Introduction to the exploration of epidemiological models using EMULSION

EGAAL doctoral training 2021 / BIOEPAR / Sébastien Picault

RÉPUBLIQUE FRANÇAISE
Liberté
Égalité
Fraternité

INRAE

Oniris
ÉCOLE NATIONALE

DOCTORAT / ECOLOGIE
BRETAGNE \ GEOSCIENCES
LOIRE / AGRONOMIE ALIMENTATION

About us

DYNAMO team, BIOEPAR

- **methods** in epidemiological modelling
  stochastic + mechanistic, parameter inference
- **applications**: endemic diseases of livestock (mainly)
- **understand, predict and control** pathogen spread at **multiple scales**
- assess the impact of diseases and control measures

Trainers & contributors

- **Sébastien Picault** – researcher
  AI, simulation, modelling
- **Sandie Arnoux** – engineer
  software engineering, modelling
- **Vianney Sicard** – engineer/Ph.D.
  software engineering → modelling
- Pauline Ezanno – senior scientist
  epidemiological modelling
- Gaël Beaunée – researcher
  inference methods, modelling

CC BY NC ND

Outline of the training

- Mechanistic epidemiological models with EMULSION
  - reminders on epidemiological modelling
  - classical issues in model development
  - EMULSION: why and how
- From compartments to individual-based models
  exercises of progressive complexity, to learn designing a model with EMULSION
- Scale change: from within-population to between-population models
- Connection between a mechanistic model and data
- Case study: reimplementing a complex model from literature
- Integration within a workflow: scenario comparison

CC BY NC ND

– 1 –

Epidemiological modelling with EMULSION

Key concepts: model

Experimental sciences aim at studying observable natural phenomena
→ build a **deliberately simplified representation** of reality = **a model** = consistent set of **assumptions**

A model is not "true" or "false" but:

- more or less **compatible with a theory** (larger set of assumptions)
- more or less **fecund** (lever → new insights)

... and can have several goals:

- **describe and classify** existing knowledge
- **predict the consequences** of given conditions
- **help understand** the underlying mechanisms

CC BY NC ND

Key concepts: experiment

experiments = reproducing phenomena within controlled conditions to **corroborate** or **refute** a model/assumption

but experiments are not always possible:

- rarity of the target phenomenon
  few observations, sparse data
- impossibility to control parameters
  - potential system destruction
    ecosystems, global trade...
  - ethical reasons
    human/animal welfare/privacy

→ requires **simulation**

CC BY NC ND

Key concepts: simulation

Simulation of a phenomenon:

- test on a **substitute** of real system
- assuming that the substitute reproduces **accurately** the **essential relevant features** of the real system

Examples:

- analogical mock-up
- numerical methods (equation-like)
- agent-based simulation (rule-based)

CC BY NC ND

Analogical simulation

- convenient materials adapted to scale change
- high cost !
- scenario comparison ?

CC BY NC ND

Numerical simulation

- huge amout of methods: equation solvers, Monte-Carlo methods, production systems, neural networks, agent-based simulation...
- "Black boxes" vs. explicit knowledge/processes

Criteria for choosing a method:

- mathematical vs. rule-based model (e.g. physics vs. ethology)
- average behaviour vs. individual variability
- goal: predict vs. understand
- data availability vs. domain-specific knowledge (data-based vs. mechanistic)
- spatial or temporal specificities
- capability to support assumption revision easily
- interaction with non-modellers scientists

CC BY NC ND

Epidemiological mechanistic models: foundations

- impact of smallpox inoculation (D. Bernoulli, 1760): mathematical study

- compartmental models (A. G. McKendrick & W. O. Kermack, 1927)
  → a mechanistic description of the infectious process
  - **states:** Susceptible, Infectious, Resistant...
    → underly the visible part of a disease (clinical signs...)

  - **rates:** indirect observations! e.g. transmission rate = probability of contact × probability of transmission per contact (depends on excretion by infectious, susceptibility...)

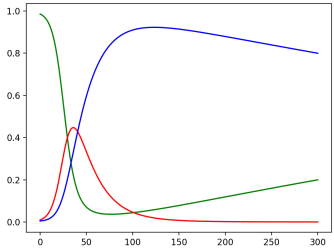
  - **indicators**: R_0 (basic reproduction number): expected total number of direct cases caused by 1 infectious individual in a fully susceptible population

CC BY NC ND

Epidemiological mechanistic models: compartments

- based on differential equations (chemical-like), e.g. for SIR with demography:

$$\begin{cases} \frac{dS}{dt} = -\beta \frac{I}{N} S + bN - \mu S \\ \frac{dI}{dt} = \beta \frac{I}{N} S - \gamma I - \mu I \\ \frac{dR}{dt} = \gamma I - \mu R \end{cases}$$

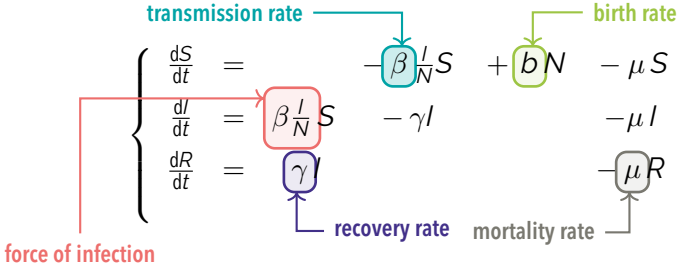


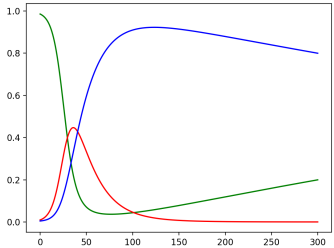
- enables analytical study (equilibria, R_0, chaotic behaviour...) + easy to compute (solvers)
  e.g. here: $R_0 = \frac{\beta}{\gamma+\mu}$

CC BY NC ND

Epidemiological mechanistic models: compartments

- based on differential equations (chemical-like), e.g. for SIR with demography:





- enables analytical study (equilibria, R_0, chaotic behaviour...) + easy to compute (solvers)
  e.g. here: $R_0 = \frac{\beta}{\gamma+\mu}$

CC BY NC ND

Limitations of equation-based approaches

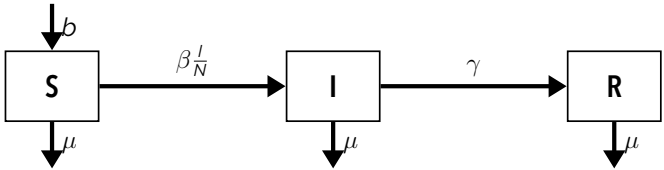
Strong underlying assumptions

- **continuous populations** $\rightarrow$ large populations, no dramatic event
  "artificial" persistence of infection ($I > 0$)
- **average behaviour** $\rightarrow$ homogeneous individuals, homogeneous mixing
- **determinism** $\leftarrow$ large populations + average behaviour
  smaller and more diversified populations are rather stochastic

Requires mathematical skills!
$\rightarrow$ hinders interaction with non-modellers scientists
esp. for assumption validation/revision

CC BY NC ND

An alternative representation: flow diagrams



- visual representation instead of mathematical formalism
- formally equivalent to an ODE system...
- ... assuming strict conventional denotations (e.g. arrow semantics)
- ... and many implicit assumptions
- may become hard to read in complex pathosystems

CC BY NC ND

From deterministic to stochastic models

- inherent stochasticity of biological systems
- manipulation of discrete, potentially small, populations
- better account of rate events (e.g. extinction)

Converting rates to probabilities ?

- assuming Markovian process
  system state at $t + 1$ only depends on state at t
- then: duration of states $\sim$ exponential distribution

Counterpart: **repeat experiments!**

Key modelling choice: **time model** (continuous vs. discrete)

CC BY NC ND

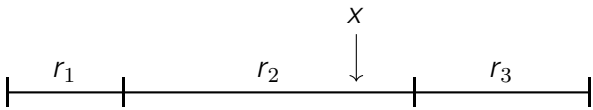
Continuous time: the Gillespie algorithm (1976)

(r_i): rates of all transitions occurring in the system

1. at time t, compute τ, **time to wait** before next event:

$$\tau \sim \text{Exponential}(\frac{1}{\sum_i r_i})$$

2. determine **which event** will occur: $x \sim \text{Uniform}(0, \sum_i r_i)$



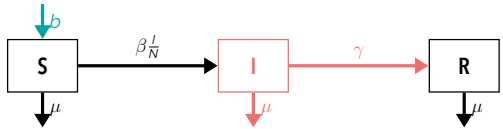
3. jump to time $t + \tau$, perform 1 transition and iterate

Accurate calculation $\rightarrow$ **complex when many interactions**

CC BY NC ND

Discrete time stochastic models

Approximation: aggregation of events occurring during **time step** δt
$\rightarrow$ easier integration of rule-based processes



Births: $\sim$ Poisson$(b\,N\,\delta t)$

Transitions: Probability calculated from rates + multinomial sampling, e.g.:

- total exit rate from I: $r_I = \gamma + \mu$
- probability to leave state I: $1 - e^{-r_I \delta t}$
- probability $I \rightarrow R$: $\frac{\gamma}{r_I}(1 - e^{-r_I \delta t})$

⚠ **KEEP δt SMALL ENOUGH!**

CC BY NC ND

From compartmental to individual-based models (IBM)

Individuals are required to model finer-grained features

- idiosyncratic differences
- history of individuals
- explicit interactions

... when such features are **necessary** in the model

Flow diagrams can be translated into IBM $\rightarrow$ **high computational cost**

- probabilities: same calculation
- Bernouilli trial (instead of multinomial sampling) $\times$ number of individuals

CC BY NC ND

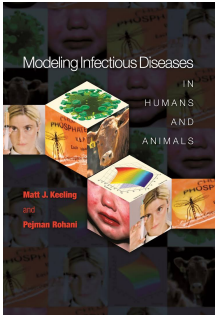
Diversity of scales

Pathogen spread and control must be considered at several scales:

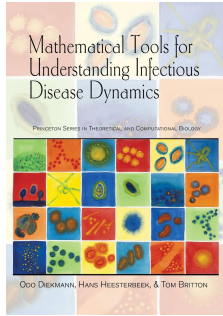
- **intra-host** ($\rightarrow$ immunology)
- **within-population**
  - interactions between individuals
  - population structure, explicit/aggregated individuals
  - control: vaccination, quarantine, selective removal...
- **between-population (metapopulations)**
  - mobility patterns, contact network
  - environmental/meteorological drivers
  - control: movement restrictions, public policies...

CC BY NC ND

To go further...



M. J. Keeling and P. Rohani
Modeling Infectious Diseases in Humans and Animals,
Princeton University Press, 2008



O. Diekmann, H. Heesterbeek, T. Britton
Mathematical Tools for Understanding Infectious Disease Dynamics
Princeton University Press, 2013

CC BY NC ND

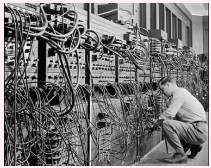
Challenges in complex epidemiological models

From model to code: three major issues

BLACK BOX

Transparency

Readability



Revisability

Impact

⚠ assumptions validation
⚠ predictions reliability
⚠ modellers' reactivity

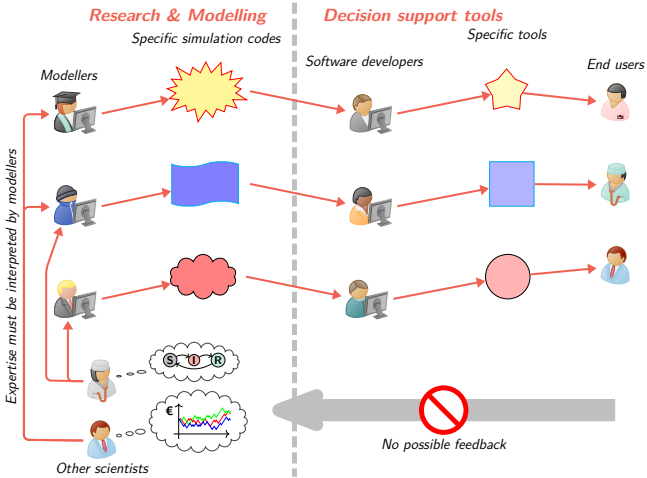
Software engineering helps!

- structured programming
- version control (e.g. `git`)
- thorough documentation
- intensive testing

... but is not enough

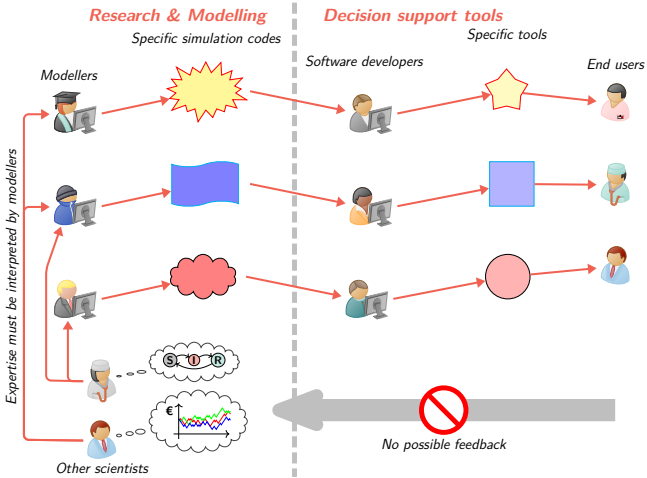
CC BY NC ND

Classical model development process



CC BY NC ND

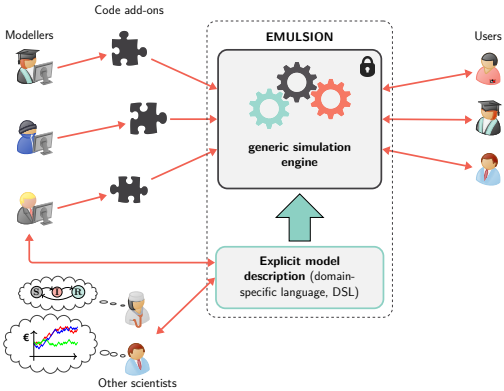
Classical model development process



- time-consuming
- lack of reliability
- no co-construction

CC BY NC ND

Artificial Intelligence methods: EMULSION framework



Generic frame for epidemiological modelling

- **Transparency**
  model assumptions/structure explicit
- **Readability**
  structured text → diagrams
- **Revisability**
  little (no) code to write

→ **faster and more reliable development**

[Picault *& al.* 2017, 2019]

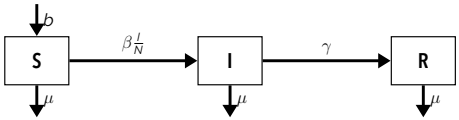
Open source diffusion: `https://sourcesup.renater.fr/www/emulsion-public`

CC BY NC ND

Processes: from flow diagrams to state machines

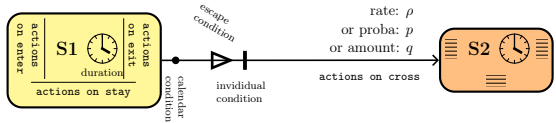
Transformation in knowledge representation

Flow diagrams



- ☺ generic representation
- ☺ paradigm-independent (compartments / IBM…)
- ☹ implicit knowledge
- ☹ domain mixing
- ☹ additions when coding

State machines



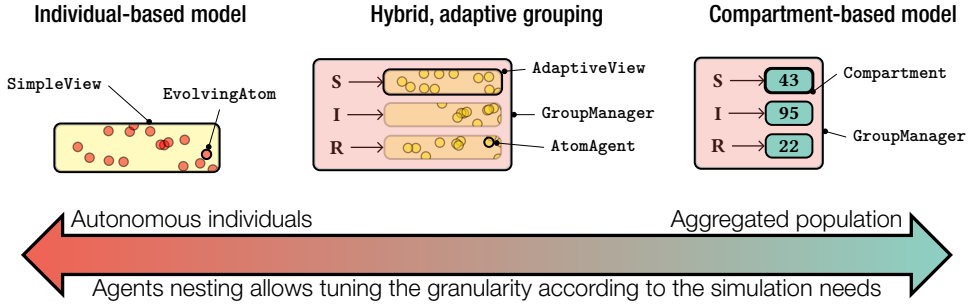
- ☺ one machine ↔ one process
- ☺ explicit **individual** durations, conditions, actions
- ☺ description ↦ univocal implementation

CC BY NC ND

Multiple paradigms and scales

Flexible multi-level agent-based simulation architecture

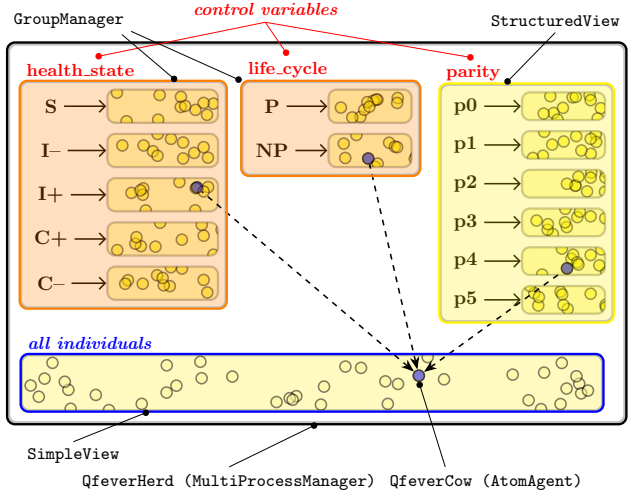
- From individuals to groups



- From single population to metapopulation

CC BY NC ND

A modular architecture for multiple processes



CC BY NC ND

DSL – basic syntactic rules: YAML (1)

EMULSION models are structured text files based on YAML syntax

- delimitation of blocks: 2-space indentation
- comments: `# this is a comment`
- values: `3.14, 'some text', yes`
- lists: `[value1, value2, value3]`

```
- value1
- value2
- value3
```

- key-value mappings: `{key1: value1, key2: value2, key3: value3}`

```
key1: value1
key2: value2
key3: value3
```


CC BY NC ND

DSL – basic syntactic rules: YAML (2)

All base elements can be combined and nested to form complex structures:

```yaml
# Here a (first-level) key mapped to a list
key1: [v1, v2, v3]
# Here a key mapped to another mapping
key2:
  subkey1:
    # the value associated with subkey1 is a list
    - item1
    - item2
  subkey2: 'an important message'
  subkey3:
    # and each item of the list below is a mapping
    - another: value1
      withother: value2
    - another: value3
      withother: value4
```


CC BY NC ND

– 2 –

EMULSION recipes

EMULSION model structure

EMULSION models are composed of several "sections" (first-level keys)

- let's dive into `step1.yaml`!
  - launch Binder configuration
  - `exercises` folder → double-click `step1.yaml`
  - read the sections

- check EMULSION documentation → **Modelling Language (basics)**

Model info
Time
Levels
Processes
Grouping
State machines
Parameters
Statevars
Input data
Initial cond.
Outputs

CC BY NC ND

EMULSION: built-in functions, variables...

Functions

- logical: `AND(x, y, ...)`, `OR(x, y, ...)`, `NOT(x)`
- conditional: `IfThenElse(cond, valThen, valElse)`
- math: `MIN(x, y, ...)`, `MAX(x, y, ...)`, `DIV(a, b))`
- wrappers for `numpy.random`: `random_bool`, `random_poisson`...
- classical math functions: `sqrt`, `exp`, `sin`... and constants: `pi`

Variables

- for each level e.g. `herd`: `total_herd`
- for each state machine e.g. `health_state`: vars of same name containing the current state (e.g. `S`, `I`...)
- for each state e.g. `S`, `I`...
  - `total_S`, `total_I`...
  - boolean vars `is_S`, `is_I`...
- for each grouping, e.g.
  `inf_by_age: [health_state, age_group]`
  → variables of the form `total_S_J` (combinations of states, in order)

CC BY NC ND

Exercise 1: EMULSION's "hello world"

- overview on model structure and syntax
- explore command-line commands and options
- modify parameters

CC BY NC ND

Exercise 2: adding a process

Demography is independent from infection → represented as a distinct process

- new state machine (`age_group`)
- new process (= when to execute the state machine)

EMULSION features

- default states
- autoremove states
- productions links
- prototypes and where to use them

CC BY NC ND

Exercise 3: play with durations

EMULSION features

- non-exponentially distributed durations in states
  constant / sampled in any distribution
- escape condition: leaving state while duration not over

CC BY NC ND

Exercise 4: contact structure

EMULSION features

- using implicit groupings based on state machines
- automatically defined variables

CC BY NC ND

Exercise 5: from compartments to IBM

EMULSION features

- syntactic transformations
- explicit groupings
- performance loss

CC BY NC ND

Exercise 6: from compartments to hybrid models

EMULSION features

- a combination between compartments and IBM
- performance improvement

CC BY NC ND

Exercise 7: more individual differences

EMULSION features

- individual variables
- actions when entering, staying, or leaving states
- variable aggregation at upper level

CC BY NC ND

Exercise 8: detection and control

EMULSION features

- explicit model for detection (more state machines !)
- actions when entering, staying, or leaving states
- variable aggregation at upper level

CC BY NC ND

Exercise 9: isolation

EMULSION features

- explicit model for the modification of the contact structure (more state machines !)
- more complex groupings
- variable aggregation at upper level

CC BY NC ND

Exercise 10: one level up!

EMULSION features

- new aggregation type (metapopulation)
- recursive initialization
- state machines working at population scale
- aggregate variables at metapopulation scale
- conditional interruption

CC BY NC ND

Exercise 11: connecting to data

- data-driven movements
- requires features not yet provided as generic components in EMULSION
- necessity to write a code add-on

EMULSION features

- data-based initial conditions
- data-based time-dependent population parameters
- link between model file and Python code add-on
- preprocessing and processes defined in the add-on
- retrieve model components (parameters, prototypes) in Python code

CC BY NC ND

Exercise 12: late revisions of initial assumptions

EMULSION features

- **modularity** of model file → limited revision impact
- examples of built-in actions e.g.
  - `become` which helps coupling state machines
  - `set_upper_var` which modifies variables at upper level

CC BY NC ND

A realistic model
(NO! It's not again about COVID!!!)

Re-implementation with EMULSION of: Massad *et al.* 2001

EMULSION features

- multiple species
- vector-borne disease
- non-trivial population dynamics
- several control methods
- connection with shell/R scripts to explore the efficacy of control methods

Guess what are the vectors ?

CC BY NC ND

First step: identify processes

To figure out how the model works:

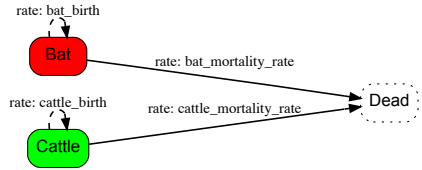
- read the flow diagram
- read the equations
- read the text

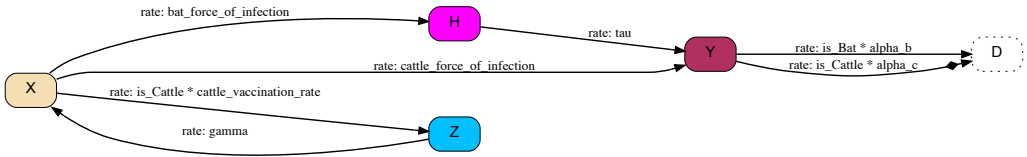
Hard to reproduce?

CC BY NC ND

Model decomposition

Two state machines: species (population dynamics) + health states (infection)





Control methods

Vaccination

Bat mortality

CC BY NC ND

Model exploration

- EMULSION is run in command-line → easy to automatize experiment plans
- EMULSION outputs are in CSV format
  → easy to handle with classical stat tools (R...) or inject into databases

A workflow with EMULSION

1. Design and test your model
2. Define relevant outputs for strategy assessment
3. Identify control parameters and build experiment plan
4. Run the scenarios (pref. with computing cluster!)
5. Assemble outputs and analyse

→ scenario exploration, sensitivity analysis, parameter inference...

CC BY NC ND

Added-value of EMULSION

- make models **readable**, **explicit**, **revisable**
- foster **modular models** based on separate processes
- facilitate changes in **paradigms** (between compartments and IBM)
- facilitate **scale** changes (back and forth)

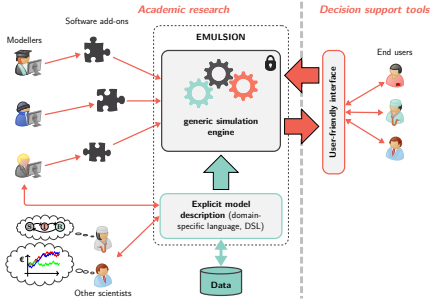
- models automatically handled by simulation engine
- yet extensible (code add-ons)
- command-line: easy deployment on calculation servers

CC BY NC ND

What's next?

- extend language and simulation engine to cover a broad range of modelling needs
- diversified diseases: ASF, BRD, Q fever, BVD, RVF, campylobacteriosis, brucellosis, PRRS...
- open-source: contributions welcome!

`https://sourcesup.renater.fr/www/emulsion-public`



Coming soon...!

Automatized production of **decision-support tools** from EMULSION models!

CC BY NC ND

That's all folks!

Thank you for participating!

Follow us: bioepar_dynamo

CC BY NC ND