Petri nets for multiscale systems biology

Simulation & Analysis

David Gilbert & Ovidiu Pârvu

School of Information Systems, Computing & Mathematics
Centre for Systems & Synthetic Biology
Brunel University, London UK

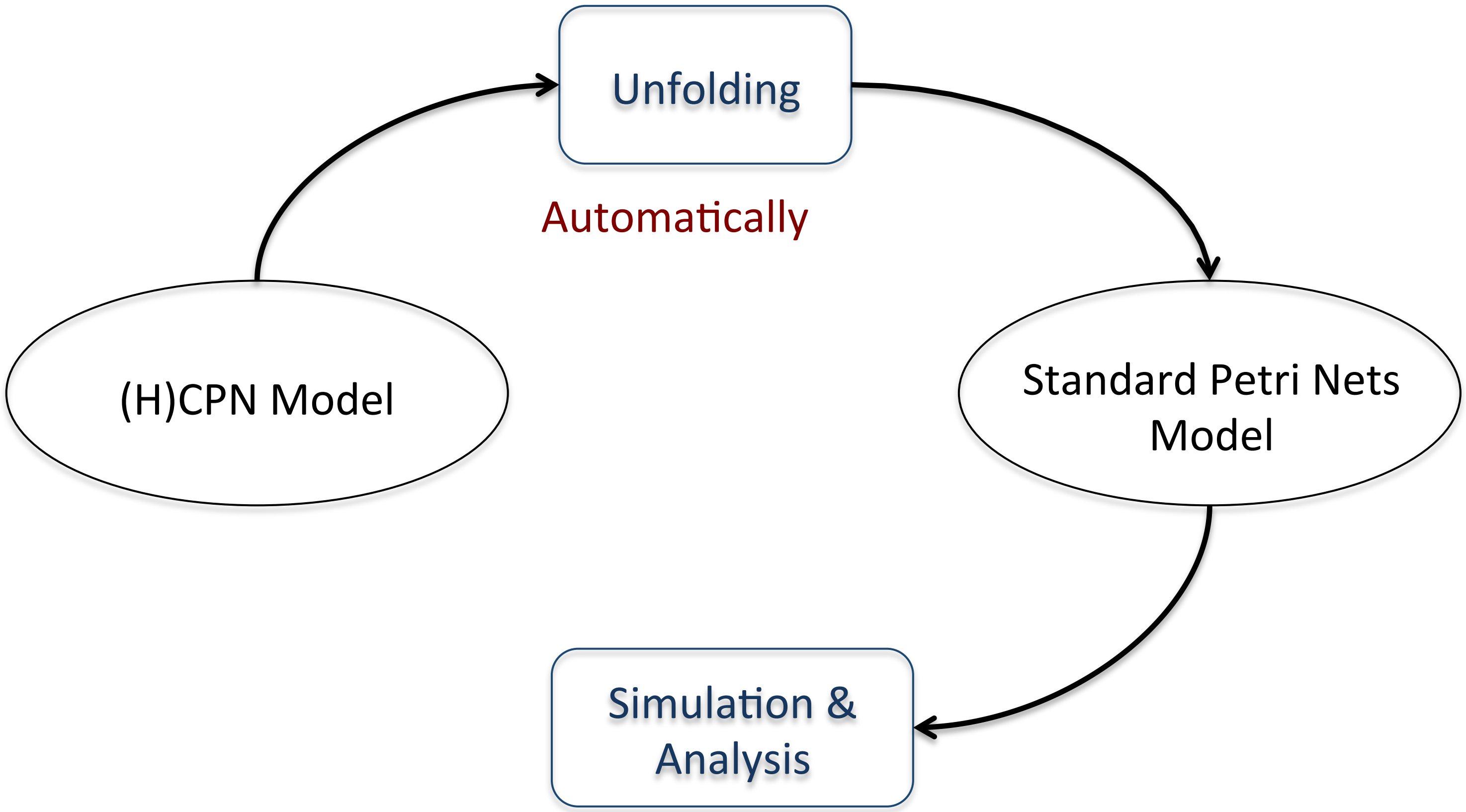
Contents

- Simulation & unfolding
- Analysis techniques:
  - Model checking in the 3 paradigms
  - Image analysis
  - Mathematical clustering

Modelling challenges

- Design & construction: model **hierarchy** over different spatial scales in a **compact** and **parameterised** manner
- Simulation: models comprise a very **large number** of underlying **ODEs, e.g.**
  - 800-cells: 164,000 ODEs/species & 229,000 reactions; more than 2 hours
- **Expensive** model fitting (parameter optimisation): large model & lacking data
  - Requires many repeated lengthy simulations
- How to **visualise**, **analyse** & **validate** multi-scale models
  - Comparison against semi-quantitative data

Unfolding, Simulation & Analysis (Snoopy)


Unfolding

Automatically

(H)CPN Model

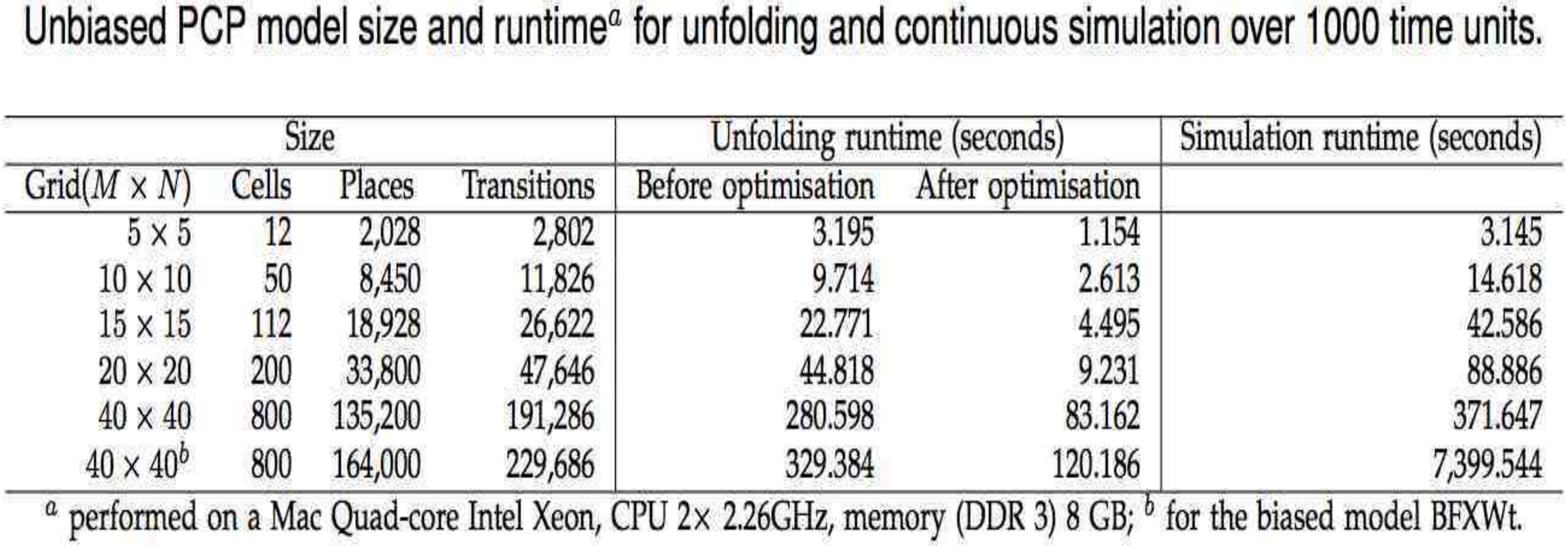
Standard Petri Nets Model

Simulation & Analysis

Unfolding - issues

- Have to unfold all possibilities
  - All combinations of the colour tuples over all the ranges of the corresponding colour types
- Expense of time & space for unfolding
- Can benefit by a constraints approach
- Computation time >> unfolding time
- Some scenarios better to simulate at folded level
  - Current challenge!

Some Statistics



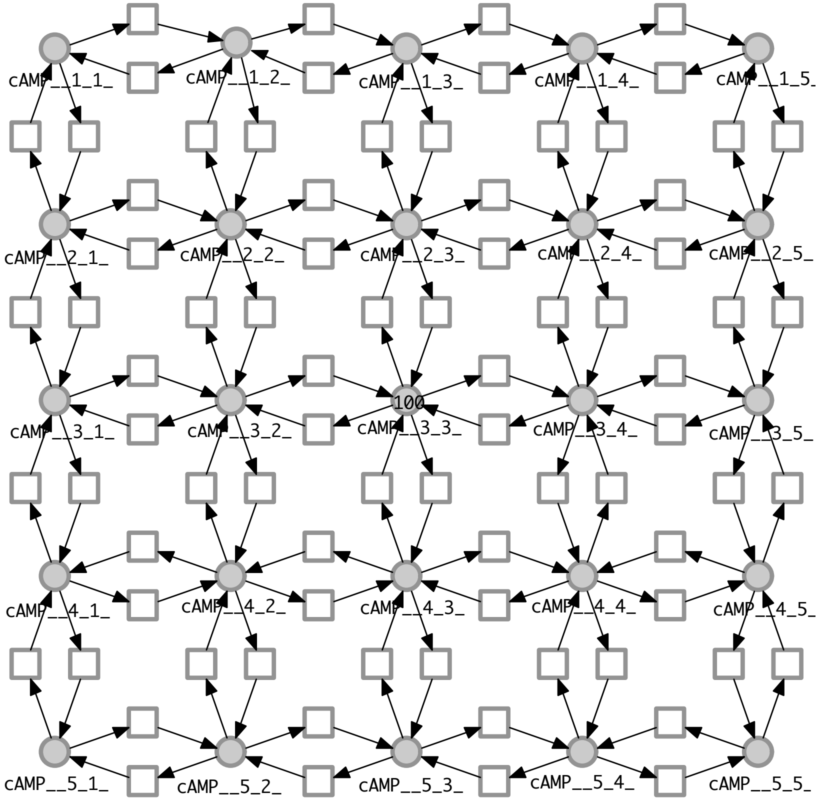
Unbiased PCP model size and runtime[a] for unfolding and continuous simulation over 1000 time units.

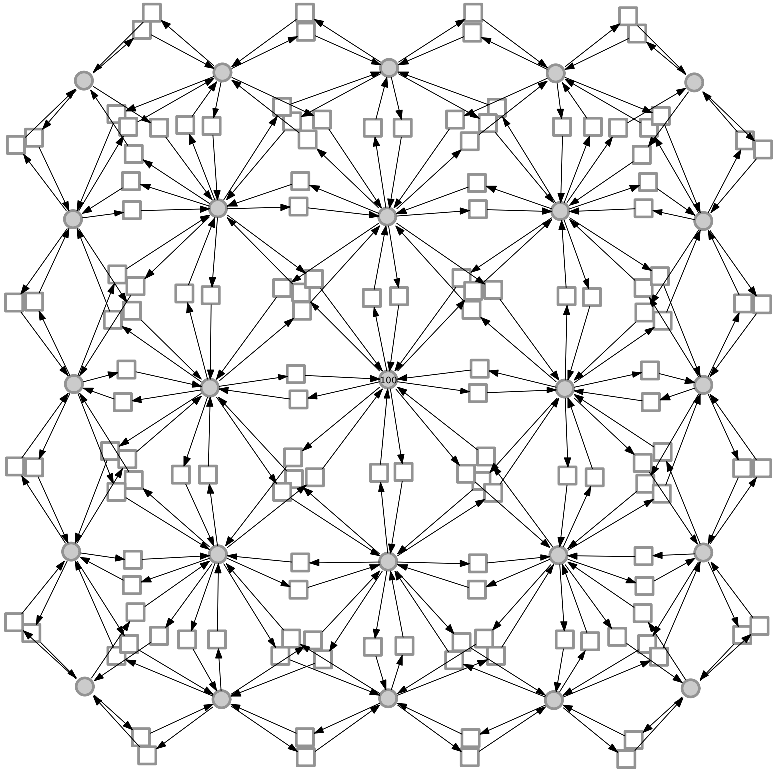
Size				Unfolding runtime (seconds)		Simulation runtime (seconds)
Grid($M \times N$)	Cells	Places	Transitions	Before optimisation	After optimisation	
5 × 5	12	2,028	2,802	3.195	1.154	3.145
10 × 10	50	8,450	11,826	9.714	2.613	14.618
15 × 15	112	18,928	26,622	22.771	4.495	42.586
20 × 20	200	33,800	47,646	44.818	9.231	88.886
40 × 40	800	135,200	191,286	280.598	83.162	371.647
40 × 40[b]	800	164,000	229,686	329.384	120.186	7,399.544

[a] performed on a Mac Quad-core Intel Xeon, CPU 2× 2.26GHz, memory (DDR 3) 8 GB; [b] for the biased model BFXWt.

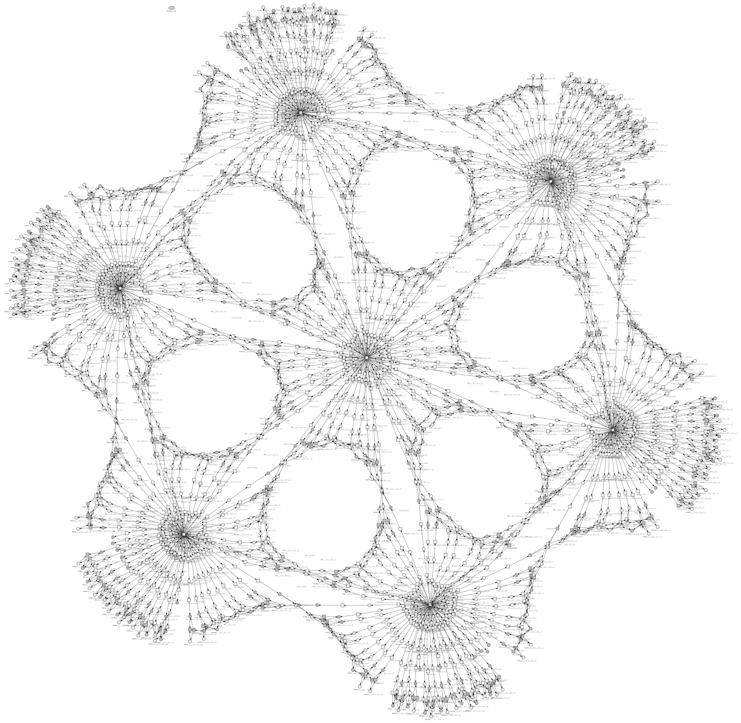
Constraint solver used for optimisation – enables larger size tissue to be simulated.

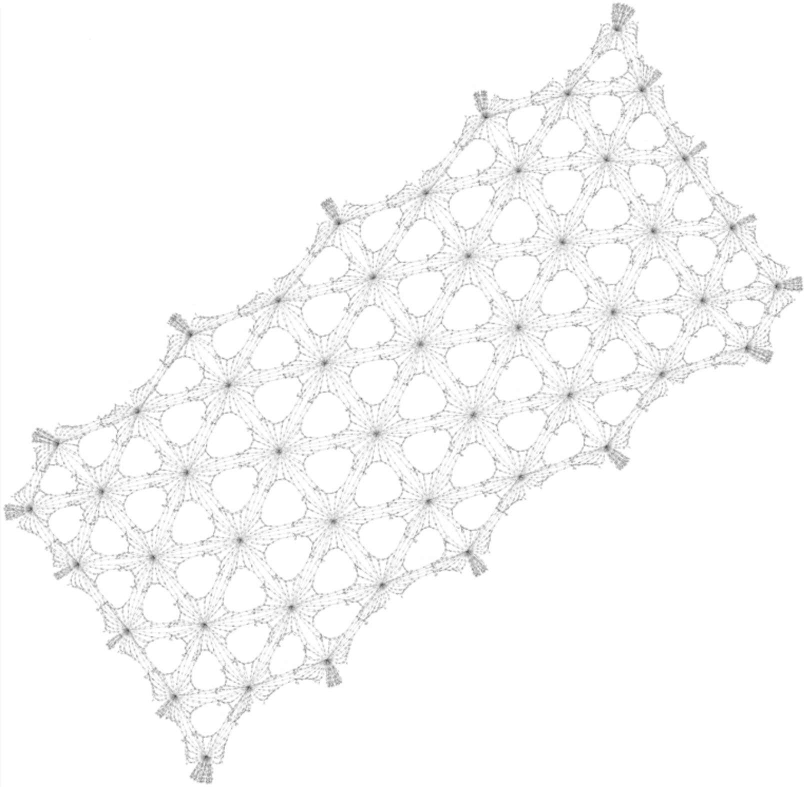
Simple example unfoldings





Example unfolding





Guess which model these are from!

A Machine Learning Approach for Generating Temporal Logic Classifications of Complex Model Behaviours

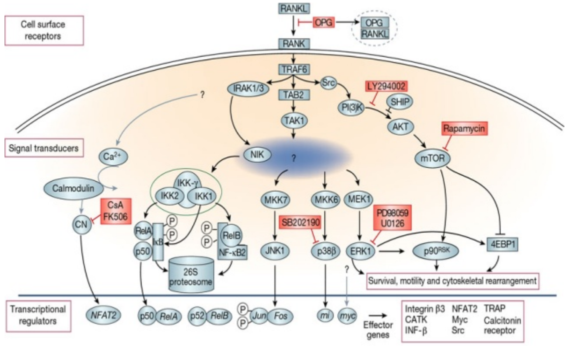
Daniele Maccagnola, Enza Messina,
Qian Gao and David Gilbert

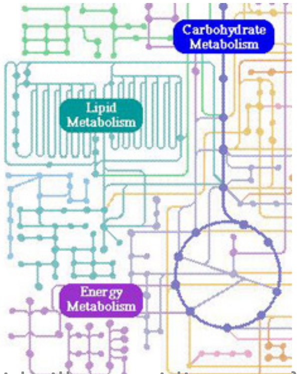
DIPARTIMENTO
DI INFORMATICA
SISTEMISTICA
E COMUNICAZIONE

Brunel
UNIVERSITY
LONDON

Understanding Biological Systems

Biological Systems





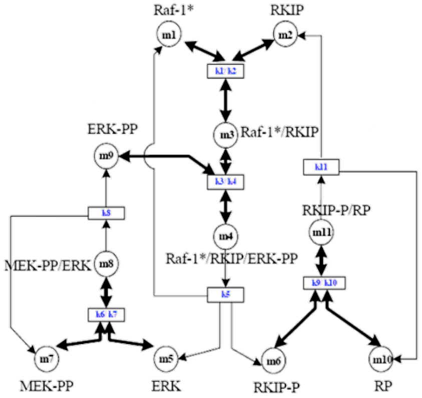
Mathematical Models

$$\frac{d[A]}{dt} = -k_1[A][E] + k_2[A|E]$$

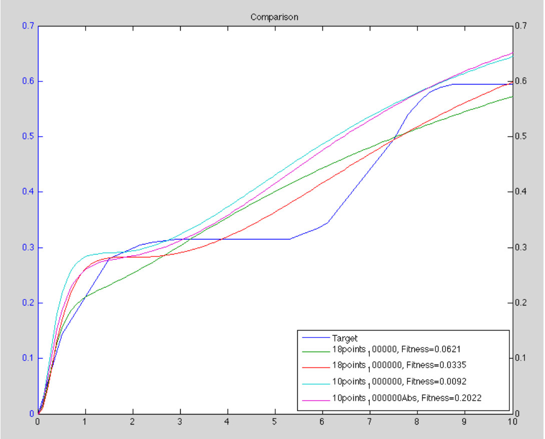
$$\frac{d[E]}{dt} = -k_1[A][E] + k_2[A|E] + k_3[A|E]$$

$$\frac{d[A|E]}{dt} = k_1[A][E] - k_2[A|E] - k_3[A|E]$$

$$\frac{d[B]}{dt} = k_3[A|E]$$



Simulation results



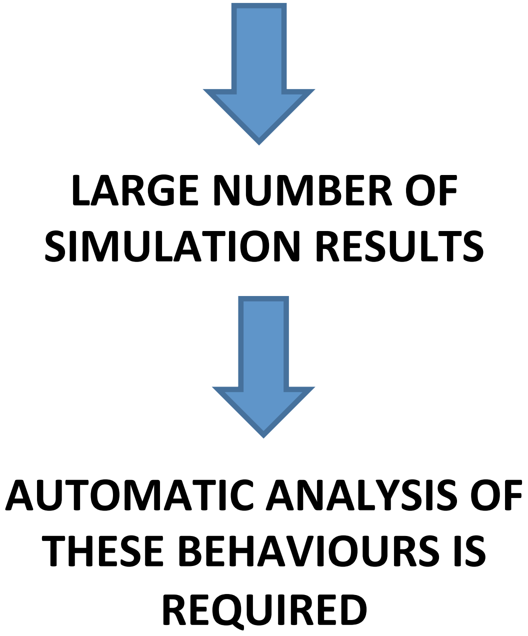
Some ideas for multiscale analysis

- Simulate model
  - many traces from different components
  - multidimensional (spatial) and multiscale (levels)
- Cluster results (from simulations)
- Analyse clusters to extract features
  - Behaviour (model) checking: how to generate properties?
    - Manually (by eye, or by ‘expected’ behaviour)
    - Automated generation

Understanding Biological Systems

- Mathematical models allows the *in-silico* investigation of behaviour of biological systems

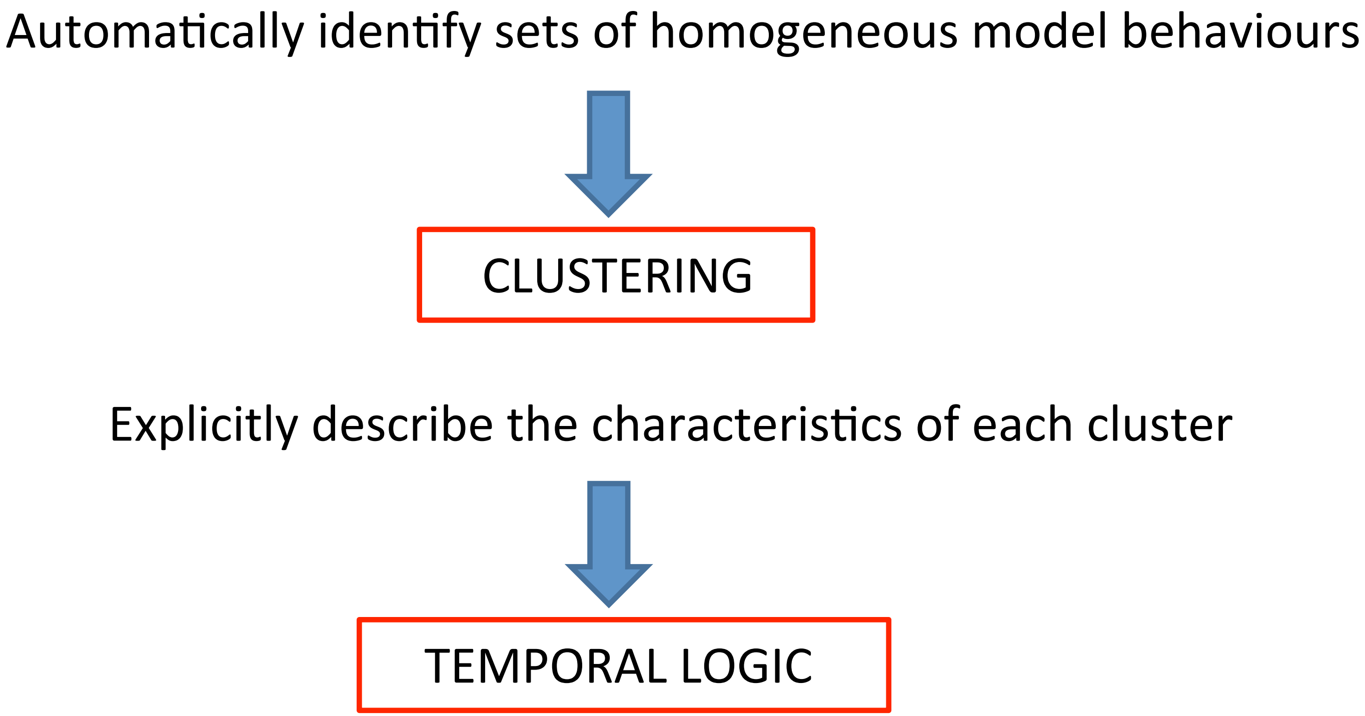
- Simulation of the model under different perturbation (parameter setting)



LARGE NUMBER OF SIMULATION RESULTS

AUTOMATIC ANALYSIS OF THESE BEHAVIOURS IS REQUIRED

Analysis and Interpretation of Time Series Data



Automatically identify sets of homogeneous model behaviours

CLUSTERING

Explicitly describe the characteristics of each cluster

TEMPORAL LOGIC

Analysis & Visualisation

- Clustering
  - DBScan
  - Hierarchical clustering
  - K-means
  - SOMs

- Model checking

Primary & Secondary Data

- Primary data

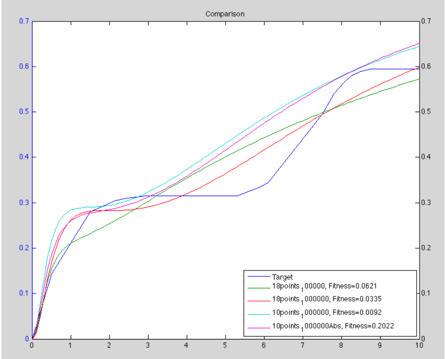
  Data obtained from simulating the model: time series of concentrations

- Secondary data

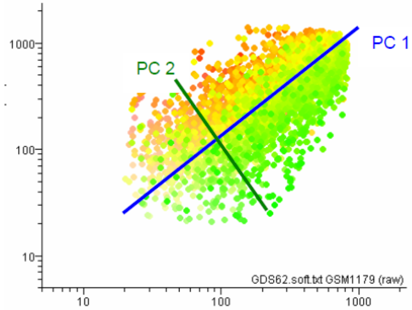
  Cumulative rewards: time series of accumulated concentrations

Clustering

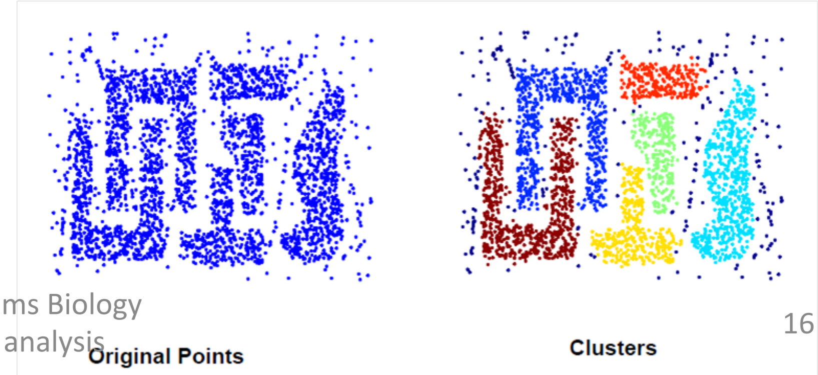
TIME SERIES RAW DATA



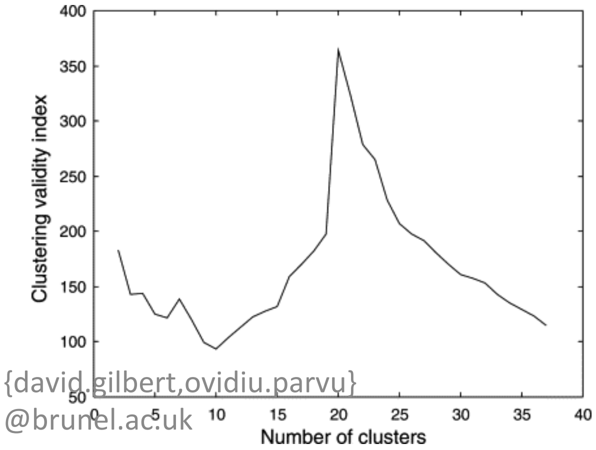
FEATURE SELECTION



DENSITY-BASED CLUSTERING



CLUSTER VALIDATION



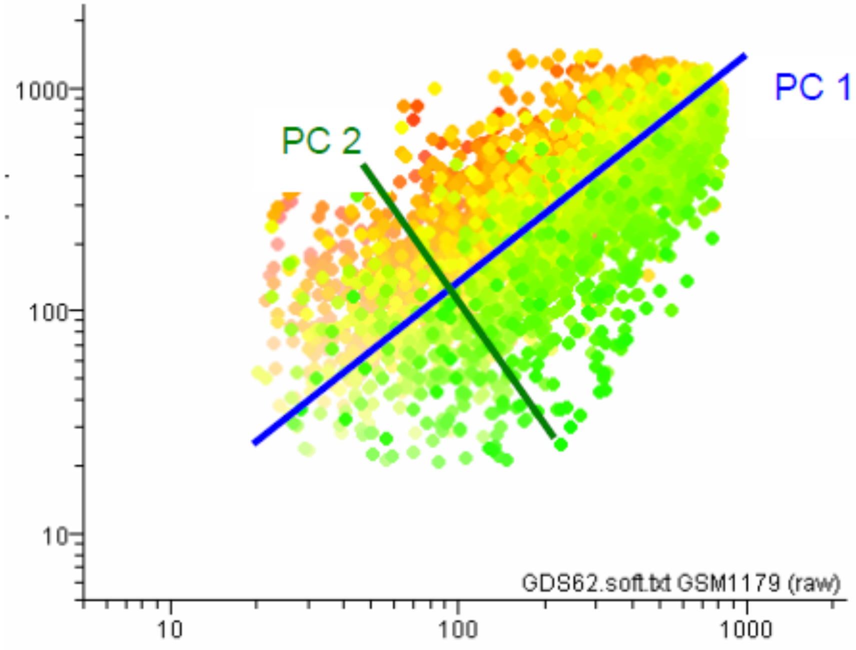
Feature Selection

PRINCIPAL COMPONENT ANALYSIS

- Principal Component Analysis (PCA) is a method to reduce data dimensionality

- Performs a covariance analysis between factors

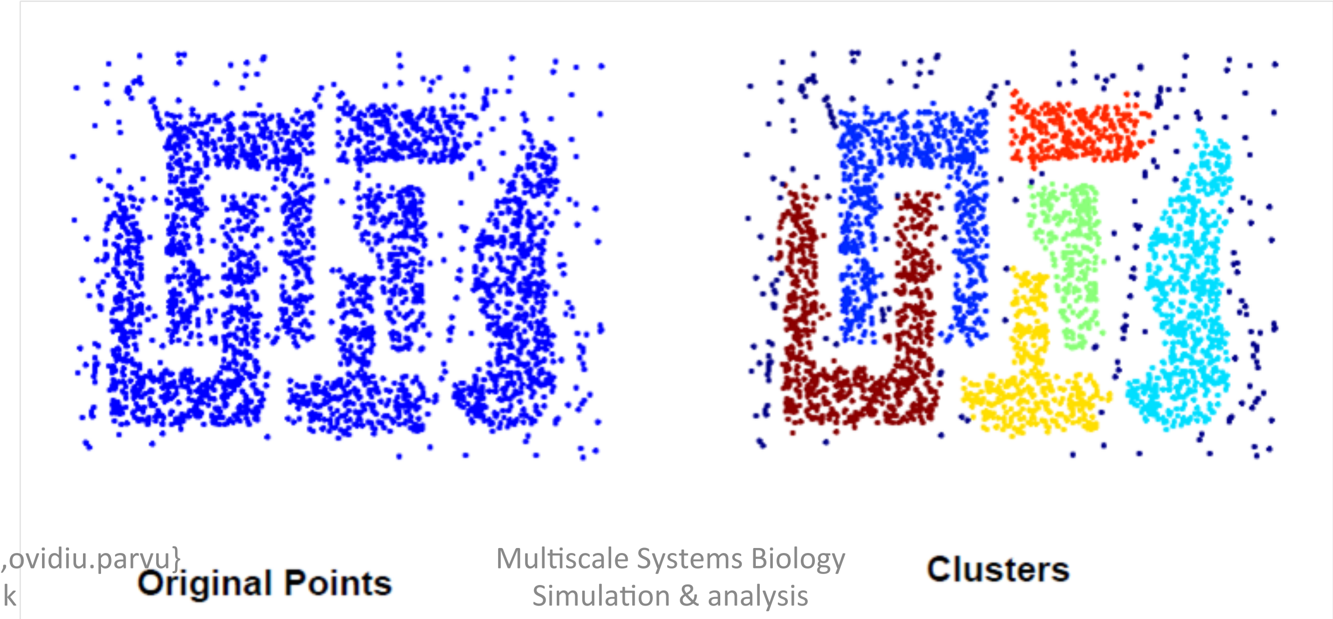
- Allows to reduce the number of dimension without much loss of information



Density-based Clustering

DBSCAN

- Analyse the space to find areas with high density of elements
- Each high-density area is labeled as a cluster
- Well suited to detect arbitrary-shaped clusters



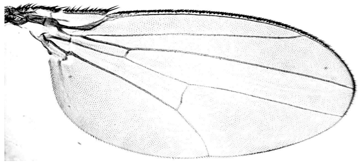
Cluster Validation

Composed Density Between and Within clusters (CDBW)

- Common evaluation indexes do not work well with clusters of arbitrary shapes (based on the concept of cluster center)

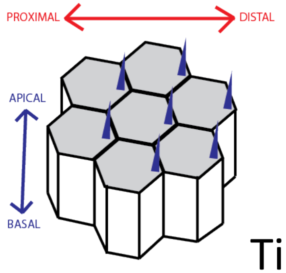
- CDBW measures the quality of clusters by considering multiple representatives per cluster

- CDBW evaluates different characteristics:
    - *compactness* (density within clusters)
    - *cohesion* (changes in density distribution within clusters)
    - *separation* (density among clusters)

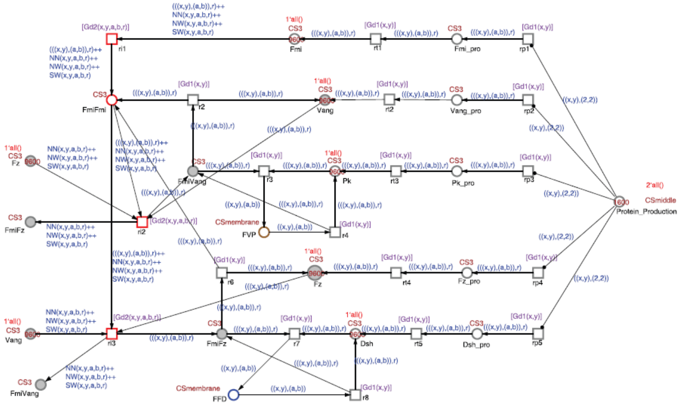


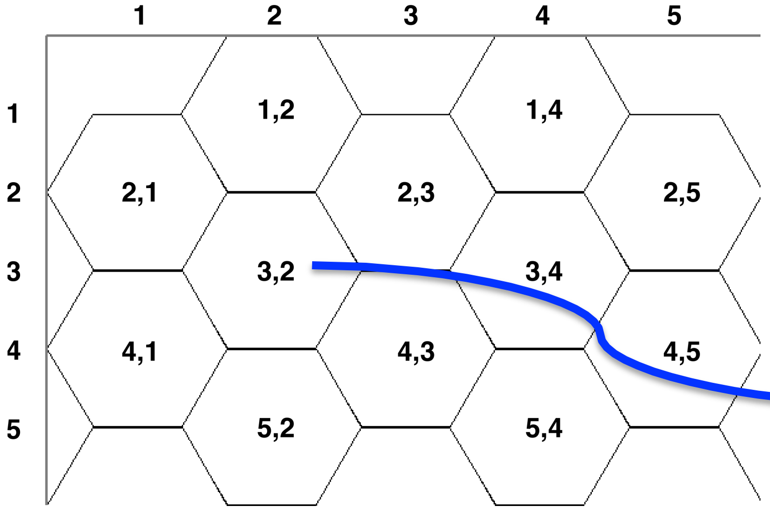
EXAMPLE

Planar Cell Polarity in Drosophila Wing

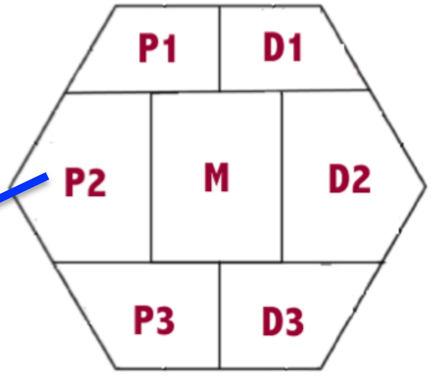


Tissue (Cells)





Cell: (3,2)
Compartment (2,



Colourset = {…, {((3,2)(1,1)), ((3,2)(2,1)), ((3,2)(3,1)),……((3,2)(3,3))}, …

Multiscale Case Study

- We want to cluster and characterize the behaviour of each cell in the tissue (a 15 x 15 hexagonal structure, for a total of 112 cells)

- The behaviour is determined by the dynamic of FFD complex in the six external compartments

- TWO CASES:

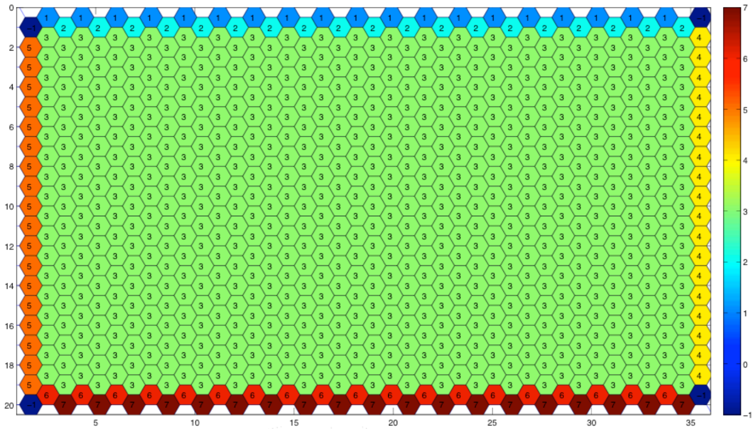
    - Wild type tissue: all the cells are "wild type"

    - Mutated tissue: there is a "clone" of mutated cells at the center of a wild type tissue

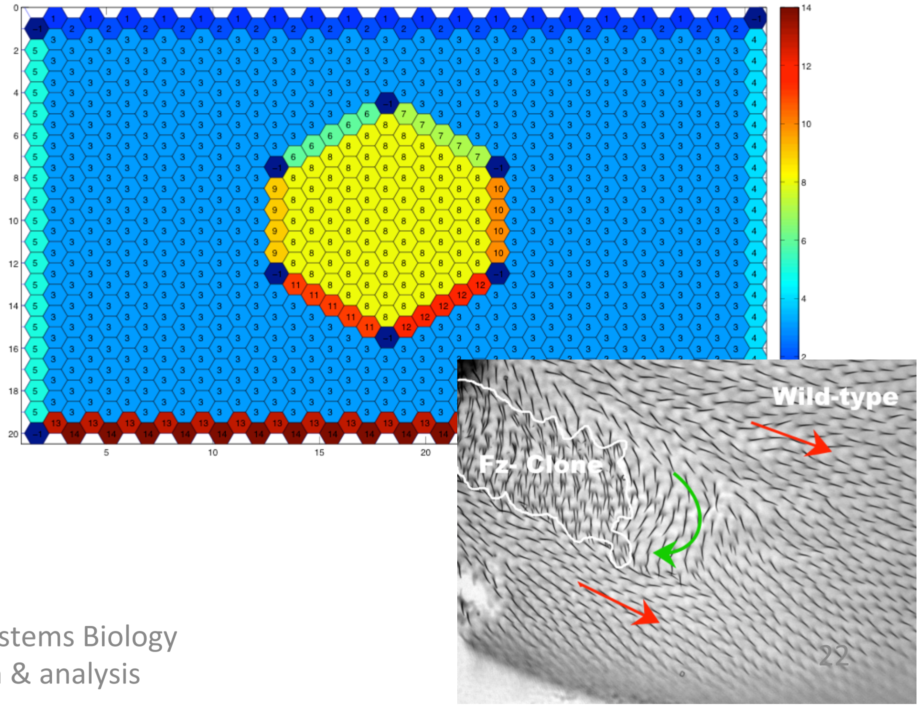
Clustering

- DBScan with Principal Component Analysis (PCA)

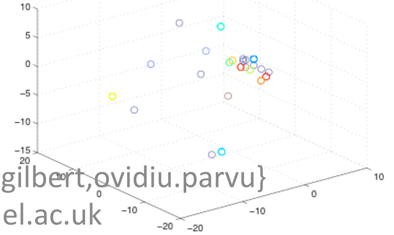
Unbiased model:
Grid 40*40 (800 cells)



Fz- mutant clone model:
A patch of mutated cells lacking Frizzled (Fz) in a wild-type background



Feature selection: PCA



Model Checking

In a sentence:

- "Formally check whether a model of a biochemical system does what we want"

Components:

- A model
  - the current description of a biochemical system of interest
- A property
  - a property which we think the system should have
- A model checker
  - a program to test whether the model has the property

To formally express time properties we use a temporal logic

"*I am hungry*."

"*I am always hungry*", "*I will eventually be hungry*",

"I will be hungry until I eat something".

Linear time logics restricted to single time line.

Branching logics can reason about multiple time lines.

"*There is a possibility that I will stay hungry forever*."

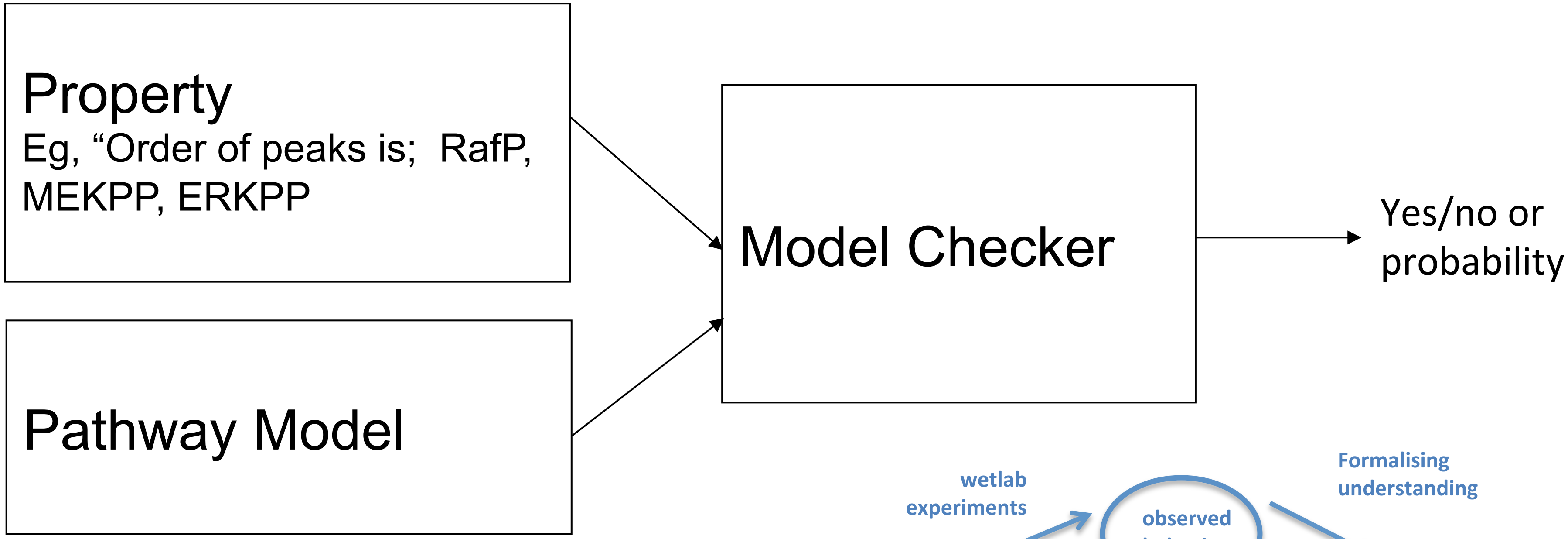
"*There is a possibility that eventually I am no longer hungry*."

Various logics :

- Computational Tree Logic (CTL)
- Continuous Stochastic Logic (CSL)
- Linear-time Temporal Logic (LTL)

each with different expressivity.

Model Checking Biochemical Pathways



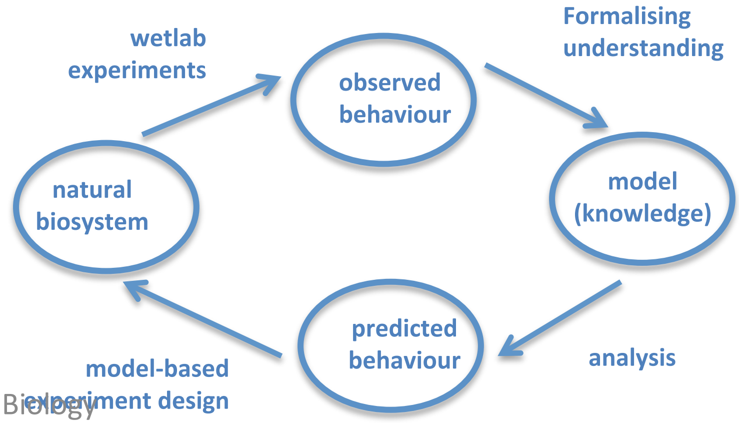
Property

Eg, “Order of peaks is; RafP, MEKPP, ERKPP

Pathway Model

Model Checker

Yes/no or probability



Analytical vs Simulative Model Checking

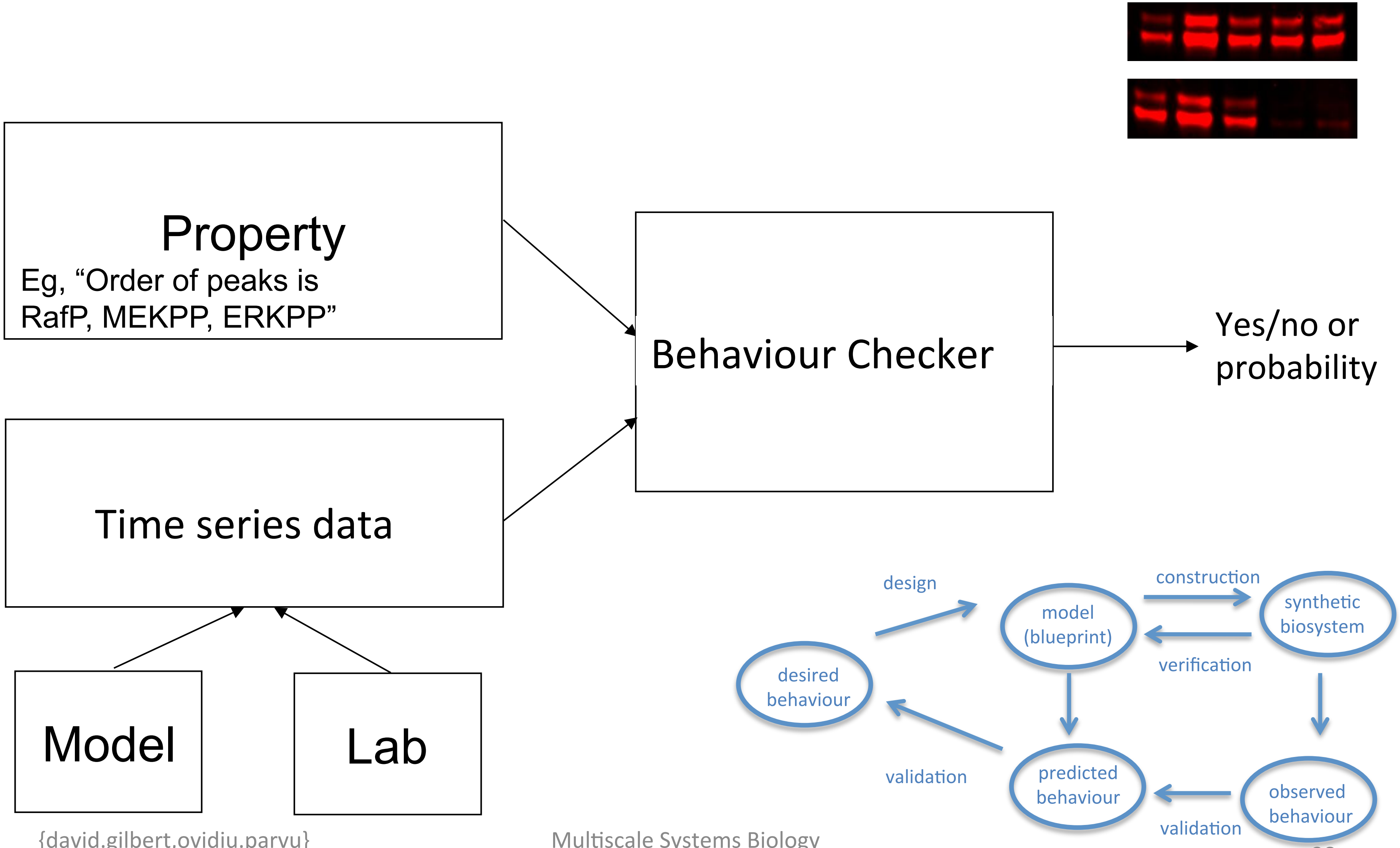
- **Analytical**:
  - Exact probabilities & prove properties
  - A model state is an association of #molecules/levels to each of the species
    - **Protein1** has 10 molecules & **Protein2** has 20 molecules
  - Analytical assesses every state that the model can be in (reachable states)
  - State space can grow even worse than exponentially with increasing molecules, or even be infinite!
  - Stochastic model checking with even as little as 12 molecules/levels can be impossible with today's technology

- **Simulative**:
  - Instead of analysing the constructed state space, analyse simulation outputs
  - Simulate the model X times and check these simulations
  - Simulation run = finite path through the state space
  - Can't prove probabilities

Simulative Model Checking

- **In-line**: check the observations as they arrive
  - Requires complex computational machinery: ‘combine’ simulator & model checker
  - Good for biochemical observations
  - Don’t always need to finish the experimental run
- **Off-line**: check the observations after all have been generated
  - Easier to implement computationally (simulate then check)
  - Need to always define when to ‘stop’ generating observations

Simulation-based Model Checking



Property

Eg, “Order of peaks is RafP, MEKPP, ERKPP”

Behaviour Checker

Yes/no or probability

Time series data

Model

Lab

PLTL Language

- Behaviours to be checked against a model is expressed in temporal logic

- Probabilistic logic called Probabilistic Linear-time Temporal Logic (PLTL) – MC2 [Donaldson&Gilbert CMSB 2008]

- Main PLTL operators:

  G (P) – P always happens

  F (P) – P happens at some time

  X (P) – P happens in the next time point

  (P1) U (P2) – P1 happens until P2 happens

  P1 { P2 } – P1 happens from the first time P2 happens

  time > ε – After a time point

Qualitative to quantitative descriptions in PLTL

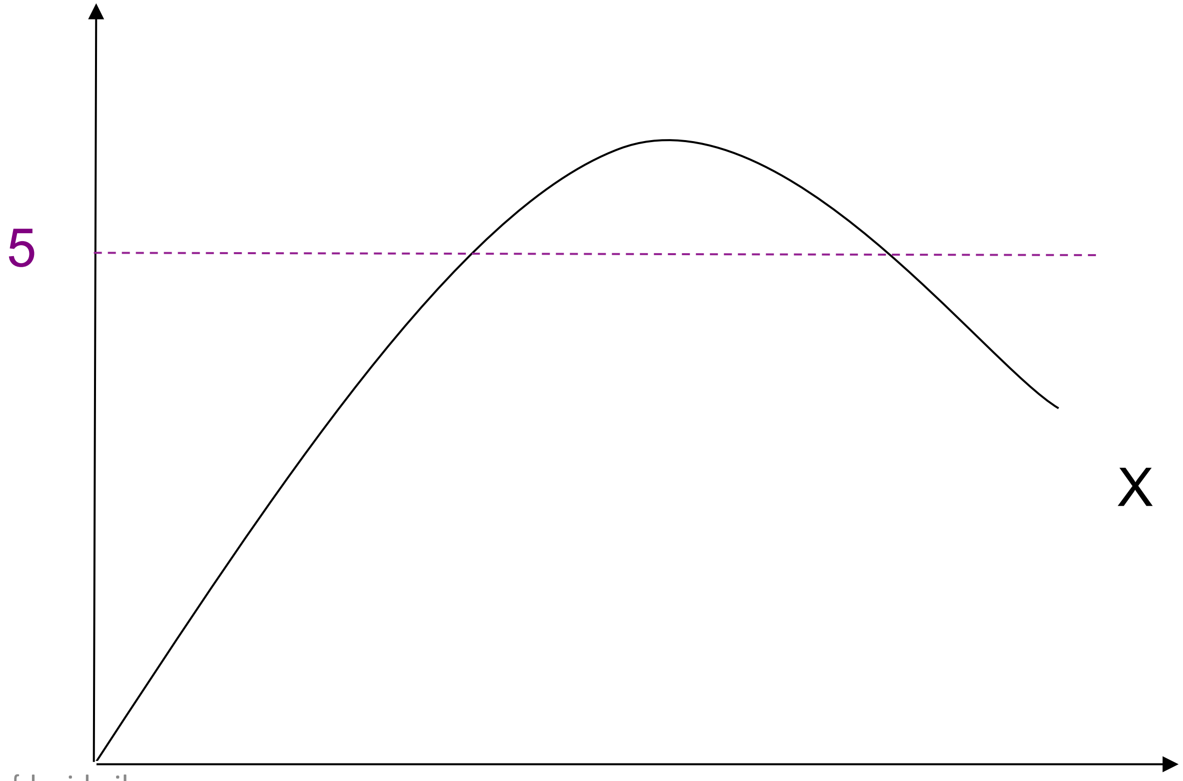
- **Qualitative**:
  Protein rises then falls
  $P=? \, [\, (d(Protein) > 0) \, U \, (G(d(Protein) < 0)) \,]$

- **Semi-qualitative**:
  Protein rises then falls to less than 50% of peak concentration
  $P=? \, [\, (d(Protein) > 0) \, U \, (G(d(Protein) < 0) \wedge F ([Protein] < 0.5 * \max[Protein])) \,]$

- **Semi-quantitative**:
  Protein rises then falls to less than 50% of peak concentration by 60 minutes
  $P=? \, [\, (d(Protein) > 0) \, U \, (G(d(Protein) < 0) \wedge F (time = 60 \wedge Protein < 0.5 * \max(Protein))) \,]$

- **Quantitative**:
  Protein rises then falls to less than 100μMol by 60 minutes
  $P=? \, [\, (d(Protein) > 0) \, U \, (G(d(Protein) < 0) \wedge F (time = 60 \wedge Protein < 100)) \,]$

Continuous output



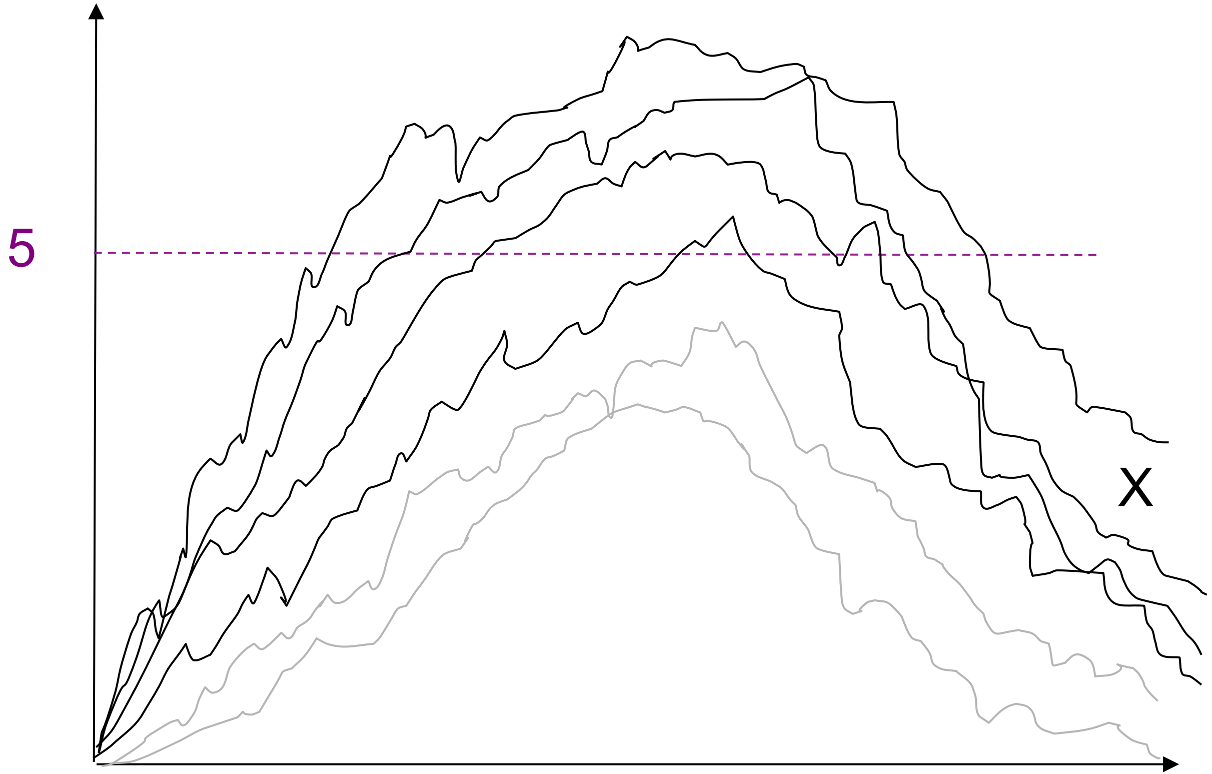
$P_{=?}[\ F(\ X > 5\ )\ ]$

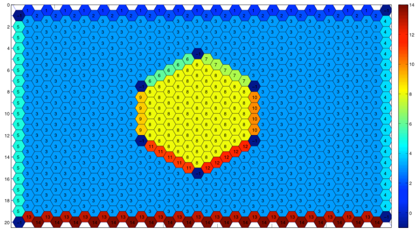
=> P = 1

Stochastic Output

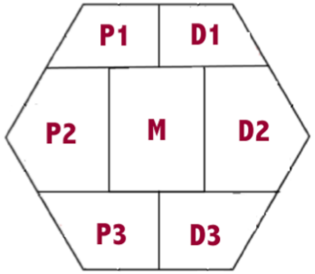
$P_{=?}[\ F(\ X > 5\ )\ ]$

=> P = 4/6





Model Checking
Primary data



Fz- mutant clone model

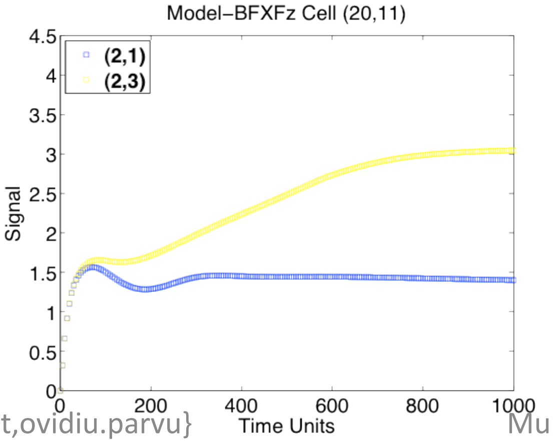
Unlike in the wild-type cells, for the **cells distally neighbouring to the Fz- clone** the concentration of FFD in the middle distal compartment is always lower than that of the middle proximal compartment:

P=? [time > 0 → G(D2 < P2)]

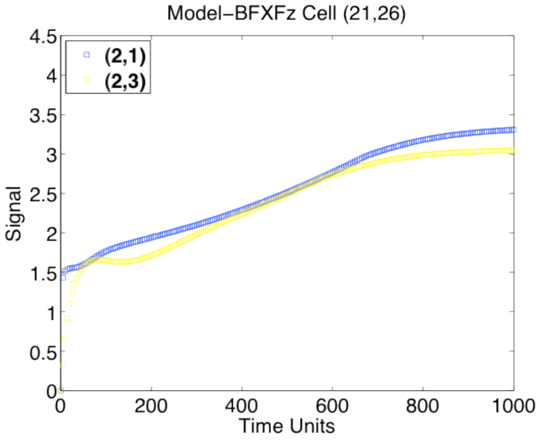
Moreover, the trace of D2 exhibits a peak followed by a trough, which is not true for P2:

P=?[F(d(D2) > 0 ∧ F(d(D2) < 0 ∧ F(d(D2) > 0)))]

Wild-type



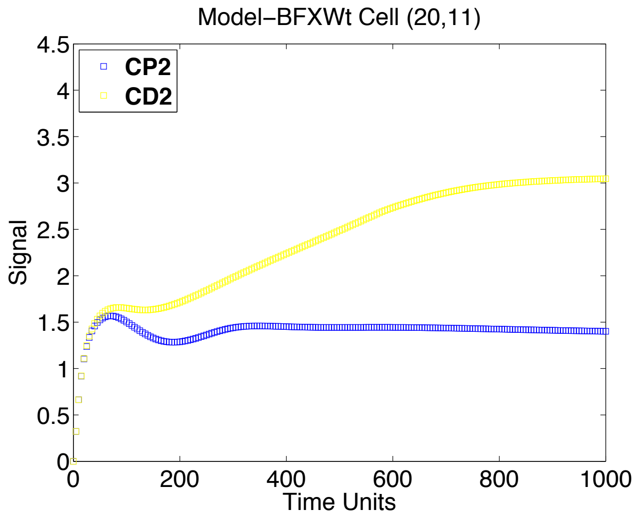
Distally neighbouring to the clone



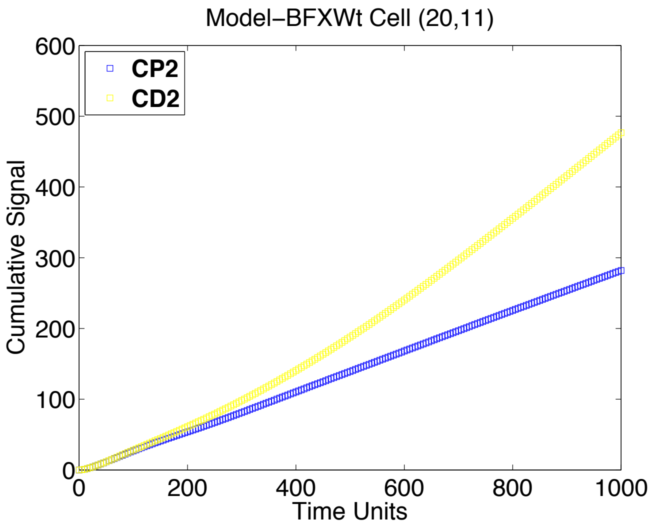
Big idea – check cumulative signal!

- Cumulative signal: time-series of accumulated concentrations of FFD (secondary data)
- Why?

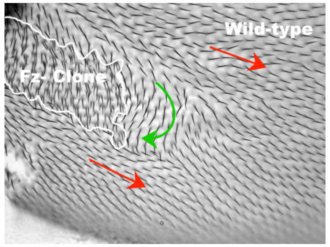
  - The localisation of PCP signalling at any given time point is the result of the cumulative effect of the sum over the signalling events until that point.



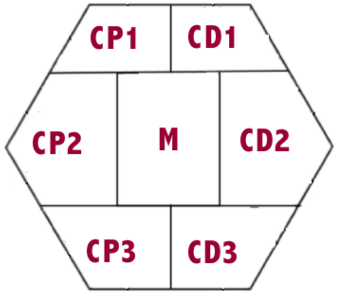
Primary data



Secondary data



Model Checking
Secondary data
Fz- mutant clone model



Wild type cells in the tissue (i.e. away from the clone area).

After short initial period: Always middle distal cumulative[FFD] greater than middle proximal cumulative[FFD]

P=? [time > ε → G(CD2 > CP2)]

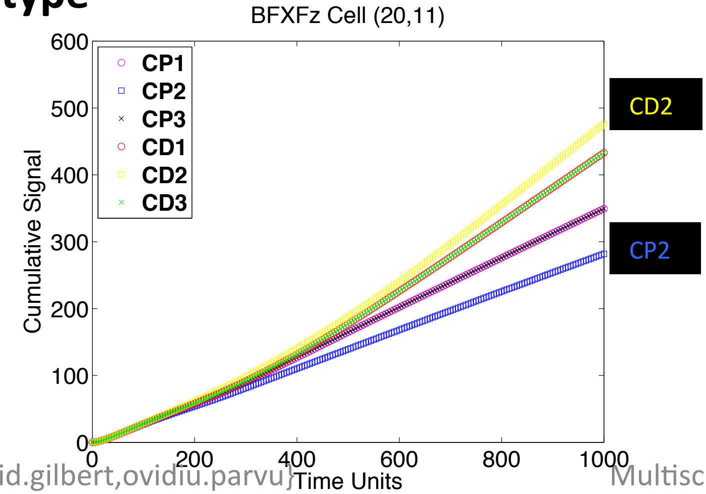
Wild type cells distally neighboring to clone in the tissue

After short initial period: Always middle distal cumulative[FFD] less than middle proximal cumulative[FFD]

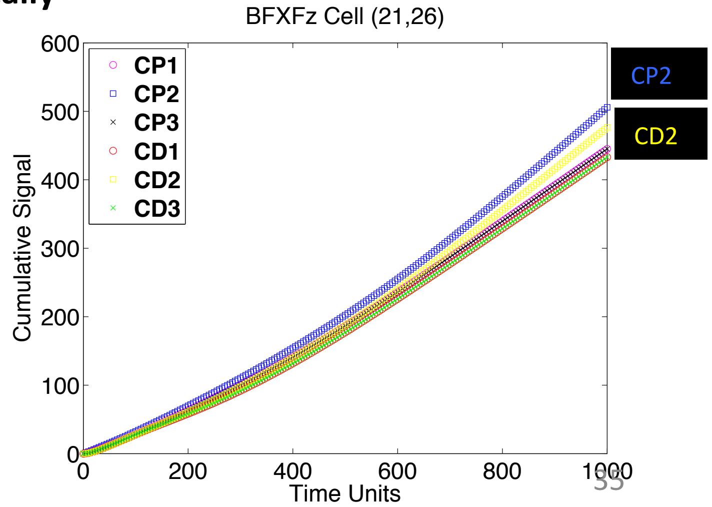
P=? [time > ε → G(CD2 < CP2)]

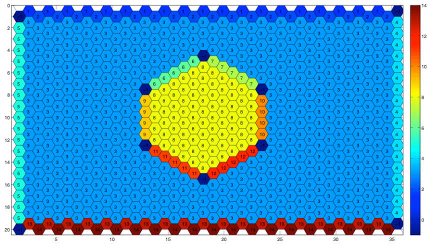
Hairs grow normally in wild-type, but disturbed in WT distally near clone, influence from the clone

Wild-type

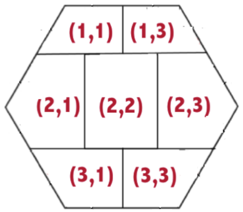


Wild-type distally neighbouring to clone





Model Checking Secondary data



Fz- mutant clone model:

A relatively higher **cumulative signal** in the middle proximal compartment (CP2) compared to the middle distal compartment (CD2) in those **cells distally directly next to the Fz-clone**:

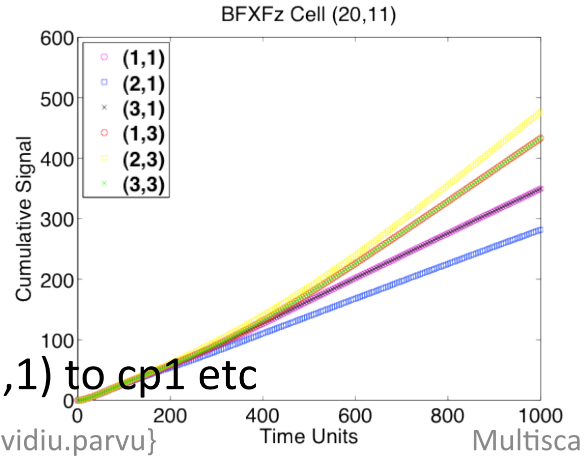
P=? [time > 0 → G(CD2 > CP2)]

Wild type cells in the tissue (i.e. away from the clone area).

P=?[time > ε → G(CD2 > CD1 ∧CD1 = CD3 ±δ∧
CD1 >CP1 ∧CP1 =CP3 ∧CP1 >CP2)]

Where ε = 50 and δ = 0.2

Wild-type

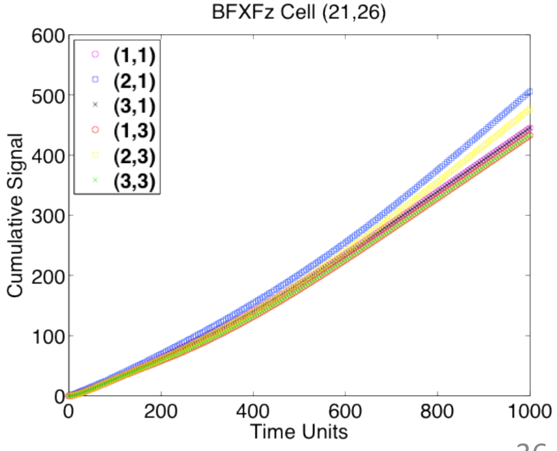


CD2

CP2

Change (1,1) to cp1 etc

Distally neighbouring to the clone



CP2

CD2

Automatic Generation of TL Descriptions

We can use PLTLc to characterize the clusters of time series

PLTLc statements should be

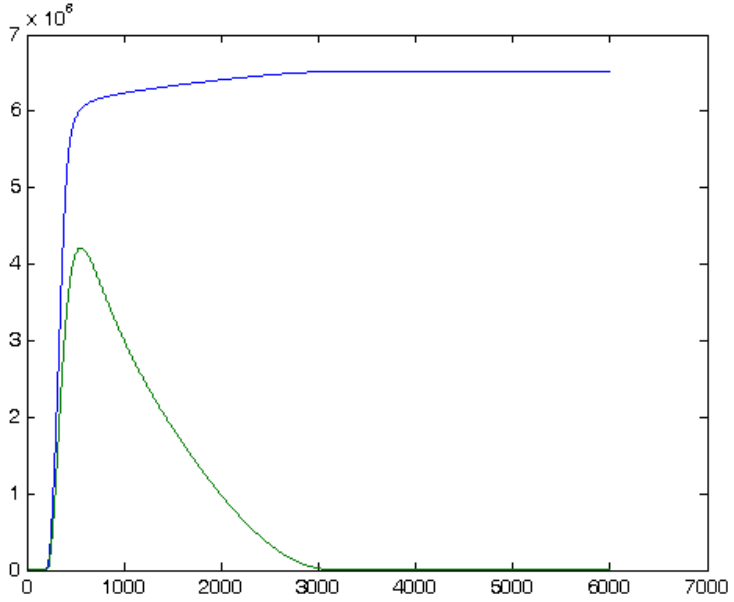
- *general* enough to describe all the time series in a given cluster
- *discriminative* enough to distinguish between time series of different clusters

The generation algorithm is based on property patterns (templates)

Automatic Generation of TL Descriptions

- ***Trend***: describes the trend of a time series as a sequence of direction ("increase", "steady", "decrease")

$$\phi_1 U(\phi_2 U(\ldots U(\phi_{m-1} U(G(\phi_m)))\ldots))$$



If a cluster shows different trends, they are ordered by frequency (F_0 is the most frequent, then F_1 and so on) and the cluster trend is defined by:

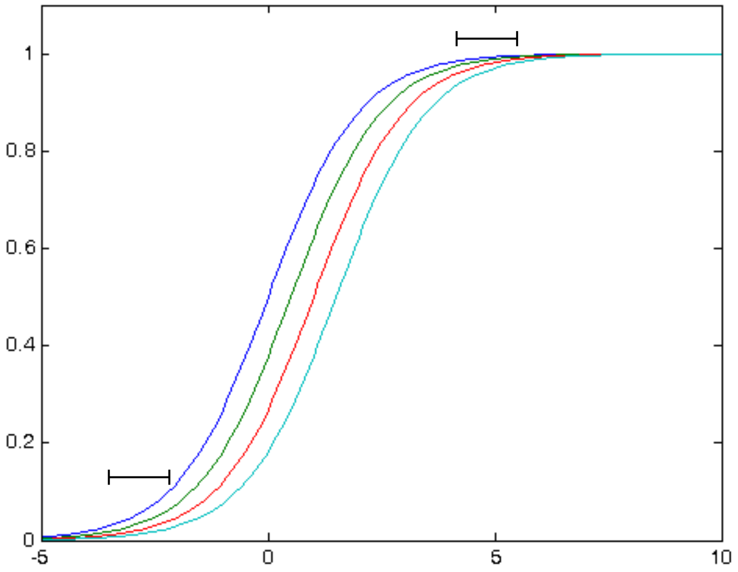
F_0 V F_1 V F_2 V …

Example:
steady-increase-steady OR
steady-increase-decrease-steady

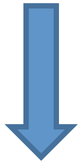
d = 0 U d > 0 U (G(d=0)) V
d = 0 U d > 0 U d<0 U (G(d=0))

Automatic Generation of TL Descriptions

- ***Time***: identifies the time points when the time series changes its direction, i.e. a set of "direction changes"



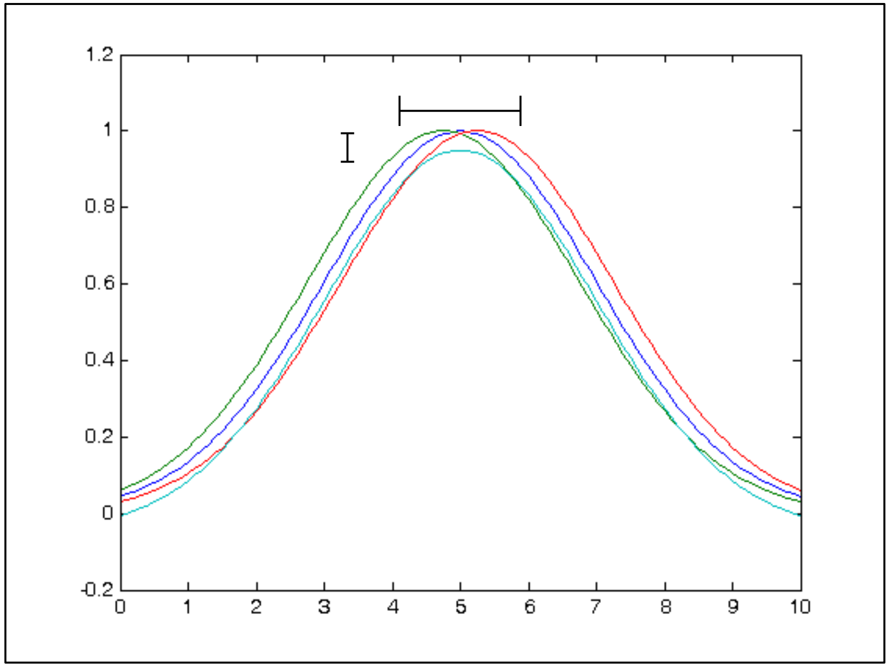
Time series with the same trend may have slightly different time patterns



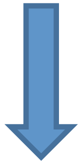
We compute a set of *time intervals*

Automatic Generation of TL Descriptions

- ***Extrema***: represents the occurrence of all the local minima and maxima of a time series



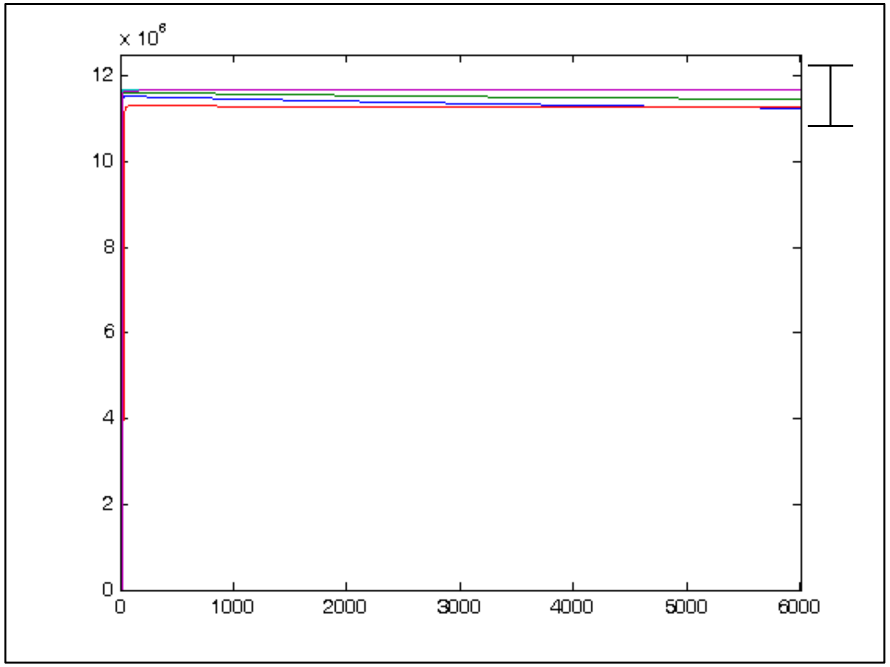
The time and value of each extrema can slightly change among the time series in a cluster



The extrema of a cluster are defined by a sequence of *time* and *value* intervals

Automatic Generation of TL Descriptions

- ***Steady state***: represents the value of the time series steady state (if exists)



The value of each steady state can slightly change among the time series in a cluster

The steady state of a cluste, if exists, is defined by a value interval

Automatic Generation of TL Descriptions

PLTLc GENERATION PROCEDURE:

1. Consider cluster C_i and the set of remaining clusters $\neg C_i$;
2. If C_i and $\neg C_i$ have different trends, stop; otherwise, continue;
3. If C_i and $\neg C_i$ have the same trend with different times, stop; otherwise, continue;
4. If C_i and $\neg C_i$ have at least one different extrema, stop; otherwise, continue;
5. If C_i and $\neg C_i$ have different steady states, stop; otherwise, the clusters are identical and the algorithm cannot return a valid description.

Automatic Generation of TL Descriptions

- The effectiveness of this algorithms is affected by:
  - The cluster's quality
  - The number of "direction changes" of the time series

- The effectiveness of this algorithm is NOT affected by the *number of time series* per cluster

Automatic Generation of TL Descriptions

Evaluation

- To evaluate the PLTLc statement, we test it as a temporal logic query over the clusters

- We use the probability $P_{=?}[\phi_{opt}(C_i)]$ hat the statement correctly classifies the time series belonging to cluster *i*

•We associate to each statement a "confidence level" *Conf :*

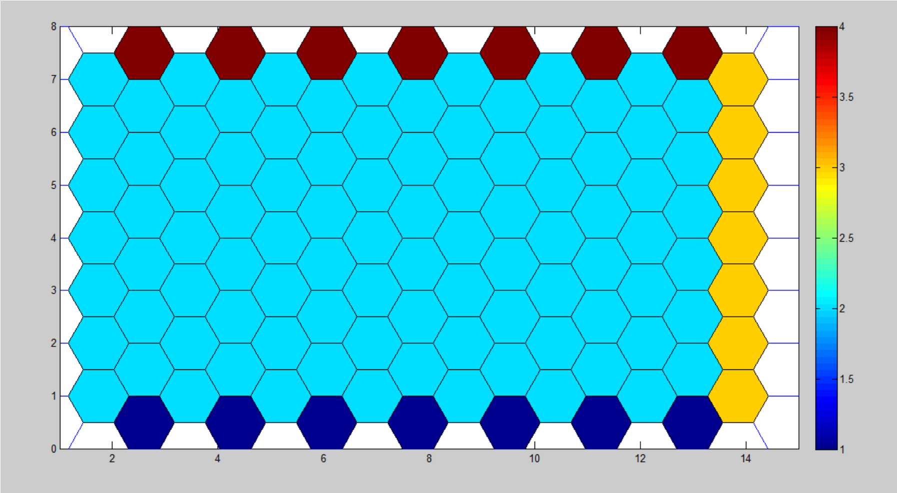
$$Conf(\phi_{opt}(C_i)) = \frac{P_{=?}[\phi_{opt}(C_i)]}{1 + max_{j \neq i} P_{=?}[\phi_{opt}(C_j)]}$$

which indicates its capability to discriminate between time series of cluster i from time series of the most similar cluster j ≠ i.

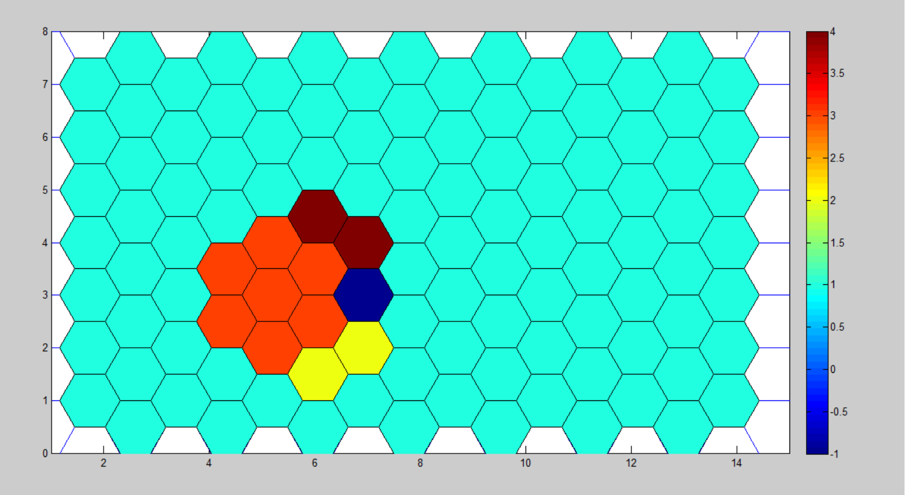
Results

Best clustering result (using DBScan)

Wild Type Tissue



Mutated Tissue



- All the cells have the same behaviour
- The borders are effect of a biased model

- The mutated clone is clearly visible
- Nearby "wild type" cells are INFLUENCED by the mutated clone

DISCOVERED PROPERTIES

PLTLc EXAMPLE:

Behaviour in the INFLUENCED CELLS

$$P_{=?}[d[FFD] > 0\ U(Time \geq 30 \wedge Time \leq 31 \wedge d[FFD] = 0\ \wedge G(d[FFD] = 0)))]$$

“The concentration of FFD increases from time zero, reaches its peak between time 30 and 31, and then becomes steady till the end”.

Publications

- Gao, Q., F. Liu, D. Gilbert, M. Heiner, and D. Tree. 2011, September. “A Multiscale Approach to Modelling Planar Cell Polarity in Drosophila Wing using Hierarchically Coloured Petri nets”. In Proc. 9th International Conference on Computational Methods in Systems Biology (CMSB 2011), 209–218: ACM digital library.

- Gao, Q., Liu, F., Tree, D., & Gilbert, D. (2011). Multi-Cell Modelling Using Coloured Petri Nets Applied to Planar Cell Polarity. In Proceedings of the 2nd International Workshop on Biological Processes & Petri Nets (Vol. 724, pp. 135-150).

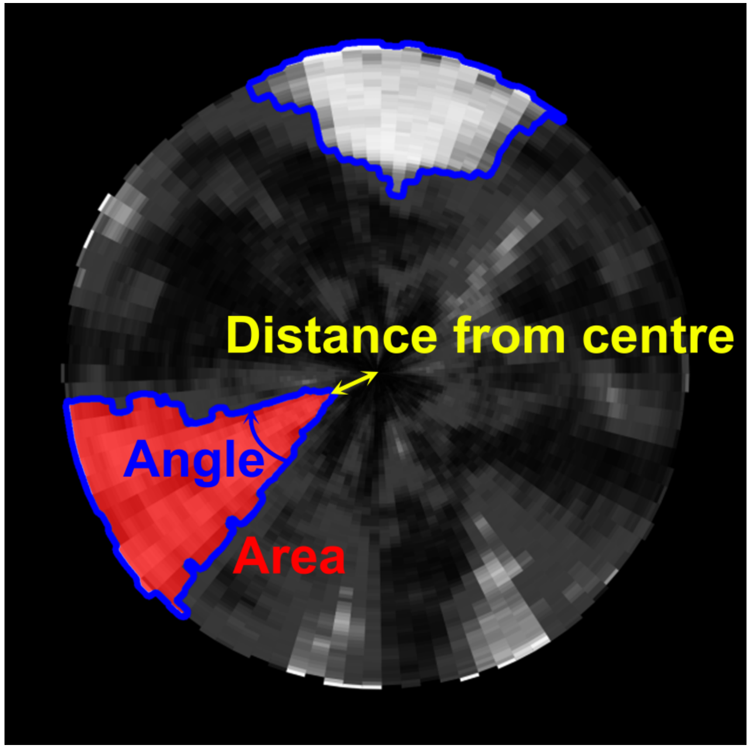
- Gao, Q., Gilbert, D., Heiner, M., Liu, F., Maccagnola, D., & Tree, D. (2012). Multiscale modelling and analysis of planar cell polarity in the drosophila wing.

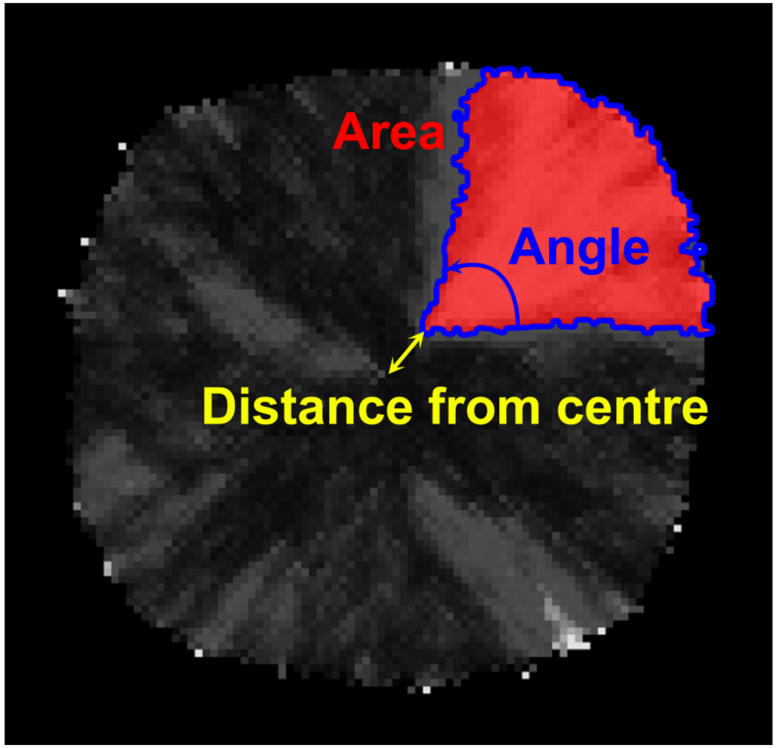
Downloads

- Matlab Codes
- Petri Net Models

→ http://multiscalepn.brunel.ac.uk

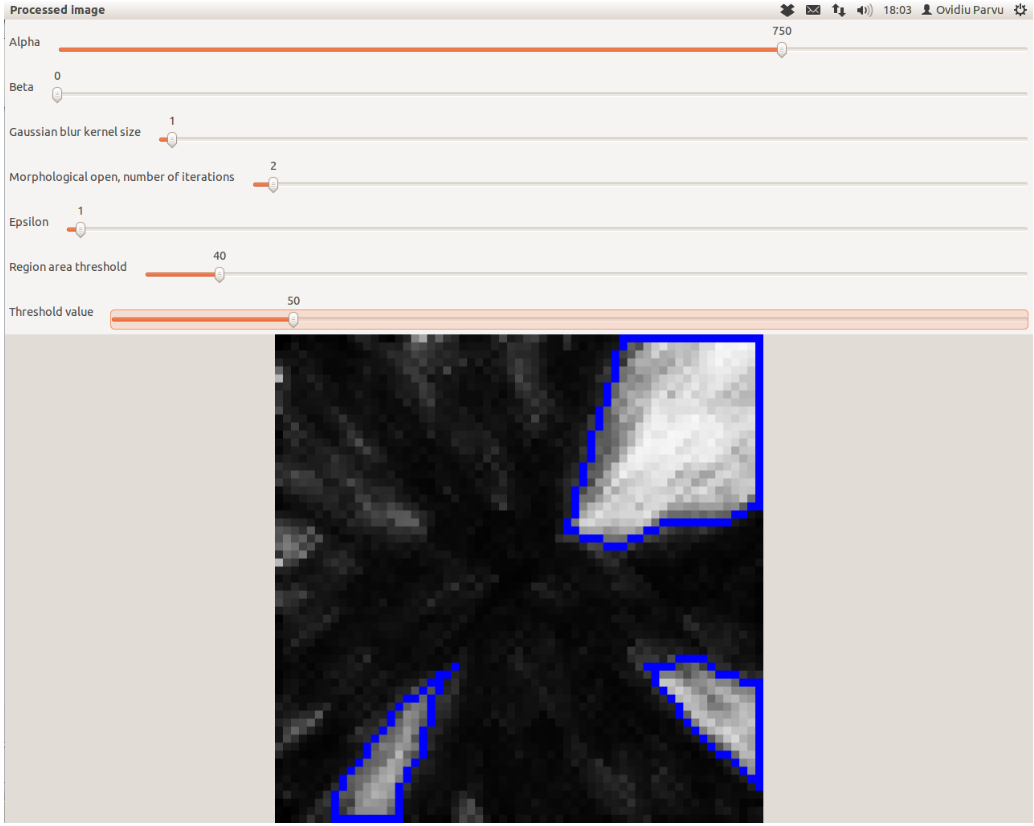
Spatial analysis





Spatial analysis cont’d

Easy to use interface in debug/interaction mode



Alpha 750

Beta 0

Gaussian blur kernel size 1

Morphological open, number of iterations 2

Epsilon 1

Region area threshold 40

Threshold value 50

Spatial analysis cont'd

```
Algorithm SpatialAnalysis is:

1. Load .csv file and convert values into concentrations (from real values to real values
   in the interval [0, 1])
2. Read the file with the concentrations and create an image out of it where each
   concentration corresponds to a pixel in the image
3. Process the image and obtain for each sector its distance from the centre, the angle
   and the total area:
    1. Change the brightness and contrast of the image, such that regions of interest
       are highlighted
    2. Filter out the noise
    3. Threshold the image
    4. Detect contours, approximate polygons, get convex hulls
    5. Get distance from centre, area
    6. Approximate angle as the angle between the closest point to the origin of the
       circle and the middle points of the sides of the sector
4. Print results in a file

endSpatialAnalysis
```

Experiments

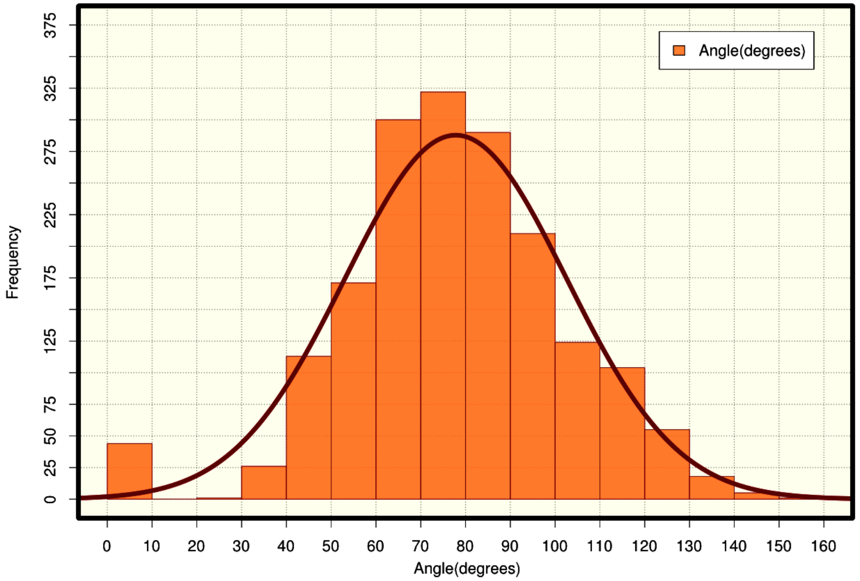
- **1,000 simulations** run for models using both circular and rectangular geometries with an average simulation time of approximately **50 minutes**.
- Each simulation ~= **24 hours** real time growth.
- **Fixed** set of parameters was used for all simulations.
- Output of each simulation analysed using our **sector detection** module.

Results

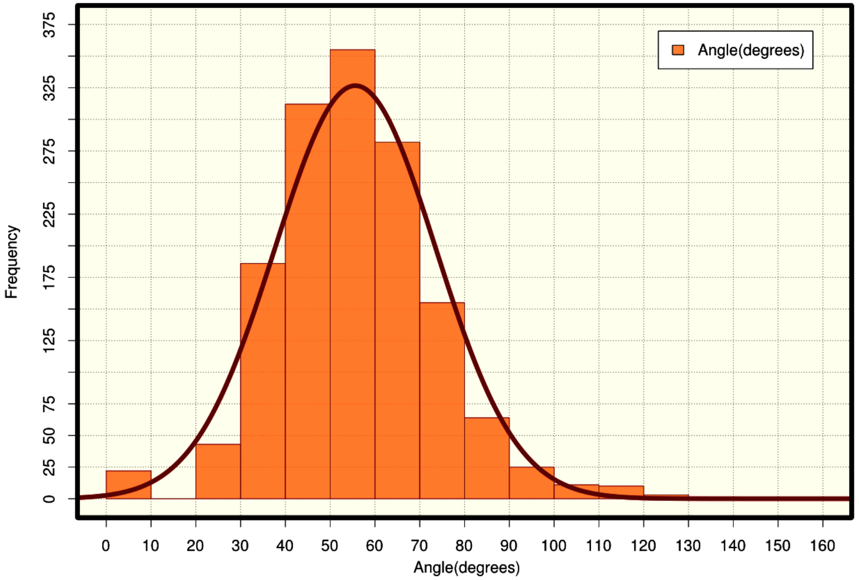
Measures	Area		Distance		Angle		Sectors	
	Rectangular	Circular	Rectangular	Circular	Rectangular	Circular	Rectangular	Circular
Mean	3%	5%	41%	39%	56^o	78^o	1.47	1.78
Std. deviation	2%	2%	17%	16%	18^o	25^o	1.14	1.03
Coeff. of variance	0.93	0.62	0.40	0.41	0.32	0.32	0.77	0.58

Results cont’d

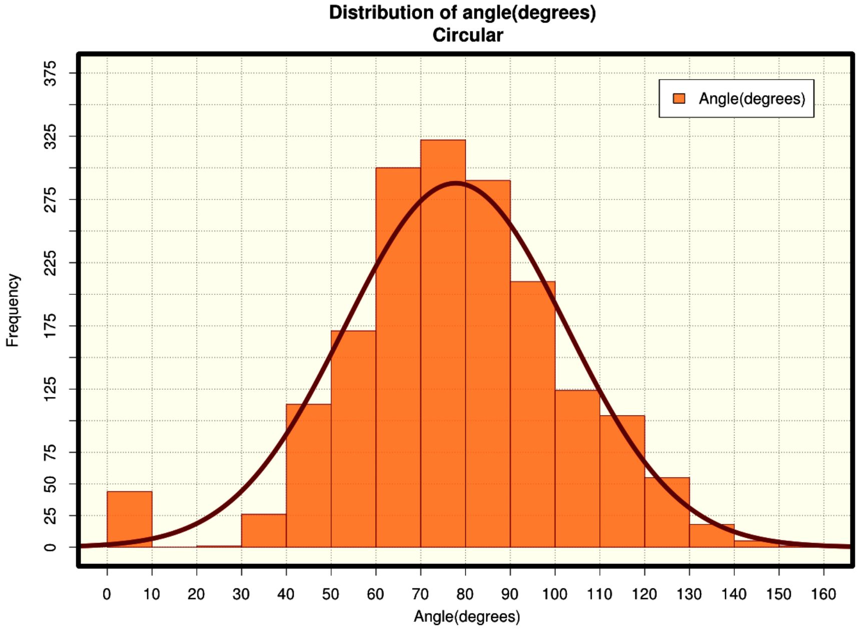
Distribution of angle(degrees)
Circular

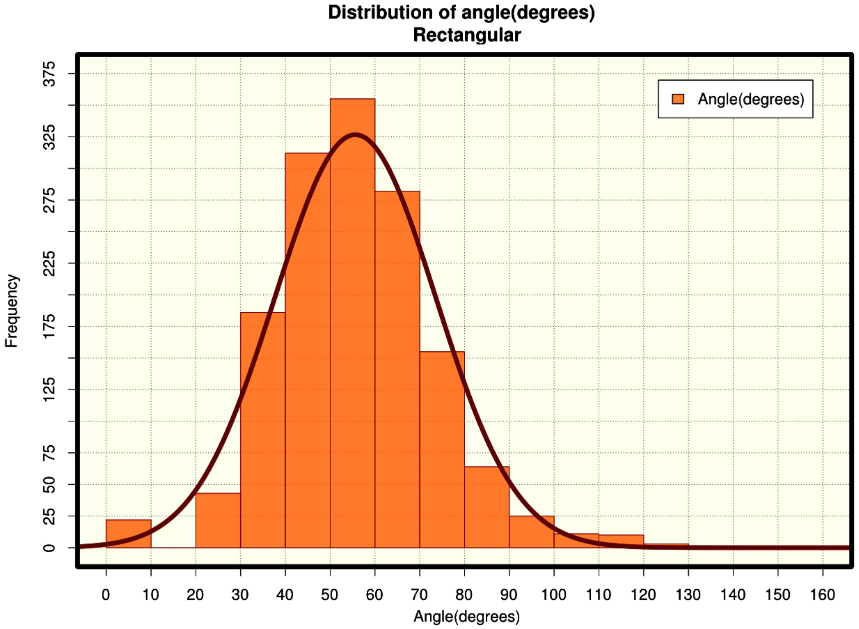


Distribution of angle(degrees)
Rectangular

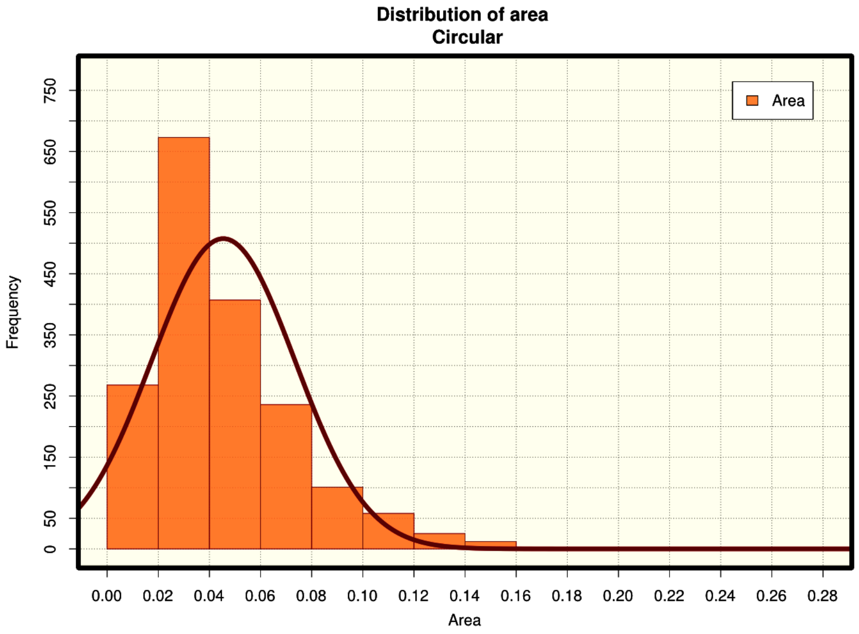


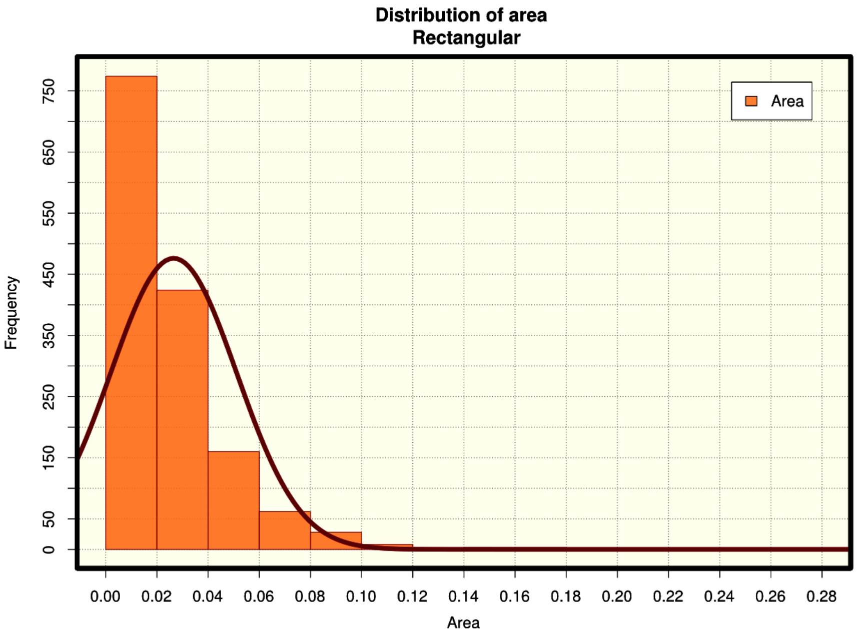
Results cont'd



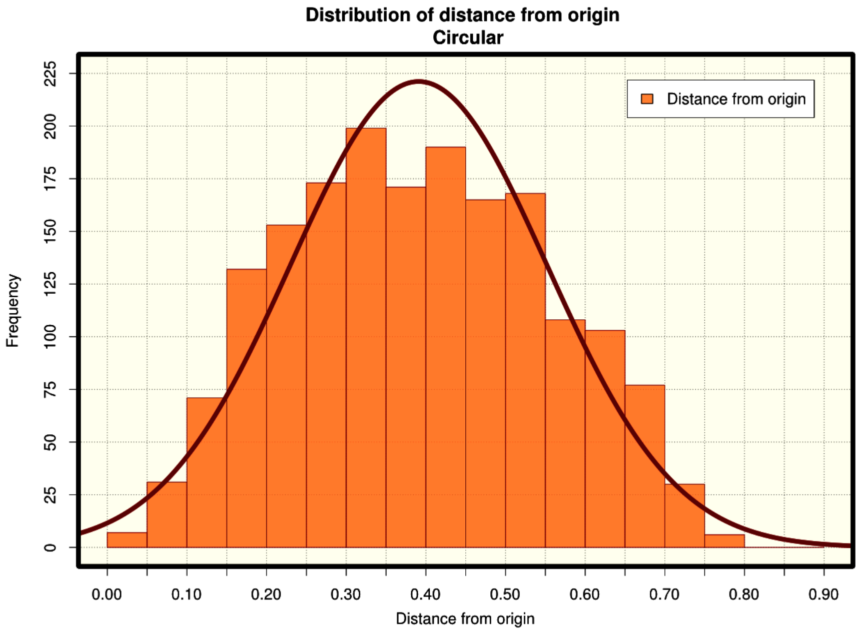


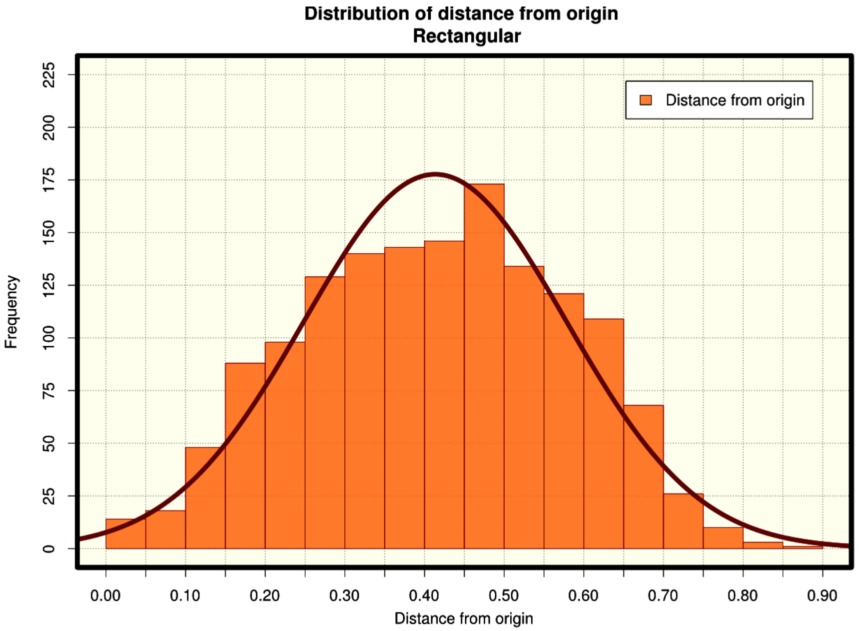
Results cont'd



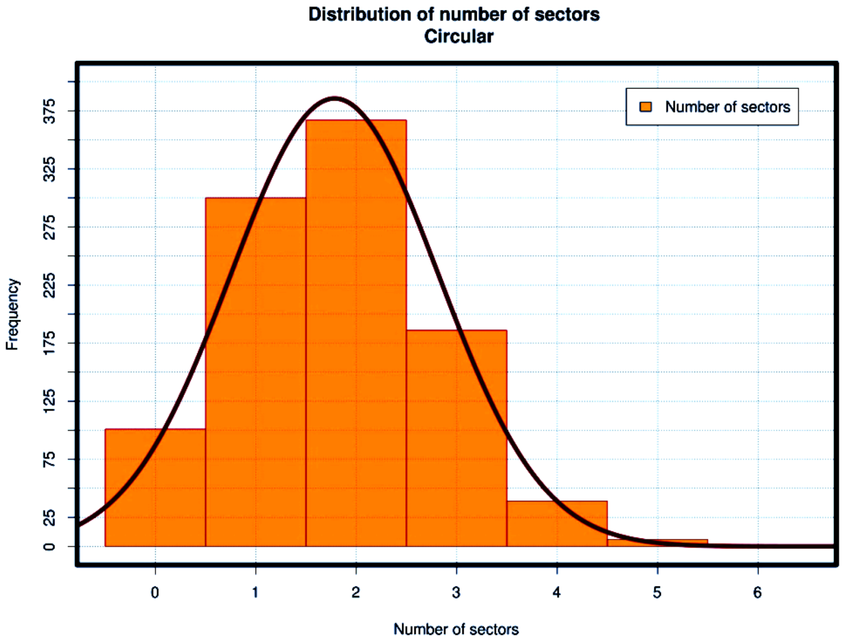


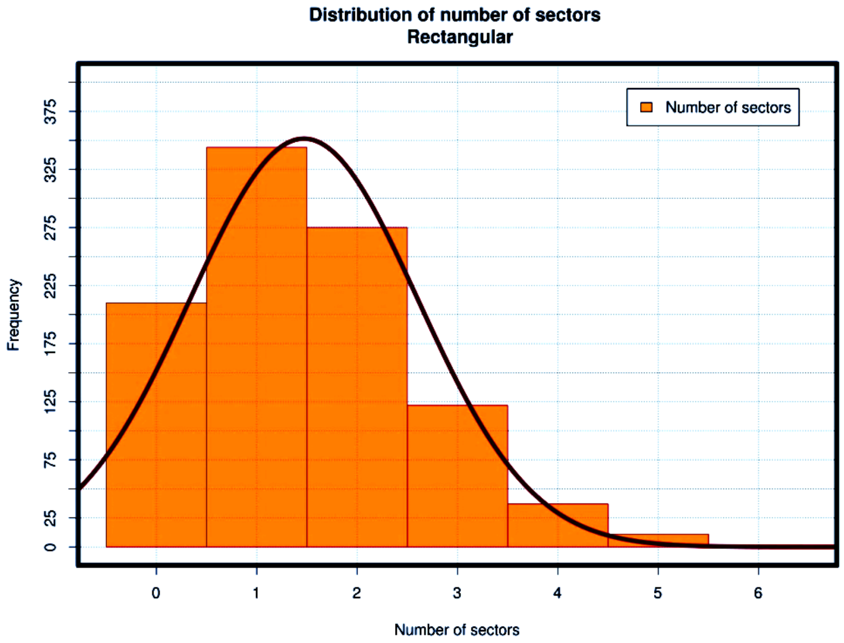
Results cont'd





Results cont'd


Distribution of number of sectors
Circular
Number of sectors
Frequency
Number of sectors


Distribution of number of sectors
Rectangular
Number of sectors
Frequency
Number of sectors