# Chapter 2

## The modeling perspective

This and the following chapter deal with concepts that are not NEURON−specific but instead pertain equally well to *any* tools used for neural modeling.

## Why model?

In order to achieve the ultimate goal of understanding how nervous systems work, it will be necessary to know many different kinds of information:

- the anatomy of individual neurons and classes of cells, pathways, nuclei, and higher levels of organization
- the pharmacology of ion channels, transmitters, modulators, and receptors
- the biochemistry and molecular biology of enzymes, growth factors, and genes that participate in brain development and maintenance, perception and behavior, learning and forgetting, health and disease

But while this knowledge will be necessary for an understanding of brain function, it isn't sufficient. This is because the moment−to−moment processing of information in the brain is carried out by the spread and interaction of electrical and chemical signals that are distributed in space and time. These signals are generated and regulated by mechanisms that are kinetically complex, highly nonlinear, and arranged in intricate anatomical structures. Hypotheses about these signals and mechanisms, and how nervous system function emerges from their operation, cannot be evaluated by intuition alone, but require empirically−based modeling. From this perspective, modeling is fundamentally a means for enhancing insight, and a simulation environment is useful to the extent that it maximizes the ratio of insight obtained to effort invested.

## From physical system to computational model

Just what is involved in creating a computational model of a physical system?

### Conceptual model: a simplified representation of a physical system

The first step is to formulate a *conceptual model* that attempts to capture just the essential features that underlie a particular function or property of the physical system. If the aim of modeling is to provide insight, then formulating the conceptual model necessarily involves simplification and abstraction (Fig. 2.1 left). When a physical system is already simple enough to understand, there is no point in further simplification because we won't learn anything new. If instead the system is complex, a conceptual model that omits excess detail can foster understanding.

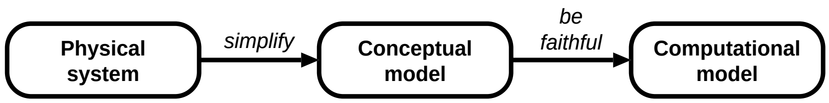



Figure 2.1. Creating a computational model of a physical system involves two steps. The first step deliberately omits real−world complexities to produce a conceptual model. In the second step, this conceptual model must be faithfully translated into a computational model, without any further subtractions or additions.

But some models contain essential irreducible complexities, and even conceptual models that are superficially simple can resist intuition. To evaluate such a model it is often necessary to devise a hypothesis or test in which the behavior of the model is compared against a prediction. *Computational models* are useful for performing such tests. The conceptual model, and the hypothesis behind it, determine what is included in the computational model and what is left out.

When we formalize our description of a biological system, the first language we use is mathematics. The conceptual model is usually expressed in mathematical form, although there are occasions when it is more convenient to express the concept in the form of a computer algorithm. **Chapter 3** is concerned with mathematical representations of chemical and electrical phenomena relevant to signaling in neurons.

## Computational model: an accurate representation of a conceptual model

A computational model is a working embodiment of a conceptual model through the medium of computer simulation. It can assist hypothesis testing by serving as a virtual laboratory preparation in which the functional consequences of the hypothesis can be examined. Such tests can be valid only if the computational model is as faithful to the conceptual model as possible. This means that the computational model must be implemented in a way that does not impose additional simplifications or introduce new properties that were not consciously chosen by the user; otherwise how can the user tell whether simulation results truly reflect the properties of the conceptual model, and are not a by−product of distortions produced by trying to implement the model with a computer? This ideal is impossible to meet, and the proper use of any simulator requires judgment by the user as to whether discrepancies between concept and concrete representation are benign or vicious.

A useful simulation environment enables experimental tests of hypotheses by facilitating the construction, use, and revision of computational models that are faithful to the original idea and its subsequent evolution. NEURON is designed to meet this goal, and one of the aims of this book is to show you how to tell whether the model you have in mind is matched by the NEURON simulation you create.

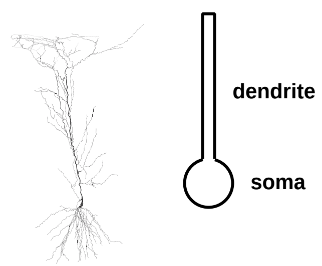
 

## An example

Suppose we are interested in how the cell of Fig. 2.2 A responds to current injected at the soma. We could imagine an enormously complicated conceptual model that attempts to mimic all of the detail of the physical system. But if we're really interested in insight, we might start with a much simpler conceptual model, like the ball and stick shown in Fig. 2.2 B. Most of the anatomical complexity of the physical system lies in the dendritic tree, but our conceptual model approximates the entire dendritic tree by a very simple abstraction: a cylindrical cable.

So going from the physical system to the model involved simplification and abstraction. What about going from the conceptual model to a computational model?

The statements in Fig. 2.2 C specify the topology of the computational model using `hoc`, NEURON's programming language. Note that everything in the conceptual model has a direct counterpart in the computational model, and vice versa: the transition between concept and computational model involves neither simplification nor additional complexity. All that remains is to assign physical dimensions and biophysical properties, and the computational model can be used to generate simulations that reflect the behavior of the conceptual model.




```
create soma, dendrite
connect dendrite(0), soma(1)
```

A B C

Figure 2.2. A. Detailed morphometric reconstruction of Ca1 pyramidal neuron (from D.A. Turner). B. "Ball and stick" conceptual model for studying charging properties of a neuron as seen from the soma. C. The computational implementation of the conceptual model in `hoc`, NEURON's programming language.