Introduction to computational modeling of biological systems; A guide to drying up your lab work

L.D.M. van der Meer, 29 February 2024
Supervised by: J.L. Gardiner

Layman's summary

Natural sciences aim to understand the world around us. Biology is no exception and is particularly interested in the structures and interactions that govern how living things interact. Where physics and chemistry can fall back on fundamental building blocks of the universe, biologists can only go back to the simplest known life forms. Even these present complex self-organizing features. To explain the observed behavior, we create models which take the known components of the system and describe their interactions. In recent years the development of faster and faster computers allowed biologists to transform these models into something more. The use of computational biology allows us to take existing models and predict behavior far into the future. From the growth of bacteria, to the infection of plant cells, and the development of roots. The unique advantage of biological modeling is our ability to introduce all sorts of artificial challenges. Changing temperatures to see what can be done to store food longer. Changing the distance between humans to find the optimal range for preventing COVID-19 spread. Testing altered gene expression in the survival of plants under salt stress. Even though these could all be tested in traditional laboratory settings. The use of biological models is faster, cheaper, and more flexible. In this review, we introduce the world of computational modeling and show how these models can be used to support and possibly exceed laboratory experiments.

Abstract

Biological modeling is a quickly developing subsection of biology. Combining biology and mathematics to provide computational models that can predict and explain biological phenomena. Where classic models provide an overview of a system with connections to predict outcomes when changing a single component, computational models can simultaneously update every component in the system. Allowing for more sensitive and time-dependent effects to be observed. We demonstrate how a dynamic model can be created through the use of Ordinary Differential Equations (ODEs). These inform what the values for each compartment within the system is and how they change over time. To help explain we showcase a classical thermodynamic entropy model. The principles of ODEs are then expanded to a multi-compartment system. From here dynamics of bioreactors and bacterial populations can be explained using computational models. To describe distance-dependent systems such as bacterial spread on plates, development of roots, and self-organisation of tissues we introduce an imaginary space named a lattice. These multi-dimensional planes utilize coordinate systems to characterise individual positions in space and provide information about components within the proximity. To optimize computational efficiency, we demonstrate how to connect the edges of a lattice using boundary conditions. These conditions allow for components in the system to travel indefinitely without ever leaving the lattice or becoming strapped in the corners. Once the establishment of components in a biological model is explained, we delve into strategies to parameterize the system. These strategies combine existing laboratory data with derivative-based searching tools to accurately establish values for the connections between components in the system, paving the way for more precise and insightful biological models. As technology advances, particularly with the emergence of Artificial Intelligence-guided programming, it is easier than ever for biologists to apply their field-specific knowledge and develop models to support their work. Therefore this review gives an overview of the essential components to consider when creating models. Alongside explaining every component, we provide examples of well-executed biological modeling projects.

Introduction; Conceptional models vs computational models

In the context of biology, models are often used. They provide a simplified version or simulation which describes, explains or predicts biological phenomena. Classical models are static, they define acting parties with a network of connections between them. The interactions can induce, reduce or transform other components in the model. With regards to transcription factors, these models can be followed to predict the implications of mutations, usually in an on or off-state (Beckwith, 1967).

With the introduction of computers, it became feasible to set up dynamic models. These often take preexisting static models and repeatedly compare the state and values of each component over time. By taking the initial values and iterating over time, dynamic models can show changes in the relative contribution of individual parts to the whole system (*Modeling Complex Systems*). Or highlight changes caused, either directly or indirectly, by single parts. Even though it was always possible to iterate models over time by hand, the advancements in computational modeling have expanded the possibilities in terms of time steps and the number of components tremendously. This allows the simultaneous modeling of many dynamic parts. Giving excellent insight into the complex behaviour that arises in the systems and comparing them with processes of spontaneous collaboration observed in nature. From embryogenesis (Jaeger, 2009) to the synchronisation of fireflies (Strogatz, 1997) and the formation of intricate root structures (Rutten & Tusscher, 2019).

In many cases, models help us understand and predict biological behaviour. However, the initial assumptions we make about a system shape the results obtained from biological models. Here, we discuss the principle components of biological models and present how these were implemented in relevant research. Where possible, we highlight what considerations should be taken when implementing these models and how these might shape the obtained results.

How to set up a dynamic model

To set up a dynamic model, we start by defining a set of Ordinary differential equations (ODEs) (3). These comprise two parts—namely, the compartments in the system and the interaction between these compartments. The simplest dynamic model described by statistical mechanics is a two-compartment system describing thermodynamic entropy (Friedli & Velenik, 2017). Here two connected compartments (S^1, S^2) in an ideal system are defined by three variables, their Volume (V), Number of particles (N), and their Energy (U) (Figure 1).

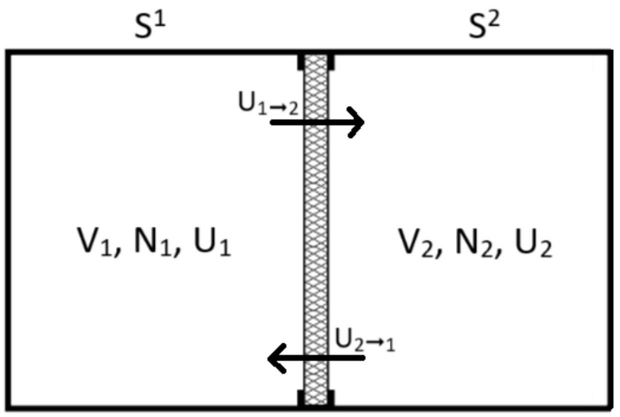

Figure 1, two-compartment system to describe thermodynamics. *The image shows two compartments (S^1, S^2) with their associated volume (V) and number of particles (N), and energy (U). In the middle, there is a fixed barrier that doesn't allow the exchange of particles but does allow energy to flow between the systems as indicated by the arrows. This figure has been adapted from (Friedli & Velenik, 2017).*

By specifying more details about the variables in the system, we find:

$$V_{total} = V_1 + V_2, \qquad N_{total} = N_1 + N_2, \qquad U_{total} = U_1 + U_2 \qquad (1)$$

This indicates that each variable has a fixed total value in the shown system. Furthermore, the separation of the two compartments by a semi permeable barrier results in both (V_1, V_2) and (N_1, N_2) being fixed (Figure 1). For the energy term, we can express both variables U_1 and U_2 in terms of U_1, because the total energy in the system is fixed:

$$(U_1, U_2) \rightarrow S(U_1, V_1, N_1) + S(U_2, V_2, N_2) = S(U_1, V_1, N_1) + S(U_{total} - U_1, V_2, N_2) \qquad (2)$$

From here, it follows that the two compartments are at equilibrium when the energy flow between them is equal. In other words, at every time step, the changes in $U_{1\to2}$ and $U_{2\to1}$ are equal:

$$\partial S/\partial U\,(U_1, V_1, N_1) = \partial S/\partial U\,(U_2, V_2, N_2) \qquad (3)$$

This equation creates a simplistic scalable system for energy flow that is core to describing the flow between any number of compartments in a lattice system. In the next section we show what happens when we increase the number of components in a one dimensional system.

Modeling at scale

In equation (3) we assumed only two compartments in the system. Therefore, any change in one meant an opposite change in the other. For S_1 to gain one unit of energy, S_2 must lose one unit of energy. For systems with more than two components, there are generally two ways of incorporating exchange between components. Either global, meaning each component has interaction with every other component. Or local, meaning each component has direct interaction with its nearest neighbours. Expanding our two-compartment model to a four-compartment model visualises how one might adapt the nearest neighbour method (**Figure 2**).

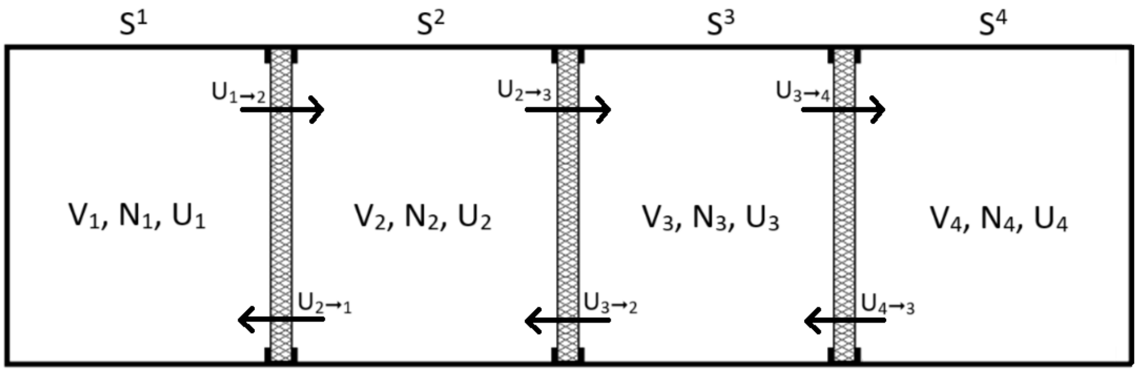

Figure 2, Expansion from a two to four compartment model. *The image shows four compartments (S^1, S^2, S^3, S^4) with their associated volume (V), number of particles (N), and energy (U). The system's total energy is not shown but can be assumed to be U_{total}. between compartments, there are fixed barriers that do not allow for the exchange of particles, but do allow energy to flow freely within the systems. The arrows passing though the barriers indicate the exchange of energy between the compartments, expressed in energy ($U_{i\to j}$) flow from compartment one (U_i) to (→) compartment two (U_j).*

Here the number of components is increased from two to four. The number of interactions between the system however, increased from two to six, which is a precursor of how these systems quickly grow to be more complex. The new notation used $U_{1\to 2}$ is sufficient for now and indicates the start and ending point of the energy flow. As we can see compartments only interact with the ones right next to them, but the previously mentioned principles about equilibrium still applies (3). Since for each compartment the number of particles (N) and the volume (V) are fixed, we can again state that the system is in equilibrium when the energy flow ($U_{i\to j}$) in and out are equal. If we were to suddenly increase the energy in S^1, we can imagine $U_{1\to 2}$ will become larger than $U_{2\to 1}$. The energy would slowly redistribute throughout the system. Even though S^1 and S^4 are not directly connected, the energy can still travel between them over enough time steps.

We can make our first model by substituting the compartments $S^{1,2,3,4}$ with biologically relevant terms. For instance, bacteria growth under viral pressure in a bioreactor (Figure 3). Here we see the compartments are defined as resistant, uninfected, and infected bacteria, together with their growth resources, and the virus particles (Middelboe et al., 2001). Again the total resources in the system are fixed, therefore any increase in uninfected bacteria population will decrease the available resources in the system. The arrows that indicate the interactions between the components in the system represent terms in the differential equations associated with the system. Working out the values for the constants such as growth rate and death rate can be done by interpreting experimental data.

Bioreactors and other liquid based systems are usually described by ODEs, since they are well mixed and every component of the system is in close contact with every other component. This is not the case for many other biological systems, where separation in space plays a large role in the likelihood of two components interacting. In the next section we will expand our compartment model to a large plane called a lattice.

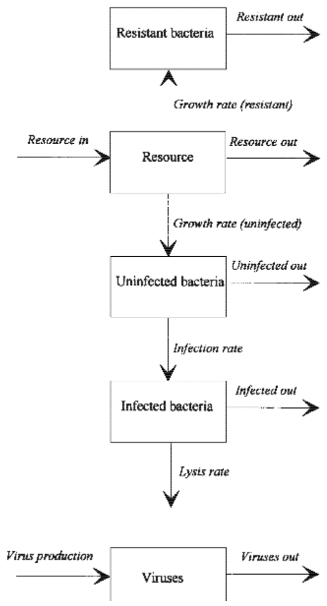

Figure 3, flow chart of bacteria growth and infection with a virus. The image shows the boxes with the different compartments and connecting arrows that indicate some interaction between them. The image was adapted from (Middelboe, 2000).

Dynamic models using a lattice space

A lattice is an imaginary space composed of neighbouring compartments, hereafter cells. The size and shape of a lattice can vary, but for simplicity, we will assume a 2D square space with sides (L_1, L_2)(Figure 4). Each cell in the lattice has a unique (0 to L_1, 0 to L_2) coordinate. Cells can represent many things and hold one or more values. The advantage of using a lattice is the ease with which you can organise interactions.

To explain this, we will use the spread of a virus though an plate with bacteria. If each cell in the lattice represents a bacteria, we can assign certain states to each cell, such as susceptible or infected. Essentially we are describing a system with 3 compartments, however they are spread out over a larger plane. To travel through the plane and find each individual cell, we can utilize the coordinate system. Going from position (0, L_2) to (L_1, L_2) implies moving though the rows of the lattice grid. We use "i" to specify the number in L_1. By iterating through these numbers, we can effectively walk through the grid, examining each individual. To identify the state of the surrounding cells we can again utilize the current value of i. The neighbouring cells can be found by subtracting 1 (neighbour to the left) or adding one (neighbour to the right). Once we have identified the state of neighbouring cells, we can calculate the likelihood of the central cell will be infected. For instance, we can sum the states of the surrounding cells (uninfected = 0, infected = 1) and multiply by some infection rate λ_{12} (Figure 4).

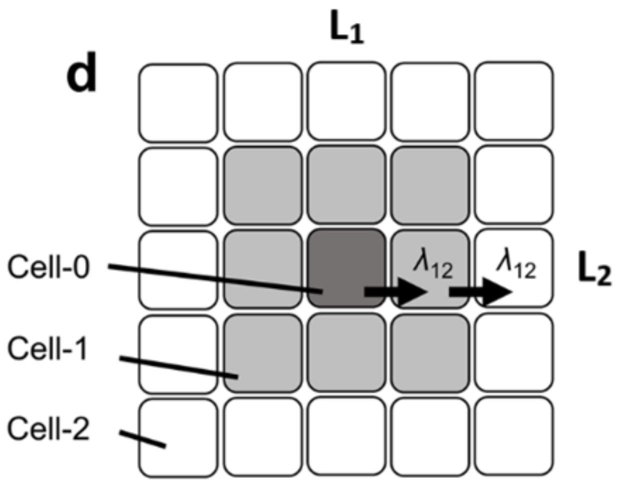

Figure 4, Schematic representation of viral infection sight in N. benthamiana leaf*. The dark grey cell (Cell-0) in the middle of the lattice represents the original infection sight. The surrounding rings are the closest neighbours (Cell-1) and their closest neighbours (Cell-1) and their closest neighbours (Cell-2).* λ_{12} *represents the likelihood of a virus particle moving between cells. This figure has been adapted from (Abebe et al., 2021).*

An example that utilizes this idea, describes the spread of multiple virus particles from one *N. benthamiana* cell to another using several differential equation (Abebe et al., 2021; Miyashita & Kishino, 2010). These describes the small chance of the virus particles moving to a neighbouring cell dependent on the number of virus particles in the host cell. In the experiment presented in figure 4, two fluorescently tagged viruses were infected in the plant. The subsequent spread of the virus particles and the observed mix of co-infected or single-infected cells indicated the likelihood of neighbour infection. Ultimately, the average number of founder virus particles was determined by combining wet-lab experiments with this dynamic model (Abebe et al., 2021). Here we described systems where neighbouring cells are all treated equally. In the next section we will introduce preferred directions and see how this can lead to self-organisation of large structures.

Gradient formation

In the previous example, lattice cells represented plant cells. Here we will expand on the lattice structure and consider multiple lattice cells forming one biological cell. Demonstrating how a uniformly shaped lattice grid can model plant roots, which have distinct cell types, each with their own orientation and expression patterns (Grieneisen et al., 2007). As a result of these expression patterns, the establishment of known hormone gradients can be explained. The plant hormone Auxin and its gradient across tissues is essential for the development of roots (Cruz-Ramírez et al., 2012). Auxin is transported from cell to cell by PIN-FORMED (PIN) proteins (Rutten & Tusscher, 2019; van den Berg et al., 2016). The expression of PIN proteins is tissue-specific, and the membrane localisation is dependent on the specific members of the PIN protein family (van den Berg et al., 2016). In the following model, several cell types were characterised, each with its own PIN localisation and distinct height (**Figure 5**, C). The PIN localisation was based on microscope images, showing the need for high-quality experiments to verify values for the constants used in the model. Running the model leads to a reverse fountain pattern which is common for Auxin flow (**Figure 5**, E)(Goh et al., 2014). Even though the shape of the root tip is far from biologically accurate, Auxin accumulation can be observed just below the Quiescent Center (QC) just as expected.

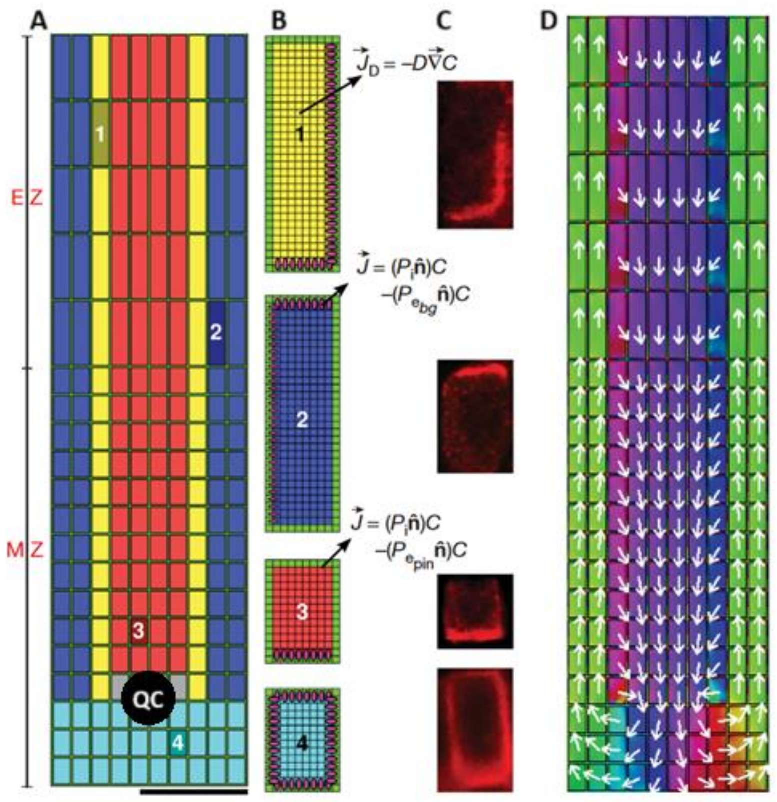
A
B
C
D
EZ
MZ
1
2
3
4
QC
$\vec{J}_D = -D\vec{\nabla}C$
$\vec{J} = (P_i\hat{n})C - (P_{e_{bg}}\hat{n})C$
$\vec{J} = (P_i\hat{n})C - (P_{e_{pin}}\hat{n})C$

Figure 5, Rudimentary model of auxin transport by PIN proteins. *The figure shows an arrangement of root models. A) A model of a single root is shown. The Different cell types are depicted in different colours. At the bottom the Quiescent Center (QC) is shown. The characteristics of the cell types are shown in panel B. B) the shape and localisation of PIN proteins are shown for each of the four cell types used in the model. Additionally the formulas for the flow of Auxin for each of the PIN proteins is linked to the specific cell expressing them. C) microscope images of single cells with stained PIN proteins is shown. These form the basis for localization of the PIN proteins in panel B. C) The same modelled root is shown with the arrows indicating the flow of Auxin through the root. The colour also indicates the flow of auxin, however the legend for this was not added as the same information can be derived from the arrows. The images were adapted from (Grieneisen et al., 2007).*

Early studies into gravity-driven root orientation showed PIN3 & PIN7 were primarily involved in the initiation of asymmetric auxin transport, while PIN2 and auxin transporter (AUX1) were important for the maintenance of the asymmetry (van den Berg et al., 2016). In salt-related stress, however, PIN2 was primarily downregulated on the salt-exposed side of the root. This means that the active export of Auxin decreased and a local maxima in auxin concentration occurred (van den Berg et al., 2016). While investigating the role of PIN2 proteins in the induction of root curvature under salt stress conditions van den Berg et al., used computer models like the one we showed to validate whether PIN2 reduction alone can explain the observed root restructuring. It turns out that, in the simple model, PIN2 reduction couldn't induce a local auxin maxima on the side opposite to the salt source. However, upon improving the model and taking into account the shape or the root tip, PIN2 reduction did lead to the observed phenotype. Additionally, the inclusion of AUX1 into the model further improved its predictive capabilities (van den Berg et al., 2016).

Boundary conditions

The aforementioned models have had clearly defined edges. Be it in the form of the outer root layer or the edge of a leaf. What happens if edges are undesirable in a system? We could simply increase the lattice space and set L_1 and L_2 to be 100 million, the problem there is the exponential growth in computational power needed to simulate the system. As the dynamics for each cell have to be calculated for each time step. Instead we can be creative about the space of the imaged lattice instead. The use of continuous boundary conditions does just that (Gros, 2008). Imagine a sheet of paper (lattice) with an ant on it. If the ant walk far enough he will fall off the edge. However, of we fold the sides of the paper into a cylinder. The ant can walk in one direction for an antless period of time. The same can be done for either end of the paper, by folding them together and creating a donut shape (**Figure 6**).

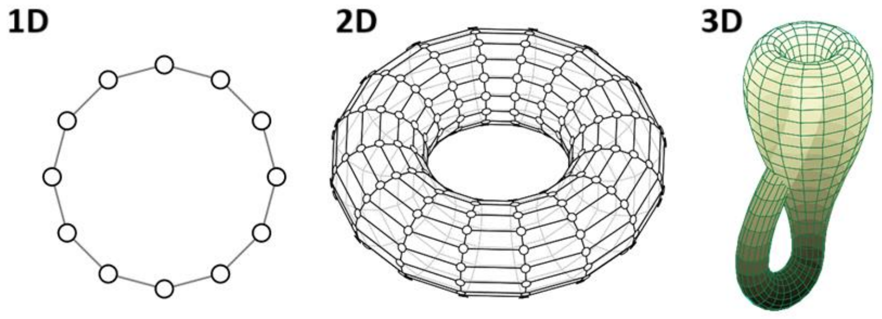

***Figure 6, periodic boundary conditions in a 1D, 2D, and 3D lattice space**. The figures show how periodic boundary conditions can be achieved for 1D, 2D, and 3D structures. The image was adapted from (Friedli & Velenik, 2017).*

Examples of when to use periodic boundary conditions are with modeling protein structures. When modeling a protein containing hydrophobic residues in water, they might cluster in a corner of the lattice, as there is the least interaction with water that way. When using periodic boundary conditions there would be no corner and the correct folding could occur instead. One more point of consideration when working with protein folding in particular is the size of the lattice. This should be at least 2.5 times longer than the unfolded protein (Dominguez et al., 2003). Otherwise it is possible for the charged ends of the protein to interact with each other over the boundary conditions, leading to unrealistic tension in the protein. Extending periodic boundary conditions to a 3D lattice would be shaped like a Klein bottle, which you need a 4th dimension for to properly represent (**Figure 6**) (Kneser, 1924). With this, we have established how elements of a model can interact. How a lattice can be used to position elements and calculate their proximity to each other. And how we can compute a nearly infinitely large space, with finite computing power. Hereafter we focus on coupling our model with reality via the use of constants.

Parameterisation of the system

Once we have identified which variables to use in a model, they need to be linked with constants to determine their relative contribution to the equation. The value of these constants can be derived from experimentally acquired data. Here we show the basis of fitting parameters of mechanistic models to experimental data. To guide us, a model for predicting some linear growth curve is compared to experimental data showing one free parameter (Figure 7).

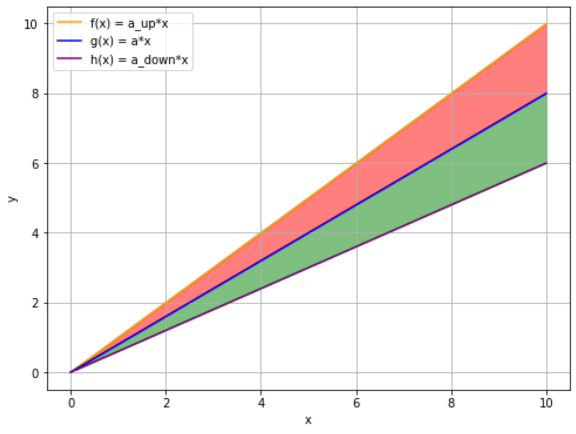

*Figure 7, Example of calculating the "cost" of a function. The graph shows three lines and highlights the area between them. The formulas for the experimental data g(x) = a*x (blue) and both approximations f(x) (orange) and h(x) (red) are given. The area between the approximation and the experimental data is highlighted in either red or green. Both axes are unitless.*

In this case, the formula describing the experimental data has one constant and one variable

$$F(x) = ax$$

To identify the constant which most closely predicts the experimental data we can take a range of values for "a" and calculate their cost. Cost is a measure of the area under the curve between the two lines (Kitano, 2002). It follows that the greater the cost/area, the more the model and experimental data differ. Therefore, our goal in parameter acquisition is to find a value with minimal cost. Combining all the different costs for a value range of one parameter gives you a cost function (Figure 8). For a single parameter it is quite easy to see where the local minima in its cost function appears.

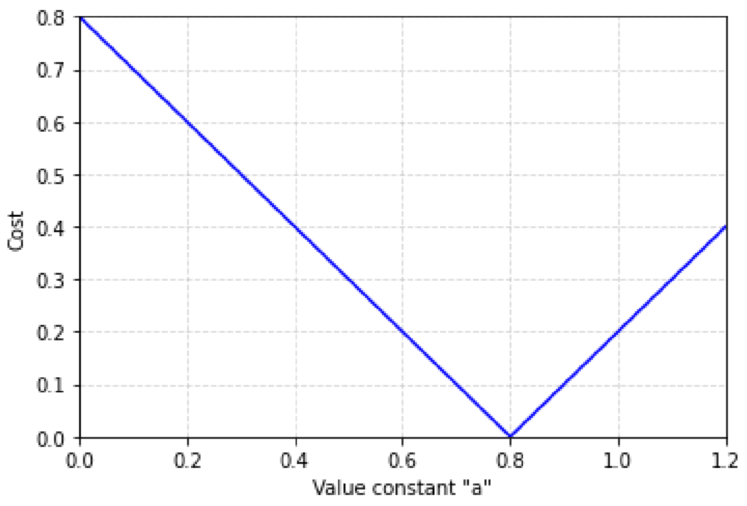

Figure 8, Cost function for single variable "a". The graph shows the cost function for a range of values for the variable "a". The cost is expressed as a unitless area under the curve shown in figure 7.

However, when multiple constants are at play it is more likely some will mask the others, resulting in difficult to acquire minima. Note that the number of variables is not limited to one or two, but can be any number. In Figure 9, we show a 3D space with two parameters, 4 and 6, and their associated cost function. Finding the local minima in this plane requires the use of optimisation models (Slezak et al., 2010). Here we present two families of optimisation methods, namely descent methods and sampling methods.

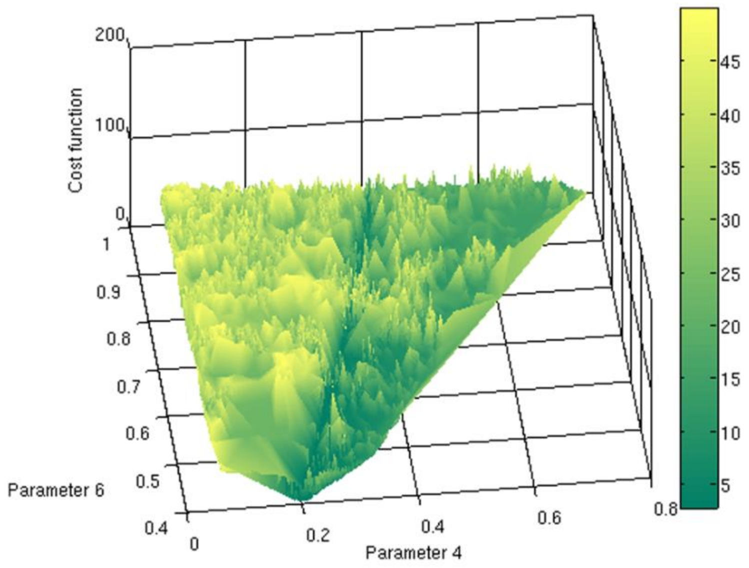

Figure 9, 3D representation of cost landscape for 2 parameters. The cost function for two variables is represented for any possible combination in a 3D landscape. The colour range indicates areas of low cost in dark green and peaks in cost as bright green. The image was adapted from (Slezak et al., 2010).

Decent models

In general, decent models assume a position on the cost function plane and calculate direction of the force (Meza, 2010). Remember that one can calculate the force by taking the negative derivative of the function. In our simple model, there are two components to the function (Figure 8). To the left of the minima F(x) = -x, to the right G(x) = x. Taking the derivative of these functions gives F'(x) = -1 & G'(x) = 1 respectively. Taking the negative of these results shows the force to the left of the minimum is in the positive direction, while to the right it is in the negative direction. From here a step in the direction of the force can be taken. This is where different approaches to minimizing in decent models are employed. In the simplest form a steepest descent model will take a fixed step size, recalculate the force and take the next step (Fliege & Svaiter, 2000). If you are lucky this could find the minima, but more often than not, this leads to an oscillation around the minimum of the function. Since if the step size is larger than the distance between the starting point and the true minima, you will shoot past it. With a fixed step size this keeps happening. The Levenberg-Marquardt (LM) method aims to solve this issue by implementing a dampening factor (Choi, 2010; Moré, 1978). Every time a step is taken the step size increase if the force at the next step is in the same direction. On the other hand the step size decreases when the direction of the force changes. Even though this will more accurately find the minima over time. There is a trade of between accuracy and speed when using descending models.

Sampling methods

Sampling methods or random search models such as the Monte Carlo optimization technique asses the minima of a cost landscape by sampling (Shapiro, 2003). Instead of spending computational power moving towards the possible minima, they repeatedly sample random positions and calculate their cost (Choi, 2010; Shapiro, 2003). When all samples have been taken, they are compared and the location with the lowest cost is considered the optimum. The upside of these models is their ability to sample the whole plane, without getting stuck in local minima. The down side is that you can't always predict the size of the cost function, so knowing how many samples is sufficient can be challenging. Evolutionary algorithms are another a subclass of random search models. These use the principles of biology by combining the best scoring samples and either crossing or mutating them (Kansal et al., 2020). Over multiple generations, these models are able to both escape local minima and search especially noisy energy landscapes.

Going back to the previously mentioned article from figure 9 adds some nuance to the use of minimization methods (Slezak et al., 2010). In their analysis they took a preexisting model of vasculair tumor formation and added additional parameters to more accurately predict its development. The model was expended to use 6 constants instead of 3 and multiple minimization techniques were used to find the minima of the cost function. In the end even though they were able to fit the model more accurately to the training data using the LM method, the overall performance of the model decreased. This was due to new found parameters falling outside the range of values observed in experiments (Slezak et al., 2010). This "overfitting" of a model can occur when to much weight is placed on the training data, resulting in random noise being interpreted as biologically relevant data. This example again shows that computational models should be developed in parallel with wet lab experiments.

conclusion

Together the discussed properties of biological models should provide enough insight to conceptualize your own. Depending on the specific application relevant literature can be sought after. These are often accompanied with the code used to run the presented model. From there, identify your own components and establish a basic set of interactions between them. To prevent unrealistic situations the ODEs should be balanced, meaning any addition to one component subtracts from another. Once you have the basis of your model specific problems can be addressed. We did not touch upon it in this review, but population dynamics should have restricted growth rates or maximum sizes. This can be established using density dependent growth and death rates. As we saw with the model for auxin transport, it is sometimes required to add more components to explain certain phenomena such as the formation of gradients. Despite this, it is often better to keep the model as simple as possible. With regards to parameterisation, in simple acquisition of the specific constant values through experiments could be preferable over minimization methods. For more complex systems, developments in the area of finance have provided a plethora of techniques for efficient and fast derivative based minimization. Overall the most effective way of learning about biological modeling is by creating your own. As mentioned before, the onset of advanced AI language models has made it easier than ever to translate an idea into usable code and we strongly encourage the reader to try setting up a project of their own.

References

Abebe, D. A., van Bentum, S., Suzuki, M., Ando, S., Takahashi, H., & Miyashita, S. (2021). Plant death caused by inefficient induction of antiviral R-gene-mediated resistance may function as a suicidal population resistance mechanism. *Communications Biology*, *4*(1), Article 1. https://doi.org/10.1038/s42003-021-02482-7

Beckwith, J. R. (1967). Regulation of the Lac Operon. *Science*, *156*(3775), 597–604. https://doi.org/10.1126/science.156.3775.597

Choi, S. (Ed.). (2010). *Systems Biology for Signaling Networks*. Springer. https://doi.org/10.1007/978-1-4419-5797-9

Cruz-Ramírez, A., Díaz-Triviño, S., Blilou, I., Grieneisen, V. A., Sozzani, R., Zamioudis, C., Miskolczi, P., Nieuwland, J., Benjamins, R., Dhonukshe, P., Caballero-Pérez, J., Horvath, B., Long, Y., Mähönen, A. P., Zhang, H., Xu, J., Murray, J. A. H., Benfey, P. N., Bako, L., … Scheres, B. (2012). A Bistable Circuit Involving SCARECROW-RETINOBLASTOMA Integrates Cues to Inform Asymmetric Stem Cell Division. *Cell*, *150*(5), 1002–1015. https://doi.org/10.1016/j.cell.2012.07.017

Dominguez, C., Boelens, R., & Bonvin, A. M. J. J. (2003). HADDOCK: A Protein−Protein Docking Approach Based on Biochemical or Biophysical Information. *Journal of the American Chemical Society*, *125*(7), 1731–1737. https://doi.org/10.1021/ja026939x

Fliege, J., & Svaiter, B. F. (2000). Steepest descent methods for multicriteria optimization. *Mathematical Methods of Operations Research*, *51*(3), 479–494. https://doi.org/10.1007/s001860000043

Friedli, S., & Velenik, Y. (2017). *Statistical Mechanics of Lattice Systems: A Concrete Mathematical Introduction* (1st ed.). Cambridge University Press. https://doi.org/10.1017/9781316882603

Goh, T., Voβ, U., Farcot, E., Bennett, M. J., & Bishopp, A. (2014). Systems biology approaches to understand the role of auxin in root growth and development. *Physiologia Plantarum*, *151*(1), 73–82. https://doi.org/10.1111/ppl.12162

Grieneisen, V. A., Xu, J., Marée, A. F. M., Hogeweg, P., & Scheres, B. (2007). Auxin transport is sufficient to generate a maximum and gradient guiding root growth. *Nature*, *449*(7165), 1008–1013. https://doi.org/10.1038/nature06215

Gros, C. (2008). *Complex and Adaptive Dynamical Systems: A Primer*. https://doi.org/10.48550/ARXIV.0807.4838

Jaeger, J. (2009). Modelling the Drosophila embryo. *Molecular BioSystems*, *5*(12), 1549–1568. https://doi.org/10.1039/B904722K

Kansal, S., Kumar, H., Kaushal, S., & Sangaiah, A. K. (2020). Genetic algorithm-based cost minimization pricing model for on-demand IaaS cloud service. *The Journal of Supercomputing*, *76*(3), 1536–1561. https://doi.org/10.1007/s11227-018-2279-8

Kitano, H. (2002). Computational systems biology. *Nature*, *420*(6912), Article 6912. https://doi.org/10.1038/nature01254

Kneser, H. (1924). Reguläre Kurvenscharen auf den Ringflächen. *Mathematische Annalen*, *91*(1), 135–154. https://doi.org/10.1007/BF01498385

Meza, J. C. (2010). Steepest descent. *WIREs Computational Statistics*, *2*(6), 719–722. https://doi.org/10.1002/wics.117

Middelboe, M. (2000). Bacterial Growth Rate and Marine Virus–Host Dynamics. *Microbial Ecology*, *40*(2), 114–124. https://doi.org/10.1007/s002480000050

Middelboe, M., Hagström, A., Blackburn, N., Sinn, B., Fischer, U., Borch, N. H., Pinhassi, J., Simu, K., & Lorenz, M. G. (2001). Effects of Bacteriophages on the Population Dynamics of Four Strains of Pelagic Marine Bacteria. *Microbial Ecology*, *42*(3), 395–406. https://doi.org/10.1007/s00248-001-0012-1

Miyashita, S., & Kishino, H. (2010). Estimation of the Size of Genetic Bottlenecks in Cell-to-Cell Movement of Soil-Borne Wheat Mosaic Virus and the Possible Role of the Bottlenecks in Speeding Up Selection of Variations in trans-Acting Genes or Elements. *Journal of Virology*, *84*(4), 1828–1837. https://doi.org/10.1128/jvi.01890-09

Modeling Complex Systems | SpringerLink. (n.d.). Retrieved 29 February 2024, from https://link-springer-com.proxy.library.uu.nl/book/10.1007/b97378

Moré, J. J. (1978). The Levenberg-Marquardt algorithm: Implementation and theory. In G. A. Watson (Ed.), *Numerical Analysis* (pp. 105–116). Springer. https://doi.org/10.1007/BFb0067700

Rutten, J. P., & Tusscher, K. ten. (2019). In Silico Roots: Room for Growth. *Trends in Plant Science*, *24*(3), 250–262. https://doi.org/10.1016/j.tplants.2018.11.005

Shapiro, A. (2003). Monte Carlo Sampling Methods. In *Handbooks in Operations Research and Management Science* (Vol. 10, pp. 353–425). Elsevier. https://doi.org/10.1016/S0927-0507(03)10006-0

Slezak, D. F., Suárez, C., Cecchi, G. A., Marshall, G., & Stolovitzky, G. (2010). When the Optimal Is Not the Best: Parameter Estimation in Complex Biological Models. *PLOS ONE*, *5*(10), e13283. https://doi.org/10.1371/journal.pone.0013283

Strogatz, S. H. (1997). Spontaneous synchronization in nature. *Proceedings of International Frequency Control Symposium*, 2–4. https://doi.org/10.1109/FREQ.1997.638513

van den Berg, T., Korver, R. A., Testerink, C., & ten Tusscher, K. H. W. J. (2016). Modeling halotropism: A key role for root tip architecture and reflux loop remodeling in redistributing auxin. *Development*, *143*(18), 3350–3362. https://doi.org/10.1242/dev.135111