A Graphical and Computational Modelling Platform for Biological Pathways

A Graphical and Computational Modelling Platform for Biological Pathways

Alessandra Livigni[1*], Laura O'Hara[1,2*], Marta E. Polak[3,4], Tim Angus[1], Derek Wright[1], Lee B. Smith[2,5] and Tom C. Freeman[1+]

[1]The Roslin Institute and Royal (Dick) School of Veterinary Studies, University of Edinburgh, Easter Bush, Edinburgh, Midlothian EH25 9RG, UK. [2]MRC Centre for Reproductive Health, 47 Little France Crescent, Edinburgh, EH16 4TJ, UK. [3]Clinical and Experimental Sciences, Sir Henry Wellcome Laboratories, Faculty of Medicine, University of Southampton, SO16 6YD, Southampton, [4]Institute for Life Sciences, University of Southampton, SO17 1BJ, UK. [5]Faculty of Science, University of Newcastle, Callaghan, NSW 2308, Australia.

[+]Corresponding author: Tom Freeman (tom.freeman@roslin.ed.ac.uk)

* These authors contributed equally to this work.

EDITORIAL SUMMARY: This is a biologist-friendly modelling scheme facilitating the capture and visualization of knowledge on biological pathways and how components interact. Moreover, when parameterised, these pathway models can be used directly to run simulations of their activity and test hypotheses.

TWEET: A biologist-friendly modelling scheme to visualize and computationally model biological pathways @roslininstitute @mrc_crh

[AU: Please highlight to the editor up to 3 key references from your lab that demonstrate the development/use of the protocol; we hope to highlight 1 or more of these particular references in the 'associated links' box on the online article page].

Please use references

#1: O'Hara et al PMID: 27509052

#14: Raza et al PMID: 20478018

#41: Polak et al PMID: 28386100

Key words: Pathway, notation system, model, dynamic modelling, Petri Net, simulation, SBGN, mEPN

Abstract

A major endeavour of systems biology is the construction of graphical and computational models of biological pathways as a means to better understand their structure and function. Here, we present a protocol for a biologist-friendly graphical modelling scheme which facilitates the construction of detailed network diagrams, summarising the components of a biological pathway (such as proteins, biochemicals etc.) and how they interact. These diagrams can then be used to simulate activity flow through a pathway, thereby modelling its dynamic behaviour. The protocol is divided into four sections: 1) Assembly of network diagrams using the modified Edinburgh Pathway Notation (mEPN) scheme and yEd network editing software using pathway information obtained from published literature and databases of molecular interaction data, 2) parameterisation of the pathway model within yEd through the placement of 'tokens' based on the known or imputed amount or activity of a component, 3) model testing through visualization and quantitative analysis of the movement of tokens through the pathway using network analysis tool BioLayout *Express*^{3D}, 4) optimisation of model parameterisation and experimentation. This is the first modelling approach that combines a sophisticated notation scheme for depicting biological events at the molecular level, with a Petri net-based flow simulation algorithm and powerful visualisation engine with which to observe the dynamics of the system being modelled. Unlike many mathematical approaches to modelling pathways, it does not require the construction of a series of equations or rate constants for model parametrisation. Depending on a model's complexity and the availability of information, its construction can take days to months, and, with refinement, possibly years. However, once assembled and parameterised, a simulation run, even on a large model, typically takes only seconds. Models

constructed using this approach provide a means of knowledge management, information exchange, and through the computation simulation of their dynamic activity, a means to generate and test hypotheses, and predict a system's behaviour when perturbed.

Introduction

The era of molecular biology has resulted in the generation of vast amounts of data on biological processes, ranging from in-depth studies of one or two molecules and their interactions, to large sets of omics data. These data are currently scattered in the literature and databases and difficult to connect together. Those trying to learn about a particular biological process or pathway often start by studying the primary literature and reviews. However, relying only on the medium of the written word it can often be a struggle to understand the available knowledge as a series of interconnected events. The task is made more difficult as the literature often refers to the same pathway component by different names, which may not or not be their official names (as dictated by nomenclature committees). Representation of biological systems as graphical models, i.e. diagrams, can in principle circumvent these issues by presenting a system in a visually intuitive manner using a standardized notation scheme to represent pathway components and the interactions between them. One of the ultimate goals of a pathway model is the ability to use it for computational simulations, thereby supporting hypothesis generation and experimental design. Use of a system that fulfils these criteria could benefit any scientist working in experimental biology.

We have developed a modelling platform that combines elements of other approaches[1]. The modified Edinburgh Pathway Notation (mEPN) scheme was first published in 2008[2], refined in 2010[3], and is presented here in its current form (BOX 1). Representing the interactions between biological components in the context of a pathway diagram is a challenge, and a number of notation schemes have been proposed[3-9]. In an effort to standardise pathway diagrams, the Systems Biology Graphical Notation (SBGN) community proposed standards for pathway depiction, including the process description (PD) language[7] based on ideas first proposed by Kitano *et al.*[4]. In SBGN-PD diagrams (and in the modelling approach described here), components of a pathway are depicted using a standard set of shapes

(glyphs), and both the nature of the interactions between components and the products of those interactions must be shown explicitly. Pathway models are constructed where *entity nodes* represent molecular components, *process nodes* represent the different types of interactions that can occur between the components, and *edges* link entity and process nodes. Since PDs were first described various models have been constructed based on this approach[2,10-15] and there is a growing number of software tools that support model creation (using SBGN-compliant languages), e.g., CellDesigner[16,17], NaviCell[11], KEGG Mapper[18], ReactomeFiViz[19], iPath[20] and SBGN-Ed[21]. There are also a number of centralised databases providing pathway resources of this type[22-25]. The mEPN scheme used here to model pathway systems is similar to the SBGN 'process description language' but with important differences in how both components and events are represented. In particular, mEPN supports the representation of wider variety of biological components and processes, simplifies the depiction of complexes, promotes the use of standard nomenclature, and importantly, diagrams can be used directly for the computational modelling of system dynamics (for a more complete description of mEPN and comparison to the SBGN-PD language, see O'Hara 2016[1]). mEPN pathway models can be drawn using the free graph editing software yEd (yWorks, Tübingen, Germany; www.yworks.com), and since the notation scheme was first described [2,3] has been formalised so as to support the use of models for pathway activity simulations[1].

Numerous mathematical approaches exist to simulate system dynamics including ordinary and partial differential equations, qualitative differential equations and stochastic equations. The Systems Biology Markup Language (SBML) has been developed as an open interchange format for such mathematical models[26] and the SBML site also has an extensive list of existing tools and resources supporting pathway modelling primarily by equation-based approaches (http://sbml.org/SBML_Software_Guide/SBML_Software_Summary). Most mathematical models require experimentally derived rate constants to feed into equations and significant computational power to solve a series of equations. This generally limits equation-based approaches to modelling relatively small and well characterised systems. Moreover, the level of mathematics skills required to construct and run these models is often a deterrent to adoption by biologists. The platform described here uses Petri nets, as the basis for pathway activity simulations. The primary resource for Petri net modelling is a network diagram

consisting of nodes, called ‘places’, and other nodes representing the interactions between them, called ‘transitions’, to which the user need only add ‘tokens’, that represent the amount or activity of a place prior to performing a simulation (for more details of Petri nets see BOX 2). There is a long and established precedent for the use of Petri nets in the modelling of biological pathways [27-33] and several tools and algorithms are available that allow the user to construct models based on Petri nets [34-36]. The Petri net algorithm employed here was first described by Ruths *et al*.[37] who named their approach the signalling Petri net simulator (SPN). It combines elements of a Boolean network simulator[38] with a synchronized Petri net model[39], and models the stochastic flow of tokens through a network. A great advantage of Petri nets is the relative ease of model parameterisation, the scale to which models can be constructed, the computational speed of simulations, as well as the fact that the user does not need to directly modify the maths when experimenting. The downside to most of the tools that currently support pathway modelling using Petri nets is the inability to represent pathway models in anything but the standard Petri net notation (open circles and black rectangles), and limited options for the visualization of results. Here, we describe how a pathway model drawn according to the rules of mEPN, can then be parameterised for computational modelling by the addition of tokens, whose quantity can be based on experimental results such as quantitative transcriptomics or proteomics data. When a mEPN model is imported into the open-source software BioLayout *Express*^{3D} [40] it is visualised in a 3D environment, with nodes now represented as 3D shapes. Simulations can subsequently be performed that calculate the flow of tokens through the pathway over time. Pathway activity can then be visualised as plots or animations, where token accumulation is represented by the size and colour of an entity node (Supplementary Video 1). Altering the simulation parameters can change the flow of tokens, allowing the dynamics of pathways to be modelled under different conditions. The modelling approach described in this protocol can be applied to model any system, large or small, biological or otherwise, that consists of a series of components that interact in a predefined manner. In the case of biological pathways, the mEPN notation scheme allows for the detailed representation of signalling cascades, metabolic pathways, transcriptional networks, as well as feedback/feedforward loops. To date, we have used this approach to model a wide variety of biological pathways, particularly associated with immune signalling, e.g., Toll-like receptors (TLR), NF-kB, complement activation and antigen presentation, but also biochemical pathways e.g. cholesterol

metabolism, TCA cycle, and even pathways spanning multiple organ systems e.g. glucocorticoid, oxytocin/prolactin signalling (see: www.virtuallyimmune.org and O'Hara *et al*.[1] for examples). Many of these models were built as graphical representations of events as described in the literature and as such act as a graphical bibliography, with pathway components or processes hyperlinked to research papers or reports. However, with additional work they can also be used as the basis for performing simulation experiments. These simulations not only test the logic of what is depicted, but also predict the behaviour of the system and its response to perturbation. Using this approach models can be assembled at scale, representing tens or thousands of components and the interactions between them. The overall aim of the modelling approach described here is to, provide a platform for the assembly of information on a particular system into an informative diagram, to allow the use of the diagram explore how the system might operate, and through experimentation, make testable predictions [41 42].

The protocol provided here complements the paper published recently by O'Hara *et al*.[1], which describes the development of and underlying concepts associated with this approach. This modelling platform may not be appropriate in situations where many of the interactions between components are not known due to the requirement to define both components and interactions, or where there is need to use specific rate parameters to regulate the dynamics of a system, as the approach does not allow for the integration of rate constants for specific reactions. Other limitations of Petri net-based approaches are the requirement that all tokens and transitions behave the same way. In other words how a protein binding event is modelled, is the same as how an enzymatically catalysed biochemical reaction would be modelled, where outputs are determined by the number of tokens on the reactants. This is not likely to be an issue for many applications, but could limit the approach's applicability in certain circumstances requiring a more complicated concepts to be built within the model.

Experiment Design

First we describe how to construct a graphical model of a biological pathway using the mEPN scheme (steps 1-8). We then explain how to convert this purely graphical representation into a resource that

supports computational modelling of the system (steps 9-11). Next, we present how to test the dynamic properties of the model through running simulations and visualizing results (steps 12-20), and in the final section (steps 21-25), describe how to optimise and validate the pathway model. The workflow is shown schematically in Figure 1. To illustrate our approach, we use a model of interferon-β signalling. The model is small and simple, but encompasses many of the basic concepts associated with pathway construction and motifs such as a negative feedback loop, a common feature of many biological systems [43]. However, we encourage the examination of other larger models we have constructed, covering a range of biological systems, in order to appreciate the scale and complexity models can achieve (examples can be found at www.virtuallyimmune.com). Before embarking on model construction, users should search the literature and pathway databases, such as those listed in Table 1, for existing diagrams of their system of interest. Careful consideration should be given to the initial scope of the model, the level of detail to be represented and what the model is to be used for once constructed. For instance, given the interconnectivity of biological pathways, it is easy to begin with the aim of modelling one thing and end up spending a lot of time modelling something entirely different, because it is one way or another related to the first. Having said this, models will inevitably evolve as information is gathered and assimilated, and the journey taken is part of the reward of modelling.

Materials

Equipment

A computer with Windows, Apple Mac or Linux operating system (preferably 64-bit), internet connection and a web browser with JavaScript enabled. The hardware configuration may limit the size of models that can be displayed within yEd, as well as when running pathway simulations within BioLayout *Express*^{3D}, where it will influence the speed of simulations and the frame rate for animations of flow. We recommended >4Gb main RAM, a Dual-core CPU, NVidia GeForce / Quadro series or ATI equivalent

184 graphics card for advanced visualization with GLSL Shaders, preferably two monitors capable of
185 displaying at 1,600 x 1,200 resolution and a three-button mouse to aid navigation.

186

187 Equipment setup

188 Installation of yEd Graph editor

189 yEd is a free and intuitive software application that can be used to create high-quality network diagrams, it
190 runs on all major platforms: Windows, Unix/Linux and Mac OS X. Download and install the latest release
191 of the yEd Graph editor from the yWorks (Tübingen, Germany) website www.yworks.com. yEd will use up
192 approximately 215 MB of hard disk space. If you encounter any problems with the installation of yEd
193 contact: support@yworks.com

194 Loading the mEPN palette

195 Download the GraphML (.graphml) file containing the mEPN glyphs (Supplementary Data 1) and load it
196 into yEd by selecting Edit → Manage Palette → Import Section. This will provide the standard palette of
197 mEPN glyphs that can be selected as required when constructing a pathway model. To display the mEPN
198 symbols palette select from the menu bar Windows → Palette. Alternatively, create each node type
199 afresh by adding a node and changing its visual properties [F6]. As an example, see the list of
200 components present in the interferon-β pathway (Figure 2A).

201 Installation of BioLayout *Express*^{3D}

202 BioLayout *Express*^{3D} software allows the visualization and analysis of large network graphs in two and
203 three-dimensional space and supports the computational modelling of networks using the signalling Petri
204 net (SPN) algorithm[37]. BioLayout *Express*^{3D} runs on Windows, Mac OS X and Linux platforms. Java SE 6
205 or 7 is required and can be downloaded from http://www.java.com/getjava. BioLayout will use up
206 approximately 41 MB of hard disk space. To download the BioLayout *Express*^{3D} installer**,** navigate to
207 http://www.biolayout.org/download/ and download an installer for Windows (.exe) or Mac OS X (.dmg).
208 For Linux platform use the universal JAR file that may be run without an installer. When BioLayout

Express[3D] runs for the first time it creates a preferences file that can be changed and saved at any time from the menu option Tools → Save Preferences. Users can customize many options selecting from the menu bar Tools → General Properties (Shift+P). Further details on the software interface and its customization are available in the BioLayout *Express*[3D] manual that can be downloaded from the tools support pages. If you encounter any problems with the installation of BioLayout *Express*[3D] contact: support@biolayout.org

Procedure

Pathway model construction (timing: hours to months depending on complexity of model)

1. Source information for pathway construction**.** A pathway model should aim to provide a comprehensive and reliable view of the current state of knowledge about the system. To achieve this collect and extract the relevant information about the pathway from the literature, databases and existing diagrams. Possible sources to consult are presented in Table 1. A comprehensive list of databases for data mining can be found at www.pathguide.org. For some pathways, data are available from multiple species and/or cellular systems, therefore users must decide whether to piece together information from heterogeneous sources or to restrict their model to reflect a particular species, cell type or developmental stage. To keep track of the data, create a spreadsheet that includes: molecular identifiers, e.g. HUGO, Entrez IDs, details about nature of the molecular interactions, sources of information, e.g. PubMed ID, the quality of evidence (as assessed by number of publications supporting a given interaction and the reliability of the assays used) and any additional information that may be relevant.

   !Troubleshooting

2. Identify the types of pathway information. Divide details of the pathway of interest into the categories defined the mEPN notation scheme (see BOX 1). Find the 'real' name of pathway components. The use of standard gene/protein names is essential in defining the exact identity of components, especially if models are to be used in the interpretation of omics data where the use of standardised nomenclature systems is standard practice (see BOX 3). Record and characterise the type of interaction between components.

3. Commence drawing - addition of entity nodes. Molecular components are represented using entity nodes. To add an entity node to the diagram, select and drag the appropriate glyph from the mEPN palette (BOX 1). Edit a node's properties by selecting it and pressing [F6]. The node Properties dialogue will appear. Add the component's name to the General tab, where necessary changing the size of the node to fit the label, record the reference source or insert a brief description about a given component in the Data tab. Also add a hyperlink to an external site (for example NCBI's Gene database), which can then be activated by selecting the node and pressing [F8]. A description of the component, if available, will be shown in a pop-up window when the mouse is placed over the node in yEd.

4. Draw the interactions between entity nodes. The nature of an interaction between components may be represented using a combination of process nodes and edges. To add a process node to the diagram, select and drag the appropriate glyph from the mEPN palette (BOX 1). As with components (entity nodes) additional information may be added to the process node by selecting it and opening the Properties dialogue [F6]. Pathway modules are a special type of process node. They represent multi-reaction processes or events and are represented using octagons with a label identifying the name of the process they represent. They might be used to represent such pathway as signalling cascades, endocytosis, compartment fusion, etc. Edges are lines that join entity and process nodes. Edges denote the type of interaction (activation, catalysis, inhibition) and their directionality establishes inputs and outputs from entity/process nodes. To add an edge, click on a node using left

mouse button and keep held down, then drag the mouse to move the edge to the target node and release. If you release the mouse button on the way to the target node, a pivot point will be introduced. To change the appearance of an edge (colour, thickness, arrow type or to add text/hyperlink), select the edge by clicking on it and open the edge properties dialogue [F6]. The sample *Interferon_components.graphml* file can be used to try out the procedures described in this step (Supplementary Data 2). **Note:** yED supports the import of data in Excel or .CSV files in a variety of formats (see http://yed.yworks.com/support/ for details). This functionality may help initially in defining a set of pathway components and the interactions between them, prior to manual editing of node/edge properties and layout.

CRITICAL STEP. In general (and absolutely so when constructing a diagram to be used for computational modelling), nodes comprising a pathway should be arranged as a bipartite graph i.e. entity nodes should be connected exclusively to a process nodes and vice versa. This structure is the same structure used by Petri nets (places must be connected to transitions) and it is essential if the model is to be used for simulation experiments.

5. Add compartments. Components should be represented as existing within a given cellular compartment. Drag the desired compartment node from the mEPN palette and enlarge it to cover the section of the pathway diagram which represents a given cellular compartment such as the plasma membrane, cytoplasm or nucleus. Move the selected compartment behind the diagram by choosing Edit → Lower selection. Name the compartment, placing the name between asterisks (*compartment name*). The asterisks inform the BioLayout *Express*^{3D} parser to treat these nodes differently: they are displayed as a translucent background to the pathway and cannot be selected within this tool. If they are labelled as follows *compartment*N* where *N* is a numerical value, e.g. 100, when viewed in BioLayout the compartment becomes a 3D container where the N value determines its depth (Z-value). The decision on a compartment's 'depth' is based on purely on final aesthetics as compartments do not effect model dynamics. Generally the cell membrane would be the

largest component and its internal organelles are smaller compartments that sit within it, but other than this the values given are subjective. Suggested cellular compartment colours are defined in the mEPN palette. Generally, when compiling a model it is best to add compartment nodes at the end or at least put them to one side when editing, as they tend to get in the way. The completed pathway should look similar to the interferon-β pathway example in Figure 2B.

6. **(Optional)** Add negative feedback loops. Feedback loops are network motifs common to many systems and involve the activation of a pathway component that goes on to inhibit an earlier step in the process. The inhibition of the interferon-β receptor by SOCS1 whose transcription is activated by the interferon signalling pathway represents such a negative feedback loop (Figure 2B). To represent an inhibitory activity such as this using the mEPN scheme, place an inhibitor edge from the inhibitor molecule to a process node representing the step that is inhibited.

7. Optimise model layout. It is essential that a pathway model is compact and easy to follow i.e. be readable by a human. How this is best achieved will be influenced by a model's size and complexity. First organise the pathway components based on where they reside within the cell, i.e. their cellular compartment. Then attempt to separate out 'modules' based on connectivity amongst a group of nodes, e.g. a particular signalling cascade or other series of events. This helps with a model's readability and facilitates model expansion as new data becomes available. Further information on layout optimisation can be found in BOX 4. In practice, model optimisation is normally an iterative and time consuming process often requiring a degree of trial and error in how best to layout the diagram. When in the dynamic modelling phase it may for example be necessary to add in motifs that in effect delay the passage of tokens from place to another in order, to model processes that are not explicitly shown but may influence the order or timing of an event.

8. Save and export the pathway models. The pathway model should now represent a diagrammatic version of known events and should be saved in the GraphML file format choosing File → Save. Within yEd a model can also be exported as an image in PNG or JPG format, as PDF or as HTML by selecting File → Export and selecting the preferred format.

Model testing - conversion of a graphical model into a computation model (timing: minutes to hours)

9. Set the initial parameters. To convert a graphical representation of a pathway into a computational model, you must define the initial state of the system through a process of 'parameterisation'. To parametrise the assembled model, add defined numbers of tokens to entity nodes at the beginning of network i.e. nodes that have no 'parents' (upstream connections). To define the initial state of a component, place an input node (depicted as a black rectangle which functions as a transition node) upstream of the component to be parameterised and connect it with a standard edge. Define the number of tokens to be added to the node by selecting the edge between an input node and component. Open the edge properties dialogue [F6] and type the desired number of tokens into the edge name (text) (Figure 2C). Token values can in theory range from 0 to millions but in essence represent the relative amount or activity of a given component under initial conditions. Ideally, the initial parameters should be set with reference to some experimental data providing information on the relative initial concentrations of the pathway components where known. However, in the initial stages of pathway parameterisation and model testing, it is often sufficient to place an arbitrary number of tokens on components, e.g. 1000, just to check the connectivity between inputs and outputs is not compromised in any way.

10. Defining inhibitory reactions. An inhibitor edge originates from an inhibitory molecule and terminates at the process to be inhibited, tokens present on the inhibitor node preventing token flow through the process node. During a simulation tokens will not be lost through an

inhibitor edge and therefore tokens accumulate on an inhibitor node and irrevocably block the process to which it is connected. However, if a sink node is connected to the inhibitor it serves to give the inhibitory molecule a 'half-life' (in practice any process node will serve the same purpose, but use of the sink node helps visually define the process involved). In the absence of further input into the inhibitor node, such as during the 'off phase' of negative feedback system, tokens will now be lost from the inhibitor. The result is that its inhibitory effect will lessen and the blocked transition will eventually open and tokens may flow again through it. In presence of a constant input this can cause token flow in negative feedback systems to oscillate. There are two types of inhibitor edge included in the notation scheme that perform differently in the modelling environment; the non-competitive inhibitor edge (red with perpendicular bar at end) and a competitive inhibition edge (red with open diamond end). The non-competitive inhibitor edge completely blocks token flow through the target transmission if any tokens are present on the inhibitor node. In contrast the competitive inhibitor edge works by deducting the number of tokens residing on the inhibitor away from the number of tokens flowing through the target transition. The behaviour of negative feedback systems is not only dependent on the type of inhibitor edge used but also the distance between the input of tokens and the inhibitory step. The greater the distance the more tokens are able to accumulate in the system and the greater the time taken between the opening and closing of the inhibited transition, i.e., the longer the wavelength and the higher the amplitude of the oscillating signal. Other factors that can affect the oscillatory behaviour of the feedback loop are the number of inhibitors acting on the pathway and assumptions about the stochasticity of token flow.

11. Save the parameterised model in the GraphML file format choosing File → Save. GraphML files can be loaded directly into BioLayout *Express*^{3D}. A parser within the tool translates the mEPN nodes into their 3D equivalent shapes such that they can now be visualised within the tool's 3D environment. It also differentiates between nodes in the diagram that act as Petri net 'places' and that are 'transitions', and reads the parameterisation markings that define

initial token inputs. BioLayout *Express*^{3D} is also able to perform stochastic flow simulations using a modified version of the signalling Petri net algorithm[37]. A file containing simple Petri net models of all primary motifs found in pathways is provided as a means to better understand the flow characteristics of this algorithm (Supplementary Data 3).

Running simulations using BioLayout *Express*^{3D} (timing: 5-15 min)

12. Load the saved GraphML file into BioLayout *Express*^{3D}. Supplementary file 4 is the interferon-β pathway model shown in Figure 3C and as such is 'simulation ready'. Following opening of the file answer yes to the dialogue window "This looks like a Signalling Petri Net (SPN) pathway. Would you like to run a SPN simulation now?" (Figure 3A). This opens the SPN simulation dialogue (Figure 3B). The dialogue can also be selected from the main menu under the Simulation menu or by pressing the "RUN SPN" button on the sidebar.

    !Troubleshooting

13. Set the SPN simulation options. Choose the number of time blocks and the number of runs. A 'time block' is when all transitions are fired exactly once in a random order and tokens moved as a result. A series of time blocks is referred to as a 'run', the more time blocks the longer the run (Figure 3B1). The bigger the model or the more conditions you want to test within a simulation, the more time blocks you will require for a simulation. It is good practice to check the nodes furthest away from token input points to ensure that token accumulation has plateaued or in the case of negative feedback circuits, that enough time blocks have been run to evaluate the oscillatory behaviour of the system. The Petri net algorithm employed here is stochastic in nature. That is to say that the number of tokens passed on, when a transition is 'fired' is variable depending on the algorithms stochastic setting (see below), and furthermore the order in which transitions are fired is random. Therefore, the result of individual runs can be highly variable. For this reason a simulation is generally

comprised of multiple runs, where the outcomes from individual runs are averaged to calculate the mean number of tokens present on a given node at each time block (Figure 3B2). The more runs used the less variable the results between simulations, but the more time it will take to perform a simulation. To visualise the variation associated with a given simulation check the 'Calculate Variance' and pick either standard deviation or standard error (Figure 3B3).

14. Select the token stochasticity setting. The possible modes for token flow can be selected as shown in (Figure 3B4) and are described below:

    Uniform Distribution: Each time a transition is fired, an entirely random number of tokens between zero and the maximum number of tokens are moved from an input place to the output place (assuming there no other inputs on the transition which may influence flow). This mode is as originally described by Ruths et al.37 in their description of the SPN algorithm.

    Standard Normal: Each time a transition is fired, the number of tokens moved between input and end places will be randomly chosen from a standard normal distribution around 50% of the number of tokens on the input place.

    Deterministic: This moves exactly half of the tokens from input place to the output place each time a transition is fired.

Normally we would use the standard normal distribution setting, as a halfway house between the other two modes of token flow. In some settings, the average result of simulations is similar with all these settings although variation between runs is greater with the more stochastic token flow settings, especially the uniform mode. However, when simulating feedback loops the mode of token flow can have a marked effect on the behaviour of such systems. The mode you select may be based on which best models your system of interest.

15. Selection of SPN simulation transition type. Token movement between places is via transition nodes which all operate using the same set of rules governing token flow. We have introduced two options, consumptive transitions and original transitions, which differ in how they operate with respect to token accumulation (Figure 3B5). Generally, we use the consumptive transition mode as this prevents the accumulation of tokens on entities where there is a constant input of tokens throughout the simulation but where flow through the target transition may be intermittent.

    Consumptive Transitions: Tokens are consumed from place nodes irrespective of whether the transition is 'open' or not i.e. if there are two inputs into a transition and one has tokens and the other does not, tokens will still be lost from the input place with tokens as if flow were unrestricted.

    Original Transitions: Tokens accumulate on input nodes where flow from them is blocked i.e. if there are two inputs into a transition and one has tokens and the other does not, tokens will not be lost from the input with tokens. This mode was as originally described by Ruths et al.[37].

15. Run the simulation. Press the **'**Run the Simulation**'** button to initiate the computation of the SPN algorithm (Figure 3B6).

    Critical Step. The time it takes to run a computation depends on the number of time blocks/runs, the size of the pathway model and hardware on which the simulation is run. However, for most small to medium size pathways (10's-100's of entity/process nodes) and hardware configurations, the time taken is usually a few seconds or less for a typical modern laptop.

!Troubleshooting

16. Save the results. Once the SPN simulation algorithm has finished, a Simulation Results dialogue appears (Figure 3C). Token level per node per time block results can be saved as a *.txt* or *.spn* file by ticking the "Save SPN Results" or pressing ALT+S (Figure 3C7). Saved simulation results files can be loaded pressing ALT+L in the main window. *.spn* result files can also be opened in programs such as Excel as a spreadsheet, or viewed and edited in a text editor. An mEPN model can also be exported as a Systems Biology Graphical Notation (SBGN)[44] diagram via File -> Export -> SBGN file. This can be opened in any SBGN compliant software, e.g. VANTED with SBGN-ED add-on[21].

17. Visualize token flow as node output graphs. To visualize the simulation results for a selected entity/place node, close the SPN simulation results dialogue (Figure 3C8), position the cursor over the node of interest and a pop-up window will appear showing the token flow associated with that node (Figure 3D). To view and compare token flow in multiple nodes, select the nodes of interest by pressing Shift+left mouse button and dragging the select window over the nodes of interest or by pressing the Shift+ALT+left mouse button to select multiple nodes. The corresponding flow graphs can be viewed using the Class Viewer by pressing CTRL+C (Figure 3E) or the button with cog icon on left menu bar of the main window (Figure 3G).

    A range of options are available within BioLayout to adjust graph appearance. Shift+> or shift+< will increase or decrease node size, respectively; under the General tab you may turn on or off the visualisation of the compartments (yEd Graphml Container Rendering); and by pressing the 'Render Plot to File button' on the top of the Class viewer window the graph can be saved as .jpg or .png image file (Figure 3E9).

18. Visualize token flow as an animation. Open the Simulation Animation Control window (ALT + A) when the simulation has finished (Figure 3F). Use options provided within BioLayout *Express*[3D] tool to control simulation visualization:

Node Animation (Figure 3F10). Choose which nodes are animated (all, selected or pathway components only), and the type of animated transition that takes place between the node value associated with one time block and the next (discrete, linear, polynomial).

Timing (Figure 3F11). Define how many time blocks per second are displayed and therefore the speed of the animation. If necessary also define which time block the animation begins from.

Size Transition (Figure 3F12). Set maximum size of nodes during the visualization of token flow, i.e. when the number of tokens is at its maximum. The 'Set (fixed) node value' is the number of tokens on a node at which the maximum node size/colour is reached. The default value for the 'Max value' is determined by the maximum number of tokens that accumulates on any node during a simulation. It is often the case that some nodes accumulate tokens much in excess of others, e.g. when their output is blocked. This can result in the majority of nodes seemingly to change little in size or colour during a simulation. Click on this value and add a value of your choice, and click on the associated check box, to maintain this value for subsequent runs.

Colour Palette Spectrum Transition (Figure 3F13). A number of colour palettes are available, or one can be loaded by user, to colour nodes so as to reflect their token value. Select colour spectra from dropdown menu, load your own, or more normally, use default.

19. Visualize token flow as an animation. Select 'Start Animation' (Figure 3F14) to watch tokens flow through your model (Figure 3G and Supplementary Video 1).

!Troubleshooting

Model optimisation, parameterisation and validation (timing: days to months)

20. Check for errors. Errors in a diagram's structure (predominately a failure to adhere to the strict requirement for a bipartite graph or improper logic), can lead to bottlenecks in token flow. When the bipartite graph structure is not maintained (e.g. an entity node is directly linked to another), tokens will accumulate on the node upstream of the issue and tokens are not passed downstream of the error. Check the reactions preceding any entity node whose token output is zero (Figure 4A) and correct mistakes. Place and transition spacer nodes are available to position between two nodes of the same type where the graphical description of events leads to this situation. Another common issue encountered is where the presence a pathway component is under the control of the system in which it operates. This can lead to a situation whereby for component to be synthesised it needs the pathway to be active, but the pathway is not active because it requires the activity of that component. In these circumstances it may be necessary to 'prime' the system adding tokens to the component in question prior to beginning the simulation. Mistakes and errors in logic are easy to make, but equally easy to spot and rectify with this approach. It is normal practice to run a simulation, find out where the issues are, edit the model in yEd and rerun the simulation. There will likely be a need to repeat this process a number of times.

21. End of the line. Without a downstream transition, a component at the end of a line of flow will simply accumulate tokens (Figure 4A). A final transition node is required to dissipate tokens from such entities. One option is to place a 'pathway node' at these points by dragging and dropping from the palette to allow indication of what happens next without showing it in detail. Alternatively, a 'sink' node can be placed as described above to signify that a component is removed from the system, e.g., the destruction of a protein by proteosomal degradation.

23. Amplify or reduce token flow at specific sites. To simulate the amplification or reduction of a signal at specific sites in the network, add a numeric value to a transition-to-place edge. Select

the edge, press [F6] and write a number in the Text field of the edge Properties dialogue. When a transition fires, the number of tokens produced on the downstream entity will correspond to the number of input tokens multiplied by the weight of the output edge, e.g., an edge weight of 2 will result in a doubling in the number of input tokens, where as an edge weight of 0.5 will halve the number of tokens going forward. For example, one can amplify tokens as means to model the production of numerous protein molecules from a single transcript during protein translation (Figure 4C).

24. Varying token input during a simulation. To simulate variation in the level of an input signal at different time blocks of a simulation, assign tokens to an input edge (as described in Step 9) using the following notation : a-b,c;d-e,f where 'a-b' defines the first and last time blocks that the number of tokens 'c' will be added to the model and 'd-e' are the first and last time blocks that you would like the number of tokens 'f' to be added to the model. For example: 0-5,0;6-15,100,16-20,0 translates into, add no tokens between time blocks 1-5, 100 tokens between time blocks 6-15, and then remove token input until the end of the run, time block 20. Any number of these statements may be added to an input. This allows modelling of a system before and after a stimulus, or when a stimulus is transient or delayed.

25. Validate the model by comparing it to experimental data**.** Once a model has been constructed, checked for structural errors and parameterised according to known variables, the first question is whether the model recapitulates the known activity of the system. For example, check if genes are expressed as transcriptomics data suggests, or does the flow of metabolic pathways under different conditions reflect what is known? The simulation of pathway dynamics should recapitulate the known activity of the system. If not, the obvious conclusion is the model is wrong. This could be because it is poorly constructed or parameterised, in which case the model needs improving. More interestingly, it could reflect the fact that there is part of

554 the system that is as yet undiscovered. Once a model is working, i.e., it verifies the known
555 characteristics of the pathway, it can be used to test known perturbations of the system e.g.
556 the effect of knocking down/out a gene or inhibiting an enzyme. With confidence in a model's
557 characteristics it is then reasonable to use it to predict the effect of perturbing it, proving
558 results that can be tested experimentally: one of the ultimate aims of dynamic modelling.

559

560

561 **Troubleshooting**

562 Troubleshooting information can be found in Table 2

563

564 **Table 2: Troubleshooting table.**

Step	Problem	Possible reason	Solution
1	Information about a part of the pathway is unavailable.	The experiments to elucidate the process have not been done.	Use a 'pathway module' node to indicate that a process occurs but the details of which are undefined.
12	BioLayout *Express*^{3D} fails to run.	Incompatible hardware or software configuration.	Contact support@biolayout.org. Improve hardware specification.
15	'Error with Vertex weight!' error popup is shown during simulation.	Token input to node(s) is not readable by software.	Check that token input is numerical (token input can be any positive number, including decimals) at all token input nodes.
19	Flow stops and downstream nodes do not accumulate tokens.	Bipartite graph structure has not been adhered to.	Identify bottleneck node by watching BioLayout flow animation. Return to yEd graph to edit model and rerun.
	Node accumulates tokens at linear rate.	Node has no output.	Return to yEd graph to add sink node or other transition to node accumulating tokens.
	Token flow occurs, but the size of nodes changes little.	The maximum token value is set too high.	Open Animation Control window and lower value in 'Set Max Value' dialogue box.

565

566

Timing

Steps 1 to 8, Information mining and pathway construction: Hours to months

Steps 9 to 11, Conversion of the graphical model into a computable format: Minutes to hours

Steps 12 to 19, Visualization of pathway and running simulations using BioLayout *Express*^{3D}: 5 to 15 minutes

Steps 20 to 25, Model optimisation and parameterisation: Days to months

Anticipated Results

This protocol describes the generation of pathway models using the mEPN language for graphically representing biological systems. Graphical models can be used both as a resource to store and display what is known about a pathway and can be considered an end point in their own right, that can be updated or extended as new information becomes available. In addition, they can be converted to a computational model by setting parameters to define the initial state of the pathway. Using the sample interferon-β components GraphML file (Supplementary Data 2) a pathway model can be produced that represents events from the binding of interferon-β to its receptor and the signalling pathway leading the activation of target genes. The resulting model can be parameterised by adding tokens to obtain a simulation-ready model (Supplementary Video 1). This is a relatively small diagram; we also provide a model of the hedgehog signalling pathway as an example of a larger model (Supplementary Data 6). This was produced as part of a 10 week elective course by an undergraduate student with no previous modelling experience**,** indicating our modelling scheme can easily be performed by biologists with no prior modelling knowledge. Other pathway models are available at: www.virtuallyimmune.com.

The BioLayout *Express*^{3D} software is fully compatible with the mEPN notation and can be used for pathway visualization and to perform stochastic flow simulations using a modified version of SPN

590 algorithm[37]. Running simulations provides insights into the dynamic behaviour of the system by
591 enabling users to visualize the signal flow within the network. The signal flow is simulated by the
592 accumulation of tokens at entity nodes (places) and can be visualized in 2D graphs (as seen in Figure
593 4) or by 3D animation (as seen in Supplementary Video 1). The inflation and contraction of the entity
594 nodes represents the accumulation and degradation of reactants in the pathway. In this way, complex
595 biological processes with multiple components can be modelled.

596

597 **The interferon-β signalling network - a feedback control system**

598 Biological systems display a variety of dynamic behaviours ranging from stable steady states to
599 oscillations. Oscillations in protein concentrations or gene expression levels are commonly associated
600 with the presence of negative feedback loop(s) in the regulatory network[45]. Based on the analyses
601 performed using this system many factors can affect the amplitude, frequency and stability of oscillations.
602 For instance, the time-delay (path length) between token input and inhibitor, the type of inhibition edge
603 used (competitive or non-competitive), the number of inhibitors present and their half-life, may all affect
604 how a model containing a negative feedback loop will operate in practice.

605 As an example, the simple model of the interferon-β signalling pathway is presented. Interferon-β is a
606 cytokine released by immune cells in response to pathogens. It acts as an autocrine and paracrine
607 signalling system that triggers the activation of host defence systems. In the provided model it operates
608 as a damped oscillator[46], where low-dose IFN stimulation yields oscillations of lesser amplitude that are
609 damped faster than those induced at a high-dose (Figure 5). To reproduce these observations users can
610 look at the different versions of IFN signalling feedback model provided in Supplementary Data 5 or
611 modify the token input or topology of the Petri net examples provided in Supplementary Data 3.

612

613 **Contributions of the authors:** A.L., L.O'H., M.E.P developed and wrote the protocol based on their
614 experience in using the approach for modelling their own pathway systems of interest, T.A. developed a
615 number of the features within BioLayout *Express*[3D] including SBGN export and refinement of the Petri net

algorithm, D.W. developed and has helped maintain the VirtuallyImmune.org website that is associated with this work, and L.B.S. helped with writing and editing the manuscript. T.C.F. has lead the development of the mEPN notation scheme and its use in modelling a variety of pathway systems; oversaw the implementation of model import into BioLayout *Express*^{3D}; model visualisation within this tool, refactoring and refinement of the Petri net algorithm, and conceived of and assisted in writing the paper.

Figure legends

Figure 1. Workflow with steps described in the Procedure.

Figure 2. Construction of a simple pathway model describing type-1 interferon signalling. (A) Components of the interferon-β signalling pathway drawn using mEPN notation (BOX 1). The information necessary to construct the interferon-β pathway has been highlighted in the pathway description: entity and transition nodes, edges and cellular compartments. **(B)** interferon-β pathway diagram assembled in yEd software using the pathway parts shown in A. **(C)** Parameterisation of interferon-β pathway by the addition of token inputs and a sink node output on SOCS1 inhibitor node (highlighted by red rectangles). Also shown is the edge properties dialogue in yEd where tokens can be added to an input edge. Model adapted from O'Hara *et al.*, 2016 [1].

Figure 3. Visualization of token flow. (A) When a model is loaded into BioLayout *Express*^{3D} it is displayed in a 3D environment using 3D equivalents to the 2D node glyphs as rendered in yEd. The software automatically recognizes a diagram as having been parameterised for computational modelling (based on the presence of process nodes as defined in the notation system) and prompts users to run a SPN simulation. **(B)** In the SPN Simulation dialogue users can set constraints on how to run the SPN simulation algorithm. (1) Defines the number of time blocks in a simulation; (2) Defines the number of runs in a simulation; (3) Calculates the variance in token flow between runs; (4) Defines the nature of the stochastic flow of tokens; (5) Defines the rules governing token flow through transitions; (6) Run the simulation. **(C)** SPN Simulation Results dialogue box summarises the simulation. (7) SPN results may be saved before the user chooses to (8) run the simulation again, close the dialogue box or proceed to

642 animate the simulation. **(D)** After a simulation has been run token accumulation at specific nodes can be
643 visualised by placing the cursor over the node. **(E)** The flow of tokens across one or a number of selected
644 nodes can be plotted using the Class Viewer showing plots of token flow over the time course of the
645 experiment for selected nodes. The name and class of selected nodes is also displayed, and below are a
646 range of options available for node selection and data export. (9) The token plot can be saved as a .png
647 or .jpg image file. **(F)** Animation Control dialogue. (10) Options for the type of nodes to be animated and
648 interpolation of flow between time bocks; (11) Speed of animation (time blocks per second); (12) Node
649 size at maximum token number; (13) Colour palette selection to highlight change in token number; (14)
650 Animation control: start, pause, step, stop. **(G)** BioLayout *Express*^{3D} can produce animations of token flow
651 through the model across time blocks. Node size will increase/decrease depending on the number of
652 tokens passing through them and the colour of the node will also change according to a predefined
653 spectrum of colours.

654 **Figure 4. Influencing token flow along a linear pathway.** Figure shows the flow of tokens through a
655 small linear pathway motif depicting a gene being transcribed into mRNA, which is then translated into the
656 encoded protein. Addition of a sink node represents the protein's degradation. **(A)** Blocked flow. Top of
657 the three illustrations all is as described above, and over the 20 time blocks of the simulation there is an
658 initial rise in the level of the protein followed by a steady-state, as the number of tokens entering the entity
659 node matches those leaving through the sink i.e. the rate of production of the protein matches the rate of
660 its degradation (blue line). Failure to maintain bipartite graph structure (shown here by connecting two
661 entity nodes for mRNA and protein without a process node between them) causes tokens to accumulate
662 on the first entity node (mRNA) and not pass to the second (protein), which stays at zero tokens
663 throughout the simulation (pink line). In the absence of a sink node, tokens to accumulate on the protein
664 node (green line). **(B)** Modelling time. The diagrams show the effect of increasing the length of linear
665 networks on the rate of token accumulation. Input tokens are introduced into each of the networks at the
666 same time but the protein accumulates at different rates. The delay is proportional to the number of
667 transition steps introduced in the network and such delay motifs may be added where a transition nodes
668 represents a multistep process such as transcription or translation. **(C)** Amplifying or depleting signal.
669 The addition of a value to the edge leaving a transition can be used increase (pink line) or decrease

(green line) downstream token flow. In this way one might model one mRNA molecule leading to production of multiple protein molecules, or the inefficient translation of mRNA where only a single protein molecule is produced from multiple mRNAs. Simulation options in A, B and C: 100 tokens, 100 runs, 20 time blocks, Normal Standard distribution and with standard deviation of token flow between runs shown.

Figure 5: Effect of parameterisation on activity of feedback loop. (A) A simple pathway model representing the type-1 interferon signalling pathway constructed and parameterized in yED using mEPN. Graphs show token accumulation on the activated transcription complex ISGF3 (circled in red) following simulations with **(B)** 1000 input tokens (blue line) or 100 input tokens (green line) added to interferon-β with SOCS1 acting as a non-competitive inhibitor or **(C)** as a competitive inhibitor of the activated receptor (dashed red edges). **(D)** Oscillatory activity of the SOCS1 feedback loop with (magenta line) or without (blue line) a delay introduced between the transcription and translation of SOCS1 (dashed black edges), again with SOCS1 acting as a non-competitive or **(E)** competitive inhibitor of the activated receptor. Simulation options: 100 runs, 500 time blocks, Normal Standard distribution and with standard deviation of token flow between runs shown.

Supplementary Files

Supplementary Data 1. mEPN palette for loading within yEd software. This is a graphml file containing all the different types of entity nodes to represent the different classes of molecules that might play a part in a pathway, as well as the process nodes that represent different types of interactions. This file can be loaded into yEd to provide a palette of mEPN nodes for pathway construction (see step 4 of the procedure).

Supplementary Data 2. Interferon-β signalling pathway components. This is a graphml file containing all the different parts of the simple model shown in figure 2B, to allow practicing model construction. Try assembling the model using only the text below: '*Interferon B (IFNB1) is a cytokine released from many cell types in response to immune stimulation. It homodimerises and binds to its cell surface receptor complex composed of the receptor proteins IFNAR1 and IFNAR2 and the intracellular kinases TYK2 and JAK1. The complex is composed of 2 of each of these proteins. Binding causes a conformation change in the complex resulting in the autophosphorylation of JAK1. Once activated, the complex catalyses the phosphorylation of STAT2 which forms a heterodimer with STAT1. This complex then binds interferon regulatory factor 9 (IRF9), forming the complex often referred to as ISGF3, and translocates to the nucleus. Here it binds to the IRF sequence in the promotor of a number of genes including MX1, MX2, IFIT1, IFITM3, TAP1, OAS1, GBP1, PSMB9, SOCS1. In turn SOCS1 inhibits the autophosphorylation of the receptor thereby inhibiting further activation.*'

Supplementary Data 3. Primary network motifs drawn in Petri net style for testing SPN algorithm. This is a graphml file containing a series of different network motifs and parameterisations potentially found in pathway diagrams. This includes linear networks, multiple inputs/outputs to transitions and nodes, and a series of models representing a range of feedback loops with varying path lengths between token input and inhibition, inhibition type (competitive vs. non-competitive), and one or multiple feedback inhibitors. The file is designed to allow you to explore the different interaction types and algorithm settings when setting up a simulation run. Certain nodes are coloured such that when a simulation has been run within BioLayout, these nodes may be selected and you can compare results across motifs.

712 **Supplementary Data 4.** Interferon-β signalling pathway. This is a graphml file of the simple model
713 shown in figure 2B.

714 **Supplementary Data 5.** Changing parameters - Interferon-β signalling pathway. This is a graphml file
715 containing six versions of the model shown figure 2B (Supplementary Data 4), each version is
716 parameterised slightly differently, with variation in: type of inhibition (competitive vs. non-competitive);
717 introduction of a delay between SOCs1 expression and protein; and an amplification of signal between
718 gene and mRNA.

719 **Supplementary Data 6.** Example of a more complex model - Hedgehog signalling pathway. This model
720 (given here as graphml file) is a representation of Hedgehog signalling from the binding at its receptor,
721 activation of the GLI protein on the tip of the primary cilium through the activation of various downstream
722 pathways (not shown in detail). It contains 550 nodes and 601 edges and was assembled using pathway
723 resources such as Reactome[22] and the primary literature. Its parameterisation, in terms of token
724 placement is arbitrary.

725

726 **Supplementary Video 1.** Movie of the interferon-β signalling pathway (Supplementary Data 4) simulation
727 run within BioLayout. The movie shows the process of model loading, running the simulation, inspecting
728 token accumulation on specific components and watching the flow of tokens run through the model as an
729 animation.

730

731

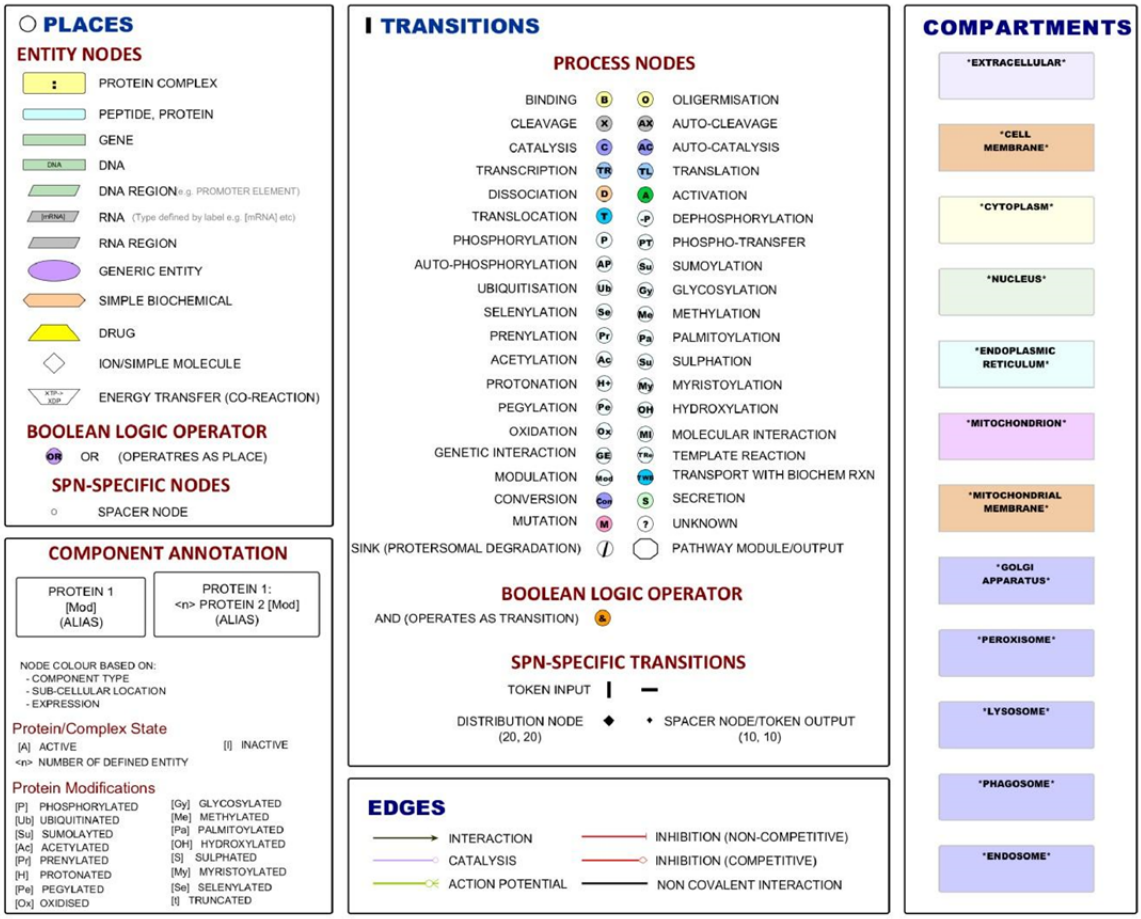
732

733

734

BOX 1: mEPN Notation 2017

The modified Edinburgh Pathway Notation (mEPN) scheme[3] is a graphical notation system based on the concepts of the process diagram[4], below the glyph library is shown in its current form (reproduced from O'Hara *et al.* [1]).



Pathway components

Types of pathway information. The information depicted in a pathway diagram drawn using the mEPN scheme may be divided into the following categories:

- *Entity:* any component involved in a pathway, e.g. protein, protein complex, nucleic acid sequence (promoter, gene, RNA), simple biochemical, drug etc., and depicted as an 'entity node'. Different shaped nodes are used to represent different types of components. The mEPN scheme consists of twelve different entity nodes (detailed in the top left panel). Also included are a Boolean logic 'OR' operator node and spacer node (represented as a white circle with a black border) that may be required to maintain bipartite pathway arrangement. Entity nodes function as the equivalent of Petri net 'places' and all entity nodes are equivalent in Petri net simulations
- *Process:* an interaction that occurs between pathway components, where one component interacts with or influences the state of another through its binding, inhibition, catalytic conversion, etc., is depicted as a process node. Processes are generally depicted as a circular node with a 1-3 letter code to indicate the type of process, e.g., P = phosphorylation, B = binds, X =cleavage etc. Included in the scheme are 38 different process nodes (top central panel). Additional to these, but also acting as transitions, are a sink node which is placed at the end of pathway and represents removal of a component from the system; a pathway module node that summarises not one process but a series of events; a Boolean logic 'AND' operator node; token input nodes that are placed at the start of a pathway and oriented either horizontally or vertically to fit in with the pathway layout; a spacer node represented as a black diamond. A larger version of this node can also be used as a distribution node when multiple edges exit from an entity node. Process nodes function as the equivalent of Petri net 'transitions' and all process nodes are equivalent in Petri net simulations.
- *Interaction:* a directional edge that links an entity node to a process node or vice versa that indicates direction and the nature of the interaction. There are six possible connecting edges (bottom central panel) that represent the nature of the interaction between nodes. The first three (interaction, catalysis and action potential) operate identically within a Petri net and serve to carry tokens between entity and process nodes. The two inhibitor edges act to inhibit the flow of tokens through target process nodes albeit based upon different rules. The non-competitive inhibitor edge completely blocks token flow through the target transmission if any tokens are present on the inhibitor node. In contrast the competitive inhibitor edge works by deducting the number of

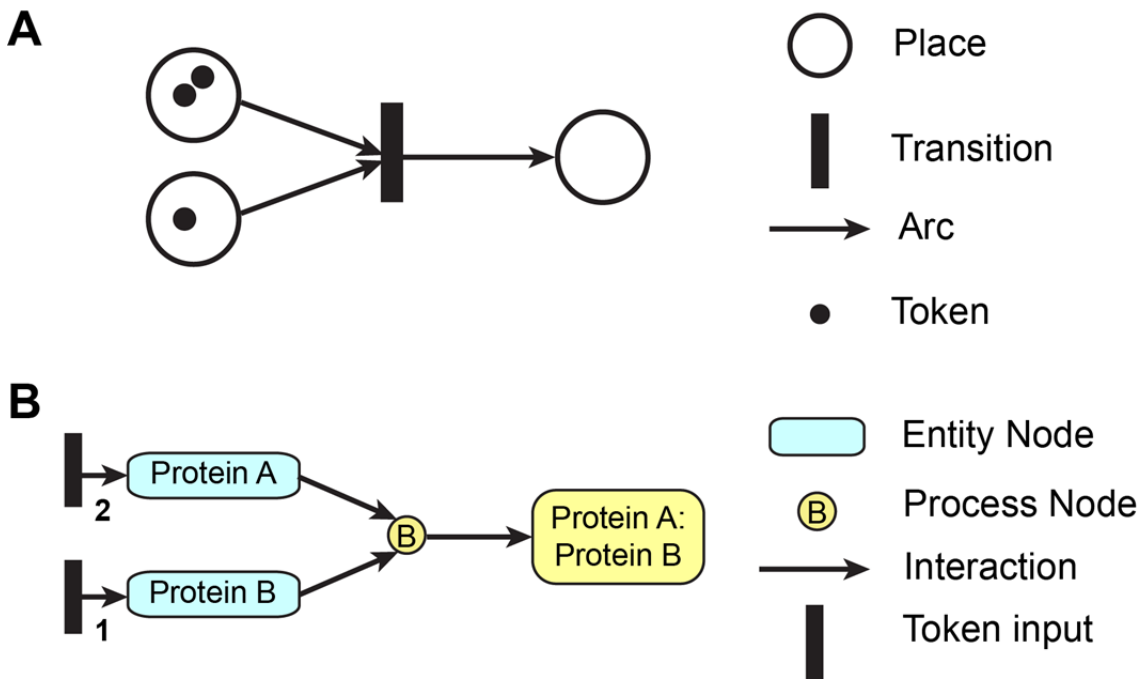
tokens residing on the inhibitor away from the number of tokens flowing through the target transition. Finally, the non-covalent interaction edge can be used to depict two separate entities within a complex. This may be a useful when describing large complexes, but these edges do not operate within the context of a Petri net simulation.

- *Cellular compartment:* define where pathway components reside and interactions take place, such as an organelle (e.g. mitochondrion, nucleus) or a transient cellular compartment (e.g. vesicle). In the diagrams they are shown as large coloured nodes that sit behind the interaction model (right panel).

End of Box 1

BOX 2: Petri nets to model biological systems

Petri nets (panel A) are a mathematical approach for describing distributed systems and have been used extensively in the modelling of many different kinds of systems including biological pathways. There are numerous types of Petri net algorithms and software that support modelling using them. The Petri net algorithm employed here was first developed and described by Ruths *el al.*[37], a modification of a synchronized Petri net model and firing policy, they called the signalling Petri net (SPN). Petri nets share a number of common features. Models are constructed as directional bipartite networks where nodes are considered to be either 'places' or 'transitions' connected by 'arcs'. Places usually represent an entity or state and by convention are represented as a white circle. Transitions represent interactions between entities or the transition of an entity from one state to another and are usually represented as a black rectangle. Arcs are directional arrows that connect places to transitions and *vice versa*. The availability of an entity and its abundance can be represented by the initial placement of tokens. The flow of information through the network is represented using tokens that move between places through transitions, following in the direction of the arcs. In the context of biological pathways paces represent pathway components, transitions correspond to processes that modify the components in some way, such as phosphorylation, binding etc., and are referred to as 'process nodes'. The interactions between molecules are depicted by edges, equivalent to arcs in Petri net parlance. Panel B shows how the notation for representing pathways using mEPN map on to Petri nets.



801

802 **Rules determining token flow through transitions.** Activity flow is represented by the movement of
803 tokens between places. The state of each place (component) is determined by the number of tokens held
804 by it. When a transition is fired, tokens are moved from each input place and redistributed downstream,
805 the transition acting as rule-based controller of flow. A transition will pass on tokens only if all the input
806 places contain tokens and where one input has less tokens than others, the passage of tokens will be
807 governed by the input place holding the least number of tokens. In the case of the Petri net algorithm
808 employed here, the movement of tokens is also stochastic. This is because during a time block all
809 transitions in a model are fired once but in a random order, and the number of tokens taken forward when
810 a transition is fired will be a random number between zero and the maximum available (although we have
811 implemented other versions of this rule, see step 13). Due to the stochastic nature of token flow, a
812 simulation usually comprises of a number of runs, the answer being based on the average token flow
813 across runs. Furthermore, when a transition is fired the number of tokens moved forward through the
814 transition will be subtracted from the amount available on the input places. One innovation not found in
815 most other Petri net simulators, is our implementation of a consumptive transition mode. When running in
816 this mode (for us this is standard), a constant input of tokens is applied to an entity node throughout a
817 simulation, but token levels on the node remain constant (unless others are fed in from another source),

as tokens are lost from it at the same rate they are added even when there is no flow through the target transition. In this way places representing entities such as enzymes (or indeed any other molecule) can receive a constant input of tokens throughout a simulation without accumulating or losing tokens.

Modelling time. Time in Petri nets is measured in abstract units called time blocks. When constructing a model, it is useful to consider how many experimental seconds, minutes, or hours correspond to a time block. Timing depends on the network topology and the further away a node is from the start of flow the longer it will take tokens to reach it. To simulate such time delays users can create a linear network that alternates transitions with spacer nodes multiple times. When tokens are passed through such a linear network the number of output tokens corresponds to the input but the time taken for tokens to reach the end is proportional to the number of spacer nodes (Figure 4B).

End of Box 2

BOX 3: Component Annotation

Multiple names are frequently employed to describe molecular species. This is particularly the case for one of the main components of biological pathways: proteins. Any given protein may be referred to in the literature by a number of different names concurrently. The use of non-standard nomenclature frequently leads to confusion in written texts and diagrams. If the naming of pathway components is not clear, then uncertainty arises as to what exactly is being depicted in a diagram and it ends up representing little more than a series of abstract concepts.

Our models have generally been focused on human pathways and we have used standard Human Gene Nomenclature Committee (HGNC) names to label nodes representing genes and proteins (www.genenames.org). This and related nomenclature systems such as the Mouse Genome Database (MGD) standard (http://www.informatics.jax.org/mgihome/nomen/), now provide a near complete annotation of all human and mouse genes, and their use in the naming of proteins provides a direct link between the identity of the gene and the corresponding protein. Of course, not everyone has adopted these naming systems so where other names (aliases) are in common use, these names are often included as part of the node's label after the official gene symbol in rounded brackets, but generally only on its first appearance in the pathway. Use of standard nomenclature also assists in the comparison and overlay of experimental data (which is usually annotated using standard gene nomenclature) onto pathway models. At the present time there are no standard and universally recognized nomenclature systems available for naming certain types of pathway components. For instance, protein isoforms tend to be named in an *ad hoc* manner by those who study them, and biochemical compounds are known by both their common names or by names that reflect their chemical composition. The IUPAC (www.iupac.org/) provides a standard nomenclature system for organic chemicals, but most names would have little relevance to a biologist. In cases such as these, the important thing is to be consistent and, where possible, to cross-reference the component's ID to other sources such that the identity of the component depicted, where at all possible, is unambiguous. We have used the excellent ChemSpider resource (www.chemspider.com/) as a reference for the naming of biochemical entities, although other resources, e.g., ChEMBL (www.ebi.ac.uk/chembl/) are potentially equally good. Using the node

properties dialogue [F6], nodes may be hyperlinked to external web resources and additional notes to nodes can be added using in the Data tab within yEd.

Protein state: The particular 'state' of an individual protein may determine its functional activity. With mEPN, a component's state is indicated as a text addition to the node label using square brackets following the component's name; each modification being placed in separate brackets. The system can be used to describe a wide range of protein modifications like phosphorylation [P], acetylation [Ac], ubiquitination [Ub] etc., and where details of the site of modification are known this may be represented, e.g., [P@L232] = phosphorylation at leucine 232.

Protein complexes: Names of the components are given as a concatenation of the proteins belonging to the complex, separated by a colon. If a complex is commonly referred to by a generic name this may be shown below the constituent parts in rounded brackets. Where a specific protein is present multiple times within a complex, this may be represented by placing the number of times the protein is present within the complex in angle brackets i.e., <n>. A node representing a component may be coloured to impart visual information on the component's type, e.g. to differentiate between a protein and a complex. Similarly, other types of pathway components may be represented using a range of shapes and colours – see palette (BOX 1, downloadable as Supplementary Data 1) for list of glyphs used to represent different entity types. A component may only be shown once in any given cellular compartment (in a given state). A component may however alter from one state to another, e.g., inactive to active, unbound to bound, in which case both forms are represented as separate entities. A different state may be indicated by including the name in square brackets, as described above.

End of Box 3

BOX 4: Layout Optimisation

There is a part of pathway modelling that could be considered art, or at least creative cartography. When starting a diagram the number of components is small, and visual comprehension of the system of nodes and edges is relatively easy. However, this situation soon changes as a diagram grows, and one of the greatest challenges is to render the inherently complex connections between components of a network model in a human readable form. This necessitates the careful placement of nodes and edges in the network layout. There are large number of layout algorithms available for network visualization but unfortunately none come close to the results achievable by a skilled human curator. Certain rules can be applied to this process to aid readability of the model:

- Models should be constructed, where possible, along a horizontal or vertical axis, arranged top down or left to right in the direction of information flow.
- Nodes should be evenly spaced and aligned along the chosen axis of layout but within the cellular compartment in which they reside.
- Crossing over of edges should be kept to a minimum and changes in edges direction should be avoided when possible. When multiple edges run parallel to each other it is important to keep the lines straight to maintain an easy-to-follow diagram.
- Space can be organised effectively by structuring sub-pathways into modules.
- Modules of the pathway should be arranged so that the connected glyphs are in close proximity, to minimise overlapping of connective edges.
- Hierarchical relationships between components should be shown in the layout of interactions. To do this, an orientation of pathway flow is chosen (e.g. left to right or top to bottom) and should be maintained throughout the diagram where possible.

However, each diagram is essentially unique and each comes with its own challenges. There is no one solution that fits all models so the layout of the diagram must be able to be easily adapted to take in new components and concepts whilst maintaining its readability.

End of Box 4

913

BOX 5: Glossary

- **Arc:** By convention in Petri net parlance, arcs are the directional edges that connect a place to a transition or vice versa, (but never between places or between transitions). Arcs are referred to as **edges** in mEPN.
- **Edge**: In mEPN notation, edge is used to refer to any line used to connect entity and process nodes to indicate an interaction (activating or inhibitory), and also the direction of that interaction. The Petri net equivalent is an **arc**.
- **Entity node:** Entity nodes are glyphs that represent pathway components such as molecules or genes. The Petri net equivalent is a **place**.
- **Glyph**: a visual representation of an entity, process or transition node.
- **GraphML:** a XML-based file format used to describe the structural and visual properties of a model.
- **Layout**: the way in which nodes and edges are set out in 2D or 3D space (physical topology).
- **Model**: the visual representation of a network.
- **mEPN:** modified Edinburgh Pathway Notation scheme is a graphical notation system based on the concepts of the process diagram.
- **Network**: a number of nodes connected by edges.
- **Notation scheme**: a system of symbols used to represent biological entities and interactions between them in a semantically and visually unambiguous manner.
- **Parameterization:** defining the initial state of the system through the placement of tokens.
- **Petri net:** a directed bipartite graph that alternates places (entities) and transitions (events that occur).
- **Place:** In Petri nets, places represent possible states of the system. They are referred to as **entity nodes** in mEPN notation.
- **Process diagram:** a diagram used to formally describe the components of a system, their activation state and the interactions between them.

- **Process node**: In mEPN notation, process nodes are glyphs that represent and define interactions between entities or the transition of an entity from one state to another. The Petri net equivalent is a **transition**.
- **SPN:** Signalling Petri Net as defined by Ruths *et al* [37]*.* Please note, in the context of Petri nets SPN is also often used to refer to Stochastic Petri Nets, a class of Petri net algorithms used to model the stochastic flow of tokens, as in the case here.
- **Tokens:** Tokens represent quantitative information that is introduced and distributed through a Petri net. Here they can be thought of representing the amount and/or activity of a pathway component.
- **Topology**: arrangement of various elements of the pathway (nodes, edges, etc.) that illustrates how information flows within the network.
- **Transition:** In Petri nets, transitions are events or actions. They are represented as **process nodes** in mEPN notation.

End of Box 5

Acknowledgements

We are grateful to Paul Digard and David Hume for helpful discussions and advice and thank Athanasios Theocharidis for all his work on developing BioLayout *Express*^{3D}. The Hedgehog signalling pathway model (Suppl. File 6) was generated by Miriam Graute, a University of Edinburgh final year BSc student as part of a 10 week elective course. We also thank the BBSRC who funded the development of the BioLayout *Express*^{3D} (BB/F003722/1 and BB/I001107/1) and T.C.F. is supported by an Institute Strategic Program Grant on Transcriptomes, Networks and Systems (BBS/E/D/20211552).

Conflict of Interest

The authors declare competing financial interests: details are available in the online version of the paper. There is now a commercial and supported version of BioLayout *Express*^{3D} called Miru, produced by Kajeka Ltd, (Edinburgh, UK) that possesses all the functionality described here for pathway modelling. T.C.F. is a founder and director of Kajeka.

973 **Table 1: A list of resources useful for the compilation of pathway diagrams**

Literature Databases	
NCBI Pubmed	http://www.ncbi.nlm.nih.gov/pubmed/
Web of Science	http://wok.mimas.ac.uk/
Google Scholar	http://scholar.google.co.uk/
Scopus	http://www.scopus.com/
iHop	http://www.ihop-net.org/
Component Annotation	
NCBI Entrez Gene	http://www.ncbi.nlm.nih.gov/sites/entrez
Gene Cards	http://www.genecards.org/
Gene Ontology	http://www.geneontology.org/GO.downloads.annotations.shtml
PubChem	http://pubchem.ncbi.nlm.nih.gov/
Chemspider	http://www.chemspider.com
Interaction Databases	
ConsensusPathDB	http://cpdb.molgen.mpg.de/
BioGRID	http://thebiogrid.org/
IntAct	http://www.ebi.ac.uk/intact/
GeneMANIA	http://www.genemania.org/
Human Protein Reference Database	http://www.hprd.org/index_html
MINT	http://mint.bio.uniroma2.it/mint/Welcome.do
STRING	http://string-db.org/
DIP	http://dip.doe-mbi.ucla.edu/dip/Main.cgi
MIPS CORUM	http://mips.helmholtz-muenchen.de/genre/proj/corum

Pathway Repositories	
KEGG	http://www.genome.jp/kegg/
Reactome	http://www.reactome.org/
Biocarta	http://www.biocarta.com/genes/index.asp
WikiPathways	http://wikipathways.org/index.php/WikiPathways

974

975

References

1 O'Hara, L. *et al.* Modelling the Structure and Dynamics of Biological Pathways. *PLoS biology* **14**, e1002530, doi:10.1371/journal.pbio.1002530 (2016).

2 Raza, S. *et al.* A logic-based diagram of signalling pathways central to macrophage activation. *BMC systems biology* **2**, 36, doi:10.1186/1752-0509-2-36 (2008).

3 Freeman, T. C., Raza, S., Theocharidis, A. & Ghazal, P. The mEPN scheme: an intuitive and flexible graphical system for rendering biological pathways. *BMC systems biology* **4**, 65, doi:10.1186/1752-0509-4-65 (2010).

4 Kitano, H., Funahashi, A., Matsuoka, Y. & Oda, K. Using process diagrams for the graphical representation of biological networks. *Nat Biotechnol* **23**, 961-966 (2005).

5 Kohn, K. W., Aladjem, M. I., Weinstein, J. N. & Pommier, Y. Molecular interaction maps of bioregulatory networks: a general rubric for systems biology. *Mol Biol Cell* **17**, 1-13 (2006).

6 Moodie, S. L., Sorokin, A., Goryanin, I. & Ghazal, P. A Graphical Notation to Describe the Logical Interactions of Biological Pathways. *Journal of Integrative Bioinformatics* **3**, 11 (2006).

7 Novere, N. L. *et al.* The systems biology graphical notation. *Nat Biotechnol* **27**, 735-741, doi:nbt.1558 [pii] 10.1038/nbt.1558 (2009).

8 Lopez, C. F., Muhlich, J. L., Bachman, J. A. & Sorger, P. K. Programming biological models in Python using PySB. *Mol Syst Biol* **9**, 646, doi:10.1038/msb.2013.1 (2013).

9 Beltrame, L. *et al.* The Biological Connection Markup Language: a SBGN-compliant format for visualization, filtering and analysis of biological pathways. *Bioinformatics* **27**, 2127-2133, doi:10.1093/bioinformatics/btr339 (2011).

10 Calzone, L., Gelay, A., Zinovyev, A., Radvanyi, F. & Barillot, E. A comprehensive modular map of molecular interactions in RB/E2F pathway. *Mol Syst Biol* **4**, 173, doi:msb20087 [pii] 10.1038/msb.2008.7 (2008).

11 Kuperstein, I. *et al.* Atlas of Cancer Signalling Network: a systems biology resource for integrative analysis of cancer data with Google Maps. *Oncogenesis* **4**, e160, doi:10.1038/oncsis.2015.19 (2015).

12 Oda, K. & Kitano, H. A comprehensive map of the toll-like receptor signaling network. *Mol Syst Biol* **2**, 2006 0015 (2006).

13 Oda, K., Matsuoka, Y., Funahashi, A. & Kitano, H. A comprehensive pathway map of epidermal growth factor receptor signaling. *Mol Syst Biol* **1**, 2005 0010 (2005).

14 Raza, S. *et al.* Construction of a large scale integrated map of macrophage pathogen recognition and effector systems. *BMC systems biology* **4**, 63, doi:10.1186/1752-0509-4-63 (2010).

15 Wentker, P. *et al.* An Interactive Macrophage Signal Transduction Map Facilitates Comparative Analyses of High-Throughput Data. *Journal of immunology* **198**, 2191-2201, doi:10.4049/jimmunol.1502513 (2017).

16 Matsuoka, Y., Funahashi, A., Ghosh, S. & Kitano, H. Modeling and simulation using CellDesigner. *Methods Mol Biol* **1164**, 121-145, doi:10.1007/978-1-4939-0805-9_11 (2014).

17 Demir, E. *et al.* The BioPAX community standard for pathway data sharing. *Nat Biotechnol* **28**, 935-942, doi:10.1038/nbt.1666 (2010).

18 Kanehisa, M., Goto, S., Sato, Y., Furumichi, M. & Tanabe, M. KEGG for integration and interpretation of large-scale molecular data sets. *Nucleic Acids Res* **40**, D109-114, doi:10.1093/nar/gkr988 (2012).

19 Wu, G., Dawson, E., Duong, A., Haw, R. & Stein, L. ReactomeFIViz: a Cytoscape app for pathway and network-based data analysis. *F1000Res* **3**, 146, doi:10.12688/f1000research.4431.2 (2014).

20 Yamada, T., Letunic, I., Okuda, S., Kanehisa, M. & Bork, P. iPath2.0: interactive pathway explorer. *Nucleic Acids Res* **39**, W412-415, doi:10.1093/nar/gkr313 (2011).
21 Czauderna, T., Klukas, C. & Schreiber, F. Editing, validating and translating of SBGN maps. *Bioinformatics* **26**, 2340-2341, doi:10.1093/bioinformatics/btq407 (2010).
22 Croft, D. *et al.* The Reactome pathway knowledgebase. *Nucleic Acids Res* **42**, D472-477, doi:10.1093/nar/gkt1102 (2014).
23 Joshi-Tope, G. *et al.* Reactome: a knowledgebase of biological pathways. *Nucleic Acids Res* **33**, D428-432, doi:33/suppl_1/D428 [pii] 10.1093/nar/gki072 (2005).
24 Kuperstein, I. *et al.* NaviCell: a web-based environment for navigation, curation and maintenance of large molecular interaction maps. *BMC systems biology* **7**, 100, doi:10.1186/1752-0509-7-100 (2013).
25 Mi, H. & Thomas, P. PANTHER pathway: an ontology-based pathway database coupled with data analysis tools. *Methods Mol Biol* **563**, 123-140, doi:10.1007/978-1-60761-175-2_7 (2009).
26 Hucka, M. *et al.* The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics* **19**, 524-531 (2003).
27 Bause, F. & Kritzinger, P. S. *Stochastic Petri nets: An introduction to the theory*. (Vieweg+Teubner, 1996).
28 Reddy, V. N., Mavrovouniotis, M. L. & Liebman, M. N. Petri net representations in metabolic pathways. *Proceedings. International Conference on Intelligent Systems for Molecular Biology* **1**, 328-336 (1993).
29 Bahi-Jaber, N. & Pontier, D. Modeling transmission of directly transmitted infectious diseases using colored stochastic Petri nets. *Math Biosci* **185**, 1-13, doi:10.1016/S0025-5564(03)00088-9 (2003).
30 Chaouiya, C. Petri net modelling of biological networks. *Brief Bioinform* **8**, 210-219, doi:10.1093/bib/bbm029 (2007).
31 Heiner, M., Koch, I. & Will, R. Model validation of biological pathways using Petri nets - demonstrated for apoptosis. *Biosystems* **75**, 15-28, doi:10.1016/j.jbiosystems.2004.03.003 (2004).
32 Peleg, M., Rubin, D. & Altman, R. B. Using Petri net tools to study properties and dynamics of biological systems. *J Am Med Inform Assn* **12**, 181-199, doi:10.1197/jamia.M1637 (2005).
33 Taubner, C., Mathiak, B., Kupfer, A., Fleischer, N. & Eckstein, S. Modelling and simulation of the TLR4 pathway with coloured petri nets. *Conf Proc IEEE Eng Med Biol Soc* **1**, 2009-2012, doi:10.1109/IEMBS.2006.259902 (2006).
34 Balazki, P., Lindauer, K., Einloft, J., Ackermann, J. & Koch, I. MONALISA for stochastic simulations of Petri net models of biochemical systems. *BMC Bioinformatics* **16**, 215, doi:10.1186/s12859-015-0596-y (2015).
35 Marwan, W., Rohr, C. & Heiner, M. Petri Nets in Snoopy: A Unifying Framework for the Graphical Display, Computational Modelling, and Simulation of Bacterial Regulatory Networks. *Bacterial Molecular Networks: Methods and Protocols* **804**, 409-437, doi:10.1007/978-1-61779-361-5_21 (2012).
36 Ramos, H. *et al.* The Protein Information and Property Explorer 2: gaggle-like exploration of biological proteomic data within one webpage. *Proteomics* **11**, 154-158, doi:10.1002/pmic.201000459 (2011).
37 Ruths, D., Muller, M., Tseng, J. T., Nakhleh, L. & Ram, P. T. The signaling petri net-based simulator: a non-parametric strategy for characterizing the dynamics of cell-specific signaling networks. *PLoS Comput Biol* **4**, e1000005, doi:10.1371/journal.pcbi.1000005 (2008).
38 Li, C. *et al.* Structural modeling and analysis of signaling pathways based on Petri nets. *Journal of bioinformatics and computational biology* **4**, 1119-1140 (2006).

39 David, R. & Alla, H. Discrete, Continuous, and Hybrid Petri Nets, Second Edition. *Discrete, Continuous, and Hybrid Petri Nets, Second Edition*, 1-550, doi:10.1007/978-3-642-10669-9 (2010).
40 Theocharidis, A., van Dongen, S., Enright, A. J. & Freeman, T. C. Network visualization and analysis of gene expression data using BioLayout Express(3D). *Nat Protoc* **4**, 1535-1550, doi:10.1038/nprot.2009.177 (2009).
41 Polak, M. E., Ung, C. Y., Masapust, J., Freeman, T. C. & Ardern-Jones, M. R. Petri Net computational modelling of Langerhans cell Interferon Regulatory Factor Network predicts their role in T cell activation. *Sci Rep* **7**, 668, doi:10.1038/s41598-017-00651-5 (2017).
42 Di Ventura, B., Lemerle, C., Michalodimitrakis, K. & Serrano, L. From in vivo to in silico biology and back. *Nature* **443**, 527-533, doi:10.1038/nature05127 (2006).
43 de Jong, H. Modeling and simulation of genetic regulatory systems: a literature review. *J Comput Biol* **9**, 67-103, doi:10.1089/10665270252833208 (2002).
44 Le Novere, N. *et al.* The Systems Biology Graphical Notation. *Nat Biotechnol* **27**, 735-741, doi:10.1038/nbt.1558 (2009).
45 Friesen, W. O. & Block, G. D. What is a biological oscillator? *Am J Physiol* **246**, R847-853 (1984).
46 Pertsovskaya, I., Abad, E., Domedel-Puig, N., Garcia-Ojalvo, J. & Villoslada, P. Transient oscillatory dynamics of interferon beta signaling in macrophages. *BMC systems biology* **7**, 59, doi:10.1186/1752-0509-7-59 (2013).