

Programming with models: modularity and abstraction provide powerful capabilities for systems biology

Aneil Mallavarapu*, Matthew Thomson, Benjamin Ullian
and Jeremy Gunawardena*

*Department of Systems Biology, Harvard Medical School, 200 Longwood Avenue,
Cambridge, MA 02115, USA*

Mathematical models are increasingly used to understand how phenotypes emerge from systems of molecular interactions. However, their current construction as monolithic sets of equations presents a fundamental barrier to progress. Overcoming this requires modularity, enabling sub-systems to be specified independently and combined incrementally, and abstraction, enabling generic properties of biological processes to be specified independently of specific instances. These, in turn, require models to be represented as programs rather than as data-types. Programmable modularity and abstraction enables libraries of modules to be created, which can be instantiated and reused repeatedly in different contexts with different components. We have developed a computational infrastructure that accomplishes this. We show here why such capabilities are needed, what is required to implement them and what can be accomplished with them that could not be done previously.

Keywords: model building; systems biology; modularity; abstraction

1. INTRODUCTION

Systems biology seeks to understand how physiology emerges from molecular interactions (Ideker *et al.* 2001; Kirschner 2005). Mathematical models are increasingly used to shed light on this (Kitano 2002; Longabaugh *et al.* 2005; Aldridge *et al.* 2006). The construction of such models presents unusual challenges, not previously encountered in physics or engineering, upon which this paper focuses.

Models may be static, as in constraint-based models (Becker *et al.* 2007), or explicitly incorporate time. The latter are typically some form of dynamical system: they specify a set of molecular states and how those states evolve in time and space. Depending on the type of model, the states may be represented in different ways as follows: discrete levels, as in Boolean models or forms of automata (Li *et al.* 2006; Fisher & Henzinger 2007); concentrations, as in ordinary or partial differential equation models (Schoeberl *et al.* 2002; Slepchenko *et al.* 2002); molecular numbers, as in stochastic models (Ramsey *et al.* 2005); or sets of individual molecules, as in agent-based models (Danos & Laneve 2004). Time and space may themselves be treated either continuously or discretely. While much of what follows may be broadly generalized, we focus here on dynamic models

represented by ordinary differential equations,

$$\frac{dx}{dt} = f(\mathbf{x}; \mathbf{a}), \quad (1.1)$$

where $\mathbf{x} \in \mathbb{R}^n$ is a state vector of species concentrations; $\mathbf{a} \in \mathbb{R}^m$ is a vector of parameter values; and f expresses the balance between the rates of production and consumption of each species. In such models, time is continuous and space, if it is represented explicitly at all, is discretized as a finite set of cellular compartments. Such models are predominantly nonlinear. Although they may occasionally be analysed mathematically (Gunawardena 2005), they are usually simulated numerically, for which parameter values must be specified and initial conditions chosen.

Our motivation for the work described here comes from wanting to integrate model building into an experimental research programme. We seek to use models to reason rigorously about biological assumptions and thereby to guide understanding and experimental strategy. The resulting models can be very high dimensional in both states and parameters and a central concern in the field is how such complex models can be used when many of the parameter values are unknown. There are many perspectives on this problem: some properties of systems can be proved to be independent of parameter values (Feinberg 1995; Angeli *et al.* 2004); parameters can be estimated from data in statistically meaningful ways (Jaqaman & Danuser 2006); methods of dimensionality reduction can reduce complexity (Barbano *et al.* 2007); some properties of systems have been found empirically to be robust to parameter

*Authors for correspondence (aneilbaboo@gmail.com; jeremy@hms.harvard.edu).

Electronic supplementary material is available at <http://dx.doi.org/10.1098/rsif.2008.0205> or via <http://journals.royalsociety.org>.

127 variation (Alon *et al.* 1999; von Dassow *et al.* 2000); and
 128 both theory and empirical results suggest that, for any
 129 given phenotypic behaviour, only an exponentially small
 130 number of parameters are significant (Rand *et al.* 2005;
 131 Gutenkunst *et al.* 2007). We note further that both
 132 pharmaceutical and biotechnology companies are adopt-
 133 ing modelling in drug development (Bangs & Paterson
 134 2003; Hendriks *et al.* 2006; Haberichter *et al.* 2007),
 135 suggesting that model complexity is not a barrier to
 136 usefulness. Methodologies for building complex models
 137 in a flexible and controlled manner therefore become all
 138 the more important. We assume in this paper that the
 139 parameter problem can be addressed and focus on
 140 the problem of building models which can formalize
 141 biological assumptions about molecules and cells.

142 A variety of modelling tools are available (Arkin 2001;
 143 Ramsey *et al.* 2005), standards and ontologies formu-
 144 lated (Hucka *et al.* 2003; Novère *et al.* 2005) and public
 145 model repositories established (Novère *et al.* 2006). With
 146 the right information to hand, it is straightforward to
 147 build a model. However, if model building is to be inte-
 148 grated into an experimental research programme, a Q3
 149 single monolithic model is never sufficient. Simply to
 150 understand how a model works, it is usually essential
 151 to create it incrementally, adding a few ingredients at a
 152 time and exploring the effect of alternative assumptions.
 153 More importantly, feedback between experiments and
 154 models leads to corrections or new assumptions, links
 155 appear to systems studied by others and new knowledge
 156 continually emerges in the laboratory and in the broader
 157 scientific community. Furthermore, experiments may be
 158 carried out in a variety of cell types or mutational back-
 159 grounds, in which the configuration of the system under
 160 study may vary. Such differences are readily accommo-
 161 dated in the informal mental ‘models’ maintained by all
 162 biologists. Mathematical models lack such plasticity.
 163 Even simple biological changes can have profound effects
 164 on the equations, requiring new equations to be intro-
 165 duced or the modification of many existing equations and
 166 many terms in each. Beyond a certain level of model
 167 complexity, it is easier to build a new monolithic model
 168 from scratch.

169 For example, Huang & Ferrell (1996) constructed an
 170 influential model of the MAP kinase cascade, which
 171 shed light on the decision-making underlying matu-
 172 ration of *Xenopus* oocytes. Levchenko *et al.* (2000)
 173 subsequently elucidated the surprising effect of a
 174 scaffold protein on MAP kinase signalling. The second
 175 model contains exactly the same MAP kinase cascade
 176 as the first, based on identical assumptions, and differs
 177 only in the addition of one new component, the scaffold.
 178 Nevertheless, it was not obtained from the former by
 179 incremental extension; a new monolithic set of
 180 equations was constructed. At present, anyone wishing
 181 to build upon these prior contributions would have to
 182 do the same. The need to reinvent the wheel each time
 183 is a fundamental barrier to progress.

184 It also makes it difficult for model building to scale as
 185 systems become increasingly complex. Notwithstanding
 186 this complexity, much of molecular biology is built from
 187 general processes that operate on different components
 188 in broadly similar ways. For instance, the scaffold in the
 189 MAP kinase cascade behaves in much the same way as

190 all scaffold proteins: it has no intrinsic enzymatic
 191 function but binds other proteins (Morrison & Davis
 192 2003). Scaffolds may differ in the number of binding
 193 partners and in the behaviour of partners when bound
 194 but there is a core mechanism that must be incorporated
 195 in any model in which a scaffold participates. Model
 196 construction would become far easier if this core
 197 mechanism could be described once (by scaffolding
 198 experts, say) and this description reused repeatedly by
 199 instantiating it with the particular binding partners and
 200 binding assumptions that are relevant to the context
 201 being modelled. Many other molecular processes are
 202 generic, in the sense that the same core mechanism is
 203 used with different components in different contexts. For
 204 example, all receptor tyrosine kinase signalling
 205 pathways are built from the following generic processes:
 206 receptor dimerization; endo- and exocytosis; endosomal
 207 recycling; multisite post-translational modification
 208 (including phosphorylation and ubiquitination); scaf-
 209 folding; GTPase switching; MAP kinase cascades;
 210 membrane localization; nuclear import and export; etc.
 211 Q3 Model construction would be revolutionized if models
 212 were built in a modular and incremental fashion from a
 213 library of expert descriptions of such generic processes.
 214 Model building would then begin to scale with increasing
 215 biological complexity and models could build upon each
 216 other, greatly increasing their scope and credibility.

217 To accomplish such an attractive transformation
 218 requires new capabilities. Among them are modularity,
 219 which allows sub-systems to be specified independently
 220 and composed together, and abstraction, which allows
 221 generic properties of components or sub-systems to be
 222 specified independently of specific instances. These
 223 capabilities are familiar from engineering, where they
 224 are used to design and build artificial systems of
 225 significantly greater complexity than any current
 226 model in systems biology. However, their application
 227 to biological models is not straightforward. In particular,
 228 they require models to be treated as high-level programs
 229 within a computational infrastructure, leading to a new
 230 style of model building. Although the significance of
 231 these capabilities has been acknowledged, as will be
 232 reviewed in the Discussion, none of the available
 233 modelling tools provide programmable modularity and
 234 abstraction. We have developed a computational infra-
 235 structure that does. It is open source and freely available
 236 (see §2). Here, our purpose is not to describe it in detail
 237 but rather to show why it is necessary, what is
 238 challenging about implementing modularity and
 239 abstraction and what can be accomplished with them
 240 that could not be done previously. We hope to show how
 241 these capabilities bring us closer to the transformation of
 242 the model building described above.

Q4 2. METHOD

2.1. Computational infrastructure

248 *Little b* was developed within the LispWorks environ-
 249 ment (LispWorks Ltd, Cambridge, UK). It is freely
 250 and publicly available from vcp.med.harvard.edu,
 251 littleb.org or sourceforge.net. The computational infra-
 252 structure compiles a biological description expressed as

253 a *little b* program into MATLAB (The MathWorks,
 254 Natick, MA) files. Rate equations are either automati-
 255 cally derived using mass-action assumptions or the
 256 user can provide phenomenological rate functions (for
 257 instance, the Hill functions used in the segment polarity
 258 models). The symbolic mathematics sub-system
 259 Q5 (figure 2) can accommodate rational functions of
 260 several variables, with arbitrary real exponents.
 261 Dimensions and units are consistently handled. The
 262 correctness of the infrastructure was tested by the
 263 construction of a series of examples of increasing
 264 complexity, including four previously developed models
 265 (Huang & Ferrell 1996; Bhalla & Iyengar 1999;
 266 Levchenko et al. 2000; von Dassow et al. 2000). In
 267 addition to reproducing MATLAB results, the equations
 268 and their internal representations were checked. Lisp
 269 evaluation times range from 0.2 s for the multisite
 270 phosphorylation model in figure 3d to 11 min for the
 271 segment polarity model of the *Drosophila* lattice, which
 272 has 104 cells, 3439 species and 13 328 reactions.
 273 (Timings on an IBM T43p laptop, Pentium M,
 274 2.1 GHz, 1 Gb RAM). The latest release of *little b*
 275 incorporates the graph-based syntax pioneered in the
 276 BioNetGen system.

277 2.2. Segment polarization

280 Lattices were generated in MATLAB by choosing a set of
 281 points and using a Voronoi tessellation to produce
 282 polygonal cells. For the *Drosophila* lattice, the points
 283 were selected manually as the centres of the biological
 284 cells in the embryo photograph, as in figure 6a.
 285 A MATLAB script generates the (x, y) coordinates of
 286 the vertices of each cell, along with the cell areas and
 287 the lengths of the apposed membrane segments. The
 288 generic cellular lattice module then uses these data to
 289 construct the resulting compartments and membranes.
 290 Figure 6b shows the four lattices used for this study, on
 291 which are superimposed the pre-pattern (initial condi-
 292 tion) of cells in which the levels of Wingless mRNA
 293 (wg) and protein (WG) and Engrailed mRNA (en) and
 294 protein (EN) are set to normalized concentrations of 1,
 295 as previously (von Dassow et al. 2000). All other
 296 components are initially zero, with the exception of the
 297 basal activator of *cid* expression, which has normalized
 298 concentration of 0.4 in each cell, as previously (von
 299 Dassow et al. 2000). The four lattices are: *Hexagonal*,
 300 corresponding to the regular hexagonal lattice used
 301 previously (von Dassow et al. 2000); *Drosophila*,
 302 extracted from the embryo photograph; *Rectangular*,
 303 in which the cells are rectangular but come in two sizes,
 304 arranged in alternating columns; *Shifted rectangular*, in
 305 which the lattice is identical to the rectangular lattice
 306 but the pre-pattern is shifted to the right.

307 We used identical assumptions to von Dassow et al
 308 to represent the regulatory network in figure 5a,
 309 forgoing later modifications (Ingolia 2004). We did
 310 not wrap lattices onto a torus (von Dassow et al. 2000),
 311 as such double periodicity makes no sense for irregular
 312 lattices. We checked for edge effects by embedding one
 313 lattice inside a larger one; we found no evidence for
 314 major changes in behaviour. We chose two previously
 315 used parameter sets (von Dassow et al. 2000) but found

that the high Hill coefficients gave rise to unphysiolo-
 gical oscillations in some components (electronic
 supplementary material, figure 2). We were able to
 find lower Hill coefficients without jeopardizing correct
 segmentation on the hexagonal lattice. The parameter
 values used in figure 6c were derived in this way from
 the Yippee parameter set, while those in figure 6d were
 derived from parameter set four, as detailed in the
 electronic supplementary material, table 1. In running
 the simulations, we found occasional slow decays
 beyond the 1000 minutes used previously (von Dassow
 et al. 2000; electronic supplementary material, figure 2).
 We therefore scored correct segmentation by
 simulating for 5000 simulated minutes, thresholding
 Wingless and Engrailed values as previously (von
 Dassow et al. 2000), and checking if the results agreed
 with the pre-pattern. Edge cells were ignored in scoring.

3. RESULTS

3.1. Modularity and abstraction

Modularity is a fundamental method for building
 complex engineering systems. Modules are sub-systems
 that are encapsulated to hide their internal complexity
 and inter-module communication is only permitted
 through specified interfaces (figure 1a). By subdividing
 the design problem, modularity allows the engineer to
 subdue complexity. (Modularity is also used to describe
 a property of biological systems that may have been
 selected for by evolution (Hartwell et al. 1999). No such
 meaning is intended here; modularity for us is always a
 method of model construction.)

The kind of ‘engineering modularity’ just described
 was introduced into biological modelling in the
 ProMoT tool (Ginkel et al. 2003) and aspects of it are
 provided in other methodologies (Lloyd et al. 2004;
 Webb & White 2005). However, encapsulating a
 module and specifying its interface, at the time it is
 designed, restrict the module’s interactions to situ-
 ations envisaged at design time. Biological modules
 have no natural encapsulation other than membranes.
 In the absence of such physical separation, the com-
 ponents of one module may, in principle, always interact
 biochemically with the components of another. More-
 over, biochemical interaction can create entirely new
 entities. For instance, figure 1b illustrates a situation in
 which module 1 contains *A* but neither *E* nor *B*, while
 module 2 contains *E* but neither *A* nor *B*. Suppose now
 that *E* is an enzyme that converts *A* to *B*. This
 information may have been present in the system when
 modules 1 and 2 were composed but not necessarily as
 part of either module 1 or module 2. It could also have
 been supplied subsequently, as a result, perhaps, of new
 experimental data. In either case, once that information
 is present, the composition of modules 1 and 2 implies
 that *E* has access to its substrate. Hence, *B* should also be
 present, despite the fact that it was not previously
 present in either module. Such new entities need to be
 identified and the corresponding mathematical variables
 introduced into the dynamical system. In engineering
 modularity, this process is effectively carried out by
 the designer at design time. Modularity without

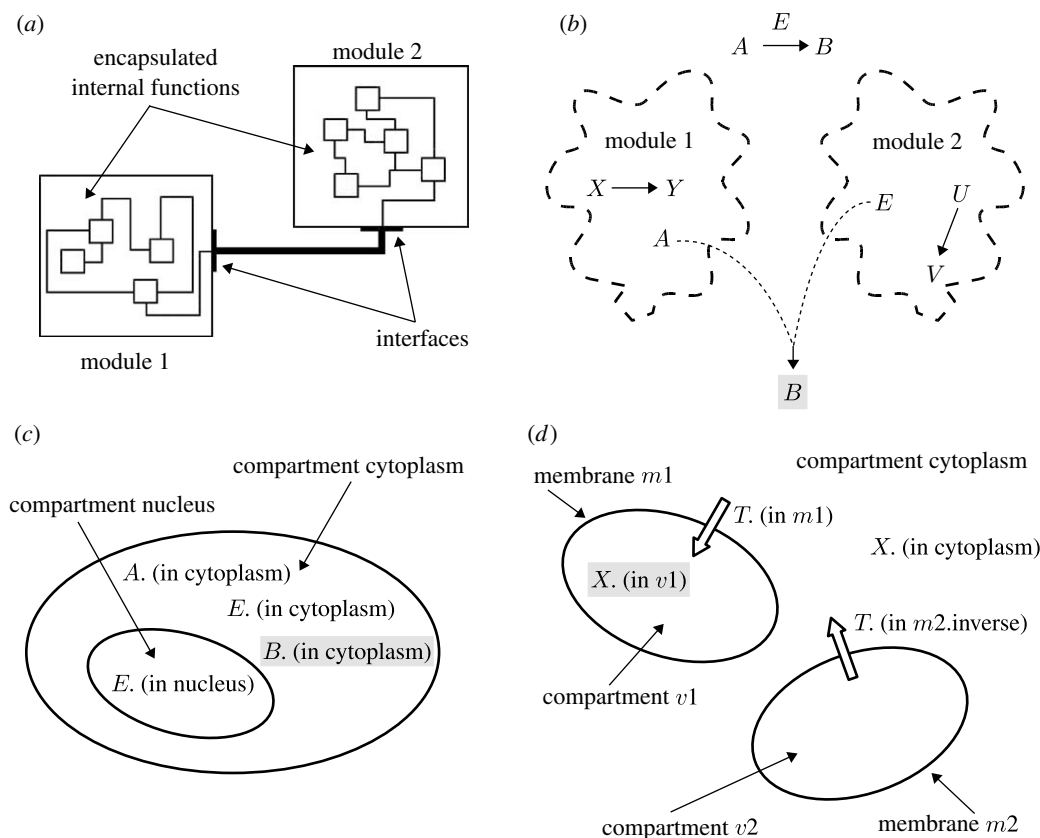


Figure 1. Modularity, inference and identity. (a) Engineering modularity exposes restricted functionality through interfaces, while hiding internal complexity behind barriers. (b) Modularity for biological models must allow for the possibility that any molecule may interact with any other molecule. E is an enzyme that converts A to B . Module 1 contains A but not E or B while module 2 contains E but not A or B . Composing modules 1 and 2 results in a new species, B , not previously present in either module. A module must work correctly in contexts determined by other modules whose characteristics are not known in advance of module composition. *Little b's* computational infrastructure uses reasoning to infer the presence of the highlighted entities. (c) Unique identities must encode information about location. E converts A to B and is present in both the *cytoplasm* compartment and the *nucleus* compartment. However, A is only present in the *cytoplasm*. The system should infer that B is present in the *cytoplasm* but, in the absence of other information, should not infer that it is present in the *nucleus*. (d) Membranes are complex locations. T transports X unidirectionally across a membrane. T is oriented in membrane $m1$ of vesicle $v1$ to transport X into $v1$ but is oppositely oriented in the membrane $m2$ of vesicle $v2$. If X is present in the *cytoplasm* compartment then the computational infrastructure should infer that it is in $v1$ but, in the absence of other information, should not infer that it is in $v2$. In *little b*, membranes encode information about their two adjacent volume compartments and molecules are oriented by locating them in either the standard (default) membrane or its *inverse*. The behaviour of T and X can be described once but then works correctly irrespective of T 's location.

encapsulation requires a computational infrastructure capable of reasoning over the information provided by the user and working out which new entities are required.

The creation of new entities leads to two further difficulties. Firstly, it may have knock-on consequences in other modules, requiring yet more entities to be created, and these may in turn have further consequences and so on. A similar issue arises in reasoning systems used in artificial intelligence, where, for instance, pattern-action rules, of the form $P \Rightarrow A$, specify actions (A) to be performed—including modification, removal or creation of entities—whenever there are entities matching the corresponding set of patterns (P). The execution of a rule can then trigger the execution of further rules. The widely used RETE algorithm (Forgy 1982) provides an efficient way of controlling the resulting chain of actions and ensuring that it terminates in a consistent way.

The second problem is that an entity that needs to be created may already exist in the system. If so, it is essential that a duplicate is not to be created. This may

be achieved by giving each entity a unique identity. However, this identity cannot simply be a random tag. Figure 1c illustrates a situation in which enzyme E is present in both the cytoplasm and the nucleus of a cell, while its substrate, A , is also present but only in the cytoplasm. Biochemistry tells us that the enzymatic product B should be present in the cytoplasm but not in the nucleus. The computational infrastructure needs to be able to distinguish between A being present somewhere and A being present in the same location as the enzyme for which it acts as a substrate. Hence, identity needs to encode location. To describe the enzymatic conversion of A to B by E , it is then sufficient to say that A and E must have the same location, which is then inherited by the product B . Note that this is a simple instance of abstraction: the properties of a reaction are described independently of where it takes place. Whether the product is present or not is only determined once the reaction and its components are instantiated in some specific location. Such a facility

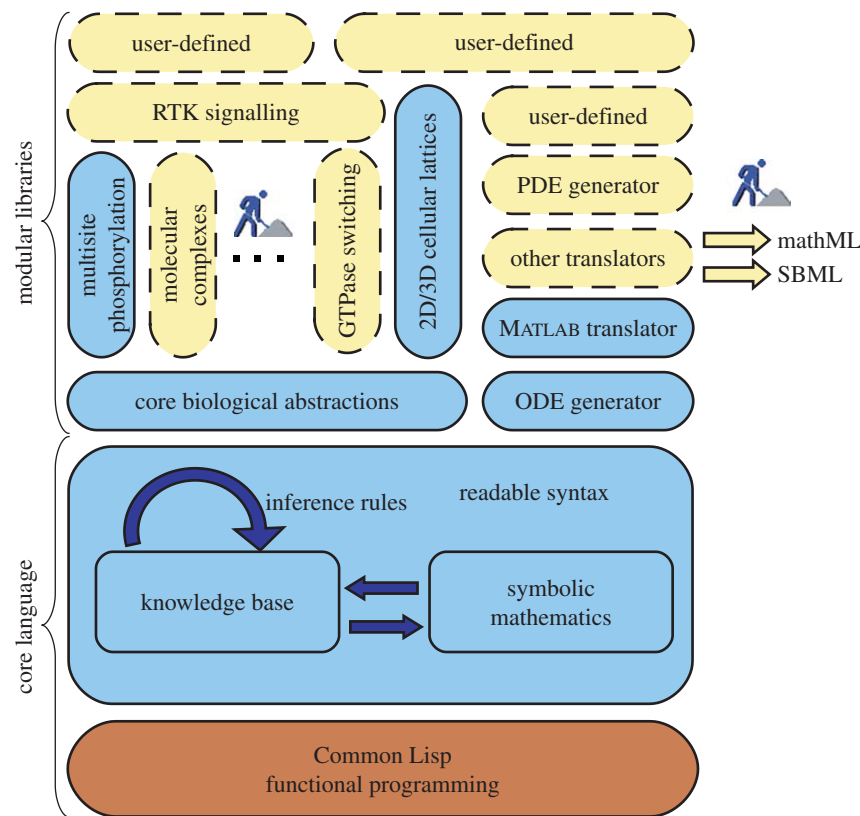


Figure 2. *Little b* provides an extensible architecture, permitting the development of new generic modules. The core language extends Common Lisp with new syntax, a reasoning system and symbolic mathematics. Modular libraries provide both biological and mathematical abstractions in a hierarchical fashion. A library of core biological abstractions defines reusable constructs for representing and reasoning about reactions, molecular complexes and biochemical locations. Higher order modules, such as ‘multisite phosphorylation’ and ‘two-dimensional (2D)/three-dimensional (3D) cellular lattices’ discussed in the text, can be programmed on the top of the core abstractions. Users have access to all levels of the hierarchy and can build new modules that extend the biological or mathematical capabilities. Yellow dashed boxes indicate libraries that are envisaged or under development, while pink full boxes show the currently implemented *little b* computational infrastructure.

becomes invaluable when there are many locations, as in a multicellular tissue, as will be seen in §3.3. Transmembrane reactions involve several locations (figure 1d) and their behaviour may depend on how the corresponding molecules are oriented within the membrane. Locations, therefore, have to be treated in general as structured entities, which describe the local hierarchy of membranes and compartments. Controlling the relationship between identities and locations is one of the crucial requirements for implementing modularity.

These considerations make it clear that modularity without encapsulation requires some form of programming, in order to infer which new entities need to be created and to generate and compare their identities. Programming also allows further levels of abstraction to be implemented, as will be seen in §3.2. We have designed a high-level programming language, *little b*, for this purpose. Biological information specified in a program is evaluated by the *little b* computational infrastructure and translated into equations expressed in MATLAB (figure 2). *Little b* is implemented as a macro language of Common Lisp (Graham 1996). Lisp is used in several biological applications (Krummenacker et al. 2005; Massar et al. 2005). We chose it for the following reasons: Common Lisp is open source; an open source implementation of the RETE algorithm is

available in the Lisp-based Intelligent Software Agents (LISA) system; Lisp metaprogramming allows *little b* to present a readable, biologically meaningful, non-Lisp-like syntax to the user (figure 3d); Lisp’s symbolic processing capability provides powerful mechanisms for implementing modularity and abstraction (Sinclair & Moon 1991). The resulting capabilities for model building are explored in the two sections that follow.

3.2. Multisite phosphorylation as a generic library module

Phosphorylation and dephosphorylation on serine, threonine and tyrosine are ubiquitous and vital cellular regulatory mechanisms (Cohen 2001). One of their most striking features is the remarkable extent of multisite modification. The mammalian tumour suppressor p53 has at least 16 sites known to be functionally significant (Holmberg et al. 2002) while the microtubule-associated protein tau can become hyperphosphorylated on 39–45 sites in Alzheimer’s disease (Hanger et al. 2007). This makes multisite phosphorylation a prime candidate for a generic library module, which abstracts current biological understanding and can be reused repeatedly to make different models in different contexts. The module discussed here

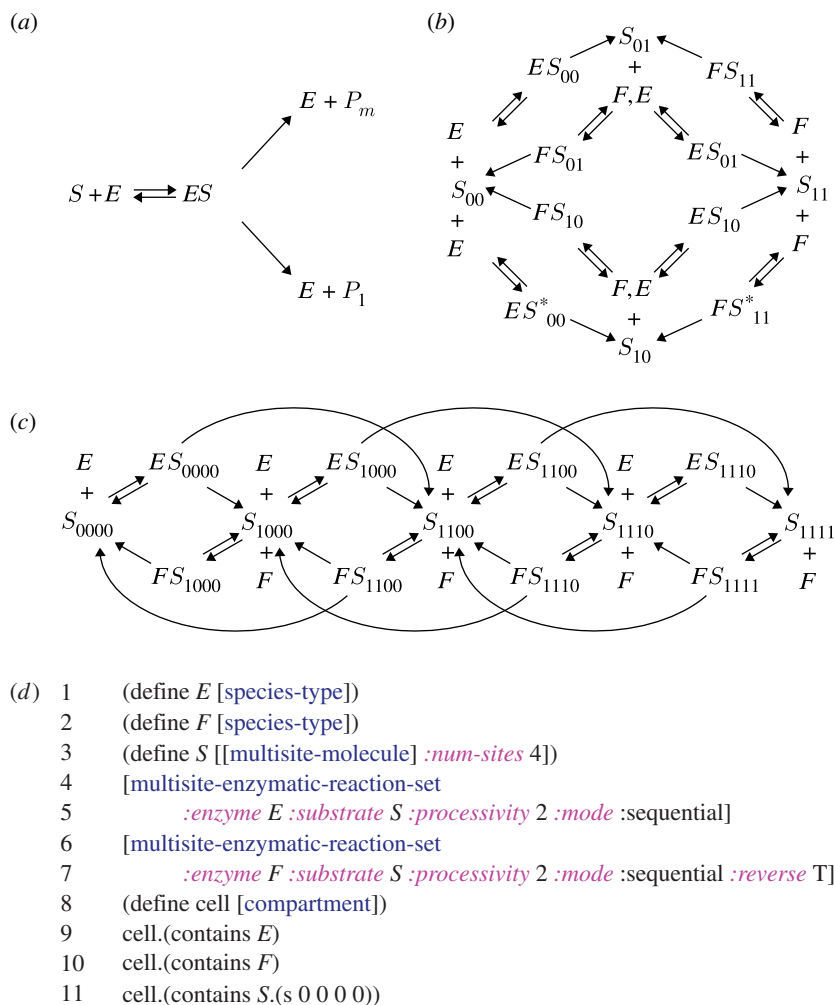


Figure 3. Reaction schemes for multisite phosphorylation. (a) The biochemical scheme for phosphorylation and dephosphorylation assumes a single enzyme-substrate complex and irreversible release of products, P_1, \dots, P_m . Distributivity corresponds to $m=1$, processivity to $m>1$. ATP, ADP and phosphate are assumed held constant and the kinetics are given by mass action. (b) Distributive, non-sequential phosphorylation and dephosphorylation with $n=2$ sites. Phospho forms are denoted by S_b , where b is a sequence of n bits indicating the presence or absence (1 or 0, respectively) of phosphate. E kinase, F phosphatase and S substrate. (c) Processive, sequential phosphorylation and dephosphorylation with $n=4$ and processivity degree 2. Enzyme release steps are omitted for clarity. (d) Example *little b* program using the generic module (lines 4/5 and 6/7) for multisite phosphorylation, which can generate any reaction scheme like (b, c). Bold blue text, object classes; bold italic pink text, keywords. The module is instantiated for (c) but can be instantiated for (b) by changing the number of sites to 2 (line 3), the processivity to 1 and the mode to non-sequential (lines 5/7). Model equations are generated after rate constants and initial conditions are specified (not shown). Despite appearances, *little b* code is Lisp code and is interpreted by the Lisp read-eval-print mechanism in the usual way. *Little b* uses reader macros to alter Lisp's default syntax to one more suited for describing biology. Lisp allows a new language to be created within itself, a capability that we find particularly useful for biology.

has the capability to generate most of the models previously used in studies of multisite phosphorylation. We treat phosphorylation and dephosphorylation together, in keeping with a systems approach.

The current understanding of the mechanisms of multisite phosphorylation and dephosphorylation is typically complex. First, kinases and phosphatases may differ in the number of modifications made in a single encounter between enzyme molecule and substrate molecule. If at most a single modification is made, the enzyme will be distributive; if more than one is sometimes made, it will be processive (Huang & Ferrell 1996). The degree of processivity is the maximal number of modifications in a single encounter. Examples of both distributivity and processivity are known. Mek phosphorylation of Erk on its two activating sites is

distributive (Burack & Sturgill 1997; Ferrell & Bhatt 1997), as is MKP3 dephosphorylation of Erk on the same sites (Zhao & Zhang 2001). The p38 MAP kinase distributively phosphorylates ATF2 on two sites (Waes et al. 2001) while Fcp1 distributively dephosphorylates three out of four sites on a tetra-heptad repeat segments of the RNA polymerase II carboxy-terminal domain (Hausmann et al. 2004). The phosphorylation of Pho4 on four sites by Pho80-Pho85 is processive with an average of 2.1 phosphorylations per molecular encounter (Jeffrey et al. 2001). The phosphorylation of the focal adhesion protein p130Cas by Src on 15 tyrosines is highly processive (Pellicena & Miller 2001). The alternative splicing factor ASF/SF2 is an SR protein, containing multiple serine/arginine repeat sequences, which is phosphorylated by the SR-specific protein kinase

SRPK1 on 8–9 sites and by Clk/Sty on 20 sites, in both cases in a highly processive manner (Aubol *et al.* 2003; Velazquez-Dones *et al.* 2005). Degrees of processivity have not been measured for either p130Cas or ASF/SF2. Processive phosphatases have not been identified but phosphatases have generally been less well studied than kinases.

A second aspect of multisite phosphorylation is that some kinases are known to be sequential, phosphorylating in a strict order of sites. For instance, GSK3, in its ‘primed’ phosphorylation mode (Roach 1991; Fiol & Roach 1996), phosphorylates (S/T)XXX(S/T) repeat motifs in a strict C-to-N order (Harwood 2001). The fibroblast growth factor receptor also autophosphorylates on five tyrosines in a specific order (Furdui *et al.* 2006). While these examples suggest that sequential phosphatases may also exist, none have yet been identified. Notwithstanding this, most models assume that both enzymes are sequential and that the phosphatase works in the reverse order to the kinase. In this case, only $n+1$ phospho forms are produced in contrast to the 2^n phospho forms present when the enzymes show no preference for site order.

A wide variety of networks of reactions can arise from these assumptions (figure 3*b,c*). To generate models from such networks, it is necessary to either impose an *ad hoc* rate function, such as Michaelis–Menten, or to make further assumptions about the biochemistry of phosphorylation and dephosphorylation and then calculate the rate function using mass-action. Either method can be programmed in *little b* but Michaelis–Menten is, at best, a dubious approximation (Ciliberto *et al.* 2007) and we will illustrate instead the use of mass action. Figure 3*a* shows a generic biochemical scheme that allows for enzyme processivity. S is a substrate phospho form and E an enzyme. They reversibly associate to form an enzyme–substrate complex, ES, which irreversibly disassociates to release E and multiple product phospho forms, P_1, \dots, P_m . Such a scheme can cover both kinases and phosphatases, provided ATP, ADP and phosphate are assumed to be held at constant concentrations by some mechanism that is not explicitly modelled. This is usually true *in vivo* and is assumed in all models in the literature. The effects of ATP, ADP and phosphate are then effectively absorbed into the mass-action rate constants. The scheme in figure 3*a* applies to each relevant phospho form, for both kinase and phosphatase.

A model can now be generated by using mass action to calculate the rates of production and consumption of each chemical species. Writing down such a model by hand for any given network of reactions is laborious and error prone. It quickly becomes infeasible as the number of sites increases. The program in figure 3*d* uses a generic *little b* module to generate models for a broad range of different assumptions, for any specified number of phosphorylation sites. It accommodates the following use cases: kinase and phosphatase are both sequential but operate in opposite order (figure 3*c*); kinase and phosphatase have no preference for site order (figure 3*b*); and either enzyme is processive to varying degree (figure 3*c* has degree 2 for both enzymes), with distributivity corresponding to a processivity degree

of 1 (figure 3*b*). These different cases can be selected by merely setting the values of the corresponding attributes in the module. This covers most of the mass-action models known to us in the literature. Note that, although the enzyme–substrate complexes are part of the model, the user does not need to specify them independently. *Little b* creates them automatically according to the biochemical scheme in figure 3*a* and takes them into account when calculating the rate equations.

We used the program in figure 3*d* to study the decision-making capacity of multisite phosphorylation. A single substrate molecule with n sites can, in principle, encode 2^n different states. The state of a population of such molecules, however, is described by a frequency distribution giving the relative stoichiometry of each phospho form. The decision-making capacity of this phospho-form distribution is quite different from that of a single molecule. It is determined not by the molecular structure but by the dynamics of interaction between the substrate and its cognate kinases and phosphatases. We have proved in recent work (Thomson & Gunawardena submitted) that, when the enzymes are distributive, there may be as many as $\lfloor (n+2)/2 \rfloor$ stable distributions of phospho forms at steady state ($\lfloor x \rfloor$ being the greatest integer not greater than x). This suggests that increasing numbers of sites can support increasingly complex decision-making. To reveal this multistability experimentally, we sought different initial conditions that would lead to different stable phospho-form distributions. Although the steady states of such models can be analysed mathematically, the dynamics leading to them can only be studied by simulation, which needs to be undertaken for different numbers of sites and for sequential as well as non-sequential systems. The generic module in figure 3*d* makes this straightforward. Figure 4 shows two scenarios in which each of the stable phospho-form distributions can be reached by starting with some mixture of unphosphorylated and fully phosphorylated substrate. This behaviour was the representative of the multistable systems we studied. These predictions from simulation suggest a simple method for detecting multistability experimentally, which we are now testing in the laboratory.

The multisite phosphorylation module shows that powerful abstractions for describing generic biological processes can be programmed from the core biological abstractions in *little b*. The value of a programmatic approach is further illustrated by considering how alternative assumptions can be dealt with. For example, there are situations in which it is helpful to explicitly model ATP. A number of important drugs are kinase inhibitors which compete for the ATP-binding pocket (Cohen 1999). If such drug effects are to be studied, the scheme in figure 3*a* has to be modified accordingly. (In fact, it must be altered in two different ways because the kinase uses ATP and produces ADP while the phosphatase only produces phosphate.) While these are entirely natural changes from a biological perspective, the recalculation of the rate functions is so awkward to undertake manually that all models would have to be reconstructed from scratch. Using the programmatic approach, it is only necessary to modify the generic

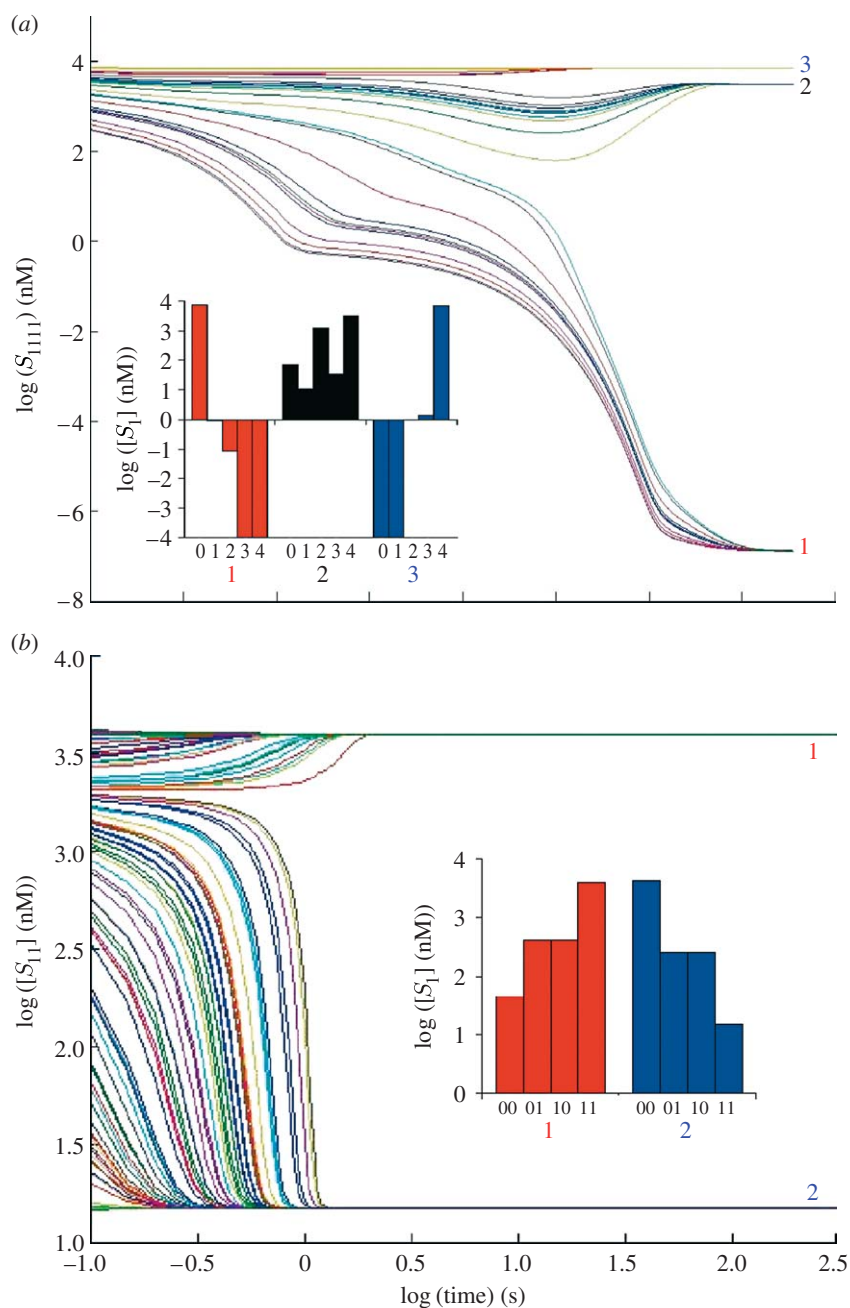


Figure 4. Multistability in multisite phosphorylation. Rate constants are given in figure 1 in the electronic supplementary material. (a) Distributive, sequential phosphorylation and dephosphorylation, with $n=4$, as figure 3c but with $k=1$. Substrate is initially present as $[S_{0000}] = \alpha[S_{\text{tot}}]$, $[S_{1111}] = (1-\alpha)[S_{\text{tot}}]$, where α is drawn randomly from the uniform distribution on $[0,1]$ and $[S_{\text{tot}}]$ is the total amount of substrate present. $[-]$ denotes concentration. Vertical axis, concentration of S_{1111} ; horizontal axis, time; log scales on both. The initial conditions find the three stable phospho-form distributions shown in the inset, for appropriate values of α . In the inset, phospho forms are designated 0, 1, 2, 3, 4 by number of phosphorylations. (b) Distributive, non-sequential phosphorylation and dephosphorylation with $n=2$, as figure 3b. Initial substrate is a random combination of S_{00} and S_{11} , as previously, leading to the two stable phospho-form distributions shown in the inset. Vertical axis, concentration of S_{11} ; horizontal axis, time; log scales on both.

module to operate with the new schemes in preference to that in figure 3a. The corresponding models can then be generated with no greater difficulty than before.

3.3. Developmental patterning on realistic cellular lattices

During initial patterning of the *Drosophila* embryo, maternal mRNAs stimulate expression of gap genes, followed by pair rule genes, followed by segment polarity

genes, thereby establishing the anterior–posterior pattern of parasegments (Lawrence 1992). The early stages of this process take place in the syncytial blastoderm but the segment polarity genes turn on after cellularization. von Dassow *et al.* created a computational infrastructure, Ingeneue (Meir *et al.* 2002), for building models of gene regulatory networks in a two-dimensional lattice of regular hexagonal cells and used it to build a model of *Drosophila* segment polarization (von Dassow *et al.* 2000). The proposed segment polarity network (figure 5b)

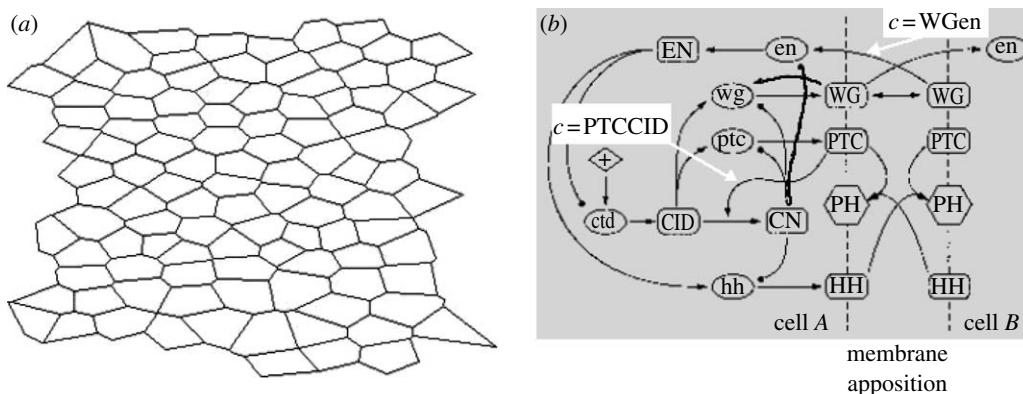


Figure 5. Modular construction of developmental networks in arbitrary cellular lattices. (a) Polygonal lattice of cells. The user provides the vertex coordinates to *little b*'s generic lattice module, which creates an internal representation of the lattice. (b) The segment polarity gene regulation network after von Dassow *et al.* (2000). The positive feedback of Wingless protein (WG) on its mRNA (*wg*) and the repression of Engrailed mRNA (*en*) by cleaved Cubitus Interruptus (CN) are both included. Labels show the interactions ($c = WGen$ and $c = PTCCID$) that are varied in figure 6. *Little b* can take any polygonal lattice and any network of reactions and put the two together in a modular way. Each cell acquires a copy of the regulatory network and two adjacent cells interact across their common membrane segment using the same mechanism as in von Dassow *et al.* (2000). The bookkeeping scheme required to identify each species in each location is automatically worked out by *little b* and used to build the the model equations.

was able to correctly stabilize a pre-pattern of Wingless and Engrailed expression over a wide range of different parameter values, suggesting that such robustness might be an evolutionary criterion for selecting network designs. This idea stimulated much interest (Wolpert 2001).

In reality, *Drosophila* embryo epithelia are not regular hexagonal lattices of cells (figure 6a). Cells may be of different sizes and shapes and have different numbers of neighbours. Indeed, in proliferating animal epithelia, approximately 29% of the cells resemble pentagons, 46% resemble hexagons and 21% resemble heptagons (Gibson *et al.* 2006). The segment polarity network should produce the right patterning irrespective of which lattice emerges from cellularization. Robustness to lattice variation is a more stringent requirement than robustness to parameter variation. The latter involves a mere numerical change in parameters without any change to the structure of the underlying equations; the former involves a restructuring of the equations themselves because the pattern and rates of cell–cell communication are altered. Robustness to lattice variation may thus exert greater selective pressure on network designs than does robustness to parameter variation.

Ingeneue provides computational support for building reactions in regular hexagonal cells, with cell–cell communication being modelled by interaction across apposed membrane segments. It keeps track of each apposed pair of membrane segments and all the molecular interactions across each apposition. The bookkeeping required is substantial but straightforward for a regular hexagonal lattice and such functionality is hard-wired into Ingeneue. Each irregular lattice, however, needs its own bookkeeping scheme and the equations need to be rewritten to reflect each new scheme. Although the robustness to lattice variation seems a relevant and important question, Ingeneue has no capability to address it, nor does any existing tool.

We implemented a generic cellular lattice module in *little b* for two-dimensional polygonal (or three-dimensional polyhedral) cells. The lattice module reads

a list of coordinates of the vertices of the lattice, supplied by the user, and creates an internal representation of the cellular lattice. This module may be composed with any regulatory or protein interaction network module, using the same membrane apposition assumption for cell–cell communication as used in Ingeneue (figure 5b). *Little b* builds the resulting model automatically. The bookkeeping scheme appropriate to the chosen lattice emerges automatically from modular composition, the location abstraction (see §3.1) instantiates all the species and reactions in each cell of the lattice and the intercellular links are wired together according to the membrane appositions. Programmable modularity and abstraction enables the modelling of any gene regulatory network in any lattice of polygonal cells, subsuming and greatly extending the currently available functionality for multicellular modelling.

We used this module to examine the behaviour of figure 5 in four lattices, including a physiologically realistic lattice extracted from a *Drosophila* embryo image (figure 6a,b). The parameter robustness observed for the regular hexagonal lattice arises from a combination of ultrasensitive Hill functions and feedback (von Dassow & Odell 2002; Ingolia 2004). We chose two previously defined sets of parameter values (von Dassow *et al.* 2000), as described in §2, and varied in one set the Hill function controlling an intercellular activation and in the other set an intracellular negative feedback loop (both labelled in figure 5b). The intercellular activation showed a range of parameter values for which all four lattices produced correct patterning (figure 6c), although the range was substantially smaller for the hexagonal lattice alone. However, the intracellular negative feedback showed no parameter values for which all four lattices work correctly (figure 6d). Indeed, the *Drosophila* lattice never produces the correct pattern, despite a substantial range in which the regular hexagonal lattice works. The regulatory network in figure 5b seems to be highly sensitive to lattice geometry.

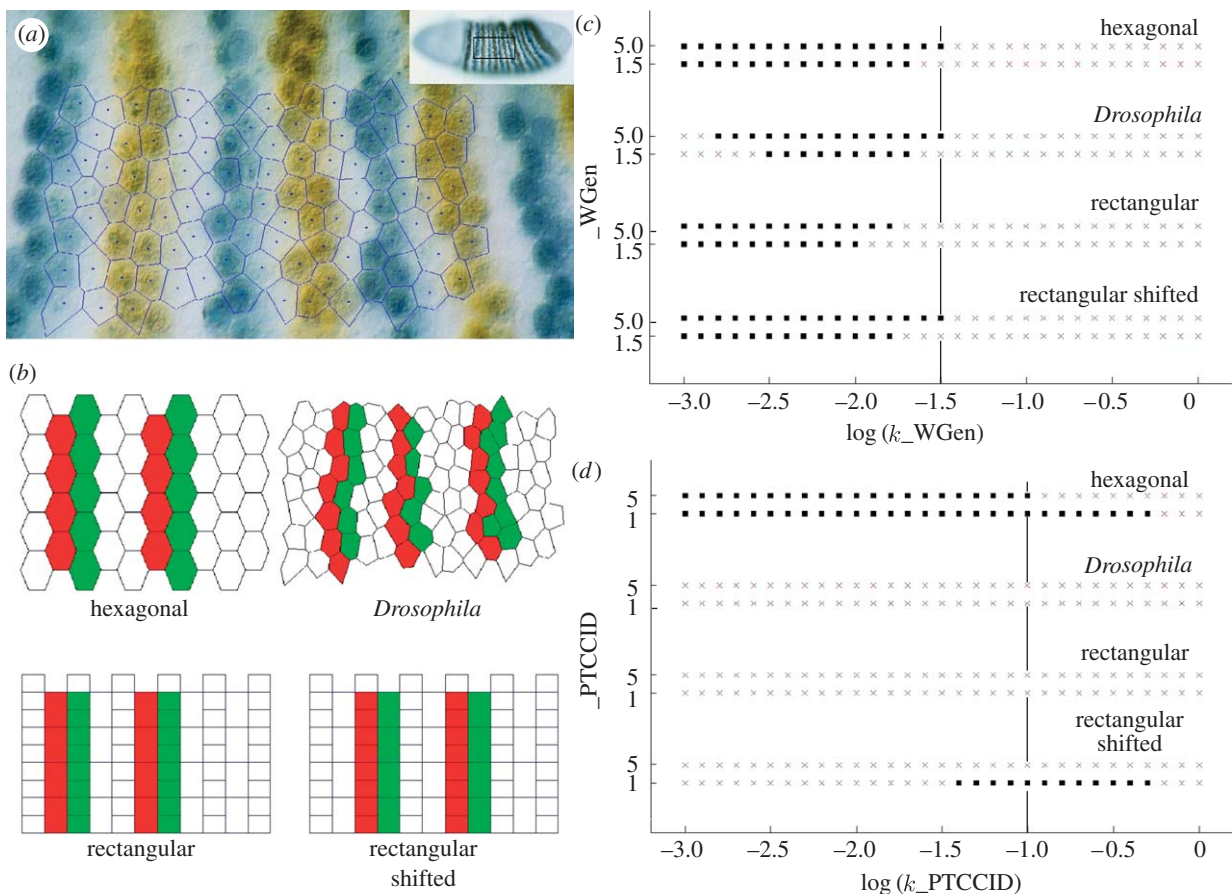


Figure 6. Segment polarization in different cellular lattices. (a) Image of a *Drosophila* embryo, with the extracted cellular lattice superimposed. (b) Four lattices showing the pre-pattern where Wingless (red) and Engrailed (green) are high. For correct segmentation, the regulatory network must stabilize this pre-pattern, starting from the pre-pattern as initial condition. (c,d) Correct (filled square) or incorrect (cross) segmentation for the lattices listed on the right. The half-maximal value, k_c (horizontal axis), and the Hill coefficient, v_c (vertical axis), of a Hill function, $x^{v-c}/((k_c)^{v-c} + x^{v-c})$, describing one of the connections c in figure 5b, are varied. The half-maximal value varies horizontally on a log scale, while the Hill coefficient takes either a low (1 or 1.5) or high (5) value. The parameter values other than v_c and k_c are obtained from two previously defined parameter sets (von Dassow et al. 2000), as described in §2, and are listed in the electronic supplementary material, table 1. (c) Intercellular transcriptional activation of Engrailed by Wingless (c =WGen). (d) Intracellular cleavage of Cubitus Interruptus by Patched (c =PTCCID).

It is conceivable that a relatively small change in the regulatory network will generate robust patterning for realistic lattices. Alternatively, significant control loops may be missing from our current understanding of this system. The point we wish to make in this paper is that *little b* allows the exploration of these scientifically interesting questions, which were entirely out of reach previously.

4. DISCUSSION

The multisite phosphorylation example shows that a generic biological process can be abstracted into the succinct module used in figure 3d, while the *Drosophila* segmentation example shows the power of modular composition. These capabilities lay the foundation for the vision of model building articulated in the Introduction. This needs to be set in the context of currently available tools and methodologies for model building.

Many excellent ones are available. Some focus on particular biological processes, such as gene regulation (Ramsey et al. 2005), some favour-specific biological

domains, such as immunology (Meier-Schellersheim et al. 2006) or neuroscience (Bower & Beeman 1998; Carnevale & Hines 2006), and some provide specialized capabilities for simulation (Stiles & Bartol 2001; Lok & Brent 2005; Meier-Schellersheim et al. 2006) or analysis (Ermentrout 2002).

What makes *little b* different from existing tools is its programming language for modularity and abstraction. A language allows its user to describe novel situations not previously envisaged by the language designer. Most tools support model building through some form of ‘template’, accessible through a menu of available templates. Hierarchical templates can be an efficient means of providing access to the kinds of features discussed here. However, while such templates are implemented in a programming language, this language is not itself accessible to the user, who cannot define new templates. For instance, while users of Ingeneue (§3.3) are restricted to the template for a regular, hexagonal cellular lattice, *little b*’s programmability allowed us to define a polygonal cell in terms of pre-existing definitions of locations, compartments and

1261 membranes, and define a polygonal lattice in terms of
 1262 polygonal cells. These hierarchically defined com-
 1263 ponents could have been devised by any users of the
 1264 language. Indeed, further studies in developmental
 1265 patterning or cellular physiology might be aided by
 1266 more detailed abstractions describing tissues having
 1267 particular arrangements of cell types with distinct
 1268 morphologies—for example, a generic neuron param-
 1269 etrized by axon length, dendrite number, etc., or a
 1270 stratified epithelium parametrized by cell types, cell
 1271 numbers or other properties. These too can be imple-
 1272 mented by individual users as required and made
 1273 available to all other users as generic modules. The
 1274 programmatic approach supports a decentralized
 1275 approach, not just to model building, but also to create
 1276 the abstractions and generic modules that allow model
 1277 building to scale with increasing biological complexity.

1278 The Systems Biology Markup Language (SBML)
 1279 marked a watershed in biological modelling (Hucka
 1280 et al. 2003). Although nominally a language, it is an
 1281 Extensible Markup Language, not a programming
 1282 language. SBML treats a model as a datatype, not as
 1283 a program. It provides the syntax for describing the
 1284 model's components, thereby enabling a model con-
 1285 structed by one tool to be read and understood by
 1286 another. This has helped to nucleate the model-building
 1287 community and create a de facto standard for model
 1288 curation (Novère et al. 2005, 2006). Software that
 1289 is SBML compliant can provide facilities for analysing
 1290 SBML models and, in principle, for composing them.
 1291 However, generic abstractions, such as the tissue
 1292 abstractions suggested above, would have to be incor-
 1293 porated within the ontology provided by SBML. This
 1294 ontology is, in effect, agreed upon by the SBML
 1295 community and cannot be altered by any one user; it
 1296 has to be updated by the developers with each release of
 1297 SBML. As we have seen, *little b* allows the user to create
 1298 and share new abstractions at will in a decentralized
 1299 manner. The two approaches may become comp-
 1300 lementary: once novel abstractions have been intro-
 1301 duced and rigorously tested within a programming
 1302 language community, they can be codified in an updated
 1303 ontology and thereby made accessible to a broader
 1304 biological community, who need not necessarily be
 1305 conversant with the programming language itself.

1306 The BioNetGen system introduced an elegant graph-
 1307 based syntax for specifying molecular species in terms
 1308 of their interaction domains (Blinov et al. 2004) and
 1309 similar ideas were independently developed in other
 1310 systems (Danos & Laneve 2004; Lok & Brent 2005;
 1311 Meier-Schellersheim et al. 2006). Biochemical reactions
 1312 can then be specified as graph-rewriting rules, leading
 1313 to rule-based languages for model building (Hlavacek
 1314 et al. 2006). Sets of rules may be composed, providing
 1315 a modular approach to model building, as well as
 1316 access to new analytical capabilities (Danos et al. 2007,
 1317 2008). However, these languages are low level, akin to
 1318 computer assembly languages, and lack the control and
 1319 data structures commonplace in high-level program-
 1320 ming languages. In the context of multisite phosphoryl-
 1321 ation (§3.2), for instance, it would be necessary to
 1322 provide different rules for a substrate with one site, two
 1323 sites, three sites, etc. The functionality of figure 3d, in

1324 which the number of sites is a parameter, could only be
 1325 implemented within the rule-based language by first
 1326 creating the machinery for high-level programming.
 1327 While this might be possible in principle (since rule-
 1328 based languages are usually Turing powerful), the effort
 1329 required would be prohibitive. Rule-based languages
 1330 are naturally complementary to *little b*: the former
 1331 provide a low-level syntax for molecular complexes
 1332 while the latter provides higher level abstractions.

1333 In contrast to rule-based languages, Cellerator
 1334 (Shapiro et al. 2003) is a model building tool that
 1335 uses MATHEMATICA to assemble models. The function-
 1336 ality of figure 3d could certainly be implemented in
 1337 MATHEMATICA or other high-level programming
 1338 languages such as Perl or C, either on their own or in
 1339 conjunction with a rule-based language. However, this
 1340 would solve only a single isolated problem—that of
 1341 multisite phosphorylation on varying numbers of sites.
 1342 This solution could not be composed with others in a
 1343 modular way, as the lattice module of §3.3 can be
 1344 composed with any regulatory network module. To do
 1345 this would require also implementing the core function-
 1346 ality of *little b* (identity, pattern-action rules, etc.)
 1347 within a unified linguistic framework.

1348 Graphical languages have also been proposed for
 1349 biological specification and can provide synoptic
 1350 summaries of much complex biological information
 1351 (Kitano et al. 2005; Kohn et al. 2006). Experience in
 1352 engineering design suggests that while they are
 1353 excellent for describing structure, such as the layout
 1354 of an integrated circuit, function is best described
 1355 through textual languages, as in VERILOG or VHDL.
 1356 Accordingly, we anticipate that tools based on graphi-
 1357 cal languages will come to take advantage of textual
 1358 languages like *little b* ‘under the hood’.

1359 We have focused in this paper on the transition from
 1360 monolithic to modular models and on the compu-
 1361 tational infrastructure needed to support this. Not only
 1362 is this required to build models more effectively, but it is
 1363 also essential for their credibility. Models, particularly
 1364 complex ones, are usually published as supplementary
 1365 information. Even the most conscientious reviewer is
 1366 unlikely to be able to subject such a model to the same
 1367 level of scrutiny as a published experimental method or
 1368 mathematical proof. Models may sometimes be sub-
 1369 mitted to a public repository but few others are likely to
 1370 want to use an existing model without also wanting to
 1371 change it, with all the attendant difficulties noted
 1372 previously. Accordingly, monolithic models may have
 1373 been closely studied only by their creators, a situation
 1374 of some concern in an emerging discipline. Modular
 1375 models, by contrast, can be pulled apart and their
 1376 component modules evaluated, modified and recom-
 1377 bined. Generic library modules, such as that for
 1378 multisite phosphorylation, could be developed and
 1379 refined by experts and made available to all model
 1380 builders, thereby creating a scientific ‘commons’ for
 1381 model building. *Little b*'s extensible open source
 1382 architecture allows all users to develop novel abstrac-
 1383 tions of their particular biological domain and contrib-
 1384 ute these back to the community, allowing the field
 1385 to evolve in a decentralized manner and enabling us to
 1386 build upon each other's work rather than recreate it.

1387 The models that result from this may be more complex
 1388 but their credibility, reliability and usefulness will
 1389 be more easily established. Programmable model
 1390 building will provide a more robust foundation for
 1391 systems biology.

1392 We are grateful to the Bauer Center for Genomics Research
 1393 and to Craig Muir for support during the initial phase of this
 1394 work. We thank, especially, Steve Harrison, Ed Harlow, Marc
 1395 Kirschner and Rebecca Ward of the Harvard Medical School
 1396 for enabling this work to come to fruition through their
 1397 support for the Virtual Cell Program. We thank Peter
 1398 Lawrence for the *Drosophila* image in figure 6a, Radhika
 1399 Nagpal for access to material in press, David Young for his
 1400 open source implementation of LISA and Dave Fox of
 1401 LispWorks for his assistance with the Lisp environment. We
 1402 thank Carl Pabo, Brian Seed and Rebecca Ward for their
 1403 insightful comments and the other members of the Virtual Cell
 1404 Program for their assistance. The authors declare that they
 1405 have no competing financial interests. Correspondence should
 1406 be addressed to J.G. (email: jeremy@hms.harvard.edu) and
 1407 A.M. (email: aneilbaboo@gmail.com).

1409 REFERENCES

1410 Aldridge, B. B., Burke, J. M., Lauffenburger, D. A. & Sorger,
 1411 P. K. 2006 Physicochemical modelling of cell signalling
 1412 pathways. *Nat. Cell Biol.* **8**, 1195–1203. (doi:10.1038/
 1413 ncb1497)
 1414 Alon, U., Surette, M. G., Barkai, N. & Leibler, S. 1999
 1415 Robustness in bacterial chemotaxis. *Nature* **307**, 168–171.
 1416 (doi:10.1038/307168a0)
 1417 Angeli, D., Ferrell, J. E. & Sontag, E. D. 2004 Detection of
 1418 multistability, bifurcations, and hysteresis in a large class
 1419 of biological positive-feedback systems. *Proc. Natl Acad.*
 1420 *Sci. USA* **101**, 1822–1827. (doi:10.1073/pnas.0308265100)
 1421 Arkin, A. P. 2001 Synthetic cell biology. *Curr. Opin.*
 1422 *Biotechnol.* **12**, 638–644. (doi:10.1016/S0958-1669(01)
 1423 00273-7)
 1424 Aubol, B. E., Chakrabarti, S., Ngo, J., Shaffer, J., Nolen, B.,
 1425 Fu, X.-D., Ghosh, G. & Adams, J. A. 2003 Processive
 1426 phosphorylation of alternative splicing factor/splicing
 1427 factor 2. *Proc. Natl Acad. Sci. USA* **100**, 12 601–12 606.
 1428 (doi:10.1073/pnas.1635129100)
 1429 Bangs, A. L. & Paterson, T. S. 2003 Finding value in *in silico*
 1430 biology. *Biosilico* **1**, 18–22. (doi:10.1016/S1478-5382(03)
 1431 02218-2)
 1432 Barbano, P. E., Spivak, M., Flajolet, M., Nairn, A. C.,
 1433 Greengard, P. & Greengard, L. 2007 A mathematical tool
 1434 for exploring the dynamics of biological networks. *Proc.*
 1435 *Natl Acad. Sci. USA* **104**, 19 169–19 174. (doi:10.1073/
 1436 pnas.0709955104)
 1437 Becker, S. A., Feist, A. M., Mo, M. L., Hannum, G., Palsson,
 1438 B. O. & Herrgard, M. J. 2007 Quantitative prediction of
 1439 cellular metabolism with constraint-based models: the
 1440 COBRA Toolbox. *Nat. Protoc.* **2**, 727–738. (doi:10.1038/
 1441 nprot.2007.99)
 1442 Bhalla, U. & Iyengar, R. 1999 Emergent properties of
 1443 networks of biological signalling pathways. *Science* **283**,
 1444 381–387. (doi:10.1126/science.283.5400.381)
 1445 Blinov, M. L., Faeder, J. R., Goldstein, B. & Hlavacek, W. S.
 1446 2004 BioNETGEN: software for rule-based modeling of
 1447 signal transduction based on the interactions of molecular
 1448 domains. *Bioinformatics* **20**, 3289–3291. (doi:10.1093/
 1449 bioinformatics/bth378)
 1450 Bower, J. M. & Beeman, D. 1998 *The book of GENESIS:
 1451 exploring realistic neural models with the GEneral NEural
 1452 Simulation System*. New York, NY: Springer.

Burack, W. R. & Sturgill, T. W. 1997 The activating dual
 phosphorylation of MAPK by MEK is nonprocessive. *Biochemistry* **36**, 5929–5933. (doi:10.1021/bi970535d)
 Carnevale, N. T. & Hines, M. L. 2006 *The NEURON book*.
 Cambridge, UK: Cambridge University Press.
 Ciliberto, A., Capuani, F. & Tyson, J. J. 2007 Modeling
 networks of coupled enzymatic reactions using the total
 quasi-steady state approximation. *PLoS Comput. Biol.* **3**,
 e45. (doi:10.1371/journal.pcbi.0030045)
 Cohen, P. 1999 The development and therapeutic potential of
 protein kinase inhibitors. *Curr. Opin. Chem. Biol.* **3**,
 459–465. (doi:10.1016/S1367-5931(99)80067-2)
 Cohen, P. 2001 The role of reversible protein phosphorylation
 in health and disease. *Eur. J. Biochem.* **268**, 5001–5010.
 (doi:10.1046/j.0014-2956.2001.02473.x)
 Danos, V. & Laneve, C. 2004 Formal molecular biology.
Theor. Comput. Sci. **325**, 69–110. (doi:10.1016/j.tcs.2004.
 03.065)
 Danos, V., Feret, J., Fontana, W. & Krivine, J. 2007 Scalable
 simulation of cellular signalling networks. In *Proc. APLAS
 2007*. Lecture Notes in Computer Science, vol. 4807.
 Berlin, Germany: Springer.
 Danos, V., Feret, J., Fontana, W. & Krivine, J. 2008 Abstract
 interpretation of cellular signalling networks. In *Proc.
 VMCAI 2008*. Lecture Notes in Computer Science,
 vol. 4905. Berlin, Germany: Springer.
 Ermentrout, B. 2002 *Simulating, analyzing, and animating
 dynamical systems: a guide to Xppaut for researchers and
 students*. Philadelphia, PA: SIAM.
 Feinberg, M. 1995 The existence and uniqueness of steady
 states for a class of chemical reaction networks. *Arch. Rat.
 Mech. Anal.* **132**, 311–370. (doi:10.1007/BF00375614)
 Ferrell, J. E. & Bhatt, R. R. 1997 Mechanistic studies of the
 dual phosphorylation of mitogen-activated protein kinase.
J. Biol. Chem. **272**, 19 008–19 016. (doi:10.1074/jbc.272.
 30.19008)
 Fiol, C. J. & Roach, P. J. 1996 Hierarchical phosphorylation of
 proteins. In *Protein phosphorylation* (ed. F. Marks). Federal
 Republic of Germany, Germany: VCH Verlagsgesellschaft.
 Fisher, J. & Henzinger, T. 2007 Executable cell biology. *Nat.*
Biotechnol. **25**, 1239–1249. (doi:10.1038/nbt1356)
 Forgey, C. L. 1982 Rete: a fast algorithm for the many
 pattern/many object pattern match problem. *Artif. Intell.*
19, 17–37. (doi:10.1016/0004-3702(82)90020-0)
 Furdui, C. M., Lew, E. D., Schlessinger, J. & Anderson, K. S.
 2006 Autophosphorylation of FGFR1 kinase is mediated
 by a sequential and precisely ordered reaction. *Mol. Cell*
21, 711–717. (doi:10.1016/j.molcel.2006.01.022)
 Gibson, M. C., Patel, A. B., Nagpal, R. & Perrimon, N. 2006
 The emergence of geometric order in proliferating
 metazoan epithelia. *Nature* **442**, 1038–1041. (doi:10.
 1038/nature05014)
 Ginkel, A., Kremling, A., Nutsch, T., Rehner, R. & Gilles,
 E. D. 2003 Modular modelling of cellular systems with
 ProMot/Divi. *Bioinformatics* **19**, 1169–1176. (doi:10.
 1093/bioinformatics/btg128)
 Graham, P. 1996 *ANSI common lisp Ser. Artif. Intel.* New
 Jersey, NJ: Prentice Hall.
 Gunawardena, J. 2005 Multisite protein phosphorylation
 makes a good threshold but can be a poor switch. *Proc.*
Natl Acad. Sci. USA **102**, 14 617–14 622. (doi:10.1073/
 pnas.0507322102)
 Gutenkunst, R. N., Waterfall, J. J., Casey, F. P., Brown,
 K. S., Myers, C. R. & Sethna, J. P. 2007 Universally sloppy
 parameter sensitivities in systems biology models. *PLoS
 Comput. Biol.* **3**, 1871–1878. (doi:10.1371/journal.pcbi.
 0030189)

- 1513 Haberichter, T. *et al.* 2007 A systems biology dynamical
1514 model of mammalian G1 cell cycle progression. *Mol. Syst.*
1515 *Biol.* **3**, 84. (doi:10.1038/msb4100126) 1578
- 1516 Hanger, D. P., Byers, H. L., Wray, S., Leung, K. Y., Saxton,
1517 M. J., Seereeram, A., Reynolds, C. H., Ward, M. A. &
1518 Anderton, B. H. 2007 Novel phosphorylation sites in tau
1519 from Alzheimer brain support a role for casein kinase 1 in
1520 disease pathogenesis. *J. Biol. Chem.* **282**, 23 645–23 654.
1521 (doi:10.1074/jbc.M703269200) 1581
- 1522 Hartwell, L. H., Hopfield, J. J., Leibler, S. & Murray, A. W.
1523 1999 From molecular to modular cell biology. *Nature* **402**,
1524 C47–C52. (doi:10.1038/35011540) 1582
- 1525 Harwood, A. J. 2001 Regulation of GSK-3: a cellular
1526 multiprocessor. *Cell* **105**, 821–824. (doi:10.1016/S0092-
1527 8674(01)00412-3) 1583
- 1528 Hausmann, S., Erdjument-Bromage, H. & Shuman, S. 2004
1529 Schizosaccharomyces pombe carboxyl-terminal domain
1530 (CTD) phosphatase Fcp1: distributive mechanism, mini-
1531 mal CTD substrate and active site mapping. *J. Biol.*
1532 *Chem.* **209**, 10 892–10 900. (doi:10.1074/jbc.M312513200) 1584
- 1533 Hendriks, B. S. *et al.* 2006 Decreased internalisation of ErbB1
1534 mutants in lung cancer is linked with a mechanism
1535 conferring sensitivity to gefitinib. *IEE Proc. Syst. Biol.*
1536 **153**, 457–466. (doi:10.1049/ip-syb:20050108) 1585
- 1537 Hlavacek, W. S., Faeder, J. R., Blinov, M. L., Posner, R. G.,
1538 Hucka, M. & Fontana, W. 2006 Rules for modeling signal-
1539 transduction systems. *Sci. STKE* **344**, re6. (doi:10.1126/
1540 stke.3442006re6) 1586
- 1541 Holmberg, C. I., Tran, S. E. F., Eriksson, J. E. & Sistonen, L.
1542 2002 Multisite phosphorylation provides sophisticated
1543 regulation of transcription factors. *Trends Biochem. Sci.*
1544 **27**, 619–627. (doi:10.1016/S0968-0004(02)02207-7) 1587
- 1545 Huang, C.-Y. F. & Ferrell, J. E. 1996 Ultrasensitivity in the
1546 mitogen-activated protein kinase cascade. *Proc. Natl*
1547 *Acad. Sci. USA* **93**, 10 078–10 083. (doi:10.1073/pnas.93.
1548 19.10078) 1588
- 1549 Hucka, M., Finney, A., Bolouri, H., Doyle, J. C. & Kitano, H.
1550 2003 The Systems Biology Markup Language (SBML): a
1551 medium for representation and exchange of biochemical
1552 network models. *Bioinformatics* **19**, 524–531. (doi:10.
1553 1093/bioinformatics/btg015) 1589
- 1554 Ideker, T., Galitski, T. & Hood, L. 2001 A new approach to
1555 decoding life: systems biology. *Annu. Rev. Genom. Hum.*
1556 *Genet.* **2**, 343–372. (doi:10.1146/annurev.genom.2.1.343) 1590
- 1557 Ingolia, N. T. 2004 Topology and robustness in the *Drosophila*
1558 segment polarity network. *PLoS Biol.* **2**, 0805–0815.
1559 (doi:10.1371/journal.pbio.0020123) 1591
- 1560 Jaqaman, K. & Danuser, G. 2006 Linking data to models:
1561 data regression. *Nat. Rev. Mol. Cell Biol.* **7**, 813–819.
1562 (doi:10.1038/nrm2030) 1592
- 1563 Jeffrey, D. A., Springer, M., King, D. S. & O’Shea, E. K. 2001
1564 Multi-site phosphorylation of Pho4 by the cyclin-CDK
1565 Pho80–Pho85 is semi-processive with site preference.
1566 *J. Mol. Biol.* **306**, 997–1010. (doi:10.1006/jmbi.2000.
1567 4417) 1593
- 1568 Kirschner, M. 2005 The meaning of systems biology. *Cell* **121**,
1569 503–504. (doi:10.1016/j.cell.2005.05.005) 1594
- 1570 Kitano, H. 2002 Computational systems biology. *Nature* **420**,
1571 206–210. (doi:10.1038/nature01254) 1595
- 1572 Kitano, H., Funahashi, A., Matsuoka, Y. & Oda, K. 2005
1573 Using process diagrams for the graphical representation of
1574 biological networks. *Nat. Biotechnol.* **23**, 961–966. (doi:10.
1575 1038/nbt1111) 1596
- 1576 Kohn, K. W., Aladjem, M. I., Weinstein, J. N. & Pommier, Y.
1577 2006 Molecular interaction maps of bioregulatory net-
1578 works: a general rubric for systems biology. *Mol. Biol. Cell*
1579 **17**, 1–13. (doi:10.1091/mbc.E05-09-0824) 1597
- 1580 Krummenacker, M., Paley, S., Mueller, L., Yan, T. & Karp,
1581 P. D. 2005 Querying and computing with BioCyc
1582 databases. *Bioinformatics* **21**, 3454–3455. (doi:10.1093/
1583 bioinformatics/bti546) 1598
- 1584 Lawrence, P. A. 1992 *The making of a fly*. Oxford, UK:
1585 Blackwell Science. 1599
- 1586 Levchenko, A., Bruck, J. & Sternberg, P. W. 2000 Scaffold
1587 proteins may biphasically affect the levels of mitogen-
1588 activated protein kinase signaling and reduce its threshold
1589 properties. *Proc. Natl Acad. Sci. USA* **97**, 5818–5823.
1590 (doi:10.1073/pnas.97.11.5818) 1600
- 1591 Li, S., Assmann, S. M. & Albert, R. 2006 Predicting essential
1592 components of signal transduction networks: a dynamic
1593 model of guard cell abscissic acid signaling. *PLoS Biol.* **4**,
1594 e312. (doi:10.1371/journal.pbio.0040312) 1601
- 1595 Lloyd, C. M., Halstead, M. D. B. & Nielsen, P. F. 2004
1596 CellML: its future, present and past. *Prog. Biophys. Mol.*
1597 *Biol.* **85**, 433–450. (doi:10.1016/j.pbiomolbio.2004.01.004) 1602
- 1598 Lok, L. & Brent, R. 2005 Automatic generation of cellular
1599 reaction networks with MOLEculizer 1.0. *Nat. Biotechnol.*
1600 **23**, 131–136. (doi:10.1038/nbt1054) 1603
- 1601 Longabaugh, W. J. R., Davidson, E. H. & Bolouri, H. 2005
1602 Computational representation of developmental genetic
1603 regulatory networks. *Dev. Biol.* **283**, 1–16. (doi:10.1016/
1604 j.ydbio.2005.04.023) 1604
- 1605 Massar, J. P., Travers, M., Elhai, J. & Shrager, J. 2005
1606 BIOLINGUA: a programmable knowledge environment for
1607 biologists. *Bioinformatics* **21**, 199–207. (doi:10.1093/bioin-
1608 formatics/bth465) 1605
- 1609 Meier-Schellersheim, M., Xu, X., Angermann, B., Kunkel,
1610 E. J., Jin, T. & Germain, R. N. 2006 Key role of local
1611 regulation in chemosensing revealed by a new molecular
1612 interaction-based modeling method. *PLoS Comput. Biol.* **2**,
1613 0710–0724. (doi:10.1371/journal.pcbi.0020082) 1606
- 1614 Meir, E., Mumro, E. M., Odell, G. M. & von Dassow, G. 2002
1615 INGENEUE: a versatile tool for reconstituting genetic
1616 networks, with examples from the segment polarity
1617 network. *J. Exp. Zool. Part B* **294**, 216–251. (doi:10.
1618 1002/jez.10187) 1607
- 1619 Morrison, D. K. & Davis, R. J. 2003 Regulation of MAP
1620 kinase signalling modules by scaffold proteins in mammals.
1621 *Annu. Rev. Cell Dev. Biol.* **19**, 91–118. (doi:10.1146/
1622 annurev.cellbio.19.111401.091942) 1608
- 1623 Novère, N. L. *et al.* 2005 Minimum information requested in
1624 the annotation of biochemical models (MIRIAM). *Nat.*
1625 *Biotechnol.* **23**, 1509–1515. (doi:10.1038/nbt1156) 1609
- 1626 Novère, N. L. *et al.* 2006 Biomodels database: a free,
1627 centralized database of curated, published, quantitative
1628 kinetic models of biochemical and cellular systems. *Nucleic*
1629 *Acids Res.* **34**, D689–D691. (doi:10.1093/nar/gkj092) 1610
- 1630 Pellicena, P. & Miller, W. T. 2001 Processive phosphorylation
1631 of p130Cas by Src depends on SH3-polyproline interac-
1632 tions. *J. Biol. Chem.* **276**, 28 190–28 196. (doi:10.1074/jbc.
1633 M100055200) 1611
- 1634 Ramsey, S., Orrell, D. & Bolouri, H. 2005 DIZZY: stochastic
1635 simulation of large-scale genetic regulatory networks.
1636 *J. Bioinform. Comput. Biol.* **3**, 415–436. (doi:10.1142/
1637 S0219720005001132) 1612
- 1638 Rand, D. A., Shulgin, B. V., Salazar, J. D. & Millar, A. J. 2005
1639 Uncovering the design principles of circadian clocks:
1640 mathematical analysis of flexibility and evolutionary
1641 goals. *J. Theor. Biol.* **238**, 616–635. (doi:10.1016/j.jtbi.
1642 2005.06.026) 1613
- 1643 Roach, P. J. 1991 Multisite and hierarchal protein phos-
1644 phorylation. *J. Biol. Chem.* **266**, 14 139–14 142. 1614
- 1645 Schoeberl, B., Eichler-Jonsson, C., Gilles, E. D. & Müller, G.
1646 2002 Computational modeling of the dynamics of the 1615

1639	MAP kinase cascade activated by surface and internalized	1702
1640	EGF receptors. <i>Nat. Biotechnol.</i> 20 , 370–375. (doi:10.	1703
1641	1038/nbt0402-370)	1704
1642	Shapiro, B. E., Levchenko, A., Meyerowitz, E. M., Wold, B. J.	1705
1643	& Mjolsness, E. D. 2003 Cellerator: extending a computer	1706
1644	algebra system to include biochemical arrows for signal	1707
1645	transduction simulations. <i>Bioinformatics</i> 19 , 677–678.	1708
1646	(doi:10.1093/bioinformatics/btg042)	1709
1647	Sinclair, K. H. & Moon, D. A. 1991 The philosophy of Lisp.	1710
1648	<i>Commun. ACM</i> 34 , 41–47. (doi:10.1145/114669.133119)	1711
1649	Slepchenko, B. M., Schaff, J. C., Carson, J. H. & Loew, L. M.	1712
1650	2002 Computational cell biology: spatiotemporal simulation	1713
1651	of cellular events. <i>Annu. Rev. Biophys. Biomol. Struct.</i> 31 ,	1714
1652	423–441. (doi:10.1146/annurev.biophys.31.101101.140930)	1715
1653	Stiles, J. R. & Bartol, T. M. 2001 Monte Carlo methods for	1716
1654	simulating realistic synaptic microphysiology using MCell.	1717
1655	In <i>Computational neuroscience: realistic modeling for</i>	1718
1656	<i>experimentalists</i> (ed. E. de Schutter), pp. 87–127. Boca	1719
1657	Raton, FL: CRC Press.	1720
1658	Thomson, M. & Gunawardena, J. Submitted. The decision-	1721
1659	making capacity of multisite phosphorylation increases	1722
1660	with the number of sites.	1723
1661	Velazquez-Dones, A., Hagopian, J. C., Ma, C.-T., Zhong, X.-Y.,	1724
1662	Zhou, H., Ghosh, G., Fu, X.-D. & Adams, J. A. 2005 Mass	1725
1663	spectrometric and kinetic analysis of ASF/SF2 phosphoryl-	1726
1664	ation by SRPK1 and Clk/Sty. <i>J. Biol. Chem.</i> 280 ,	1727
1665	41 761–41 768. (doi:10.1074/jbc.M504156200)	1728
1666	von Dassow, G. & Odell, G. M. 2002 Design and constraints	1729
1667	of the <i>Drosophila</i> segment polarity module: robust spatial	1730
1668	patterning emerges from intertwined cell state switches.	1731
1669	<i>J. Exp. Zool. Part B</i> 294 , 179–215. (doi:10.1002/jez.	1732
1670	10144)	1733
1671	von Dassow, G., Meir, E., Munro, E. M. & Odell, G. M. 2000	1734
1672	The segment polarity network is a robust developmental	1735
1673	module. <i>Nature</i> 406 , 188–192. (doi:10.1038/35018085)	1736
1674	Waas, W. F., Lo, H.-H. & Dalby, K. N. 2001 The kinetic	1737
1675	mechanism of the dual phosphorylation of the ATF2	1738
1676	transcription factor by p38 mitogen activated protein	1739
1677	(MAP) kinase α . <i>J. Biol. Chem.</i> 278 , 5676–5684. (doi:10.	1740
1678	1074/jbc.M008787200)	1741
1679	Webb, K. & White, T. 2005 UML as a cell and biochemistry	1742
1680	modelling language. <i>BioSystems</i> 50 , 283–302. (doi:10.	1743
1681	1016/j.biosystems.2004.12.003)	1744
1682	Wolpert, L. 2001 <i>Principles of development</i> . Oxford, UK:	1745
1683	Oxford University Press.	1746
1684	Zhao, Y. & Zhang, Z.-Y. 2001 The mechanism of depho-	1747
1685	sphorylation of extracellular signal-regulated kinase 2 by	1748
1686	mitogen-activated protein kinase phosphatase 3. <i>J. Biol.</i>	1749
1687	<i>Chem.</i> 276 , 32 382–32 391. (doi:10.1074/jbc.M103369200)	1750
1688		1751
1689		1752
1690		1753
1691		1754
1692		1755
1693		1756
1694		1757
1695		1758
1696		1759
1697		1760
1698		1761
1699		1762
1700		1763
1701		1764

1765	Author Queries	1828
1766	<i>JOB NUMBER:</i> 20080205	1829
1767		1830
1768	<i>JOURNAL:</i> RSIF	1831
1769		1832
1770	Q1 We have inserted a short title. Please approve or	1833
1771	provide an alternative.	1834
1772	Q2 Please supply a short description to appear alongside	1835
1773	the electronic supplementary material online.	1836
1774	Q3 Please check the edit if the sentence ‘Model construc-	1837
1775	tion would be revolutionized...’	1838
1776	Q4 We have moved the ‘Methods’ section after ‘Introduc-	1839
1777	tion’ section. Please check and confirm.	1840
1778	Q5 Please note that the figures are not cited in proper	1841
1779	order.	1842
1780	Q6 Please check the edit of the sentence ‘If so, it is	1843
1781	essential...’	1844
1782	Q7 Please check the edit of the sentence ‘Whether or not	1845
1783	the product is present...’	1846
1784	Q8 Please check the edit of the sentence ‘If at most a	1847
1785	single modification...’	1848
1786	Q9 Please check the edit of the sentence ‘The inter-	1849
1787	cellular activation...’	1850
1788		1851
1789	Q10 Please update publication details in Thomson &	1852
1790	Gunawardena (submitted).	1853
1791		1854
1792		1855
1793		1856
1794		1857
1795		1858
1796		1859
1797		1860
1798		1861
1799		1862
1800		1863
1801		1864
1802		1865
1803		1866
1804		1867
1805		1868
1806		1869
1807		1870
1808		1871
1809		1872
1810		1873
1811		1874
1812		1875
1813		1876
1814		1877
1815		1878
1816		1879
1817		1880
1818		1881
1819		1882
1820		1883
1821		1884
1822		1885
1823		1886
1824		1887
1825		1888
1826		1889
1827		1890