Quantitative Temporal Logics for Systems Biology

Workshop on Identification and Control of Biological Interaction Networks, INRIA Grenoble - Rhône-Alpes

Alexandre Donzé

VERIMAG Laboratory, Grenoble

February 8, 2011

Computer Scientists in the Wild

Formal methods essence is pure computer science: writing programs verifying programs

Yet sometimes messing with control engineering, embedded systems, (*analog*) circuits design

Messing with biologists is more recent and even more scary... How did we end up there ?

Common problematics: complex systems design and analysis from small elements obeying simple, robust and well known rules

Formal Verification

A domain taking its roots in early computer science theory (language and automata theory), discrete mathematics, logics, even philosophy

Its goal: to prove correctness

Growing in applicability steadily since the early 80s and the advent of Model Checking (Turing award of Edmund Clarke and Joseph Sifakis in 2008)

Its popularity "benefited" from spectacular failure of simple testing and bug finding in the 90s (Pentium bug, Ariane 5 self-destruction due to a software bug)

Nowadays an industry standard in digital circuits design

What does it mean *to prove correctness* ?

Correctness is a subjective notion until it is defined *formally*.
Thus we need:

- a description of the systems behaviors
- a specification language to describe *desired* (good) and *unwanted* (bad) properties

What does it mean *to prove correctness* ?

Correctness is a subjective notion until it is defined *formally*.
Thus we need:

- a description of the systems behaviors
- a specification language to describe *desired* (good) and *unwanted* (bad) properties

Coffee machine example

- a good property is: *if I insert a coin and push 'coffee', I get coffee*
- a bad one: *I get a tea (and no change)*

What does it mean *to prove correctness* ?

Correctness is a subjective notion until it is defined *formally*.
Thus we need:

- a description of the systems behaviors
- a specification language to describe *desired* (good) and *unwanted* (bad) properties

Coffee machine example

- a good property is: *if I insert a coin and push 'coffee', I get coffee*
- a bad one: *I get a tea (and no change)*

The system is declared correct iff

all the behaviors of the system satisfies *all* the good properties and *none* of the bad ones

From Verification to Synthesis

Verification of mis-conceived systems can be tedious and frustrating. Rather than chasing bugs, can't we prevent them from happening in the first place ?

Synthesis is the ultimate goal of Formal Verification:

Building correct-by-construction systems directly from specifications

For synthesized systems, verification is unnecessary (or by nature, redundant).

Synthesis in the Wild

Synthesis is a difficult problem: decades of research, actually applied for hardly a couple of years to produce small digital circuits

Attempts to apply synthesis in even more challenging context: software, analog circuits , control engineering, biology, etc

Is this reasonnable/useful ?

Synthesis in the Wild

Synthesis is a difficult problem: decades of research, actually applied for hardly a couple of years to produce small digital circuits

Attempts to apply synthesis in even more challenging context: software, analog circuits , control engineering, biology, etc

Is this reasonnable/useful ? In most cases, no. A common syndrome:

When you have a hammer, everything looks like a nail

Synthesis in the Wild

Synthesis is a difficult problem: decades of research, actually applied for hardly a couple of years to produce small digital circuits

Attempts to apply synthesis in even more challenging context: software, analog circuits , control engineering, biology, etc

Is this reasonnable/useful ? In most cases, no. A common syndrome:

When you have a hammer, everything looks like a nail

Still, genuine belief that diffusing formal methods to other, more barbaric scientific domains, if done in an humble and intelligent way, can still do some good

Reductionism and Systems Biology

As far as my understanding goes, biologists have two alternating paradigms: reductionism and systems biology

Reductionism mood: break every-living-thing apart and list the components

Implicit postulate of reductionism: exhaustive listing will imply perfect understanding of the living thing

Reductionism and Systems Biology

As far as my understanding goes, biologists have two alternating paradigms: reductionism and systems biology

Reductionism mood: break every-living-thing apart and list the components

Implicit postulate of reductionism: exhaustive listing will imply perfect understanding of the living thing

Systems biology is concerned with the tentative realisation of the postulate, i.e., reconstructing a system from its core elements

Illustration: the achievement of DNA sequencing and the challenge of relating genotypes to phenotypes

How Formal Methods can help Systems Biology

Life is robustly correct thanks to the bug chasing virtues of evolution...

To be useful (quantitative, predictive), a reconstructed mechanistic model of a biological system has to robustly satisfy a number of properties

How Formal Methods can help Systems Biology

Life is robustly correct thanks to the bug chasing virtues of evolution...

To be useful (quantitative, predictive), a reconstructed mechanistic model of a biological system has to robustly satisfy a number of properties

Formal methods can help

1. to specify precisely these properties
2. to verify automatically that the model satisfy them
3. to constrain further the model to enforce their robust satisfaction
4. in multiscale modelling, to validate the different layers of abstraction (Example: going from complete models of single cells to models of tissue (ongoing Syne2Arti project))

How Formal Methods can help Systems Biology

Life is robustly correct thanks to the bug chasing virtues of evolution...

To be useful (quantitative, predictive), a reconstructed mechanistic model of a biological system has to robustly satisfy a number of properties

Formal methods can help

1. to specify precisely these properties
2. to verify automatically that the model satisfy them
3. to constrain further the model to enforce their robust satisfaction
4. in multiscale modelling, to validate the different layers of abstraction (Example: going from complete models of single cells to models of tissue (ongoing Syne2Arti project))

Let us be more specific on the models and properties that we consider.

Outline

Outline

Models and First Example

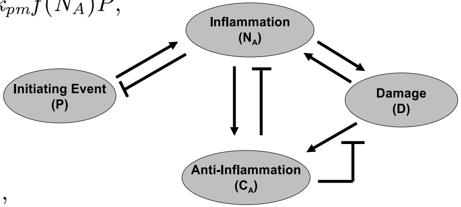
We focus on parametrized ordinary differential equations (ODEs) of the form $\dot{x} = f(x, p)$

Models and First Example

We focus on parametrized ordinary differential equations (ODEs) of the form $\dot{x} = f(x,p)$

Example the acute inflamatory response to a pathogen infection

$$
\begin{aligned}
\frac{dP}{dt} &= k_{pg}P\Big(1-\frac{P}{p_\infty}\Big) - \frac{k_{pm}s_m P}{\mu_m + k_{mp}P} - k_{pm}f(N_A)P,\\
\frac{dN_A}{dt} &= \frac{s_{nr}R}{\mu_{nr}+R} - \mu_n N_A,\\
\frac{dD}{dt} &= k_{dn}f_s(f(N_A)) - \mu_d D,\\
\frac{dC_A}{dt} &= s_c + \frac{k_{cn}f(N_A + k_{cmd}D)}{1+f(N_A+k_{cmd}D)} - \mu_c C_A,
\end{aligned}
$$

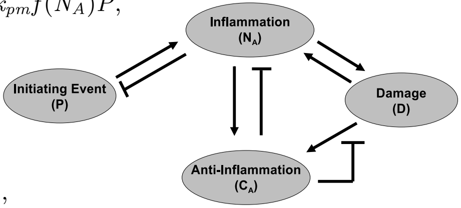


Models and First Example

We focus on parametrized ordinary differential equations (ODEs) of the form $\dot{x} = f(x, p)$

Example the acute inflamatory response to a pathogen infection

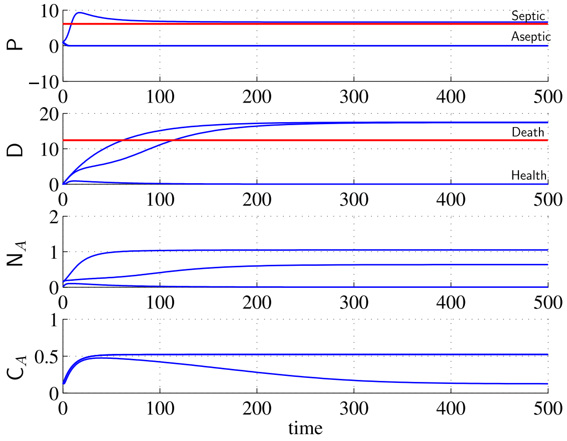
$$\begin{aligned}
\frac{dP}{dt} &= k_{pg}P\Big(1-\frac{P}{p_\infty}\Big) - \frac{k_{pm}s_m P}{\mu_m + k_{mp}P} - k_{pm}f(N_A)P, \\
\frac{dN_A}{dt} &= \frac{s_{nr}R}{\mu_{nr}+R} - \mu_n N_A, \\
\frac{dD}{dt} &= k_{dn}f_s(f(N_A)) - \mu_d D, \\
\frac{dC_A}{dt} &= s_c + \frac{k_{cn}f(N_A + k_{cmd}D)}{1+f(N_A+k_{cmd}D)} - \mu_c C_A,
\end{aligned}$$



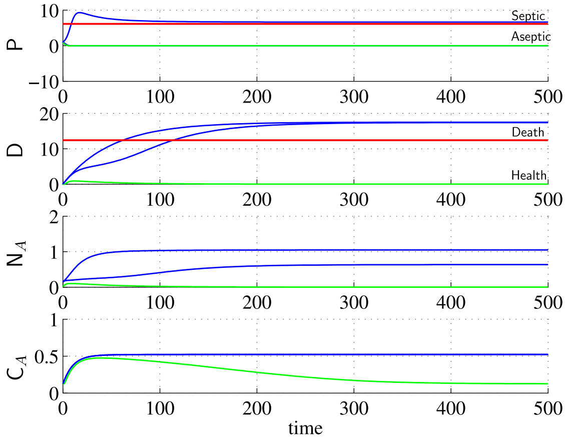
Three possible outcomes :

- Health: pathogen and damage are driven to a low steady state
- Aseptic death: pathogen is eliminated but not tissue damage
- Septic death: tissue damage and pathogen remain high

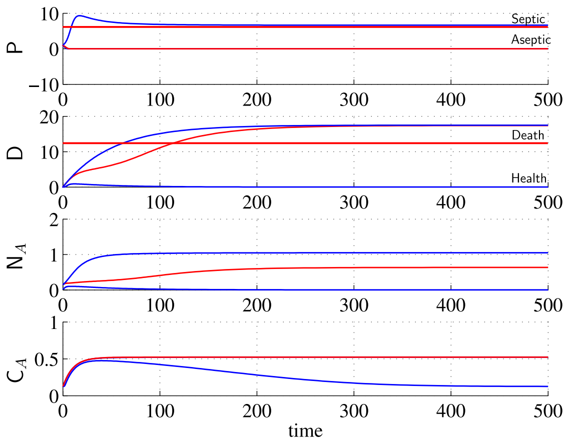
Evolution w.r.t. time



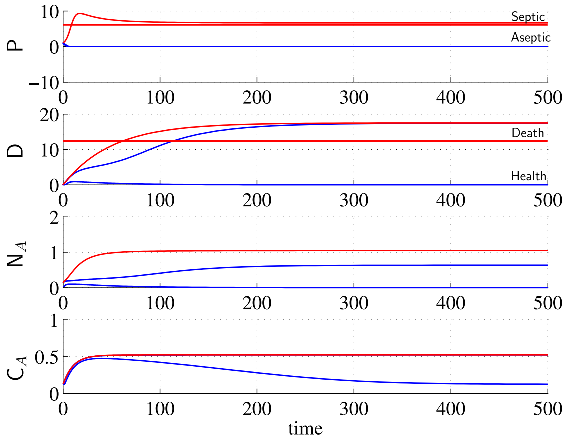
Health outcome



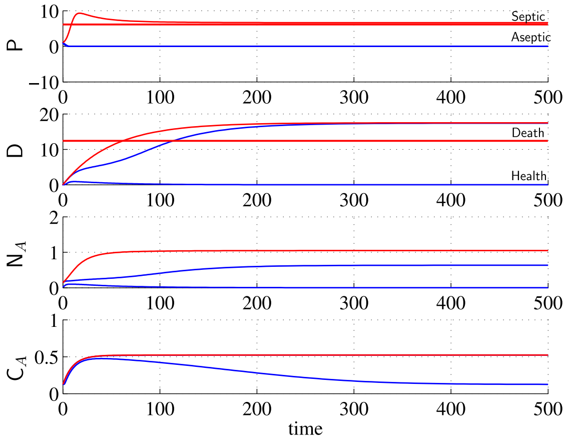
Aseptic death outcome



Septic death outcome



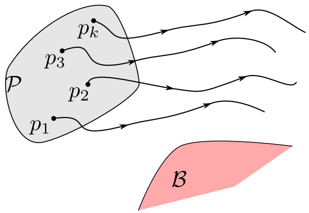
Septic death outcome



Goal identify ranges for initial conditions (e.g. initial concentrations) and parameters (e.g., kinetic constants) in the model that lead to predictable outcomes

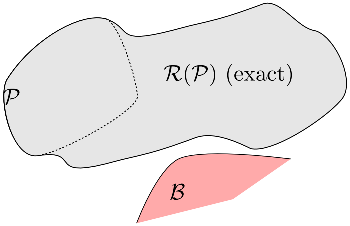
Safety Verification through Reachability Analysis

Define a set $\mathcal{P}$ of parameters p (init. cond. or param), each corresponding to one traj. and some forbidden region $\mathcal{B}$. How to *verify* that all traj. avoid $\mathcal{B}$?



Safety Verification through Reachability Analysis

Define a set $\mathcal{P}$ of parameters p (init. cond. or param), each corresponding to one traj. and some forbidden region $\mathcal{B}$. How to *verify* that all traj. avoid $\mathcal{B}$?

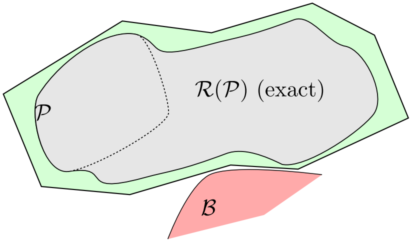


Reachability analysis

- Trying to compute the set containing *all* trajectories

Safety Verification through Reachability Analysis

Define a set $\mathcal{P}$ of parameters p (init. cond. or param), each corresponding to one traj. and some forbidden region $\mathcal{B}$. How to *verify* that all traj. avoid $\mathcal{B}$?

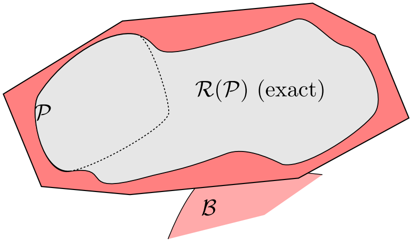


Reachability analysis

- Trying to compute the set containing *all* trajectories
- Using simple set representation
- Empty intersection with $\mathcal{B}$ proves safety

Safety Verification through Reachability Analysis

Define a set $\mathcal{P}$ of parameters p (init. cond. or param), each corresponding to one traj. and some forbidden region $\mathcal{B}$. How to *verify* that all traj. avoid $\mathcal{B}$?



Reachability analysis

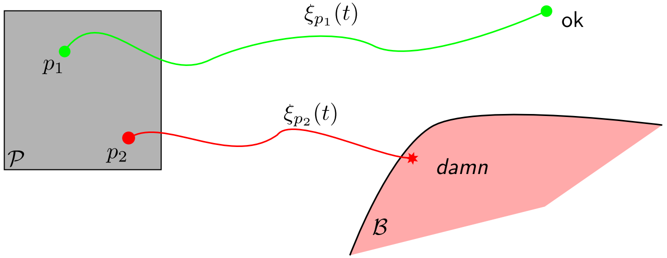
- Trying to compute the set containing *all* trajectories
- Using simple set representation
- Empty intersection with $\mathcal{B}$ proves safety

Difficulties

- Spurious results in case of imprecise over-approximation + difficult for nonlinear system with more than a few continuous variables
- We developped a method based on simulation and sensitivity analysis : good compromise between scalability and precision

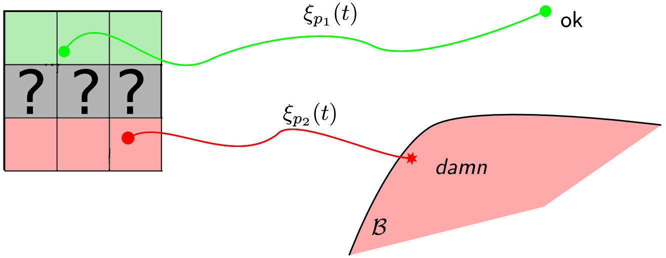
Parameter Synthesis by Refinement

- Trying to find regions for which the system is safe



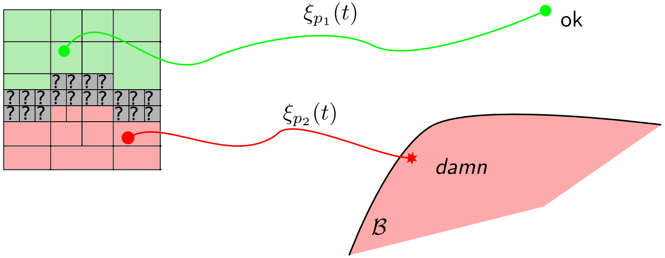
Parameter Synthesis by Refinement

- Trying to find regions for which the system is safe
- Using local reachability analysis, we can certify sub-regions



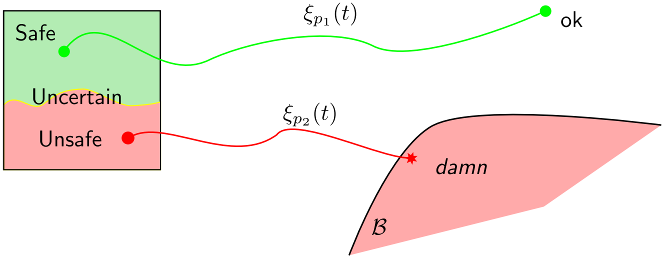
Parameter Synthesis by Refinement

- Trying to find regions for which the system is safe
- Using local reachability analysis, we can certify sub-regions
- Iteratively repeating the process we can find precise boundaries



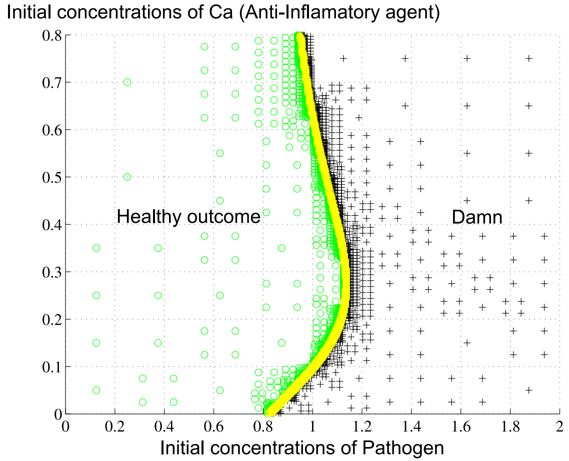
Parameter Synthesis by Refinement

- Trying to find regions for which the system is safe
- Using local reachability analysis, we can certify sub-regions
- Iteratively repeating the process we can find precise boundaries



Application to Acute Inflammation

Each point corresponds to a trajectory representative of its neighborhood.



Outline

Motivations

The technique presented so far deals with *safety* properties

Theory shows that every temporal property on a bounded timed horizon can be expressed as a safety property

Motivations

The technique presented so far deals with *safety* properties

Theory shows that every temporal property on a bounded timed horizon can be expressed as a safety property

Since life has a bounded time horizon, this should be enough...

Motivations

The technique presented so far deals with *safety* properties

Theory shows that every temporal property on a bounded timed horizon can be expressed as a safety property

Since life has a bounded time horizon, this should be enough...

However, translating a property of interest into a safety property is not always trivial nor intuitive, and error prone

In the spirit of synthesis, the specifications of inputs should be as close as possible as plain natural language

Reactive Systems and Temporal Logics

A key issue is the appropriate choice of language to describe properties:

- Enough expressivity
- Ease of writing specification

Reactive Systems and Temporal Logics

A key issue is the appropriate choice of language to describe properties:

- Enough expressivity
- Ease of writing specification

Temporal logics popularized in 1978 by Amir Pnueli when programs shifted from simple input-output relations to reactive programs.

A typical reactive program is an operating system:

- a good property is *always when the mouse is moved, the cursors moves*
- a bad one: *always eventually a blue screen appears and nothing happens*

Reactive Systems and Temporal Logics

A key issue is the appropriate choice of language to describe properties:

- Enough expressivity
- Ease of writing specification

Temporal logics popularized in 1978 by Amir Pnueli when programs shifted from simple input-output relations to reactive programs.

A typical reactive program is an operating system:

- a good property is *always when the mouse is moved, the cursors moves*
- a bad one: *always eventually a blue screen appears and nothing happens*

In our jargon, a good property such as the one above is a *liveness property*. Also, a metabolism *is* a reactive program...

Temporal Logics in a Nutshell

Temporal logics allow to specify patterns that timed behaviors of systems may or may not satisfy. They come in many flavors

The most intuitive is the Linear Temporal Logic (LTL), defined over discrete sequences of states

It is based on logic operators ($\neg$, $\wedge$, $\vee$) and temporal operators : "next", "always" (alw), "eventually" (ev) and "until" ($\mathcal{U}$)
Examples:

- $\varphi\ \varphi\ \varphi\ \varphi\ \cdots$ satisfies alw φ
- $\psi\ \psi\ \psi\ \varphi\ \psi\ \cdots$ satisfies ev φ
- $\varphi\ \varphi\ \varphi\ \varphi\ \psi\ \cdots$ satisfies $\varphi\ \mathcal{U}\ \psi$

From discrete to continuous

Temporal logics mostly developped for discrete systems, a natural way to go is to discretize time and space

However this means that formulas applies to an abstraction of the system, thus introducing a distance between specification and the "real" system

We prefer to adapt temporal logics to continuous time and space :

- Spatial constraints are specified on the real-valued quantities
- Temporal constraints involve dense-time intervals rather than e.g. fixed time steps

Temporal logic formulas: atomic predicates

A predicate is a general inequality constraints on the variables (say A, B, C etc) and parameters at time τ

```
% distance to (A0,B0) is more than 1.
 sqrt((A[tau]-A0)^2 + (B[tau]-B0)^2) > 1.

% the system reached quasi stationnary steady state
abs(ddt{A}[tau])+abs(ddt{A}[tau])) < 1e-10

% A is sensitive to parameter p
abs(d{A}{p}[tau]) > 10*A[tau]/p
```

The canonical form of a predicate μ is:

$$\mu \equiv \mu(\xi_{\mathbf{p}}, \tau) \geq 0$$

Temporal logic operators

Metric Interval Temporal Logic (MITL) syntax:

$$\varphi := \mu \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi\, \mathcal{U}_{[a,b)}\, \varphi | \mathsf{ev}_{[a,b)}\, \varphi | \mathsf{alw}_{[a,b)}\, \varphi$$

```
% The concentration of A becomes more than 1e-6 within 2 s
ev_[0,2] (A[tau]> 1e-6)

% A remains low until B is quasi stationary before 10 seconds
(A[tau] < 1e-8) until_[0, 10] always ((abs(ddt{B}[tau]) < 1e-9))
```

The result is a query language which is close enough to English formulation.
These examples are actually syntaxically valid in our implementation tool Breach

Quantitative Semantics: the Satisfaction Function

A formula φ is evaluated against a single trajectory $\xi_{\mathbf{p}}$ at (the future of) a given time τ

The satisfaction function is a *real-valued* function

$$\tau \rightarrow \rho(\varphi, \xi_{\mathbf{p}}, \tau) \in \mathbb{R}$$

Its semantics is as follows:

- $\rho(\varphi, \xi_{\mathbf{p}}, \tau) \geq 0$ iff $\xi_{\mathbf{p}}$ satisfies φ
- $\|\rho(\varphi, \xi_{\mathbf{p}}, \tau)\|$ represents the *robustness* of the satisfaction or non satisfaction

Intuitively ρ is a measure of the signed distance to the set where φ is false.

Computing the Satisfaction Function

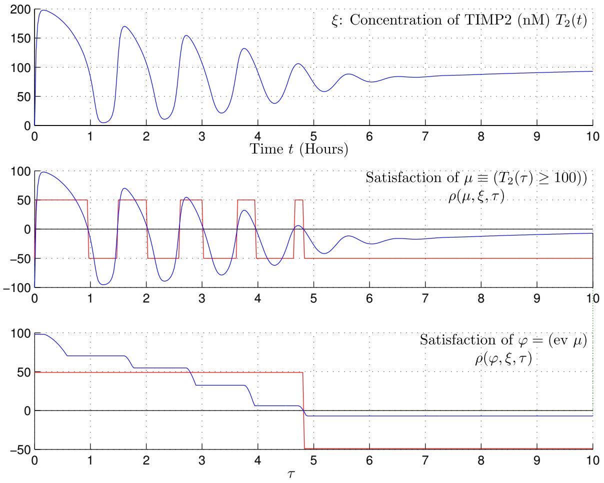
The cost of computing ρ is linear in the size of the formula and the length of the simulation (small computional overhead)

For a predicate $\mu(\xi_p, \tau) \geq 0$, we have simply $\rho(\mu, \xi_p, \tau) = \mu(\xi_p, \tau)$

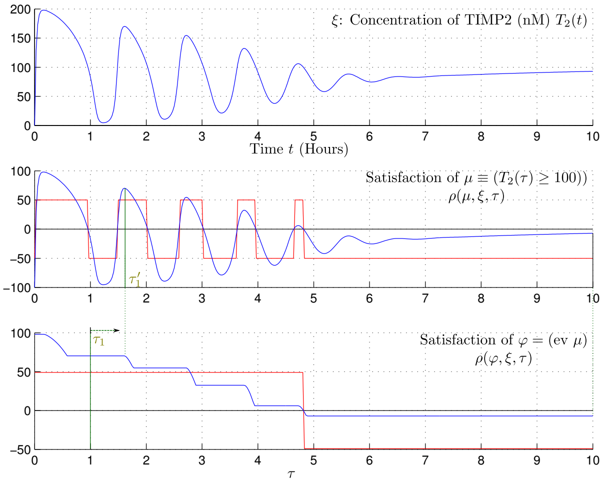
For operators: extension of the known correspondance between $\min - \max$ operators and boolean operators:

$$\begin{array}{lcl}
\rho(\neg\varphi, \xi_p, \tau) & = & -\rho(\varphi, \xi_p, \tau) \\
\rho(\varphi_1 \wedge \varphi_2, \xi_p, \tau) & = & \min(\rho(\varphi_1, \xi_p, \tau), \rho(\varphi_2, \xi_p, \tau)) \\
\rho(\mathsf{ev}_{[a,b]}\ \varphi) & = & \max\limits_{\tau' \in [\tau+a,\ \tau+b]} \rho(\varphi, \xi_p, \tau') \\
\rho(\varphi_1 \mathcal{U}_{[a,b]}\ \varphi_2, \xi_p, t) & = & \max\limits_{r \in \tau+[a,b]} (\min(\rho(\varphi_2, \xi_p, r),\ \min\limits_{s \in [\tau, r]} \rho(\varphi_1, \xi_p, s))
\end{array}$$

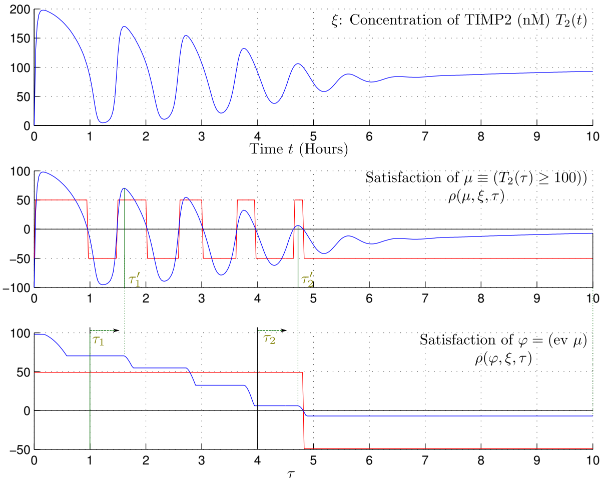
A Simple Formula



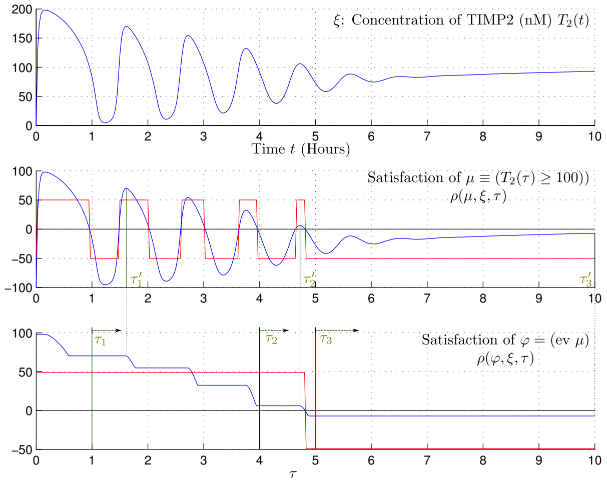
A Simple Formula



A Simple Formula



A Simple Formula



Interpreting the Robust Satisfaction Function

ρ relates a trajectory (thus a parameter value) and a formula φ to a number. This number measures the local robustness of φ.

To some extent, we can also compute the gradient of ρ, thus the sensitivity of φ to parameters.

Interpreting the Robust Satisfaction Function

ρ relates a trajectory (thus a parameter value) and a formula φ to a number. This number measures the local robustness of φ.

To some extent, we can also compute the gradient of ρ, thus the sensitivity of φ to parameters.

If φ defines a forbidden region, ρ measures the distance to this region. The zero level set of ρ defines boundaries between safe and unsafe parameter regions.
These can be computed with the same refinement procedure as before, or using more advanced boundary detection algorithms.

Interpreting the Robust Satisfaction Function

ρ relates a trajectory (thus a parameter value) and a formula φ to a number. This number measures the local robustness of φ.

To some extent, we can also compute the gradient of ρ, thus the sensitivity of φ to parameters.

If φ defines a forbidden region, ρ measures the distance to this region. The zero level set of ρ defines boundaries between safe and unsafe parameter regions.
These can be computed with the same refinement procedure as before, or using more advanced boundary detection algorithms.

For high-dimensional parameter sets, quasi-Monte Carlo sampling can be used for an efficient coverage and a measure of a global robustness

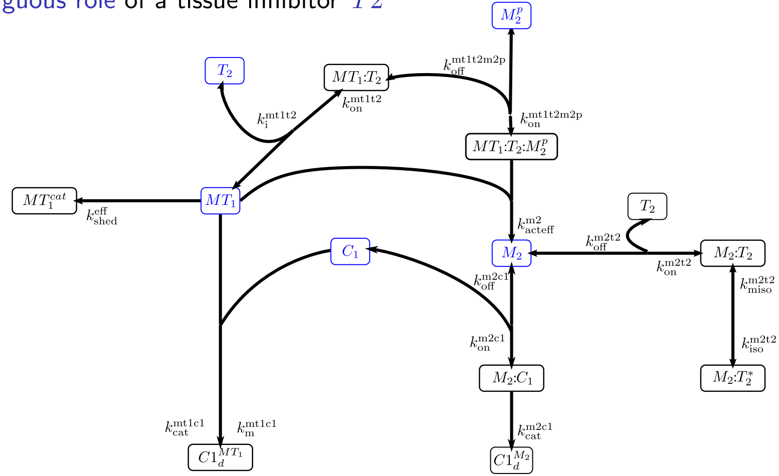
For parameter synthesis, ρ can be used as an objective function in generic global optimization algorithm

Outline

An Enzymatic Network Involved in Angiogenesis

Collagen (C_1) degradation by matrix metalloproteinase (M_2^P) and membrane type 1 metalloproteinase (MT_1) [KP04]

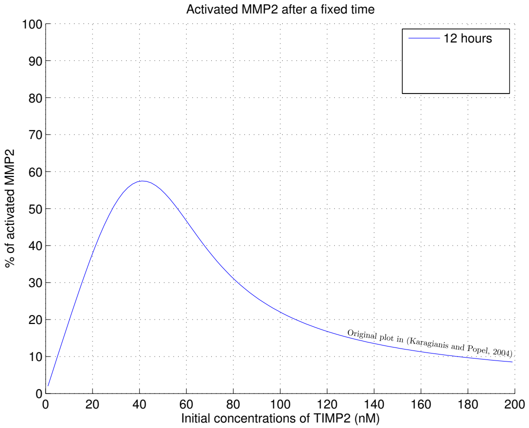
Ambiguous role of a tissue inhibitor $T2$



M_2^p
T_2
$MT_1{:}T_2$
$k_{\text{off}}^{\text{mt1t2m2p}}$
$k_{\text{on}}^{\text{mt1t2}}$
$k_{\text{i}}^{\text{mt1t2}}$
$k_{\text{on}}^{\text{mt1t2m2p}}$
$MT_1{:}T_2{:}M_2^p$
MT_1^{cat}
MT_1
$k_{\text{shed}}^{\text{eff}}$
T_2
$k_{\text{acteff}}^{\text{m2}}$
C_1
M_2
$k_{\text{off}}^{\text{m2t2}}$
$M_2{:}T_2$
$k_{\text{on}}^{\text{m2t2}}$
$k_{\text{off}}^{\text{m2c1}}$
$k_{\text{miso}}^{\text{m2t2}}$
$k_{\text{on}}^{\text{m2c1}}$
$k_{\text{iso}}^{\text{m2t2}}$
$M_2{:}C_1$
$M_2{:}T_2^*$
$k_{\text{cat}}^{\text{mt1c1}}$
$k_{\text{m}}^{\text{mt1c1}}$
$k_{\text{cat}}^{\text{m2c1}}$
$C1_d^{MT_1}$
$C1_d^{M_2}$

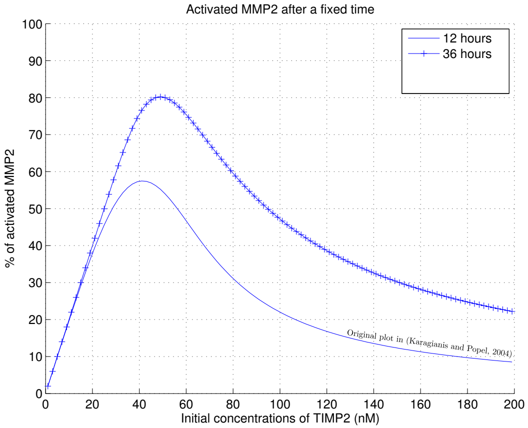
Rigorous Steady State Analysis

In [KP04], activation of M_2^P after 12h "Nearly steady state" for $T_2(0)$ between 0 and 200 nM.



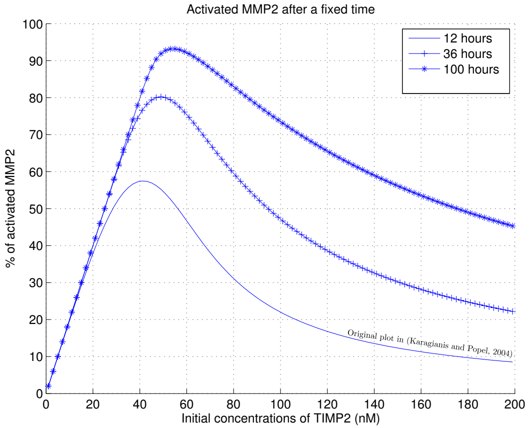
Rigorous Steady State Analysis

In [KP04], activation of M_2^P after 12h "Nearly steady state" for $T_2(0)$ between 0 and 200 nM. It turned out that steady state was not reached for $T_2(0) > 20$ nM !



Rigorous Steady State Analysis

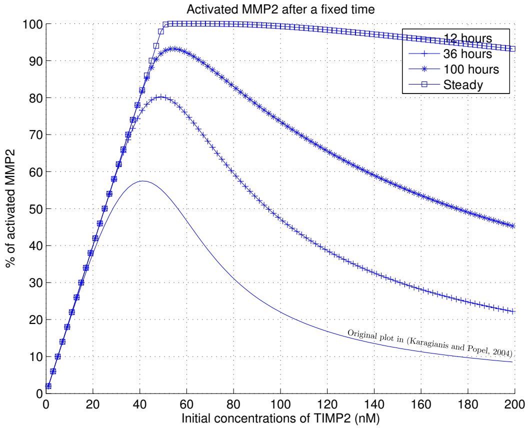
In [KP04], activation of M_2^P after 12h "Nearly steady state" for $T_2(0)$ between 0 and 200 nM. It turned out that steady state was not reached for $T_2(0) > 20$ nM !



Rigorous Steady State Analysis

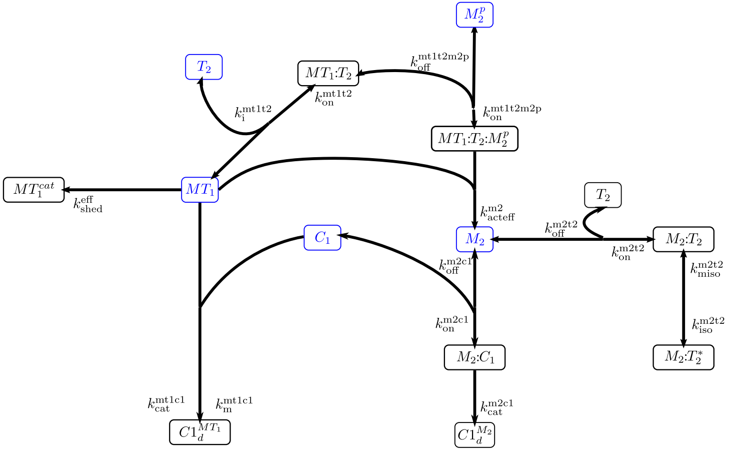
In [KP04], activation of M_2^P after 12h "Nearly steady state" for $T_2(0)$ between 0 and 200 nM. It turned out that steady state was not reached for $T_2(0) > 20$ nM !

Using $\varphi \equiv \text{ev alw } (|\dot{M}_2(t)| < \epsilon \times M_2^P(0))$ we could guarantee the correct plot.



Formalizing Synergism

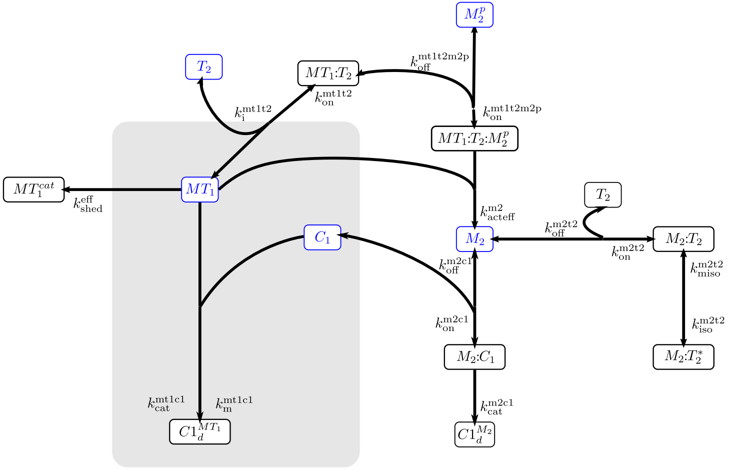
Collagen can be degraded either by MT_1 or by M_2.



M_2^p
T_2
MT_1:T_2
$k_{\text{off}}^{\text{mt1t2m2p}}$
$k_{\text{on}}^{\text{mt1t2}}$
$k_{\text{i}}^{\text{mt1t2}}$
$k_{\text{on}}^{\text{mt1t2m2p}}$
MT_1:T_2:M_2^p
MT_1^{cat}
MT_1
$k_{\text{shed}}^{\text{eff}}$
$k_{\text{acteff}}^{\text{m2}}$
T_2
C_1
M_2
$k_{\text{off}}^{\text{m2t2}}$
M_2:T_2
$k_{\text{on}}^{\text{m2t2}}$
$k_{\text{off}}^{\text{m2c1}}$
$k_{\text{miso}}^{\text{m2t2}}$
$k_{\text{on}}^{\text{m2c1}}$
$k_{\text{iso}}^{\text{m2t2}}$
M_2:C_1
M_2:T_2^*
$k_{\text{cat}}^{\text{mt1c1}}$
$k_{\text{m}}^{\text{mt1c1}}$
$k_{\text{cat}}^{\text{m2c1}}$
$C1_d^{MT_1}$
$C1_d^{M_2}$

Formalizing Synergism

Collagen can be degraded either by MT_1 or by M_2.

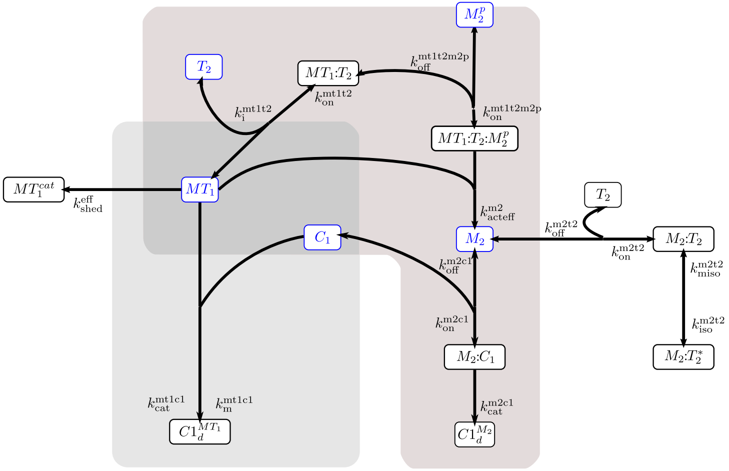


M_2^p
T_2
$MT_1{:}T_2$
$k_{\text{off}}^{\text{mt1t2m2p}}$
$k_{\text{on}}^{\text{mt1t2}}$
$k_{\text{i}}^{\text{mt1t2}}$
$k_{\text{on}}^{\text{mt1t2m2p}}$
$MT_1{:}T_2{:}M_2^p$
MT_1^{cat}
$k_{\text{shed}}^{\text{eff}}$
MT_1
$k_{\text{acteff}}^{\text{m2}}$
T_2
C_1
M_2
$k_{\text{off}}^{\text{m2t2}}$
$M_2{:}T_2$
$k_{\text{on}}^{\text{m2t2}}$
$k_{\text{off}}^{\text{m2c1}}$
$k_{\text{miso}}^{\text{m2t2}}$
$k_{\text{on}}^{\text{m2c1}}$
$k_{\text{iso}}^{\text{m2t2}}$
$M_2{:}C_1$
$M_2{:}T_2^*$
$k_{\text{cat}}^{\text{mt1c1}}$
$k_{\text{m}}^{\text{mt1c1}}$
$k_{\text{cat}}^{\text{m2c1}}$
$C1_d^{MT_1}$
$C1_d^{M_2}$

Formalizing Synergism

Collagen can be degraded either by MT_1 or by M_2. We defined a notion of synergism by :

“Before 12h, 90 % of initial collagen is degraded: $\text{ev}_{[0,12h]}(C_1(\tau)/C_1(0) < 0.1)$ and at least 50 % by $M2$: $\text{ev}_{[0,12h]}(C1_d^{M_2}(\tau) > C1_d^{MT_1}(\tau))$ ”

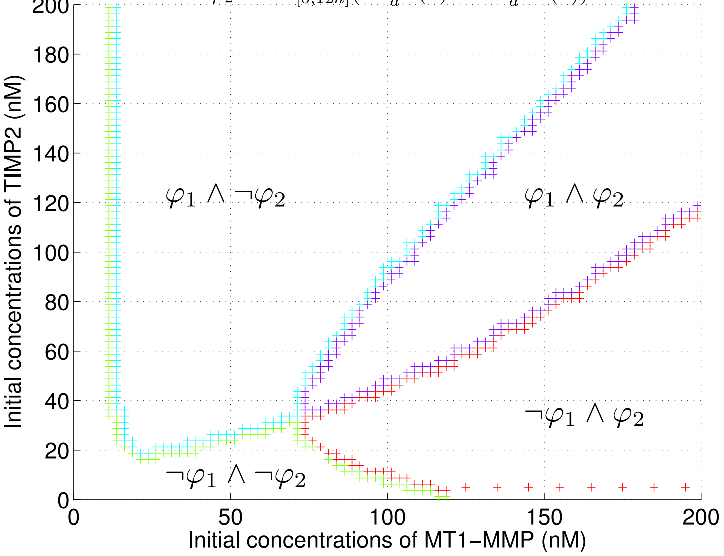


M_2^p
T_2
$MT_1{:}T_2$
$k_{\text{off}}^{\text{mt1t2m2p}}$
$k_{\text{on}}^{\text{mt1t2}}$
$k_{\text{i}}^{\text{mt1t2}}$
$k_{\text{on}}^{\text{mt1t2m2p}}$
$MT_1{:}T_2{:}M_2^p$
MT_1^{cat}
MT_1
$k_{\text{shed}}^{\text{eff}}$
$k_{\text{acteff}}^{\text{m2}}$
T_2
C_1
M_2
$k_{\text{off}}^{\text{m2t2}}$
$M_2{:}T_2$
$k_{\text{on}}^{\text{m2t2}}$
$k_{\text{off}}^{\text{m2c1}}$
$k_{\text{miso}}^{\text{m2t2}}$
$k_{\text{on}}^{\text{m2c1}}$
$k_{\text{iso}}^{\text{m2t2}}$
$M_2{:}C_1$
$M_2{:}T_2^*$
$k_{\text{cat}}^{\text{mt1c1}}$
$k_{\text{m}}^{\text{mt1c1}}$
$k_{\text{cat}}^{\text{m2c1}}$
$C1_d^{MT_1}$
$C1_d^{M_2}$

Synergism, Result

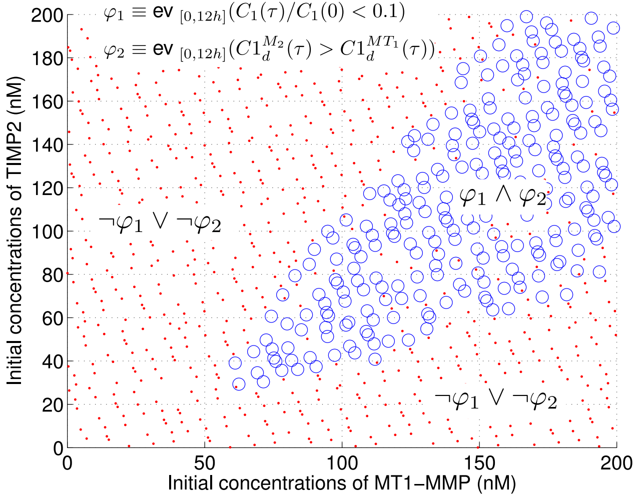
$\varphi_1 \equiv \text{ev}_{[0,12h]}(C_1(\tau)/C_1(0) < 0.1)$

$\varphi_2 \equiv \text{ev}_{[0,12h]}(C1_d^{M_2}(\tau) > C1_d^{MT_1}(\tau))$



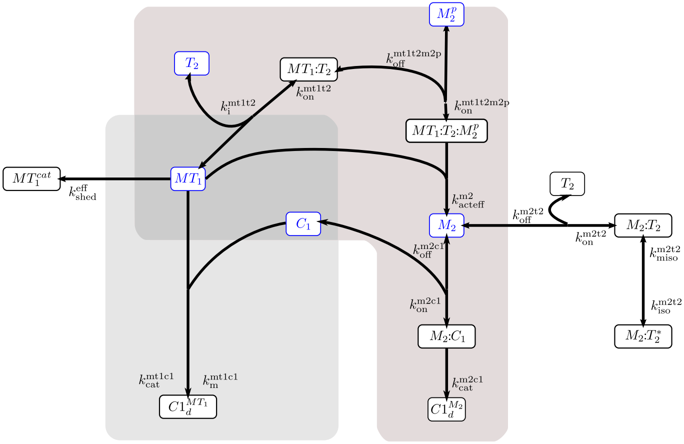
Synergism, Global Analysis

Varying all other parameters around 10% of nominal value, and using quasi-Monte-Carlo sampling, we measure the robustness of the regions found



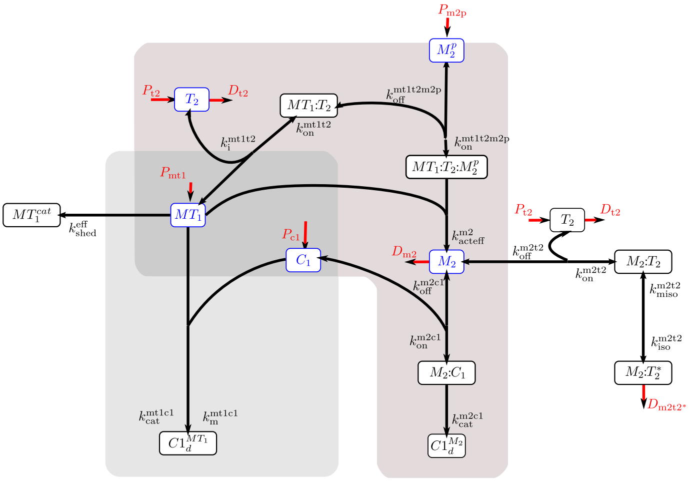
Open Model

To extend the model, we introduced production and degradation terms


M_2^p
T_2
$MT_1{:}T_2$
$k_{\mathrm{off}}^{\mathrm{mt1t2m2p}}$
$k_{\mathrm{on}}^{\mathrm{mt1t2}}$
$k_{\mathrm{i}}^{\mathrm{mt1t2}}$
$k_{\mathrm{on}}^{\mathrm{mt1t2m2p}}$
$MT_1{:}T_2{:}M_2^p$
MT_1^{cat}
MT_1
$k_{\mathrm{shed}}^{\mathrm{eff}}$
$k_{\mathrm{acteff}}^{\mathrm{m2}}$
T_2
C_1
M_2
$k_{\mathrm{off}}^{\mathrm{m2t2}}$
$M_2{:}T_2$
$k_{\mathrm{on}}^{\mathrm{m2t2}}$
$k_{\mathrm{off}}^{\mathrm{m2c1}}$
$k_{\mathrm{miso}}^{\mathrm{m2t2}}$
$k_{\mathrm{on}}^{\mathrm{m2c1}}$
$k_{\mathrm{iso}}^{\mathrm{m2t2}}$
$M_2{:}C_1$
$M_2{:}T_2^*$
$k_{\mathrm{cat}}^{\mathrm{mt1c1}}$
$k_{\mathrm{m}}^{\mathrm{mt1c1}}$
$k_{\mathrm{cat}}^{\mathrm{m2c1}}$
$C1_d^{MT_1}$
$C1_d^{M_2}$

Open Model

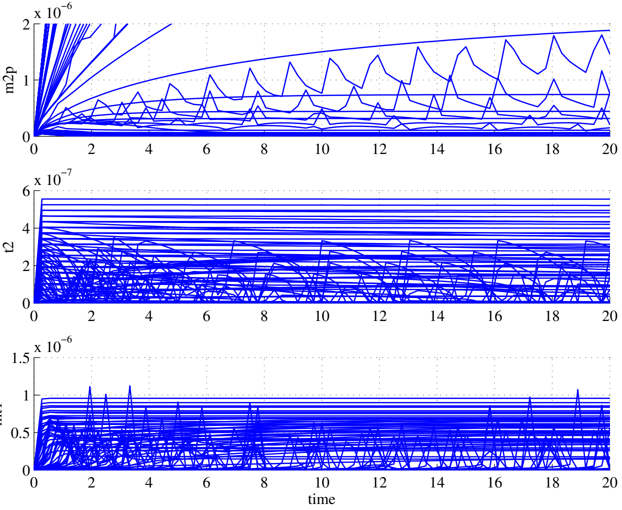
To extend the model, we introduced production and degradation terms



P_{m2p}
M_2^p
P_{t2}
T_2
D_{t2}
$MT_1{:}T_2$
$k_{\mathrm{off}}^{\mathrm{mt1t2m2p}}$
$k_{\mathrm{on}}^{\mathrm{mt1t2}}$
$k_{\mathrm{i}}^{\mathrm{mt1t2}}$
$k_{\mathrm{on}}^{\mathrm{mt1t2m2p}}$
$MT_1{:}T_2{:}M_2^p$
P_{mt1}
MT_1^{cat}
$k_{\mathrm{shed}}^{\mathrm{eff}}$
MT_1
P_{c1}
C_1
$k_{\mathrm{acteff}}^{\mathrm{m2}}$
D_{m2}
M_2
P_{t2}
T_2
D_{t2}
$k_{\mathrm{off}}^{\mathrm{m2t2}}$
$k_{\mathrm{on}}^{\mathrm{m2t2}}$
$M_2{:}T_2$
$k_{\mathrm{off}}^{\mathrm{m2c1}}$
$k_{\mathrm{miso}}^{\mathrm{m2t2}}$
$k_{\mathrm{on}}^{\mathrm{m2c1}}$
$k_{\mathrm{iso}}^{\mathrm{m2t2}}$
$M_2{:}C_1$
$M_2{:}T_2^*$
$D_{\mathrm{m2t2*}}$
$k_{\mathrm{cat}}^{\mathrm{mt1c1}}$
$k_{\mathrm{m}}^{\mathrm{mt1c1}}$
$k_{\mathrm{cat}}^{\mathrm{m2c1}}$
$C1_d^{MT_1}$
$C1_d^{M_2}$

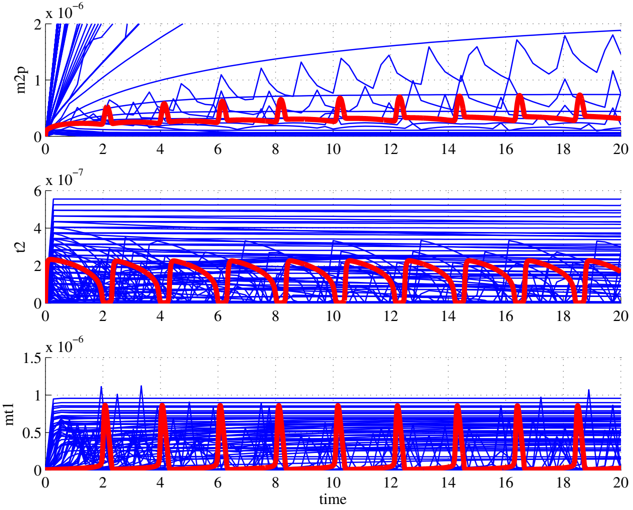
Open Model, Behaviors

As a consequence, the model exihibits more complex behaviors



Open Model, Behaviors

As a consequence, the model exihibits more complex behaviors, in particular oscillations



Detecting oscillations in M_2^P

We used the formula

$$\varphi_{\neg div} = \mathsf{alw}(M_2^P[t] < M_{2\mathsf{max}}^P)$$

to guarantee that the oscillation remains in a given range of amplitudes, in conjunction with

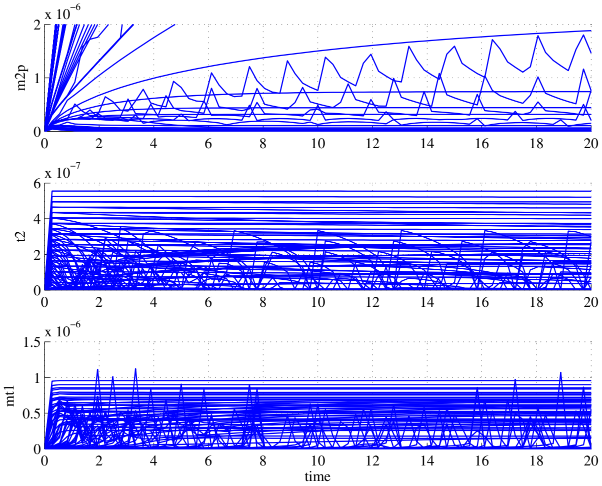
$$\mathsf{ev}\ \mathsf{alw}\ \left(\mathsf{ev}_{[0,6h)}\ \left(\frac{dM_2^P}{dt}[t] > k_h \wedge \mathsf{ev}_{[0,6h)}\ \left(\frac{dM_2^P}{dt}[t] < k_l\right)\right)\right)$$

The first “eventually” suppresses the transient phase before the oscillations and the “always” filters damped oscillations

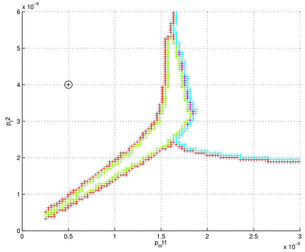
Then requires that the concentration of M_2^P alternates between periods when the it strictly increases and periods when it strictly decreases

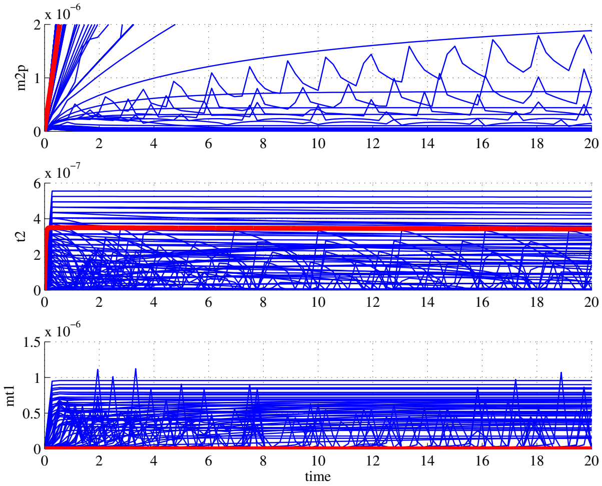
The formula filters oscillations with a period greater than $12h$

Oscillations Map

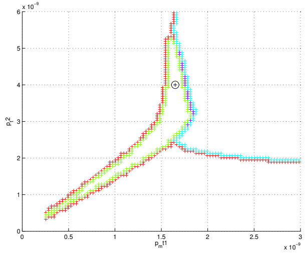


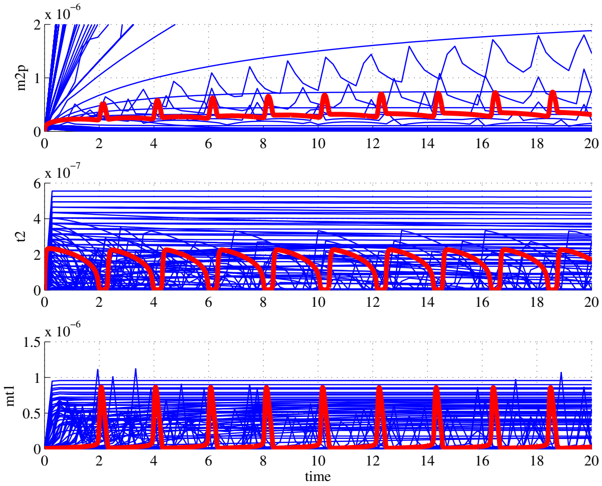
Oscillations Map



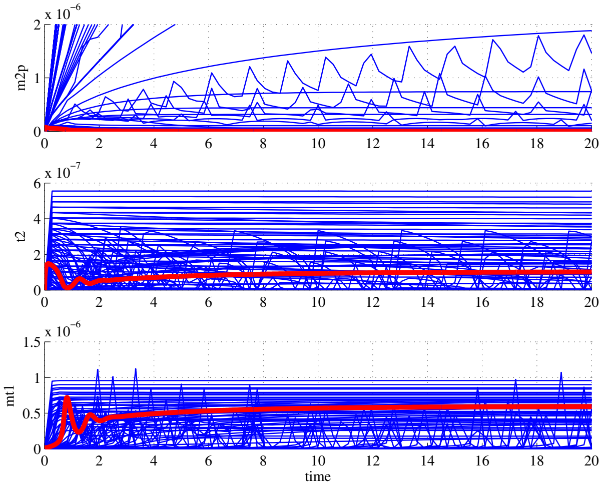


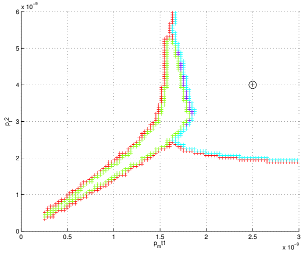
Oscillations Map



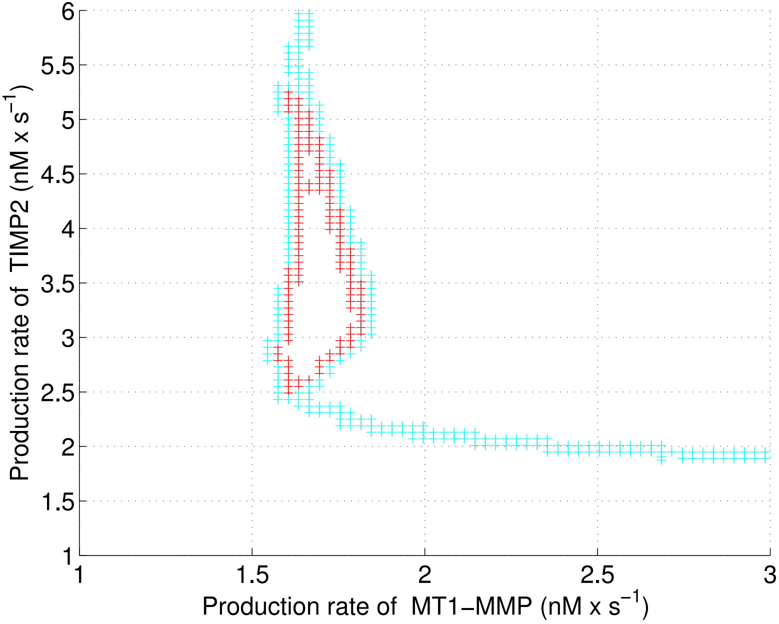


Oscillations Map

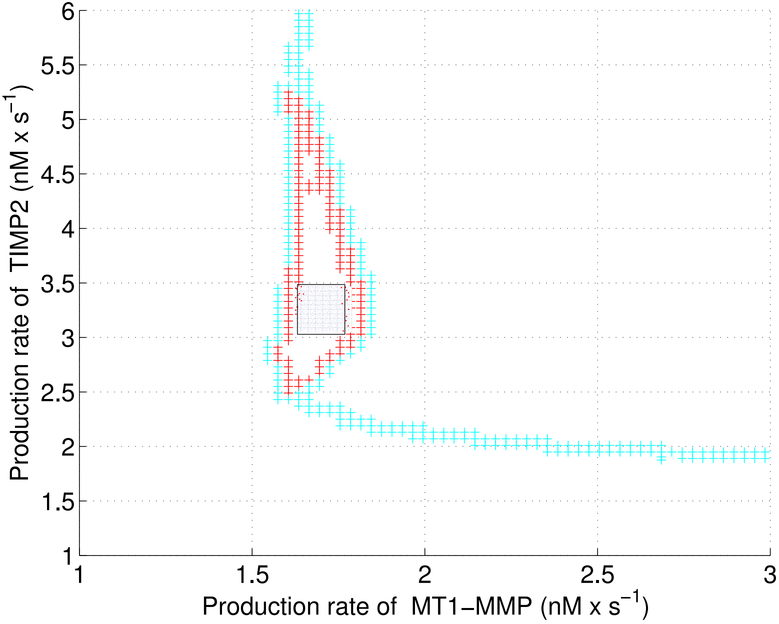




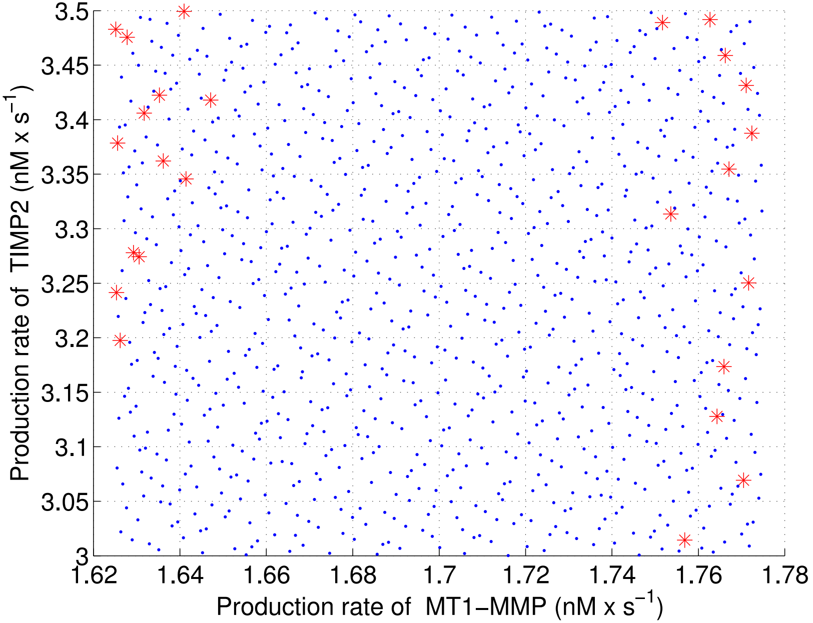
Oscillation, Robustness



Oscillation, Robustness



Oscillation, Robustness



Outline

Summary

As a field studying complex systems, systems biology can certainly benefit from the development of formal methods

Specifically, temporal logics were developped to study rigourously dynamic interactions and relations between systems components

This can be crucial in the developpment of useful (predictice, quantitative) mechanistic models

I genuinely believe that the work I presented is already a potentially useful framework for better understanding and designing models (though my experience with biological models is still young)

Summary

This work combines classical dynamical systems theory:

- Deterministic models of ordinary differential equations
- Uncertain initial conditions and parameters
- Numerical simulation, local and global sensitivity analysis

with

- A convenient query language to specify spatial and temporal constraints on variables and parameters
- A satisfaction function which computes by how much a simulation satisfies or violate a property
- Heuristics to synthesize sets of parameters generating trajectories satisfying a property

A toolbox implementation, Breach, is available on my webpage

Bibliography

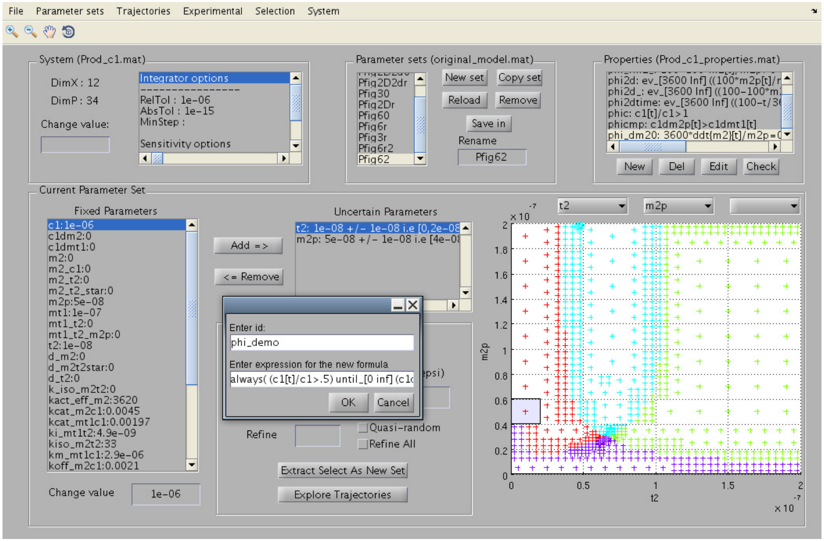
A. Donzé, C.J. Langmead, G. Clermont
Parameter Synthesis in Nonlinear Dynamical Systems: Application to Systems Biology. Journal of Computational Biology, 2010.

A. Donzé, O. Maler
Robust Satisfaction of Temporal Logic over Real-Valued Signals, FORMATS'10.

A. Donzé, E. Franchon, L. Gatepaille, O. Maler, P. Tracqui
Robustness Analysis and Behavior Discrimination in Enzymatic Reaction Networks, Submitted.

Thanks !

Breach, parameters et properties



System (Prod_c1.mat)

DimX : 12

DimP : 34

Change value:

Integrator options

RelTol : 1e-06
AbsTol : 1e-15
MinStep :

Sensitivity options

Parameter sets (original_model.mat)

Pfig2D2dr
Pfig30
Pfig2Dr
Pfig60
Pfig6r
Pfig3r
Pfig6r2
Pfig62

Rename: Pfig62

Properties (Prod_c1_properties.mat)

phi2d: ev_[3600 Inf] ((100*m2p[t]/r
phi2d_: ev_[3600 Inf] ((100-100*m
phi2dtime: ev_[3600 Inf] ((100-t/36
phic: c1[t]/c1>1
phicmp: c1dm2p[t]>c1dmt1[t]
phi_dm20: 3600*ddt{m2}[t]/m2p=0

Current Parameter Set

Fixed Parameters

c1:1e-06
c1dm2:0
c1dmt1:0
m2:0
m2_c1:0
m2_t2:0
m2_t2_star:0
m2p:5e-08
mt1:1e-07
mt1_t2:0
mt1_t2_m2p:0
t2:1e-08
d_m2:0
d_m2t2star:0
d_t2:0
k_iso_m2t2:0
kact_eff_m2:3620
kcat_m2c1:0.0045
kcat_mt1c1:0.00197
ki_mt1t2:4.9e-09
kiso_m2t2:33
km_mt1c1:2.9e-06
koff_m2c1:0.0021

Change value: 1e-06

Uncertain Parameters

t2: 1e-08 +/- 1e-08 i.e [0,2e-08
m2p: 5e-08 +/- 1e-08 i.e [4e-08

Enter id:

phi_demo

Enter expression for the new formula

always((c1[t]/c1>.5) until_[0 inf] (c1c

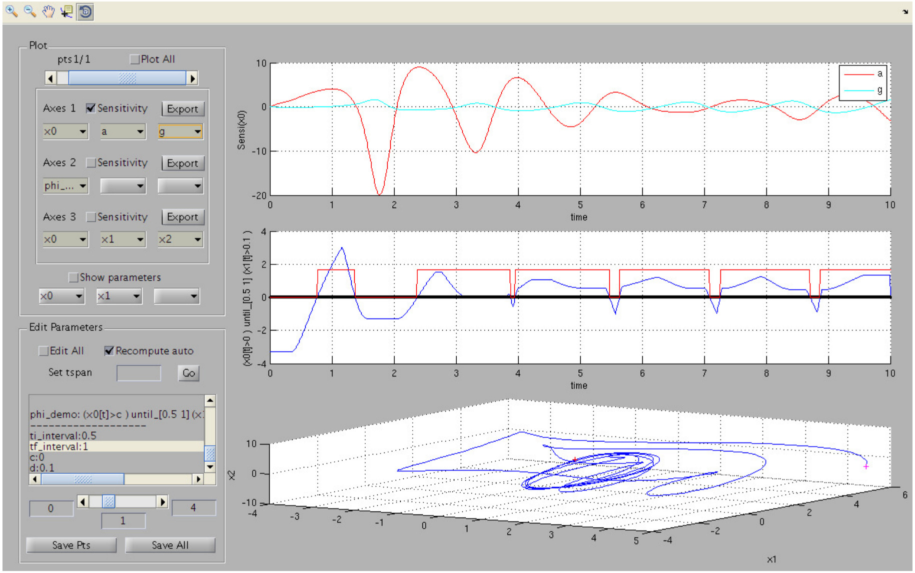
epsi)

Refine

Quasi-random

Refine All

Breach, trajectories



Plot

pts 1/1 ☐ Plot All

Axes 1 ☑ Sensitivity Export

x0 a g

Axes 2 ☐ Sensitivity Export

phi_...

Axes 3 ☐ Sensitivity Export

x0 x1 x2

☐ Show parameters

x0 x1

Edit Parameters

☐ Edit All ☑ Recompute auto

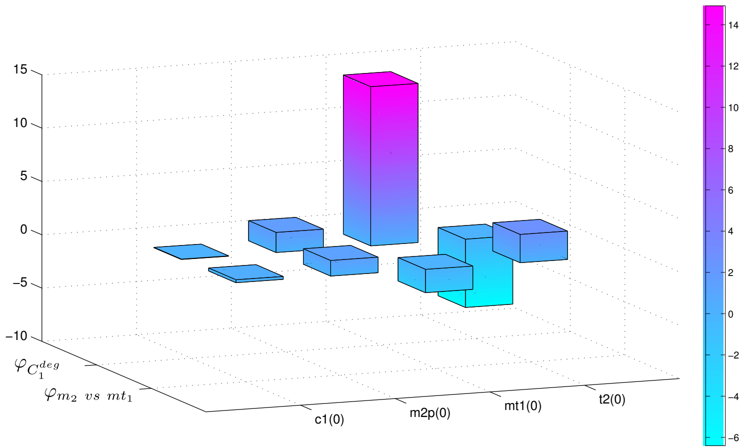
Set tspan Go

```
phi_demo: (x0[t]>c ) until_[0.5 1] (x
-------------------
ti_interval:0.5
tf_interval:1
c:0
d:0.1
```

0 1 4

Save Pts Save All

Global Sensitivities histogram


$\varphi_{C_1^{deg}}$
$\varphi_{m_2\ vs\ mt_1}$
c1(0)
m2p(0)
mt1(0)
t2(0)