17/05/2023

An automated technique for model parameterization using data

B. Delahaye – Nantes Université / LS2N

LABORATOIRE DES SCIENCES DU NUMÉRIQUE DE NANTES

cnrs

Nantes Université

Model Checking

Automated technique for **proving properties** of **systems** using **models**

Algorithms / Tools

Formal guarantee

- Mathematical certainty
- Precision
- Error rate

Anything we want?

- Behaviour
- Qualitative
- Quantitative

Real object

- Software
- Electronics
- Physics
- Biology

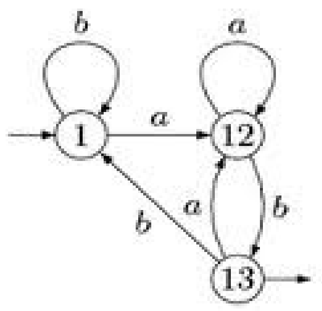
Abstract object

- Equation
- Program
- Automaton

Types of models

Computer Science : Formal Models

Automata



Dedicated languages

$$\frac{X \triangleq E \in \Gamma \quad E \xrightarrow{\mu} E'}{X \xrightarrow{\mu} E'}$$

```
left : bool; // being repaired?

[startLeft] !left & (left_n<left_mx) -> 1 : (left'=true);
```

Extended with all the details we need:

- Costs
- Time
- Probabilities
- Discrete / Continuous
- Hybrid

$$x(\xi, t) = (R_0 - \epsilon) \int_{-1}^{0} G(\xi, \xi') d\xi'$$

But also: **Standard programs**

```
 # Masse initiale en poids humide (en g)
            CW=0.0036*WM;
            cw=CW
            for i in Temps_D:
                if score_degrowth<lim_degrowth:
                # Application du modèle
                    X0=[CW, T, 0]
                    [Clear, Ingest, Assim, Respi, Excre, Prod, Egest, Scope_grow
th] = pelagia_feed(X0,param1,param2,0);
                    # Stockage des variables
                    CW=CW+ Scope_growth*dt;
                    wm=CW*(100/0.36);
                    pelsize=(wm/a)**(1/b);
                    if not pelsize>0:
                        print(pelsize)
                    cw=np.append(cw,CW) ;
                    WM=np.append(WM,wm);
                    Size1=np.append(Size1,pelsize);

                    val=vec_D[int(i*10)]
                    val=val[0]

                    if (val):
                        d=vec_D[int(i*10)]
                        d=d[1]
                        d=d[0]
                        if not (degrowth[d,1]-degrowth[d,2]<Size1[x_d[d]]<degrow
th[d,1]+degrowth[d,2]):
                            score_degrowth=score_degrowth+1

                else:
                    bool_break1=True
                    break




U(DOS)---  modele2017-02-20-1.py   43% L241   (Python)
```

But also: **Differential Equation Systems**

$$\begin{cases} \dfrac{du}{dt}(t) = \rho v(t) - \gamma\big(v(t)\big)u(t) - \alpha u(t) v(t) \left(1 - \dfrac{u(t)+v(t)}{K_{\max}}\right), \\ \dfrac{dv}{dt}(t) = \alpha u(t) v(t) \left(1 - \dfrac{u(t)+v(t)}{K_{\max}}\right) - h v(t), \end{cases}$$

LABORATOIRE DES SCIENCES DU NUMÉRIQUE DE NANTES

Nantes Université

Types of Properties

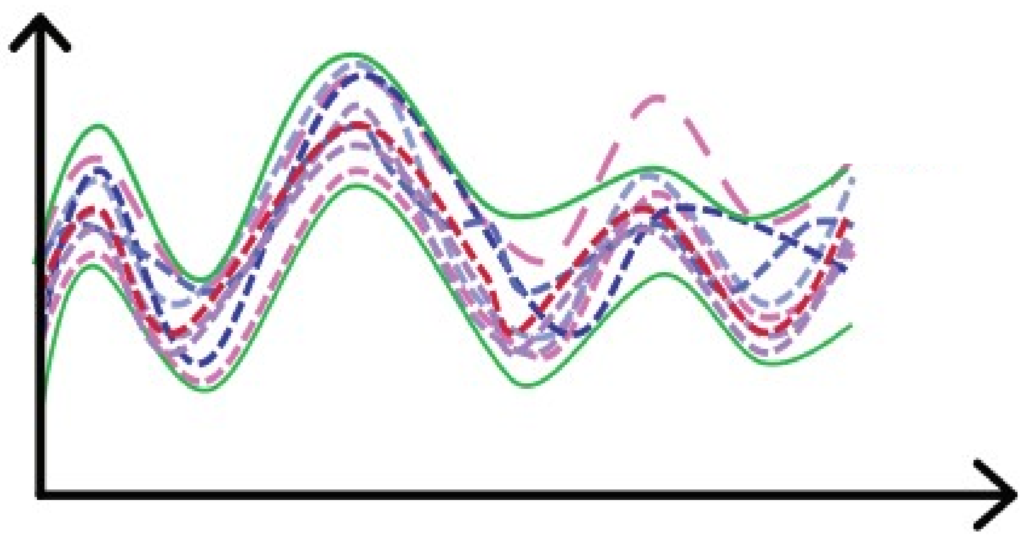
Qualitative / Quantitative and Dynamic

- Linear properties (traces)
- Ex : The size is always smaller than 4cm

- Branching properties
- Ex : Regardless of the food supply, the jellyfish grows

- + necessary details (Time, Probabilities, …)
- Ex : The **probability** that the jellyfish reaches a **size > 2cm** in less than **15 days** is always greater than **10 %**

In some cases (cf SMC), it is enough to give an « oracle »



… But mostly using dedicated Languages

$$TS_C \models \Box\,(\,y \rightarrow (\bigcirc\neg y \wedge \bigcirc\bigcirc\neg y)\,)$$

$$\mathbb{P}_{>0}(\bigcirc\,\mathbb{P}_{>0}(\Diamond\Phi)\,) \equiv \mathbb{P}_{>0}(\Diamond\,\mathbb{P}_{>0}(\bigcirc\,\Phi)\,).$$

Statistical Model Checking

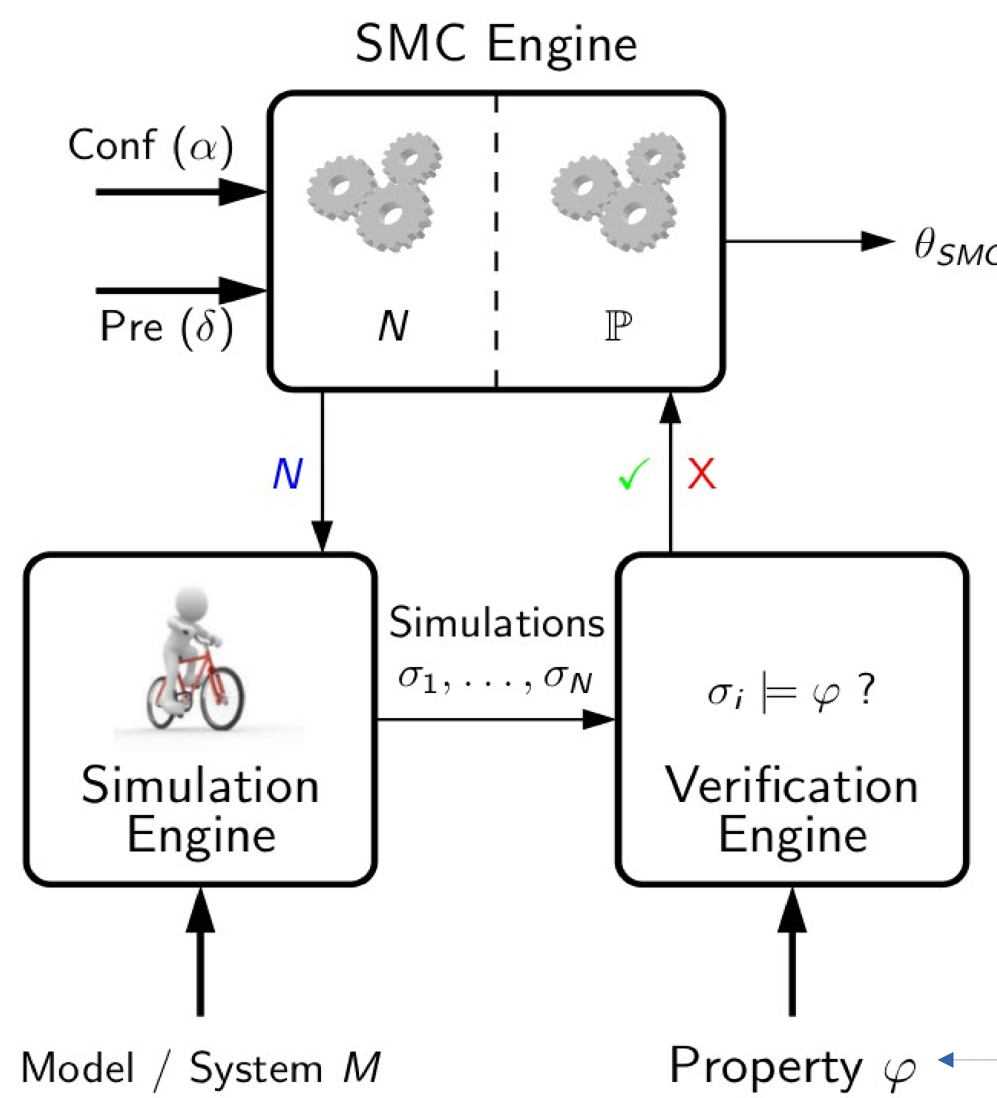
<u>Goal</u>: Estimate the probability with which a model satisfies a property

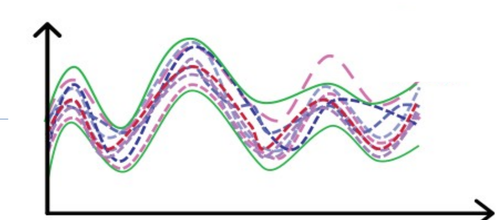
Precision (δ) / Error rate(α) guaranteed

Based on simulations

- Trace properties
- Purely probabilistic models
- Executable models

```
# Masse initiale en poids humide (en g)
            CW=0.0036*WM;
            cw=CW
            for i in Temps_D:
                if score_degrowth<lim_degrowth:
                # Application du modèle
                    X0=[CW, T, 0]
                    [Clear, Ingest, Assim, Respi, Excre, Prod, Egest, Scope_grow
th] = pelagia_feed(X0,param1,param2,0);
                    # Stockage des variables
                    CW=CW+ Scope_growth*dt;
                    wm=CW*(100/0.36);
                    pelsize=(wm/a)**(1/b);
                    if not pelsize>0:
                        print(pelsize)
                    cw=np.append(cw,CW) ;
                    WM=np.append(WM,wm);
                    Size1=np.append(Size1,pelsize);

                    val=vec_D[int(i*10)]
                    val=val[0]

                    if (val):
                        d=vec_D[int(i*10)]
                        d=d[1]
                        d=d[0]
                        if not (degrowth[d,1]-degrowth[d,2]<Size1[x_d[d]]<degrow
th[d,1]+degrowth[d,2]):
                            score_degrowth=score_degrowth+1

                else:
                    bool_break1=True
                    break
U(DOS)---  modele2017-02-20-1.py   43% L241   (Python)
```

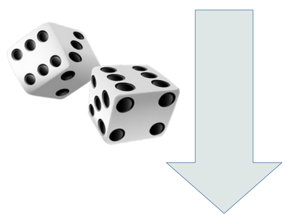




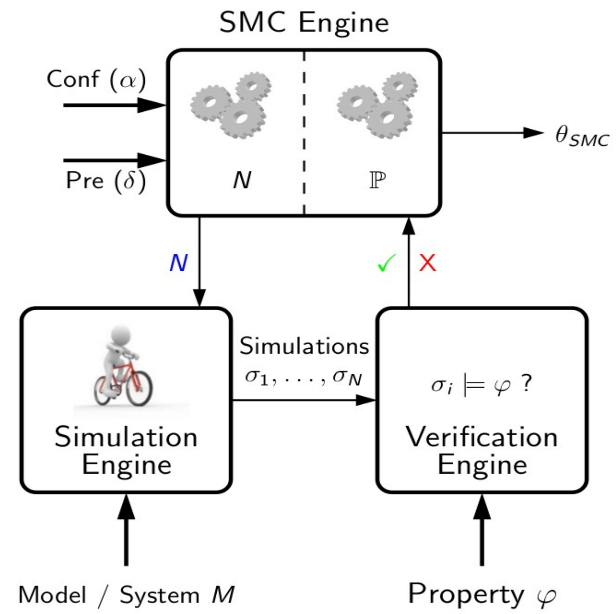
SMC in practice

Two ways of using SMC

Uncertainties / Variability



Model
(Already parametrised)



Guaranteed Certification

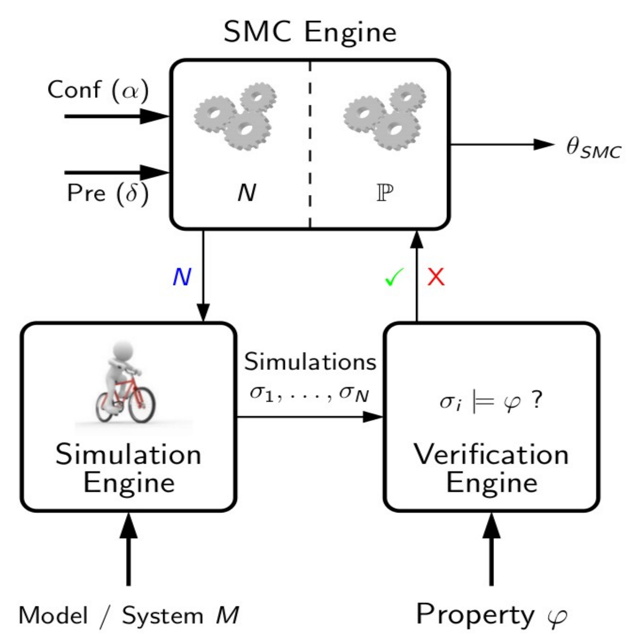
- Fixed Precision
- Fixed Error Rate
- Probabilistic guarantee

1 **Sensitivity / Variability analysis**

Model
(with or w/o uncertainties)

Unknown parameters
(with variability)

Experimental data

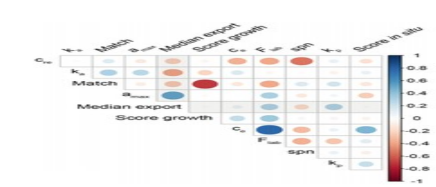


Optimal parameter values

- Formal guarantees
- Considered variability

\+

Statistical analysis

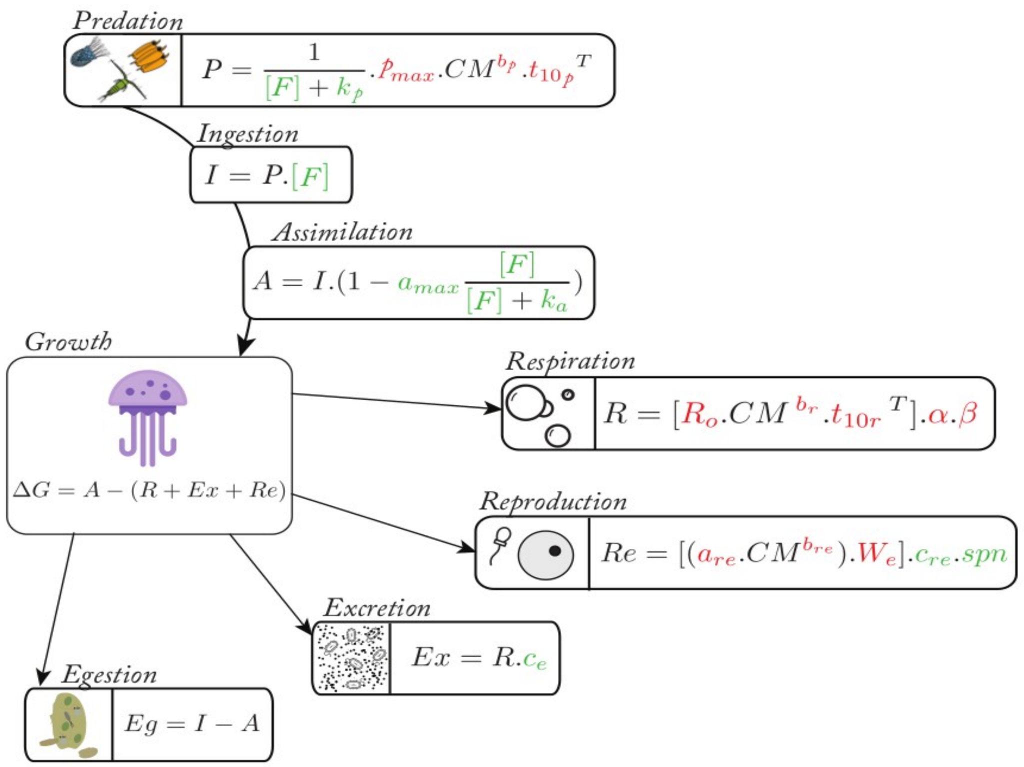


2 **Parameterization**

Automated parameterization using SMC

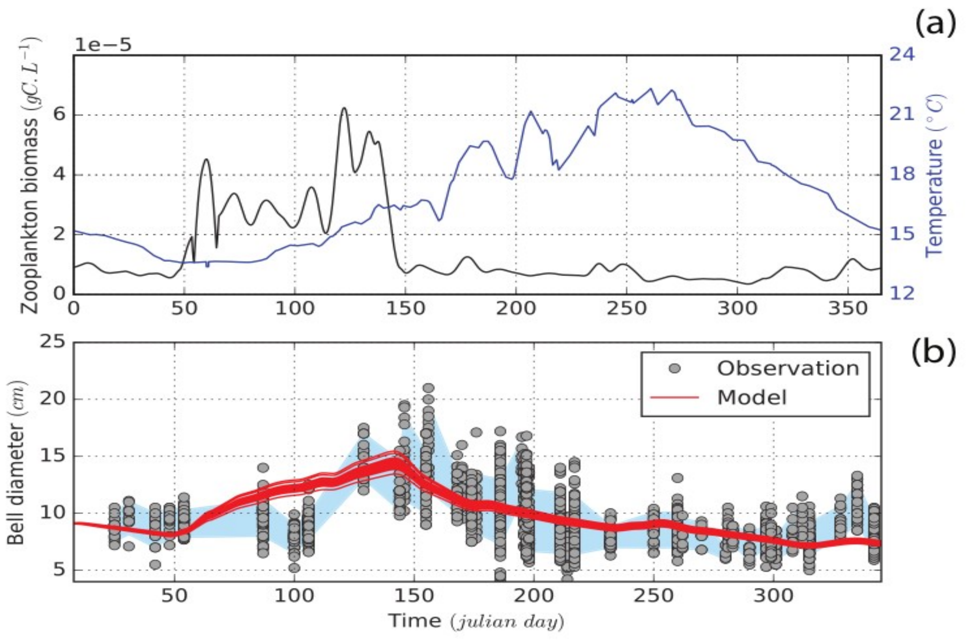
Probabilistic modeling to estimate jellyfish ecophysiological properties and size distributions

Simon Ramondenc[1*], Damien Eveillard[2], Lionel Guidi[1,3], Fabien Lombard[1], Benoît Delahaye[2]



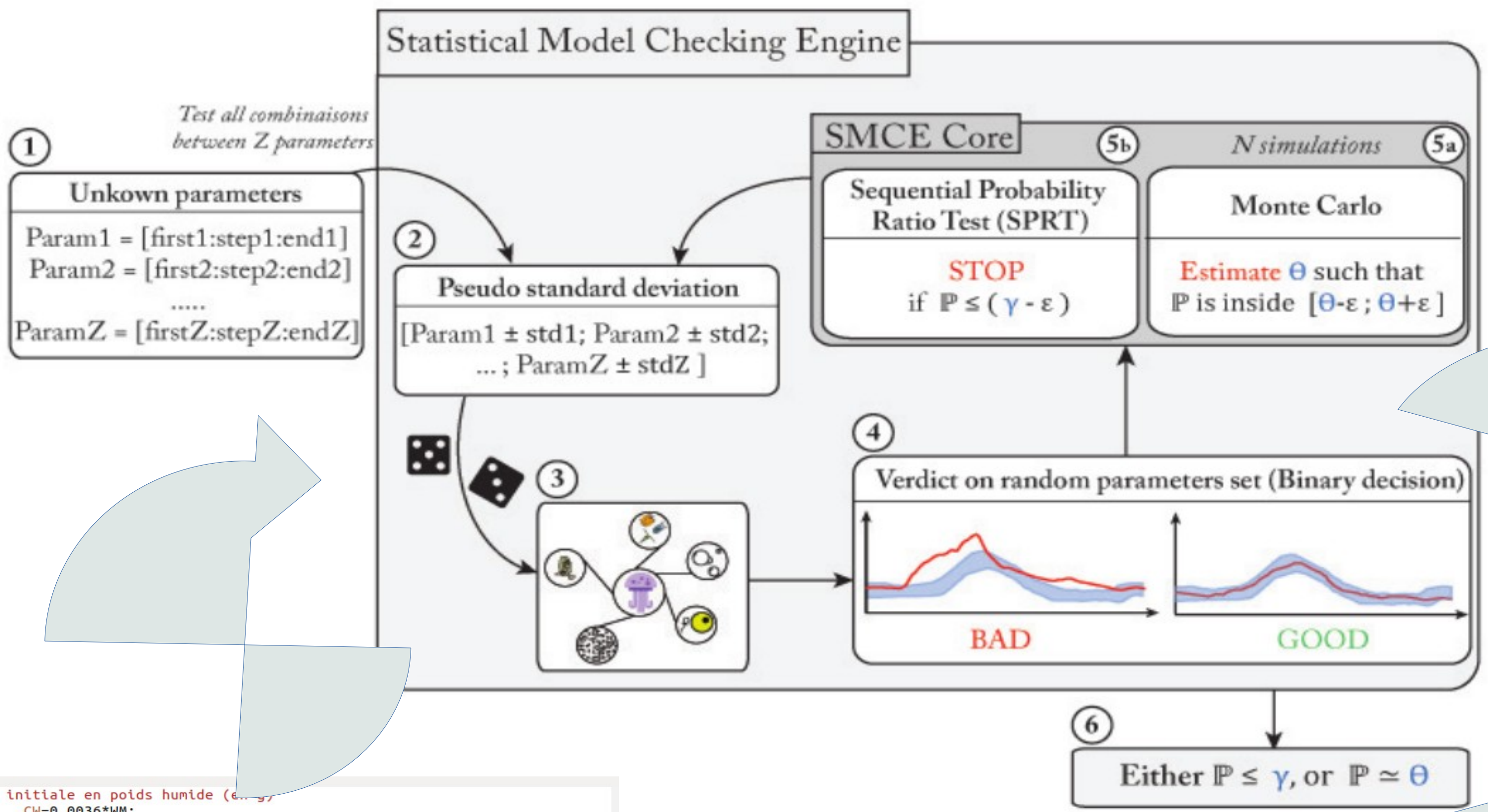
+ experimental dataset

```
  # Masse initiale en poids humide (en g)
            CW=0.0036*WM;
            cw=CW
            for i in Temps_D:
                if score_degrowth<lim_degrowth:
                # Application du modèle
                    X0=[CW, T, 0]
                    [Clear, Ingest, Assim, Respi, Excre, Prod, Egest, Scope_grow
th] = pelagia_feed(X0,param1,param2,0);
                    # Stockage des variables
                    CW=CW+ Scope_growth*dt;
                    wm=CW*(100/0.36);
                    pelsize=(wm/a)**(1/b);
                    if not pelsize>0:
                        print(pelsize)
                    cw=np.append(cw,CW) ;
                    WM=np.append(WM,wm);
                    Size1=np.append(Size1,pelsize);

                    val=vec_D[int(i*10)]
                    val=val[0]

                    if (val):
                        d=vec_D[int(i*10)]
                        d=d[1]
                        d=d[0]
                        if not (degrowth[d,1]-degrowth[d,2]<Size1[x_d[d]]<degrow
th[d,1]+degrowth[d,2]):
                            score_degrowth=score_degrowth+1

                else:
                    bool_break1=True
                    break
U(DOS)---  modele2017-02-20-1.py    43% L241    (Python)
```



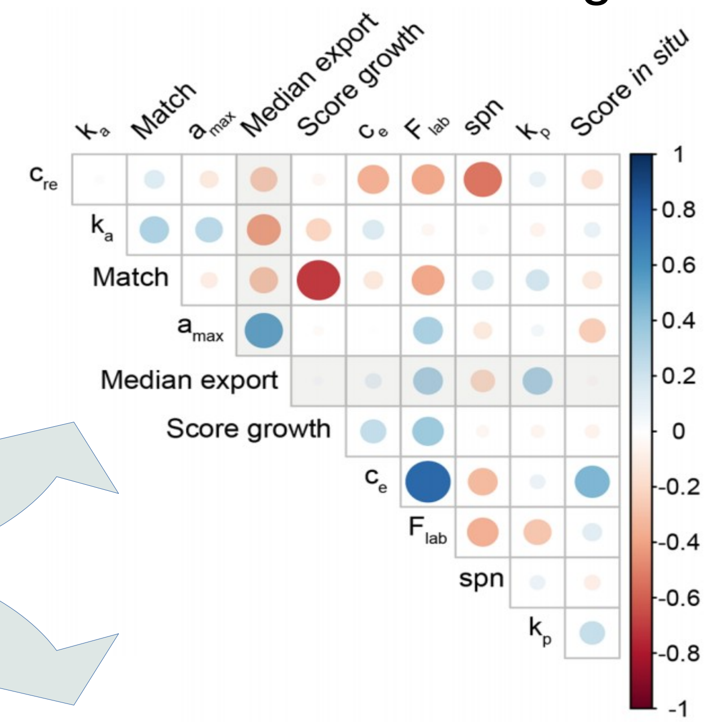
Probabilistic modeling to estimate jellyfish ecophysiological properties and size distributions

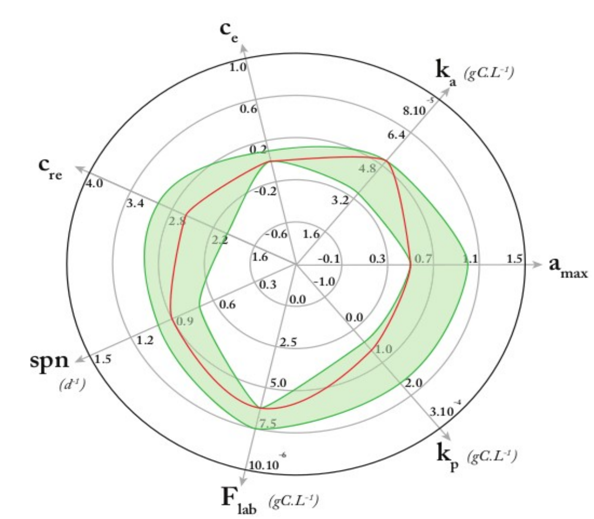
Simon Ramondenc[1*], Damien Eveillard[2], Lionel Guidi[1,3], Fabien Lombard[1], Benoît Delahaye[2]




```python
# Masse initiale en poids humide (en g)
            CW=0.0036*WM;
            cw=CW
            for i in Temps_D:
                if score_degrowth<lim_degrowth:
                # Application du modèle
                    X0=[CW, T, 0]
                    [Clear, Ingest, Assim, Respi, Excre, Prod, Egest, Scope_grow
th] = pelagia_feed(X0,param1,param2,0);
                    # Stockage des variables
                    CW=CW+ Scope_growth*dt;
                    wm=CW*(100/0.36);
                    pelsize=(wm/a)**(1/b);
                    if not pelsize>0:
                        print(pelsize)
                    cw=np.append(cw,CW) ;
                    WM=np.append(WM,wm);
                    Size1=np.append(Size1,pelsize);

                    val=vec_D[int(i*10)]
                    val=val[0]

                    if (val):
                        d=vec_D[int(i*10)]
                        d=d[1]
                        d=d[0]
                        if not (degrowth[d,1]-degrowth[d,2]<Size1[x_d[d]]<degrow
th[d,1]+degrowth[d,2]):
                            score_degrowth=score_degrowth+1

                else:
                    bool_break1=True
                    break

U(DOS)---  modele2017-02-20-1.py   43% L241   (Python)
```

Correlogram

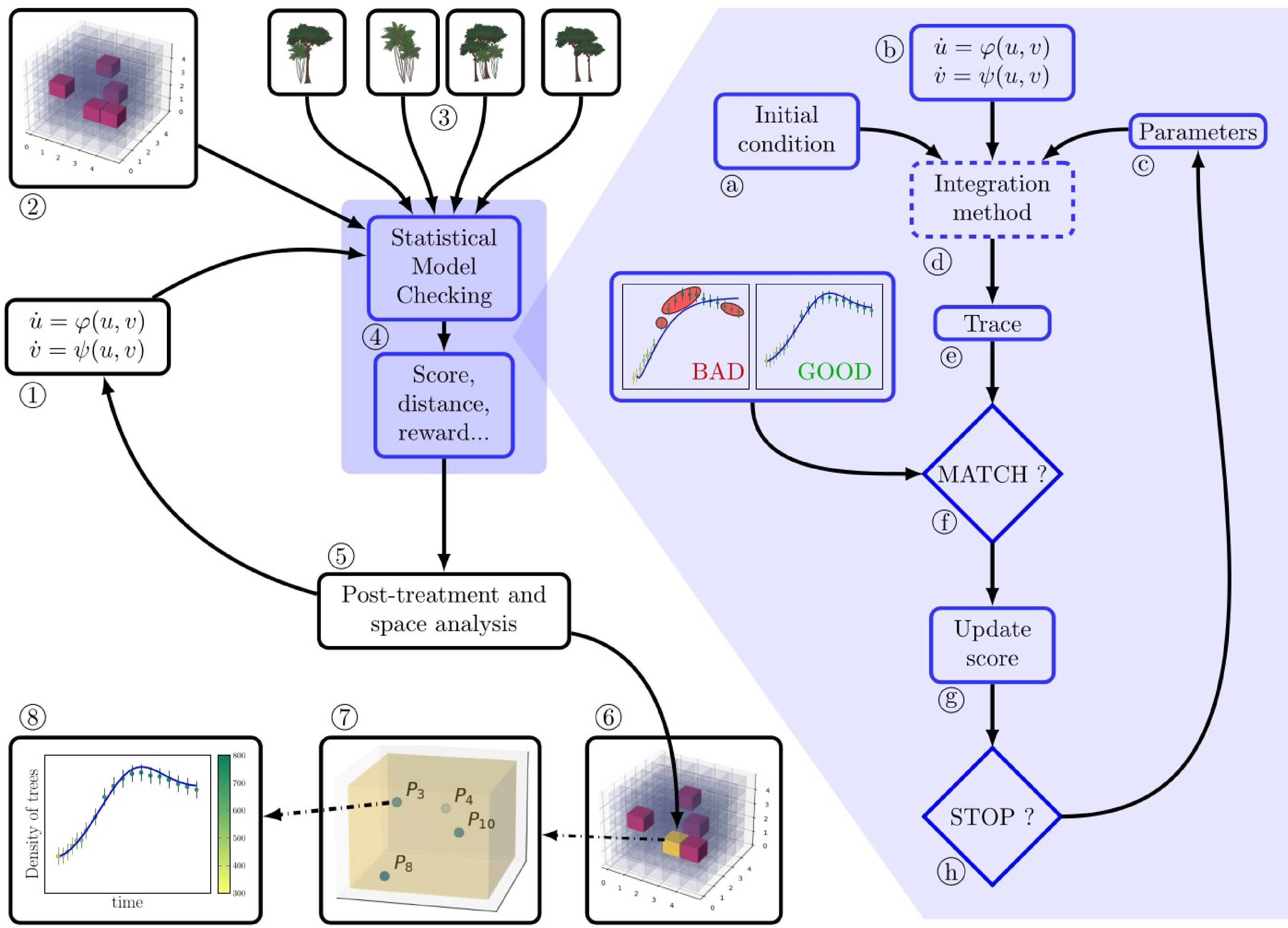




Optimal parameter values

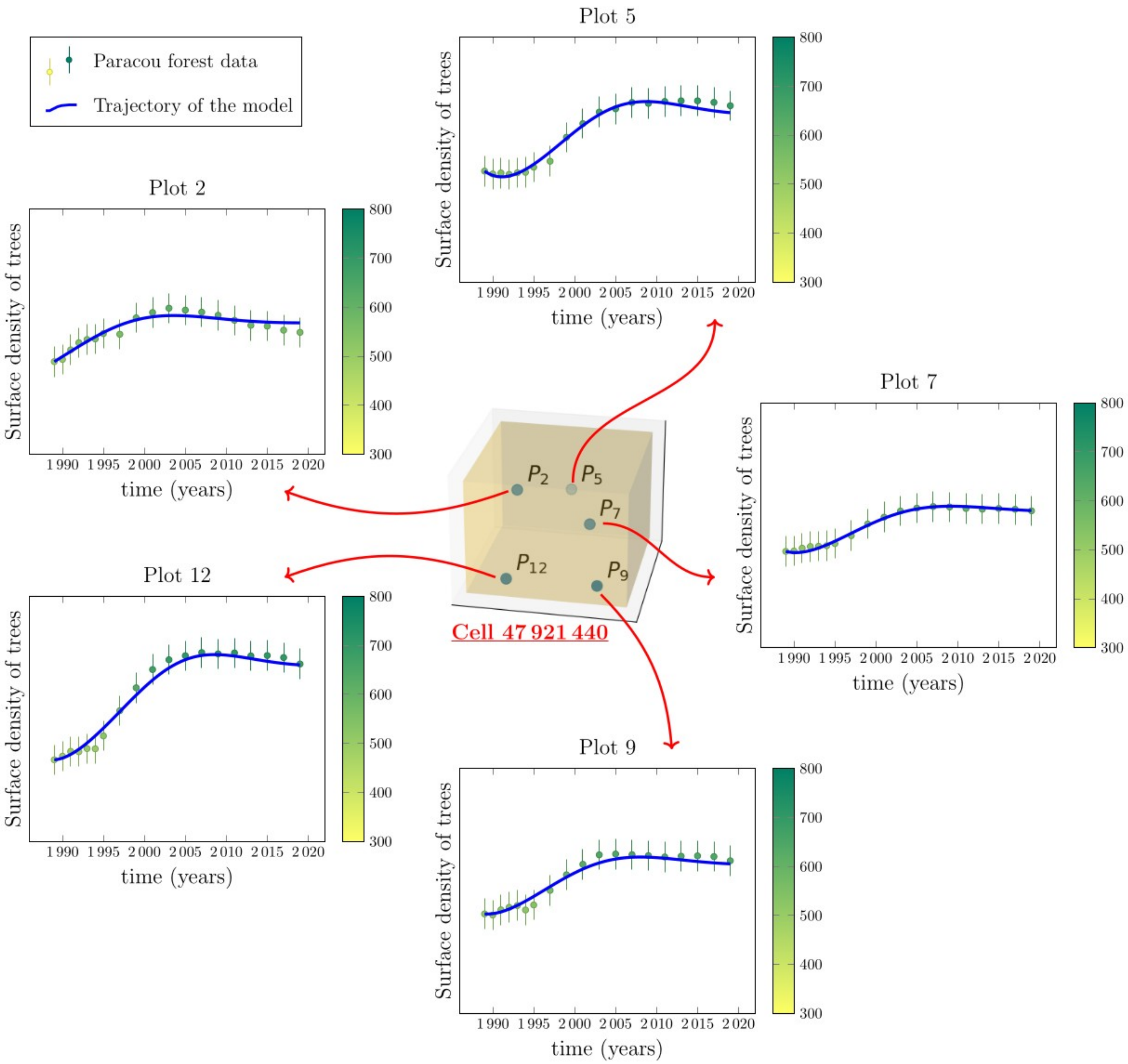
Computational assessment of Amazon forest patches regrowth capacity under strong spatial variability

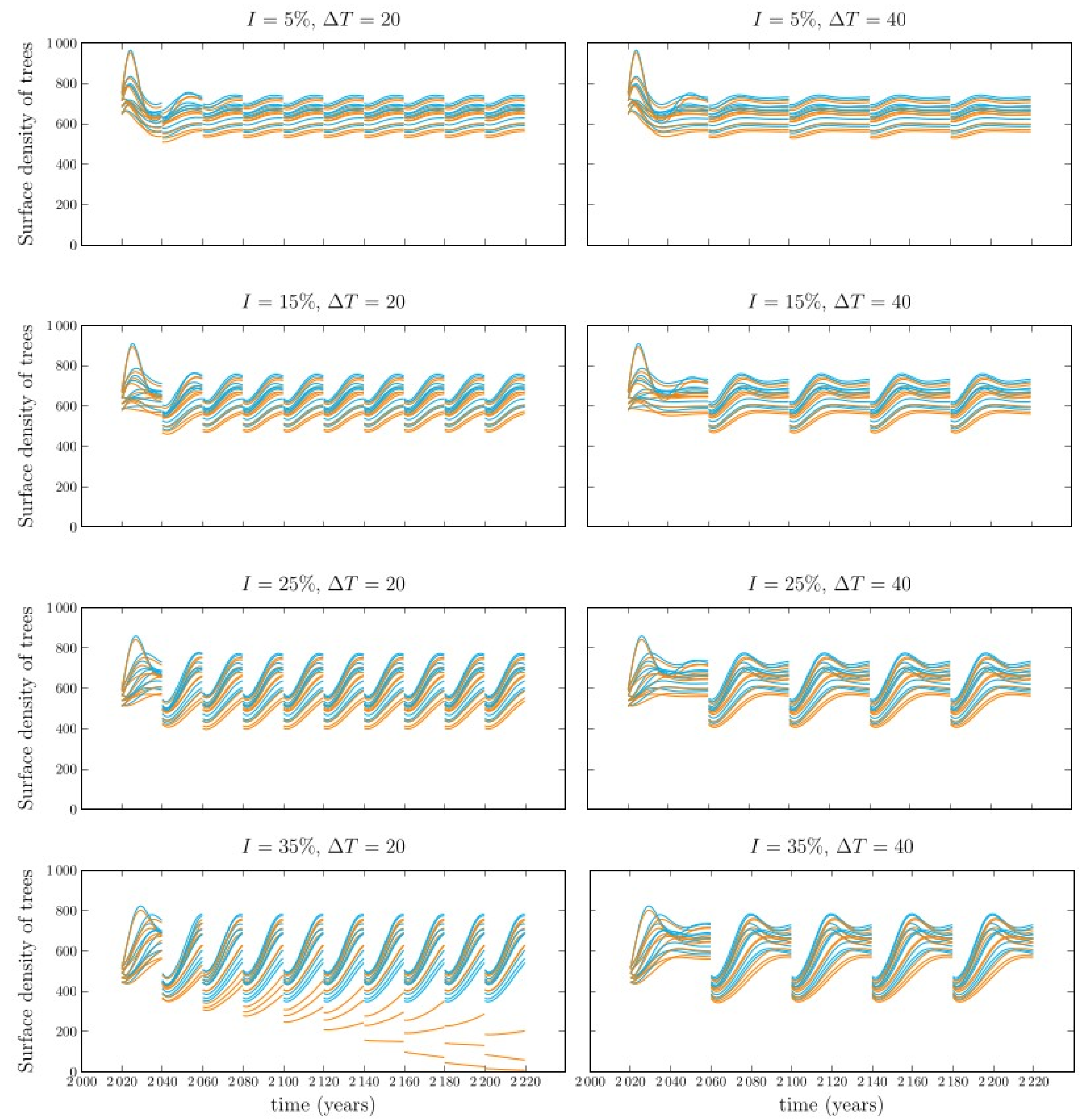
Authors (by alphabetical order): Gilles Ardourel[1], Guillaume Cantin[1,2] Benoît Delahaye[1], Géraldine Derroire[3], Beatriz M.Funatsu[4], David Julien[1].



Computational assessment of Amazon forest patches regrowth capacity under strong spatial variability

Authors (by alphabetical order): Gilles Ardourel[1], Guillaume Cantin[1,2] Benoît Delahaye[1], Géraldine Derroire[3], Beatriz M.Funatsu[4], David Julien[1].





LABORATOIRE DES SCIENCES DU NUMÉRIQUE DE NANTES

Nantes Université

Conclusion

Take home message

Automated parameterization technique

- Based on formal methods : Certified guarantees
- Few restrictions on the input model
- Uncertainties / Variability taken into account

Ongoing / Future work

- Coupling with control techniques
- Use of Neural Networks when ODEs are not available
- Automated verification/evaluation of *model-based intellectual rights?*

Unrelated food for thoughts

Resulting from discussions with Damien Eveillard

LABORATOIRE DES SCIENCES DU NUMÉRIQUE DE NANTES

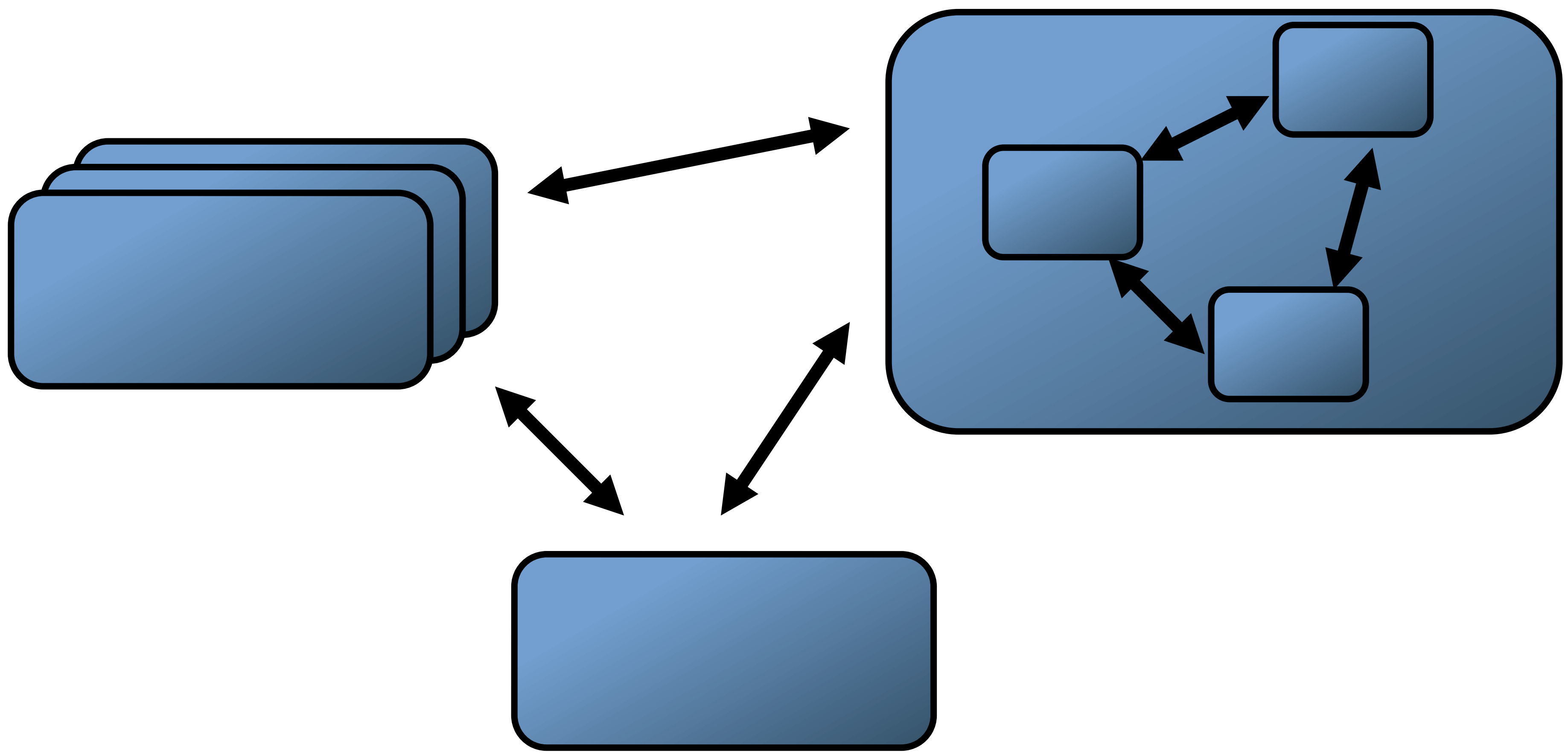
Nantes Université

Words and pictures

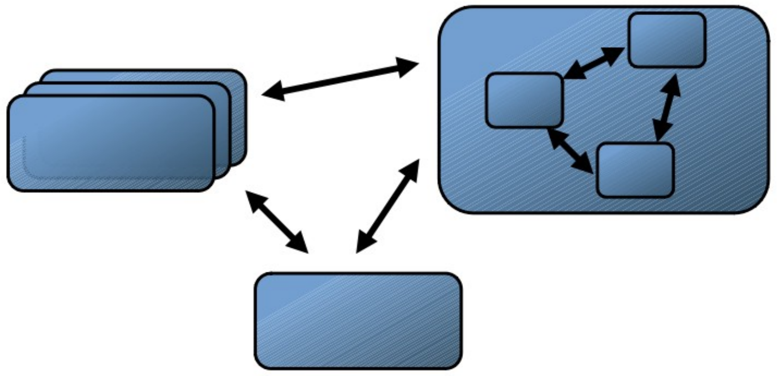
Words that ring a bell

- **Models**
- **Processes**
- **Composition**
- **Combination**
- **Digital Twin**
- **Heterogeneous**
- **Hierarchical**
-

One picture



Component algebras and interface theories

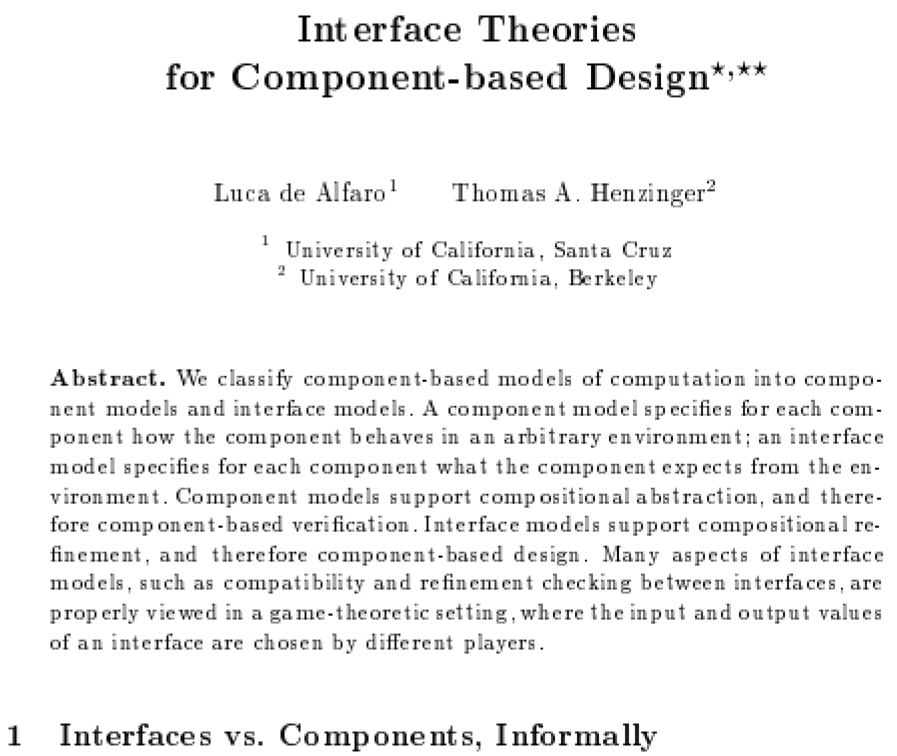


Component algebras:

How to formalize what is inside a given component
How to combine different views of a given process

Interface theories:

How to formalize the interactions between components



Interface Theories
for Component-based Design*,**

Luca de Alfaro[1] Thomas A. Henzinger[2]

[1] University of California, Santa Cruz
[2] University of California, Berkeley

Abstract. We classify component-based models of computation into component models and interface models. A component model specifies for each component how the component behaves in an arbitrary environment; an interface model specifies for each component what the component expects from the environment. Component models support compositional abstraction, and therefore component-based verification. Interface models support compositional refinement, and therefore component-based design. Many aspects of interface models, such as compatibility and refinement checking between interfaces, are properly viewed in a game-theoretic setting, where the input and output values of an interface are chosen by different players.

1 Interfaces vs. Components, Informally