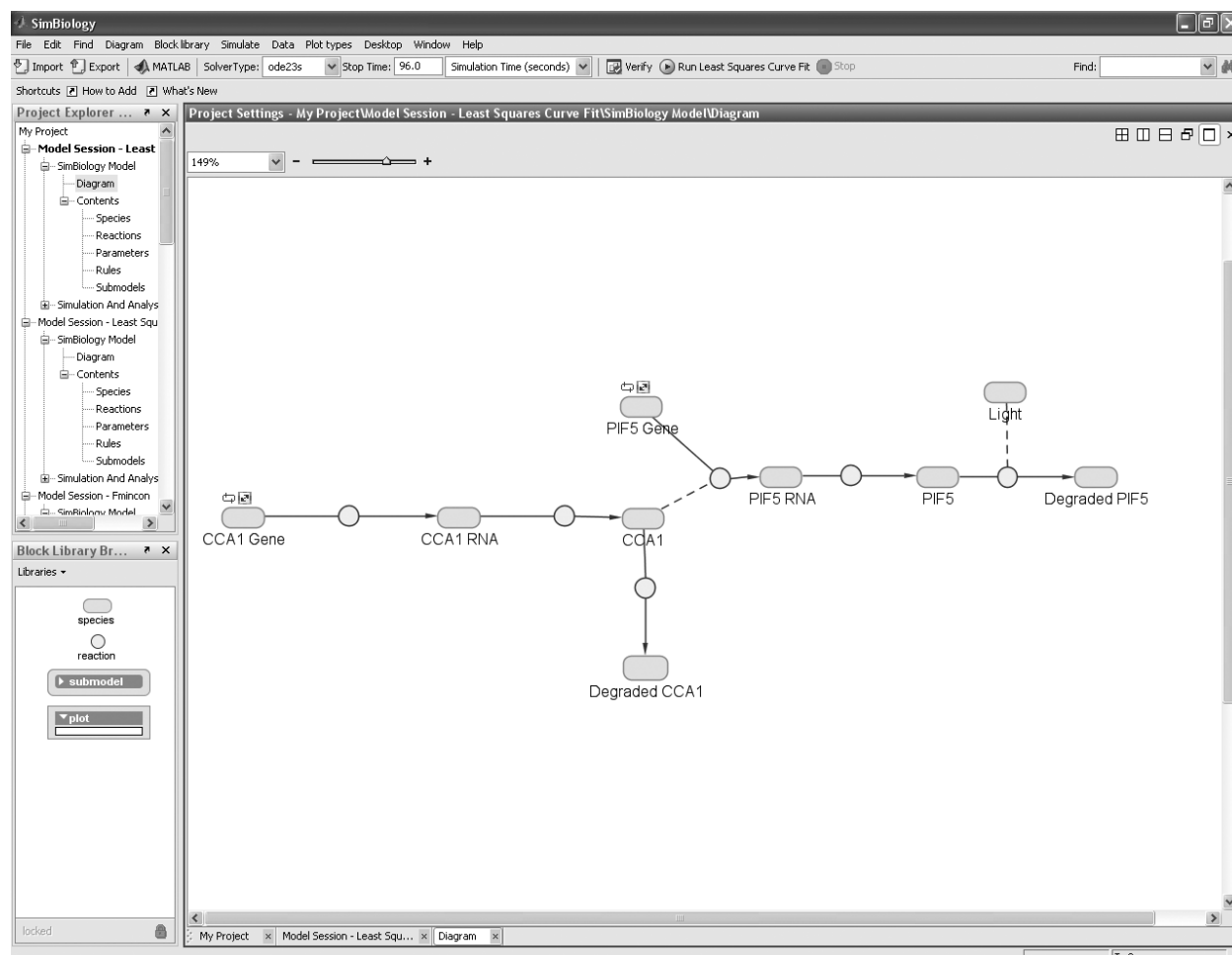


Figure 4. Current model.

The data used to produce the parameters in our model were obtained from a previous experiment performed by Dr. Kazunari Nozue. Plants of varying mutant lines were grown under a camera under regular day and night conditions (8hrs light/16 hrs dark). Specifically the mutant lines used in this model were PIF5 and CCA1 overexpressors, *hy2* and *hy5* knockouts, and two *pif5* knockouts. Pictures of the plants taken at twenty minute intervals were used to determine their growth rate. This growth data was fed directly into the model as PIF5 concentration. Again, although in any strict sense this data does not accurately represent what is actually occurring in plants, it seemed an appropriate course of action in our efforts to simplify the model.

Estimating Model Parameters

Once the relationships of the components have been mapped, one must quantify those relationships through some sort of rate equation. The choice of rate equation is based on a general understanding of how different species interact with one another. Understanding the reaction kinetics can help in choosing which of the many available general formulas to use--among which are Michaelis-Menten for enzymes that become saturated, mass-action for enzymes that don't become saturated, and Hill

Cooperative Kinetics for reactions with multiple factors that work together. We chose to use mass-action kinetics for our model because it provided a rate equation with the fewest number of parameters.

Knowing the general form of these rate equations allows a researcher to begin solving for the constants or parameters in them. A parameter estimation program produces its results based on the model, a range of possible values for all the parameters, and some experimental data. By using different sets of guessed parameters for each of the reactions in the model, and then comparing the output of these guesses with experimental data, these programs can output optimal parameters for the model. *Optimal* here means that out of all the possible values the parameters could have produced, these particular values match the experimental data the best. There are many different ways to do parameter estimation. However, trying every possible combination is not a recommended course of action. Even with the powerful computers that exist today, a simple model can take days to produce estimates, depending on the possible values of the parameters. Numerous mathematical options help limit the number of possible solutions, so that a manageable number of possible solutions can be tested.

For our model we used a program called the System Biology Markup Language Parameter Estimation Tool (SBML-PET), produced by Zhike Zi at the Max Planck Institute for Molecular Genetics (Zi and Klipp 2006). This program is freely available online and utilizes an evolutionary algorithm to obtain parameters. An evolutionary algorithm emulates certain processes from evolution in order to select for progressively “fitter” or more optimal parameters (Tsai and Wang 2005). The user supplies experimental data regarding the species quantities over time as well as the bounds on the parameters to be estimated. Additionally, the user can provide information about multiple experiments under varying conditions, and constraints on different parameters that go beyond simple bounds. When the program is run, it begins by creating a population of parameter sets, which consists of several random sets that are within the bounds and constraints specified by the user. Next, it simulates the given model with each of the parameter sets it has generated and produces data about the species concentrations over time. It then compares how well each of the data produced by each of the parameter sets matches the data provided to the program. The parameter sets that do not yield good approximations of the given data are removed from the population. Those that do “survive” are bred and mutated and their offspring are then tested against the model to see if they “survive.” This generational process is repeated many times until an optimal solution is reached. The program can give an estimate of how optimal a solution it has obtained up to that point by using an objective or cost function, which allows the user to determine when to stop the program. It is important to note that, although this method is much faster than many other deterministic methods for estimating parameters, it does not guarantee an optimal solution. Given the heuristic characteristic of the algorithm, we completed two sets of parameter estimations.

Although the manual for this program admits that the time it takes to reach an optimal solution depends on a number of factors, including CPU speed, it suggests letting the program run anywhere from several hours to an entire day, depending on the complexity of the model. All of our estimations were run on a personal computer with an AMD Athlon 64 X2 Dual Core 4200+ processor clocked at 2.21Ghz as well as 2 GB of DDR2 PC2 6400 RAM. For the first set of estimations, we ran SBML-PET for 10-15 minutes, and for the second set, we ran SBML-PET for 8-10 hours. In some but not all cases, the longer run of SBML-PET produced a better parameter set than the shorter runs. For each parameter set, we picked the best result to use in our simulations.

A point of interest in this experiment was the effect that different sets of data had on the estimated parameters. Although an overall parameter estimation was performed with all of the data sets, we also grouped the data sets by general phenotypic effect to see how well they would simulate a variety of conditions. Overall there were 5 parameter sets:

- Parameter Set 1: hy2, hy5 knockout data
 Parameter Set 2: CCA1 and PIF5 overexpressor data
 Parameter Set 3: *pif5* knockout data
 Parameter Set 4:
 Parameter Set 5: All of the above data sets

$$f_{obj} = \sum_n \sum_c \sum_i \sum_t^{NEC, NTC, NED, NSP} (Y_{pred}(n, c, i, t) - Y_{exp}(n, c, i, t))^2 * w_{(n, c, i)}$$

where *NEC* is the Number of the Experimental Conditions for the data.

NTC is the Number of the Time Courses for Condition *n*.

NED is the Number of Experimental Data in Time Course *c*.

NSP is the Number of Sampling Points in Time Course *c*.

$Y_{pred}(n, c, i, t)$ is the prediction data from the mathematical model.

$Y_{exp}(n, c, i, t)$ is the experimentally measured data.

$w_{(n, c, i)}$ is $1/(\text{maximum value of experimental data } i)^2$ in time course *c* at condition *n*.

$\sigma_{(n, c, i)}$ is the standard deviation (noise or measurement error) of experimental data *i* at sampling point *t* in time course *c* at condition *n*.

Figure 5. The cost function from SBML-PET. This figure was taken from Zi and Klipp 2006.

Wild type data only Each of these parameter files was constructed with the appropriate data for growth and corresponding light conditions over a two-day period. We included data on the light species of the model because the growth data were obtained under varying light conditions. In the input data, the light species was assigned a value of 1 for 'on' and 0 for 'off.' The cost function used to determine the optimality of a solution, and a table describing how the terms of the cost function varied for the different parameter sets, appear below (see Fig. 5; Table 2). It is worth noting that all of the above parameter sets were estimated with wild type data, specifically data from Columbia, in addition to the data mentioned.

Table 2.

Listed are some of the effects the different estimation settings had on the cost function. Information about different experimental conditions is separated by a comma. In the case of parameter set 1, a total of 289 sampling points were input for all three time courses.

	NEC	NTC	NED	NSP
ParameterSet1	1	3	2	289
ParameterSet2	3	1, 1, 1	2, 2, 2	289, 289, 289
ParameterSet3	3	1, 1, 1	2, 2, 2	289, 289, 289
ParameterSet4	1	1	2	289
ParameterSet5	5	3, 1, 1, 1, 1	2, 2, 2	867, 289, 289

Simulating the Model

The final step in this process is to simulate the model with the estimated parameters and to compare these results with previously obtained experimental data. For our model, once parameters were estimated using their respective data sets, they were plugged into the model in MATLAB and run through two sets of conditions. The first simulation was based on the experimental conditions under which the data for that parameter set were obtained. Each simulation was run under an 8:16 hour light/dark cycle with gene concentrations set at 2, to represent the wild-type strain, unless otherwise specified. Below is how each simulation was set up:

CCA1-OX Gene Conditions: CCA1 Gene was set to 10

PIF5-OX Gene Conditions: PIF5 Gene was set to 20

pif4pif5 Gene Conditions: PIF5 Gene was set to 0

pif5s1 Gene Conditions: PIF5 Gene was set to 1

No Light Conditions: Light was set to a constant 0

For the second set of simulations, the parameter sets under normal conditions were used.

Results from Simulations

Below are graphs of the simulated results of the different parameters sets compared with different sets of real data (see Figs. 6-20). Included before the graphs is a table of the parameters with which each parameter set was run (see Table 3). The title of each graph contains the parameter set with which it was run as well as the conditions under which it was run. The appropriate experimental data are also included in the graphs for reference.

Table 3. A summary of all the parameter estimates that were used during the simulations for each parameter set.

Parameter	Parameter	Parameter	Parameter	Parameter
CCA1 Translation	1.46E-05	3.62449	3.62449	9.67E-06
CPIF5 Transcription	970	8.85185E-06	8.85185E-06	6.11
PIF5 Translation	22.3	1467.66877	1467.66877	13.8
PIF5 Degradation	3.14	0.02843	0.02843	1.23
CCA1 Degradation	8.01	33.66612	33.66612	6.12

Parameter set 1 failed to reproduce experimental data under no light conditions, and instead predicted a linear growth of PIF5 concentration over time. However, it did manage to produce correct rhythms under normal light conditions. A similar failure was seen in parameter set 5 under no light conditions as well as parameter set 3 under pif5 single knockout conditions. In the case of parameter set 2, it did not reproduce the PIF5 overexpressor data, but did manage to produce appropriate rhythms for both the CCA1 overexpressor and normal data sets. All models failed to reproduce the data from the pif5 double knockout data set, due to the fact that this mutation would prevent the formation of PIF5, which was the only factor that promotes growth in our model. Obviously there are other growth-promoting factors in plants, and in this case, equating PIF5 concentrations to growth made it impossible for our model to predict these results. Although parameter set 3 did not correctly reproduce either of the pif5 knockout data sets, it did manage to create some correct rhythms under normal conditions. Parameter set 4 managed to produce the correct rhythms under normal conditions. Parameter set 5 created incorrect rhythms under the PIF5 overexpressor and pif5 single knockout conditions, but produced correctly timed rhythms under the CCA1 overexpressor and normal conditions.

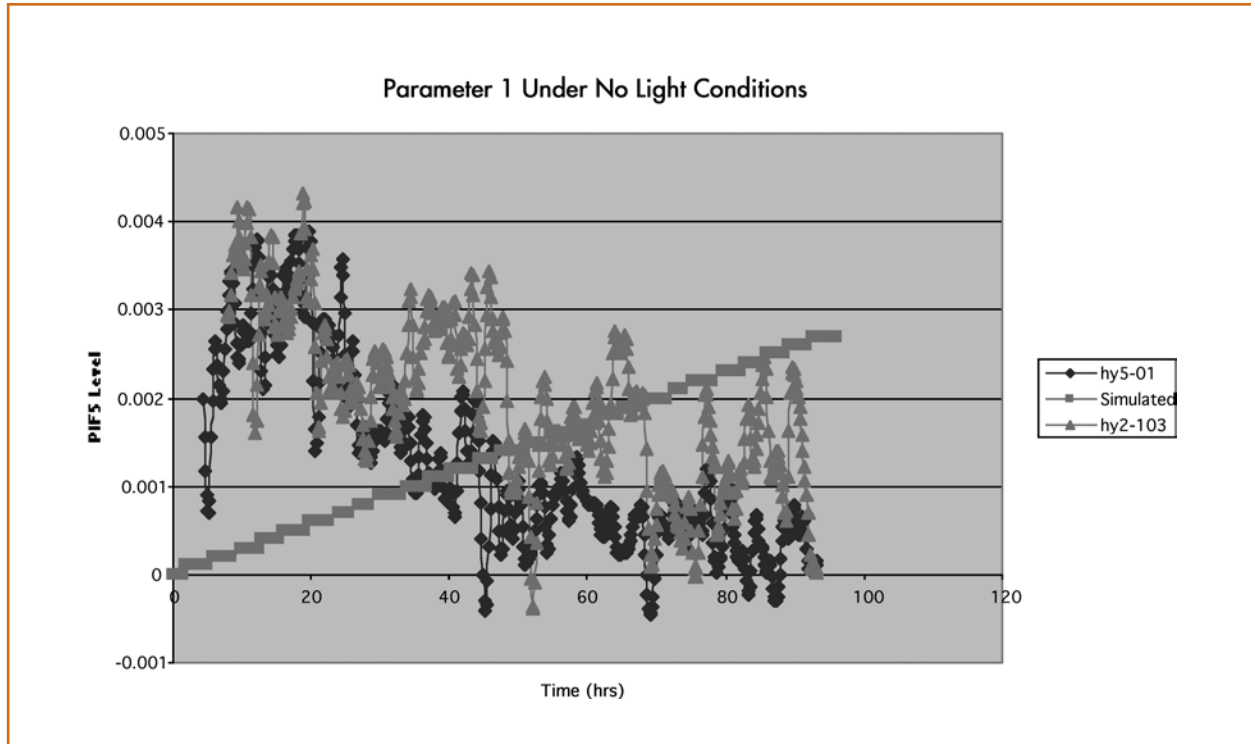


Figure 6. Parameter Set 1 simulated under dark conditions.

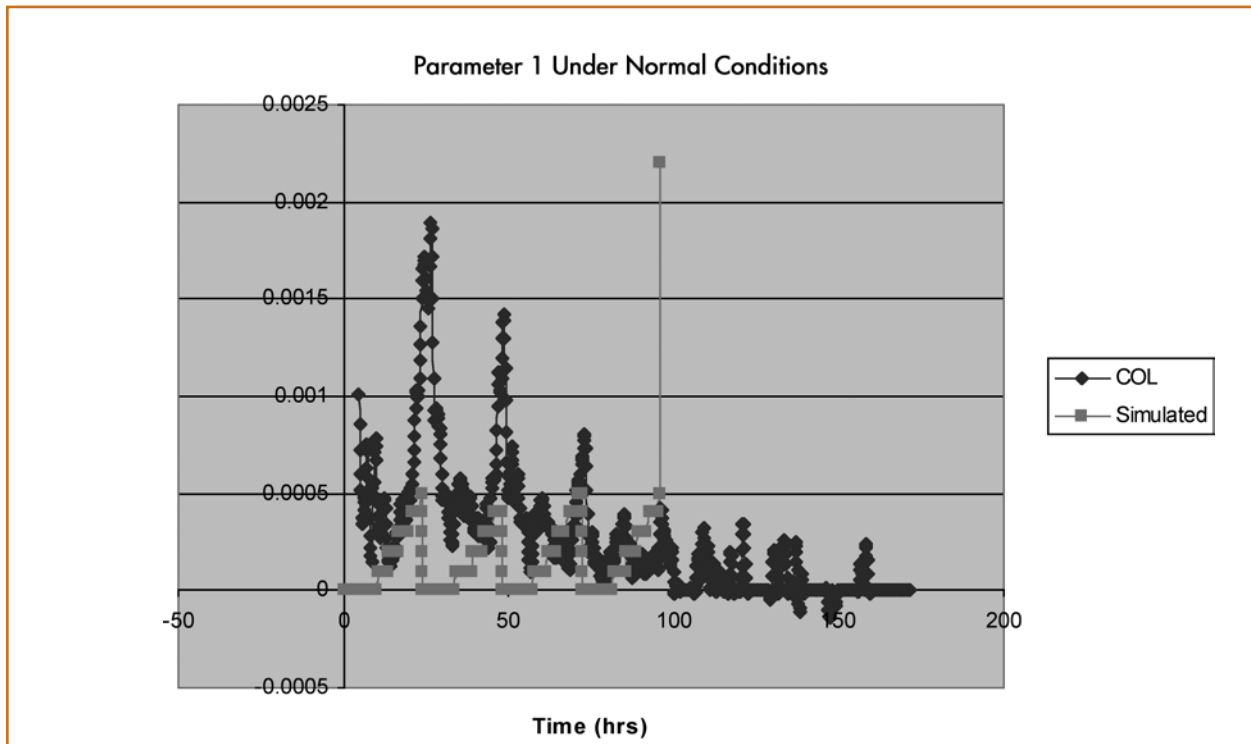


Figure 7. Parameter Set 1 simulated under normal conditions.

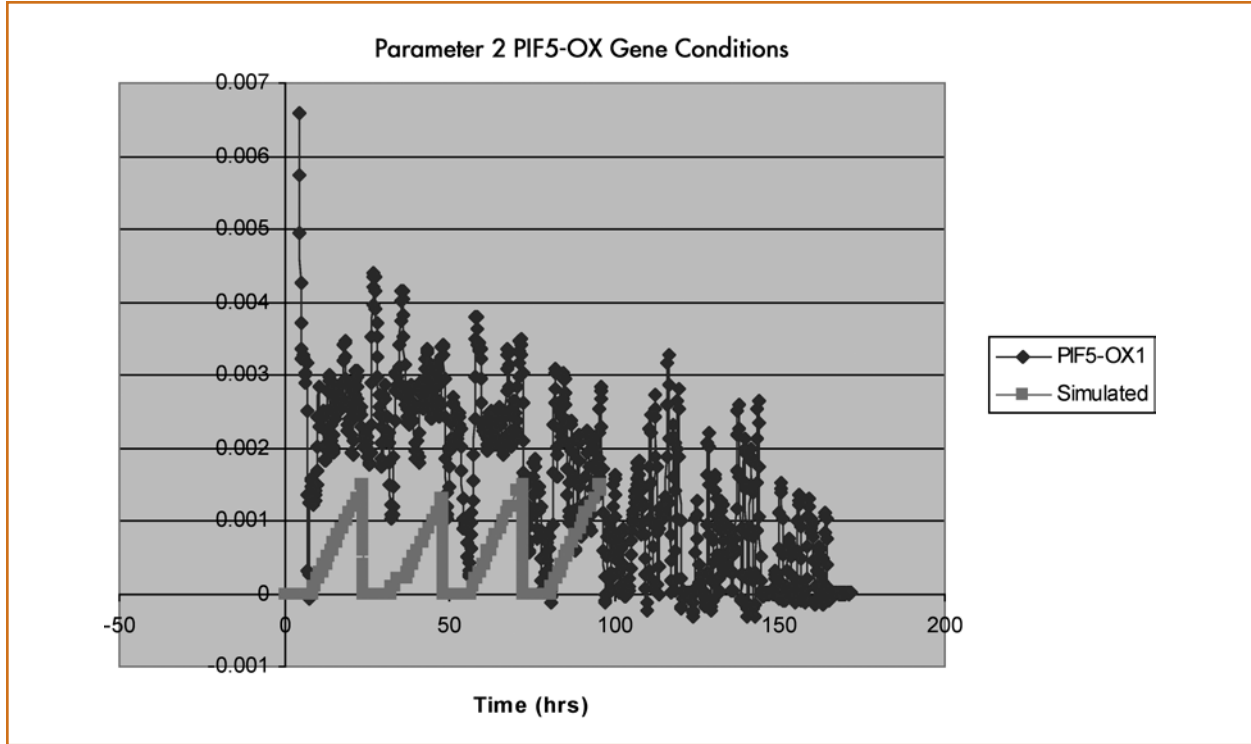


Figure 8. Parameter Set 2 simulated in a plant overexpressing PIF5.

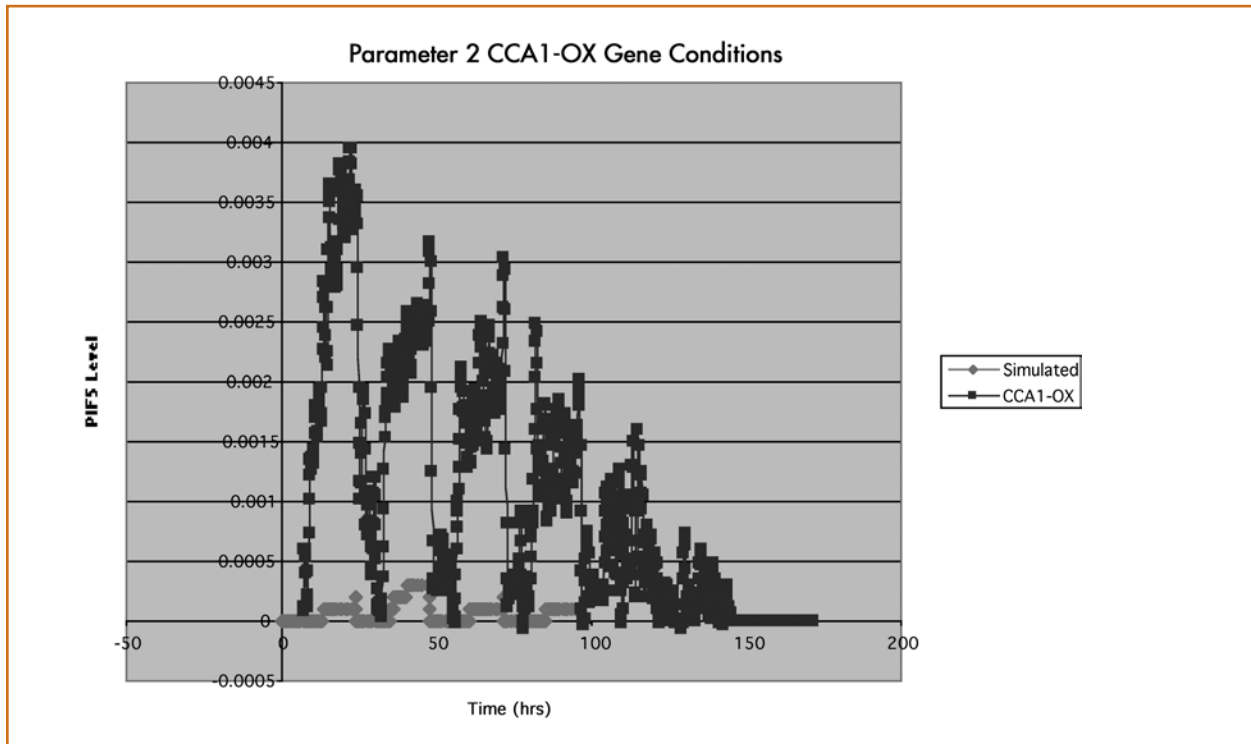


Figure 9. Parameter Set 2 simulated in a plant overexpressing CCA1.

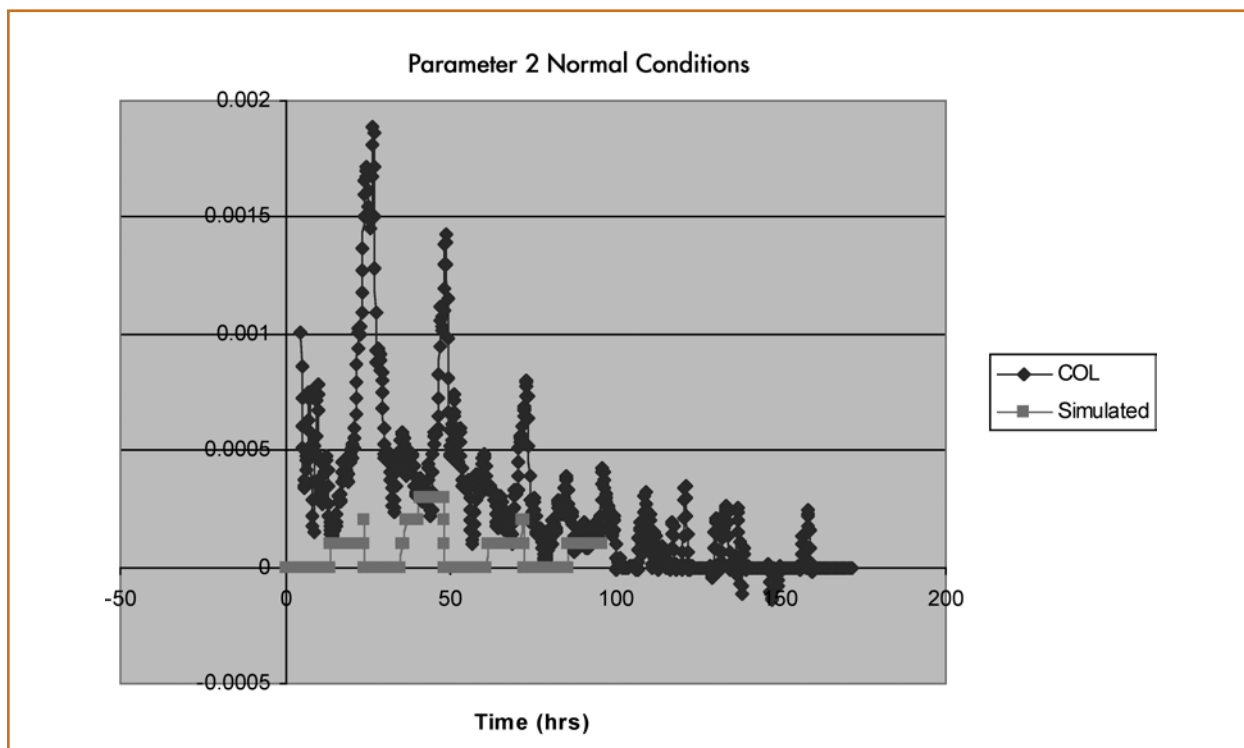


Figure 10. Parameter Set 2 under normal light conditions.

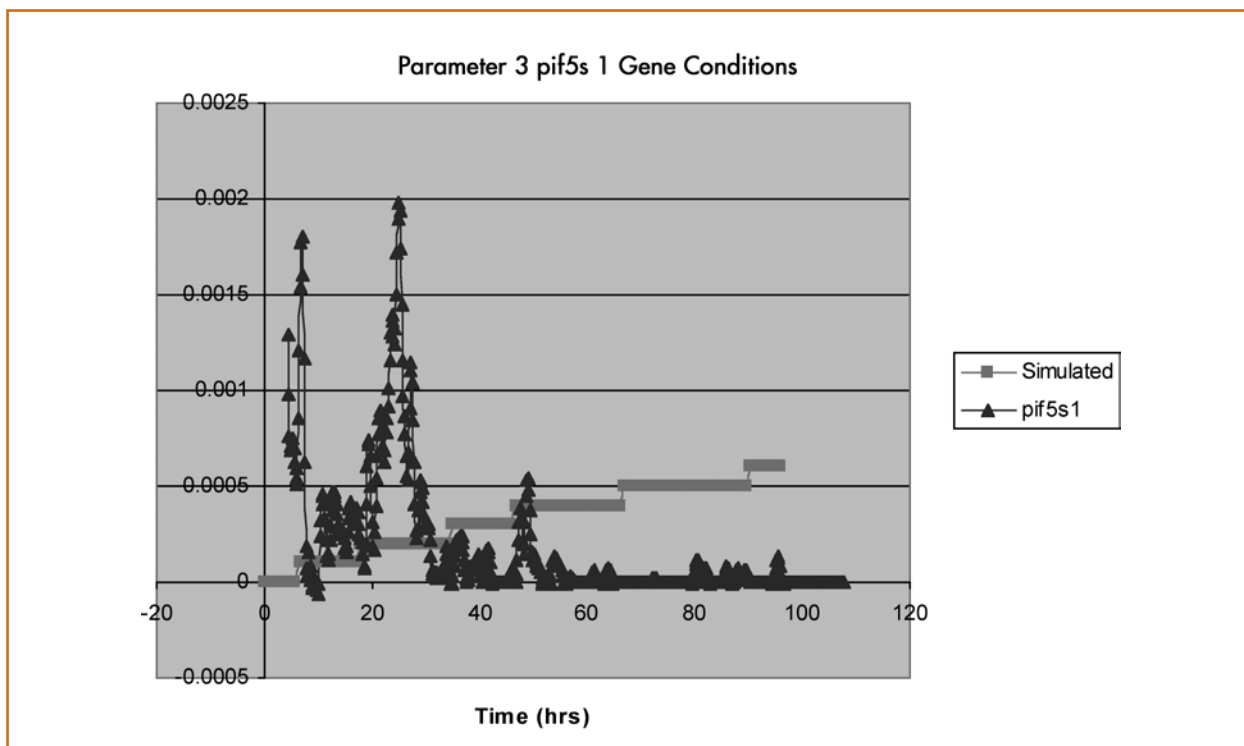


Figure 11. Parameter Set 3 simulated in a pif5 knockout plant.

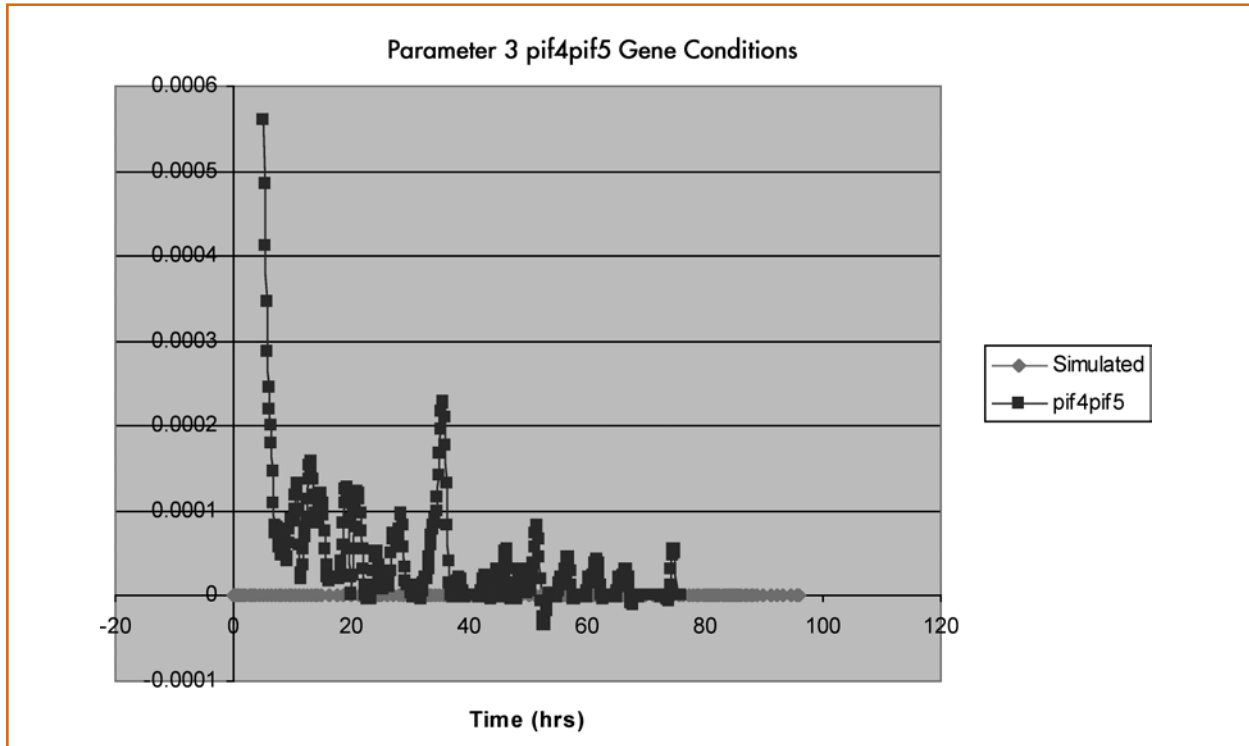


Figure 12. Parameter Set 3 simulated in a pif5 double knockout plant.

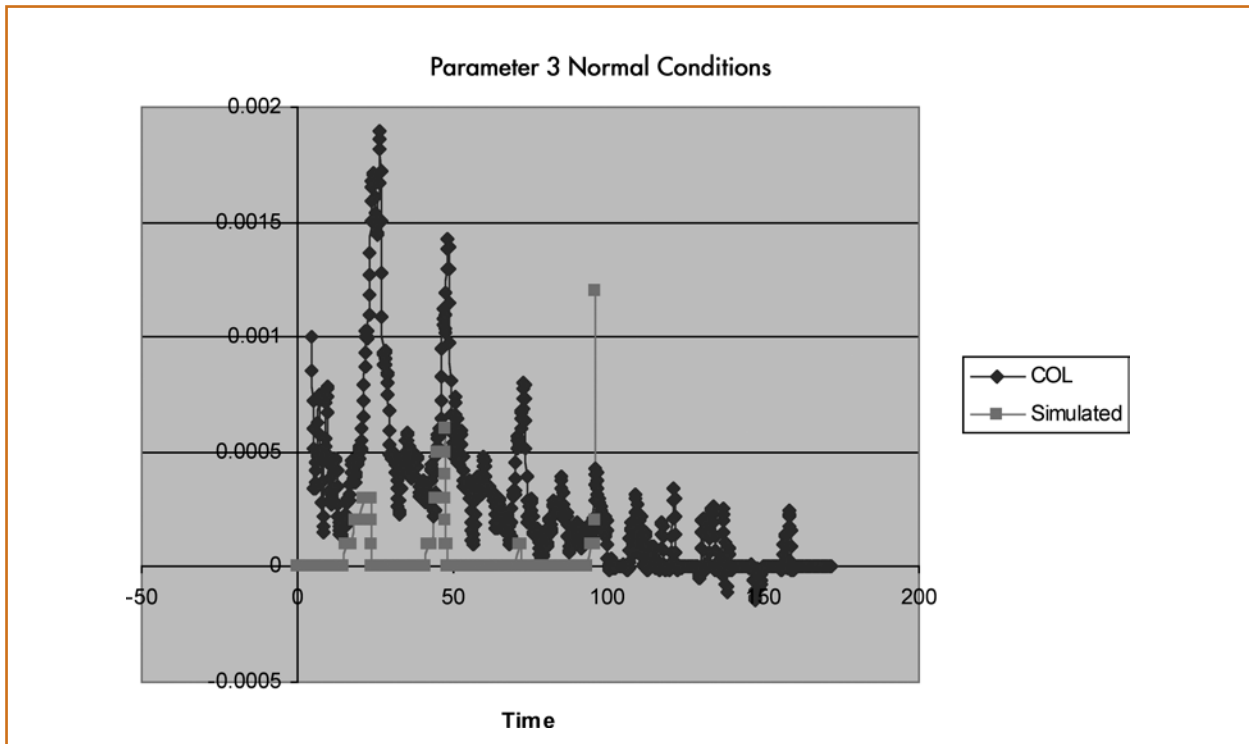


Figure 13. Parameter Set 3 simulated under normal light conditions.

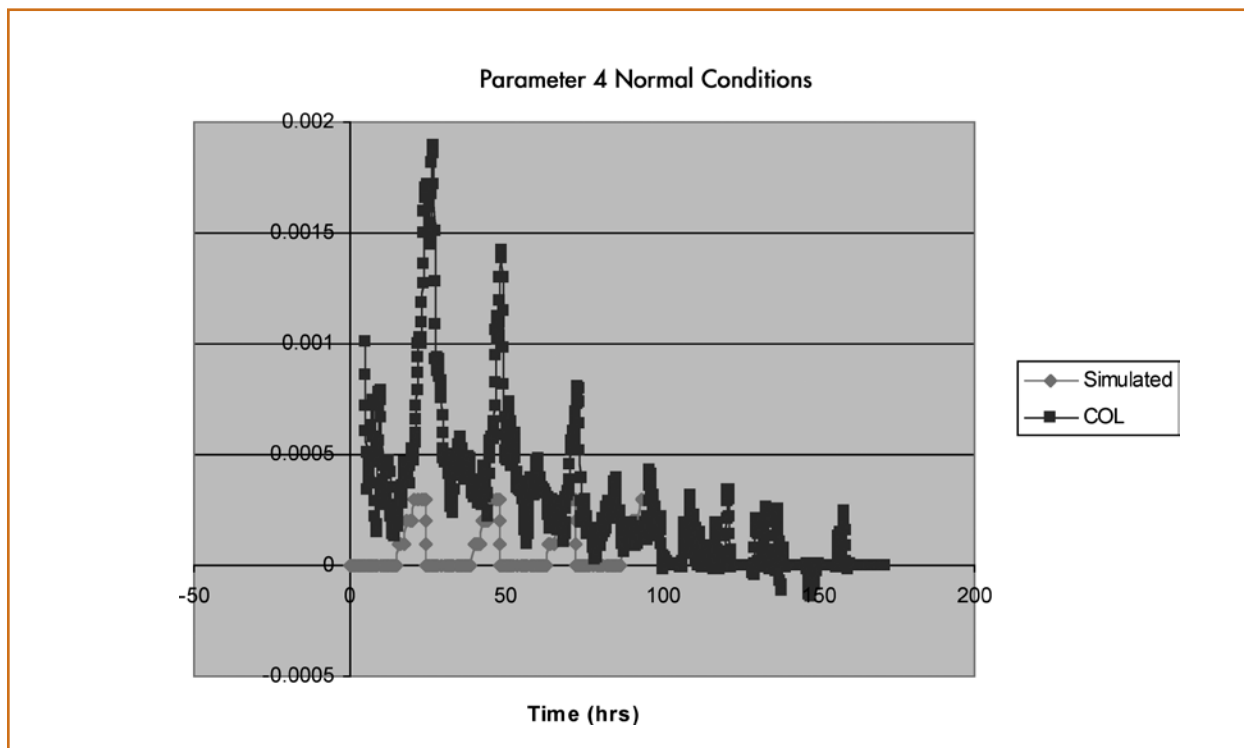


Figure 14. Parameter Set 4 simulated under normal light conditions.

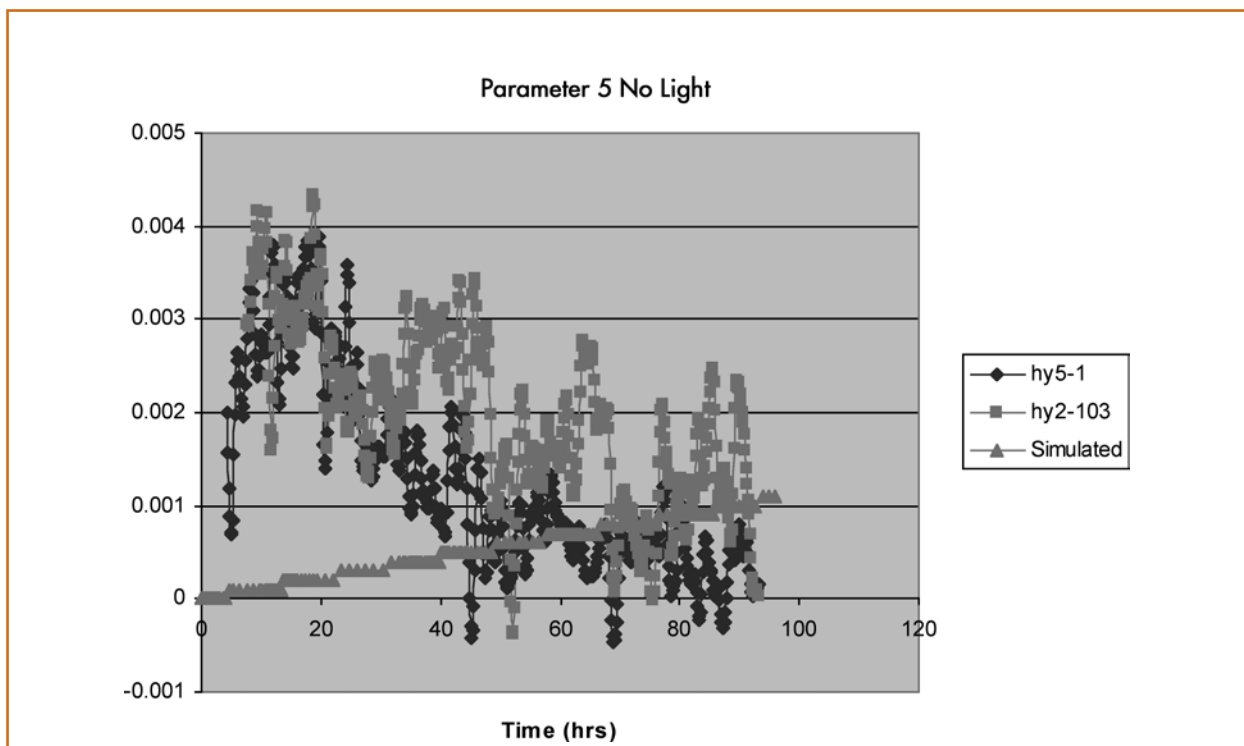


Figure 15. Parameter Set 5 simulated in darkness.

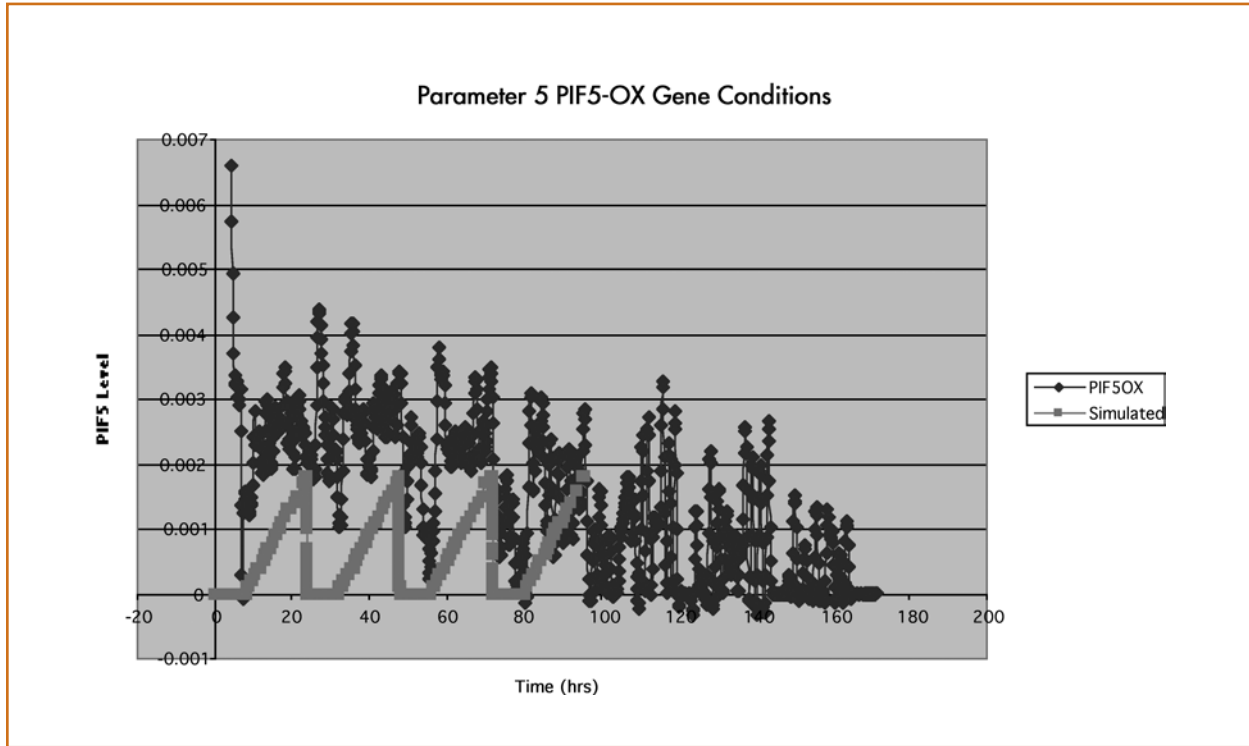


Figure 16. Parameter Set 5 simulated in a PIF5 overexpressing plant.

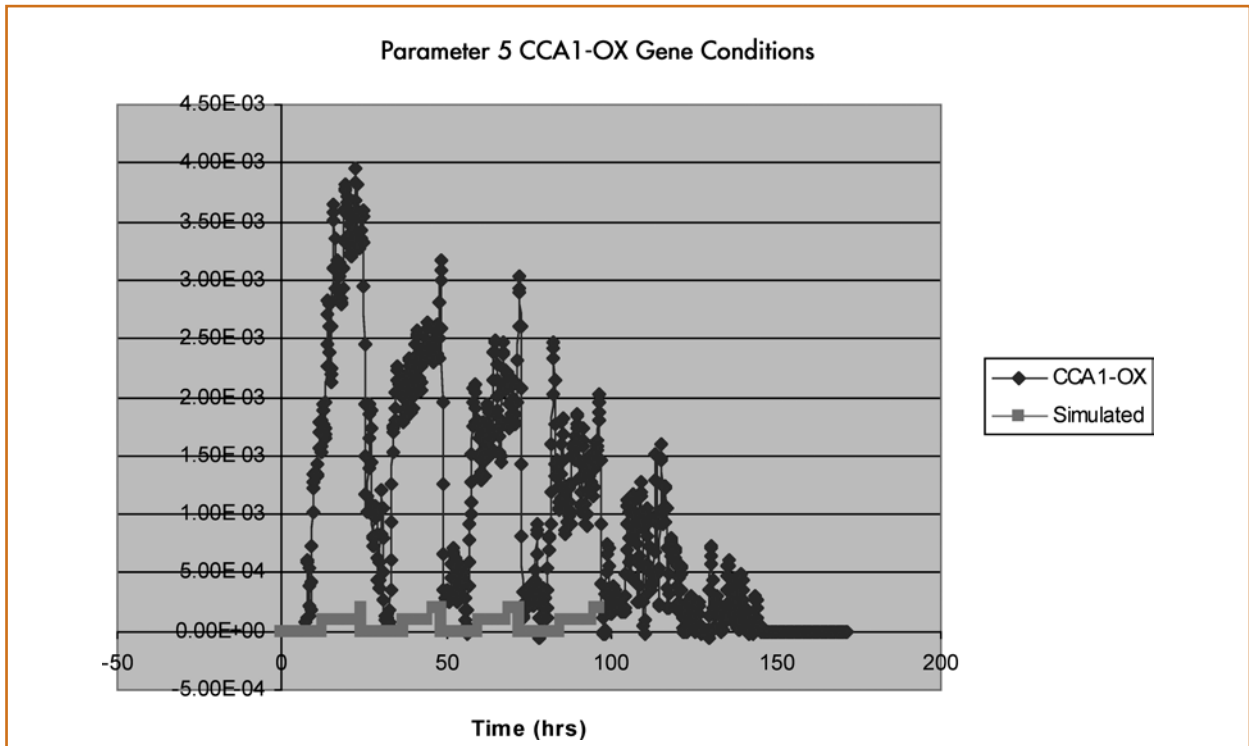


Figure 17. Parameter Set 5 simulated in a CCA1 overexpressing plant.



Figure 18. Parameter Set 5 simulated under normal light conditions.

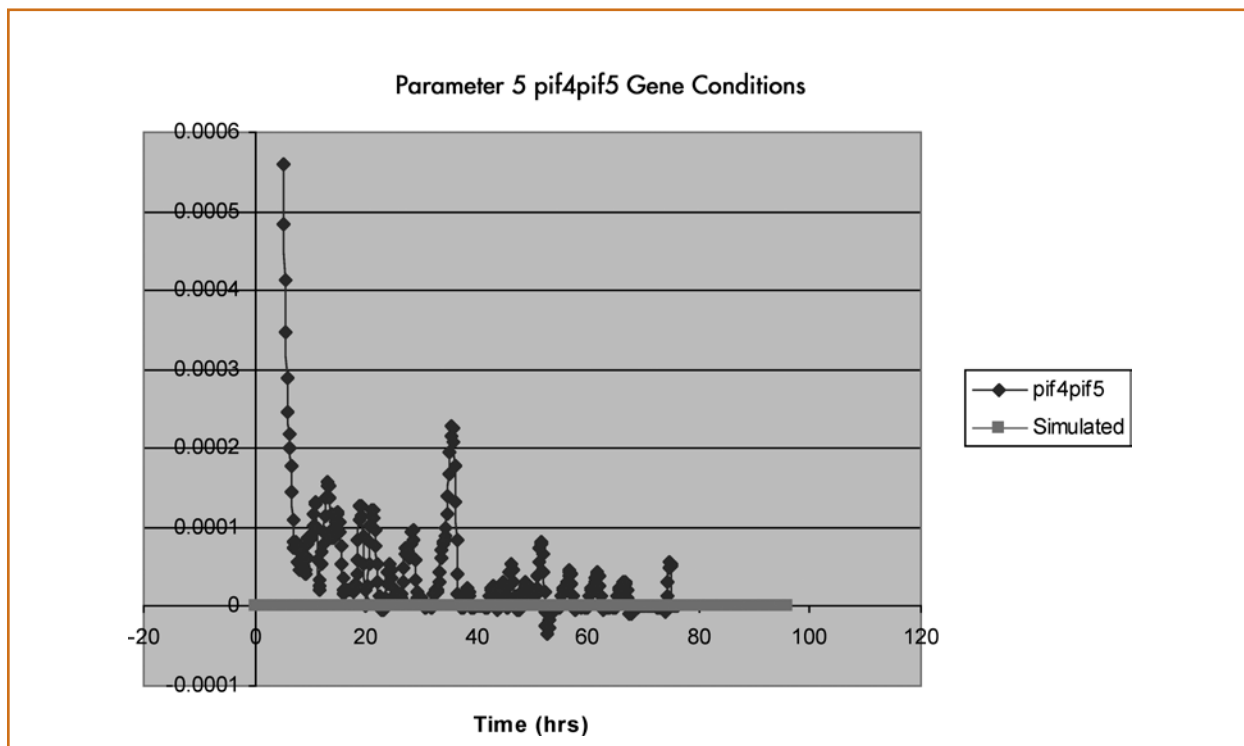


Figure 19. Parameter Set 5 simulated in a pif5 double knockout plant.

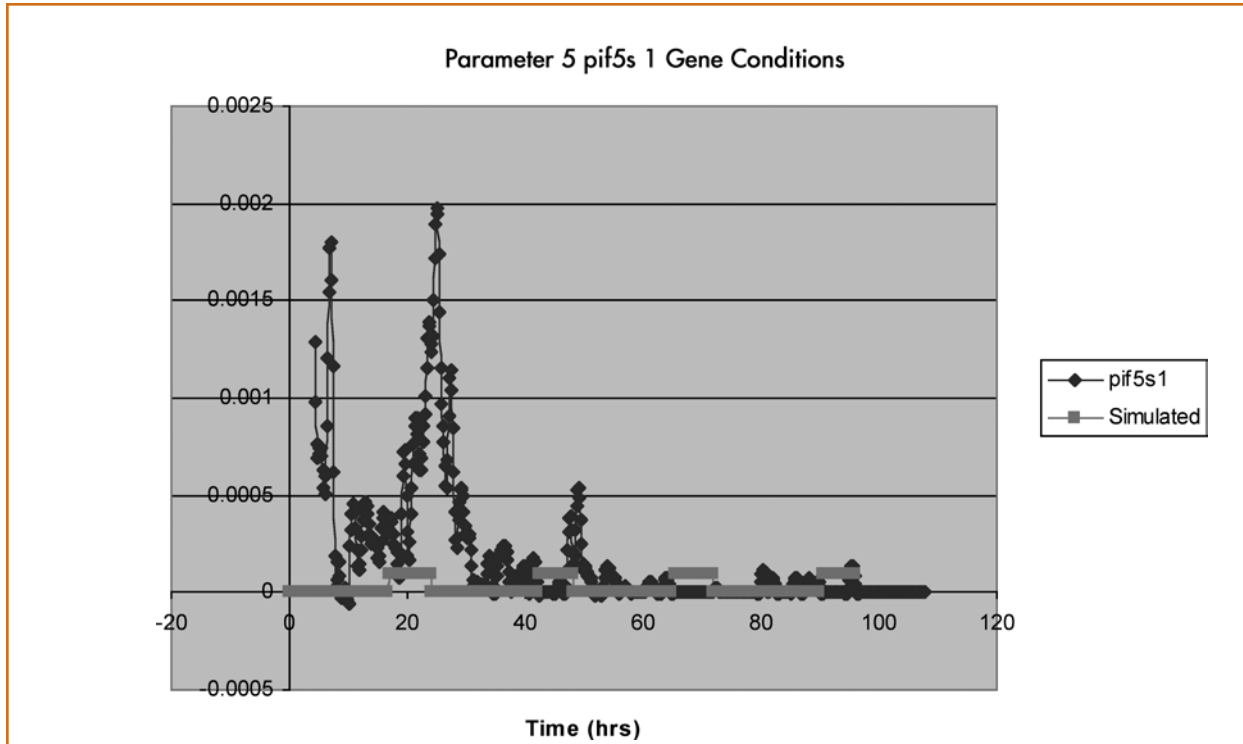


Figure 20. Parameter Set 5 simulated in a *pif5* knockout plant.

Conclusions

As can be seen, all of our parameter sets failed to even come close to approximating the experimental data. This failure can most likely be attributed to the simplifications made during the construction of the model. When we made assumptions—such as the experimental growth rate data being equivalent to PIF5 concentrations—we most likely lost some accuracy. However, these results are not surprising given our choices to reduce the complexity of the model. If we were to go back, and for example, replace mass action equations with reaction types better suited to the actual kinetics, we might improve the accuracy of the model. Michaelis-Menten rate equations would be a more appropriate choice for most of our reactions except for the transcription steps, which would be better modeled by Hill Cooperative Kinetic equations. The lack of success of the parameter1 and parameter 5 sets to reproduce conditions where light sensitivity was reduced indicates that there might be light sensing components other than *hy2* and *hy5* that are required for proper PIF5 expression, or perhaps that the way the photoreceptor pathways interact with the PIF5 pathway is more complex than previously thought. The inability of any of the parameter sets to reproduce experimental data when simulated under the *pif4pif5* conditions suggests that PIF5 is not the only factor involved in activating hypocotyl growth, and that hypocotyl growth is possible without it.

The most encouraging result is that some of the simulations, specifically parameters 2, 3, and 5, although not displaying the correct magnitude, showed the correct rhythms under CCA1-OX and Normal conditions. This result demonstrates that, at least in some respects, our model is on the right track. As stated before, the inaccuracies also give us clues as to where the model needs improvement. Our observations strongly support the validity of using a model to analyze a biological pathway, and further justify the further exploration of this model in particular.

References Cited

- Chen, M., J. Chory, et al. (2004). Light signal transduction in higher plants. *Annu Rev Genet* 38: 87-117.
- Gutenkunst, R. N., J. J. Waterfall, et al. (2007). Universally sloppy parameter sensitivities in systems biology models. *PLoS Comput Biol* 3(10): 1871-78.
- Kohchi, T., K. Mukougawa, et al. (2001). The Arabidopsis HY2 gene encodes phytochromobilin synthase, a ferredoxin-dependent biliverdin reductase. *Plant Cell* 13(2): 425-36.
- Locke, J. C., M. M. Southern, et al. (2005). Extension of a genetic network model by iterative experimentation and mathematical analysis. *Mol Syst Biol* 1: 2005 0013.
- McClung, C. R. (2006). Plant circadian rhythms. *Plant Cell* 18(4): 792-803.
- Nozue, K., M. F. Covington, et al. (2007). Rhythmic growth explained by coincidence between internal and external cues. *Nature* 448(7151): 358-61.
- Tsai, K.-Y. and F.-S. Wang (2005). Evolutionary optimization with data collocation for reverse engineering of biological networks. *Bioinformatics* 21(7): 1180-1188.
- Vandenbussche, F., J. P. Verbelen, et al. (2005). Of light and length: regulation of hypocotyl growth in Arabidopsis. *Bioessays* 27(3): 275-84.
- Wang, Z. Y. and E. M. Tobin (1998). Constitutive expression of the CIRCADIAN CLOCK ASSOCIATED 1 (CCA1) gene disrupts circadian rhythms and suppresses its own expression. *Cell* 93(7): 1207-17.
- Zi, Z. and E. Klipp (2006). SBML-PET: a Systems Biology Markup Language-based parameter estimation tool. *Bioinformatics* 22(21): 2704-5.