Introduction aux Problèmes Inverses des Équations Différentielles.

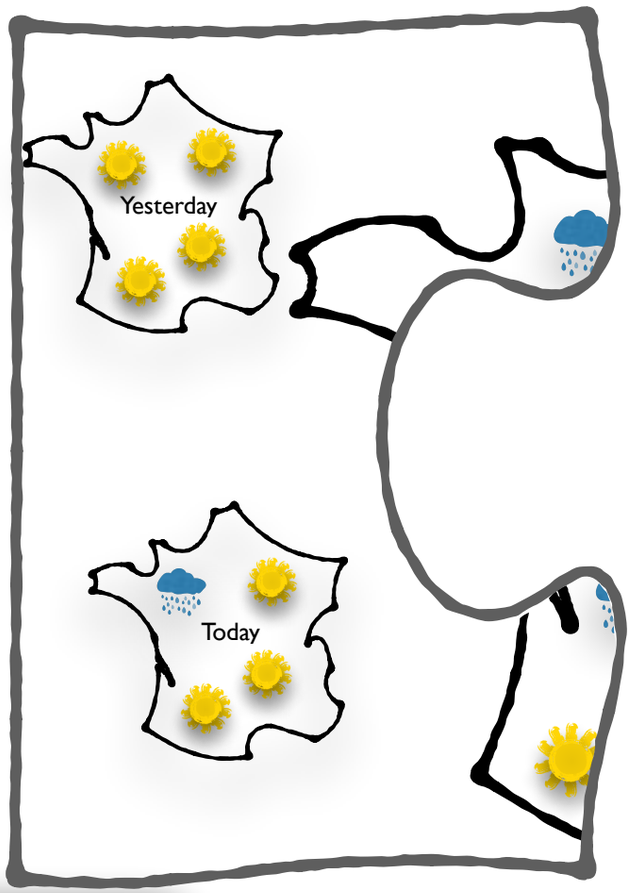
Journées Math Bio Santé 2023

Annabelle Collin

INTRODUCTION

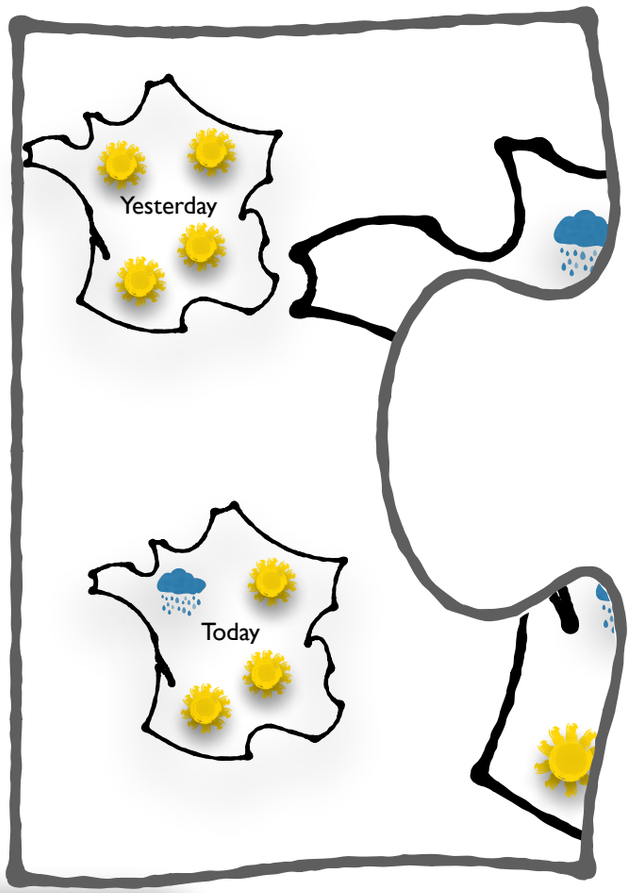
What is an inverse problem? [wiki]

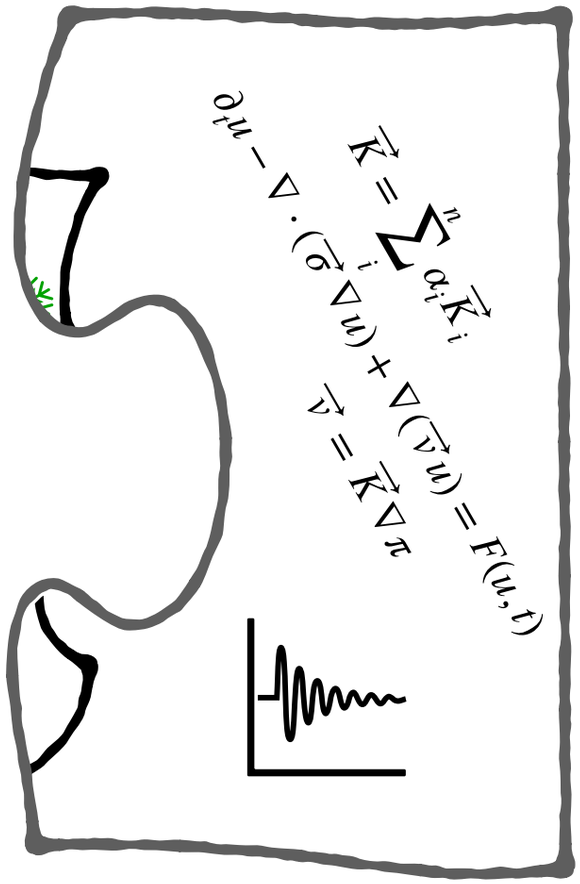
- An **inverse problem** in science is the process of calculating from a set of **observations** the causal factors that produced them.



What is an inverse problem? [wiki]

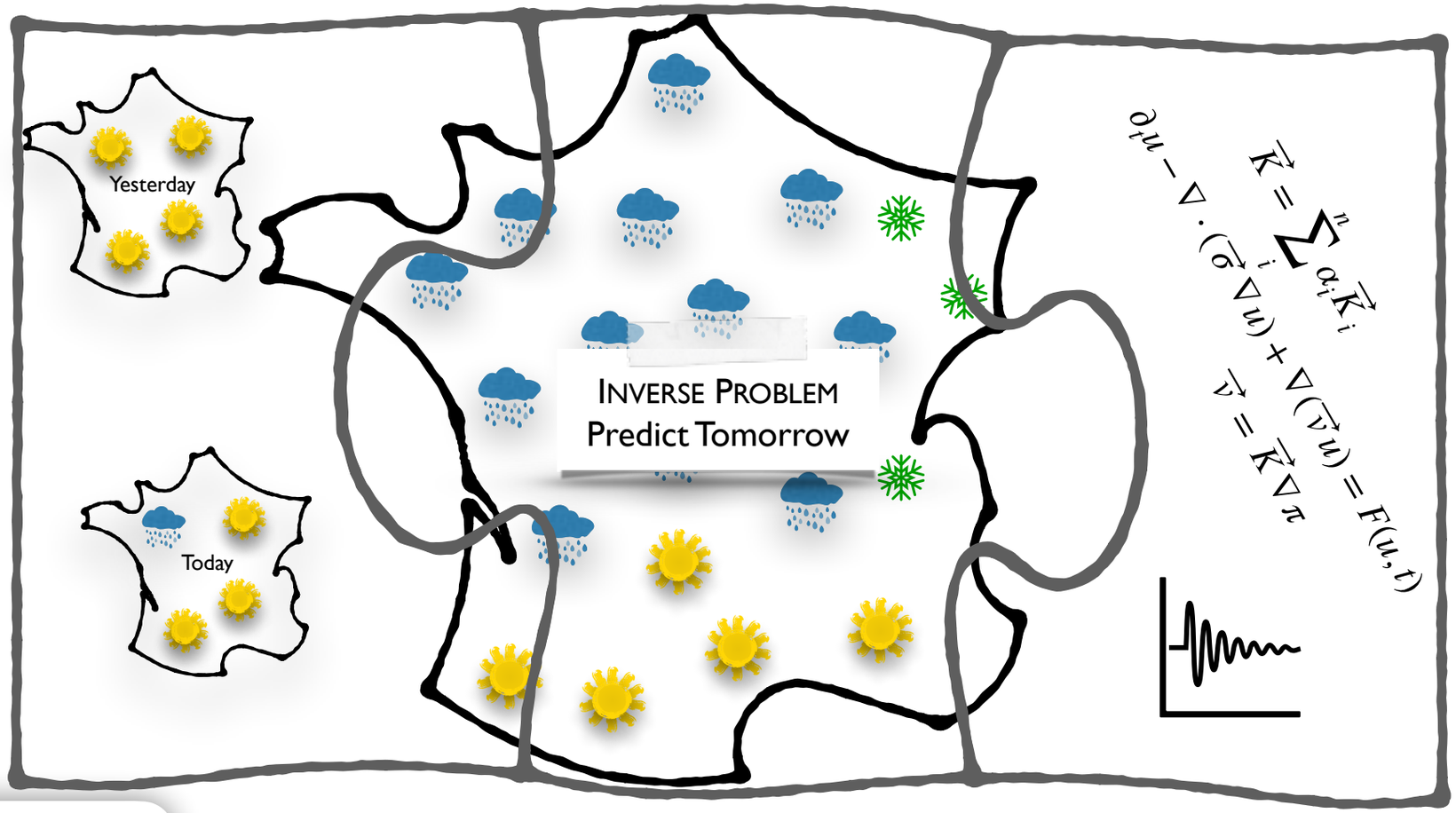
- An **inverse problem** in science is the process of calculating from a set of **observations** the causal factors that produced them**.**
- The resolution of the inverse problem passes through an initial stage of modeling of the phenomenon, known as **direct problem.**





What is an inverse problem? [wiki]

- An **inverse problem** in science is the process of calculating from a set of **observations** the causal factors that produced them.
- The resolution of the inverse problem passes through an initial stage of modeling of the phenomenon, known as **direct problem.**
- Then the approach consists in approximating as well as possible the parameters, the initial conditions, the geometry etc… which make it possible to account for these measurements.



Statement

$$\begin{cases} \dot{u}(t) = A(u, \theta, t) \\ u(0) = u_\diamond + \zeta^u \\ \theta(0) = \theta_\diamond + \zeta^\theta \end{cases} \qquad \begin{array}{l} (u_\diamond, \theta_\diamond), \textit{ a priori} \\ \\ (\zeta^u, \zeta^\theta), \textit{ unknown parts} \end{array}$$

Statement

Model

State u
(ODE or PDE unknown)
Parameters
Uncertainties on the parameters,
on the initial condition …

$$\begin{cases} \dot{u}(t) = A(u, \theta, t) & \\ u(0) = u_\diamond + \zeta^u & (u_\diamond, \theta_\diamond), \textit{ a priori} \\ \theta(0) = \theta_\diamond + \zeta^\theta & (\zeta^u, \zeta^\theta), \textit{ unknown parts} \end{cases}$$

Statement

Model
State u
(ODE or PDE unknown)
Parameters
Uncertainties on the parameters,
on the initial condition …

$$\begin{cases} \dot{u}(t) = A(u, \theta, t) \\ u(0) = u_\diamond + \zeta^u \\ \theta(0) = \theta_\diamond + \zeta^\theta \end{cases} \qquad \begin{array}{l} (u_\diamond, \theta_\diamond), \textit{ a priori} \\ \\ (\zeta^u, \zeta^\theta), \textit{ unknown parts} \end{array}$$

Observations z

Data
Partial in time/space
Noisy
Comparison with the state of the model can be difficult

Statement

Model
State u
(ODE or PDE unknown)
Parameters
Uncertainties on the parameters,
on the initial condition …

$$\begin{cases} \dot{u}(t) = A(u, \theta, t) \\ u(0) = u_\diamond + \zeta^u \\ \theta(0) = \theta_\diamond + \zeta^\theta \end{cases} \qquad \begin{array}{l} (u_\diamond, \theta_\diamond), \textit{ a priori} \\ \\ (\zeta^u, \zeta^\theta), \textit{ unknown parts} \end{array}$$

Observations z

Data
Partial in time/space
Noisy
Comparison with the state of the model can be difficult

Find ζ which minimizes the discrepancy*

$$z - C(\bar{u}_{|_\zeta})$$

(where C is the observation operator)

*Can be implicit

under the constraint of the model dynamics

$$\begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, \theta_\diamond + \zeta^\theta, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta^u \end{cases}$$

with $\zeta = (\zeta^\theta, \zeta^u)$.

Statement

Model
State u
(ODE or PDE unknown)
Parameters
Uncertainties on the parameters,
on the initial condition …

$$\begin{cases} \dot{u}(t) = A(u, \theta, t) & (u_\diamond, \theta_\diamond), \textit{ a priori} \\ u(0) = u_\diamond + \zeta^u & \\ \theta(0) = \theta_\diamond + \zeta^\theta & (\zeta^u, \zeta^\theta), \textit{ unknown parts} \end{cases}$$

Observations z

Find ζ which minimizes the discrepancy*

$$z - C(\bar{u}_{|_\zeta})$$

(where C is the observation operator)

under the constraint of the model dynamics

$$\begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, \theta_\diamond + \zeta^\theta, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta^u \end{cases}$$

with $\zeta = (\zeta^\theta, \zeta^u)$.

Data
Partial in time/space
Noisy
Comparison with the state of the model can be difficult

*Can be implicit

- Rarely explicit solution
- Inverse problems are typically ill-posed, as opposed to the well-posed problems usually met in mathematical modeling

Inverse problems are typically ill-posed. Why?

What is a **well-posed problem**? Jacques Hadamard gave the following definition of a well-posed problem in mathematical physics: Existence and Uniqueness of a solution & Continuous dependence of this solution on the data.

Inverse problems are typically ill-posed. Why?

What is a **well-posed problem**? Jacques Hadamard gave the following definition of a well-posed problem in mathematical physics: Existence and Uniqueness of a solution & Continuous dependence of this solution on the data.

Two kinds of **identifiability** issues:

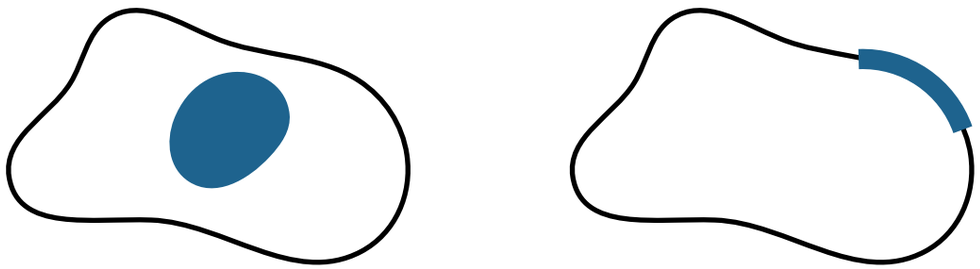
- Theoretical identifiability - The task of uniqueness is crucial in parameter (or initial condition or …) identification, since it is essential that the given data are sufficient for determining the searched for parameter (or initial condition or …) uniquely. Needs a mathematical formalism ; Leads to prove injection of function

Inverse problems are typically ill-posed. Why?

What is a **well-posed problem**? Jacques Hadamard gave the following definition of a well-posed problem in mathematical physics: Existence and Uniqueness of a solution & Continuous dependence of this solution on the data.

Two kinds of **identifiability** issues:

- Theoretical identifiability - The task of uniqueness is crucial in parameter (or initial condition or …) identification, since it is essential that the given data are sufficient for determining the searched for parameter (or initial condition or …) uniquely. Needs a mathematical formalism ; Leads to prove injection of function



Observation domain

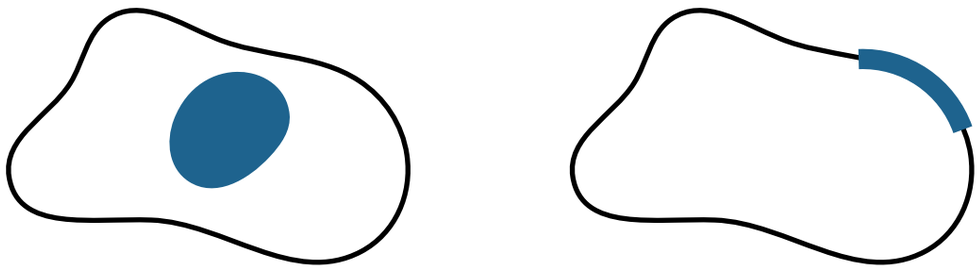
One variable?
All the variables?
More complex observation …

Inverse problems are typically ill-posed. Why?

What is a **well-posed problem**? Jacques Hadamard gave the following definition of a well-posed problem in mathematical physics: Existence and Uniqueness of a solution & Continuous dependence of this solution on the data.

Two kinds of **identifiability** issues:

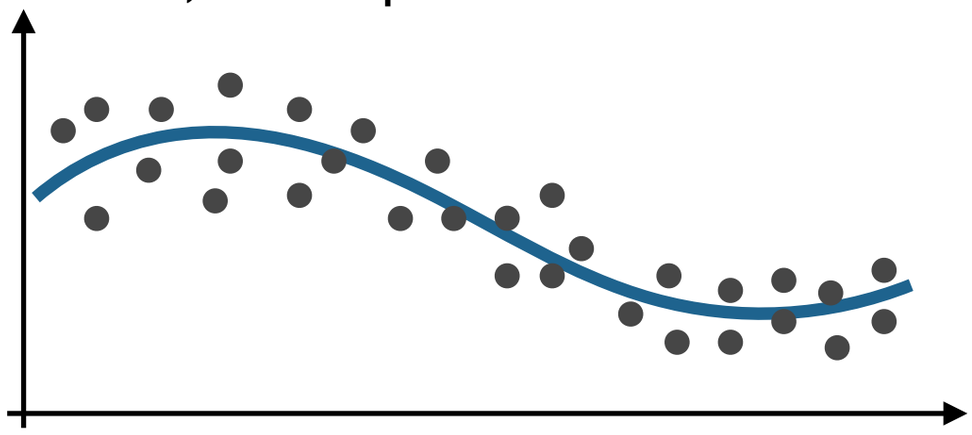
- Theoretical identifiability - The task of uniqueness is crucial in parameter (or initial condition or …) identification, since it is essential that the given data are sufficient for determining the searched for parameter (or initial condition or …) uniquely. Needs a mathematical formalism ; Leads to prove injection of function



Observation domain

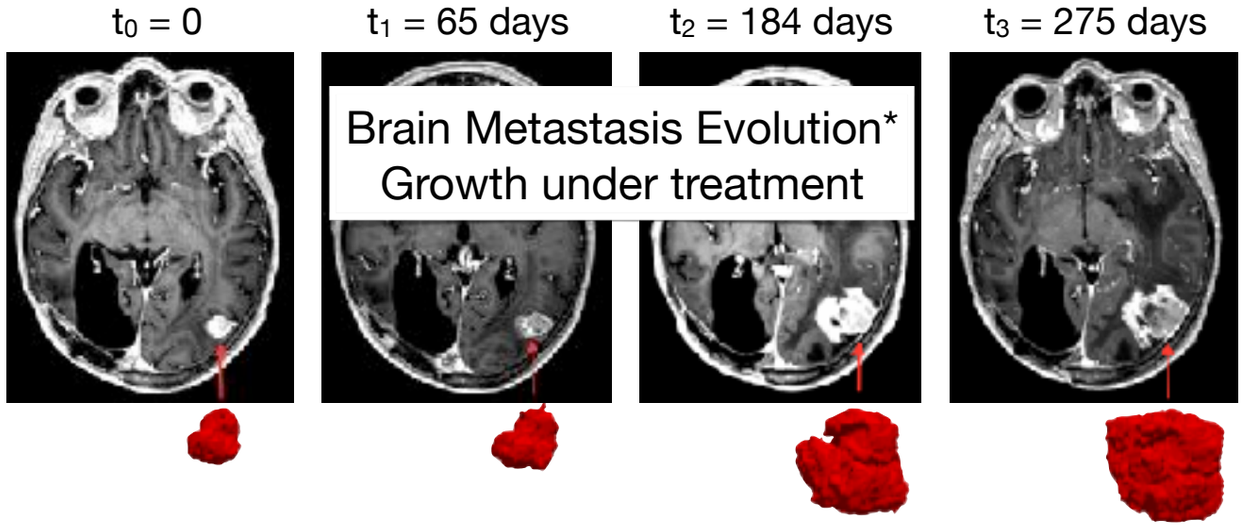
One variable?
All the variables?
More complex observation …

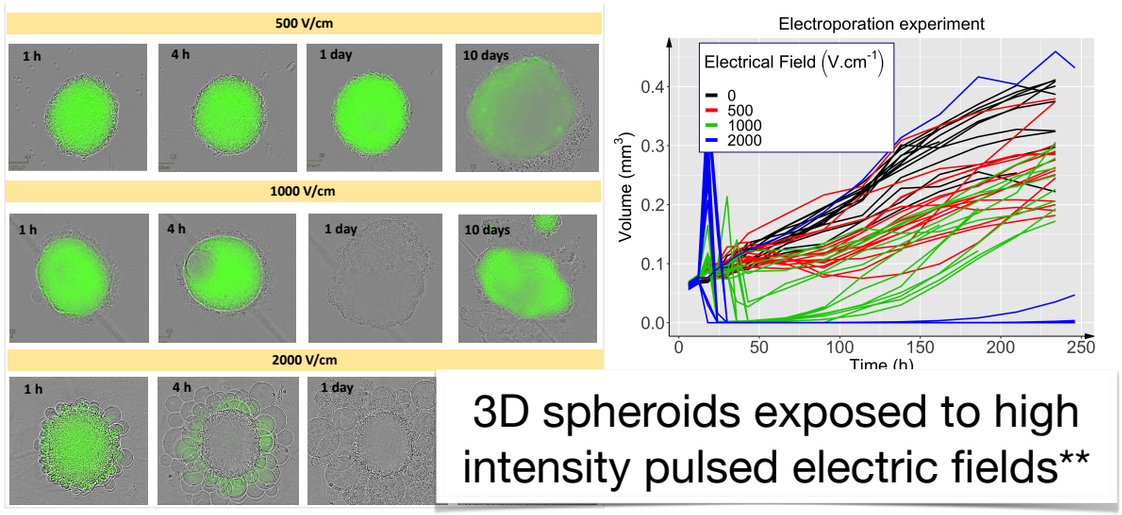
- Practical identifiability - Stability is even mostly violated in inverse problems, so especially also for parameter identification: indeed, small perturbations in the data can lead to large deviations in the solution.



Few examples: dynamical processes

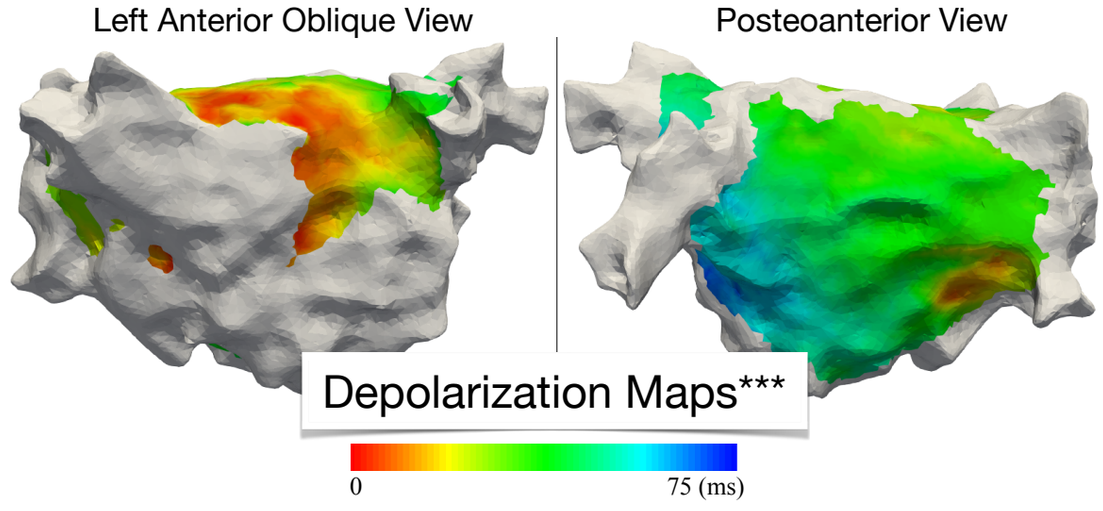
ONCOLOGY



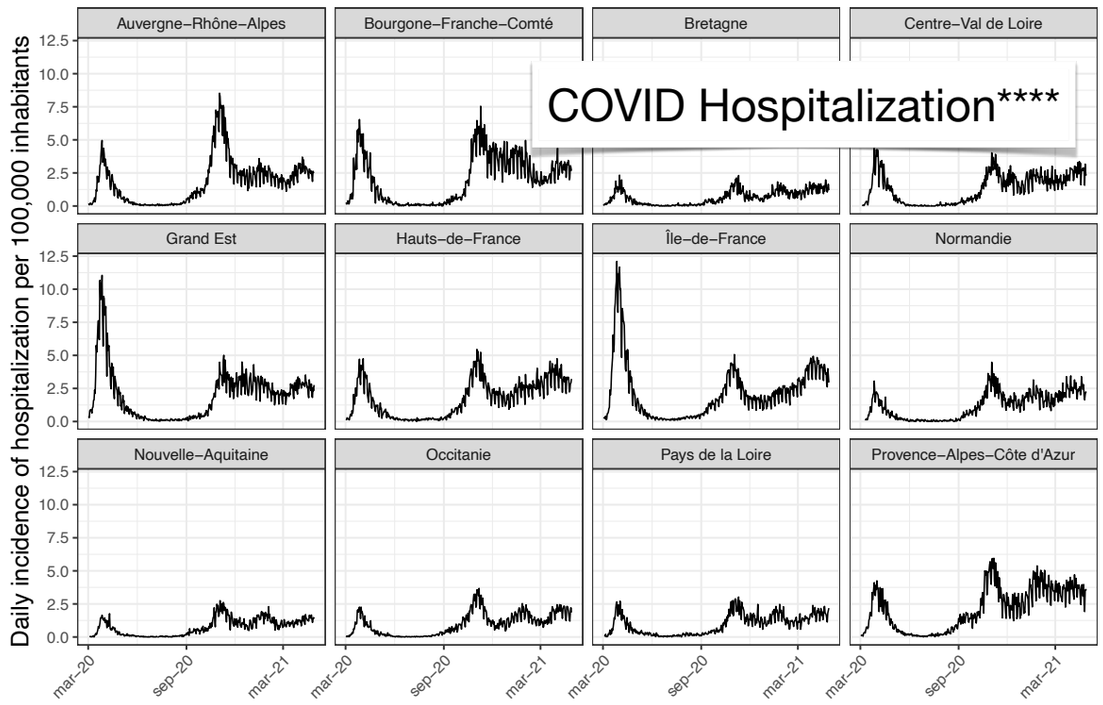


3D spheroids exposed to high intensity pulsed electric fields**

HEART



EPIDEMIOLOGY



* Brain MRI, HM Hospitales, Spain.
** Fluorescence, IPBS, Univ. Toulouse, France.
*** Carto® data, Liryc IHU, Bordeaux, France.
**** SI-VIC database https://www.data.gouv.fr/, France.

Starting points

MATHEMATICAL MODEL

Modeling

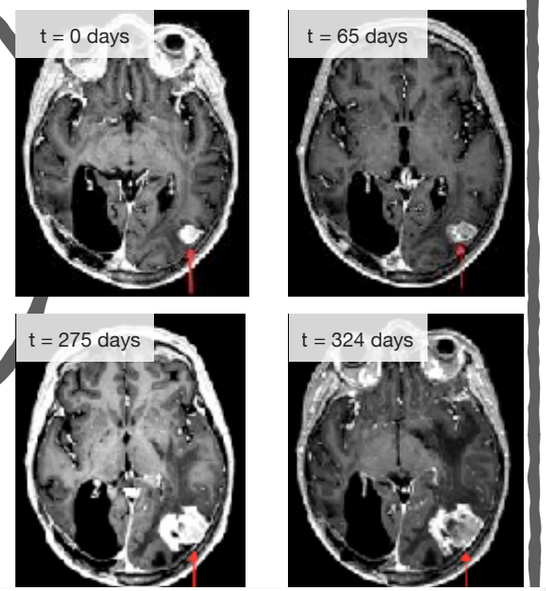
Mathematical study of the solution of the model (existence and uniqueness)

Approximation of the solution using a numerical scheme (in case of non explicit solution)

AVAILABLE OBSERVATION

Partial in time/space

Noisy



Brain Metastasis Evolution
Growth under treatment
Brain MRI, HM Hospitales, Spain.

Starting points

MATHEMATICAL MODEL

Modeling

Mathematical study of the solution of the model (existence and uniqueness)

Approximation of the solution using a numerical scheme (in case of non explicit solution)

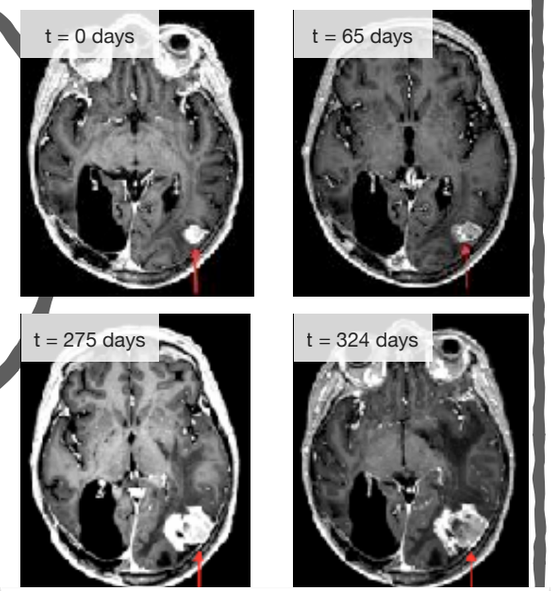
INVERSE PROBLEMS

Knowing what we want to estimate (parameters ; initial condition ; boundary condition?)

AVAILABLE OBSERVATION

Partial in time/space

Noisy



Brain Metastasis Evolution
Growth under treatment
Brain MRI, HM Hospitales, Spain.

Course Map

1. **Mathematical Formalism**

2. **Variational Strategies:** Iterative Procedures

3. **Sequential Strategies:** *Kalman* and *Luenberger* (Nudging) Filters

4. **Illustrative Examples**

Mathematical Formalism

Back to general cases

$$\begin{cases} \dot{u}(t) = A(u, \theta, t) \\ u(0) = u_\diamond + \zeta^u \\ \theta(0) = \theta_\diamond + \zeta^\theta \end{cases}$$

$(u_\diamond, \theta_\diamond)$, *a priori*

(ζ^u, ζ^θ), *unknown parts*

Back to general cases

Model

State u
(ODE or PDE unknown)
Parameters
Uncertainties on the parameters,
on the initial condition …

$$\begin{cases} \dot{u}(t) = A(u, \theta, t) & \\ u(0) = u_\diamond + \zeta^u & (u_\diamond, \theta_\diamond), \textit{ a priori} \\ \theta(0) = \theta_\diamond + \zeta^\theta & (\zeta^u, \zeta^\theta), \textit{ unknown parts} \end{cases}$$

Back to general cases

Model
State u
(ODE or PDE unknown)
Parameters
Uncertainties on the parameters,
on the initial condition …

$$\begin{cases} \dot{u}(t) = A(u, \theta, t) \\ u(0) = u_\diamond + \zeta^u \\ \theta(0) = \theta_\diamond + \zeta^\theta \end{cases} \qquad \begin{array}{l} (u_\diamond, \theta_\diamond), \textit{ a priori} \\ \\ (\zeta^u, \zeta^\theta), \textit{ unknown parts} \end{array}$$

Observations z

Data
Partial in time/space
Noisy
Comparison with the state of the model can be difficult

Back to general cases

Model
State u
(ODE or PDE unknown)
Parameters
Uncertainties on the parameters,
on the initial condition …

$$\begin{cases} \dot{u}(t) = A(u, \theta, t) \\ u(0) = u_\diamond + \zeta^u \\ \theta(0) = \theta_\diamond + \zeta^\theta \end{cases} \qquad \begin{array}{l} (u_\diamond, \theta_\diamond), \textit{ a priori} \\ \\ (\zeta^u, \zeta^\theta), \textit{ unknown parts} \end{array}$$

Observations z

Data
Partial in time/space
Noisy
Comparison with the state of the model can be difficult

Find ζ which minimizes the discrepancy*

$$z - C(\bar{u}_{|_\zeta})$$

(where C is the observation operator)

*Can be implicit

under the constraint of the model dynamics

$$\begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, \theta_\diamond + \zeta^\theta, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta^u \end{cases}$$

with $\zeta = (\zeta^\theta, \zeta^u)$.

Back to general cases

Model
State u
(ODE or PDE unknown)
Parameters
Uncertainties on the parameters,
on the initial condition …

$$\begin{cases} \dot{u}(t) = A(u, \theta, t) & (u_\diamond, \theta_\diamond), \textit{ a priori} \\ u(0) = u_\diamond + \zeta^u & \\ \theta(0) = \theta_\diamond + \zeta^\theta & (\zeta^u, \zeta^\theta), \textit{ unknown parts} \end{cases}$$

Observations z

Find ζ which minimizes the discrepancy*

$$z - C(\bar{u}_{|_\zeta})$$

(where C is the observation operator)

under the constraint of the model dynamics

$$\begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, \theta_\diamond + \zeta^\theta, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta^u \end{cases}$$

with $\zeta = (\zeta^\theta, \zeta^u)$.

Data
Partial in time/space
Noisy
Comparison with the state of the model can be difficult

*Can be implicit

- Rarely explicit solution
- Inverse problems are typically ill-posed, as opposed to the well-posed problems usually met in mathematical modeling

Maximum likelihood estimation

Find ζ which minimizes the discrepancy*

$$z - C(\bar{u}_{|_\zeta})$$

(where C is the observation operator)

under the constraint of the model dynamics

$$\begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, \theta_\diamond + \zeta^\theta, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta^u \end{cases}$$

with $\zeta = (\zeta^\theta, \zeta^u)$.

Maximum likelihood estimation

Find ζ which minimizes the discrepancy*

$$z - C(\bar{u}_{|_\zeta})$$

(where C is the observation operator)

under the constraint of the model dynamics

$$\begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, \theta_\diamond + \zeta^\theta, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta^u \end{cases}$$

with $\zeta = (\zeta^\theta, \zeta^u)$.

Equivalent to estimate: $\max_\zeta \mathbb{P}(\zeta \,|\, z)$

Maximum likelihood estimation

Find ζ which minimizes the discrepancy*

$$z - C(\bar{u}_{|_\zeta})$$

(where C is the observation operator)

under the constraint of the model dynamics

$$\begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, \theta_\diamond + \zeta^\theta, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta^u \end{cases}$$

with $\zeta = (\zeta^\theta, \zeta^u)$.

Equivalent to estimate: $\max_\zeta \mathbb{P}(\zeta \,|\, z)$

Using Bayes theorem, we obtain:

$$\max_\zeta \mathbb{P}(\zeta \,|\, z) = \max_\zeta \left[f(z \,|\, \zeta) \frac{\mathbb{P}(\zeta)}{\mathbb{P}(z)} \right] = \max_\zeta \left[f(z \,|\, \zeta)\, \mathbb{P}(\zeta) \right]$$

Maximum likelihood estimation

Find ζ which minimizes the discrepancy*

$$z - C(\bar{u}_{|_\zeta})$$

(where C is the observation operator)

under the constraint of the model dynamics

$$\begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, \theta_\diamond + \zeta^\theta, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta^u \end{cases}$$

with $\zeta = (\zeta^\theta, \zeta^u)$.

$\mathbb{P}(\zeta)$?

Finite dimension

For example if $\zeta \sim \mathcal{N}(0, P_0)$

P_0 Covariance matrix

Can be known or can be estimated

Equivalent to estimate: $\max_\zeta \mathbb{P}(\zeta \,|\, z)$

Using Bayes theorem, we obtain:

$$\max_\zeta \mathbb{P}(\zeta \,|\, z) = \max_\zeta \left[f(z \,|\, \zeta) \frac{\mathbb{P}(\zeta)}{\mathbb{P}(z)} \right] = \max_\zeta \left[f(z \,|\, \zeta)\, \mathbb{P}(\zeta) \right]$$

Maximum likelihood estimation

Find ζ which minimizes the discrepancy*

$$z - C(\bar{u}_{|_\zeta})$$

(where C is the observation operator)

under the constraint of the model dynamics

$$\begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, \theta_\diamond + \zeta^\theta, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta^u \end{cases}$$

with $\zeta = (\zeta^\theta, \zeta^u)$.

$\mathbb{P}(\zeta)$?

Finite dimension

For example if $\zeta \sim \mathcal{N}(0, P_0)$

P_0 Covariance matrix

Can be known or can be estimated

$$\mathbb{P}(\zeta) = \frac{1}{(2\pi)^{\frac{N}{2}} \det(P_0)^{\frac{1}{2}}} e^{-\zeta^T P_0^{-1} \zeta}$$

Equivalent to estimate: $\max_\zeta \mathbb{P}(\zeta \,|\, z)$

Using Bayes theorem, we obtain:

$$\max_\zeta \mathbb{P}(\zeta \,|\, z) = \max_\zeta \left[f(z \,|\, \zeta) \frac{\mathbb{P}(\zeta)}{\mathbb{P}(z)} \right] = \max_\zeta \left[f(z \,|\, \zeta)\, \mathbb{P}(\zeta) \right]$$

Maximum likelihood estimation

Find ζ which minimizes the discrepancy*

$$z - C(\bar{u}_{|_\zeta})$$

(where C is the observation operator)

under the constraint of the model dynamics

$$\begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, \theta_\diamond + \zeta^\theta, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta^u \end{cases}$$

with $\zeta = (\zeta^\theta, \zeta^u)$.

Finite dimension

$\mathbb{P}(\zeta)$? For example if $\zeta \sim \mathcal{N}(0, P_0)$

P_0 Covariance matrix

Can be known or can be estimated

$$\mathbb{P}(\zeta) = \frac{1}{(2\pi)^{\frac{N}{2}} \det(P_0)^{\frac{1}{2}}} e^{-\zeta^T P_0^{-1} \zeta}$$

$f(z|\zeta)$? For example if $z = C(\bar{u}_\zeta) + \varepsilon$

With $\varepsilon \sim \mathcal{N}(0, W)$

W Covariance matrix

Can be known or can be estimated

Equivalent to estimate: $\max_\zeta \mathbb{P}(\zeta|z)$

Using Bayes theorem, we obtain:

$$\max_\zeta \mathbb{P}(\zeta|z) = \max_\zeta \left[f(z|\zeta) \frac{\mathbb{P}(\zeta)}{\mathbb{P}(z)} \right] = \max_\zeta \left[f(z|\zeta)\, \mathbb{P}(\zeta) \right]$$

Maximum likelihood estimation

Find ζ which minimizes the discrepancy*

$$z - C(\bar{u}_{|_\zeta})$$

(where C is the observation operator)

under the constraint of the model dynamics

$$\begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, \theta_\diamond + \zeta^\theta, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta^u \end{cases}$$

with $\zeta = (\zeta^\theta, \zeta^u)$.

Finite dimension

$\mathbb{P}(\zeta)$? For example if $\zeta \sim \mathcal{N}(0, P_0)$

P_0 Covariance matrix

Can be known or can be estimated

$$\mathbb{P}(\zeta) = \frac{1}{(2\pi)^{\frac{N}{2}} \det(P_0)^{\frac{1}{2}}} e^{-\zeta^T P_0^{-1} \zeta}$$

$f(z|\zeta)$? For example if $z = C(\bar{u}_\zeta) + \varepsilon$

With $\varepsilon \sim \mathcal{N}(0, W)$

W Covariance matrix

Can be known or can be estimated

$$f(z|\zeta) = \frac{1}{(2\pi)^{\frac{N_{obs}}{2}} \det(W)^{\frac{1}{2}}} e^{-(z - C(\bar{u}_\zeta))^T W^{-1} (z - C(\bar{u}_\zeta))}$$

Equivalent to estimate: $\max_\zeta \mathbb{P}(\zeta|z)$

Using Bayes theorem, we obtain:

$$\max_\zeta \mathbb{P}(\zeta|z) = \max_\zeta \left[f(z|\zeta) \frac{\mathbb{P}(\zeta)}{\mathbb{P}(z)} \right] = \max_\zeta \left[f(z|\zeta)\, \mathbb{P}(\zeta) \right]$$

Maximum likelihood estimation

We want to maximize: $\max_{\zeta} \left[f(z \mid \zeta)\, \mathbb{P}(\zeta) \right]$

Finite dimension

$\mathbb{P}(\zeta)$? For example if $\zeta \sim \mathcal{N}(0, P_0)$

P_0 Covariance matrix

Can be known or can be estimated

$$\mathbb{P}(\zeta) = \frac{1}{(2\pi)^{\frac{N}{2}} \det(P_0)^{\frac{1}{2}}} e^{-\zeta^T P_0^{-1} \zeta}$$

$f(z \mid \zeta)$? For example if $z = C(\bar{u}_\zeta) + \varepsilon$

With $\varepsilon \sim \mathcal{N}(0, W)$

W Covariance matrix

Can be known or can be estimated

$$f(z \mid \zeta) = \frac{1}{(2\pi)^{\frac{N_{obs}}{2}} \det(W)^{\frac{1}{2}}} e^{-(z - C(\bar{u}_\zeta))^T W^{-1} (z - C(\bar{u}_\zeta))}$$

Maximum likelihood estimation

We want to maximize: $\max_{\zeta} \left[f(z \mid \zeta)\, \mathbb{P}(\zeta) \right]$

Finite dimension

Log-likelihood

$\mathbb{P}(\zeta)$? For example if $\zeta \sim \mathcal{N}(0, P_0)$

P_0 Covariance matrix

Can be known or can be estimated

$$\mathbb{P}(\zeta) = \frac{1}{(2\pi)^{\frac{N}{2}} \det(P_0)^{\frac{1}{2}}} e^{-\zeta^T P_0^{-1} \zeta}$$

$f(z \mid \zeta)$? For example if $z = C(\bar{u}_\zeta) + \varepsilon$

With $\varepsilon \sim \mathcal{N}(0, W)$

W Covariance matrix

Can be known or can be estimated

$$f(z \mid \zeta) = \frac{1}{(2\pi)^{\frac{N_{obs}}{2}} \det(W)^{\frac{1}{2}}} e^{-(z - C(\bar{u}_\zeta))^T W^{-1} (z - C(\bar{u}_\zeta))}$$

Maximum likelihood estimation

We want to maximize: $\max_{\zeta} \left[f(z \mid \zeta)\, \mathbb{P}(\zeta) \right]$

Finite dimension

$\mathbb{P}(\zeta)$? For example if $\zeta \sim \mathcal{N}(0, P_0)$

P_0 Covariance matrix

Can be known or can be estimated

$$\mathbb{P}(\zeta) = \frac{1}{(2\pi)^{\frac{N}{2}} \det(P_0)^{\frac{1}{2}}} e^{-\zeta^T P_0^{-1} \zeta}$$

$f(z \mid \zeta)$? For example if $z = C(\bar{u}_\zeta) + \varepsilon$

With $\varepsilon \sim \mathcal{N}(0, W)$

W Covariance matrix

Can be known or can be estimated

$$f(z \mid \zeta) = \frac{1}{(2\pi)^{\frac{N_{obs}}{2}} \det(W)^{\frac{1}{2}}} e^{-(z - C(\bar{u}_\zeta))^T W^{-1} (z - C(\bar{u}_\zeta))}$$

Log-likelihood

$\max_{\zeta} \left[f(z \mid \zeta)\, \mathbb{P}(\zeta) \right]$

Maximum likelihood estimation

We want to maximize: $\max_{\zeta} \left[f(z \mid \zeta)\, \mathbb{P}(\zeta) \right]$

Finite dimension

$\mathbb{P}(\zeta)$? For example if $\zeta \sim \mathcal{N}(0, P_0)$

P_0 Covariance matrix

Can be known or can be estimated

$$\mathbb{P}(\zeta) = \frac{1}{(2\pi)^{\frac{N}{2}} \det(P_0)^{\frac{1}{2}}} e^{-\zeta^T P_0^{-1} \zeta}$$

$f(z \mid \zeta)$? For example if $z = C(\bar{u}_\zeta) + \varepsilon$

With $\varepsilon \sim \mathcal{N}(0, W)$

W Covariance matrix

Can be known or can be estimated

$$f(z \mid \zeta) = \frac{1}{(2\pi)^{\frac{N_{obs}}{2}} \det(W)^{\frac{1}{2}}} e^{-(z - C(\bar{u}_\zeta))^T W^{-1} (z - C(\bar{u}_\zeta))}$$

Log-likelihood

$$\max_{\zeta} \left[f(z \mid \zeta)\, \mathbb{P}(\zeta) \right]$$

Equivalent to

$$\min_{\zeta} \left[-\log(f(z \mid \zeta)) - \log(\mathbb{P}(\zeta)) \right]$$

Maximum likelihood estimation

We want to maximize: $\max_{\zeta} \left[f(z \mid \zeta)\, \mathbb{P}(\zeta) \right]$

Finite dimension

$\mathbb{P}(\zeta)$? For example if $\zeta \sim \mathcal{N}(0, P_0)$

P_0 Covariance matrix

Can be known or can be estimated

$$\mathbb{P}(\zeta) = \frac{1}{(2\pi)^{\frac{N}{2}} \det(P_0)^{\frac{1}{2}}} e^{-\zeta^T P_0^{-1} \zeta}$$

$f(z \mid \zeta)$? For example if $z = C(\bar{u}_\zeta) + \varepsilon$

With $\varepsilon \sim \mathcal{N}(0, W)$

W Covariance matrix

Can be known or can be estimated

$$f(z \mid \zeta) = \frac{1}{(2\pi)^{\frac{N_{obs}}{2}} \det(W)^{\frac{1}{2}}} e^{-(z - C(\bar{u}_\zeta))^T W^{-1} (z - C(\bar{u}_\zeta))}$$

Log-likelihood

$$\max_{\zeta} \left[f(z \mid \zeta)\, \mathbb{P}(\zeta) \right]$$

Equivalent to

$$\min_{\zeta} \left[-\log(f(z \mid \zeta)) - \log(\mathbb{P}(\zeta)) \right]$$

We then obtain:

$$\min_{\zeta} \left[-\log(f(z \mid \zeta)) - \log(\mathbb{P}(\zeta)) \right] =$$

$$\min_{\zeta} \left[\zeta^T P_0^{-1} \zeta + (z - C(\bar{u}_\zeta))^T W^{-1} (z - C(\bar{u}_\zeta)) \right]$$

Maximum likelihood estimation

We want to maximize: $\max_{\zeta} \left[f(z \mid \zeta)\, \mathbb{P}(\zeta) \right]$

Finite dimension

$\mathbb{P}(\zeta)$? For example if $\zeta \sim \mathcal{N}(0, P_0)$

P_0 Covariance matrix

Can be known or can be estimated

$$\mathbb{P}(\zeta) = \frac{1}{(2\pi)^{\frac{N}{2}} \det(P_0)^{\frac{1}{2}}} e^{-\zeta^T P_0^{-1} \zeta}$$

$f(z \mid \zeta)$? For example if $z = C(\bar{u}_\zeta) + \varepsilon$

With $\varepsilon \sim \mathcal{N}(0, W)$

W Covariance matrix

Can be known or can be estimated

$$f(z \mid \zeta) = \frac{1}{(2\pi)^{\frac{N_{obs}}{2}} \det(W)^{\frac{1}{2}}} e^{-(z - C(\bar{u}_\zeta))^T W^{-1} (z - C(\bar{u}_\zeta))}$$

Log-likelihood

$$\max_{\zeta} \left[f(z \mid \zeta)\, \mathbb{P}(\zeta) \right]$$

Equivalent to

$$\min_{\zeta} \left[-\log(f(z \mid \zeta)) - \log(\mathbb{P}(\zeta)) \right]$$

We then obtain:

$$\min_{\zeta} \left[-\log(f(z \mid \zeta)) - \log(\mathbb{P}(\zeta)) \right] =$$

$$\min_{\zeta} \left[\zeta^T P_0^{-1} \zeta + (z - C(\bar{u}_\zeta))^T W^{-1} (z - C(\bar{u}_\zeta)) \right]$$

Least squares

Maximum likelihood estimation

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

Maximum likelihood estimation

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

Least squares

Maximum likelihood estimation

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

Tikhonov regularization

Least squares

Maximum likelihood estimation

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

Tikhonov regularization — Least squares

- How do we do this in practice? As we do every time we want to find a minimum …
  (1) Does it exist?
  (2) Is it unique?
  (3) Can we know it explicitly?
  (4) How can we approximate it? …

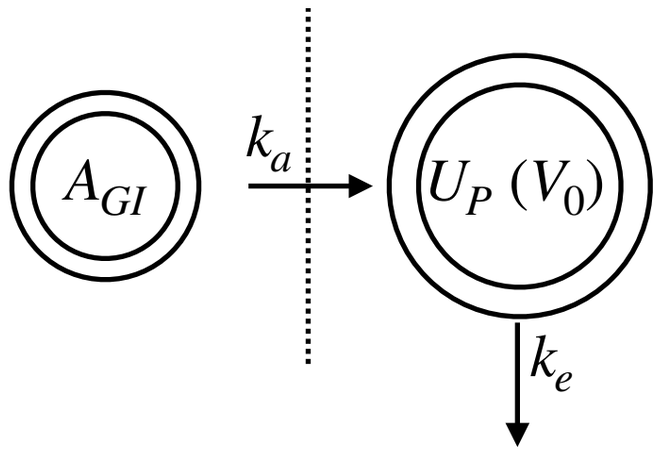
- Major particularity: the constraints of the dynamics!

Toy model

- Pharmacokinetics one-compartment model with first-order absorption and elimination

$$\begin{cases} \dot{A}_{GI} = -k_a A_{GI} \\ \dot{U}_P = \dfrac{k_a}{V_0} A_{GI} - k_e U_P \end{cases}$$

A_{GI} Dose in Gastro-intestinal compartment (mg)
U_P Concentration Plasma compartment (mg/L)
k_a Drug absorption rate (1/h)
k_e Drug elimination rate (1/h)
V_0 Volume of distribution (L)



Toy model

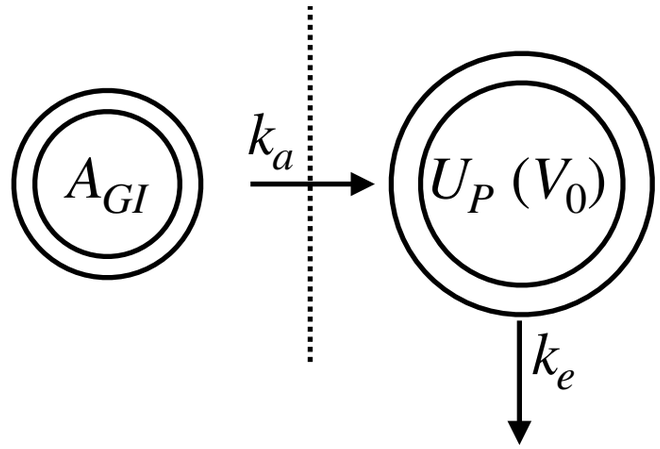
- Pharmacokinetics one-compartment model with first-order absorption and elimination

$$\begin{cases} \dot{A}_{GI} = -k_a A_{GI} \\ \dot{U}_P = \dfrac{k_a}{V_0} A_{GI} - k_e U_P \end{cases}$$

A_{GI} Dose in Gastro-intestinal compartment (mg)
U_P Concentration Plasma compartment (mg/L)
k_a Drug absorption rate (1/h)
k_e Drug elimination rate (1/h)
V_0 Volume of distribution (L)

This model is linear and can be rewritten

$$\begin{pmatrix} \dot{A}_{GI} \\ \dot{U}_P \end{pmatrix} = \begin{pmatrix} -k_a & 0 \\ \dfrac{k_a}{V_o} & -k_e \end{pmatrix} \begin{pmatrix} A_{GI} \\ U_P \end{pmatrix}. = M$$

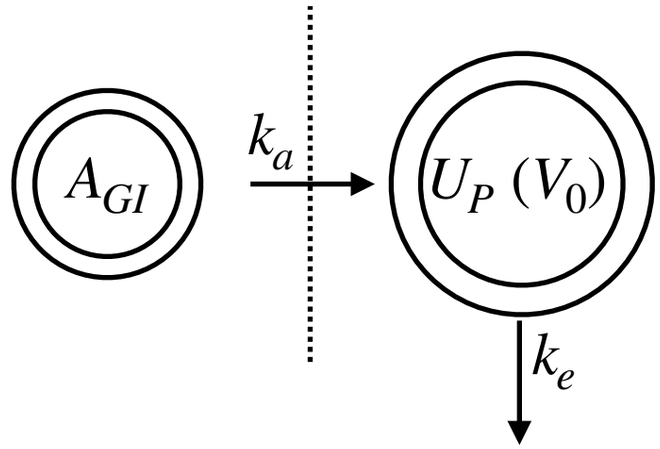


Toy model

- Pharmacokinetics one-compartment model with first-order absorption and elimination

$$\begin{cases} \dot{A}_{GI} = -k_a A_{GI} \\ \dot{U}_P = \dfrac{k_a}{V_0} A_{GI} - k_e U_P \end{cases}$$

A_{GI} Dose in Gastro-intestinal compartment (mg)
U_P Concentration Plasma compartment (mg/L)
k_a Drug absorption rate (1/h)
k_e Drug elimination rate (1/h)
V_0 Volume of distribution (L)



This model is linear and can be rewritten

$$\begin{pmatrix} \dot{A}_{GI} \\ \dot{U}_P \end{pmatrix} = \begin{pmatrix} -k_a & 0 \\ \dfrac{k_a}{V_o} & -k_e \end{pmatrix} \begin{pmatrix} A_{GI} \\ U_P \end{pmatrix}. = M$$

Then the unique solution is given by

$$\begin{pmatrix} A_{GI} \\ U_P \end{pmatrix} = e^{Mt} \begin{pmatrix} A_{GI}(0) \\ U_P(0) \end{pmatrix} = P^{-1} \begin{pmatrix} e^{-k_a t} & 0 \\ 0 & e^{-k_e t} \end{pmatrix} P \begin{pmatrix} A_{GI}(0) \\ U_P(0) \end{pmatrix},$$

where P depends on k_a, k_e and V_0 and concatenates the eigenvectors associated to the two eigenvalues -k_a and -k_e.

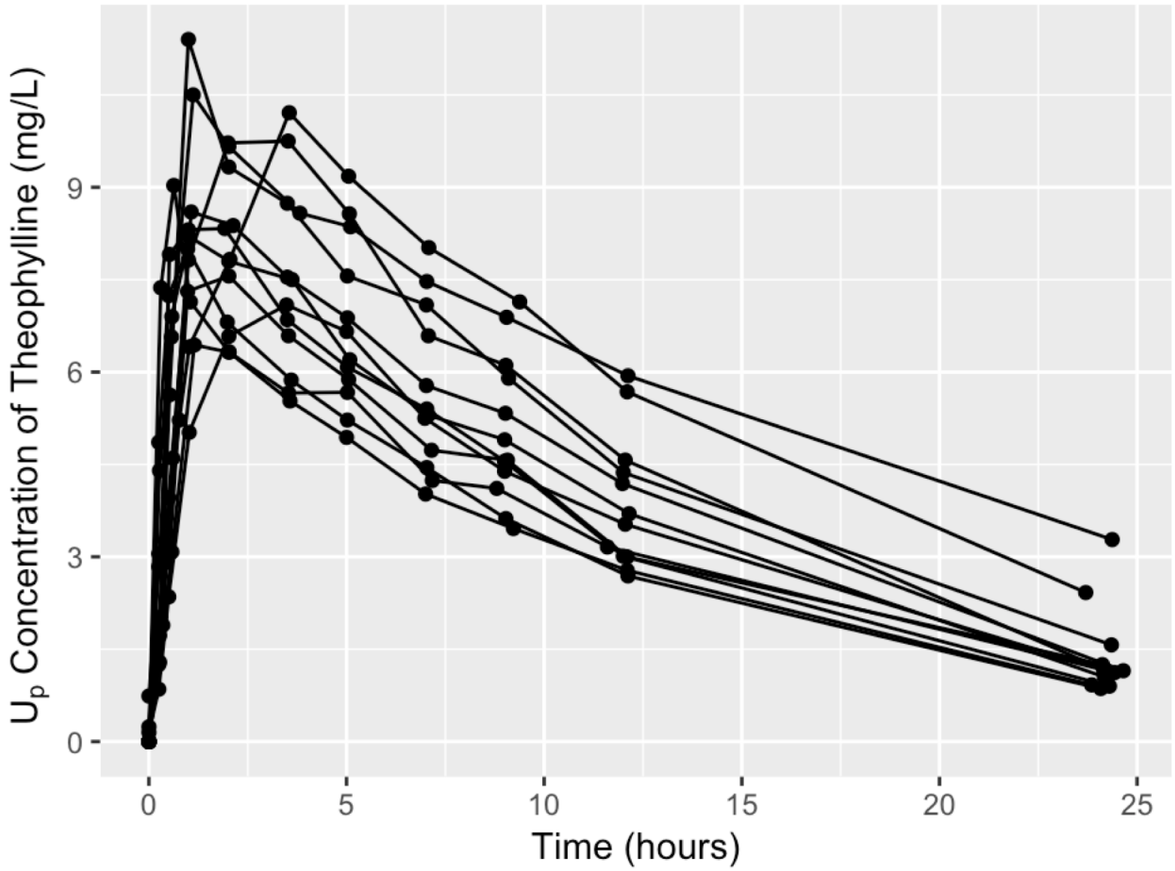
Toy model

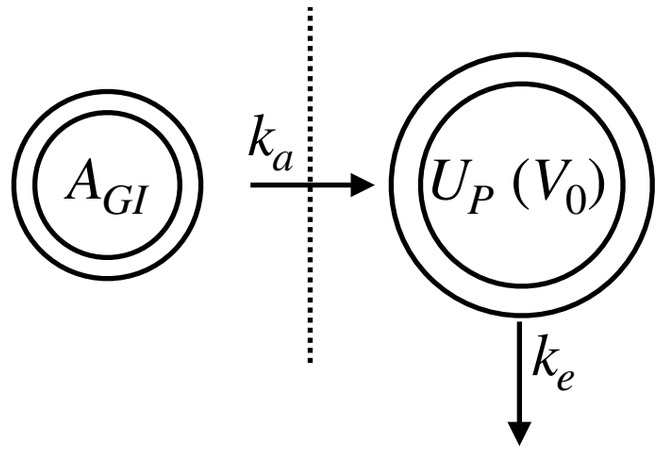
- Pharmacokinetics one-compartment model with first-order absorption and elimination

$$\begin{cases} \dot{A}_{GI} = -k_a A_{GI} \\ \dot{U}_P = \dfrac{k_a}{V_0} A_{GI} - k_e U_P \end{cases}$$

A_{GI} Dose in Gastro-intestinal compartment (mg)
U_P Concentration Plasma compartment (mg/L)
k_a Drug absorption rate (1/h)
k_e Drug elimination rate (1/h)
V_0 Volume of distribution (L)

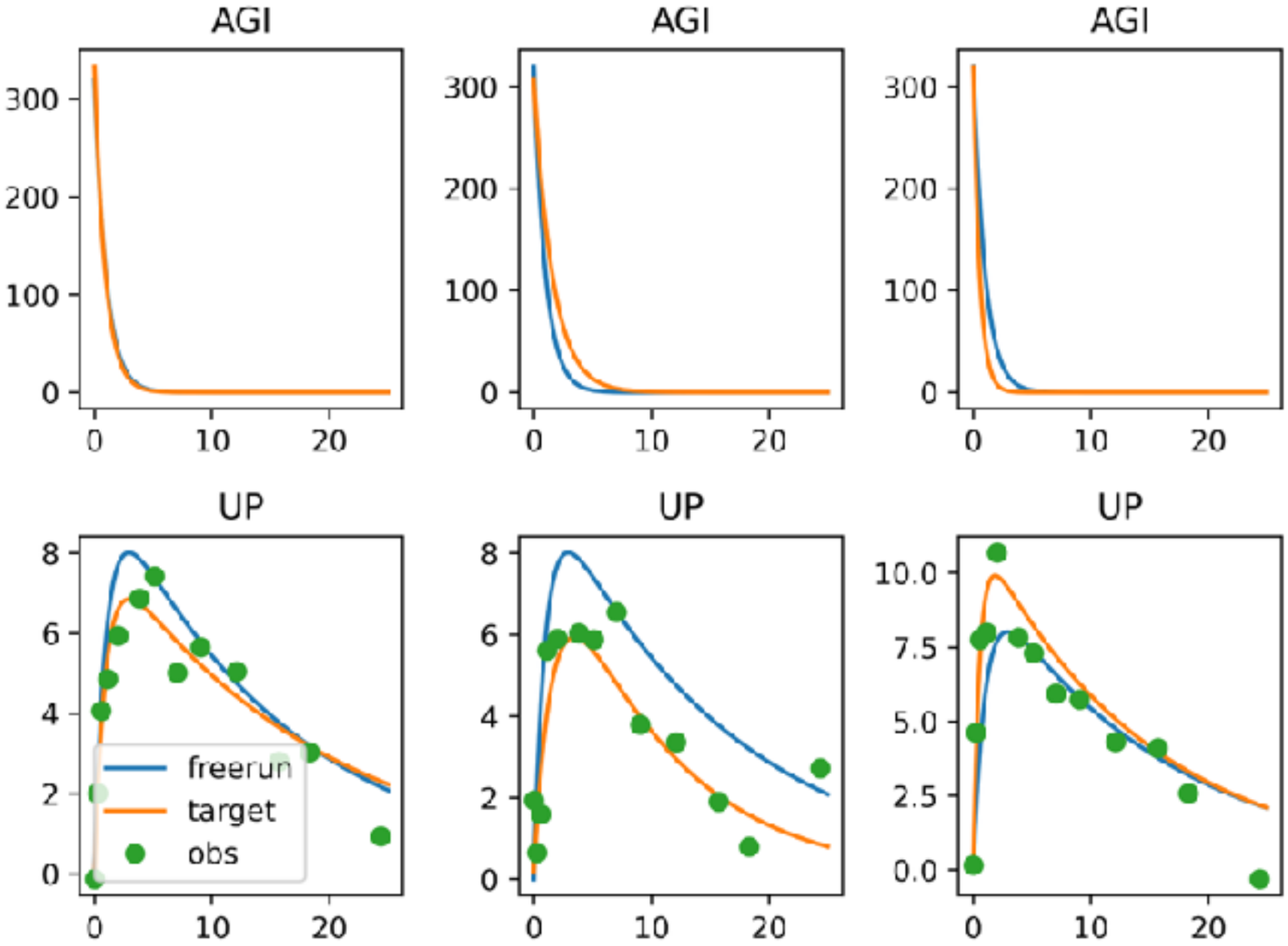
- Observations $z = U_P$





Toy model

- Freerun obtained with priors values
- Target obtained with random values following the Gaussian laws
- Observations obtained by adding noise and considering only few times



Generate different values of parameters and different initial conditions with

$$k_e = e^{\tilde{k}_e}, \ \textbf{s.t}\ \tilde{k}_e \sim \mathcal{N}(\mu_{k_e}, \sigma_{k_e})$$

$$V_0 = e^{\tilde{V}_0}, \ \textbf{s.t}\ \tilde{V}_0 \sim \mathcal{N}(\mu_{V_0}, \sigma_{V_0})$$

$$A_{GI}(0) \sim \mathcal{N}(\mu_{A_{GI}}, \sigma_{A_{GI}})$$

$$U_P(0) \sim \mathcal{N}(\mu_{U_P}, \sigma_{U_P})$$

We build noisy and time-sampled observations of synthetic patients.

Toy model

script_target_PK.py

```python
# Read parameters file
paramsfilename = "Parans/parans_PK_synthetic_data.yaml"
with open(paramsfilename) as f:
    par = yaml.load(f, Loader=yaml.FullLoader)

# Freerun model (mode) with mean of parameters and IC)
IC_freerun = np.array([float(par['prior_AGI']), float(par['prior_UP'])])
theta_freerun = np.array([float(par['prior_ka']), float(par['prior_ke']), float(par['prior_V0'])
[t_freerun,u_freerun] = forward(par,IC_freerun,theta_freerun)

# Build random variables (IC)
IC = np.random.randn(int(par['N_subjects']),2)
IC[:,0] = float(par['prior_AGI'])+float(par['std_prior_AGI'])*IC[:,0]
IC[:,1] = float(par['prior_UP'])+float(par['std_prior_UP'])*IC[:,1]
# Build random variables (parameters)
theta = np.random.randn(int(par['N_subjects']),3)
theta[:,0] = float(par['prior_ka'])+float(par['std_prior_ka'])*theta[:,0]
theta[:,1] = float(par['prior_ke'])+float(par['std_prior_ke'])*theta[:,1]
theta[:,2] = float(par['prior_V0'])+float(par['std_prior_V0'])*theta[:,2]

for i in range(0, int(par['N_subjects'])):
    # Target model
    [t,u] = forward(par,IC[i,:],theta[i,:])
    # Build noisy and subsampling data
    noise = np.random.randn(1,np.array(par['time_obs']).size)
    u_obs = np.interp(np.array(par['time_obs']), t, u[1,:])+float(par['std_noise'])*noise[0,:]
    # Plot
    plt.subplot(2, int(par['N_subjects']), i+1)
    plt.plot(t_freerun,u_freerun[0,:])
    plt.plot(t, u[0,:])
    plt.title('AGI')
    plt.subplot(2, int(par['N_subjects']), i+1+int(par['N_subjects']))
    plt.plot(t_freerun,u_freerun[1,:])
    plt.plot(t, u[1,:],label='target')
    plt.plot(np.array(par['time_obs']),u_obs,'o')
    plt.title('UP')
    if (i==0):
        plt.legend(['freerun', 'target', 'obs'])

plt.tight_layout()
plt.savefig('Results/PK_forward.png',dpi=500)
plt.show()
plt.clf()
```

functions.py

```python
def forward(par,IC,theta):
    """
    Explicit Euler
    """
    Nit = int(float(par['time_simu'])/float(par['dt']))+1
    t = np.linspace(0,float(par['time_simu']),Nit)

    # Initialization
    u = np.zeros((IC.size,Nit))

    u[:,0] = IC

    # Time loop
    for i in range(1,Nit):
        u[:,i] = u[:,i-1] + float(par['dt'])*eval(par['model'] + "(u[:,i-1],t[i-1],theta)")

    return [t,u]
```

models.py

```python
def PK_model(u,t,theta):
    """
    Model dt V = PKModel(u,t,theta)
    """
    ka = theta[0]
    ke = theta[1]
    V0 = theta[2]
    # AGI - dt * ka * AGI
    # UP + dt (ka/V0 AGI - ke UP)
    y = np.array([-np.exp(ka)*u[0],np.exp(ka)/np.exp(V0)*u[0]-np.exp(ke)*u[1]])

    return y
```

params_PK_synthetic_data.yaml

```yaml
# Time
dt: 0.005
time_simu: 25

# Parameters: ka, ke, V0
prior_ka: 0
prior_ke: -2.75
prior_V0: 3.5
std_prior_ka: 0.4
std_prior_ke: 0.4
std_prior_V0: 0.4

# Initialisation of variables
prior_AGI: 320
prior_UP: 0
std_prior_AGI: 10
std_prior_UP: 0.1

# Model
model: 'PK_model'

# OBSERVATIONS
# Data noise (UP)
std_noise: 1
# Observation time
time_obs: [0, 0.25, 0.57, 1.12, 2.02, 3.82, 5.1,
7.03, 9.05, 12.1, 15.7, 18.3, 24.37]
```

Toy model

script_target_PK.py

```python
# Read parameters file
paramsfilename = "Params/params_PK_synthetic_data.yaml"
with open(paramsfilename) as f:
    par = yaml.load(f, Loader=yaml.FullLoader)

# Freerun model (model with mean of parameters and IC)
IC_freerun = np.array([float(par['prior_AGI']), float(par['prior_UP'])])
theta_freerun = np.array([float(par['prior_ka']), float(par['prior_ke']), float(par['prior_V0'])
[t_freerun,u_freerun] = forward(par,IC_freerun,theta_freerun)

# Build random variables (IC)
IC = np.random.randn(int(par['N_subjects']),2)
IC[:,0] = float(par['prior_AGI'])+float(par['std_prior_AGI'])*IC[:,0]
IC[:,1] = float(par['prior_UP'])+float(par['std_prior_UP'])*IC[:,1]
# Build random variables (parameters)
theta = np.random.randn(int(par['N_subjects']),3)
theta[:,0] = float(par['prior_ka'])+float(par['std_prior_ka'])*theta[:,0]
theta[:,1] = float(par['prior_ke'])+float(par['std_prior_ke'])*theta[:,1]
theta[:,2] = float(par['prior_V0'])+float(par['std_prior_V0'])*theta[:,2]

for i in range(0, int(par['N_subjects'])):
    # Target model
    [t,u] = forward(par,IC[i,:],theta[i,:])
    # Build noisy and subsampling data
    noise = np.random.randn(1,np.array(par['time_obs']).size)
    u_obs = np.interp(np.array(par['time_obs']), t, u[1,:])+float(par['std_noise'])*noise[0,:]
    # Plot
    plt.subplot(2, int(par['N_subjects']), i+1)
    plt.plot(t_freerun,u_freerun[0,:])
    plt.plot(t, u[0,:])
    plt.title('AGI')
    plt.subplot(2, int(par['N_subjects']), i+1+int(par['N_subjects']))
    plt.plot(t_freerun,u_freerun[1,:])
    plt.plot(t, u[1,:],label='target')
    plt.plot(np.array(par['time_obs']),u_obs,'o')
    plt.title('UP')
    if (i==0):
        plt.legend(['freerun', 'target', 'obs']

plt.tight_layout()
plt.savefig('Results/PK_forward.png',dpi=500)
plt.show()
plt.clf()
```

functions.py

```python
def forward(par,IC,theta):
    """
    Explicit Euler
    """
    Nit = int(float(par['time_simu'])/float(par['dt']))-1
    t = np.linspace(0,float(par['time_simu']),Nit)

    # Initialization
    u = np.zeros((IC.size,Nit))

    u[:,0] = IC

    # Time loop
    for i in range(1,Nit):
        u[:,i] = u[:,i-1] + float(par['dt'])*eval(par['model'] + "(u[:,i-1],t[i-1],theta)")

    return [t,u]
```

Code dispo sur demande !

models.py

```python
def PK_model(u,t,theta):
    """
    Model dt V = PKModel(u,t,theta)
    """
    ka = theta[0]
    ke = theta[1]
    V0 = theta[2]
    # AGI - dt * ka * AGI
    # UP + dt (ka/V0 AGI - ke UP)
    y = np.array([-np.exp(ka)*u[0],np.exp(ka)/np.exp(V0)*u[0]-np.exp(ke)*u[1]])

    return y
```

params_PK_synthetic_data.yaml

```yaml
# Time
dt: 0.005
time_simu: 25

# Parameters: ka, ke, V0
prior_ka: 0
prior_ke: -2.75
prior_V0: 3.5
std_prior_ka: 0.4
std_prior_ke: 0.4
std_prior_V0: 0.4

# Initialisation of variables
prior_AGI: 320
prior_UP: 0
std_prior_AGI: 10
std_prior_UP: 0.1

# Model
model: 'PK_model'

# OBSERVATIONS
# Data noise (UP)
std_noise: 1
# Observation time
time_obs: [0, 0.25, 0.57, 1.12, 2.02, 3.82, 5.1,
7.03, 9.05, 12.1, 15.7, 18.3, 24.37]
```

Toy model

- What do we want to estimate? $\theta = [U_P^0, A_{GI}^0, k_a, k_e, V_0]$
- Are they all identifiable?
- Observations: $z = U_P$
- The exact solution writes:

$$\begin{cases} \dot{A}_{GI} = -k_a A_{GI} \\ \dot{U}_P = \dfrac{k_a}{V_0} A_{GI} - k_e U_P \end{cases}$$

$$U_P(t,\theta) = \left(U_P^0 + \frac{A_{GI}^0 k_a}{V_0(k_a - k_e)} \right) e^{-ket} - \frac{A_{GI}^0 k_a}{V_0(k_a - k_e)} e^{-kat}$$

Toy model

- What do we want to estimate? $\theta = [U_P^0, A_{GI}^0, k_a, k_e, V_0]$
- Are they all identifiable?
- Observations: $z = U_P$
- The exact solution writes:

$$\begin{cases} \dot{A}_{GI} = -k_a A_{GI} \\ \dot{U}_P = \dfrac{k_a}{V_0} A_{GI} - k_e U_P \end{cases}$$

$$U_P(t,\theta) = \left(U_P^0 + \frac{A_{GI}^0 k_a}{V_0(k_a - k_e)} \right) e^{-ket} - \frac{A_{GI}^0 k_a}{V_0(k_a - k_e)} e^{-kat}$$

- The injectivity of the function leads to

$$\psi_T : \begin{cases} \mathbb{R}^5 \to \mathscr{C}^0((0,T),\mathbb{R}) \\ \theta \mapsto (t \mapsto U_P(t,\theta)) \end{cases}$$

leads to

$$(U_P^0)^1 = (U_P^0)^2$$

$$\frac{(A_{GI}^0)^1 k_a^1}{V_0^1(k_a^1 - k_e^1)} = \frac{(A_{GI}^0)^2 k_a^2}{V_0^2(k_a^2 - k_e^2)}.$$

- And robustness to measurements errors?

Variational Strategies

Formalism
Adjoint State Method
Gradient Descent-Like Methods
Simplex Method
Stochastic Methods
Toy Model

Variational Strategies

Formalism

Adjoint State Method

Gradient Descent-Like Methods

Simplex Method

Stochastic Methods

Toy Model

Variational approach

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases} \quad \zeta = (\zeta_x, \zeta_\theta)$$

$$z = C(\bar{u}_{|_\zeta}) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0,W)$$

$$\zeta \sim \mathcal{N}(0,P_0)$$

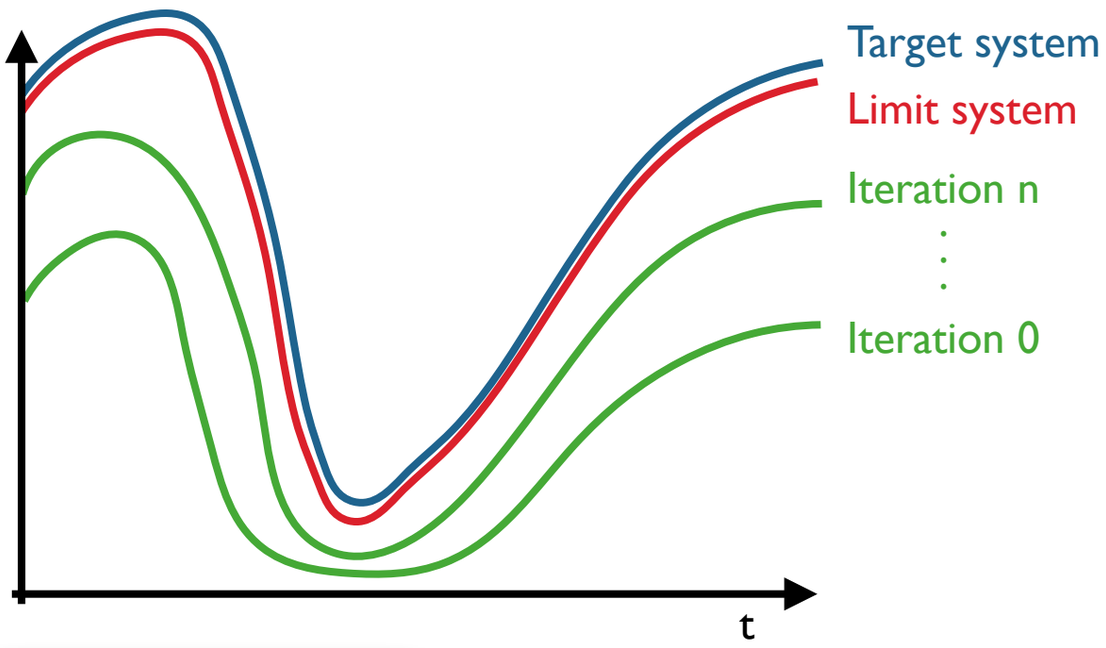
Variational approach

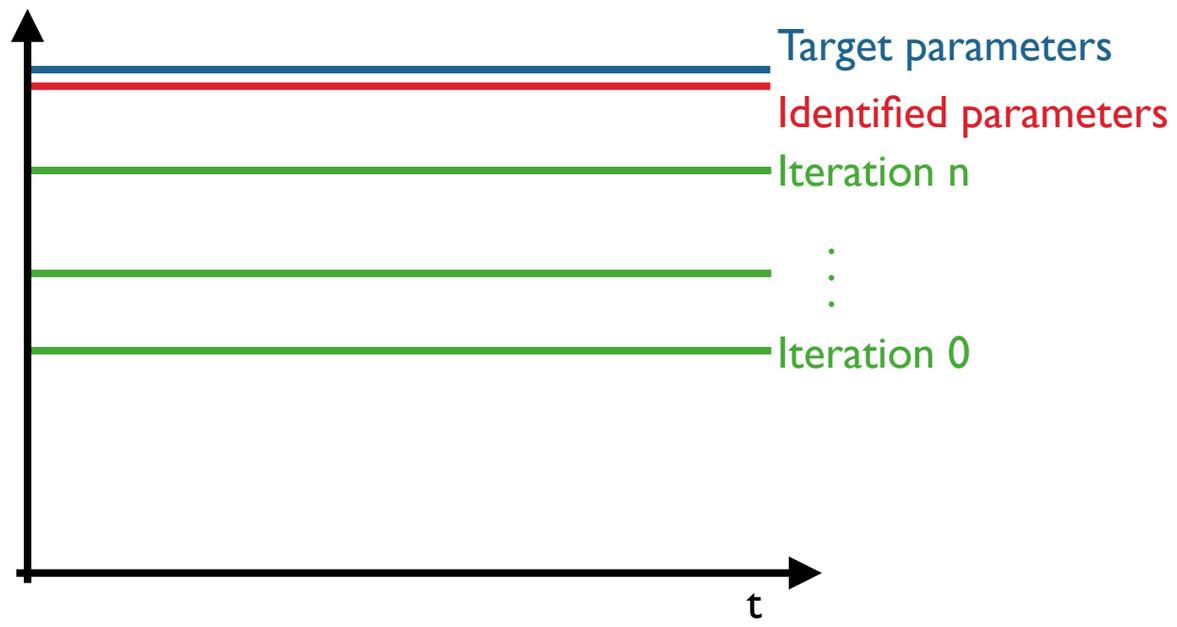
- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases} \quad \zeta = (\zeta_x, \zeta_\theta)$$

$$z = C(\bar{u}_{|_\zeta}) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0,W)$$

$$\zeta \sim \mathcal{N}(0,P_0)$$





Variational approach

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

Variational approach

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

- If there exists a unique minimizer, the condition which characterizes it is :

$$d_\zeta \mathcal{J}_n(\zeta)(d\zeta) = \frac{1}{2}\left\langle \zeta, P_0^{-1} d\zeta \right\rangle - \frac{1}{2}\sum_{k=0}^{n} \left\langle z_k - C_k \bar{u}_k, W^{-1} C_k d_\zeta \bar{u}_k(d\zeta) \right\rangle = 0.$$

Variational approach

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2} \|\zeta\|^2_{P_0^{-1}} + \frac{1}{2} \int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

- If there exists a unique minimizer, the condition which characterizes it is :

$$d_\zeta \mathcal{J}_n(\zeta)(d\zeta) = \frac{1}{2} \left\langle \zeta, P_0^{-1} d\zeta \right\rangle - \frac{1}{2} \sum_{k=0}^{n} \left\langle z_k - C_k \bar{u}_k, W^{-1} C_k d_\zeta \bar{u}_k(d\zeta) \right\rangle = 0.$$

- First strategy: compute the gradient of a functional with the Fréchet derivatives (the Jacobian matrix), which can be very **expensive** (and **difficult**) to compute.

Variational approach

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

- If there exists a unique minimizer, the condition which characterizes it is :

$$d_\zeta \mathcal{J}_n(\zeta)(d\zeta) = \frac{1}{2}\left\langle \zeta, P_0^{-1} d\zeta \right\rangle - \frac{1}{2}\sum_{k=0}^{n} \left\langle z_k - C_k \bar{u}_k, W^{-1} C_k d_\zeta \bar{u}_k(d\zeta) \right\rangle = 0.$$

- The **adjoint state method** is a numerical method for efficiently computing the gradient of a function or operator in a numerical optimization problem.

Variational Strategies

Formalism

Adjoint State Method

Gradient Descent-Like Methods

Simplex Method

Stochastic Methods

Toy Model

Adjoint state method

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

- *A* and *C* are supposed to be **linear**
- We consider a time discretization (just to simplify the presentation!)

$$\begin{cases} \bar{u}_{k+1} = A_{k+1|k}\bar{u}_k, \\ \bar{u}_0 = u_\diamond + \zeta\,. \end{cases}$$

$$d_\zeta \mathcal{J}_n(\zeta)(d\zeta) = \left\langle \zeta, P_0^{-1} d\zeta \right\rangle - \sum_{k=0}^{n} \left\langle z_k - C_k\bar{u}_k, W^{-1} C_k d_\zeta \bar{u}_k(d\zeta) \right\rangle$$

Adjoint state method

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

- A and C are supposed to be **linear**
- We consider a time discretization (just to simplify the presentation!)

$$\begin{cases} \bar{u}_{k+1} = A_{k+1|k}\bar{u}_k, \\ \bar{u}_0 = u_\diamond + \zeta\,. \end{cases}$$

$$d_\zeta \mathcal{J}_n(\zeta)(d\zeta) = \left\langle \zeta, P_0^{-1} d\zeta \right\rangle - \sum_{k=0}^{n} \left\langle z_k - C_k\bar{u}_k, W^{-1} C_k d_\zeta \bar{u}_k(d\zeta) \right\rangle$$

For a trajectory $(\bar{u}_k)_k$ we define $(q_k)_k$ the associated adjoint variable

$$\begin{cases} q_k - A^T_{k+1|k} q_{k+1} = C_k^T W^{-1}(z_k - C_k\bar{u}_k), \\ q_{n+1} = 0. \end{cases}$$

We obtain that $\tilde{\zeta} = \min_{\zeta} \mathcal{J}(\zeta) = P_0 q_0\,.$

Adjoint state method

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

- A and C are supposed to be **linear**
- We consider a time discretization (just to simplify the presentation!)

$$\begin{cases} \bar{u}_{k+1} = A_{k+1|k}\bar{u}_k, \\ \bar{u}_0 = u_\diamond + \zeta\,. \end{cases}$$

For a trajectory $(\bar{u}_k)_k$ we define $(q_k)_k$ the associated adjoint variable

$$\begin{cases} q_k - A^T_{k+1|k} q_{k+1} = C_k^T W^{-1}(z_k - C_k\bar{u}_k), \\ q_{n+1} = 0. \end{cases}$$

We obtain that $\tilde{\zeta} = \min_{\zeta} \mathcal{J}(\zeta) = P_0 q_0\,.$

$$\begin{aligned}
d_\zeta \mathcal{J}_n(\zeta)(d\zeta) &= \left\langle \zeta, P_0^{-1} d\zeta \right\rangle - \sum_{k=0}^{n} \left\langle z_k - C_k\bar{u}_k, W^{-1} C_k d_\zeta \bar{u}_k(d\zeta) \right\rangle \\
&= \left\langle \zeta, P_0^{-1} d\zeta \right\rangle - \sum_{k=0}^{n} \left\langle C_k^T W^{-1}(z_k - C_k\bar{u}_k), d_\zeta \bar{u}_k(d\zeta) \right\rangle \\
&= \left\langle \zeta, P_0^{-1} d\zeta \right\rangle + \sum_{k=0}^{n} \left\langle -(q_k - A^T_{k+1|k} q_{k+1}), d_\zeta \bar{u}_k(d\zeta) \right\rangle \\
&= \left\langle \zeta, P_0^{-1} d\zeta \right\rangle - \sum_{k=0}^{n} \left\langle q_k, d_\zeta \bar{u}_k(d\zeta) \right\rangle + \sum_{k=0}^{n} \left\langle A^T_{k+1|k} q_{k+1}, d_\zeta \bar{u}_k(d\zeta) \right\rangle \\
&= \left\langle \zeta, P_0^{-1} d\zeta \right\rangle - \sum_{k=0}^{n} \left\langle q_k, d_\zeta \bar{u}_k(d\zeta) \right\rangle + \sum_{k=0}^{n} \left\langle q_{k+1}, A_{k+1|k} d_\zeta \bar{u}_k(d\zeta) \right\rangle \\
&= \left\langle \zeta, P_0^{-1} d\zeta \right\rangle - \sum_{k=0}^{n} \left\langle q_k, d_\zeta \bar{u}_k(d\zeta) \right\rangle + \sum_{k=0}^{n} \left\langle q_{k+1}, d_\zeta \bar{u}_{k+1}(d\zeta) \right\rangle \\
&= \left\langle \zeta, P_0^{-1} d\zeta \right\rangle - \left\langle q_0, d_\zeta \bar{u}_0(d\zeta) \right\rangle + \left\langle q_{n+1}, d_\zeta \bar{u}_{n+1}(d\zeta) \right\rangle \\
&= \left\langle \zeta, P_0^{-1} d\zeta \right\rangle - \left\langle q_0, d\zeta \right\rangle
\end{aligned}$$

Chavent, G., 1974. Identification of function parameters in partial differential equations, in *Identification of parameter distributed systems*, eds Goodson, R.E. & Polis, New-York, ASME 1974.

Adjoint state method

- And in practice?

We proof that $\tilde{\zeta} = \min_{\zeta} \mathcal{J}(\zeta) = P_0 q_0$.

$$\begin{cases} \bar{u}_{k+1} = A_{k+1|k}\bar{u}_k, \\ \bar{u}_0 = u_\diamond + \zeta . \end{cases} \qquad \begin{cases} q_k - A^T_{k+1|k} q_{k+1} = C_k^T W^{-1}(z_k - C_k \bar{u}_k), \\ q_{n+1} = 0. \end{cases}$$

- Algorithm

Initialization

$$\begin{cases} \bar{u}^0_{k+1} = A_{k+1|k}\bar{u}^0_k, \\ \bar{u}^0_0 = u_\diamond . \end{cases}$$

Loop until convergence

$$\begin{cases} q^p_k - A^T_{k+1|k} q^p_{k+1} = C_k^T W^{-1}(z_k - C_k \bar{u}^p_k), \\ q_{n+1} = 0. \end{cases}$$

$$\begin{cases} \bar{u}^{p+1}_{k+1} = A_{k+1|k}\bar{u}^{p+1}_k, \\ \bar{u}^{p+1}_0 = u_\diamond + P_0 q^p_0 . \end{cases}$$

Adjoint state method

- Can be extended in non-linear case

Initialization

$$\begin{cases} \bar{u}^0_{k+1} = A_{k+1|k}(\bar{u}^0_k), \\ \bar{u}^0_0 = u_\diamond \,. \end{cases}$$

Loop until convergence

$$\begin{cases} q^p_k - (dA)^T_{k+1|k} q^p_{k+1} = (dC_k)^T W^{-1}(z_k - C_k \bar{u}^p_k), \\ q_{n+1} = 0. \end{cases}$$

$$\begin{cases} \bar{u}^{p+1}_{k+1} = A_{k+1|k}(\bar{u}^{p+1}_k), \\ \bar{u}^{p+1}_0 = u_\diamond + P_0 q^p_0 \,. \end{cases}$$

Variational Strategies

Formalism

Adjoint State Method

Gradient Descent-Like Methods

Simplex Method

Stochastic Methods

Toy Model

Gradient Descent Method

- Gradient descent is based on the observation that if the multi-variable function F is defined and differentiable in a neighborhood of a point a, then F decreases *fastest* if one goes from a in the direction of the negative gradient of F at a: $-\nabla F(a)$.

Gradient Descent Method

- Gradient descent is based on the observation that if the multi-variable function F is defined and differentiable in a neighborhood of a point a, then F decreases *fastest* if one goes from a in the direction of the negative gradient of F at a: $-\nabla F(a)$.

- Algorithm:
  Let ζ_0 a guess for a local minimum of F, ε a tolerance threshold and γ a step size
  1. Computation of $\nabla F(\zeta_k)$
  2. If $\|\nabla F(\zeta_k)\| \leq \varepsilon$: stop
  Else: $\zeta_{k+1} = \zeta_k - \gamma \nabla F(\zeta_k)$

Gradient Descent Method

- Gradient descent is based on the observation that if the multi-variable function F is defined and differentiable in a neighborhood of a point a, then F decreases *fastest* if one goes from a in the direction of the negative gradient of F at a: $-\nabla F(a)$.

- Algorithm:
  Let ζ_0 a guess for a local minimum of F, ε a tolerance threshold and γ a step size
  1. Computation of $\nabla F(\zeta_k)$
  2. If $\|\nabla F(\zeta_k)\| \leq \varepsilon$: stop
  Else: $\zeta_{k+1} = \zeta_k - \gamma \nabla F(\zeta_k)$

- Note that the value of the step size is allowed to change at every iteration.

Gradient Descent Method

- Gradient descent is based on the observation that if the multi-variable function F is defined and differentiable in a neighborhood of a point a, then F decreases *fastest* if one goes from a in the direction of the negative gradient of F at a: $-\nabla F(a)$.

- Algorithm:
  Let ζ_0 a guess for a local minimum of F, ε a tolerance threshold and γ a step size
  1. Computation of $\nabla F(\zeta_k)$
  2. If $\|\nabla F(\zeta_k)\| \leq \varepsilon$: stop
  Else: $\zeta_{k+1} = \zeta_k - \gamma \nabla F(\zeta_k)$

- Note that the value of the step size is allowed to change at every iteration.

- Convergence -
  With certain assumptions on the function F and particular choices of γ (e.g., chosen either via a line search that satisfies the Wolfe conditions, or the Barzilai–Borwein method), convergence to a local minimum can be guaranteed.
  When the function F is convex, all local minima are also global minima, so in this case gradient descent can converge to the global solution.

Gradient Descent Method

- We would like to minimize the following functional

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

Gradient Descent Method

- We would like to minimize the following functional

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

- We recall that we have

$$d_\zeta \mathcal{J}_n(\zeta)(d\zeta) = -\sum_{k=0}^{n} \left\langle z_k - C_k(\bar{u}_k), W^{-1} d_\zeta C_k(\bar{u}_k)(d\zeta) \right\rangle$$

Gradient Descent Method

- We would like to minimize the following functional

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

- We recall that we have

$$d_\zeta \mathcal{J}_n(\zeta)(d\zeta) = -\sum_{k=0}^{n} \left\langle z_k - C_k(\bar{u}_k), W^{-1} d_\zeta C_k(\bar{u}_k)(d\zeta) \right\rangle$$

- As seen before this gradient can be very difficult to compute it in practice, then we can apply the algorithm on an approximation of the gradient using for example finite difference methods.

Gauss-Newton Method

- The Gauss–Newton algorithm is used to solve non-linear least squares problems. It is a modification of Newton's method for finding a minimum of a function. Unlike Newton's method, the Gauss-Newton algorithm can only be used to minimize a sum of squared function values.
- Simplified notation

$$F(\zeta) = (z - f(\zeta))^T W^{-1} (z - f(\zeta))$$

Gauss-Newton Method

- The Gauss–Newton algorithm is used to solve non-linear least squares problems. It is a modification of Newton's method for finding a minimum of a function. Unlike Newton's method, the Gauss-Newton algorithm can only be used to minimize a sum of squared function values.
- Simplified notation

$$F(\zeta) = (z - f(\zeta))^T W^{-1} (z - f(\zeta))$$

- The function evaluated with perturbed model parameters may be locally approximated through a first-order Taylor series expansion

$$f(\zeta + d\zeta) \approx f(\zeta) + df(\zeta)(d\zeta) \approx f(\zeta) + J\, d\zeta$$

Gauss-Newton Method

- The Gauss–Newton algorithm is used to solve non-linear least squares problems. It is a modification of Newton's method for finding a minimum of a function. Unlike Newton's method, the Gauss-Newton algorithm can only be used to minimize a sum of squared function values.
- Simplified notation

$$F(\zeta) = (z - f(\zeta))^T W^{-1} (z - f(\zeta))$$

- The function evaluated with perturbed model parameters may be locally approximated through a first-order Taylor series expansion

$$f(\zeta + d\zeta) \approx f(\zeta) + df(\zeta)(d\zeta) \approx f(\zeta) + J\, d\zeta$$

- Substituting the approximation

$$F(\zeta + d\zeta) \approx z^T W^{-1} z + f(\zeta)^T W^{-1} f(\zeta) + d\zeta^T J^T W^{-1} J d\zeta - 2(z - f(\zeta))^T W^{-1} J d\zeta - 2 z^T W^{-1} f(\zeta)$$

Gauss-Newton Method

- The Gauss–Newton algorithm is used to solve non-linear least squares problems. It is a modification of Newton's method for finding a minimum of a function. Unlike Newton's method, the Gauss-Newton algorithm can only be used to minimize a sum of squared function values.
- Simplified notation

$$F(\zeta) = (z - f(\zeta))^T W^{-1} (z - f(\zeta))$$

- The function evaluated with perturbed model parameters may be locally approximated through a first-order Taylor series expansion

$$f(\zeta + d\zeta) \approx f(\zeta) + df(\zeta)(d\zeta) \approx f(\zeta) + J\, d\zeta$$

- Substituting the approximation

$$F(\zeta + d\zeta) \approx z^T W^{-1} z + f(\zeta)^T W^{-1} f(\zeta) + d\zeta^T J^T W^{-1} J d\zeta - 2(z - f(\zeta))^T W^{-1} J d\zeta - 2 z^T W^{-1} f(\zeta)$$

- The parameter update $d\zeta$ that minimizes F is found from $\partial_{d\zeta} F = 0$

$$\partial_{d\zeta} F(\zeta + d\zeta) \approx 2 d\zeta^T J^T W^{-1} J - 2(z - f(\zeta))^T W^{-1} J$$

Gauss-Newton Method

- The Gauss–Newton algorithm is used to solve non-linear least squares problems. It is a modification of Newton's method for finding a minimum of a function. Unlike Newton's method, the Gauss-Newton algorithm can only be used to minimize a sum of squared function values.
- Simplified notation

$$F(\zeta) = (z - f(\zeta))^T W^{-1} (z - f(\zeta))$$

- The function evaluated with perturbed model parameters may be locally approximated through a first-order Taylor series expansion

$$f(\zeta + d\zeta) \approx f(\zeta) + df(\zeta)(d\zeta) \approx f(\zeta) + J\, d\zeta$$

- Substituting the approximation

$$F(\zeta + d\zeta) \approx z^T W^{-1} z + f(\zeta)^T W^{-1} f(\zeta) + d\zeta^T J^T W^{-1} J d\zeta - 2(z - f(\zeta))^T W^{-1} J d\zeta - 2 z^T W^{-1} f(\zeta)$$

- The parameter update $d\zeta$ that minimizes F is found from $\partial_{d\zeta} F = 0$

$$\partial_{d\zeta} F(\zeta + d\zeta) \approx 2 d\zeta^T J^T W^{-1} J - 2(z - f(\zeta))^T W^{-1} J$$

- The resulting normal equations for the Gauss-Newton update are

$$[J W^{-1} J^T]\, d\zeta = J W^{-1} (z - f(\zeta))$$

Gauss-Newton Method

- Simplified notation

$$F(\zeta) = (z - f(\zeta))^T W^{-1} (z - f(\zeta))$$

Gauss-Newton Method

- Simplified notation

$$F(\zeta) = (z - f(\zeta))^T W^{-1} (z - f(\zeta))$$

- Algorithm:

$$\zeta_{k+1} = \zeta_k + [JW^{-1}J^T]^{-1} JW^{-1}(z - f(\zeta))$$

Gauss-Newton Method

- Simplified notation

$$F(\zeta) = (z - f(\zeta))^T W^{-1} (z - f(\zeta))$$

- Algorithm:

$$\zeta_{k+1} = \zeta_k + [JW^{-1}J^T]^{-1} JW^{-1}(z - f(\zeta))$$

- Convergence is not guaranteed, not even local convergence as in Newton's method, or convergence under the usual Wolfe conditions.

The rate of convergence of the Gauss–Newton algorithm can approach **quadratic**.

Gauss-Newton Method

- Simplified notation

$$F(\zeta) = (z - f(\zeta))^T W^{-1} (z - f(\zeta))$$

- Algorithm:

$$\zeta_{k+1} = \zeta_k + [JW^{-1}J^T]^{-1} JW^{-1}(z - f(\zeta))$$

- Convergence is not guaranteed, not even local convergence as in Newton's method, or convergence under the usual Wolfe conditions.

  The rate of convergence of the Gauss–Newton algorithm can approach **quadratic**.

- We would like to minimize the following functional

$$\min_{\zeta} \mathcal{J}(\zeta) = \min_{\zeta} \left[\int_0^T \|z - C(\bar{u}_{|_\zeta})\|^2_{W^{-1}} \, dt \right] \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|_\zeta}(t) = A(\bar{u}_{|_\zeta}, t) \\ \bar{u}_{|_\zeta}(0) = u_\diamond + \zeta \end{cases}$$

- As seen before it can be very difficult to compute J in practice, then we can apply the algorithm on an approximation of J using for example finite difference methods.

Levenberg–Marquardt algorithm

- The Levenberg-Marquardt algorithm combines two numerical minimization algorithms: the gradient descent method and the Gauss-Newton method.

Levenberg, Kenneth (1944). A Method for the Solution of Certain Non-Linear Problems in Least Squares. *Quarterly of Applied Mathematics*. 2 (2): 164–168. doi:10.1090/qam/10666.

Levenberg–Marquardt algorithm

- The Levenberg-Marquardt algorithm combines two numerical minimization algorithms: the gradient descent method and the Gauss-Newton method.

- Instead of considering (Gauss-Newton update)
  $[JW^{-1}J^T]\, d\zeta = JW^{-1}(z - f(\zeta))$,
  we consider
  $[JW^{-1}J^T + \gamma I]\, d\zeta = JW^{-1}(z - f(\zeta))$
  where small values of the damping parameter γ result in a Gauss-Newton update and large values of γ result in a gradient descent update.

Levenberg, Kenneth (1944). A Method for the Solution of Certain Non-Linear Problems in Least Squares. *Quarterly of Applied Mathematics*. 2 (2): 164–168. doi:10.1090/qam/10666.

Levenberg–Marquardt algorithm

- The Levenberg-Marquardt algorithm combines two numerical minimization algorithms: the gradient descent method and the Gauss-Newton method.

- Instead of considering (Gauss-Newton update)
  $[JW^{-1}J^T]\, d\zeta = JW^{-1}(z - f(\zeta))$,
  we consider
  $[JW^{-1}J^T + \gamma I]\, d\zeta = JW^{-1}(z - f(\zeta))$
  where small values of the damping parameter γ result in a Gauss-Newton update and large values of γ result in a gradient descent update.

- The damping parameter γ is initialized to be large so that first updates are small steps in the steepest-descent direction.
  If any iteration happens to result in a worse approximation $F(\zeta + d\zeta) \geq F(\zeta)$, then γ is increased.
  Otherwise, as the solution improves, γ is decreased, the Levenberg-Marquardt method approaches the Gauss-Newton method, and the solution typically accelerates to the local minimum.

Levenberg, Kenneth (1944). A Method for the Solution of Certain Non-Linear Problems in Least Squares. *Quarterly of Applied Mathematics*. 2 (2): 164–168. doi:10.1090/qam/10666.

Levenberg–Marquardt algorithm

- The Levenberg-Marquardt algorithm combines two numerical minimization algorithms: the gradient descent method and the Gauss-Newton method.

Levenberg, Kenneth (1944). A Method for the Solution of Certain Non-Linear Problems in Least Squares. *Quarterly of Applied Mathematics*. 2 (2): 164–168. doi:10.1090/qam/10666.

Levenberg–Marquardt algorithm

- The Levenberg-Marquardt algorithm combines two numerical minimization algorithms: the gradient descent method and the Gauss-Newton method.

- Instead of considering (Gauss-Newton update)
  $[JW^{-1}J^T]\, d\zeta = JW^{-1}(z - f(\zeta))$,
  we consider
  $[JW^{-1}J^T + \gamma I]\, d\zeta = JW^{-1}(z - f(\zeta))$
  where small values of the damping parameter γ result in a Gauss-Newton update and large values of γ result in a gradient descent update.

Levenberg, Kenneth (1944). A Method for the Solution of Certain Non-Linear Problems in Least Squares. *Quarterly of Applied Mathematics*. 2 (2): 164–168. doi:10.1090/qam/10666.

Levenberg–Marquardt algorithm

- The Levenberg-Marquardt algorithm combines two numerical minimization algorithms: the gradient descent method and the Gauss-Newton method.

- Instead of considering (Gauss-Newton update)
  $[JW^{-1}J^T]\, d\zeta = JW^{-1}(z - f(\zeta))$,
  we consider
  $[JW^{-1}J^T + \gamma I]\, d\zeta = JW^{-1}(z - f(\zeta))$
  where small values of the damping parameter γ result in a Gauss-Newton update and large values of γ result in a gradient descent update.

- The damping parameter γ is initialized to be large so that first updates are small steps in the steepest-descent direction.
  If any iteration happens to result in a worse approximation $F(\zeta + d\zeta) \geq F(\zeta)$, then γ is increased.
  Otherwise, as the solution improves, γ is decreased, the Levenberg-Marquardt method approaches the Gauss-Newton method, and the solution typically accelerates to the local minimum.

Levenberg, Kenneth (1944). A Method for the Solution of Certain Non-Linear Problems in Least Squares. *Quarterly of Applied Mathematics*. 2 (2): 164–168. doi:10.1090/qam/10666.

Trust-Region methods

- Trust-region methods define a region around the current iterate within which they trust the model to be an adequate representation of the objective function, and then choose the step to be the approximate minimizer of the model in this region.

Trust-Region methods

- Trust-region methods define a region around the current iterate within which they trust the model to be an adequate representation of the objective function, and then choose the step to be the approximate minimizer of the model in this region.

Line-search method

Algorithm

1. Compute a descent direction p_k of F in ζ_k
2. Choose γ_k to 'loosely' minimize $F(\zeta_k + \alpha_k p_k)$
3. Update $\zeta_{k+1} = \zeta_k + \alpha_k p_k$
4. Until $\|\nabla F(\zeta_k)\| \leq \varepsilon$

Easy to implement

Weak convergence results

Trust-Region methods

- Trust-region methods define a region around the current iterate within which they trust the model to be an adequate representation of the objective function, and then choose the step to be the approximate minimizer of the model in this region.

Line-search method

Algorithm

1. Compute a descent direction p_k of F in ζ_k
2. Choose γ_k to 'loosely' minimize $F(\zeta_k + \alpha_k p_k)$
3. Update $\zeta_{k+1} = \zeta_k + \alpha_k p_k$
4. Until $\|\nabla F(\zeta_k)\| \leq \varepsilon$

Easy to implement
Weak convergence results

Trust region method

ANNABELLE COLLIN

Trust-Region methods

- Trust-region methods define a region around the current iterate within which they trust the model to be an adequate representation of the objective function, and then choose the step to be the approximate minimizer of the model in this region.

$$g_k = \nabla f(\zeta_k)$$
$$M_k = \nabla^2 f(\zeta_k)$$

Line-search method

Algorithm

1. Compute a descent direction p_k of F in ζ_k
2. Choose γ_k to 'loosely' minimize $F(\zeta_k + \alpha_k p_k)$
3. Update $\zeta_{k+1} = \zeta_k + \alpha_k p_k$
4. Until $\|\nabla F(\zeta_k)\| \leq \varepsilon$

Easy to implement
Weak convergence results

Trust region method

Algorithm

1. Define a model Ψ_k of F in ζ_k: $\Psi_k(s) = \nabla g_k^T s + \frac{1}{2} s^T M_k s$ acceptable in a neighboring $\{\zeta \in \mathbb{R}^N, \|\zeta\| \leq \Delta_k\}$ leading to (Problem 1) $\min_{\|s\| \leq \Delta_k} \Psi_k(s)$.

Trust-Region methods

- Trust-region methods define a region around the current iterate within which they trust the model to be an adequate representation of the objective function, and then choose the step to be the approximate minimizer of the model in this region.

$g_k = \nabla f(\zeta_k)$
$M_k = \nabla^2 f(\zeta_k)$

Line-search method

Algorithm

1. Compute a descent direction p_k of F in ζ_k
2. Choose γ_k to 'loosely' minimize $F(\zeta_k + \alpha_k p_k)$
3. Update $\zeta_{k+1} = \zeta_k + \alpha_k p_k$
4. Until $\|\nabla F(\zeta_k)\| \leq \varepsilon$

Easy to implement
Weak convergence results

Trust region method

Algorithm

1. Define a model Ψ_k of F in ζ_k: $\Psi_k(s) = \nabla g_k^T s + \frac{1}{2} s^T M_k s$ acceptable in a neighboring $\{\zeta \in \mathbb{R}^N, \|\zeta\| \leq \Delta_k\}$ leading to (Problem 1) $\min_{\|s\| \leq \Delta_k} \Psi_k(s)$.
2. a. Solve (Problem 1): s_k and compute $\rho_k = -\frac{F(\zeta_k) - F(\zeta_k + s_k)}{\Psi_k(s_k)}$

   b. If ρ_k is *too small* ($\rho_k \leq \omega_1$), back to 1.a. after reducing the value of $\Delta_k \in]\tau_1\|s_k\|, \tau_2\|s_k\|[$
3. Update $\zeta_{k+1} = \zeta_k + s_k$ and $\Delta_{k+1} \in \begin{cases} [\tau_2 \Delta_k, \Delta_k], \text{ if } \rho_k \leq \omega_2 \\ [\Delta_k, \tau_3 \Delta_k], \text{ else.} \end{cases}$
4. Until $\|\nabla F(\zeta_k)\| \leq \varepsilon$

Trust-Region methods

- Trust-region methods define a region around the current iterate within which they trust the model to be an adequate representation of the objective function, and then choose the step to be the approximate minimizer of the model in this region.

$g_k = \nabla f(\zeta_k)$
$M_k = \nabla^2 f(\zeta_k)$

Line-search method

Algorithm

1. Compute a descent direction p_k of F in ζ_k
2. Choose γ_k to 'loosely' minimize $F(\zeta_k + \alpha_k p_k)$
3. Update $\zeta_{k+1} = \zeta_k + \alpha_k p_k$
4. Until $\|\nabla F(\zeta_k)\| \leq \varepsilon$

Easy to implement
Weak convergence results

Trust region method

Algorithm

1. Define a model Ψ_k of F in ζ_k: $\Psi_k(s) = \nabla g_k^T s + \frac{1}{2} s^T M_k s$ acceptable in a neighboring $\{\zeta \in \mathbb{R}^N, \|\zeta\| \leq \Delta_k\}$ leading to (Problem 1) $\min_{\|s\| \leq \Delta_k} \Psi_k(s)$.
2. a. Solve (Problem 1): s_k and compute $\rho_k = -\frac{F(\zeta_k) - F(\zeta_k + s_k)}{\Psi_k(s_k)}$

   b. If ρ_k is *too small* ($\rho_k \leq \omega_1$), back to 1.a. after reducing the value of $\Delta_k \in]\tau_1\|s_k\|, \tau_2\|s_k\|[$
3. Update $\zeta_{k+1} = \zeta_k + s_k$ and $\Delta_{k+1} \in \begin{cases} [\tau_2 \Delta_k, \Delta_k], \text{ if } \rho_k \leq \omega_2 \\ [\Delta_k, \tau_3 \Delta_k], \text{ else.} \end{cases}$
4. Until $\|\nabla F(\zeta_k)\| \leq \varepsilon$

Difficult to implement
Strong convergence results

Trust Region Reflective method

- Objective: minimization subject to upper and/or lower **bounds**

On the convergence of interior-reflective Newton methods for nonlinear minimization subject to bounds. Thomas F. Coleman and Yuying Li. Mathematical Programming 67 (1994) 189-224.

Trust Region Reflective method

- Objective: minimization subject to upper and/or lower **bounds**
- Idea: Use a local transformation. Not so easy if we want a quadratic method.

On the convergence of interior-reflective Newton methods for nonlinear minimization subject to bounds. Thomas F. Coleman and Yuying Li. Mathematical Programming 67 (1994) 189-224.

Trust Region Reflective method

- Objective: minimization subject to upper and/or lower **bounds**
- Idea: Use a local transformation. Not so easy if we want a quadratic method.
- Modification of the algorithm:

$$g_k = \nabla f(\zeta_k)$$
$$M_k = \nabla^2 f(\zeta_k) + C_k$$
$$C_k = D_k^{-1}\mathrm{diag}(g_k)\nabla J_k^v D_k^{-1}$$

Algorithm

1. Define a model Ψ_k of F in ζ_k: $\Psi_k(s) = \nabla g_k^T s + \frac{1}{2} s^T M_k s$

   acceptable in a neighboring $\{\zeta \in \mathbb{R}^N, \|\zeta\| \leq \Delta_k\}$ leading to

   (Problem 1) $\min_{\|s\| \leq \Delta_k} \Psi_k(s)$.

2. a. Solve (Problem 1): s_k and compute $\rho_k = -\frac{F(\zeta_k) - F(\zeta_k + s_k)}{\Psi_k(s_k)}$

   b. If ρ_k is *too small* ($\rho_k \leq \omega_1$), back to 1.a. after reducing the value of $\Delta_k \in]\tau_1\|s_k\|, \tau_2\|s_k\|[$

3. Update $\zeta_{k+1} = \zeta_k + s_k$ and $\Delta_{k+1} \in \begin{cases} [\tau_2\Delta_k, \Delta_k], \text{ if } \rho_k \leq \omega_2 \\ [\Delta_k, \tau_3\Delta_k], \text{ else.} \end{cases}$

4. Until $\|\nabla F(\zeta_k)\| \leq \varepsilon$

On the convergence of interior-reflective Newton methods for nonlinear minimization subject to bounds. Thomas F. Coleman and Yuying Li. Mathematical Programming 67 (1994) 189-224.

Variational Strategies

Formalism

Adjoint State Method

Gradient Descent-Like Methods

Simplex Method

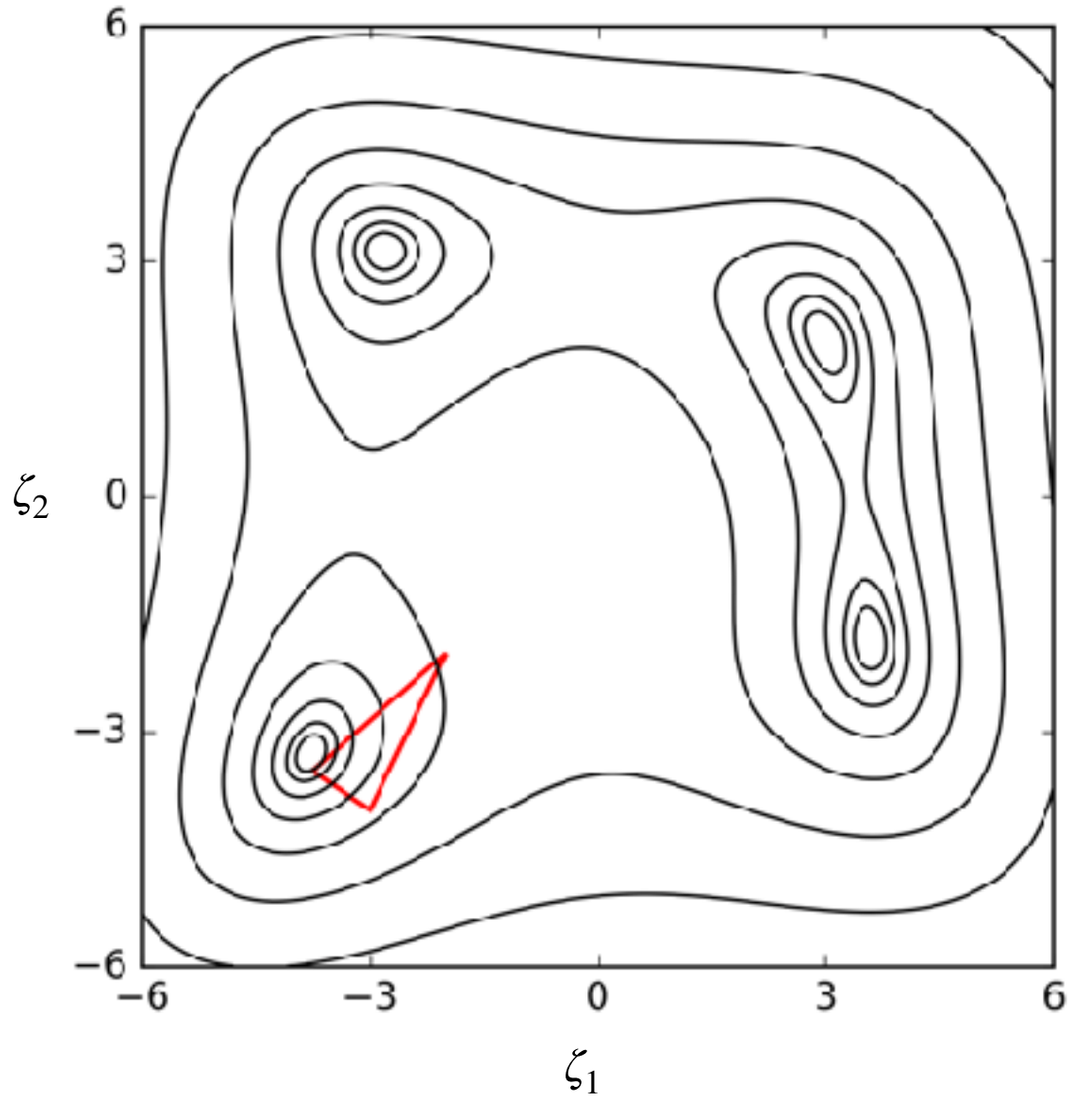
Stochastic Methods

Toy Model

Nelder–Mead method

- It is a direct search method (based on function comparison) and is often applied to nonlinear optimization problems for which derivatives may not be known.

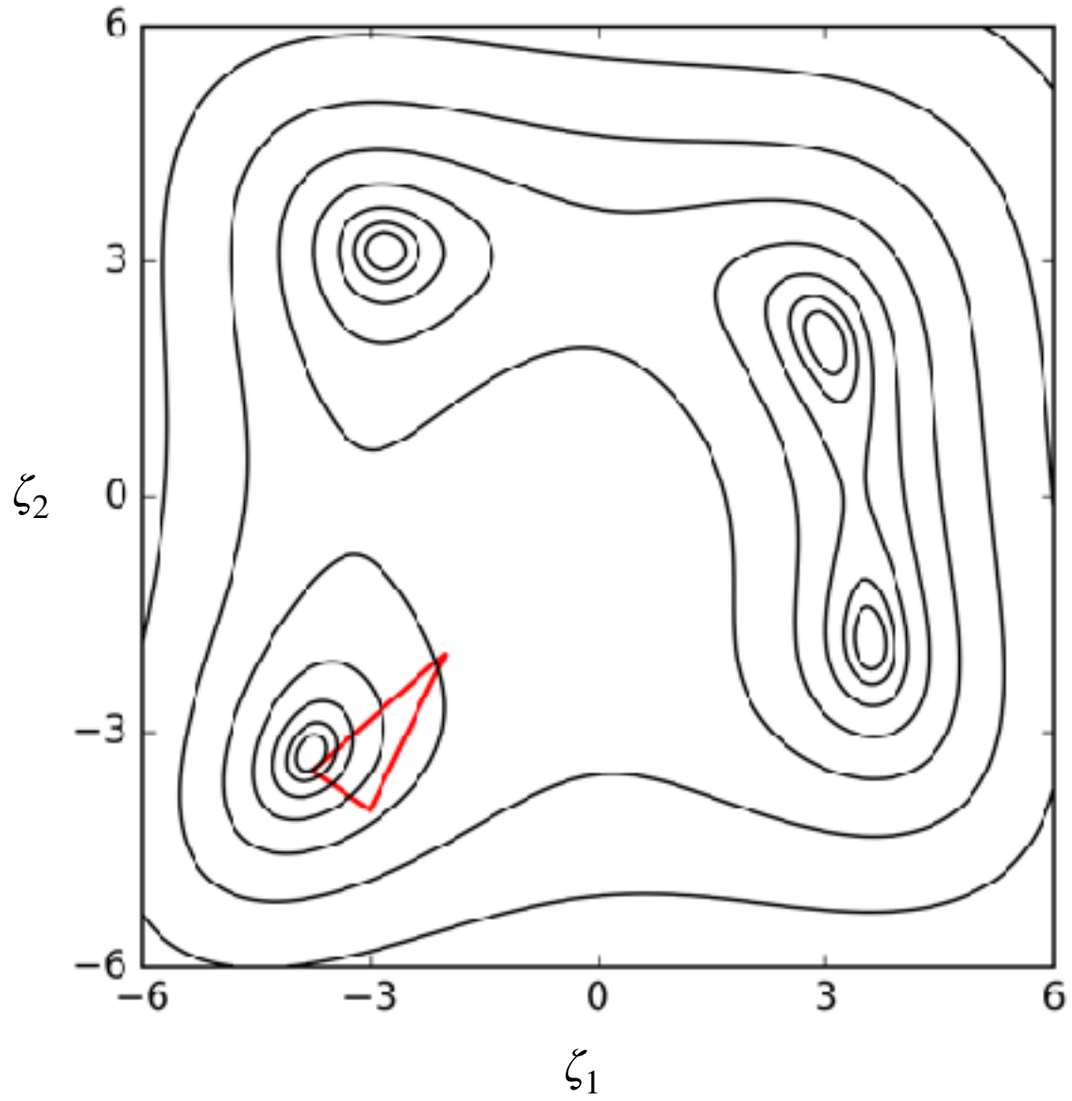
- The method uses the concept of a simplex, which is a special polytope of N+1 vertices in N dimensions. Examples of simplices include a line segment in 1D space, a triangle in 2D space, a tetrahedron in 3D space and so forth.



Nelder–Mead method

- It is a direct search method (based on function comparison) and is often applied to nonlinear optimization problems for which derivatives may not be known.

- The method uses the concept of a simplex, which is a special polytope of N+1 vertices in N dimensions. Examples of simplices include a line segment in 1D space, a triangle in 2D space, a tetrahedron in 3D space and so forth.



Nelder–Mead method

Algorithm -
Current test points: order according to the values at the vertices

1. Compute the centroid of all points except

Nelder–Mead method

Algorithm -
Current test points: order according to the values at the vertices

1. Compute the centroid of all points except

2. *Reflection* - Compute reflected point $\zeta_r = \zeta_0 + \alpha(\zeta_0 - \zeta_{N+1})$ with $\alpha > 0$. If the reflected point is better than the second worst, but not better than the best, i.e. $f(\zeta_1) \leq f(\zeta_r) < f(\zeta_N)$ then obtain a new simplex by replacing the worst point ζ_{N+1} and go to 1.

Nelder–Mead method

Algorithm -
Current test points: order according to the values at the vertices

1. Compute the centroid of all points except

2. *Reflection* - Compute reflected point $\zeta_r = \zeta_0 + \alpha(\zeta_0 - \zeta_{N+1})$ with $\alpha > 0$. If the reflected point is better than the second worst, but not better than the best, i.e. $f(\zeta_1) \leq f(\zeta_r) < f(\zeta_N)$ then obtain a new simplex by replacing the worst point ζ_{N+1} and go to 1.
3. *Expansion* - If the reflected point is the best point so far $f(\zeta_1) > f(\zeta_r)$ then compute the expanded point $\zeta_e = \zeta_0 + \gamma(\zeta_r - \zeta_0)$ with $\gamma > 1$.

If the expanded point is better than the reflected point, $f(\zeta_e) < f(\zeta_r)$ then obtain a new simplex by replacing the worst point ζ_{N+1} with the expanded point ζ_e and go to 1.

else obtain a new simplex by replacing the worst point ζ_{N+1} with the reflexed point ζ_r and go to 1.

Nelder–Mead method

Algorithm -
Current test points: order according to the values at the vertices

1. Compute the centroid of all points except

2. *Reflection* - Compute reflected point $\zeta_r = \zeta_0 + \alpha(\zeta_0 - \zeta_{N+1})$ with $\alpha > 0$. If the reflected point is better than the second worst, but not better than the best, i.e. $f(\zeta_1) \leq f(\zeta_r) < f(\zeta_N)$ then obtain a new simplex by replacing the worst point ζ_{N+1} and go to 1.
3. *Expansion* - If the reflected point is the best point so far $f(\zeta_1) > f(\zeta_r)$ then compute the expanded point $\zeta_e = \zeta_0 + \gamma(\zeta_r - \zeta_0)$ with $\gamma > 1$.

If the expanded point is better than the reflected point, $f(\zeta_e) < f(\zeta_r)$ then obtain a new simplex by replacing the worst point ζ_{N+1} with the expanded point ζ_e and go to 1.

else obtain a new simplex by replacing the worst point ζ_{N+1} with the reflexed point ζ_r and go to 1.

4. *Contraction* - If $f(\zeta_r) > f(\zeta_N)$, compute contracted point $\zeta_c = \zeta_0 + \rho(\zeta_{N+1} - \zeta_0)$ with $0 < \rho < 0.5$. If the contracted point is better than the worst point, i.e. $f(\zeta_c) < f(\zeta_{N+1})$, then obtain a new simplex ζ_{N+1} with the reflexed point ζ_c and go to 1.

Nelder–Mead method

Algorithm -
Current test points: order according to the values at the vertices

1. Compute the centroid of all points except

2. *Reflection* - Compute reflected point $\zeta_r = \zeta_0 + \alpha(\zeta_0 - \zeta_{N+1})$ with $\alpha > 0$. If the reflected point is better than the second worst, but not better than the best, i.e. $f(\zeta_1) \leq f(\zeta_r) < f(\zeta_N)$ then obtain a new simplex by replacing the worst point ζ_{N+1} and go to 1.
3. *Expansion* - If the reflected point is the best point so far $f(\zeta_1) > f(\zeta_r)$ then compute the expanded point $\zeta_e = \zeta_0 + \gamma(\zeta_r - \zeta_0)$ with $\gamma > 1$.

If the expanded point is better than the reflected point, $f(\zeta_e) < f(\zeta_r)$ then obtain a new simplex by replacing the worst point ζ_{N+1} with the expanded point ζ_e and go to 1.

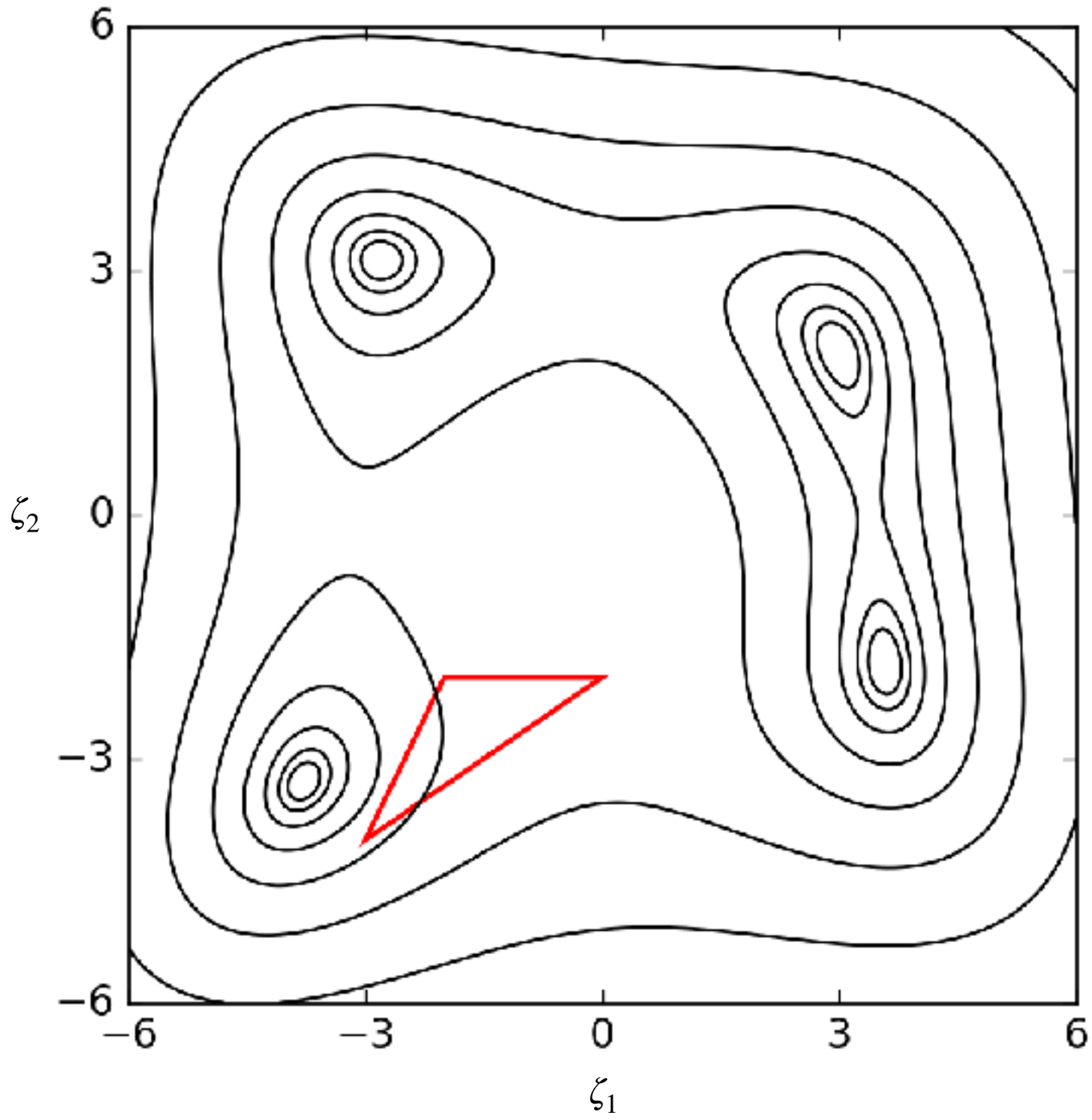
else obtain a new simplex by replacing the worst point ζ_{N+1} with the reflexed point ζ_r and go to 1.

4. *Contraction* - If $f(\zeta_r) > f(\zeta_N)$, compute contracted point $\zeta_c = \zeta_0 + \rho(\zeta_{N+1} - \zeta_0)$ with $0 < \rho < 0.5$. If the contracted point is better than the worst point, i.e. $f(\zeta_c) < f(\zeta_{N+1})$, then obtain a new simplex ζ_{N+1} with the reflexed point ζ_c and go to 1.

5. *Shrink* - Replace all points except the best ζ_1 with $\zeta_i = \zeta_1 + \sigma(\zeta_i - \zeta_1)$ and go to step 1.

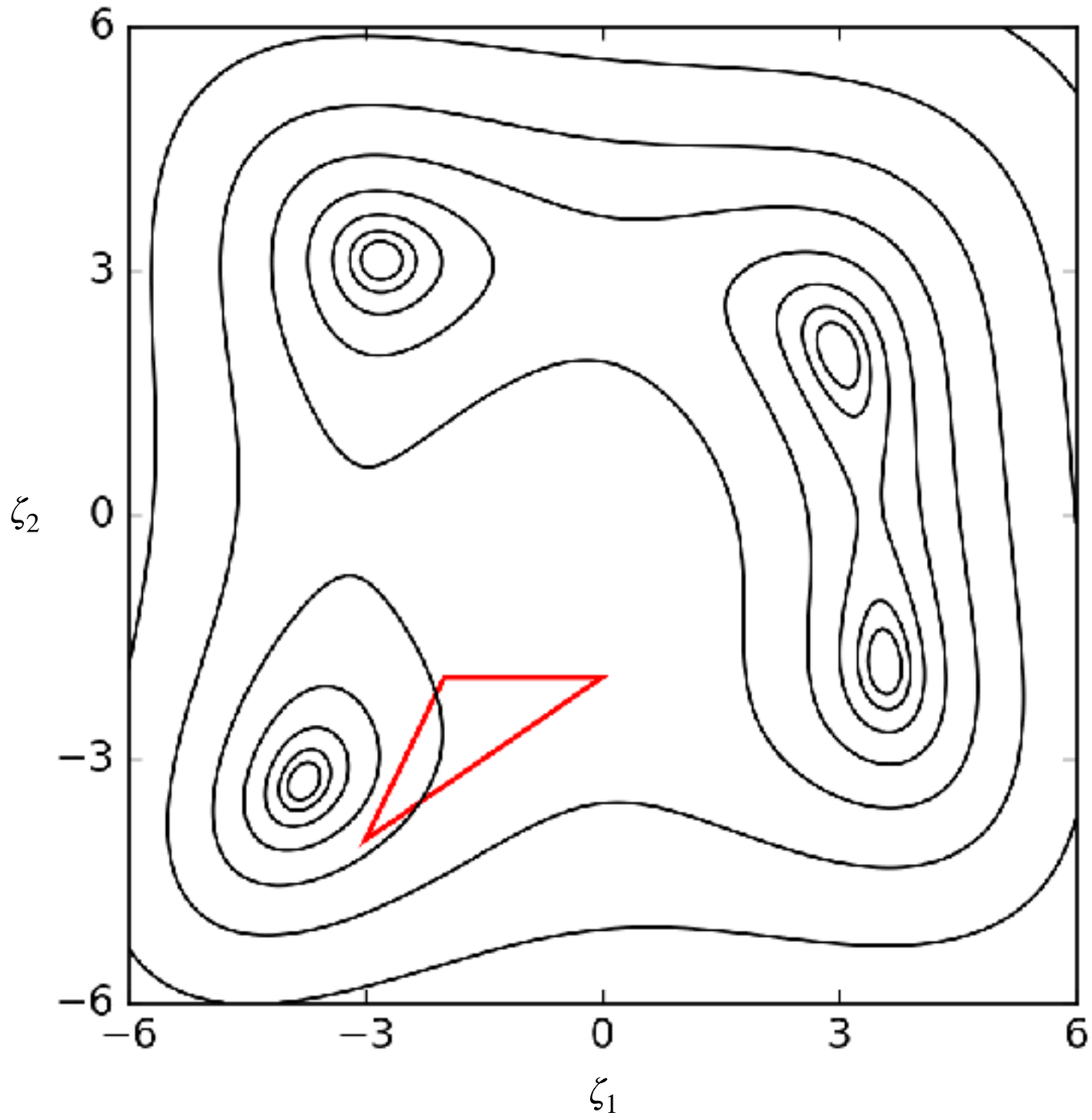
Nelder–Mead method

Simulated annealing searching for a maximum. The objective here is to get to the highest point. In this example, it is not enough to use a simple hill climb algorithm, as there are many local maxima.



Nelder–Mead method

Simulated annealing searching for a maximum. The objective here is to get to the highest point. In this example, it is not enough to use a simple hill climb algorithm, as there are many local maxima.



Variational Strategies

Formalism

Adjoint State Method

Gradient Descent-Like Methods

Simplex Method

Stochastic Methods

Toy Model

Stochastic optimization

- Stochastic optimization (SO) methods are optimization methods that generate and use random variables. As seen before, the random variables appear in the formulation of the optimization problem itself. Stochastic optimization methods generalize deterministic methods for deterministic problems.

Stochastic optimization

- Stochastic optimization (SO) methods are optimization methods that generate and use random variables. As seen before, the random variables appear in the formulation of the optimization problem itself. Stochastic optimization methods generalize deterministic methods for deterministic problems.

- Can be used to control experimental (random) error in the measurements of the criterion. In such cases, knowledge that the function values are contaminated by random "noise" leads naturally to algorithms that use statistical inference tools to estimate the "true" values of the function and/or make statistically optimal decisions about the next steps. Methods of this class include:
  - stochastic approximation (SA), by Robbins and Monro (1951)
  - stochastic gradient descent
  - finite-difference SA by Kiefer and Wolfowitz (1952)
  - simultaneous perturbation SA by Spall (1992)
  - scenario optimization

Stochastic optimization

- On the other hand, even when the data set consists of precise measurements, some methods introduce randomness into the search-process to: accelerate progress, to make the method less sensitive to modeling errors or to escape a local optimum and eventually to approach a global optimum.

- Stochastic optimization methods of this kind include:
  - Simulated annealing by S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi (1983)
  - Quantum annealing
  - Probability Collectives by D.H. Wolpert, S.R. Bieniawski and D.G. Rajnarayan (2011)
  - Reactive search optimization (RSO) by Roberto Battiti, G. Tecchiolli (1994)
  - Cross-entropy method by Rubinstein and Kroese (2004)
  - Random search by Anatoly Zhigljavsky (1991)
  - Informational search
  - Stochastic tunneling
  - Parallel tempering a.k.a. replica exchange
  - Stochastic hill climbing
  - Swarm algorithms
  - Evolutionary algorithms
  - Genetic algorithms by Holland (1975)
  - Evolution strategies
  - Cascade object optimization & modification algorithm (2016)

Stochastic optimization - Simulated annealing

- Simulated annealing (SA) is a probabilistic technique for approximating the global optimum of a given function. For problems where finding an approximate global optimum is more important than finding a precise local optimum in a fixed amount of time, simulated annealing may be preferable to exact algorithms such as gradient descent.

Stochastic optimization - Simulated annealing

- Simulated annealing (SA) is a probabilistic technique for approximating the global optimum of a given function. For problems where finding an approximate global optimum is more important than finding a precise local optimum in a fixed amount of time, simulated annealing may be preferable to exact algorithms such as gradient descent.

- The name of the algorithm comes from annealing in metallurgy, a technique involving heating and controlled cooling of a material to alter its physical properties.

Stochastic optimization - Simulated annealing

- Simulated annealing (SA) is a probabilistic technique for approximating the global optimum of a given function. For problems where finding an approximate global optimum is more important than finding a precise local optimum in a fixed amount of time, simulated annealing may be preferable to exact algorithms such as gradient descent.

- The name of the algorithm comes from annealing in metallurgy, a technique involving heating and controlled cooling of a material to alter its physical properties.

- The problems solved by SA are currently formulated by an objective function of many variables, subject to several constraints.

Stochastic optimization - Simulated annealing

- Simulated annealing (SA) is a probabilistic technique for approximating the global optimum of a given function. For problems where finding an approximate global optimum is more important than finding a precise local optimum in a fixed amount of time, simulated annealing may be preferable to exact algorithms such as gradient descent.

- The name of the algorithm comes from annealing in metallurgy, a technique involving heating and controlled cooling of a material to alter its physical properties.

- The problems solved by SA are currently formulated by an objective function of many variables, subject to several constraints.

- This notion of slow cooling implemented in the simulated annealing algorithm is interpreted as a slow decrease in the probability of accepting worse solutions as the solution space is explored.

Stochastic optimization - Simulated annealing

- Simulated annealing (SA) is a probabilistic technique for approximating the global optimum of a given function. For problems where finding an approximate global optimum is more important than finding a precise local optimum in a fixed amount of time, simulated annealing may be preferable to exact algorithms such as gradient descent.

- The name of the algorithm comes from annealing in metallurgy, a technique involving heating and controlled cooling of a material to alter its physical properties.

- The problems solved by SA are currently formulated by an objective function of many variables, subject to several constraints.

- This notion of slow cooling implemented in the simulated annealing algorithm is interpreted as a slow decrease in the probability of accepting worse solutions as the solution space is explored.

- The simulation can be performed either by a solution of kinetic equations for density functions or by using the stochastic sampling method. The method is an adaptation of the Metropolis–Hastings algorithm, a Monte Carlo method to generate sample states of a thermodynamic system, published by N. Metropolis et al. in 1953.

Stochastic optimization - Simulated annealing

- The probability of making the transition from the current state ζ to a candidate new state ζ_{new} is specified by an acceptance probability function $\mathbb{P}(F(\zeta), F(\zeta_{new}), T)$ that depends on the values of $F(\zeta)$ and $F(\zeta_{new})$ but also on a global time-varying parameter T called the temperature.

Stochastic optimization - Simulated annealing

- The probability of making the transition from the current state ζ to a candidate new state ζ_{new} is specified by an acceptance probability function $\mathbb{P}(F(\zeta), F(\zeta_{new}), T)$ that depends on the values of $F(\zeta)$ and $F(\zeta_{new})$ but also on a global time-varying parameter T called the temperature.
- Let ζ_0 be the initial state and k_{max} the maximum of steps. In the process:
  - the call neighbour(s) generate a randomly chosen neighbour(s) of a given state ζ ;
  - the call random(0, 1) pick and return a value in the range [0, 1], uniformly at random ;
  - the annealing schedule is defined by the call temperature(r), which should yield the temperature to use, given the fraction r of the time budget that has been expended so far.

Stochastic optimization - Simulated annealing

- The probability of making the transition from the current state ζ to a candidate new state ζ_{new} is specified by an acceptance probability function $\mathbb{P}(F(\zeta), F(\zeta_{new}), T)$ that depends on the values of $F(\zeta)$ and $F(\zeta_{new})$ but also on a global time-varying parameter T called the temperature.
- Let ζ_0 be the initial state and k_{max} the maximum of steps. In the process:
  - the call neighbour(s) generate a randomly chosen neighbour(s) of a given state ζ ;
  - the call random(0, 1) pick and return a value in the range [0, 1], uniformly at random ;
  - the annealing schedule is defined by the call temperature(r), which should yield the temperature to use, given the fraction r of the time budget that has been expended so far.
- Algorithm -

$\zeta = \zeta_0$

For $k = 0 : k_{max}$

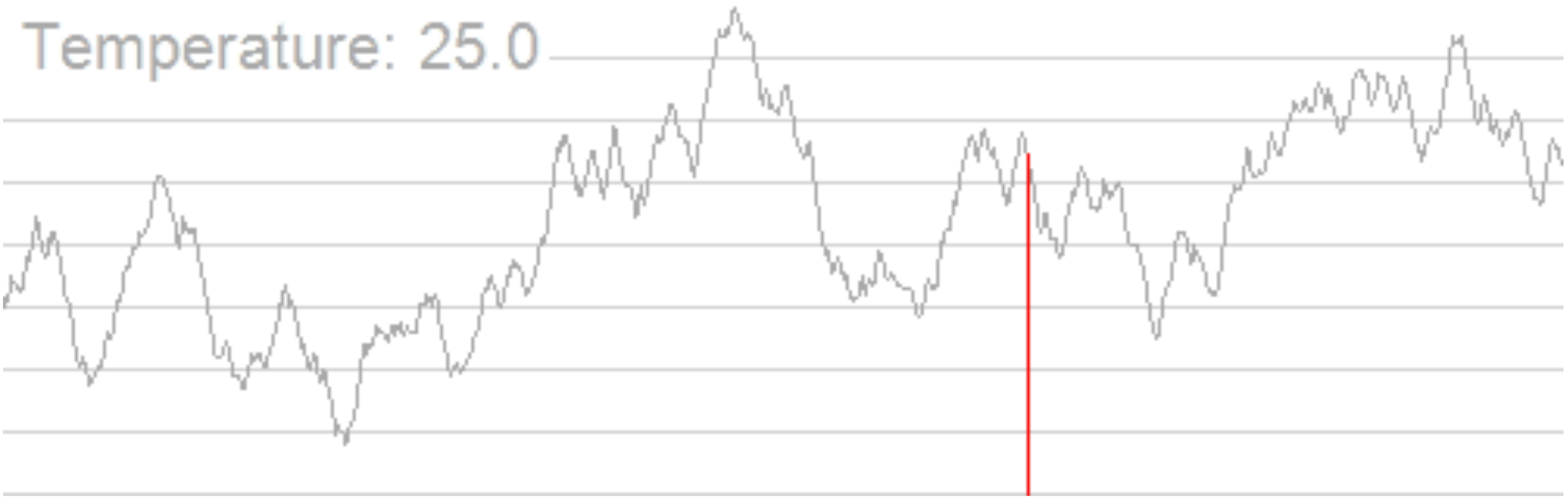
$T \leftarrow$ temperature$(1 - (k+1)/k_{max})$

Pick a random neighbour, $\zeta_{new} \leftarrow$ neighbour

If $\mathbb{P}(F(\zeta), F(\zeta_{new}), T) \geq$ random$(0,1)$: $\zeta \leftarrow \zeta_{new}$

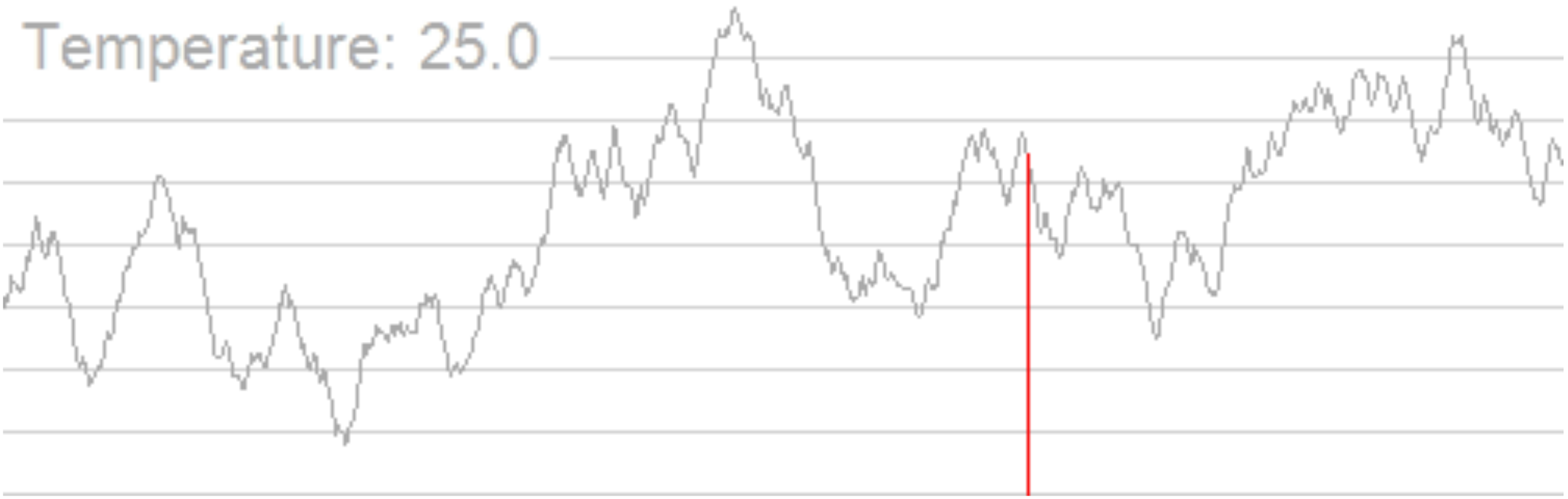
Stochastic optimization - Simulated annealing

Simulated annealing searching for a maximum. The objective here is to get to the highest point. In this example, it is not enough to use a simple hill climb algorithm, as there are many local maxima.



Stochastic optimization - Simulated annealing

Simulated annealing searching for a maximum. The objective here is to get to the highest point. In this example, it is not enough to use a simple hill climb algorithm, as there are many local maxima.



Variational Strategies

Formalism

Adjoint State Method

Gradient Descent-Like Methods

Simplex Method

Stochastic Methods

Toy Model

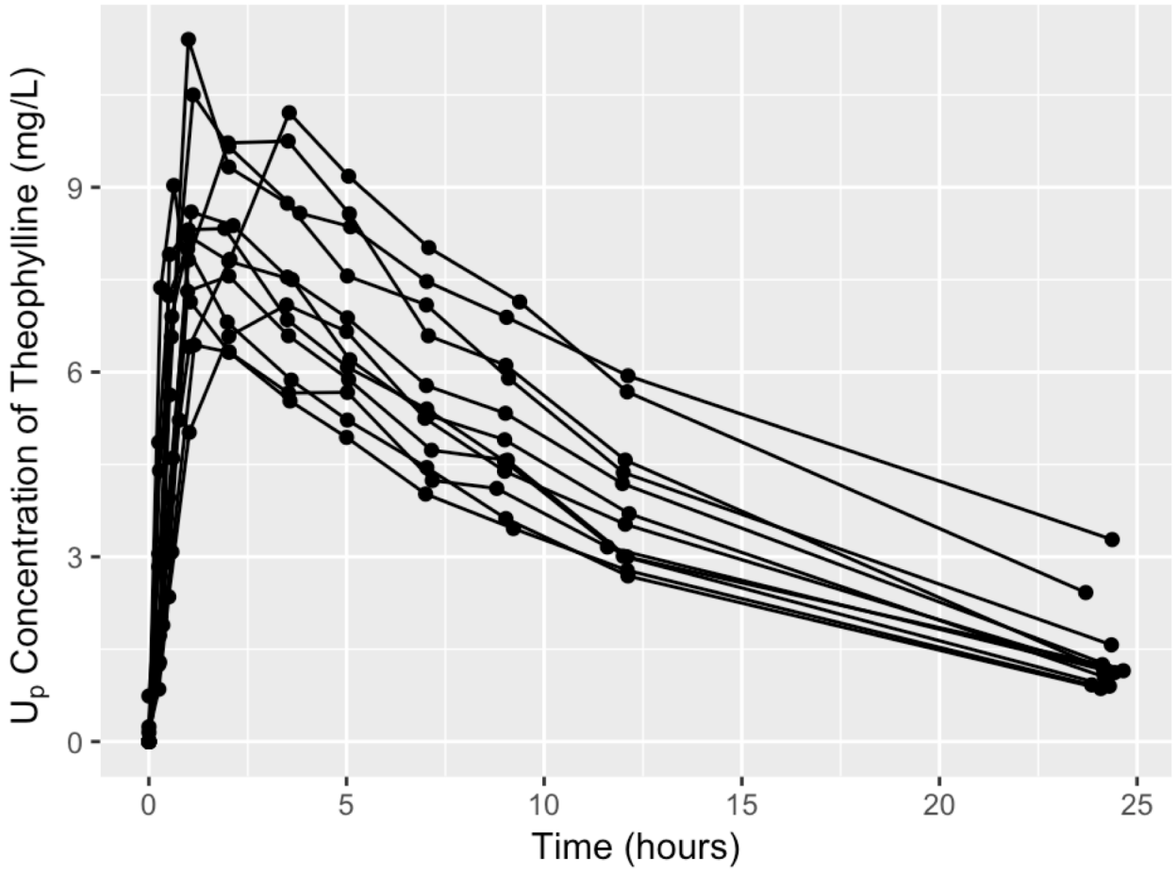
Toy model

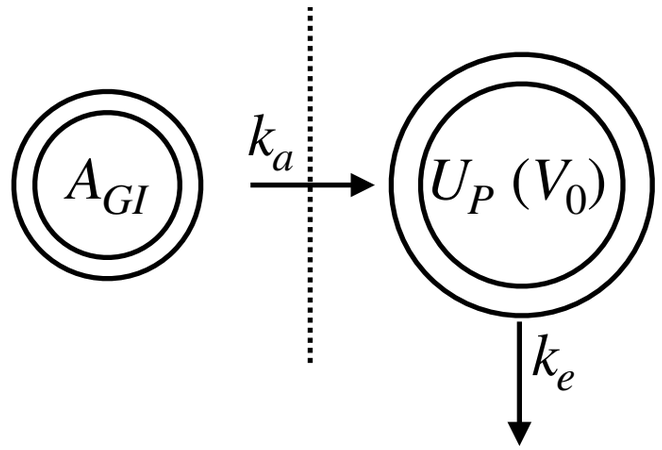
- Pharmacokinetics one-compartment model with first-order absorption and elimination

$$\begin{cases} \dot{A}_{GI} = -k_a A_{GI} \\ \dot{U}_P = \dfrac{k_a}{V_0} A_{GI} - k_e U_P \end{cases}$$

A_{GI} Dose in Gastro-intestinal compartment (mg)
U_P Concentration Plasma compartment (mg/L)
k_a Drug absorption rate (1/h)
k_e Drug elimination rate (1/h)
V_0 Volume of distribution (L)

- Observations $z = U_P$





Toy model

- What do we want to estimate? $\theta = [U_P^0, A_{GI}^0, k_a, k_e, V_0]$
- Are they all identifiable?
- Observations: $z = U_P$
- The exact solution writes:

$$\begin{cases} \dot{A}_{GI} = -k_a A_{GI} \\ \dot{U}_P = \dfrac{k_a}{V_0} A_{GI} - k_e U_P \end{cases}$$

$$U_P(t,\theta) = \left(U_P^0 + \frac{A_{GI}^0 k_a}{V_0(k_a - k_e)} \right) e^{-ket} - \frac{A_{GI}^0 k_a}{V_0(k_a - k_e)} e^{-kat}$$

Toy model

- What do we want to estimate? $\theta = [U_P^0, A_{GI}^0, k_a, k_e, V_0]$
- Are they all identifiable?
- Observations: $z = U_P$
- The exact solution writes:

$$\begin{cases} \dot{A}_{GI} = -k_a A_{GI} \\ \dot{U}_P = \dfrac{k_a}{V_0} A_{GI} - k_e U_P \end{cases}$$

$$U_P(t,\theta) = \left(U_P^0 + \frac{A_{GI}^0 k_a}{V_0(k_a - k_e)} \right) e^{-ket} - \frac{A_{GI}^0 k_a}{V_0(k_a - k_e)} e^{-kat}$$

- The injectivity of the function leads to

$$\psi_T : \begin{cases} \mathbb{R}^5 \to \mathscr{C}^0((0,T),\mathbb{R}) \\ \theta \mapsto (t \mapsto U_P(t,\theta)) \end{cases}$$

leads to

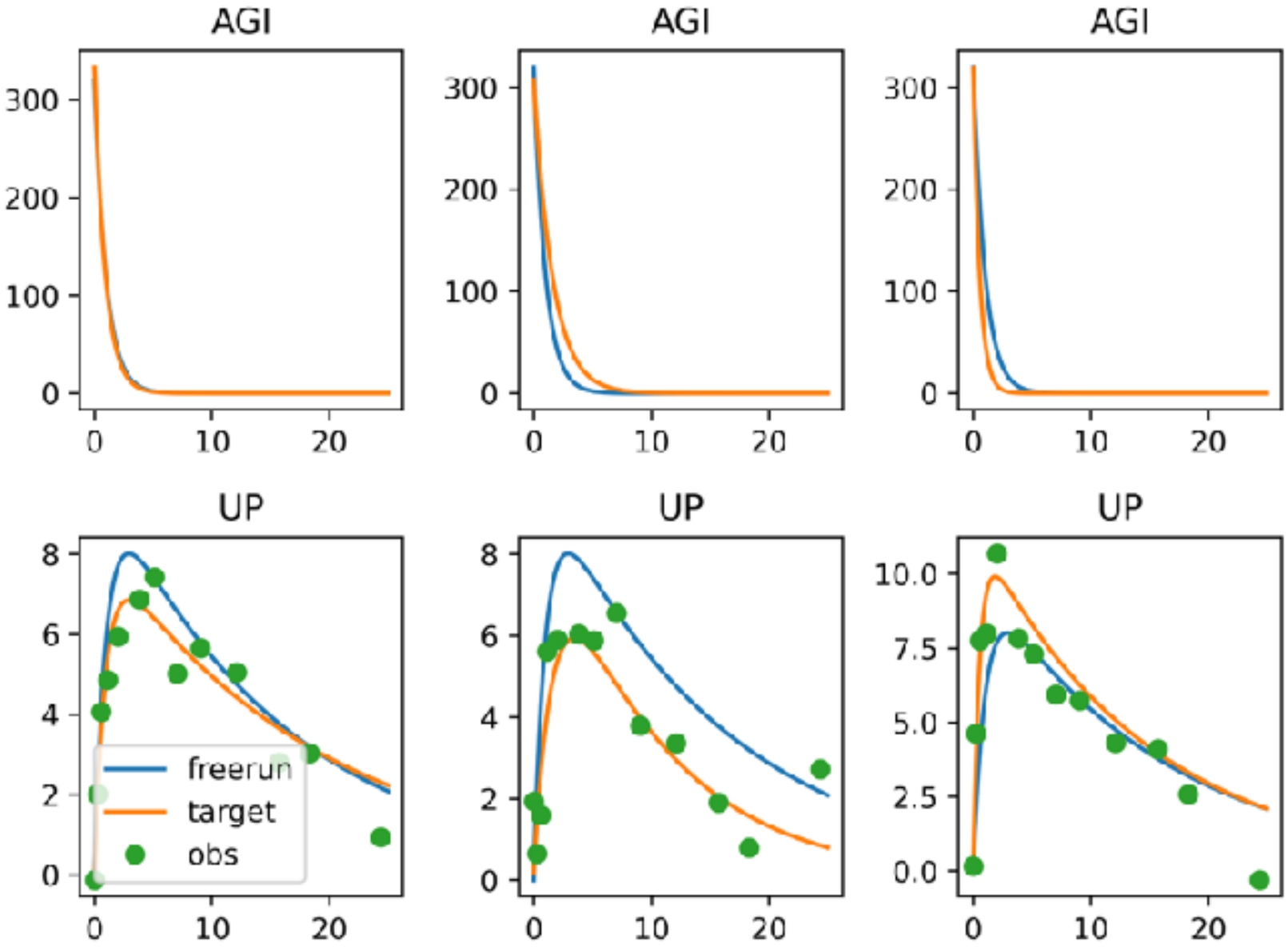
$$(U_P^0)^1 = (U_P^0)^2$$

$$\frac{(A_{GI}^0)^1 k_a^1}{V_0^1(k_a^1 - k_e^1)} = \frac{(A_{GI}^0)^2 k_a^2}{V_0^2(k_a^2 - k_e^2)} .$$

- And robustness to measurements errors?

Toy model

- Freerun obtained with priors values
- Target obtained with random values following the Gaussian laws
- Observations obtained by adding noise and considering only few times



Generate different values of parameters and different initial conditions with

$$k_e = e^{\tilde{k}_e}, \ \textbf{s.t}\ \tilde{k}_e \sim \mathcal{N}(\mu_{k_e}, \sigma_{k_e})$$

$$V_0 = e^{\tilde{V}_0}, \ \textbf{s.t}\ \tilde{V}_0 \sim \mathcal{N}(\mu_{V_0}, \sigma_{V_0})$$

$$A_{GI}(0) \sim \mathcal{N}(\mu_{A_{GI}}, \sigma_{A_{GI}})$$

$$U_P(0) \sim \mathcal{N}(\mu_{U_P}, \sigma_{U_P})$$

We build noisy and time-sampled observations of synthetic patients.

Toy model

functions.py

```python

def residual(estin,par,target):
    """
    Difference between target and solution
    """
    # Find paramaters which have to be estimated or not (etc ...)
    N_params = int(len(par['params']))
    all_params = np.linspace(0,N_params-1,N_params)
    N_estim_params = int(len(par['which_estim_params']))
    which_estim_params = np.array(par['which_estim_params'])
    which_not_estim_params = np.setdiff1d(all_params,which_estim_params)
    theta = np.zeros((N_params,1))
    for k in range(0,N_estim_params):
        theta[which_estim_params[k]] = estim[par['params'][which_estim_params[k]]].value
    for j in range(0,len(which_not_estim_params)):
        theta[int(which_not_estim_params[j])] = par['prior_'+par['parans'][int(which_not_estim_params[j])]]
    theta = theta[:,0]

    N_IC = int(len(par['IC']))
    all_IC = np.linspace(0,N_IC-1,N_IC)
    N_estim_IC = int(len(par['which_estim_IC']))
    which_estim_IC = np.array(par['which_estim_IC'])
    which_not_estim_IC = np.setdiff1d(all_IC,which_estim_IC)
    IC = np.zeros((N_IC,1))
    for k in range(0,N_estim_IC):
        IC[which_estim_IC[k]] = estim[par['IC'][which_estim_IC[k]]].value
    for j in range(0,len(which_not_estim_IC)):
        IC[int(which_not_estim_IC[j])] = par['prior_'+par['IC'][int(which_not_estim_IC[j])]]
    IC = IC[:,0]

    [t,u] = forward(par,IC,theta)
    u_obs = np.interp(np.array(par['time_obs']), t, eval(par['observator_model'] + "(u)"))

    diffmean = target-u_obs

    return diffmean
```

models.py

```python

def PK_observation(u):
    """
    Observation - only UP
    """
    return u[1,:]

```

Toy model

params_PK_synthetic_data.yaml

```yaml
N_subjects: 3

# Time
dt: 0.005
time_simu: 25

# Parameters: ka, ke, V0
prior_ka: 0
prior_ke: -2.75
prior_V0: 3.5
std_prior_ka: 0.4
std_prior_ke: 0.4
std_prior_V0: 0.4

# Initialisation of variables
prior_AGI: 320
prior_UP: 0
std_prior_AGI: 10
std_prior_UP: 0.1

# Model
model: 'PK_model'

# OBSERVATIONS
# Data noise (UP)
std_noise: 0 #1
# Observation time
time_obs: [0, 0.25, 0.57, 1.12, 2.02, 3.82, 5.1, 7.03, 9.05, 12.1, 15.7, 18.3, 24.37]

# ESTIMATION
observator_model: 'PK_observation'
params: ['ka', 'ke', 'V0']
which_estim_params: [0, 1, 2]
IC: ['AGI', 'UP']
which_estim_IC: []
#strategy: 'Levenberg-Marquardt'
#strategy: 'Trust-Region-Reflective'
#strategy: 'Nelder-Mead'
strategy: 'dual_annealing'
```

Toy model

script_estimate_PK_with_synthetic_data.py

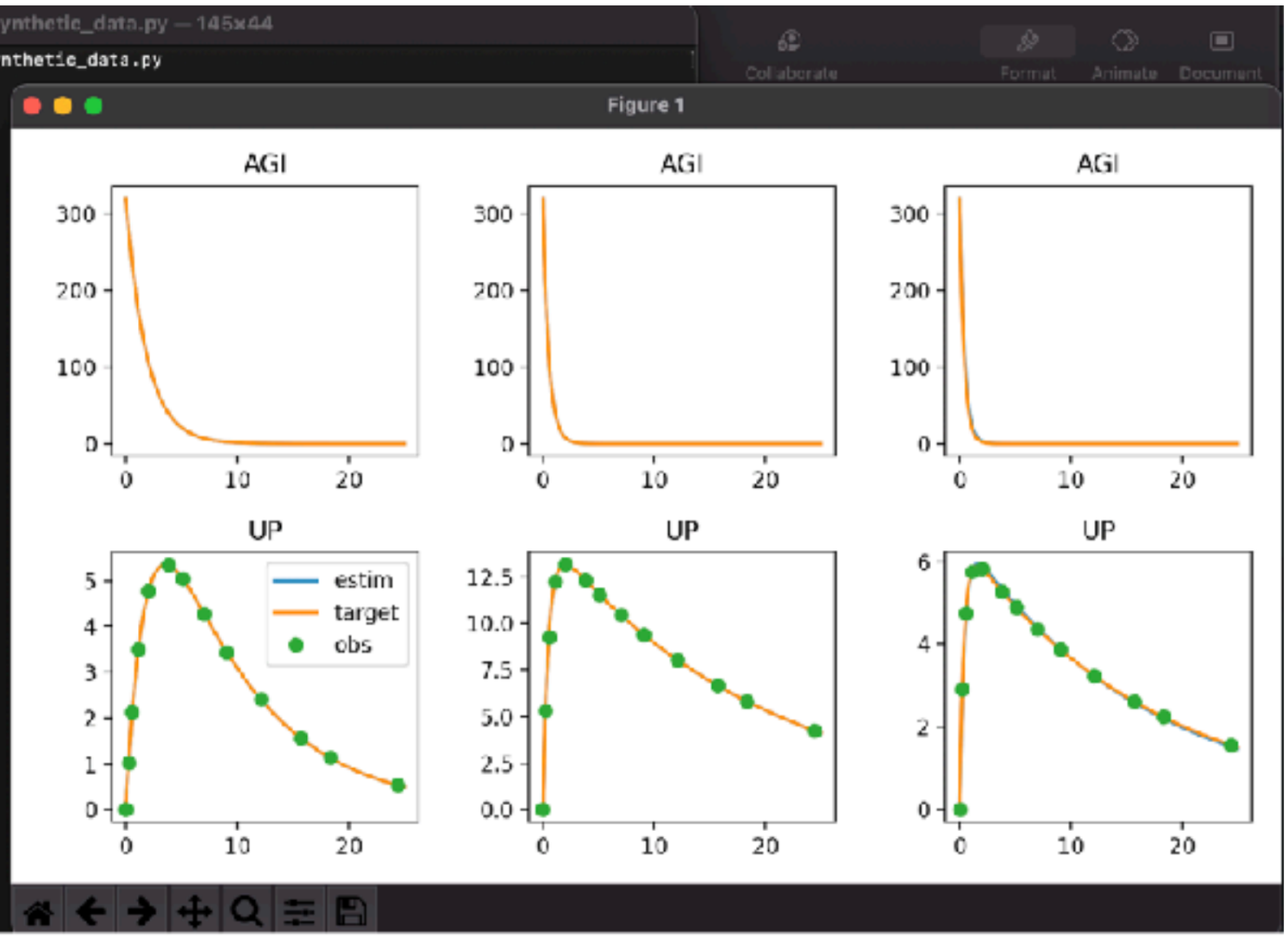
```python
# Read parameters file
paramsfilename = "Params/params_PK_synthetic_data.yaml"
with open(paramsfilename) as f:
    par = yaml.load(f, Loader=yaml.FullLoader)

# Define Parameters
params = Parameters()

## BUILD SYNTHETIC DATA -
# Build random variables (IC)
ICrand = np.random.randn(int(par['N_subjects']),2)
IC = np.zeros([int(par['N_subjects']),2])
IC[:,0] = float(par['prior_AGI'])
IC[:,1] = float(par['prior_VP'])
for i in range(0, int(len(par['which_estim_IC']))):
    k = par['which_estim_IC'][i]
    IC[:,k] = float(par['prior_'+par['IC'][k]])+float(par['std_prior_'+par['IC'][k]])*ICrand[:,k]
    IC[:,k] = IC[:,k]*(IC[:,k] > 0) # IC => > 0
    if !(par['strategy'] == 'Levenberg-Marquardt') or (par['strategy'] == 'Nelder-Mead')):
        params.add(par['IC'][k], value=par['prior_'+par['IC'][k]])
    elif ((par['strategy'] == 'Trust-Region-Reflective') or (par['strategy'] == 'dual_annealing')):
        params.add(par['IC'][k], value=par['prior_'+par['IC'][k]],
                   min=0, # IC -> > 0
                   max=par['prior_'+par['IC'][k]]+2*par['std_prior_'+par['IC'][k]])

# Build random variables (parameters)
thetarand = np.random.randn(int(par['N_subjects']),3)
theta = np.zeros([int(par['N_subjects']),3])
theta[:,0] = float(par['prior_ka'])
theta[:,1] = float(par['prior_ke'])
theta[:,2] = float(par['prior_V0'])
for i in range(0, int(len(par['which_estim_params']))):
    k = par['which_estim_params'][i]
    theta[:,k] = float(par['prior_'+par['params'][k]])+float(par['std_prior_'+par['params'][k]])*thetarand[:,k]
    params.add(par['params'][k], value=par['prior_'+par['params'][k]],
               min=par['prior_'+par['params'][k]]-2*par['std_prior_'+par['params'][k]], # for dual annealing
               max=par['prior_'+par['params'][k]]+2*par['std_prior_'+par['params'][k]]) # for dual annealing
```

```python
## ESTIMATE PARAMETERS
for i in range(0, int(par['N_subjects'])):
    # Target model
    [t,u] = forward(par,IC[i,:],theta[i,:])
    IC_pat = IC[i,:]
    theta_pat = theta[i,:]

    # Build noisy and subsampling data
    noise = np.random.randn(1,np.array(par['time_obs']).size)
    u_obs = np.interp(np.array(par['time_obs']), t, u[1,:])+float(par['std_noise'])*noise[0,:]

    timeinit = time.perf_counter()
    if (par['strategy'] == 'Levenberg-Marquardt'):
        result = minimize(residual, params, args=(par,u_obs), method='leastsq')
    elif (par['strategy'] == 'Trust-Region-Reflective'):
        result = minimize(residual, params, args=(par,u_obs), method='least_squares')
    elif (par['strategy'] == 'Nelder-Mead'):
        result = minimize(residual, params, args=(par,u_obs), method='nelder')
    elif (par['strategy'] == 'dual_annealing'):
        result = minimize(residual, params, args=(par,u_obs), method='dual_annealing')
    timeend = time.perf_counter()
    print("---------------------------")
    print("Time = ", timeend-timeinit, 's')

    print("---------------------------")
    for l in range(0, int(len(par['which_estim_params']))):
        k = par['which_estim_params'][l]
        print('estim ',par['params'][k],' = ',result.params[par['params'][k]].value,' compared to true value = ',theta[i,k])
        theta_pat[k] = result.params[par['params'][k]].value

    for l in range(0, int(len(par['which_estim_IC']))):
        k = par['which_estim_IC'][l]
        print('estim ',par['IC'][k],' = ',result.params[par['IC'][k]].value,' compared to true value = ',IC[i,k])
        IC_pat[k] = result.params[par['IC'][k]].value
    print("---------------------------")

    [testim,uestim] = forward(par,IC_pat,theta_pat)
```

Toy model

- Example with Trust-Region-Reflective algorithm for 3 synthetic patients

```
Code — Python script_estimate_PK_with_synthetic_data.py — 145×44
acollin@MacBook-Pro-de-Annabelle-Collin Code % python3 script_estimate_PK_with_synthetic_data.py
----------------------------
Time =  4.1145811720000005 s
----------------------------
estim  ka  =  -0.6874622555321639  compared to true value =  -0.6874622555321685
estim  ke  =  -2.1834131571306114  compared to true value =  -2.1834131571306123
estim  V0  =  3.6581688788054073  compared to true value =  3.6581688788548735
----------------------------
----------------------------
Time =  3.7944252380000007 s
----------------------------
estim  ka  =  0.5982894190112034  compared to true value =  0.5982894190112036
estim  ke  =  -2.9685021901377016  compared to true value =  -2.968682190137701
estim  V0  =  3.0864566974694005  compared to true value =  3.086466697469405
----------------------------
----------------------------
Time =  5.188562965 s
----------------------------
estim  ka  =  0.2999999999999999  compared to true value =  0.8878888325057861
estim  ke  =  -2.7979579357328155  compared to true value =  -2.899281518193741
estim  V0  =  3.8875945606863974  compared to true value =  3.9044943442845585
----------------------------
```



Toy model

- Comparison between algorithms for 1000 synthetic patients

Error

$$\frac{1}{N_p}\sum_{i=1}^{N_p}\frac{1}{N_\zeta}\sum_{j=1}^{N_\zeta}|\zeta_i-\zeta_i^{true}|$$

Algorithm	Error	Computational times (mean)
Levenberg-Marquardt	0	0.04 s
Trust-Region-Reflective	0.014	0.059 s
Nelder-Mead	0.001	0.312 s
Dual-Annealing (only 100 patients)	0.016	9 s

Toy model

- Comparison between algorithms for 1000 synthetic patients

Error

$$\frac{1}{N_p}\sum_{i=1}^{N_p}\frac{1}{N_\zeta}\sum_{j=1}^{N_\zeta}|\zeta_i-\zeta_i^{true}|$$

Algorithm	Error	Computational times (mean)
Levenberg-Marquardt	0	0.04 s
Trust-Region-Reflective	0.014	0.059 s
Nelder-Mead	0.001	0.312 s
Dual-Annealing (only 100 patients)	0.016	9 s

Works very well!
Locally identifiable …

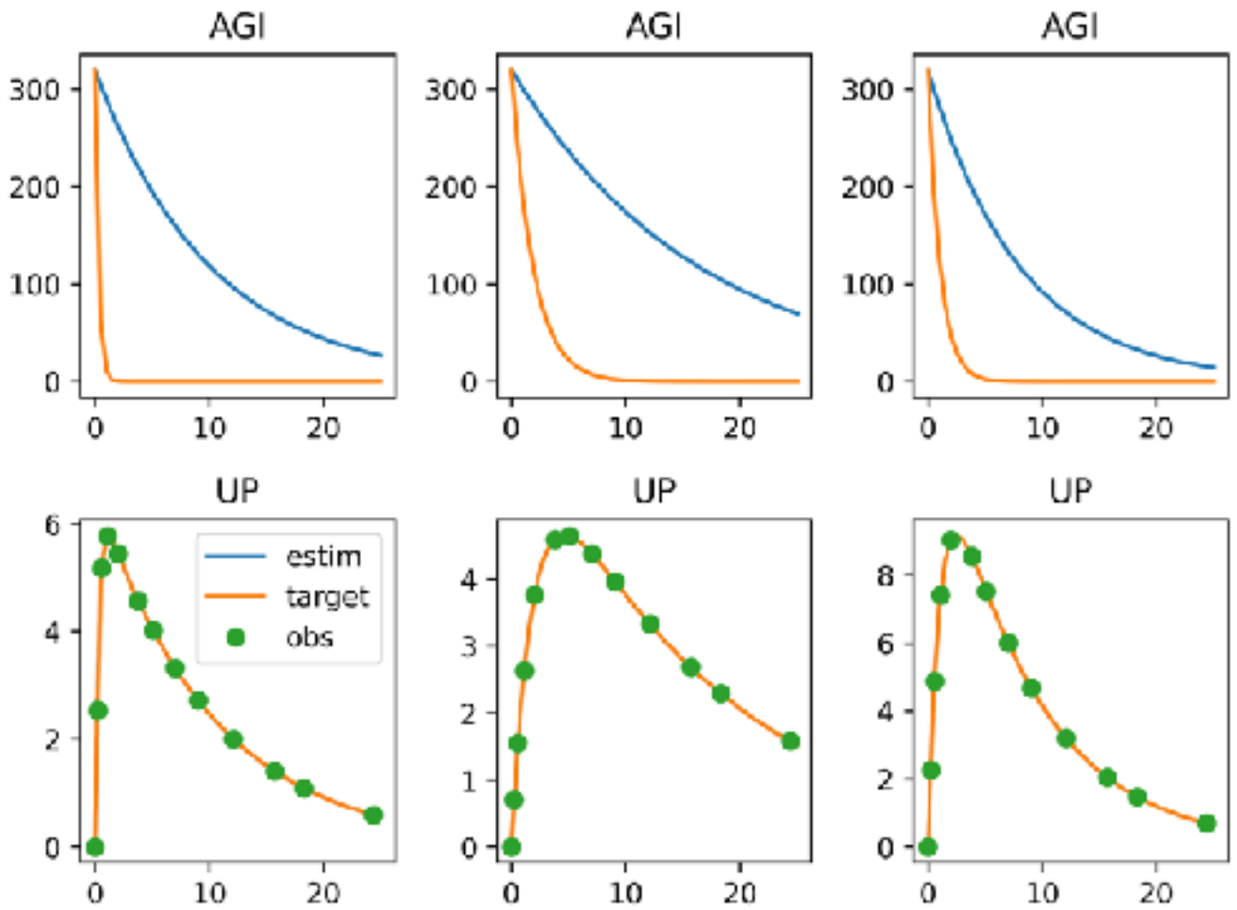
Toy model

- Modify priors for Levenberg-Marquardt algorithm

Priors	Error
True priors	0
(0,0,0)	2.755

Error

$$\frac{1}{N_p}\sum_{i=1}^{N_p}\frac{1}{N_\zeta}\sum_{j=1}^{N_\zeta}|\zeta_i-\zeta_i^{true}|$$



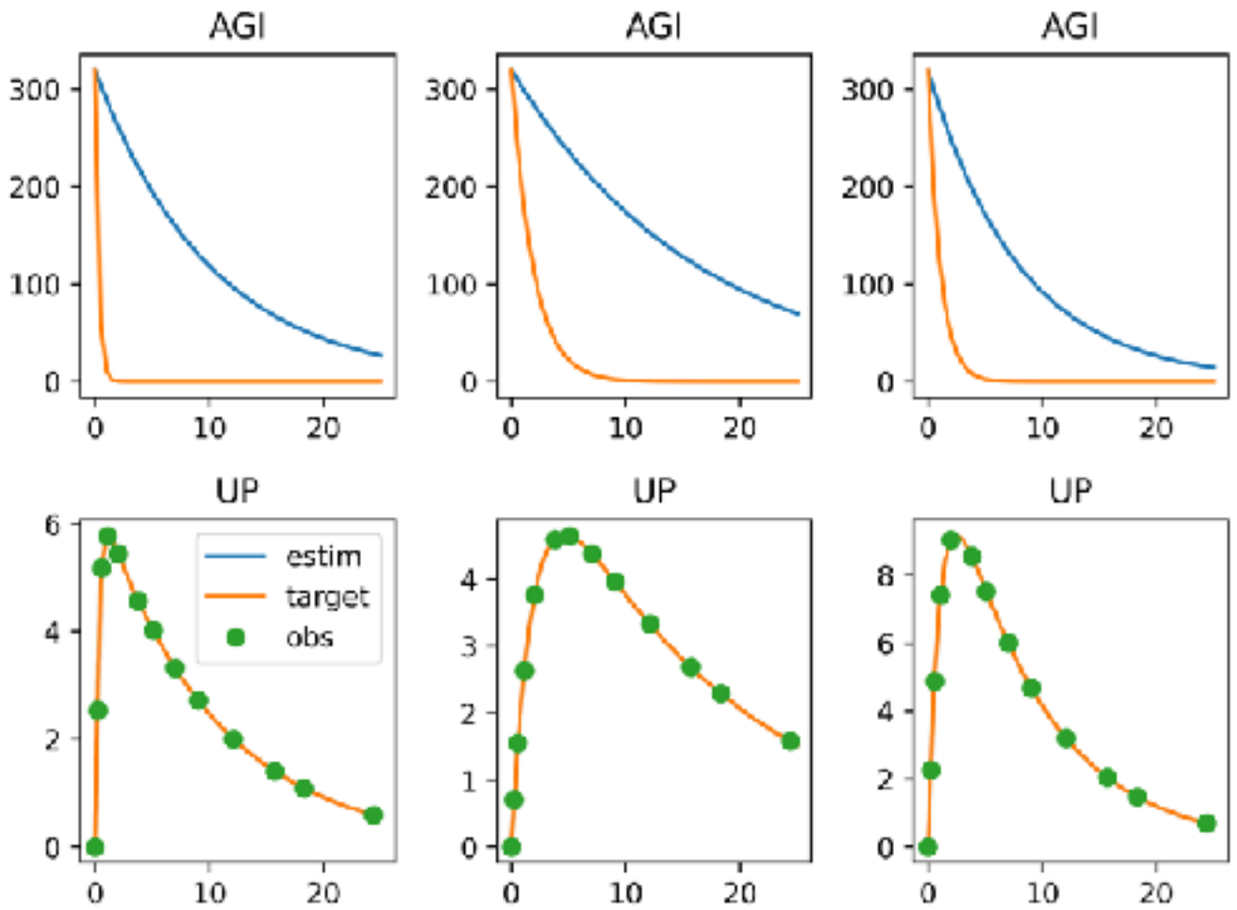
Toy model

- Modify priors for Levenberg-Marquardt algorithm

Priors	Error
True priors	0
(0,0,0)	2.755

Error

$$\frac{1}{N_p}\sum_{i=1}^{N_p}\frac{1}{N_\zeta}\sum_{j=1}^{N_\zeta}|\zeta_i-\zeta_i^{true}|$$



Only locally identifiable …

Fits are good but the estimated parameters are not.

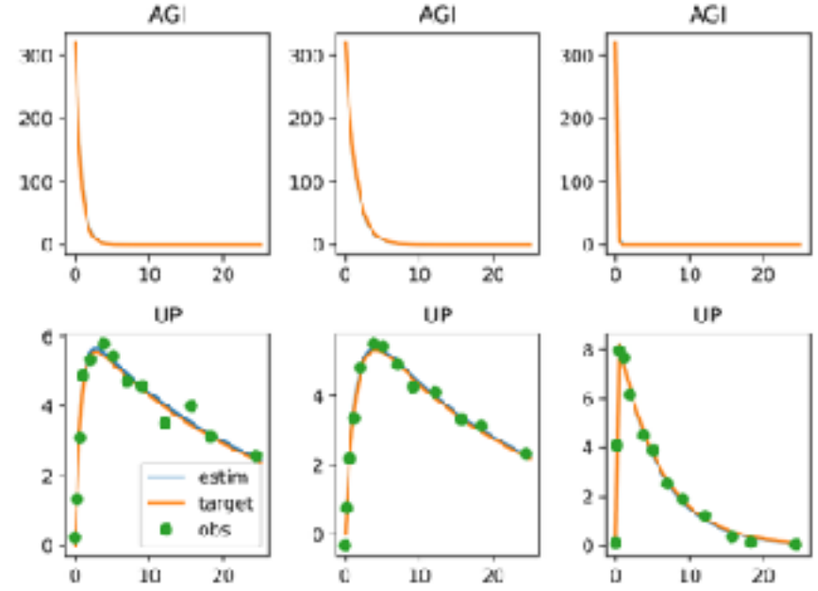
Toy model

- Increase measurement noise for Levenberg-Marquardt algorithm

Noise (standard	Error
0	0
0.2	0.03
0.4	0.102
1	0.156

Error

$$\frac{1}{N_p}\sum_{i=1}^{N_p}\frac{1}{N_\zeta}\sum_{j=1}^{N_\zeta}|\zeta_i-\zeta_i^{true}|$$



Noise standard deviation = 0.2

Noise standard deviation = 0.4

Noise standard deviation = 1

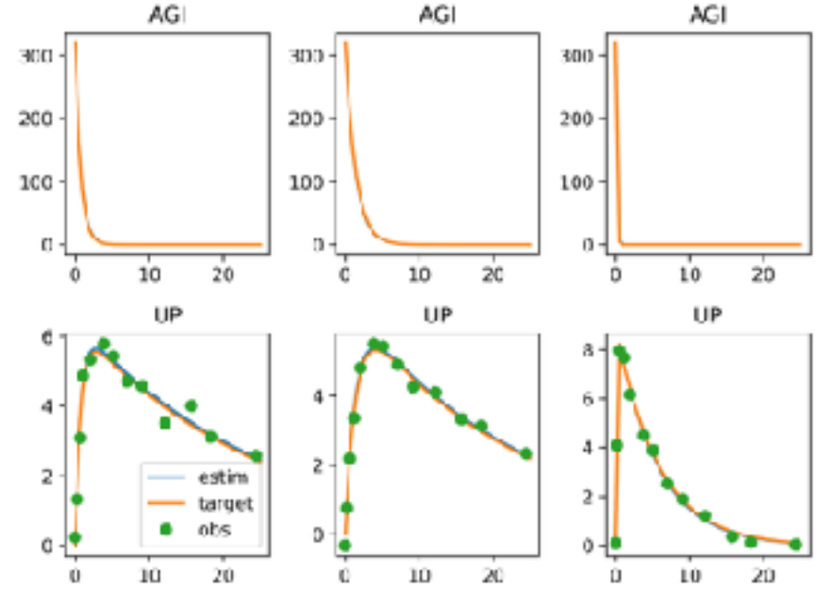
Toy model

- Increase measurement noise for Levenberg-Marquardt algorithm

Noise (standard	Error
0	0
0.2	0.03
0.4	0.102
1	0.156

Error

$$\frac{1}{N_p}\sum_{i=1}^{N_p}\frac{1}{N_\zeta}\sum_{j=1}^{N_\zeta}|\zeta_i-\zeta_i^{true}|$$



Noise standard deviation = 0.2

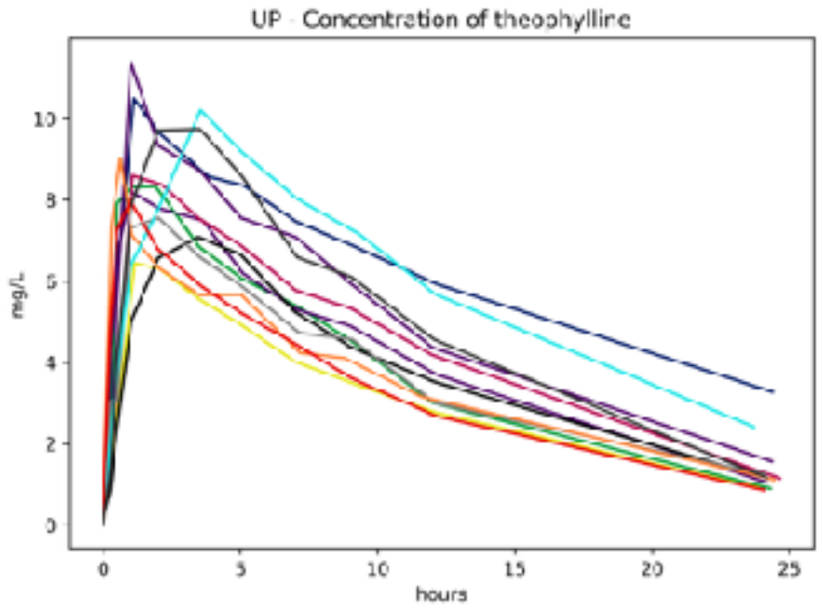
Noise standard deviation = 0.4

Noise standard deviation = 1

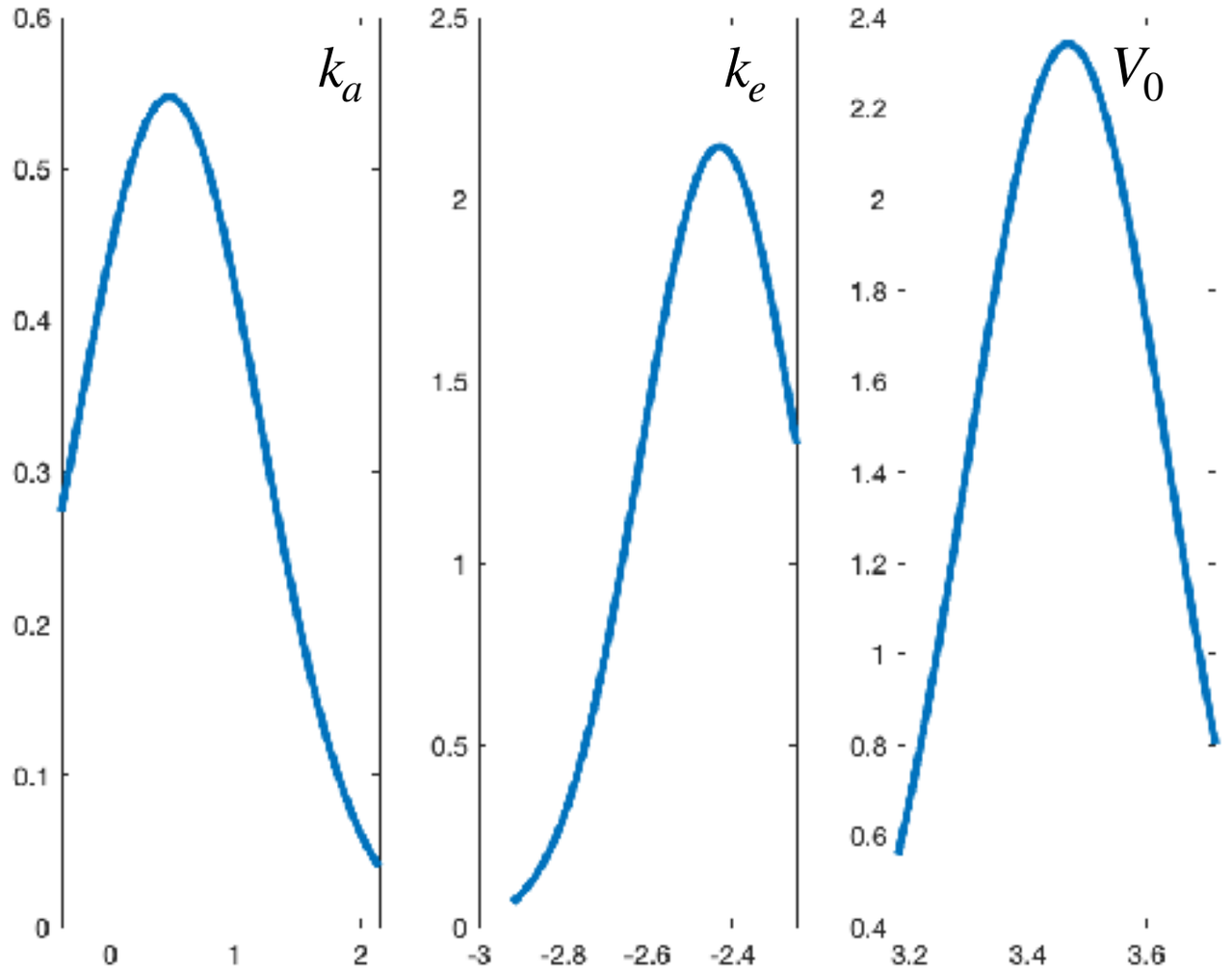
Pratical identifiability

Toy model

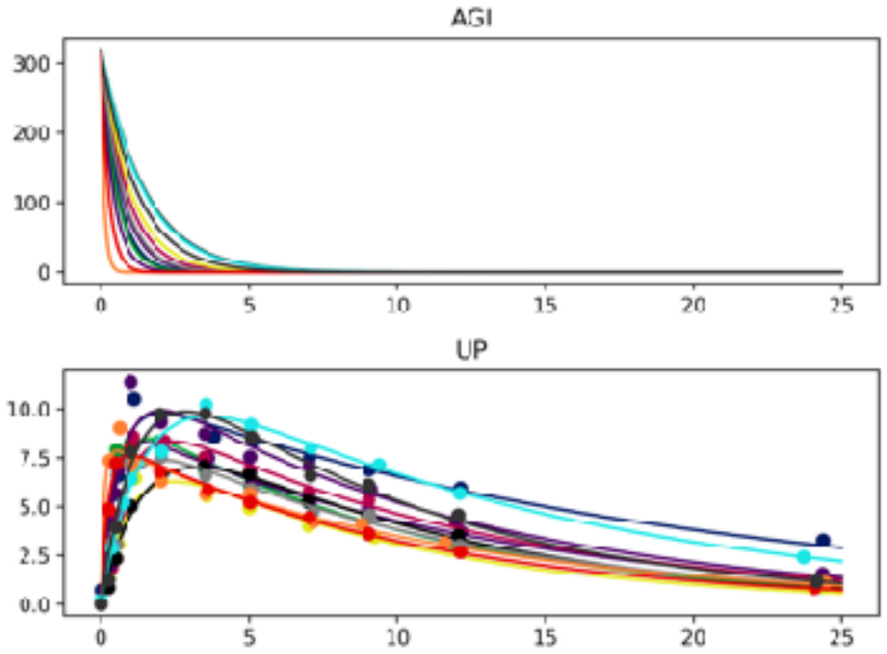
- Real data



Parameters distribution
Assuming Gaussian law



- Estimation




```
python3 script_estimate_PK_with_real_data.py
```

Population approach

- The mixed-effect approach consists of pooling all the patients together and estimating a global distribution of the model parameters in the population.

B. Ribba et al. A review of mixed-effects models of tumor growth and effects of anticancer drug treatment used in population analysis. CPT: Pharmacometrics & Systems Pharmacology, 3(5):1–10, 2014.

Population approach

- The mixed-effect approach consists of pooling all the patients together and estimating a global distribution of the model parameters in the population.

- More precisely, the individual parameters are assumed to be realizations of a random variable decomposed into two parts:

$$\theta_i = \theta^{pop} + \theta_i^{ind},\ i \in [1, N_P]$$

- where N_P is the number of considered patients, θ^{pop} correspond to the fixed effects where as θ_i^{ind} correspond to the random effects and have been assumed with mean zero.

B. Ribba et al. A review of mixed-effects models of tumor growth and effects of anticancer drug treatment used in population analysis. CPT: Pharmacometrics & Systems Pharmacology, 3(5):1–10, 2014.

Population approach

- The mixed-effect approach consists of pooling all the patients together and estimating a global distribution of the model parameters in the population.

- More precisely, the individual parameters are assumed to be realizations of a random variable decomposed into two parts:

$$\theta_i = \theta^{pop} + \theta_i^{ind},\ i \in [1, N_P]$$

- where N_P is the number of considered patients, θ^{pop} correspond to the fixed effects where as θ_i^{ind} correspond to the random effects and have been assumed with mean zero.

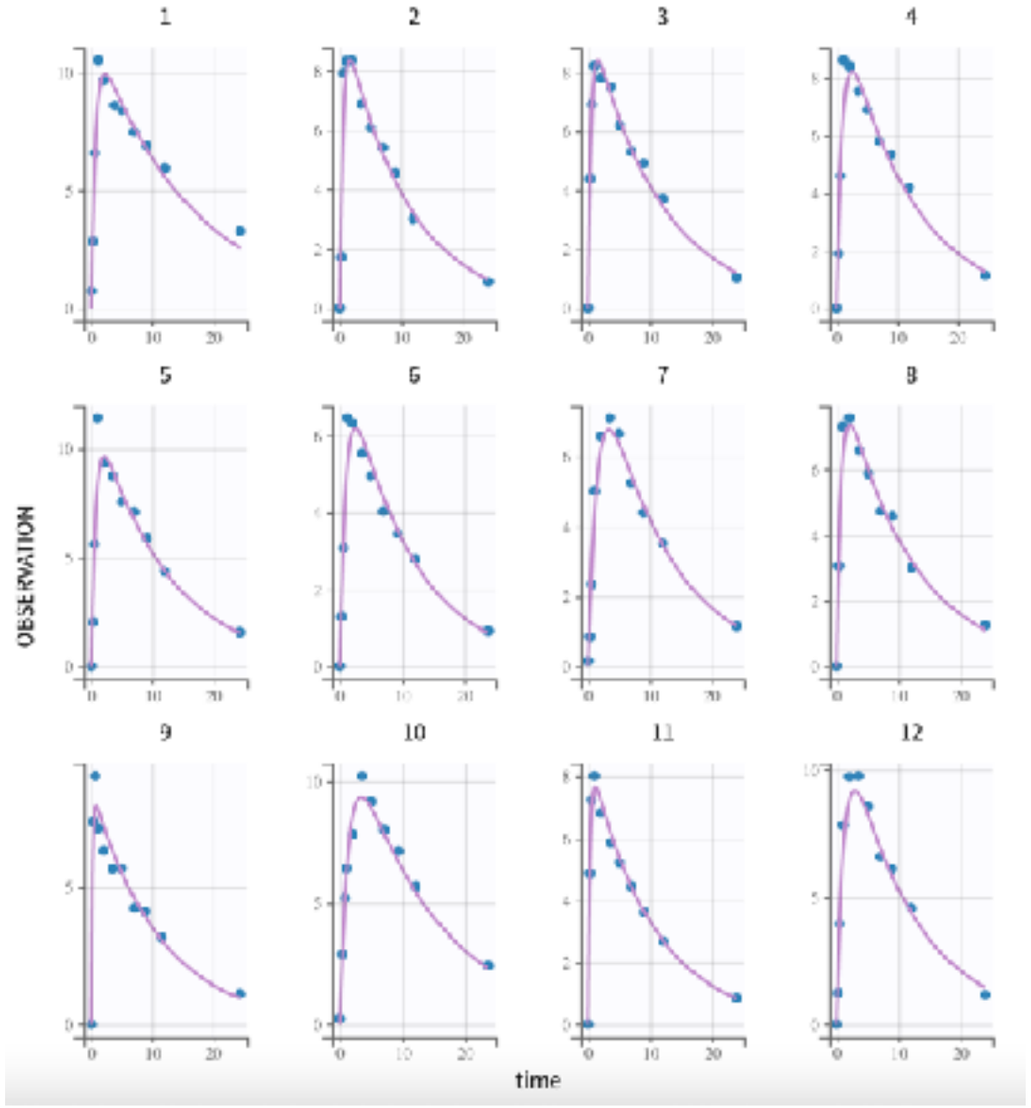
- Objective: A population approach allows to compensate for sparse sampling times and measurement uncertainties by constraining the variability of the parameters in the population.

- Classical algorithm: SAEM algorithm

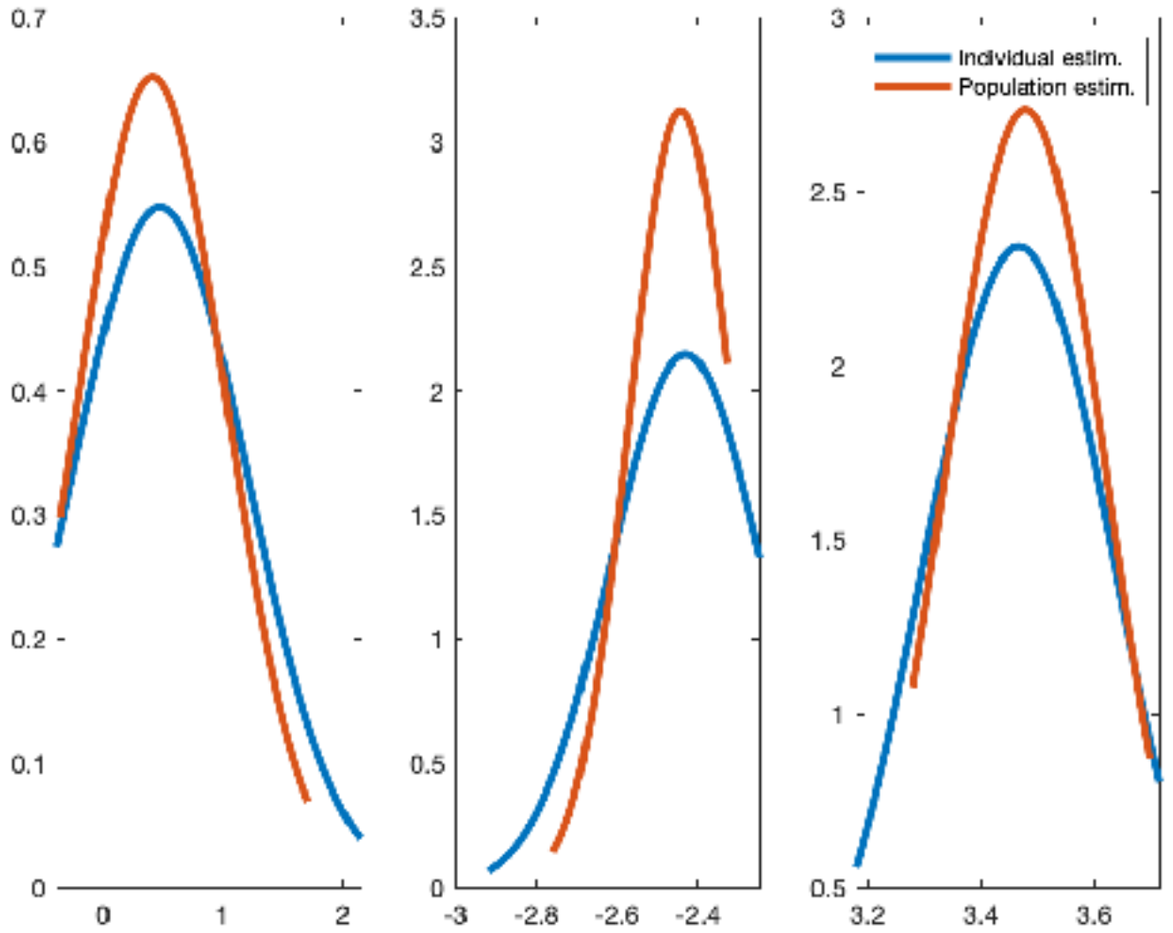
B. Ribba et al. A review of mixed-effects models of tumor growth and effects of anticancer drug treatment used in population analysis. CPT: Pharmacometrics & Systems Pharmacology, 3(5):1–10, 2014.

Population approach

- Illustration sur les données réelles
- Utilisation du logiciel *Monolix*

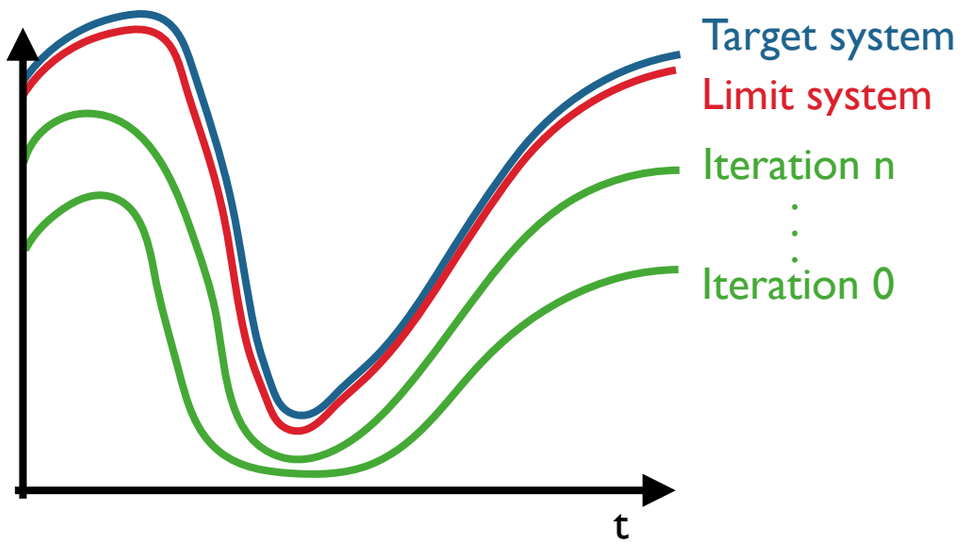


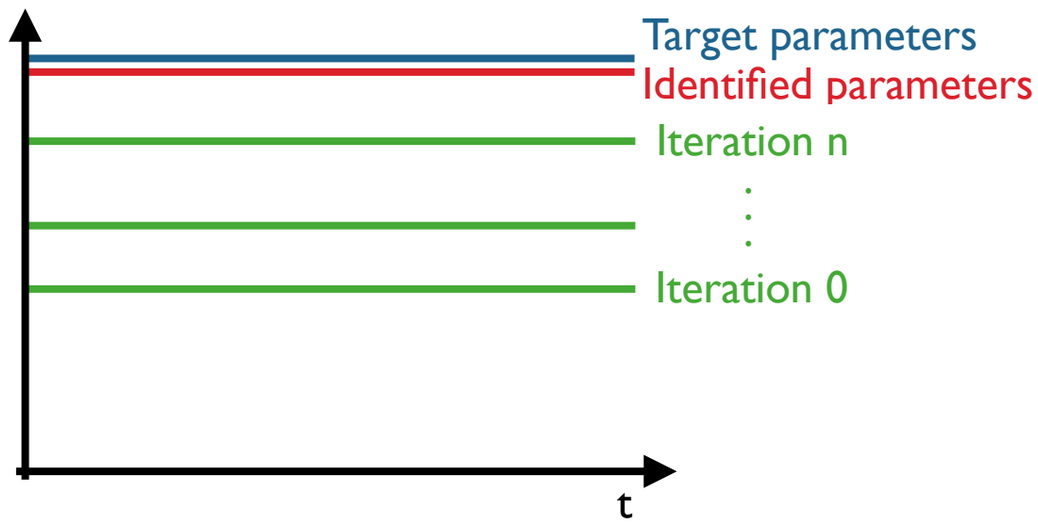
Parameters distribution
Assuming Gaussian law



Few conclusion on variational strategies

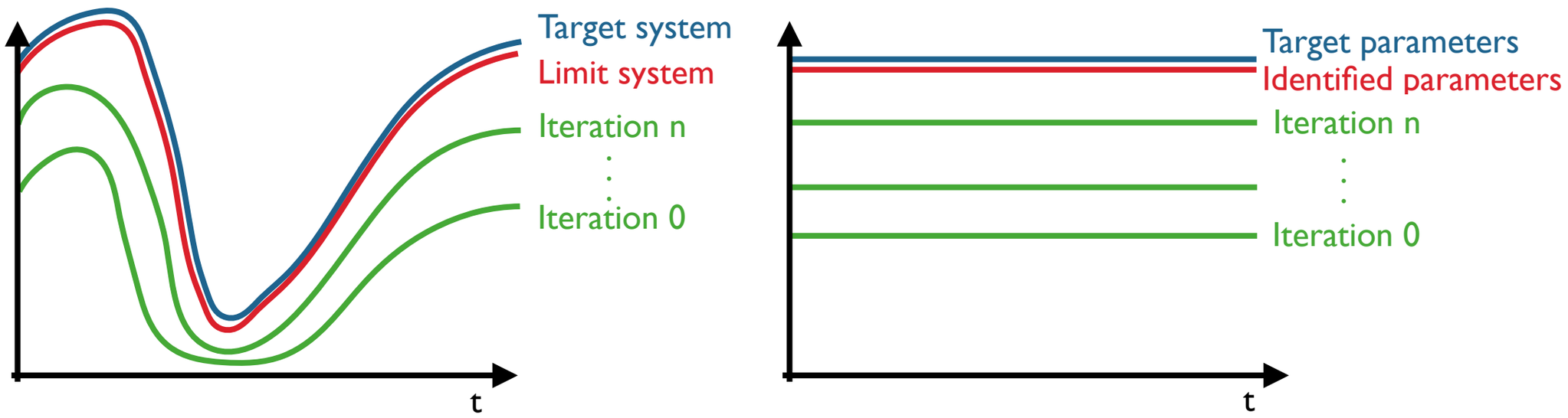
- Limitations:
  - Importance of the priors
    [if local extrema?]
  - Increase errors
    [Practical identifiability?]
  - Possible to also estimate standard deviations
  - Computational expensive for use with PDE
    [ok for small 1D … not for 2D or 3D]
  - There is no perfect method!
    [Need to be chosen depending on the system and on your question!]





Few conclusion on variational strategies

- Limitations:
  - Importance of the priors
    [if local extrema?]
  - Increase errors
    [Practical identifiability?]
  - Possible to also estimate standard deviations
  - Computational expensive for use with PDE
    [ok for small 1D … not for 2D or 3D]
  - There is no perfect method!
    [Need to be chosen depending on the system and on your question!]



- Strategies of optimization can be used in various other fields as machine and deep learning …

Sequential strategies

Formalism
State Observer - Luenberger Filter
State Observer - Linear Kalman Filter
Parameter and State Observer
Reduced Order Kalman Filter
Joint State (L) & Parameters (K) Observer

Sequential strategies

Formalism

State Observer - Luenberger Filter

State Observer - Linear Kalman Filter

Parameter and State Observer - Kalman Filter

Reduced Order Kalman Filter

Joint State (L) & Parameters (K) Observer

Sequential approach

- Correct the dynamics by a feedback based on the discrepancy combining the data and the model state
- And parameters have a dynamics too !

$$\begin{cases} \dot{u}(t) &= A(u,\theta,t) \\ u(0) &= u_\diamond + \zeta^u \\ \theta(0) &= \theta_\diamond + \zeta^\theta \end{cases} \Longrightarrow \begin{cases} \dot{\hat{u}}(t) &= A(\hat{u},\hat{\theta},t) + G^u(D(z,\hat{u})) \\ \dot{\hat{\theta}}(t) &= G^\theta(D(z,\hat{u})) \\ \hat{u}(0) &= u_\diamond \\ \hat{\theta}(0) &= \theta_\diamond \end{cases}$$

Objective: find G^u and G^θ, such that

$$(\hat{u},\theta) \longrightarrow_{t\to\infty} (u,\theta)$$

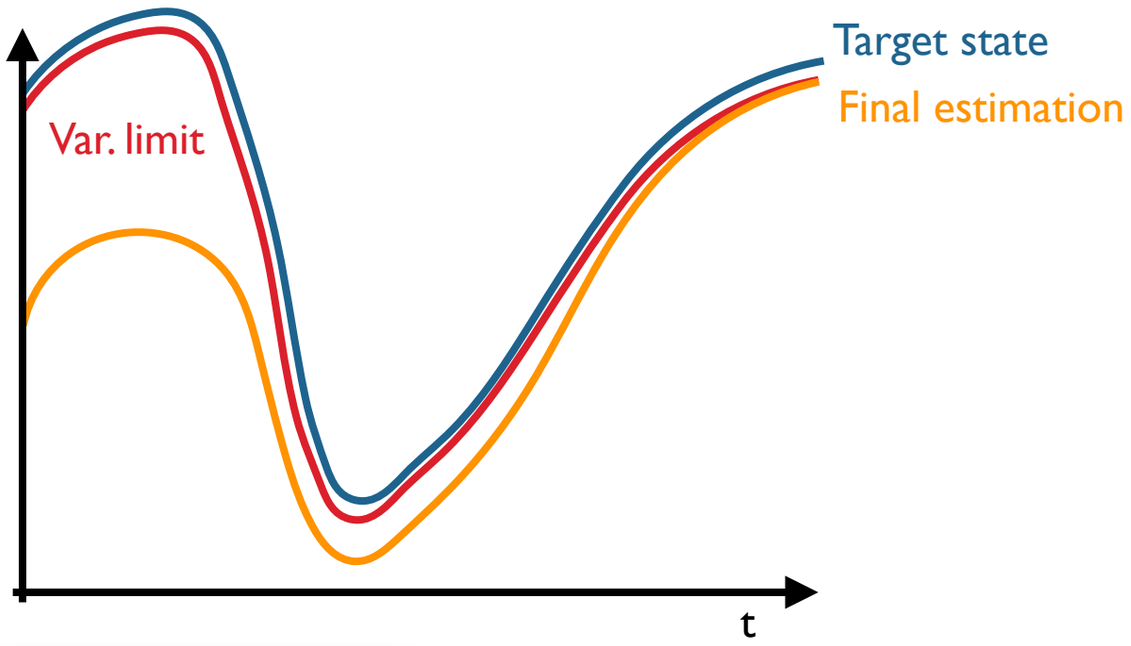
Sequential approach

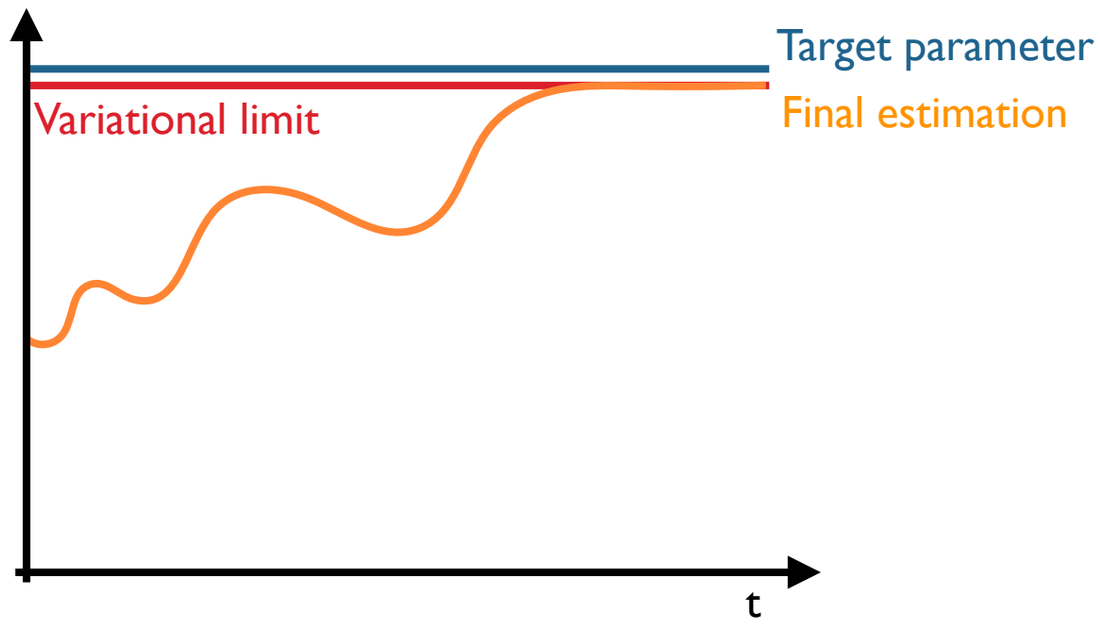
- Correct the dynamics by a feedback based on the discrepancy combining the data and the model state
- And parameters have a dynamics too !

$$\begin{cases} \dot{u}(t) &= A(u,\theta,t) \\ u(0) &= u_\diamond + \zeta^u \\ \theta(0) &= \theta_\diamond + \zeta^\theta \end{cases} \Longrightarrow \begin{cases} \dot{\hat{u}}(t) &= A(\hat{u},\hat{\theta},t) + G^u(D(z,\hat{u})) \\ \dot{\hat{\theta}}(t) &= G^\theta(D(z,\hat{u})) \\ \hat{u}(0) &= u_\diamond \\ \hat{\theta}(0) &= \theta_\diamond \end{cases}$$

Objective: find G^u and G^θ, such that

$$(\hat{u},\theta) \longrightarrow_{t\to\infty} (u,\theta)$$





Sequential strategies

State observer - Luenberger observer

- To introduce the principle of sequential method, we will assume that the uncertainties are limited to the initial conditions …

Target model

$$\begin{cases} \dot{A}_{GI} &= -k_a A_{GI} \\ \dot{U}_P &= \frac{k_a}{V_0} A_{GI} - k_e U_P \\ A_{GI}(0) &= A_{GI,\diamond} + \zeta^{A_{GI}} \\ U_P(0) &= U_{P,\diamond} + \zeta^{U_P} \end{cases}$$

Freerun model

$$\begin{cases} \dot{\hat{A}}_{GI} &= -k_a \hat{A}_{GI} \\ \dot{\hat{U}}_P &= \frac{k_a}{V_0} \hat{A}_{GI} - k_e \hat{U}_P \\ \hat{A}_{GI}(0) &= A_{GI,\diamond} \\ \hat{U}_P(0) &= U_{P,\diamond} \end{cases}$$

State observer - Luenberger observer

- To introduce the principle of sequential method, we will assume that the uncertainties are limited to the initial conditions …

Target model

$$\begin{cases} \dot{A}_{GI} &= -k_a A_{GI} \\ \dot{U}_P &= \frac{k_a}{V_0} A_{GI} - k_e U_P \\ A_{GI}(0) &= A_{GI,\diamond} + \zeta^{A_{GI}} \\ U_P(0) &= U_{P,\diamond} + \zeta^{U_P} \end{cases}$$

Freerun model

$$\begin{cases} \dot{\hat{A}}_{GI} &= -k_a \hat{A}_{GI} \\ \dot{\hat{U}}_P &= \frac{k_a}{V_0} \hat{A}_{GI} - k_e \hat{U}_P \\ \hat{A}_{GI}(0) &= A_{GI,\diamond} \\ \hat{U}_P(0) &= U_{P,\diamond} \end{cases}$$

Error model

$$\tilde{A}_{GI} = A_{GI} - \hat{A}_{GI},\ \tilde{U}_P = U_P - \hat{U}_P$$

$$\begin{cases} \dot{\tilde{A}}_{GI} &= -k_a \tilde{A}_{GI} \\ \dot{\tilde{U}}_P &= \frac{k_a}{V_0} \tilde{A}_{GI} - k_e \tilde{U}_P \\ \tilde{A}_{GI}(0) &= \zeta^{A_{GI}} \\ \tilde{U}_P(0) &= \zeta^{U_P} \end{cases}$$

State observer - Luenberger observer

- To introduce the principle of sequential method, we will assume that the uncertainties are limited to the initial conditions …

Target model

$$\begin{cases} \dot{A}_{GI} &= -k_a A_{GI} \\ \dot{U}_P &= \frac{k_a}{V_0} A_{GI} - k_e U_P \\ A_{GI}(0) &= A_{GI,\diamond} + \zeta^{A_{GI}} \\ U_P(0) &= U_{P,\diamond} + \zeta^{U_P} \end{cases}$$

Freerun model

$$\begin{cases} \dot{\hat{A}}_{GI} &= -k_a \hat{A}_{GI} \\ \dot{\hat{U}}_P &= \frac{k_a}{V_0} \hat{A}_{GI} - k_e \hat{U}_P \\ \hat{A}_{GI}(0) &= A_{GI,\diamond} \\ \hat{U}_P(0) &= U_{P,\diamond} \end{cases}$$

Error model

$$\tilde{A}_{GI} = A_{GI} - \hat{A}_{GI},\ \tilde{U}_P = U_P - \hat{U}_P$$

$$\begin{cases} \dot{\tilde{A}}_{GI} &= -k_a \tilde{A}_{GI} \\ \dot{\tilde{U}}_P &= \frac{k_a}{V_0} \tilde{A}_{GI} - k_e \tilde{U}_P \\ \tilde{A}_{GI}(0) &= \zeta^{A_{GI}} \\ \tilde{U}_P(0) &= \zeta^{U_P} \end{cases}$$

The solution of the error model is given by

$$\begin{pmatrix} \tilde{A}_{GI} \\ \tilde{U}_P \end{pmatrix} = P^{-1} \begin{pmatrix} e^{-k_a t} & 0 \\ 0 & e^{-k_e t} \end{pmatrix} P \begin{pmatrix} \zeta^{A_{GI}} \\ \zeta^{U_P} \end{pmatrix}.$$

State observer - Luenberger observer

- To introduce the principle of sequential method, we will assume that the uncertainties are limited to the initial conditions …

Target model

$$\begin{cases} \dot{A}_{GI} &= -k_a A_{GI} \\ \dot{U}_P &= \frac{k_a}{V_0} A_{GI} - k_e U_P \\ A_{GI}(0) &= A_{GI,\diamond} + \zeta^{A_{GI}} \\ U_P(0) &= U_{P,\diamond} + \zeta^{U_P} \end{cases}$$

Freerun model

$$\begin{cases} \dot{\hat{A}}_{GI} &= -k_a \hat{A}_{GI} \\ \dot{\hat{U}}_P &= \frac{k_a}{V_0} \hat{A}_{GI} - k_e \hat{U}_P \\ \hat{A}_{GI}(0) &= A_{GI,\diamond} \\ \hat{U}_P(0) &= U_{P,\diamond} \end{cases}$$

Error model

$\tilde{A}_{GI} = A_{GI} - \hat{A}_{GI},\ \tilde{U}_P = U_P - \hat{U}_P$

$$\begin{cases} \dot{\tilde{A}}_{GI} &= -k_a \tilde{A}_{GI} \\ \dot{\tilde{U}}_P &= \frac{k_a}{V_0} \tilde{A}_{GI} - k_e \tilde{U}_P \\ \tilde{A}_{GI}(0) &= \zeta^{A_{GI}} \\ \tilde{U}_P(0) &= \zeta^{U_P} \end{cases}$$

The solution of the error model is given by

$$\begin{pmatrix} \tilde{A}_{GI} \\ \tilde{U}_P \end{pmatrix} = P^{-1} \begin{pmatrix} e^{-k_a t} & 0 \\ 0 & e^{-k_e t} \end{pmatrix} P \begin{pmatrix} \zeta^{A_{GI}} \\ \zeta^{U_P} \end{pmatrix}.$$

The solution of the error model converges exponentially to 0.

Objective: find a filter which increases this convergence using the observations $z = U_P$.

State observer - Luenberger observer

- To introduce the principle of sequential method, we will assume that the uncertainties are limited to the initial conditions …

Target model

$$\begin{cases} \dot{A}_{GI} &= -k_a A_{GI} \\ \dot{U}_P &= \frac{k_a}{V_0} A_{GI} - k_e U_P \\ A_{GI}(0) &= A_{GI,\diamond} + \zeta^{A_{GI}} \\ U_P(0) &= U_{P,\diamond} + \zeta^{U_P} \end{cases}$$

Observer model

$$\begin{cases} \dot{\hat{A}}_{GI} &= -k_a \hat{A}_{GI} \\ \dot{\hat{U}}_P &= \frac{k_a}{V_0} \hat{A}_{GI} - k_e \hat{U}_P - \lambda(\hat{U}_P - z) \\ \hat{A}_{GI}(0) &= A_{GI,\diamond} \\ \hat{U}_P(0) &= U_{P,\diamond} \end{cases}$$

$z = U_P$

Error model

$\tilde{A}_{GI} = A_{GI} - \hat{A}_{GI},\ \tilde{U}_P = U_P - \hat{U}_P$

$$\begin{cases} \dot{\tilde{A}}_{GI} &= -k_a \tilde{A}_{GI} \\ \dot{\tilde{U}}_P &= \frac{k_a}{V_0} \tilde{A}_{GI} - (k_e + \lambda)\tilde{U}_P \\ \tilde{A}_{GI}(0) &= \zeta^{A_{GI}} \\ \tilde{U}_P(0) &= \zeta^{U_P} \end{cases}$$

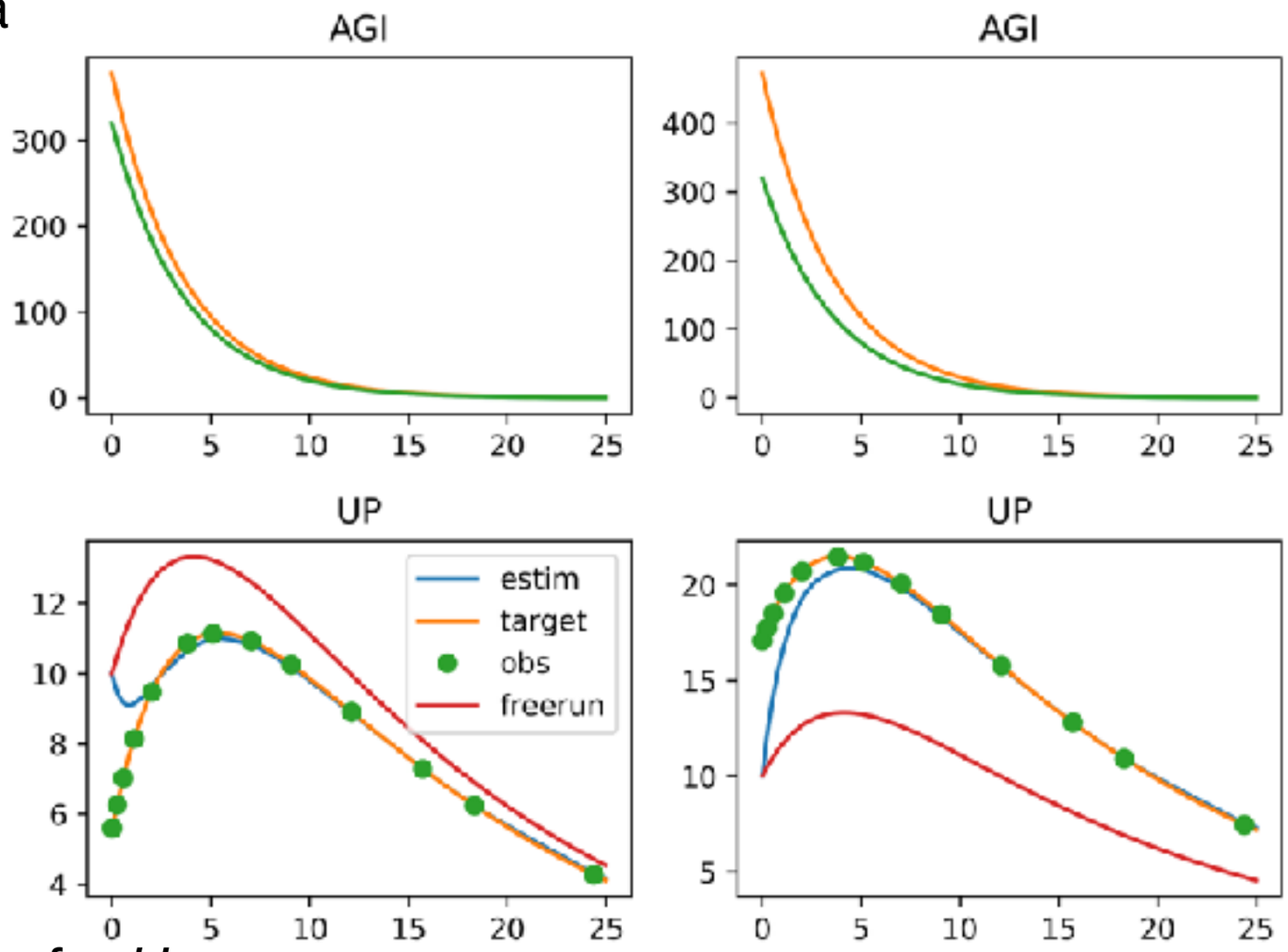
The solution of the error model is given by

$$\begin{pmatrix} \tilde{A}_{GI} \\ \tilde{U}_P \end{pmatrix} = P^{-1} \begin{pmatrix} e^{-k_a t} & 0 \\ 0 & e^{-(k_e+\lambda)t} \end{pmatrix} P \begin{pmatrix} \zeta^{A_{GI}} \\ \zeta^{U_P} \end{pmatrix}.$$

The solution of the error model quicker converges exponentially to 0 since $\lambda > 0$.

State observer - Luenberger observer

- Numerical illustrations on 2 patients without noise
- Interpolated data



- Satisfying results for U_P
  - the blue curve (observer model) converges to the orange ones (target model)
  - the error between the solutions of the observer and the target models is smaller than the error between the solutions of the freerun and target models (third column)
  - displacement of the eigenvalue associated to U_P

```
python3 script_estimate_PK_with_synthetic_data_state_observer.py
```

State observer - Luenberger observer

- Can we do better?
- Modify A_{GI}?

Target model

$$\begin{cases} \dot{A}_{GI} &= -k_a A_{GI} \\ \dot{U}_P &= \frac{k_a}{V_0} A_{GI} - k_e U_P \\ A_{GI}(0) &= A_{GI,\diamond} + \zeta^{A_{GI}} \\ U_P(0) &= U_{P,\diamond} + \zeta^{U_P} \end{cases}$$

Observer model

$$\begin{cases} \dot{\hat{A}}_{GI} &= -k_a \hat{A}_{GI} - \mu(\hat{U}_P - z) \\ \dot{\hat{U}}_P &= \frac{k_a}{V_0} \hat{A}_{GI} - k_e \hat{U}_P - \lambda(\hat{U}_P - z) \\ \hat{A}_{GI}(0) &= A_{GI,\diamond} \\ \hat{U}_P(0) &= U_{P,\diamond} \end{cases}$$

$z = U_P$

Error model

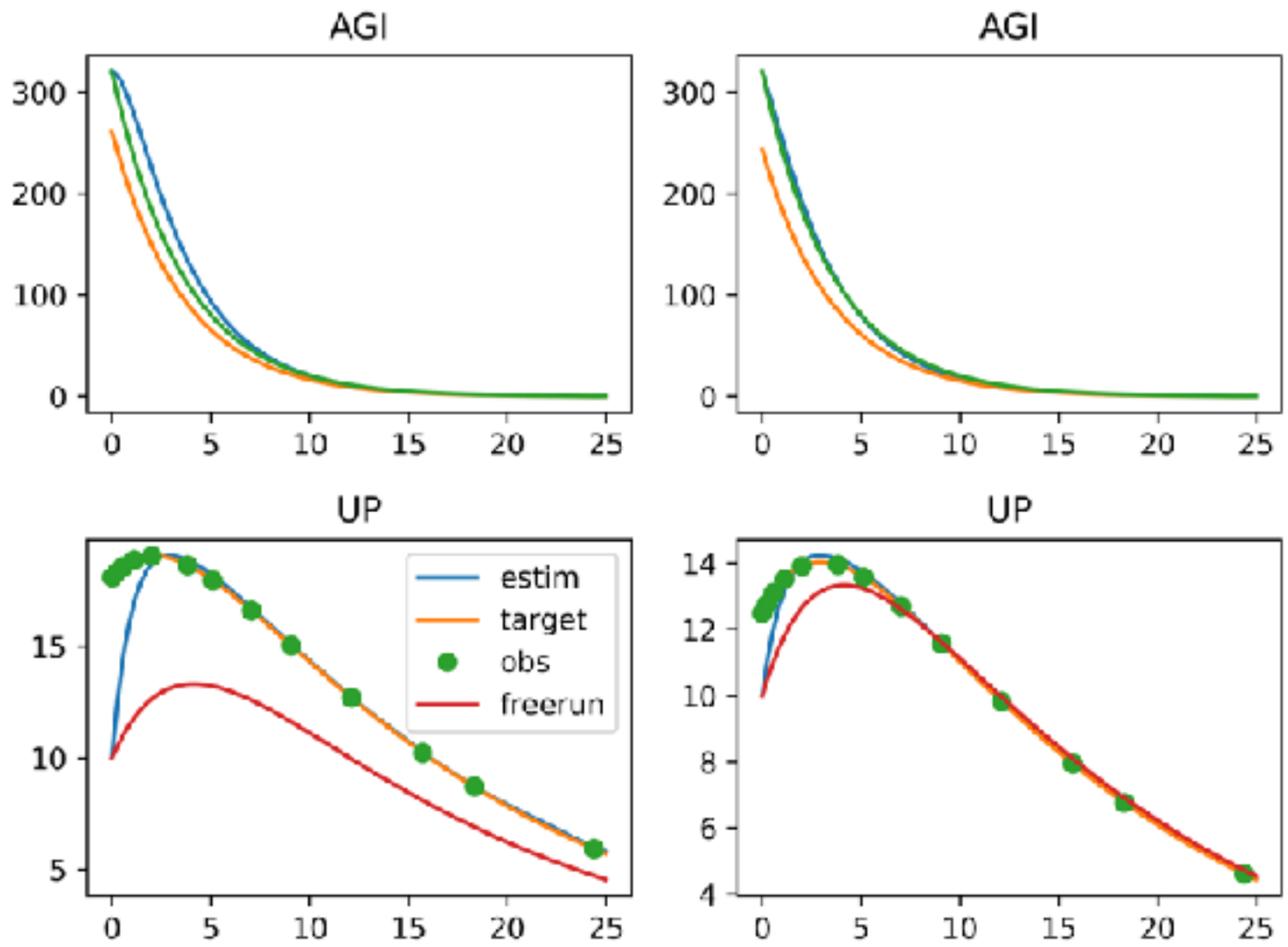
$$\tilde{A}_{GI} = A_{GI} - \hat{A}_{GI},\ \tilde{U}_P = U_P - \hat{U}_P$$

$$\begin{cases} \dot{\tilde{A}}_{GI} &= -k_a \tilde{A}_{GI} - \mu \tilde{U}_P \\ \dot{\tilde{U}}_P &= \frac{k_a}{V_0} \tilde{A}_{GI} - (k_e + \lambda)\tilde{U}_P \\ \tilde{A}_{GI}(0) &= \zeta^{A_{GI}} \\ \tilde{U}_P(0) &= \zeta^{U_P} \end{cases}$$

The eigenvalues depend on the values of the parameters
and can be complex.
We do not improve the convergence for all values of the parameters …

State observer - Luenberger observer

- Comment line 40 and uncomment line 41 of the file: params_PK_synthetic_data_state_observer



- Not so easy to conclude :)

State observer - Luenberger observer

- The time loop of the state observer is given in analyze_state_nudging_observer:

```python
def analyze_state_nudging_observer(par,IC,theta,uobs):
    """
    Explicit Euler
    """
    Nit = int(float(par['time_simu'])/float(par['dt']))+1
    t = np.linspace(0,float(par['time_simu']),Nit)

    # Initialization
    u = np.zeros((IC.size,Nit))

    u[:,0] = IC

    # Interpolate OBSERVATIONS
    z = np.interp(t, np.array(par['time_obs']), uobs)

    # Time loop
    for i in range(1,Nit):
        # Prediction scheme
        u[:,i] = u[:,i-1] + float(par['dt'])*eval(par['model'] + "(u[:,i-1],t[i-1],theta)")
        # Correction scheme
        u[:,i] = u[:,i] + float(par['dt'])*eval(par['nudging_state_observer'] + "(u[:,i],z[i],par['gain_parameter'])")

    return [t,u]
```

Only one line has been added (very simple to implement !)

```python
def PK_state_observer(u,z,gain):
    """
    State observer
    """
```

State observer - Luenberger observer

- To conclude …
  - This kind of observer/filter is called a *nudging* observer. This idea – originally introduced by Luenberger in 1971 – relies on the definition of the simplest possible filter such that the error between the observed trajectory and the observer system tends to zero.
  - Very simple to implement …
  - Very interesting in terms of computational cost
  - **But** has to be adapted to each model
  - **But** can be difficult to prove that the error tends to zero …
  - **But** can be difficult to fix the gain parameters (which corresponds to the confidence that we have in the data as we will see later …)
  - **But** can not be statically interpreted
  - **But** does not allow to estimate parameters …

D.G. Luenberger. An introduction to observers. IEEE Transactions on Automatic Control, 16:596–602, 1971.

Sequential strategies

State observer - Linear Kalman filter

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}\zeta) = \frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|\zeta})\|^2_{W^{-1}}\, dt \;\; \text{s.t} \;\; \left\{ \begin{array}{rcl} \dot{\bar{u}}_{|\zeta}(t) & = & A(\bar{u}_{|\zeta}, t) \\ \bar{u}_{|\zeta}(0) & = & u_\diamond + \zeta \end{array} \right.$$

State observer - Linear Kalman filter

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}(\zeta) = \frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|\zeta})\|^2_{W^{-1}}\, dt \;\; \text{s.t} \;\; \left\{ \begin{array}{rcl} \dot{\bar{u}}_{|\zeta}(t) & = & A(\bar{u}_{|\zeta}, t) \\ \bar{u}_{|\zeta}(0) & = & u_\diamond + \zeta \end{array} \right.$$

- A and C are supposed to be **linear**
- We consider a time discretization (just to simplify the presentation!)

$$\bar{u}_{k+1} = A_{k+1|k}\bar{u}_k, \quad \bar{u}_0 = u_\diamond + \zeta$$

State observer - Linear Kalman filter

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}\zeta) = \frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|\zeta})\|^2_{W^{-1}}\, dt \;\; \text{s.t} \;\; \left\{ \begin{array}{rcl} \dot{\bar{u}}_{|\zeta}(t) & = & A(\bar{u}_{|\zeta}, t) \\ \bar{u}_{|\zeta}(0) & = & u_\diamond + \zeta \end{array} \right.$$

- A and C are supposed to be **linear**
- We consider a time discretization (just to simplify the presentation!)

$$\bar{u}_{k+1} = A_{k+1|k}\bar{u}_k, \quad \bar{u}_0 = u_\diamond + \zeta$$

The following functional

$$\mathcal{J}_n(\zeta) = \frac{1}{2}\left\langle \zeta, P_0^{-1}\zeta \right\rangle + \sum_{k=0}^{n} \left\langle z_k - C_k\bar{u}_k, W^{-1}(z_k - C_k\bar{u}_k) \right\rangle$$

admits one and only one minimizer (quadratic functional) denoted $\tilde{\zeta}$ obtained when $\mathrm{d}_\zeta \mathcal{J}_n(\tilde{\zeta}) = 0$. The derivative of $\mathcal{J}_n(\zeta)$ with respect to ζ is given by

$$\mathrm{d}_\zeta \mathcal{J}_n(\zeta)(\mathrm{d}\zeta) = \frac{1}{2}\left\langle \zeta, P_0^{-1}\mathrm{d}\zeta \right\rangle - \frac{1}{2}\sum_{k=0}^{n} \left\langle z_k - C_k\bar{u}_k, W^{-1}C_k\mathrm{d}_\zeta\bar{u}_k(\mathrm{d}\zeta) \right\rangle .$$

State observer - Linear Kalman filter

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}\zeta) = \frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|\zeta})\|^2_{W^{-1}}\, dt \;\; \text{s.t} \;\; \left\{ \begin{array}{rcl} \dot{\bar{u}}_{|\zeta}(t) & = & A(\bar{u}_{|\zeta}, t) \\ \bar{u}_{|\zeta}(0) & = & u_\diamond + \zeta \end{array} \right.$$

For a trajectory $(\bar{u}_k)_k$ we define $(q_k)_k$ the associated adjoint variable

$$q_k - A^T_{k+1|k} q_{k+1} = C_k^T W^{-1}(z_k - C_k \bar{u}_k),$$
$$q_{n+1} = 0.$$

We obtain that $\tilde{\zeta} = \min_\zeta \mathcal{J}(\zeta) = P_0 q_0$.

$$\begin{aligned}
\mathrm{d}_\zeta \mathcal{J}_n(\zeta)(\mathrm{d}\zeta) &= \left\langle \zeta, P_0^{-1}\mathrm{d}\zeta \right\rangle - \sum_{k=0}^{n} \left\langle z_k - C_k\bar{u}_k, W^{-1} C_k \mathrm{d}_\zeta \bar{u}_k(\mathrm{d}\zeta) \right\rangle \\
&= \left\langle \zeta, P_0^{-1}\mathrm{d}\zeta \right\rangle - \sum_{k=0}^{n} \left\langle C_k^T W^{-1}(z_k - C_k\bar{u}_k), \mathrm{d}_\zeta \bar{u}_k(\mathrm{d}\zeta) \right\rangle \\
&= \left\langle \zeta, P_0^{-1}\mathrm{d}\zeta \right\rangle + \sum_{k=0}^{n} \left\langle -(q_k - A^T_{k+1|k} q_{k+1}), \mathrm{d}_\zeta \bar{u}_k(\mathrm{d}\zeta) \right\rangle \\
&= \left\langle \zeta, P_0^{-1}\mathrm{d}\zeta \right\rangle - \sum_{k=0}^{n} \langle q_k, \mathrm{d}_\zeta \bar{u}_k(\mathrm{d}\zeta) \rangle + \sum_{k=0}^{n} \left\langle A^T_{k+1|k} q_{k+1}, \mathrm{d}_\zeta \bar{u}_k(\mathrm{d}\zeta) \right\rangle \\
&= \left\langle \zeta, P_0^{-1}\mathrm{d}\zeta \right\rangle - \sum_{k=0}^{n} \langle q_k, \mathrm{d}_\zeta \bar{u}_k(\mathrm{d}\zeta) \rangle + \sum_{k=0}^{n} \left\langle q_{k+1}, A_{k+1|k} \mathrm{d}_\zeta \bar{u}_k(\mathrm{d}\zeta) \right\rangle \\
&= \left\langle \zeta, P_0^{-1}\mathrm{d}\zeta \right\rangle - \sum_{k=0}^{n} \langle q_k, \mathrm{d}_\zeta \bar{u}_k(\mathrm{d}\zeta) \rangle + \sum_{k=0}^{n} \langle q_{k+1}, \mathrm{d}_\zeta \bar{u}_{k+1}(\mathrm{d}\zeta) \rangle \\
&= \left\langle \zeta, P_0^{-1}\mathrm{d}\zeta \right\rangle - \langle q_0, \mathrm{d}_\zeta \bar{u}_0(\mathrm{d}\zeta) \rangle + \langle q_{n+1}, \mathrm{d}_\zeta \bar{u}_{n+1}(\mathrm{d}\zeta) \rangle \\
&= \left\langle \zeta, P_0^{-1}\mathrm{d}\zeta \right\rangle - \langle q_0, \mathrm{d}\zeta \rangle
\end{aligned}$$

State observer - Linear Kalman filter

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}\zeta) = \frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|\zeta})\|^2_{W^{-1}}\, dt \;\; \text{s.t} \;\; \left\{ \begin{array}{rcl} \dot{\bar{u}}_{|\zeta}(t) & = & A(\bar{u}_{|\zeta}, t) \\ \bar{u}_{|\zeta}(0) & = & u_\diamond + \zeta \end{array} \right.$$

- **Theorem** (Time-discrete version) - Defining (optimal trajectory) the solution of the *two-ends* problem,

$$\tilde{u}_{k+1} = A_{k+1|k}\tilde{u}_k, \quad \tilde{u}_0 = u_\diamond + P_0\tilde{q}_0,$$

$$\tilde{q}_k = A^T_{k+1|k}\tilde{q}_{k+1} + C_k^T W^{-1}(z_k - C_k\tilde{u}_k), \quad \tilde{q}_{n+1} = 0,$$

we have the following identity $\tilde{u}_k = \hat{u}_k + P_k\tilde{q}_k$, where $(\hat{u}_k, P_k)_k$ is defined by

$$\left\{ \begin{array}{l} \hat{u}_0^- = u_\diamond \\ \hat{u}_k^+ = \hat{u}_k^- + K_k(z_k - C_k\hat{u}_k^-) \\ \hat{u}_{k+1}^- = A_{k+1|k}\hat{u}_k^+ \\ P_0^- = P_0 \\ P_k^+ = P_k^- - K_k C_k P_k^- \\ P_{k+1}^- = A_{k+1|k}P_k^+ A^T_{k+1|k}, \end{array} \right.$$

with $K_k = P_k^- C_k^T (C_k P_k^- C_k^T + W)^{-1}$ is the *Kalman* gain.

State observer - Linear Kalman filter

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}\zeta) = \frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|\zeta})\|^2_{W^{-1}}\, dt \;\; \text{s.t} \;\; \left\{ \begin{array}{rcl} \dot{\bar{u}}_{|\zeta}(t) & = & A(\bar{u}_{|\zeta}, t) \\ \bar{u}_{|\zeta}(0) & = & u_\diamond + \zeta \end{array} \right.$$

- **Theorem** (Time-discrete version) - Defining (optimal trajectory) the solution of the *two-ends* problem,

$$\tilde{u}_{k+1} = A_{k+1|k}\tilde{u}_k, \quad \tilde{u}_0 = u_\diamond + P_0\tilde{q}_0,$$

$$\tilde{q}_k = A^T_{k+1|k}\tilde{q}_{k+1} + C_k^T W^{-1}(z_k - C_k\tilde{u}_k), \quad \tilde{q}_{n+1} = 0,$$

we have the following identity $\tilde{u}_k = \hat{u}_k + P_k\tilde{q}_k$, where $(\hat{u}_k, P_k)_k$ is defined by

$$\left\{ \begin{array}{l} \hat{u}_0^- = u_\diamond \\ \hat{u}_k^+ = \hat{u}_k^- + K_k(z_k - C_k\hat{u}_k^-) \\ \hat{u}_{k+1}^- = A_{k+1|k}\hat{u}_k^+ \\ P_0^- = P_0 \\ P_k^+ = P_k^- - K_k C_k P_k^- \\ P_{k+1}^- = A_{k+1|k}P_k^+ A^T_{k+1|k}, \end{array} \right.$$

with $K_k = P_k^- C_k^T (C_k P_k^- C_k^T + W)^{-1}$ is the *Kalman* gain.

R. Kalman and R. Bucy. New results in linear filtering and prediction theory. Trans. ASME J. Basic. Eng., 83:95—108, 1961.

State observer - Linear Kalman filter

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}\zeta) = \frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|\zeta})\|^2_W \, dt \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|\zeta}(t) &= A(\bar{u}_{|\zeta}, t) \\ \bar{u}_{|\zeta}(0) &= u_\diamond + \zeta \end{cases}$$

State observer - Linear Kalman filter

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}\zeta) = \frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|\zeta})\|^2_W \, dt \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|\zeta}(t) &= A(\bar{u}_{|\zeta}, t) \\ \bar{u}_{|\zeta}(0) &= u_\diamond + \zeta \end{cases}$$

- Proof by induction:
  - $\tilde{u}_0 = u_\diamond + P_0\tilde{q}_0 = \hat{u}_0 + P_0\tilde{q}_0$
  - Let us assume that the formula is satisfied for some k, $\tilde{u}_k = \hat{u}_k + P_k\tilde{q}_k$, we have

State observer - Linear Kalman filter

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}\zeta) = \frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|\zeta})\|^2_W \, dt \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|\zeta}(t) &= A(\bar{u}_{|\zeta}, t) \\ \bar{u}_{|\zeta}(0) &= u_\diamond + \zeta \end{cases}$$

- Proof by induction:
  - $\tilde{u}_0 = u_\diamond + P_0\tilde{q}_0 = \hat{u}_0 + P_0\tilde{q}_0$
  - Let us assume that the formula is satisfied for some k, $\tilde{u}_k = \hat{u}_k + P_k\tilde{q}_k$, we have

$$\begin{aligned}
\tilde{u}_{k+1} &= A_{k+1|k}\tilde{u}_k = A_{k+1|k}(\hat{u}_k + P_k\tilde{q}_k) \\
&= A_{k+1|k}(\hat{u}_k^+ - K_k(z_k - C_k\hat{u}_k^-) + (P_k^+ + K_kC_kP_k^-)\tilde{q}_k) \\
&= \underbrace{A_{k+1|k}\hat{u}_k^+}_{\hat{u}_{k+1}^-} - A_{k+1|k}K_k(z_k - C_k\underbrace{(\hat{u}_k^- + P_k^-\tilde{q}_k)}_{\tilde{u}_k}) + A_{k+1|k}P_k^+ \underbrace{\tilde{q}_k}_{A^T_{k+1|k}\tilde{q}_{k+1} + A_{k+1|k}C_k^TW^{-1}(z_k - C_k\tilde{u}_k)} \\
&= \hat{u}_{k+1}^- - A_{k+1|k}K_k(z_k - C_k\tilde{u}_k) + \underbrace{A_{k+1|k}P_k^+A_{k+1|k}^T}_{P_{k+1}^-}\tilde{q}_{k+1} + A_{k+1|k}P_k^+C_k^TW^{-1}(z_k - C_k\tilde{u}_k) \\
&= \hat{u}_{k+1}^- + P_{k+1}^-\tilde{q}_{k+1} - A_{k+1|k}(K_k - P_k^+C_k^TW^{-1})(z_k - C_k\tilde{u}_k).
\end{aligned}$$

State observer - Linear Kalman filter

- Minimize the criterion (for example maximum likelihood estimation with Gaussian law) with respect to the uncertainties under the constraint of the model dynamics

$$\min_{\zeta} \mathcal{J}\zeta) = \frac{1}{2}\|\zeta\|^2_{P_0^{-1}} + \frac{1}{2}\int_0^T \|z - C(\bar{u}_{|\zeta})\|^2_W \, dt \quad \text{s.t} \quad \begin{cases} \dot{\bar{u}}_{|\zeta}(t) &= A(\bar{u}_{|\zeta}, t) \\ \bar{u}_{|\zeta}(0) &= u_\diamond + \zeta \end{cases}$$

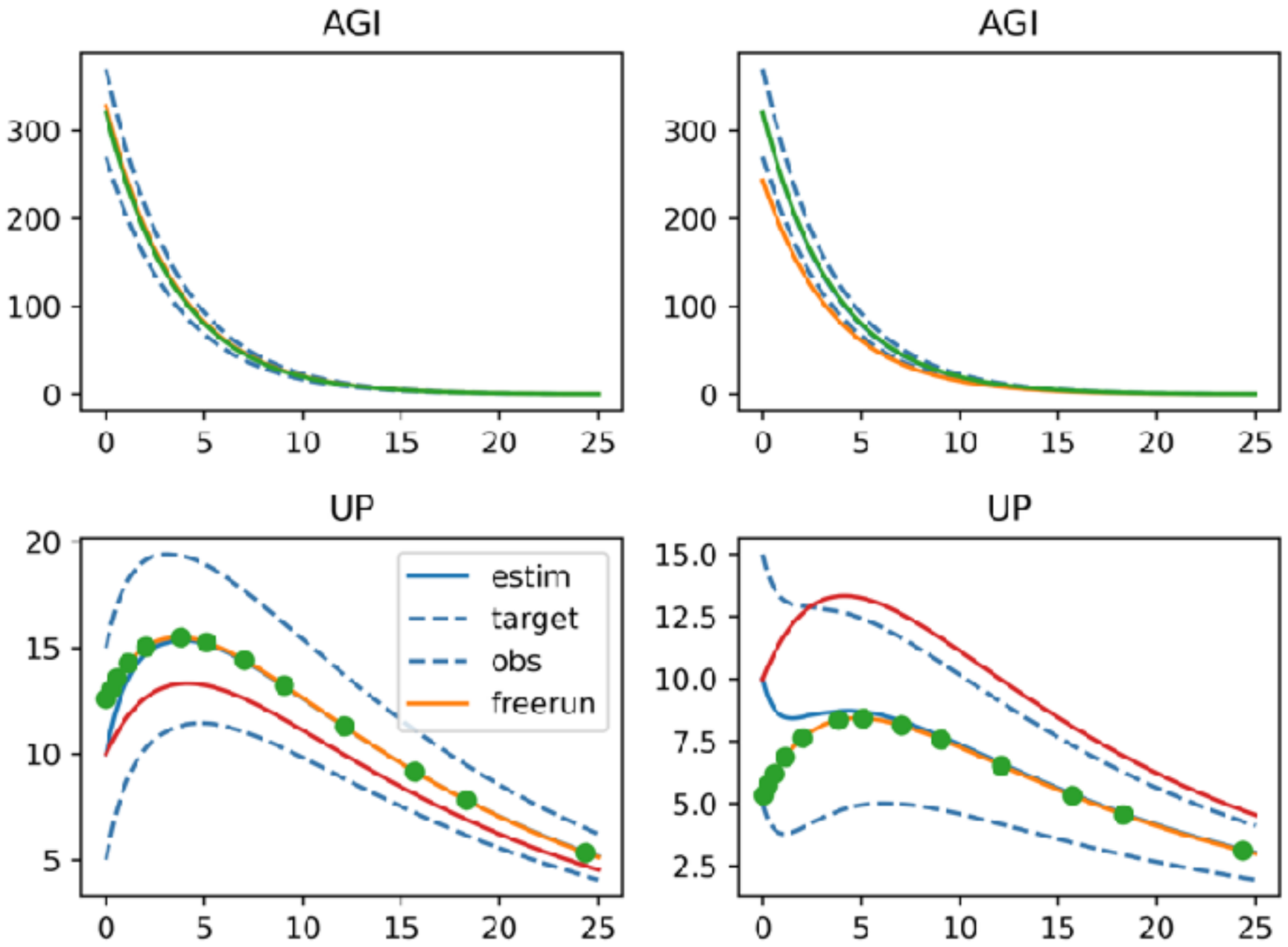
- Proof by induction:
  - $\tilde{u}_0 = u_\diamond + P_0\tilde{q}_0 = \hat{u}_0 + P_0\tilde{q}_0$
  - Let us assume that the formula is satisfied for some k, $\tilde{u}_k = \hat{u}_k + P_k\tilde{q}_k$, we have

$$\begin{aligned}
\tilde{u}_{k+1} &= A_{k+1|k}\tilde{u}_k = A_{k+1|k}(\hat{u}_k + P_k\tilde{q}_k) \\
&= A_{k+1|k}(\hat{u}_k^+ - K_k(z_k - C_k\hat{u}_k^-) + (P_k^+ + K_kC_kP_k^-)\tilde{q}_k) \\
&= \underbrace{A_{k+1|k}\hat{u}_k^+}_{\hat{u}_{k+1}^-} - A_{k+1|k}K_k(z_k - C_k\underbrace{(\hat{u}_k^- + P_k^-\tilde{q}_k)}_{\tilde{u}_k}) + A_{k+1|k}P_k^+ \underbrace{\tilde{q}_k}_{A^T_{k+1|k}\tilde{q}_{k+1} + A_{k+1|k}C_k^TW^{-1}(z_k - C_k\tilde{u}_k)} \\
&= \hat{u}_{k+1}^- - A_{k+1|k}K_k(z_k - C_k\tilde{u}_k) + \underbrace{A_{k+1|k}P_k^+A_{k+1|k}^T}_{P_{k+1}^-}\tilde{q}_{k+1} + A_{k+1|k}P_k^+C_k^TW^{-1}(z_k - C_k\tilde{u}_k) \\
&= \hat{u}_{k+1}^- + P_{k+1}^-\tilde{q}_{k+1} - A_{k+1|k}(K_k - P_k^+C_k^TW^{-1})(z_k - C_k\tilde{u}_k).
\end{aligned}$$

- We conclude by proving that $K_k - P_k^+C_k^TW^{-1} = 0$:

$$\begin{aligned}
K_k - P_k^+C_k^TW^{-1} &= K_k - (P_k^- - K_kC_kP_k^-)C_k^TW^{-1} = (K_k(W + C_kP_k^-C_k^T) - P_k^-C_k)W^{-1} \\
&= (P_k^-C_k^T(C_kP_kC_k^T + W)^{-1}(W + C_kP_k^-C_k^T) - P_kC_k)W^{-1} = 0.
\end{aligned}$$

State observer - Linear Kalman filter



- Satisfying results for U_P
  - the blue curve (observer model) converges to the orange ones (target model)

State observer - Linear Kalman filter

```python
def analyze_state_linear_kalman(par,IC,theta,W,P,uobs):
    """
    State observer Linear Kalman
    """
    # Time
    Nit = int(float(par['time_simu'])/float(par['dt']))+1
    t = np.linspace(0,float(par['time_simu']),Nit)

    # Observation operator
    C = eval(par['linear_observation_operator']+"()")
    A = eval(par['linear_model_operator']+"(par['dt'],theta)")

    # Initialization
    u = np.zeros((IC.size,Nit))
    u[:,0] = IC

    stdu = np.zeros((IC.size,Nit))
    stdu[:,0] = np.sqrt(np.diag(P))

    # Interpolate OBSERVATIONS
    z = np.interp(t, np.array(par['time_obs']), uobs)

    # Time loop
    for i in range(1,Nit):
        # Correction scheme
        K = my_multi_dot(P,np.transpose(C),my_inv(my_multi_dot(C,P,np.transpose(C))+W))
        u[:,i] = u[:,i-1] + my_dot(K,(z[i-1]-my_dot(C,u[:,i-1])))
        # Prediction scheme
        u[:,i] = np.dot(A,u[:,i])
        P = my_multi_dot(A, P, np.transpose(A))
        # Save std
        stdu[:,i] = np.sqrt(np.diag(P))

    return [t,u,stdu]
```

Matrix operations

State observers

- Study the stability:
  - Modify the observations time to consider time-sampled observations
  - Increase the value of the observations noise
  - Which filter seems to be the more sensitive to observations noise ?

- Specific to each filter:
  - For nudging filter, we can play with the valeurs of the nudging parameters. The more the noise is important the less the values of the parameters have to be … (We do not known how they are related to the deviation standard of the observations noise …)
  - For Kalman filter:

$$= P_0 = \begin{pmatrix} \sigma^2_{A_{GI}} & 0 \\ 0 & \sigma^2_{U_U} \end{pmatrix} \text{ with } A_{GI}(0) \sim \mathcal{N}(\mu_{A_{GI}}, \sigma_{A_{GI}})\ U_P(0) \sim \mathcal{N}(\mu_{U_P}, \sigma_{U_P})$$

$$= \frac{I}{dt} W \text{ with } z = U_P + \chi, \quad \chi \sim \mathcal{N}(0, W^{I/2})$$

Sequential strategies

Parameter and state observer

- The parameters can be treated as the state variable by augmenting the state dimension:

$$\begin{cases} \dot{u}(t) &= A(u,\theta,t) \\ u(0) &= u_\diamond + \zeta^u \\ \theta(0) &= \theta_\diamond + \zeta^\theta \end{cases} \Longrightarrow \begin{cases} {}^\alpha\dot{u}(t) &= {}^\alpha A({}^\alpha u, t) \\ {}^\alpha u(0) &= {}^\alpha u_\diamond + \zeta \end{cases}$$

$${}^\alpha A = \begin{pmatrix} A \\ 0 \end{pmatrix} \text{ (as } \dot{\theta} = 0\text{)}$$

$${}^\alpha u = \begin{pmatrix} u \\ \theta \end{pmatrix} \zeta = \begin{pmatrix} \zeta^u \\ \zeta^\theta \end{pmatrix} {}^\alpha u_\diamond = \begin{pmatrix} u_\diamond \\ \theta_\diamond \end{pmatrix}$$

Parameter and state observer

- The parameters can be treated as the state variable by augmenting the state dimension:

$$\begin{cases} \dot{u}(t) &= A(u,\theta,t) \\ u(0) &= u_\diamond + \zeta^u \\ \theta(0) &= \theta_\diamond + \zeta^\theta \end{cases} \Longrightarrow \begin{cases} {}^\alpha\dot{u}(t) &= {}^\alpha A({}^\alpha u, t) \\ {}^\alpha u(0) &= {}^\alpha u_\diamond + \zeta \end{cases}$$

$${}^\alpha A = \begin{pmatrix} A \\ 0 \end{pmatrix} \text{ (as } \dot{\theta} = 0\text{)}$$

$${}^\alpha u = \begin{pmatrix} u \\ \theta \end{pmatrix} \zeta = \begin{pmatrix} \zeta^u \\ \zeta^\theta \end{pmatrix} {}^\alpha u_\diamond = \begin{pmatrix} u_\diamond \\ \theta_\diamond \end{pmatrix}$$

- If ${}^\alpha A$ and ${}^\alpha C$ (the augmented observation operator) are linear, we can use the Kalman-Bucy filter presented previously

$$\begin{cases} {}^\alpha\hat{u}_0^- &= {}^\alpha u_\diamond \\ {}^\alpha\hat{u}_k^+ &= {}^\alpha\hat{u}_k^- + {}^\alpha K_k(z_k - {}^\alpha C_k\, {}^\alpha\hat{u}_k^-) \\ {}^\alpha\hat{u}_{k+1}^- &= {}^\alpha A_{k+1|k}\, {}^\alpha\hat{u}_k^+ \\ {}^\alpha P_0^- &= {}^\alpha P_0 \\ {}^\alpha P_k^+ &= {}^\alpha P_k^- - {}^\alpha K_k {}^\alpha C_k\, {}^\alpha P_k^- \\ {}^\alpha P_{k+1}^- &= {}^\alpha A_{k+1|k}\, {}^\alpha P_k^+ {}^\alpha A_{k+1|k}^T, \end{cases}$$

$$\zeta = \begin{pmatrix} \zeta^u \\ \zeta^\theta \end{pmatrix} \sim \mathcal{N}(0, {}^\alpha P_0^2)$$

with ${}^\alpha K_k = {}^\alpha P_k^- {}^\alpha C_k^T ({}^\alpha C_k\, {}^\alpha P_k^- {}^\alpha C_k^T + W)^{-1}$

Parameter and state observer

- And if it is not linear? Even the augmented dynamics of the very simple PK model is not linear:

$$\begin{cases} \dot{A}_{GI} &= -k_a A_{GI} \\ \dot{U}_P &= \frac{k_a}{V_0} A_{GI} - k_e U_P \\ \dot{k}_a &= 0 \\ \dot{k}_e &= 0 \\ \dot{V}_0 &= 0 \end{cases}$$

Parameter and state observer

- And if it is not linear? Even the augmented dynamics of the very simple PK model is not linear:

$$\begin{cases} \dot{A}_{GI} &= -k_a A_{GI} \\ \dot{U}_P &= \frac{k_a}{V_0} A_{GI} - k_e U_P \\ \dot{k}_a &= 0 \\ \dot{k}_e &= 0 \\ \dot{V}_0 &= 0 \end{cases}$$

- With non-linear model operator and observation operator, the optimal sequential estimator can still be defined but to the prize of a computational complexity for large dimensional systems.

Parameter and state observer

- And if it is not linear? Even the augmented dynamics of the very simple PK model is not linear:

$$\begin{cases} \dot{A}_{GI} &= -k_a A_{GI} \\ \dot{U}_P &= \frac{k_a}{V_0} A_{GI} - k_e U_P \\ \dot{k}_a &= 0 \\ \dot{k}_e &= 0 \\ \dot{V}_0 &= 0 \end{cases}$$

- With non-linear model operator and observation operator, the optimal sequential estimator can still be defined but to the prize of a computational complexity for large dimensional systems.

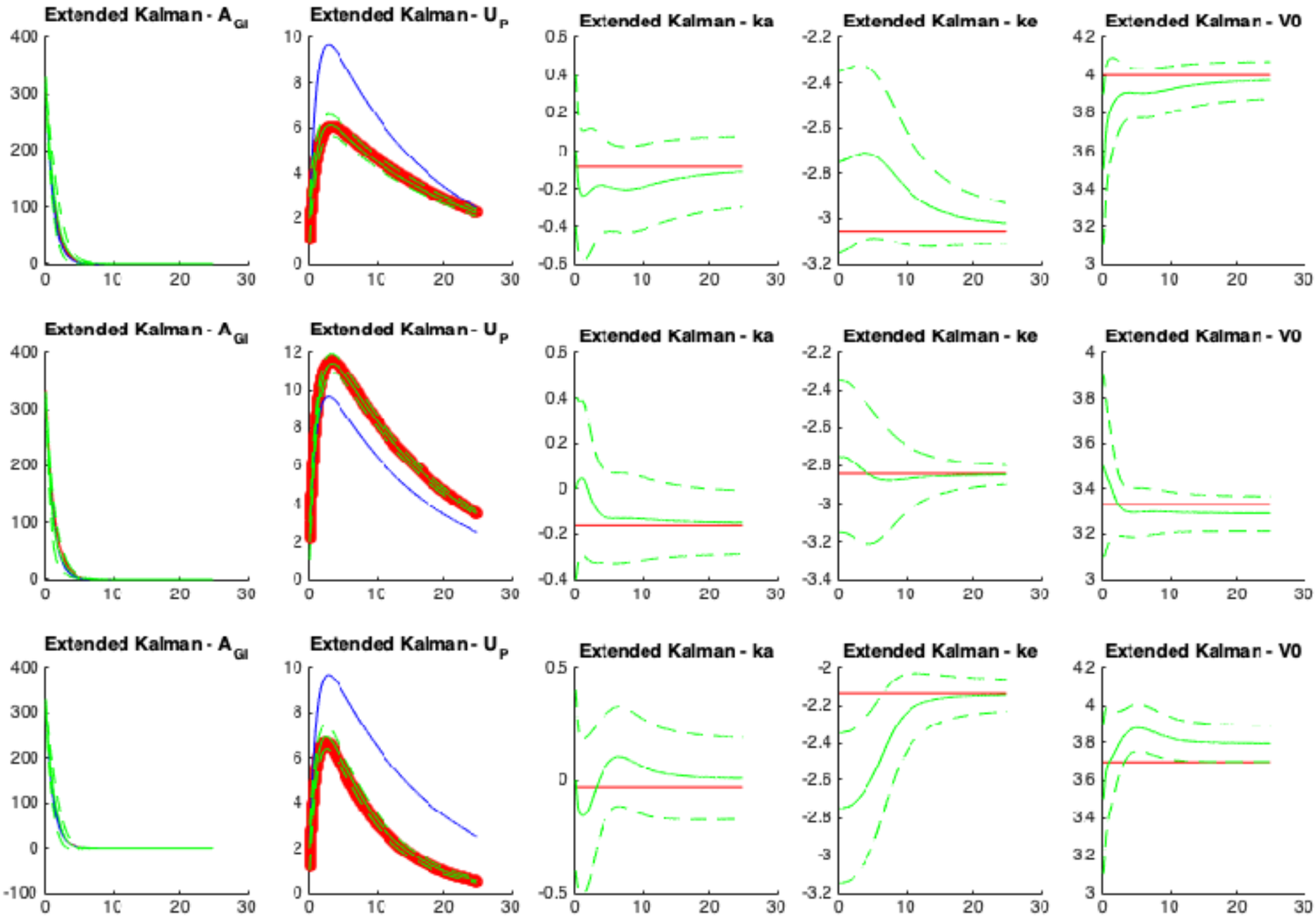
- Therefore, it is classical to rely on approximate optimal sequential estimator based on the generalization of the Kalman filter to non-linear operators. This is typically the case with the Extended Kalman filter where we have

$$\begin{cases} {}^{\alpha}\hat{u}_0^- &= {}^{\alpha}u_\diamond \\ {}^{\alpha}\hat{u}_k^+ &= {}^{\alpha}\hat{u}_k^- + {}^{\alpha}K_k\left(z_k - {}^{\alpha}C_k({}^{\alpha}\hat{u}_k^-)\right) \\ {}^{\alpha}\hat{u}_{k+1}^- &= {}^{\alpha}A_{k+1|k}({}^{\alpha}\hat{u}_k^+) \\ {}^{\alpha}P_0^- &= {}^{\alpha}P_0 \\ {}^{\alpha}P_k^+ &= {}^{\alpha}P_k^- - {}^{\alpha}K_k\,\mathrm{d}^{\alpha}C_k\,{}^{\alpha}P_k^- \\ {}^{\alpha}P_{k+1}^- &= \mathrm{d}^{\alpha}A_{k+1|k}\,{}^{\alpha}P_k^+\,\mathrm{d}^{\alpha}A_{k+1|k}^T, \end{cases}$$

Of course, it can also be used to estimate only the state

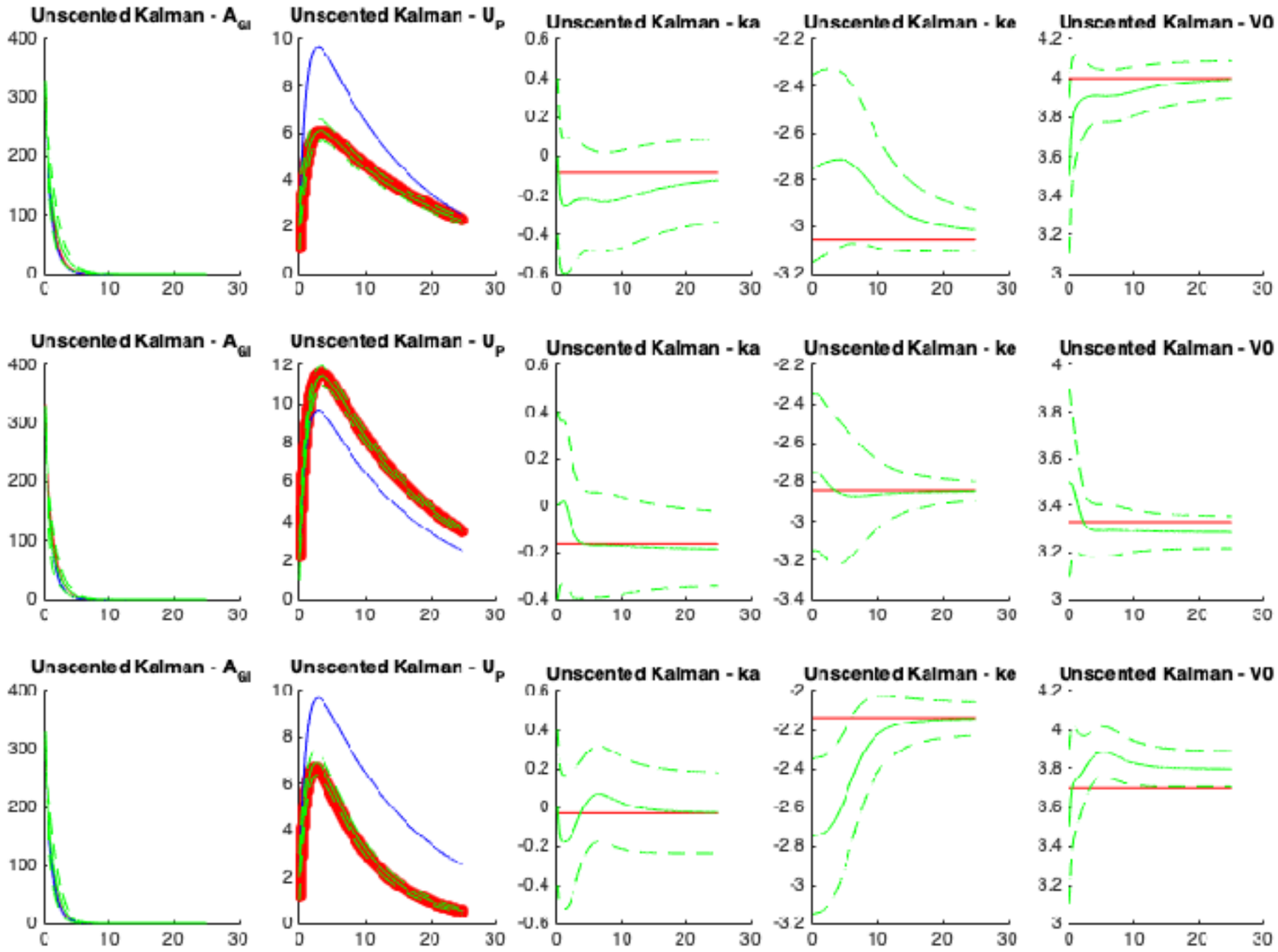
$$\text{with } {}^{\alpha}K_k = {}^{\alpha}P_k^-\,\mathrm{d}^{\alpha}C_k^T(\mathrm{d}^{\alpha}C_k\,{}^{\alpha}P_k^-\,\mathrm{d}^{\alpha}C_k^T + W)^{-1}$$

Parameter and state observer - EKF



Parameter and state observer - UKF

- It can be more convenient to rely on the Unscented-Kalman Filter where the tangent operators are replaced by finite difference approximations based on sampling points.



Conclusion on Kalman filters

- Can be used for **every** model: That is why this method is very interesting!
- Allow to estimate *state* and *parameters*
- Quite simple to implement
- Can be statically interpreted
- Is related (at least for the linear Kalman-Bucy filter) to a variational strategy

- **But** have an important drawback which is the **prohibitive computation** of the covariance matrix P. Indeed P is a full matrix of size $N_{\alpha_u} \times N_{\alpha_u}$.

Concerning PDE:
- Ok for 1D
- For small 2D (~100 x 100)
- Impossible for 3D …

Sequential strategies

Reduced Order Kalman Filter

- A possible strategy is to use reduced-order covariance operators.

- A typical example of order reduction occurs when we restrict the uncertainty space to the parameter space.

- The main idea behind the reduced order strategy is to consider a SVD decomposition of P of the form

$$P = LU^{-1}L$$

with U an invertible matrix of small size and L an extension operator.
For linear operators, this decomposition is stable over time and the equation on P leads to the two following systems with admissible computational times

$$\dot{L} = AL \text{ and } \dot{U} = L^T C^T W^{-1} C L$$

In non-linear cases, extensions of these two systems have been developed.

P. Moireau, D. Chapelle, and P. Le Tallec. Joint state and parameter estimation for distributed mechanical systems. Computer Methods in Applied Mechanics and Engineering, 197:659–677, 2008.

Sequential strategies

Formalism

State Observer - Luenberger Filter

State Observer - Linear Kalman Filter

Parameter and State Observer - Kalman Filter

Reduced Order Kalman Filter

Joint State (L) & Parameters (K) Observer

Joint state (L) - parameters (K) observer

- It is possible to combine the two sequential strategies in order to propose a joint state and parameters estimation with a reasonable computational time (for PDE).

- We use two different gains:
  - one for the state: a *Luenberger* observer,
  - and one for the parameters: a *Kalman* observer.

At each time step:

Prediction

$$\hat{u}_n^+ = A_{n+1|n}\hat{u}_n^-$$

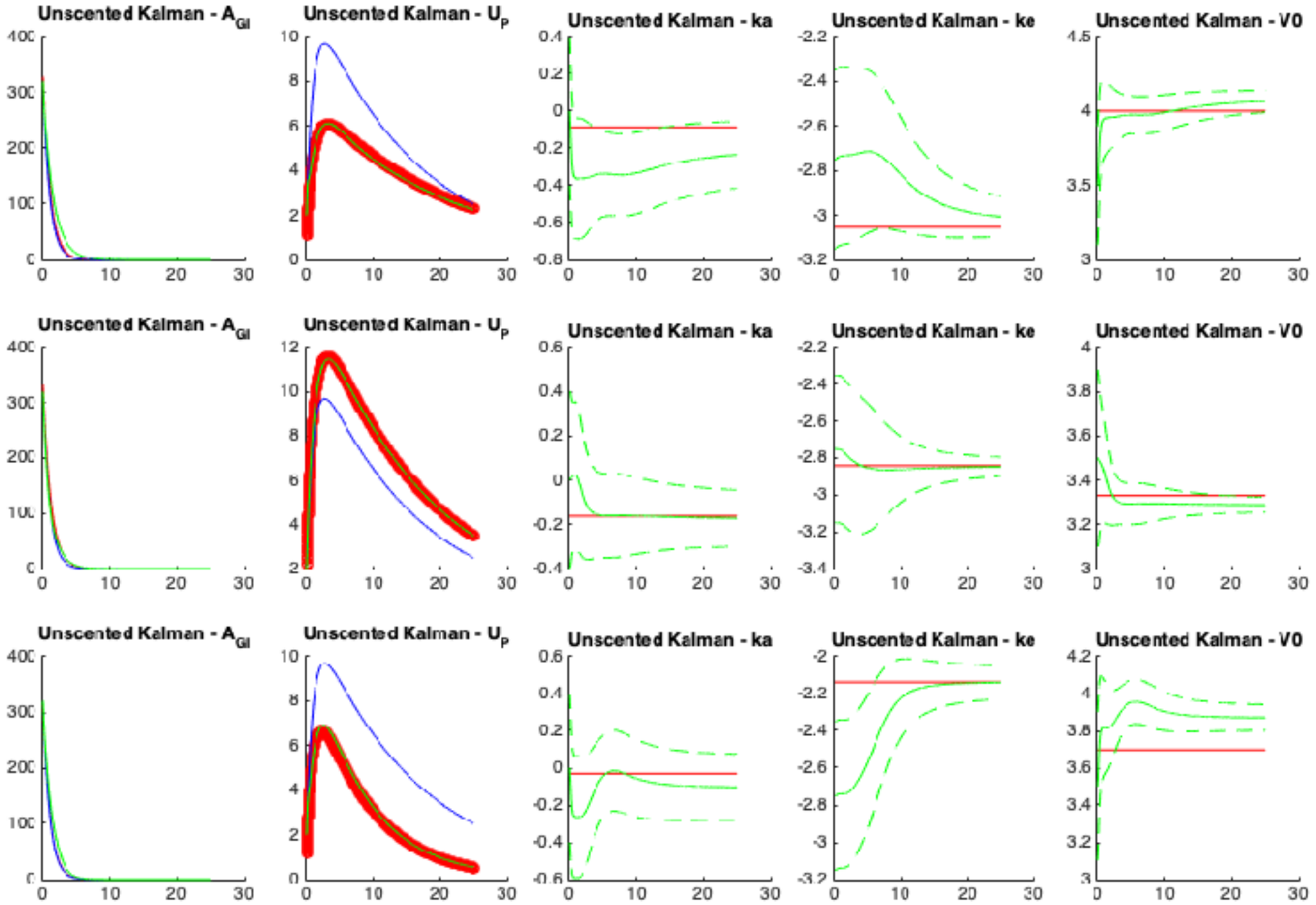
State correction

$$\hat{u}_n^{++} = G_L(\hat{u}_n^+, z_n)$$

Parameter correction

$$\hat{u}_{n+1}^- = G_{KR}^u(\hat{u}_n^{++}, z_n) \text{ and } \hat{\theta}_{n+1} = G_{KR}^\theta(\hat{\theta}_n)$$

Joint state (L) - parameters (K) observer



Parameter and state observers

- Study the stability
  - Modify the observations time to consider time-sampled observations
  - Increase the value of the observations noise
  - Which filter seems to be the more sensitive to observations noise ?

- Modify the parameters specific of the Kalman filters:

$$= P_0 = \begin{pmatrix} \sigma^2_{A_{GI}} & 0 \\ 0 & \sigma^2_{U_U} \end{pmatrix} \text{ with } A_{GI}(0) \sim \mathcal{N}(\mu_{A_{GI}}, \sigma_{A_{GI}}) \; U_P(0) \sim \mathcal{N}(\mu_{U_P}, \sigma_{U_P})$$

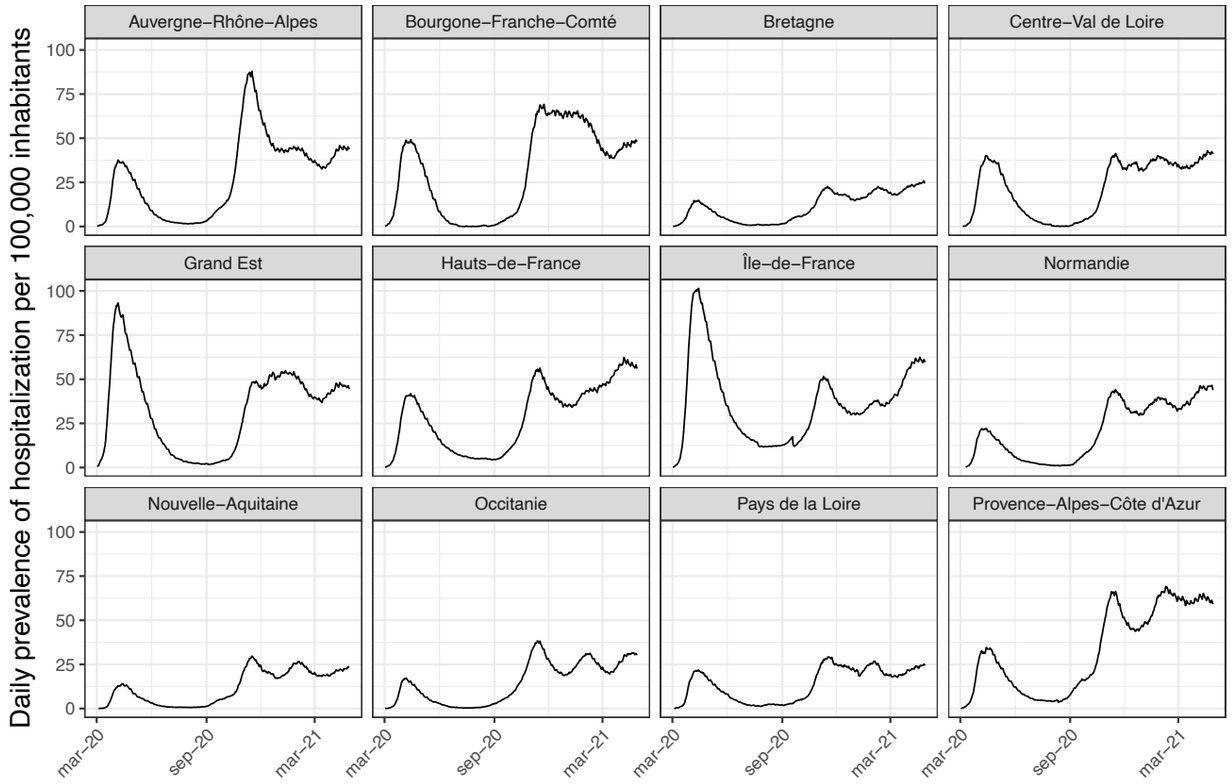
$$= P_0 = \begin{pmatrix} \sigma^2_{k_a} & 0 & 0 \\ 0 & \sigma^2_{k_e} & 0 \\ 0 & 0 & \sigma^2_{V_0} \end{pmatrix}$$

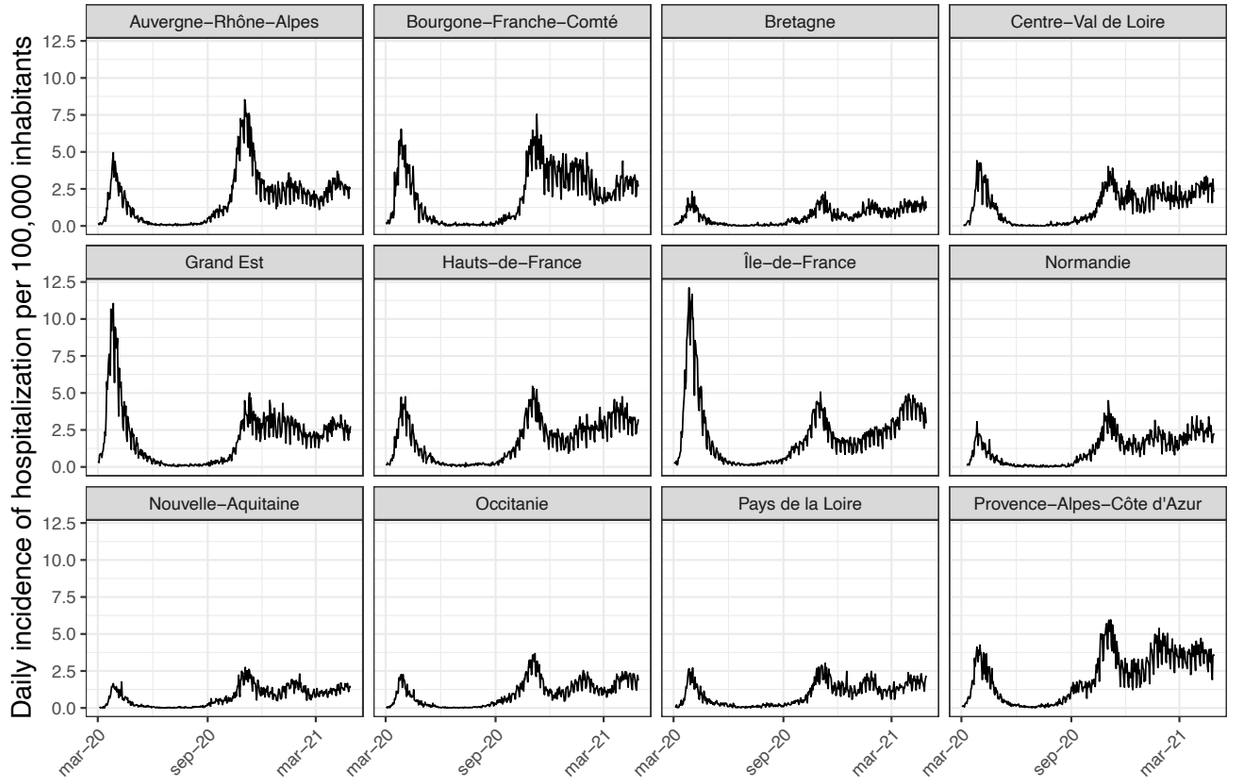
$$= \frac{1}{dt} W \text{ with } z = U_P + \chi, \quad \chi \sim \mathcal{N}(0, W^{1/2})$$

Illustrative example: Covid Crisis

Observations

- We take into account two data series :
  - Daily incident number of hospitalized
  - Hospitalization number

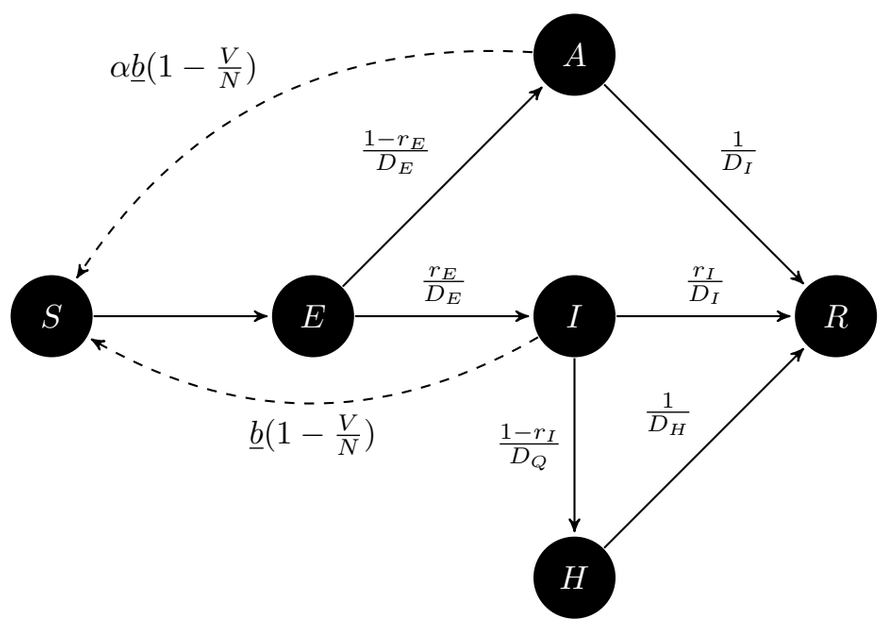




Observations

- The SEIRAH model: An extended Susceptible-Exposed-Infectious-Recovered (SEIR) model

$$
\begin{cases}
\dot{S} = -\underline{b}\left(1 - \frac{V}{N}\right)\frac{S(I+\alpha A)}{N} \\
\dot{E} = \underline{b}\left(1 - \frac{V}{N}\right)\frac{S(I+\alpha A)}{N} - \frac{E}{D_E} \\
\dot{I} = \frac{r_E}{D_E}E - \frac{1-r_I}{D_Q}I - \frac{r_I}{D_I}I, \\
\dot{R} = \frac{r_I I + A}{D_I} + \frac{H}{D_H} \\
\dot{A} = \frac{1-r_E}{D_E}E - \frac{A}{D_I} \\
\dot{H} = \frac{1-r_I}{D_Q}I - \frac{H}{D_H}.
\end{cases}
$$



Parameter	Interpretation	Value
b	Transmission rate of ascertained cases	Region Specific - **Estimated**
r_E	Ascertainment rate	0.844^*
r_I	Non hospitalized rate	0.966^*
α	Ratio of transmission between A and I	0.55^*
D_E	Latent (incubation) period (days)	5.1^*
D_I	Infectious period (days)	5^*
D_Q	Duration from I onset to H (days)	11-D_E $= 5.9^*$
D_H	Hospitalization period (days)	18.3^{**}
N	Population size	Region Specific

* Fixed with the literature
** Computed using the correlation between the data

Effective reproductive number / Attack rates

- To compute the **effective reproductive ratio**, we apply the *Next Generation Matrix approach* :

$$R_{eff}(t) = b(t)\frac{S(t)}{N}\left(D_i\alpha(1-r_e) + \frac{D_iD_qr_e}{(1-r_i)D_i + r_iD_q}\right).$$

By replacing $1 - r_E \sim \frac{A}{A+I}$, we obtain:

$$R_{eff} = b\frac{S}{N}\left(\alpha\frac{A}{A+I}D_I + \frac{D_QD_II}{(A+I)((1-r_I)D_I + r_ID_Q)}\right).$$

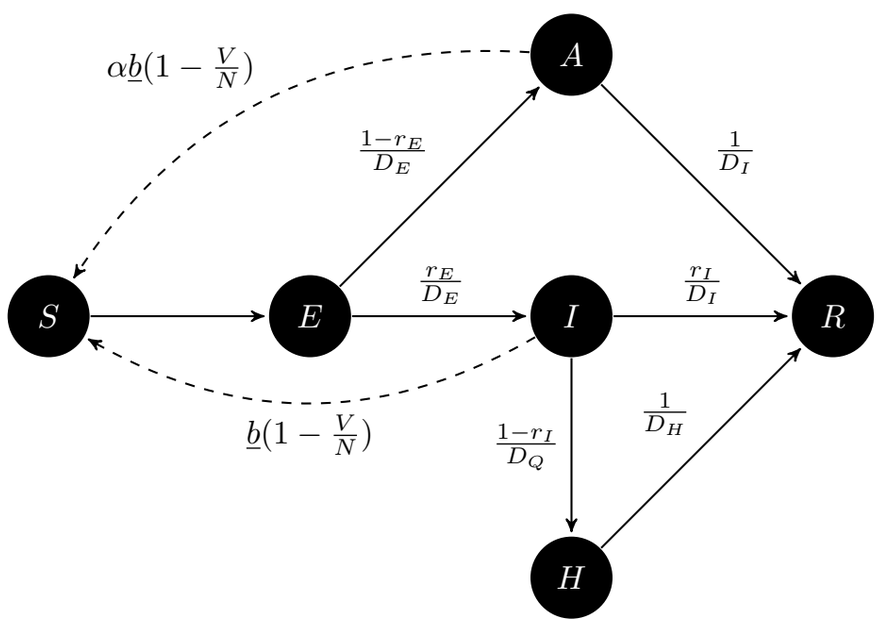
- The proportion of infected individuals - also called **attack rates** - among the population in each region at a given date is given by:

$$\frac{E+I+R+A+H}{N}.$$

Transmission rate

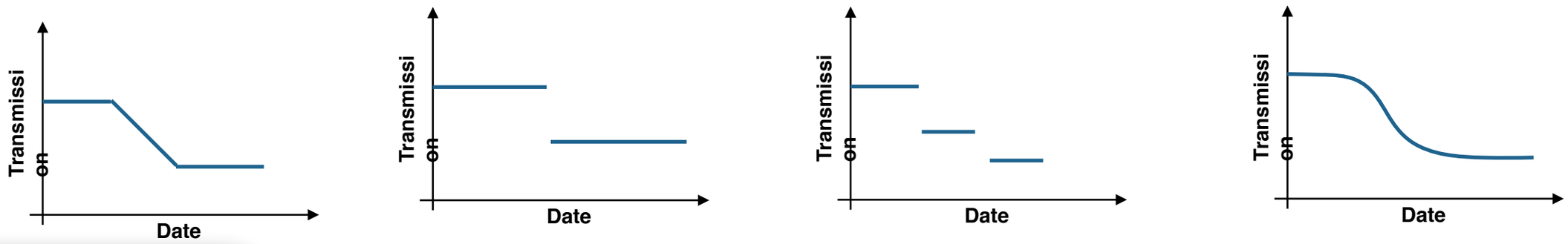
- We assume that non-pharmaceutical intervention (NPI) reduces the transmission b.

$$
\begin{cases}
\dot{S} = -\underline{b}\left(1 - \dfrac{V}{N}\right)\dfrac{S(I+\alpha A)}{N} \\
\dot{E} = \underline{b}\left(1 - \dfrac{V}{N}\right)\dfrac{S(I+\alpha A)}{N} - \dfrac{E}{D_E} \\
\dot{I} = \dfrac{r_E}{D_E}E - \dfrac{1-r_I}{D_Q}I - \dfrac{r_I}{D_I}I, \\
\dot{R} = \dfrac{r_I I + A}{D_I} + \dfrac{H}{D_H} \\
\dot{A} = \dfrac{1-r_E}{D_E}E - \dfrac{A}{D_I} \\
\dot{H} = \dfrac{1-r_I}{D_Q}I - \dfrac{H}{D_H}.
\end{cases}
$$

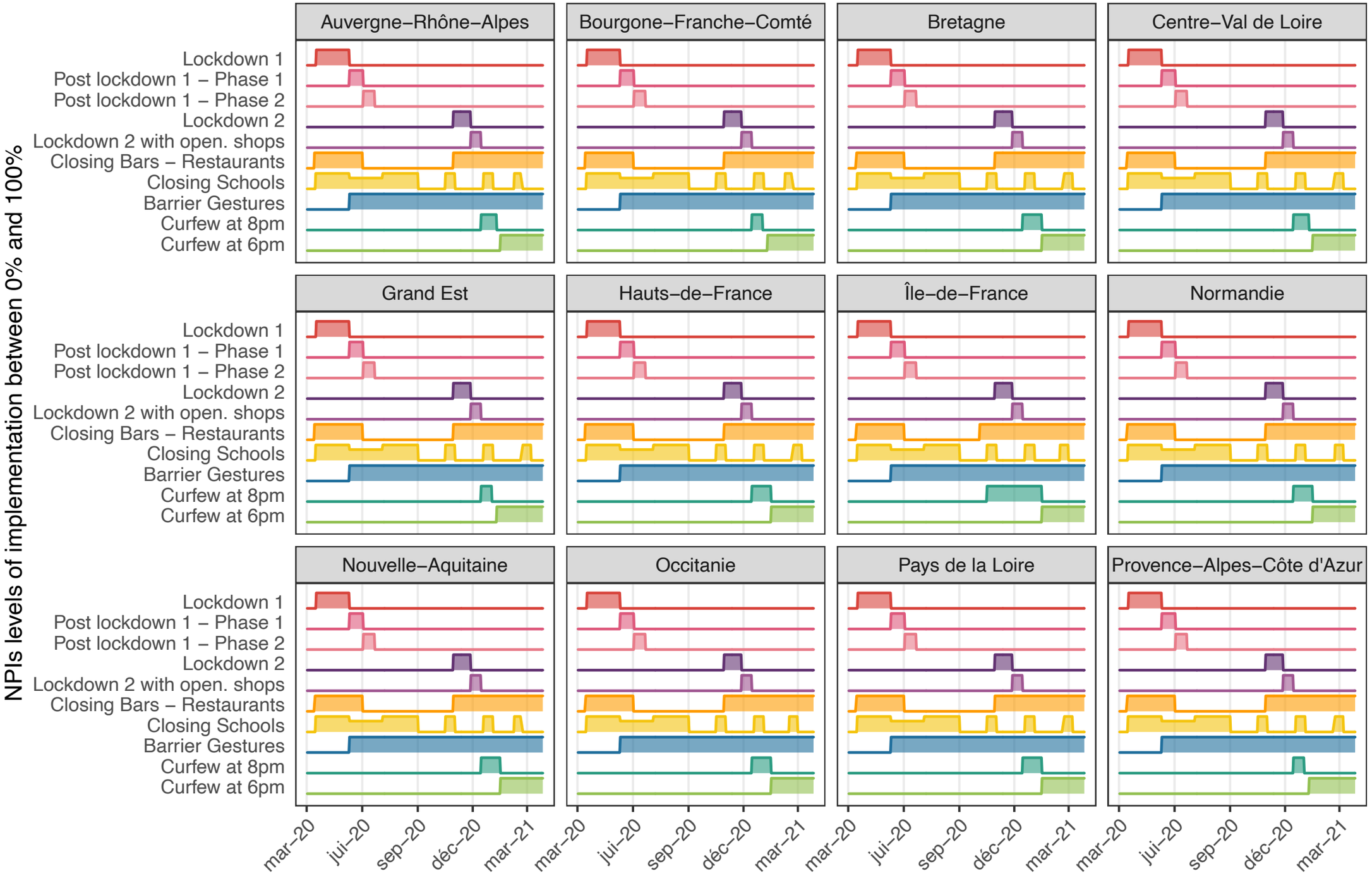


- The transmission $b = \underline{b}(1 - V/N)$ is region specific and has randoms effects.

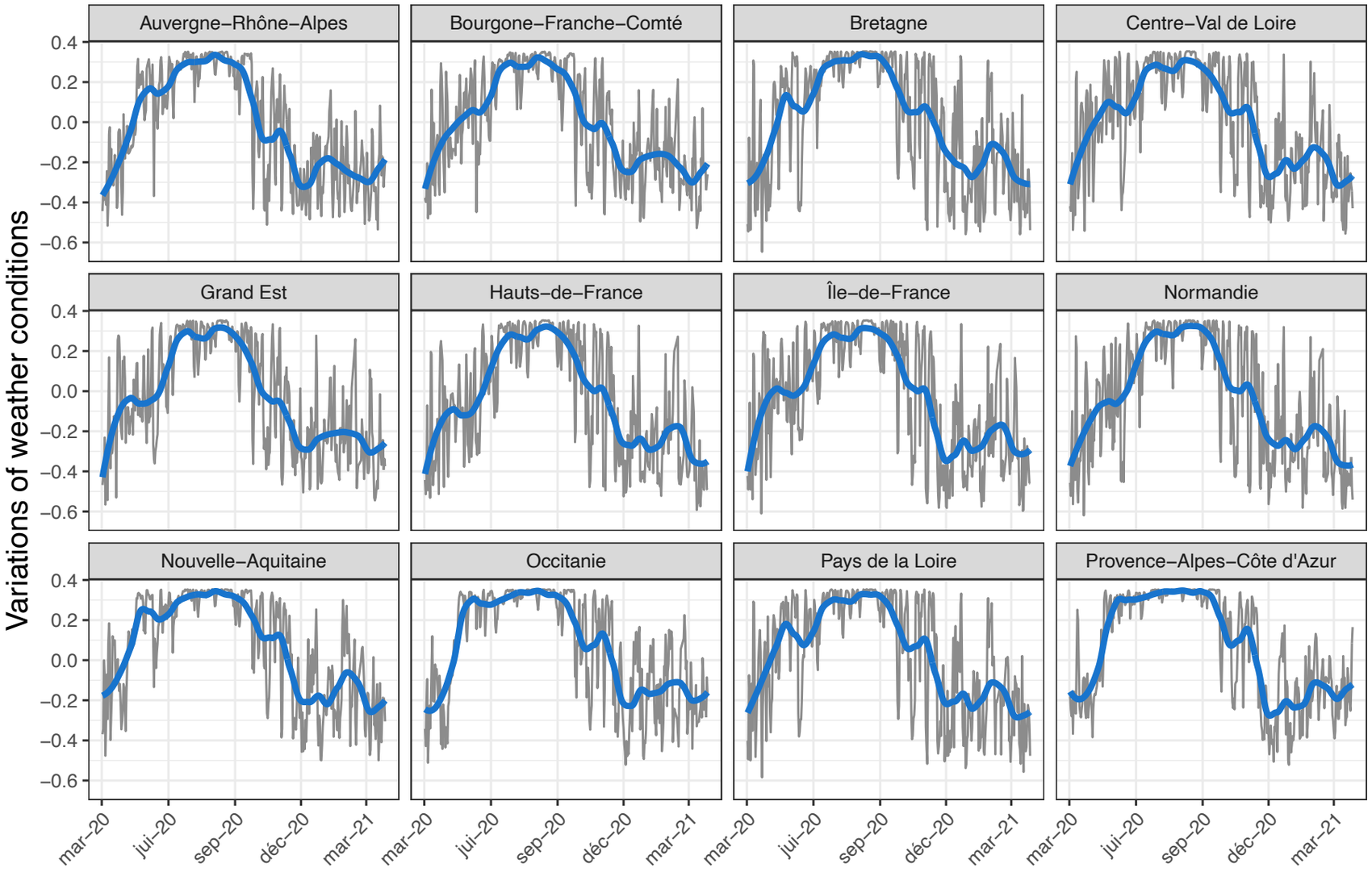
- Use sequential methods to inform the parametric shape of the effect of NPI.

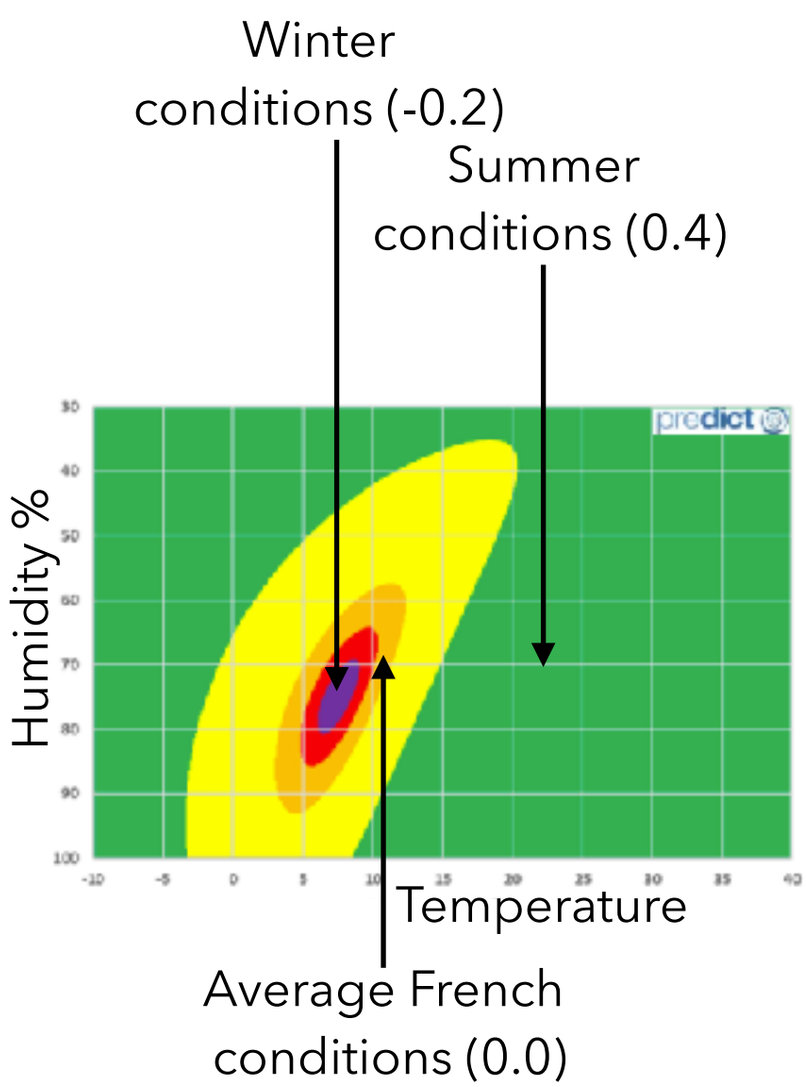


Non-Pharmaceutical interventions (NPIs)



Weather

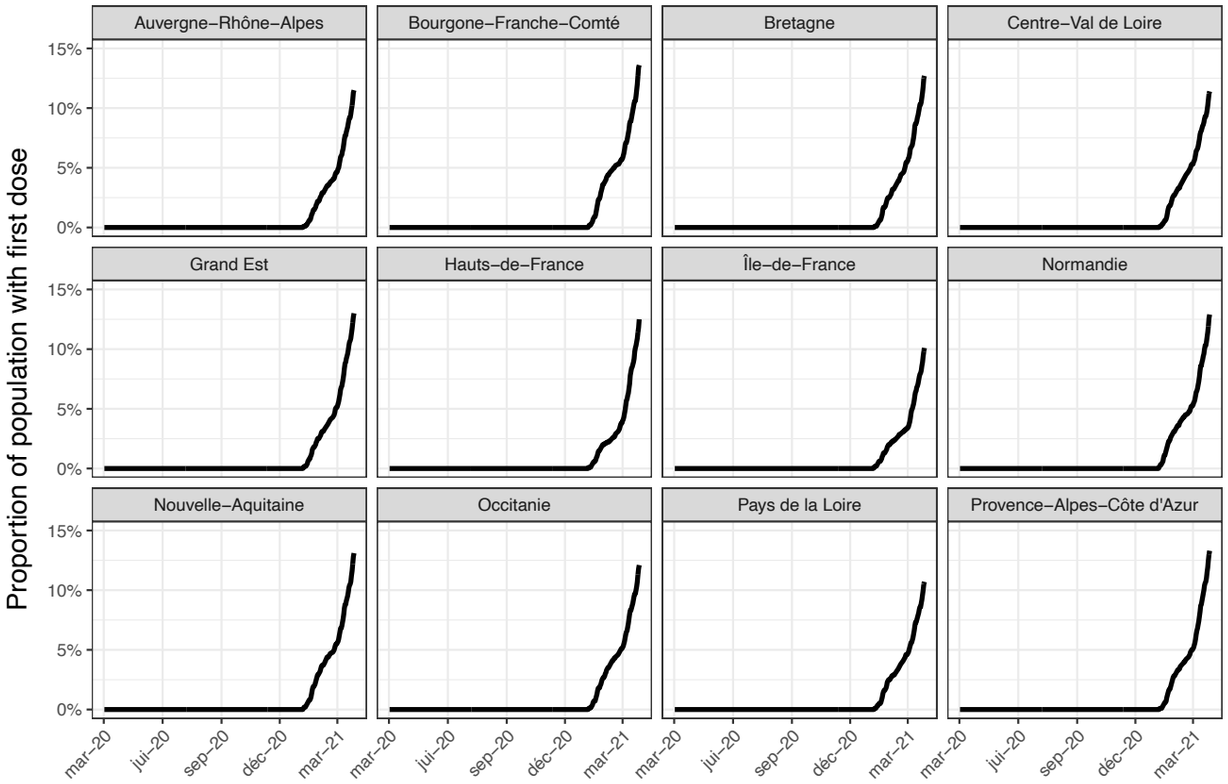




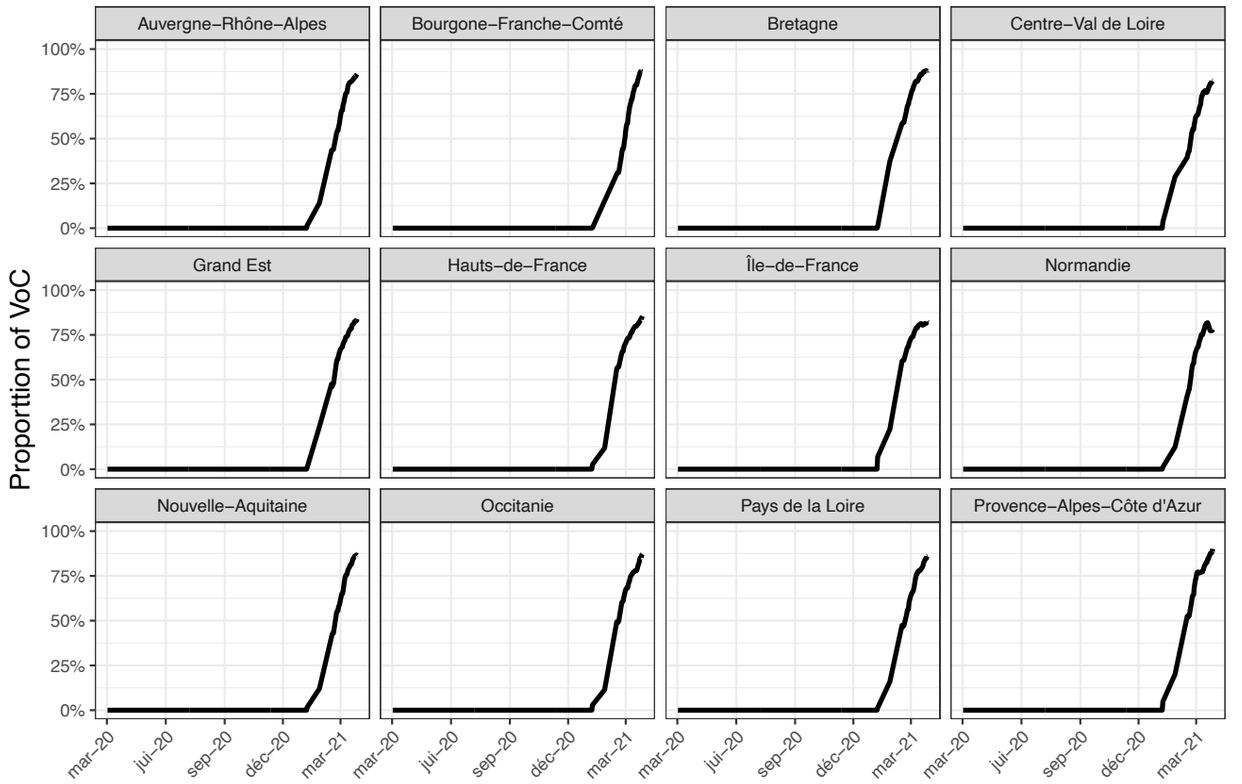
IPTCC (Bukhari et al. 2020)

Vaccination and Variants of concern

Proportion of individuals who received
At least one dose of vaccine (V) - data.gouv



Proportion of VOC in French population
from flash seroprevalence studies



i) A population criterion

- Population approach: compensate the lack of data by an available population
  - A population made of groups indexed by i
  - Criterion:

$$\boldsymbol{\xi}^i = \boldsymbol{\xi}^{pop} + \tilde{\boldsymbol{\xi}}^i \qquad \boldsymbol{\xi} = \begin{pmatrix} \boldsymbol{\xi}^1 \\ \vdots \\ \boldsymbol{\xi}^{N_P} \end{pmatrix}$$

Mixed-effect approach: pooling all the patients together and estimating a global distribution of the model parameters in the population.

$$\min_{\boldsymbol{\xi}^{pop},\boldsymbol{\xi},\boldsymbol{\nu}} \left\{ \mathscr{J}_T(\boldsymbol{\xi}^{pop},\boldsymbol{\xi},\boldsymbol{\nu}) = \frac{1}{2}\langle \boldsymbol{\xi}^{pop}, M\,\boldsymbol{\xi}^{pop}\rangle + \sum_{i=1}^{N_P}\left[\frac{1}{2}\langle(\boldsymbol{\xi}^i-\boldsymbol{\xi}^{pop}),(\hat{P}_0)^{-1}\,(\boldsymbol{\xi}^i-\boldsymbol{\xi}^{pop})\rangle + \frac{1}{2}\int_0^T \langle \nu^i(t), Q^i(t)^{-1}\nu^i(t)\rangle\,\mathrm{d}t + \frac{1}{2}\sum_{k=0}^{N_{T,\mathrm{obs}}} \langle y_k^i - C\big(z^i(t_k)\big),(W_k^i)^{-1}(y_k^i - C\big(z^i(t_k)\big)\rangle\right]\right\}$$

Classical Maximum Likelihood Estimation strategy with Gaussian disturbances

i) A population criterion

- Population approach: compensate the lack of data by an available population
  - A population made of groups indexed by i
  - Criterion:

$$\boldsymbol{\xi}^i = \boldsymbol{\xi}^{pop} + \tilde{\boldsymbol{\xi}}^i$$

$$\boldsymbol{\xi} = \begin{pmatrix} \boldsymbol{\xi}^1 \\ \vdots \\ \boldsymbol{\xi}^{N_P} \end{pmatrix}$$

Mixed-effect approach: pooling all the patients together and estimating a global distribution of the model parameters in the population.

$$\min_{\boldsymbol{\xi}^{pop},\boldsymbol{\xi},\boldsymbol{\nu}} \left\{ \mathscr{J}_T(\boldsymbol{\xi}^{pop},\boldsymbol{\xi},\boldsymbol{\nu}) = \frac{1}{2}\langle \boldsymbol{\xi}^{pop}, M\,\boldsymbol{\xi}^{pop}\rangle + \sum_{i=1}^{N_P}\left[\frac{1}{2}\langle(\boldsymbol{\xi}^i - \boldsymbol{\xi}^{pop}), (\hat{P}_0)^{-1}\,(\boldsymbol{\xi}^i - \boldsymbol{\xi}^{pop})\rangle + \frac{1}{2}\int_0^T \langle \nu^i(t), Q^i(t)^{-1}\nu^i(t)\rangle\,\mathrm{d}t + \frac{1}{2}\sum_{k=0}^{N_{T,\mathrm{obs}}} \langle y_k^i - C\big(z^i(t_k)\big), (W_k^i)^{-1}(y_k^i - C\big(z^i(t_k)\big)\rangle\right]\right\}$$

Classical Maximum Likelihood Estimation strategy with Gaussian disturbances

- Assuming that the population share the same weighted mean value:

$$\boldsymbol{\xi}^{pop} = \mathbb{E}_{\boldsymbol{\alpha},N_P}(\boldsymbol{\xi}) \overset{\text{def}}{=} \sum_{i=1}^{N_P} \alpha^i \boldsymbol{\xi}^i \text{ with } \sum_{i=1}^{N_P} \alpha^i = 1,$$

$$\min_{\boldsymbol{\xi},\boldsymbol{\nu}} \left\{ \mathscr{J}_T(\boldsymbol{\xi},\boldsymbol{\nu}) = \frac{1}{2}\langle \mathbb{E}_{\boldsymbol{\alpha},N_P}(\boldsymbol{\xi}), M\,\mathbb{E}_{\boldsymbol{\alpha},N_P}(\boldsymbol{\xi})\rangle + \sum_{i=1}^{N_P}\left[\frac{1}{2}\langle(\boldsymbol{\xi}^i - \mathbb{E}_{\boldsymbol{\alpha},N_P}(\boldsymbol{\xi})), (\hat{P}_0)^{-1}\,(\boldsymbol{\xi}^i - \mathbb{E}_{\boldsymbol{\alpha},N_P}(\boldsymbol{\xi}))\rangle + \frac{1}{2}\int_0^T \langle \nu^i(t), Q^i(t)^{-1}\nu^i(t)\rangle\,\mathrm{d}t + \frac{1}{2}\sum_{k=0}^{N_{T,\mathrm{obs}}} \langle y_k^i - C\big(z^i(t_k)\big), (W_k^i)^{-1}(y_k^i - C\big(z^i(t_k)\big)\rangle\right]\right\}$$

i) A population criterion

- Population approach: compensate the lack of data by an available population
  - A population made of groups indexed by i
  - Criterion:

$$\boldsymbol{\xi}^i = \boldsymbol{\xi}^{pop} + \tilde{\boldsymbol{\xi}}^i \qquad \boldsymbol{\xi} = \begin{pmatrix} \boldsymbol{\xi}^1 \\ \vdots \\ \boldsymbol{\xi}^{N_P} \end{pmatrix}$$

Mixed-effect approach: pooling all the patients together and estimating a global distribution of the model parameters in the population.

$$\min_{\boldsymbol{\xi}^{pop},\boldsymbol{\xi},\boldsymbol{\nu}} \left\{ \mathscr{J}_T(\boldsymbol{\xi}^{pop},\boldsymbol{\xi},\boldsymbol{\nu}) = \frac{1}{2}\langle \boldsymbol{\xi}^{pop}, M\,\boldsymbol{\xi}^{pop}\rangle + \sum_{i=1}^{N_P}\left[\frac{1}{2}\langle(\boldsymbol{\xi}^i - \boldsymbol{\xi}^{pop}), (\hat{P}_0)^{-1}\,(\boldsymbol{\xi}^i - \boldsymbol{\xi}^{pop})\rangle + \frac{1}{2}\int_0^T \langle \nu^i(t), Q^i(t)^{-1}\nu^i(t)\rangle\,\mathrm{d}t + \frac{1}{2}\sum_{k=0}^{N_{T,\mathrm{obs}}} \langle y_k^i - C\big(z^i(t_k)\big), (W_k^i)^{-1}(y_k^i - C\big(z^i(t_k)\big)\rangle\right]\right\}$$

Classical Maximum Likelihood Estimation strategy with Gaussian disturbances

- Assuming that the population share the same weighted mean value:

$$\boldsymbol{\xi}^{pop} = \mathbb{E}_{\boldsymbol{\alpha},N_P}(\boldsymbol{\xi}) \overset{\text{def}}{=} \sum_{i=1}^{N_P} \alpha^i \boldsymbol{\xi}^i \text{ with } \sum_{i=1}^{N_P} \alpha^i = 1,$$

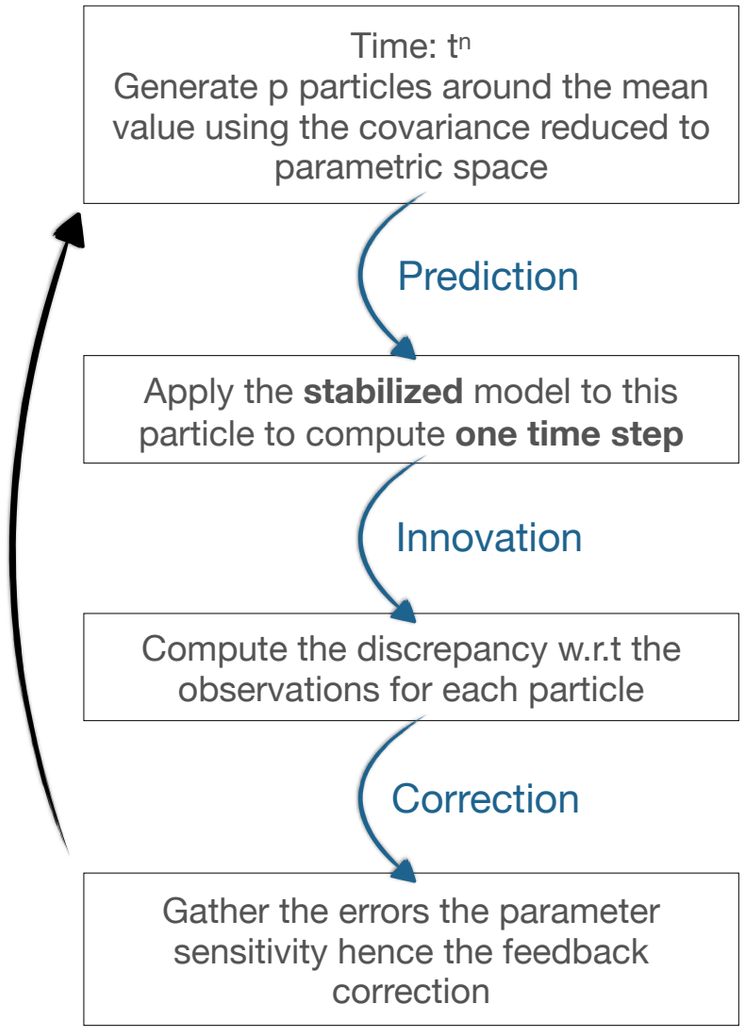
$$\min_{\boldsymbol{\xi},\boldsymbol{\nu}} \left\{ \mathscr{J}_T(\boldsymbol{\xi},\boldsymbol{\nu}) = \frac{1}{2}\langle \mathbb{E}_{\boldsymbol{\alpha},N_P}(\boldsymbol{\xi}), M\,\mathbb{E}_{\boldsymbol{\alpha},N_P}(\boldsymbol{\xi})\rangle + \sum_{i=1}^{N_P}\left[\frac{1}{2}\langle(\boldsymbol{\xi}^i - \mathbb{E}_{\boldsymbol{\alpha},N_P}(\boldsymbol{\xi})), (\hat{P}_0)^{-1}\,(\boldsymbol{\xi}^i - \mathbb{E}_{\boldsymbol{\alpha},N_P}(\boldsymbol{\xi}))\rangle + \frac{1}{2}\int_0^T \langle \nu^i(t), Q^i(t)^{-1}\nu^i(t)\rangle\,\mathrm{d}t + \frac{1}{2}\sum_{k=0}^{N_{T,\mathrm{obs}}} \langle y_k^i - C\big(z^i(t_k)\big), (W_k^i)^{-1}(y_k^i - C\big(z^i(t_k)\big)\rangle\right]\right\}$$

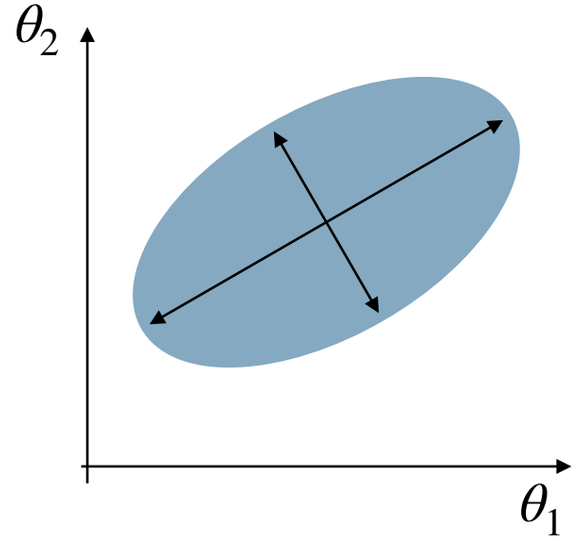
- We then introduce the matrix $\boldsymbol{M}_0$ of dimension $N_P \times N_z$ defined by blocks such that for $1 \leq i,j \leq N_P$, the block $(\boldsymbol{M}_0)_{i,j}$ of indexes in $\left[(i-1)\,N_z + 1, i\,N_z\right] \times \left[(j-1)\,N_z + 1, j\,N_z\right]$:

$$(\boldsymbol{M}_0)_{i,j} = \frac{1}{(N_P)^2} M + \left(\delta_{ij} - \frac{1}{N_P}\right)(\hat{P}_0)^{-1}.$$

ii) Unscented Kalman Filter

- Here we consider an U*nscented Kalman Filter*.
- The non linear operators are replaced by finite difference approximations based on sampling points.
- Sampling points can be seen as well-chosen “interpolation points” which propagate the mean and covariance of a random variable





S. Julier, J. Uhlmann and H. Durrant-Whyte, A new approach for filtering nonlinear systems, in American Control Conference (1995).

iii) A reduced order version

- Missing step: define the initial reduced covariance matrix and the initial extension matrix from the initial covariance matrix.
- Our strategy is based on a clustering approach applied to the observations sequence using the k-means algorithm:

$$(U_0^{-1})_{r,s} = \frac{1}{n_{c_r} n_{c_s}} \sum_{i \in c_r} \sum_{j \in c_s} (P_0)_{i,j}$$

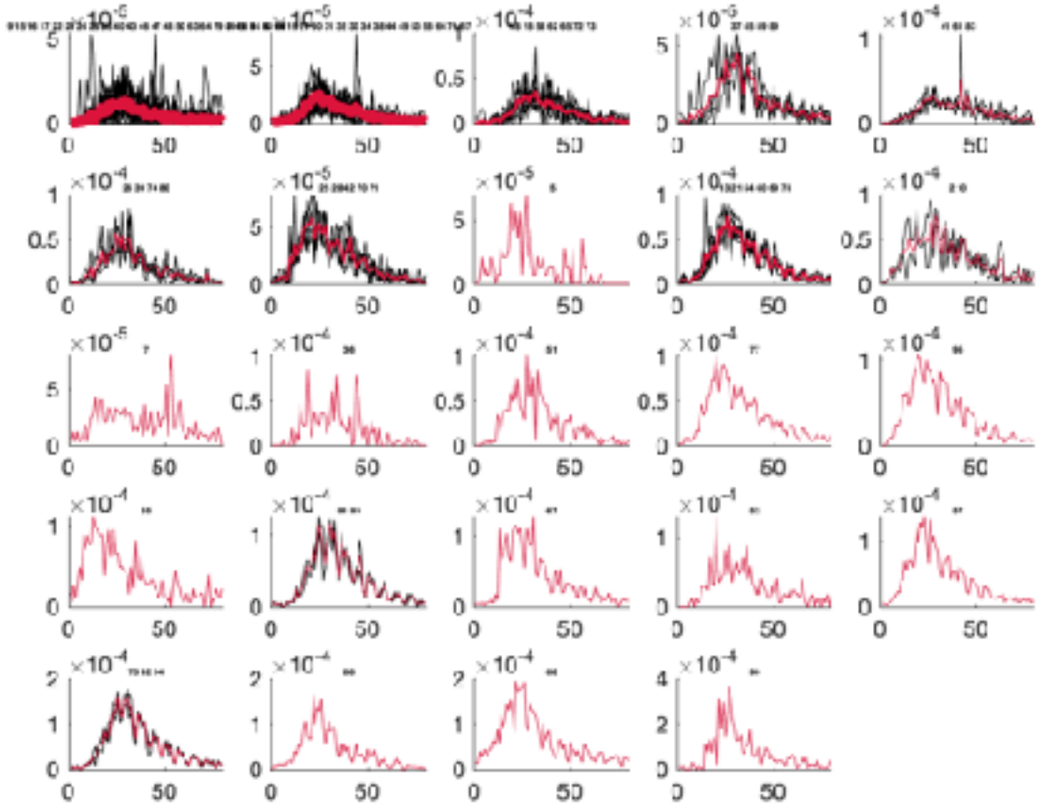
$$(L_0)_{i,r} = \beta_{i,r} \mathbb{1}_{N_z, N_z}$$

how much each region /department *i* belongs to cluster c_r

SVD

$$P = L U^{-1} L^T$$

$$\dot{L} = AL \text{ and } \dot{U} = L^T C^T W^{-1} C L$$



Means clustering with 24 clusters on H/N_{pop} for 94 departments

Some details about the estimation

- "State" variable for the dynamics with a Backward-Euler time scheme by regions without the S variables

$$\begin{pmatrix} x_{n+1}^i \\ b_{n+1}^i \\ \theta_{n+1}^i \end{pmatrix} = \begin{pmatrix} x_n^i + \Delta t\, f(x_n^i, b_n^i, \theta_n^i) \\ b_n^i + \Delta t\, g^i(t_n, \theta_n^i) \\ \theta_n^i \end{pmatrix} + \begin{pmatrix} 0_5 \\ 1 \\ 0_{N_p} \end{pmatrix} \nu_n$$

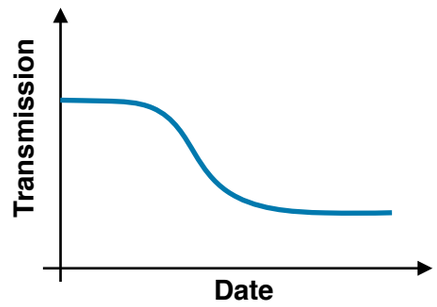
$$x = (E, I, R, A, H) \in \mathbb{R}^5$$

as the shape of *b* is undetermined

- *b*(*t*) can be modeled by a logistic function during the first lockdown for example

$$b(t) = G(t) = b_M - \frac{(b_M - b_m)}{1 + e^{-\frac{(t - t_\ell)}{\tau}}} \quad \xrightarrow{\text{dynamics}} \quad db(t) = g(t) + d\nu(t)$$

Wiener process



- State transformation ("Twisted" UKF)

$$\psi(x) = \mathbf{logit}\left(\frac{E}{N}\right), \mathbf{logit}\left(\frac{I}{D_q N}\right), \mathbf{logit}\left(\frac{R}{N}\right), \mathbf{logit}\left(\frac{A}{N}\right), \mathbf{logit}\left(\frac{H}{N}\right).$$

Estimation transmission rate in 3 steps

1. Estimate the parameter b_{init} using the data obtained before the lockdown namely when b should be constant.

$b(t) = G(t)$

	IDF	CVL	BFC	Norm.	HDF	GE	PL	Bret.	NA	Occ.	AURA	PACA	Nat. avg.
b_{init}	0.789	0.767	0.784	0.773	0.781	0.809	0.761	0.765	0.768	0.789	0.786	0.778	0.779

Estimation transmission rate in 3 steps

1. Estimate the parameter b_{init} using the data obtained before the lockdown namely when b should be constant.

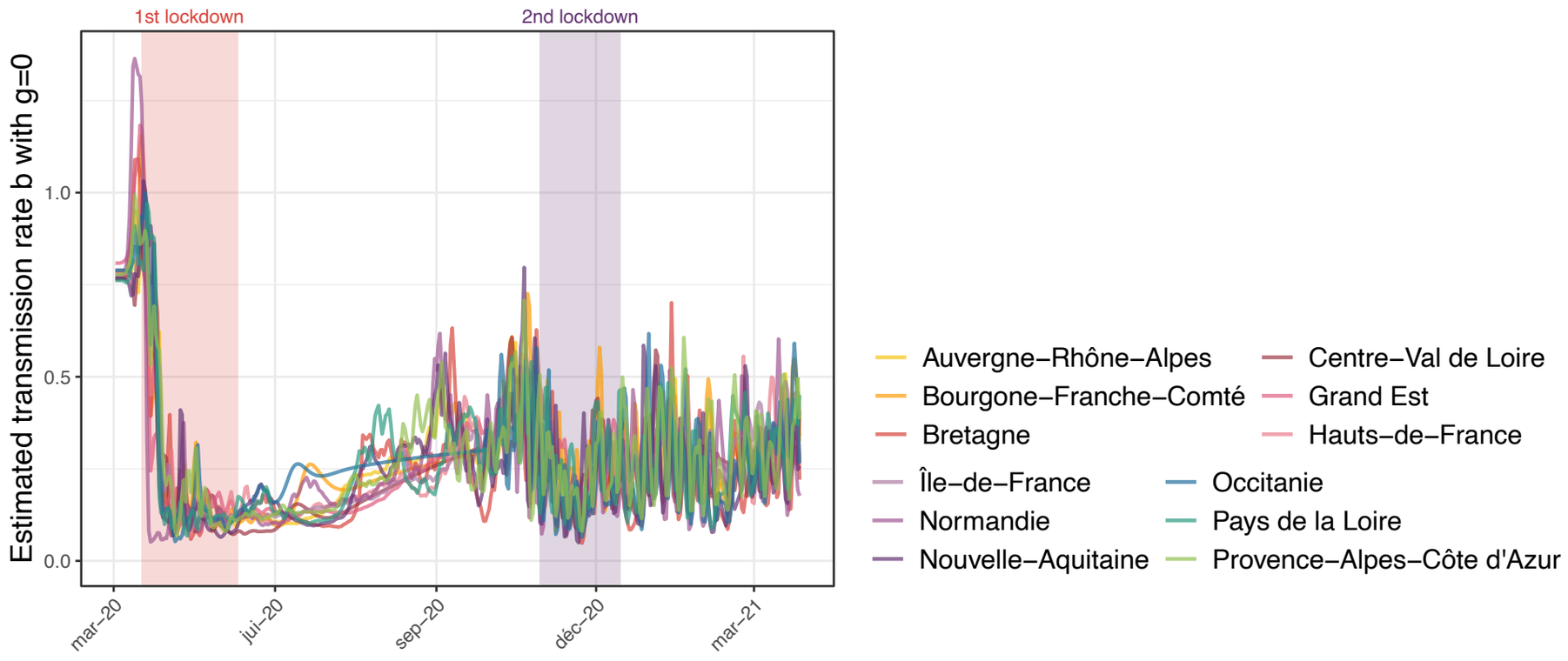
$$b(t) = G(t)$$

	IDF	CVL	BFC	Norm.	HDF	GE	PL	Bret.	NA	Occ.	AURA	PACA	Nat. avg.
b_{init}	0.789	0.767	0.784	0.773	0.781	0.809	0.761	0.765	0.768	0.789	0.786	0.778	0.779

2. Estimate the shape of b without the logistic function prior (i.*e.* $g(t) = 0$) but setting b_{init} at the value estimated during Step 1.

$$\mathrm{d}b(t) = \mathrm{d}\nu(t)$$

Kalman filters estimated transmission over time in multiple regions

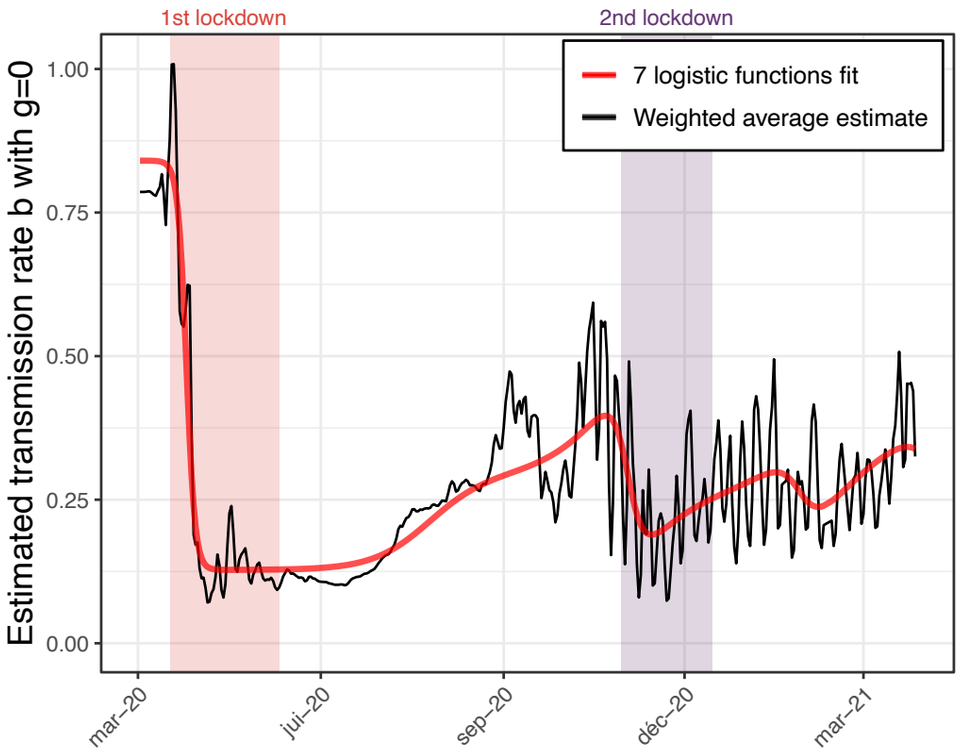


Estimation transmission rate in 3 steps

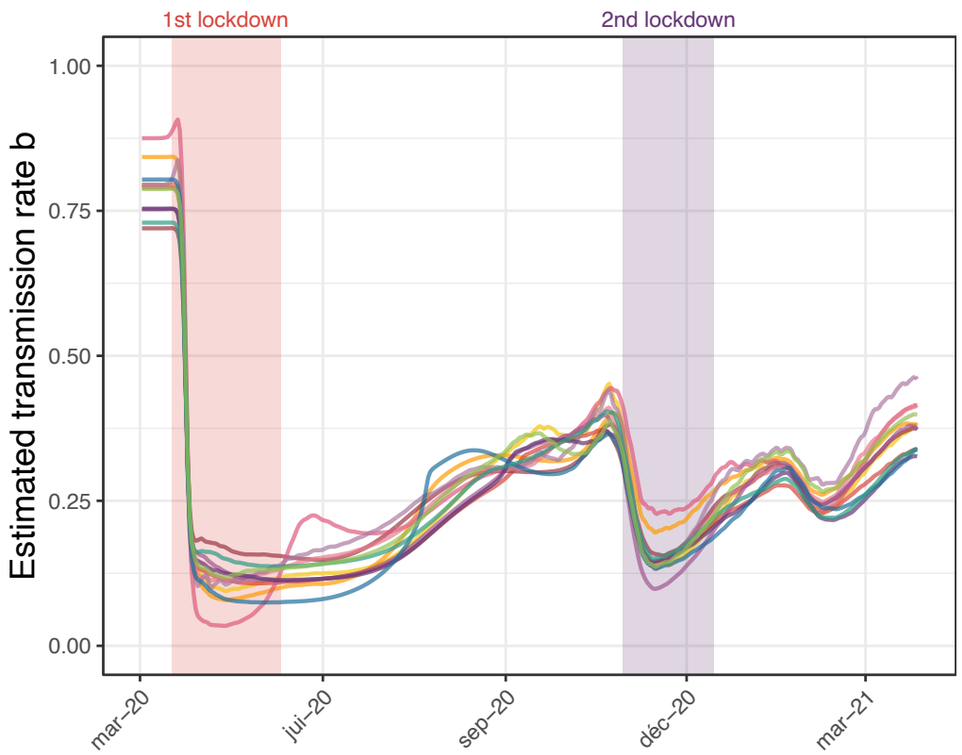
3. In order to evaluate if these weekly plateaus are just a compensation of our lack of prior information about b, we propose to fix g as a sum of 7 logistic functions

$$\mathrm{d}b(t) = g(t) + \mathrm{d}\nu(t)$$

Population-weighed mean of transmission rates



Kalman filters estimated transmission over time in multiple regions



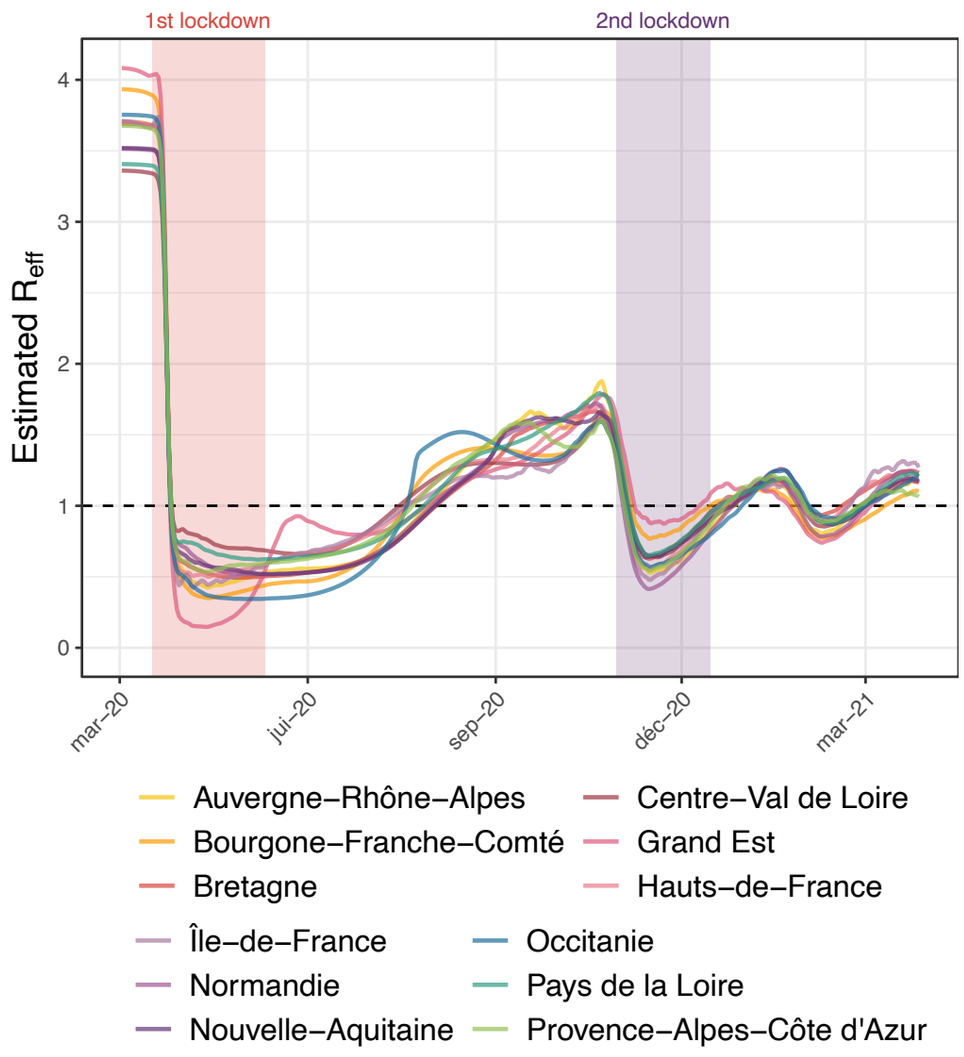
Auvergne–Rhône–Alpes
Bourgone–Franche–Comté
Bretagne
Centre–Val de Loire
Grand Est
Hauts–de–France
Île–de–France
Normandie
Nouvelle–Aquitaine
Occitanie
Pays de la Loire
Provence–Alpes–Côte d'Azur

Effects of NPIs

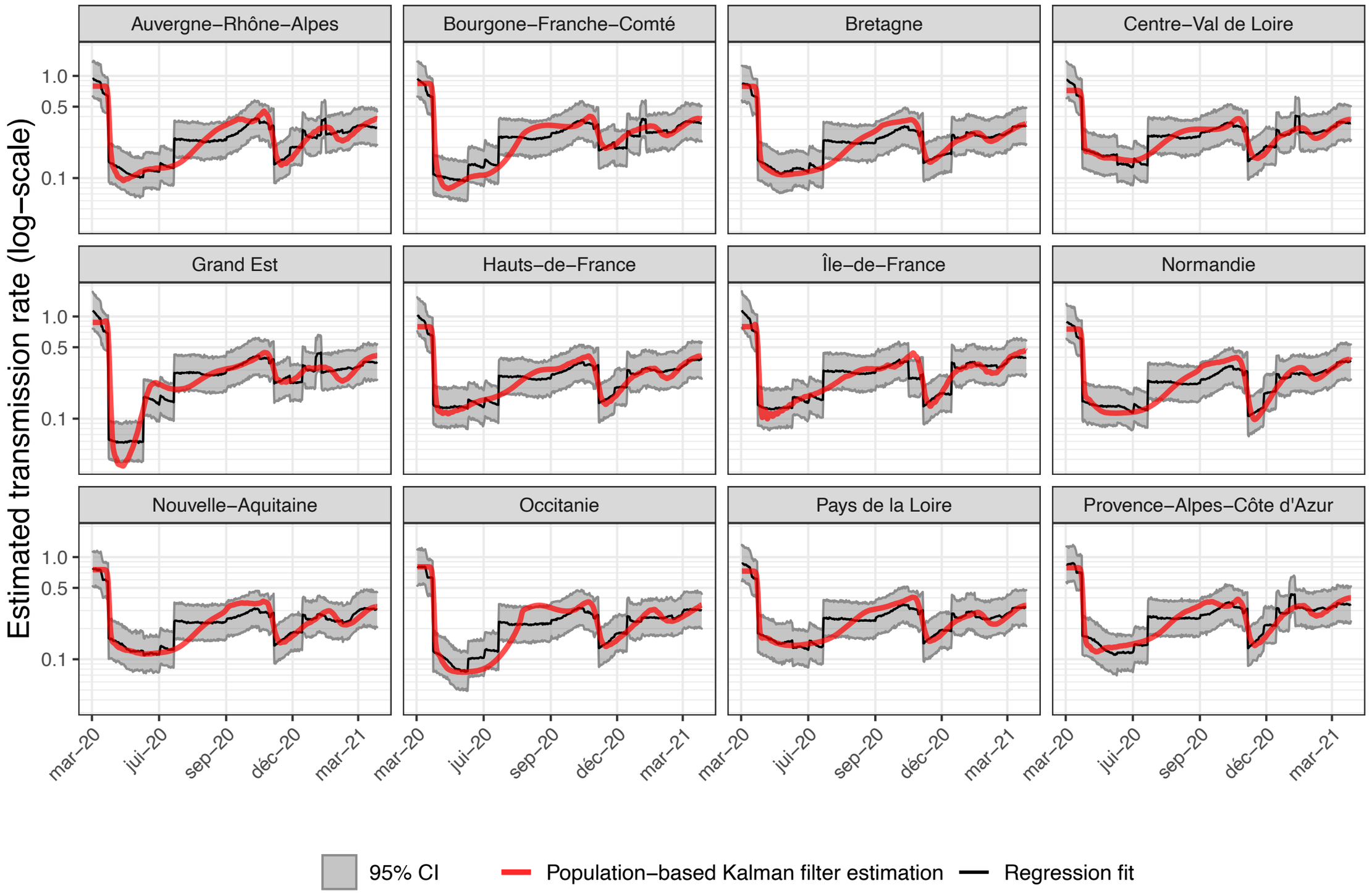
$$\log(\underline{b}^i(t)) = \alpha + \sum_{r=1}^{3} \beta_r^i I_{\text{intervention}_r^i}(t) + \sum_{r=4}^{10} \beta_r I_{\text{intervention}_r^i}(t) + \beta_{8w} I_{\text{Bar and restaurants closure}^i} \times W^i(t) + \beta_v VOC^i(t) + \beta_w W^i(t) + + u^i.$$

Covariate	Modification of the transmission rate: scale	increase/decrease
NPI		
Lockdown 1 - delay of 7 days	-78% [-82% ; -74%]	↘
Post lockdown 1 - Phase 1	-54% [-56% ; -52%]	↘
Post lockdown 1 - Phase 2	-48% [-50% ; -47%]	↘
Lockdown 2 - delay of 7 days	-54% [-57% ; -49%]	↘
Lockdown 2 with opened shops	-51% [-53% ; -49%]	↘
Closing schools	- 7% [- 8% ; - 5%]	↘
Barrier gestures	-46% [-48% ; -44%]	↘
Curfew at 6PM	-30% [-33% ; -26%]	↘
Curfew at 8PM	-28% [-31% ; -25%]	↘
Closing bars & restaurants (ref. W=0)	-10% [-13% ; - 8%]	↘
Closing bars & restaurants during summer (vs. ref.)	- 8% [-11% ; - 4%]	↘
during winter (vs. ref.)	-11% [-14% ; - 8%]	↘
Other factors		
100% of VOC circulating	22% [15% ; 28%]	↗
Weather effect during summer (ref. $W = 0$)	-22% [-24% ; -21%]	↘
during winter (ref. $W = 0$)	10% [9% ; 11%]	↗

Kalman filters estimated transmission over time in multiple regions

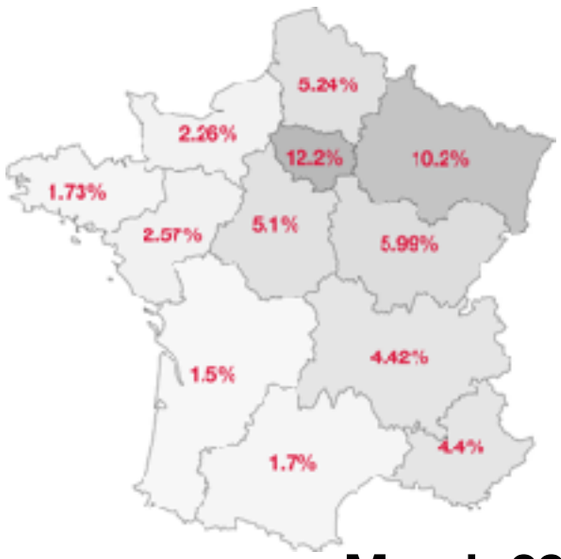


Regression adjustment

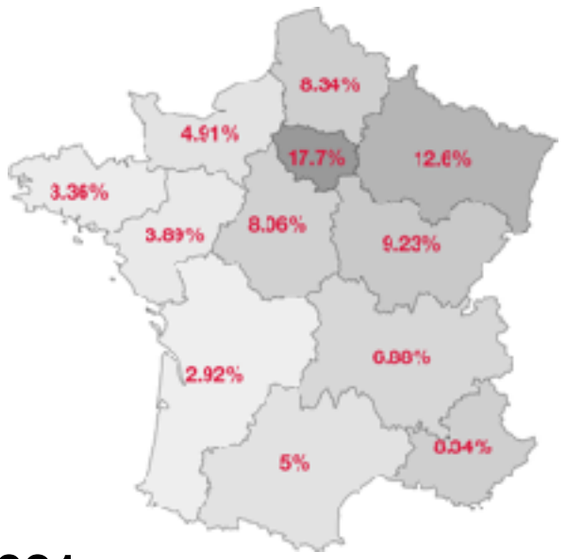


Attack rates & reproductive number

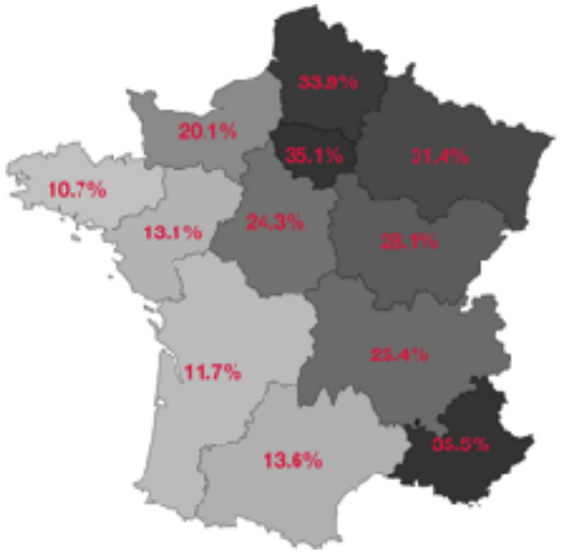
May 11, 2020
National French - 5.7%



October 5, 2020
National French - 8.8%

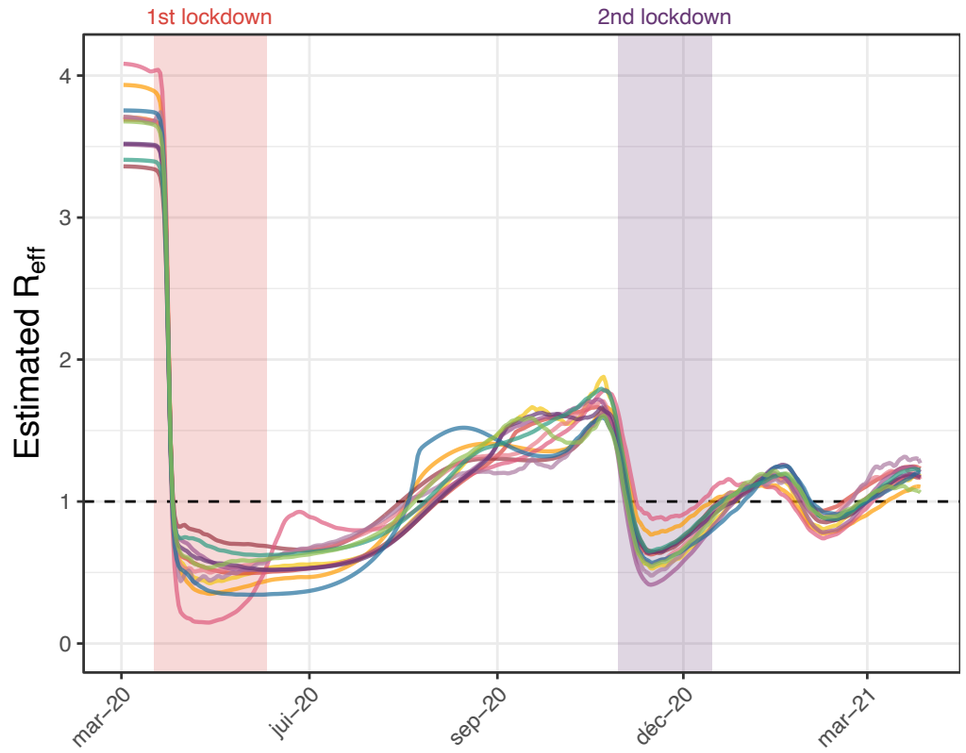


March 28, 2021
National French - 25.3%



Basic reproductive number
3.10 [2.95; 3.26]

Effective reproductive number



Illustrative example: Spheroids Electroporation

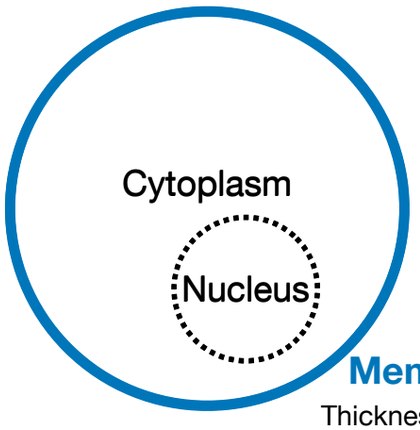
Concept of electroporation

Cell

Radius ~ 5 to 10 μm

Intra-cellular conductivity σ_c ~ 0.5 to 1 S.m^{-1}

Intra-cellular conductivity σ_e ~ 1 to 1.5 S.m^{-1}



Membrane

Thickness $h \sim$ 5nm

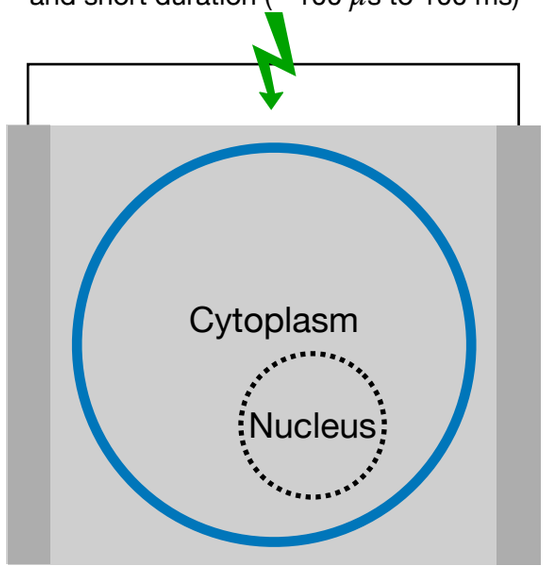
Permitivity $\varepsilon \sim 4.5\varepsilon_0$

Capacitance C_m

Conductance S_m

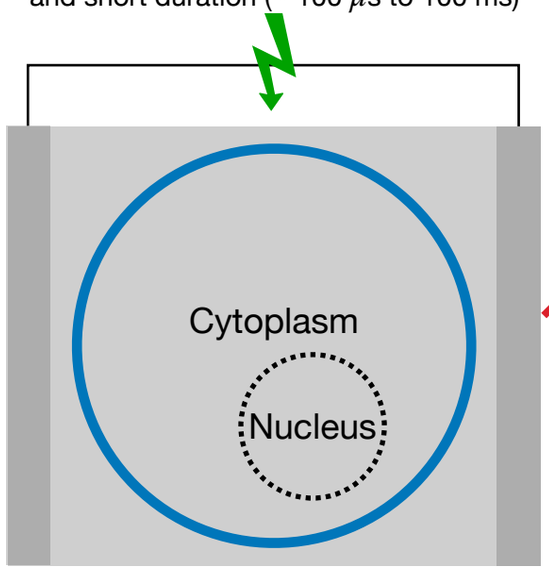
Concept of electroporation

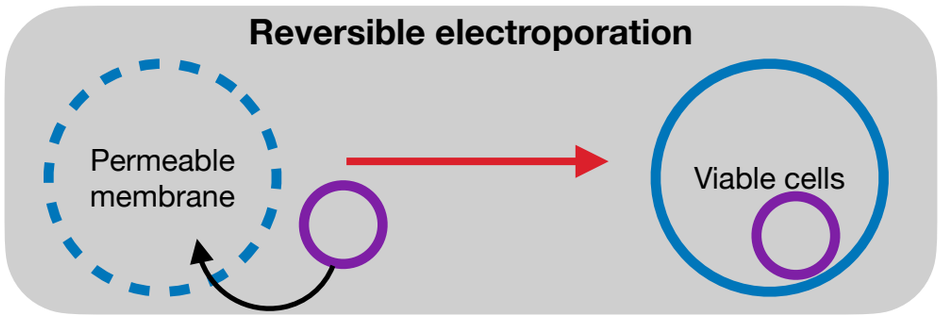
High voltages pulses (100 < |E| < 3000 V.cm^{-1})
and short duration (~ 100 μs to 100 ms)



Concept of electroporation

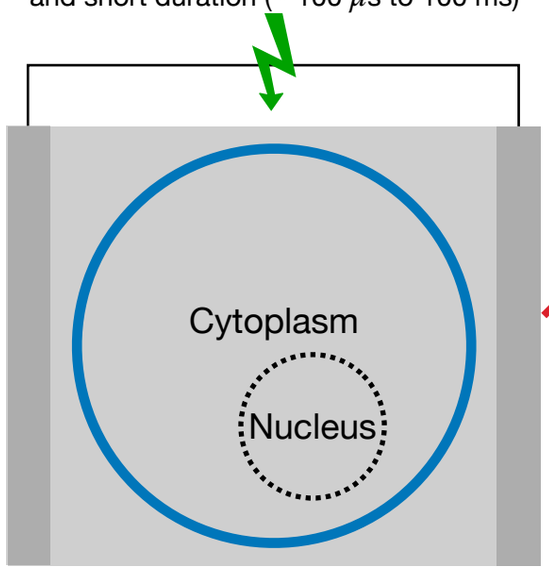
High voltages pulses (100 < |E| < 3000 V.cm^{-1})
and short duration (~ 100 μs to 100 ms)

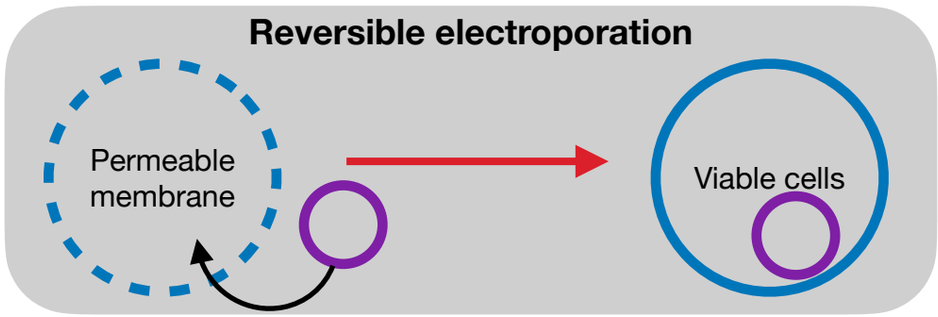




Concept of electroporation

High voltages pulses (100 < |E| < 3000 V.cm^{-1})
and short duration (~ 100 μs to 100 ms)



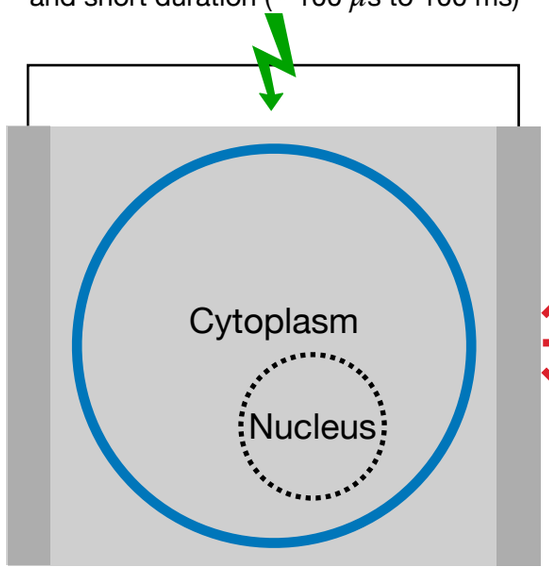


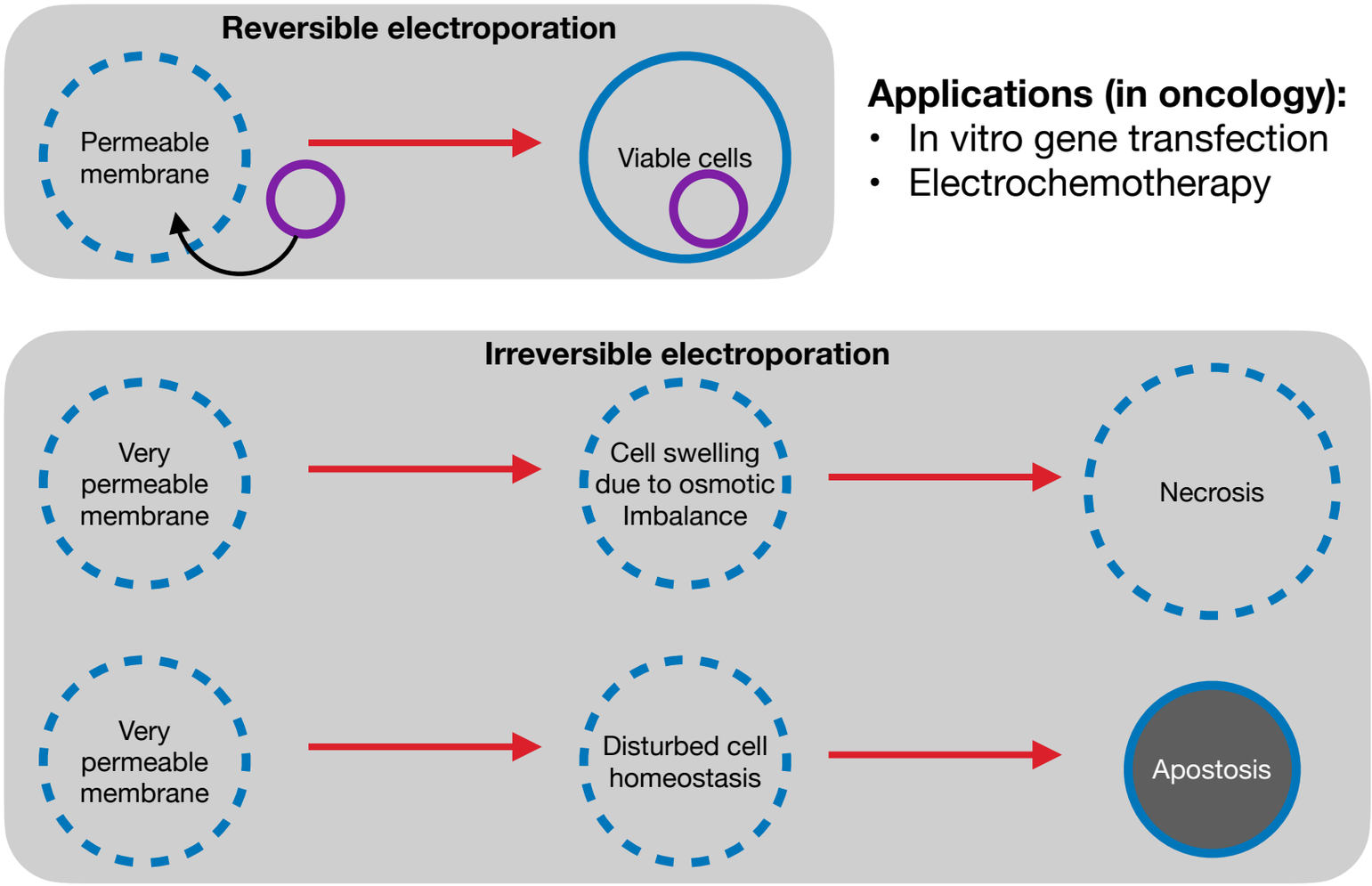
Applications (in oncology):

- In vitro gene transfection
- Electrochemotherapy

Concept of electroporation

High voltages pulses (100 < |E| < 3000 V.cm^{-1})
and short duration (~ 100 μs to 100 ms)



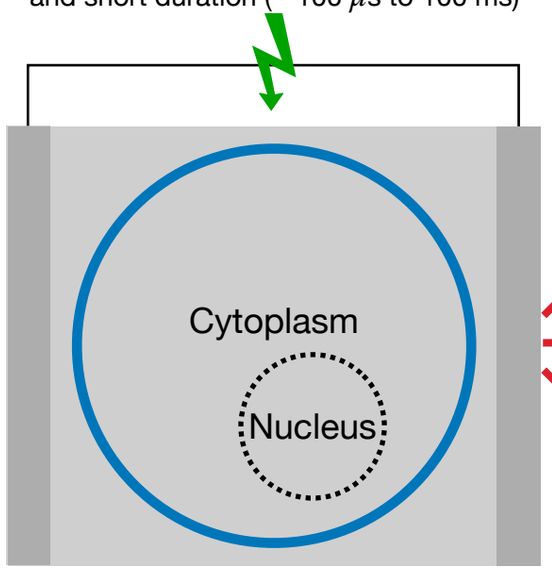


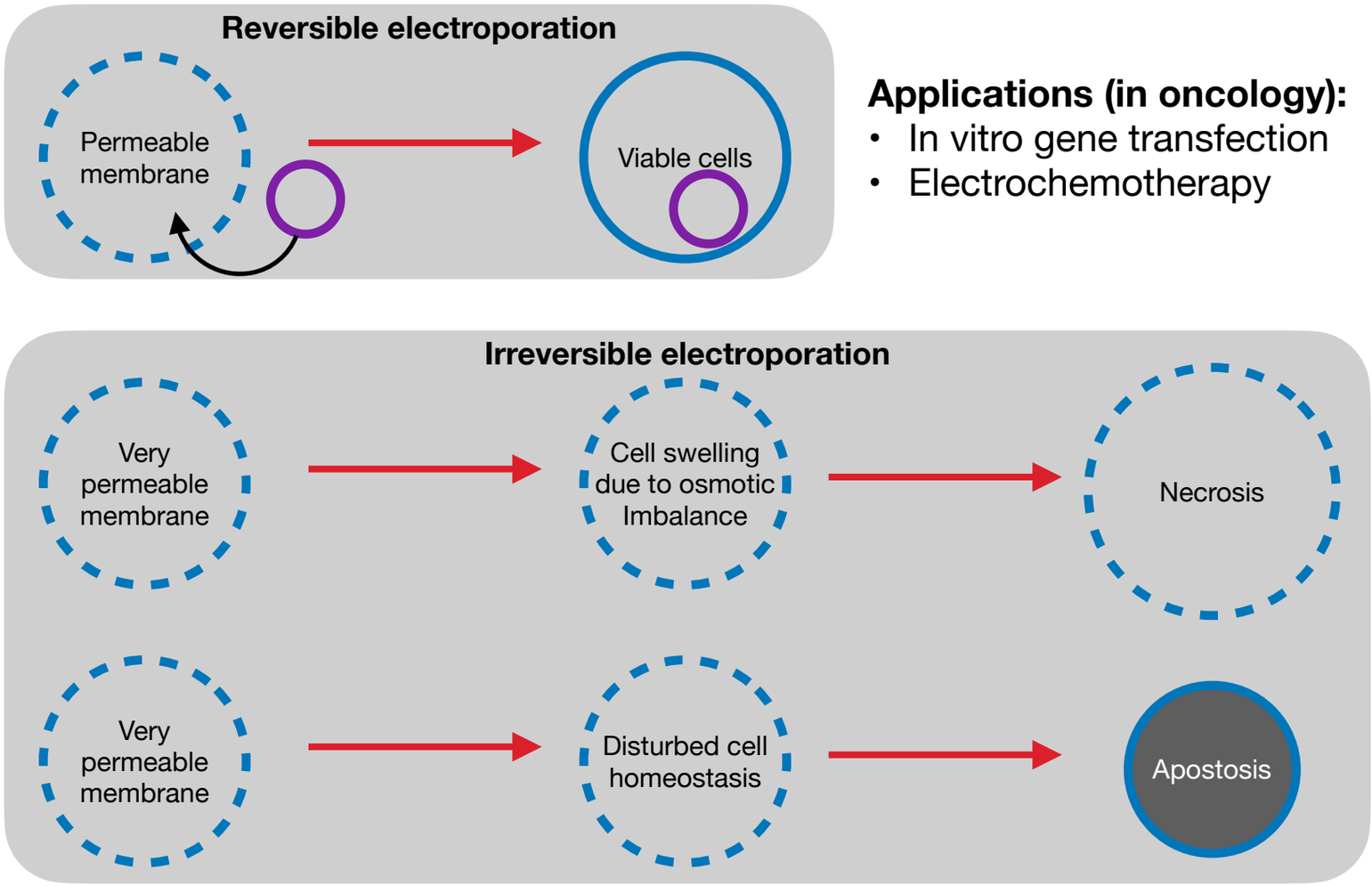
Applications (in oncology):

- In vitro gene transfection
- Electrochemotherapy

Concept of electroporation

High voltages pulses (100 < |E| < 3000 V.cm^{-1})
and short duration (~ 100 μs to 100 ms)



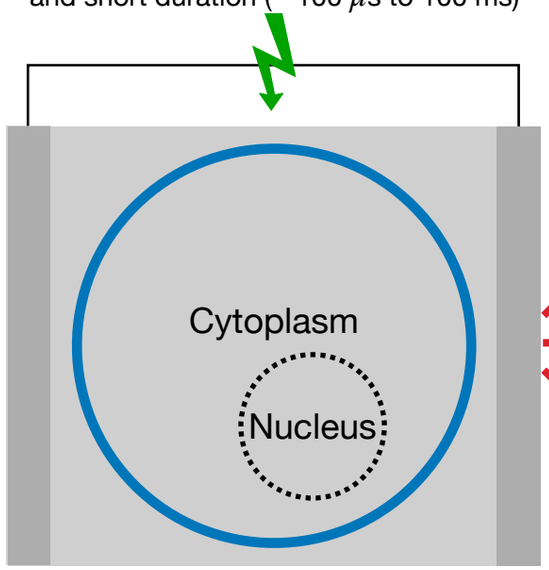


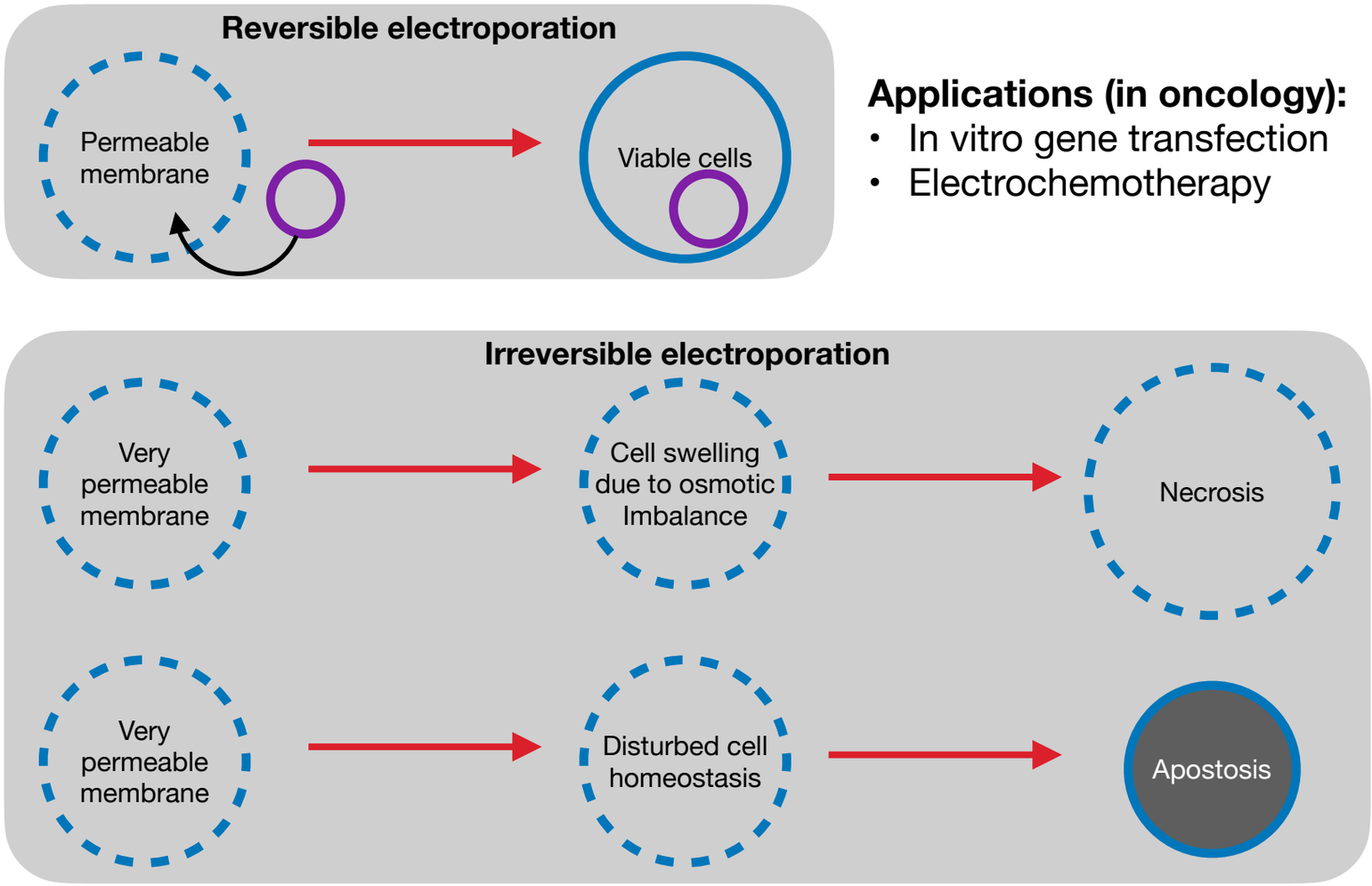
Applications (in oncology):
- In vitro gene transfection
- Electrochemotherapy

Applications:
- Tumoral ablation
- Cardiac ablation

Concept of electroporation

High voltages pulses (100 < |E| < 3000 V.cm^{-1})
and short duration (~ 100 μs to 100 ms)





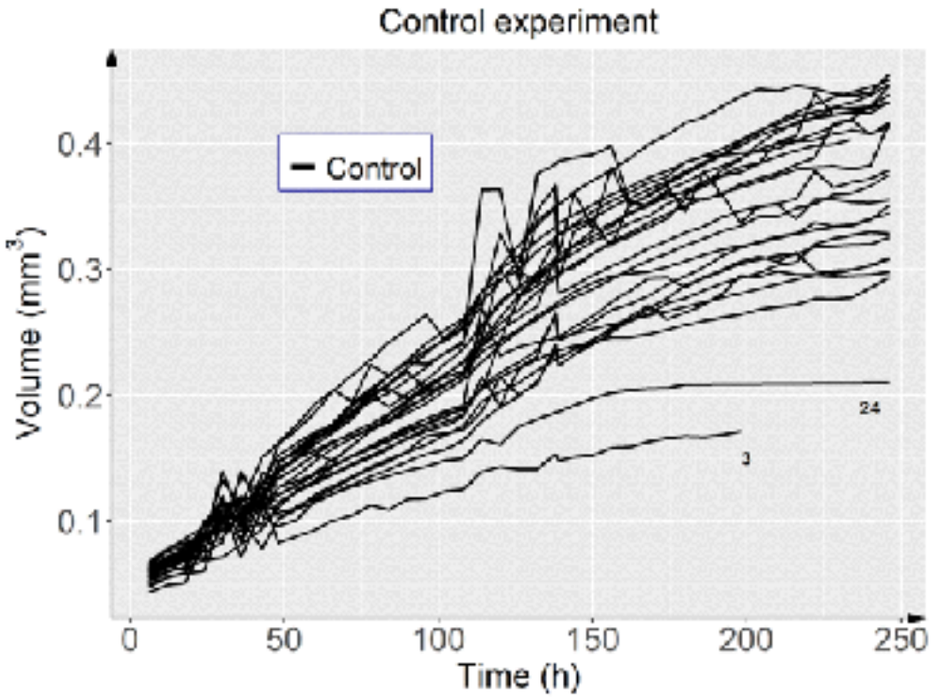
Applications (in oncology):
- In vitro gene transfection
- Electrochemotherapy

Applications:
- Tumoral ablation
- Cardiac ablation

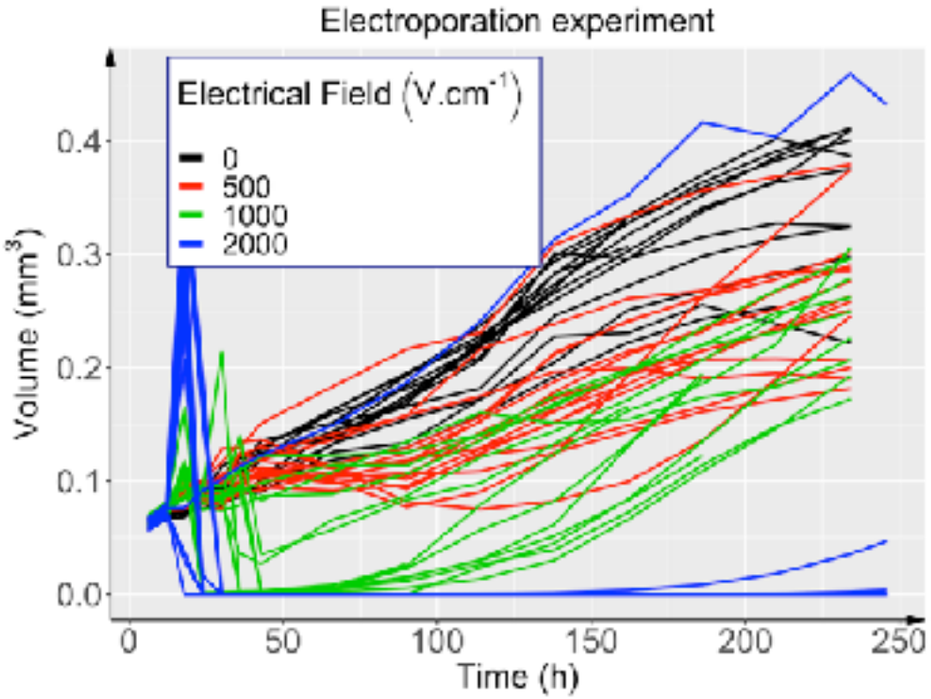
Open questions:
- Determine the treated zone
- Better understand the impact of reversible electroporation to develop efficient electrochemotherapy

Data presentation & Problematic

- Data have been provided by Institut de Pharmacologie et de Biologie Structurale, Toulouse.
- They correspond to volume evolutions of 83 spheroids (cell strain: HCT 116) over 250 h.
- Measurements are made using optical instruments.



24 spheroids (3 and 24 are excluded)



59 spheroids divided into 4 categories depending on the intensity of electrical pulses: EF 0, EF 500, EF 1000, EF 2000 ($V.cm^{-1}$) (9 will be excluded)

- Can we quantify the effects of electroporation on spheroids considering the intensity of the pulses thanks to mathematical modeling?

Free Growth - General Equations

- System of partial differential equations

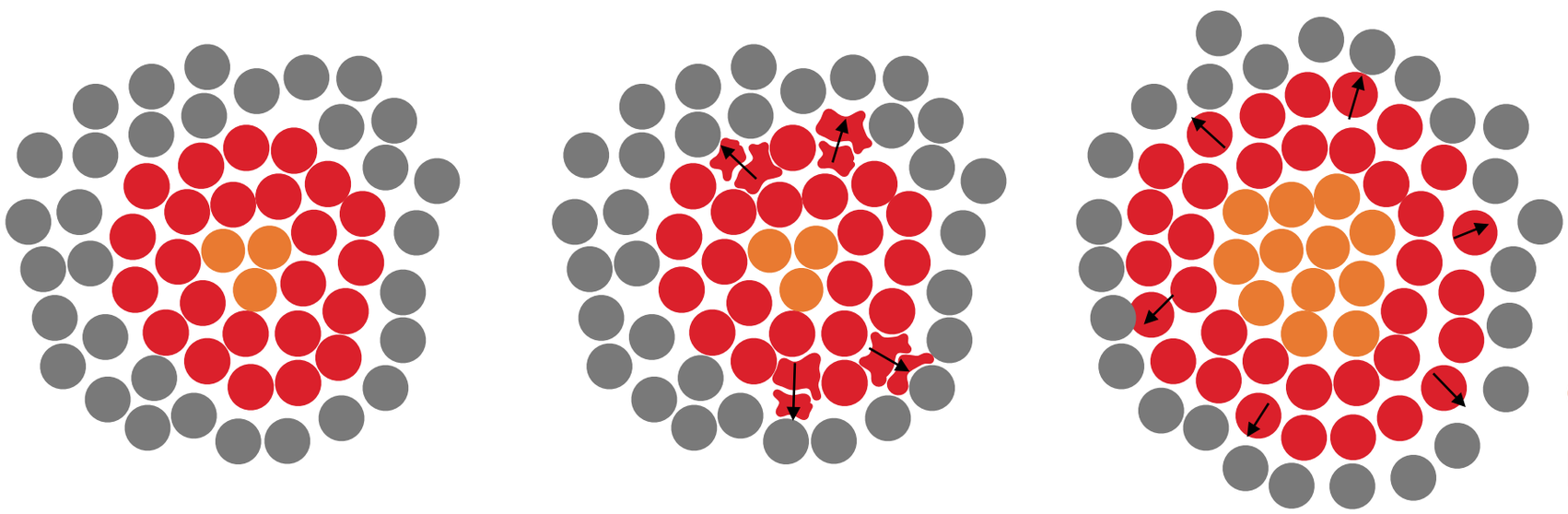
$$\partial_t P + \nabla \cdot (\vec{v} P) = \tau^G(t)(P+Q) - \tau^{PtoQ}(t)P, \quad \Omega(t), \quad \text{Proliferative cells}$$
$$\partial_t Q + \nabla \cdot (\vec{v} Q) = \tau^{PtoQ}(t)P, \quad \Omega(t). \quad \text{Quiescent cells}$$

$$\nabla \cdot \vec{v} = \tau^G(t)(P+Q).$$

$\vec{v}$ is the velocity

- Considering $\tau^G : t \mapsto ae^{-bt}$ leads to a Gompertz equation for the volume.
- The term τ^{PtoQ} can be seen as the lack of oxygen.

$P + Q = 1, \Omega(t)$ (saturation hypothesis)



Healthy cells

Quiescent cells

Proliferative cells

Electroporation - General Equations

- PDE System

Proliferative cells

$$\partial_t P + \nabla \cdot (\vec{v} P) = \tau^G (P + Q) - \tau^{PtoQ} P,\ \Omega(t)$$

Quiescent cells

$$\partial_t Q + \nabla \cdot (\vec{v} Q) = \tau^{PtoQ} P,\ \Omega(t)$$

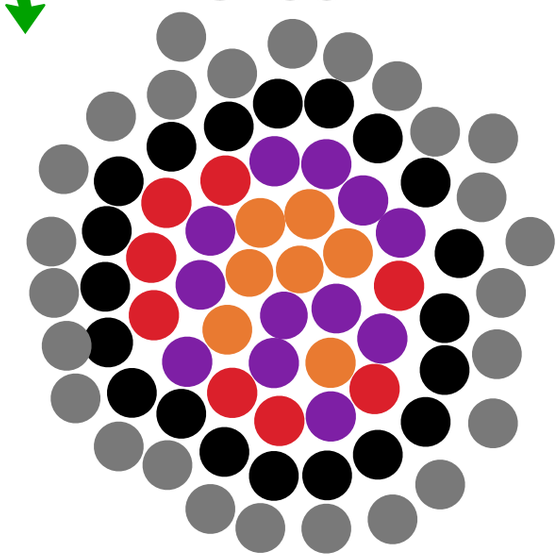
Cells with a modified metabolism

$$\partial_t F + \nabla \cdot (\vec{v} F) = 0,\ \Omega(t)$$

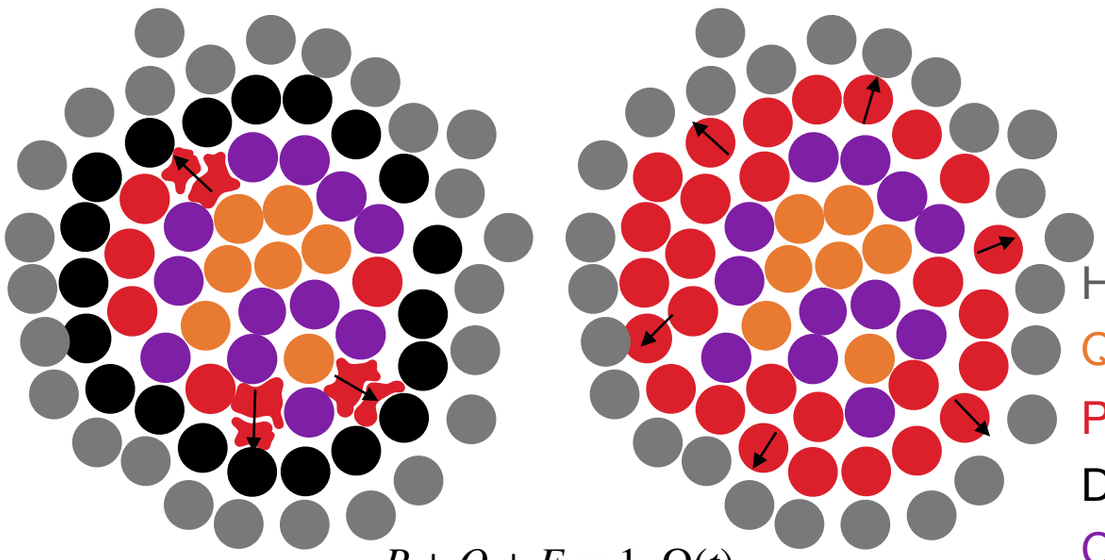
- Impacts of the electrical shock:
  - (1) a part of proliferative and quiescent cells is destroyed *i.e.* $R(t_{as}) = pR(t_{bs})$,
  - (2) the metabolism of a part of the cells is modified *i.e.* $F(t_{as}, x) = \lambda(P(t_{as}, x) + Q(t_{as}, x))$, for $x \in \Omega(t_{as})$,
  - (3) the value of a appearing in the growth rate $\tau^G = abe^{-bt}$ increases. We denote by m the multiplicative value: $a_{new} = ma$.

Just before the shock

Just after the shock



Re-growth



$P + Q + F = 1,\ \Omega(t)$
(saturation hypothesis)

Healthy cells
Quiescent cells
Proliferative cells
Death cells
Cells with a new metabolism

Electroporation - Radial Equations

- 1D PDE System

$$\partial_t P = -\tau^G(r^{-2}I - rI(t,1))\partial_r P + \tau^G(1-F)(1-P) - \tau_{PtoQ}P, \qquad [0,t_{last}]\times[0,1]$$

$$\partial_t F = -\tau^G(r^{-2}I - rI(t,1))\partial_r F - \tau^G(1-F)F, \qquad [0,t_{last}]\times[0,1]$$

$$I = \int_0^r (1-F)\underline{r}^2 d\underline{r} \qquad [0,t_{last}]\times[0,1]$$

$$R' = R\tau^G I(t,1), \qquad [0,t_{last}]$$

$$Q = 1-(P+F), \qquad [0,t_{last}]\times[0,1]$$

High quiescent proportion of cells in the spheroid center

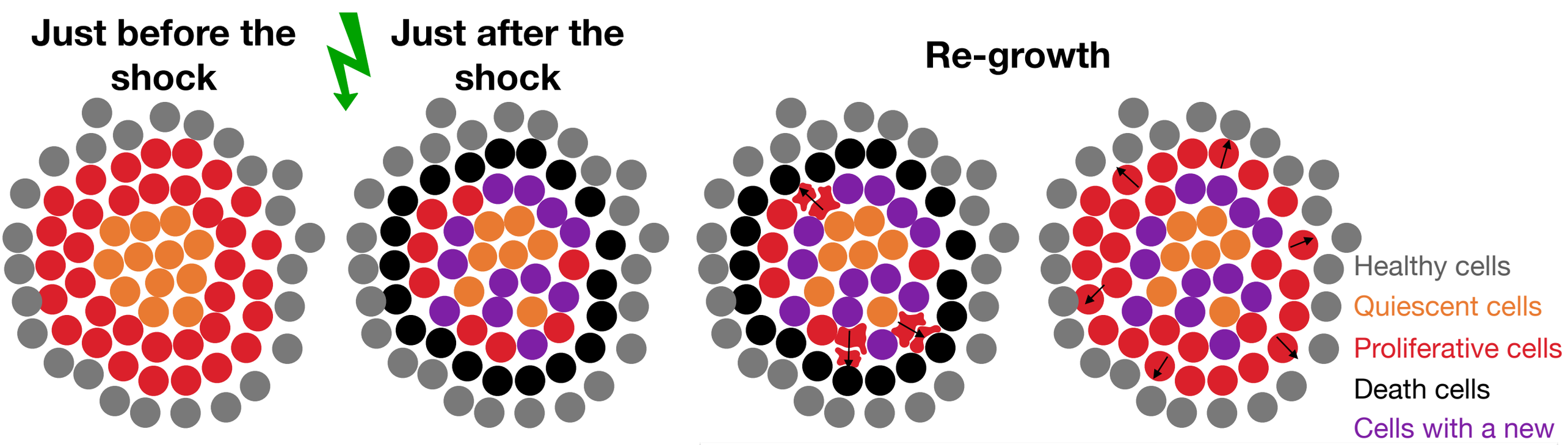
$$\tau^{PtoQ} = \tau_b - \frac{\tau_b - \tau_e}{1 + e^{\frac{(R(t)(1-r)-d)}{s}}}$$

τ_b, τ_e, d, s fixed with the literature

Observation

$$y = R$$

- Parameters to estimate: a, b, p, m, λ



A. Collin, H. Bruhier, J. Kolosnjaj, M. Golzio, M.-P. Rols, and C. Poignard. Spatial mechanistic modeling for prediction of 3D multicellular spheroids behavior upon exposure to high intensity pulsed electric fields. AIMS Bioengineering, 9(2):102–122, 2022.

Electroporation - State observer

- Luenberger observer system

$$\partial_t \hat{P} = -\tau^G(r^{-2}\hat{I} - r\hat{I}(t,1))\partial_r \hat{P} + \tau^G(1-\hat{F})(1-\hat{P}) - \hat{\tau}_{PtoQ}\hat{P}, \qquad [0,t_{last}]\times[0,1]$$

$$\partial_t \hat{F} = -\tau^G(r^{-2}\hat{I} - r\hat{I}(t,1))\partial_r \hat{F} - \tau^G(1-\hat{F})\hat{F}, \qquad [0,t_{last}]\times[0,1]$$

$$\hat{I} = \int_0^r (1-\hat{F})\underline{r}^2 d\underline{r} \qquad [0,t_{last}]\times[0,1]$$

$$\hat{R}' = \hat{R}\tau^G\hat{I}(t,1) - \boxed{\gamma_{obs}(t)(\hat{R}-R),} \qquad [0,t_{last}]$$

$$\hat{Q} = 1-(\hat{P}+\hat{F}), \qquad [0,t_{last}]\times[0,1]$$

High quiescent proportion of cells in the spheroid center

$$\tau^{PtoQ} = \tau_b - \frac{\tau_b - \tau_e}{1 + e^{\frac{(R(t)(1-r)-d)}{s}}}$$

τ_b, τ_e, d, s fixed with the literature

- Uncertainties reduced to the initial conditions

$$R(0) = R_0 + \xi_R, \quad \hat{R}(0) = R_0$$

$$P(0,r) = P_0 + \xi_P \in [0,1] \quad \hat{P}(0) = P_0 \in [0,1]$$

$$Q(0,r) = 1 - P(0,r) \quad \hat{Q}(0,r) = 1 - \hat{P}(0,r)$$

$$F(0,r) = 0 \quad \hat{F}(0) = 0$$

Electroporation - State observer

- Luenberger observer system

$$\partial_t \hat{P} = -\tau^G(r^{-2}\hat{I} - r\hat{I}(t,1))\partial_r \hat{P} + \tau^G(1-\hat{F})(1-\hat{P}) - \hat{\tau}_{PtoQ}\hat{P}, \qquad [0,t_{last}]\times[0,1]$$

$$\partial_t \hat{F} = -\tau^G(r^{-2}\hat{I} - r\hat{I}(t,1))\partial_r \hat{F} - \tau^G(1-\hat{F})\hat{F}, \qquad [0,t_{last}]\times[0,1]$$

$$\hat{I} = \int_0^r (1-\hat{F})\underline{r}^2 d\underline{r} \qquad [0,t_{last}]\times[0,1]$$

$$\hat{R}' = \hat{R}\tau^G\hat{I}(t,1) - \boxed{\gamma_{obs}(t)(\hat{R}-R),} \qquad [0,t_{last}]$$

$$\hat{Q} = 1-(\hat{P}+\hat{F}), \qquad [0,t_{last}]\times[0,1]$$

High quiescent proportion of cells in the spheroid center

$$\tau^{PtoQ} = \tau_b - \frac{\tau_b - \tau_e}{1 + e^{\frac{(R(t)(1-r)-d)}{s}}}$$

τ_b, τ_e, d, s fixed with the literature

Proposition - [*in a well-posed context*]

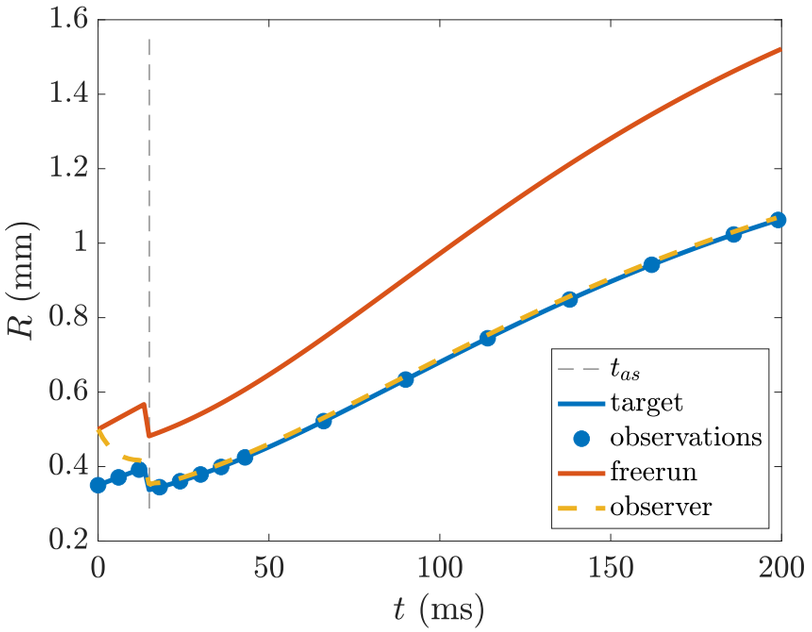
If $\gamma_{obs}^{min} > \frac{ma_{bs}}{3}$, the radius $t \mapsto \tilde{R}(t)$ converges exponentially to 0 when t goes to $+\infty$.

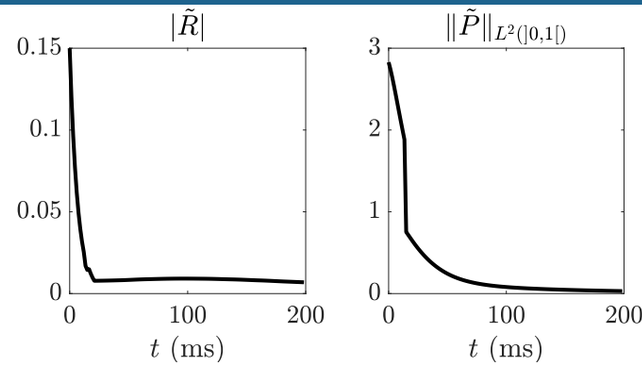
If $\gamma_{obs}^{min} > \frac{ma_{bs}}{3} + \tau_e$ and $\tilde{P}(0,\cdot) \in H^s(]0,r[)$, the norm $t \mapsto \|\tilde{P}(t,\cdot)\|^2_{L^2(]0,1[)}$ converges exponentially to 0 when t goes to $+\infty$.

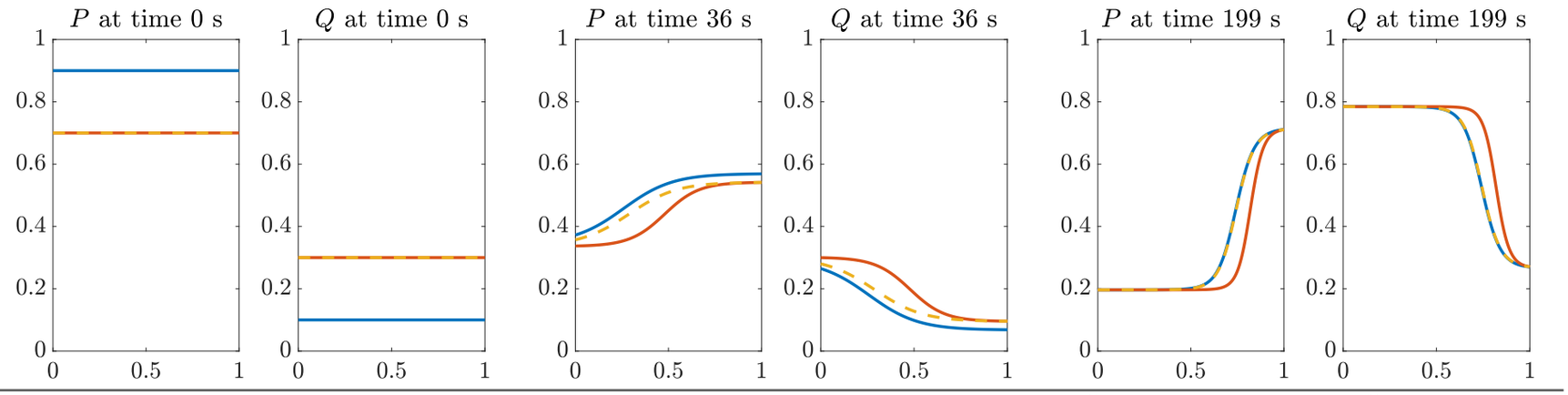
A. Collin. Population-based estimation for PDE systems – Applications in spheroids electroporation. Submitted preprint, 2023.

Electroporation - State observer

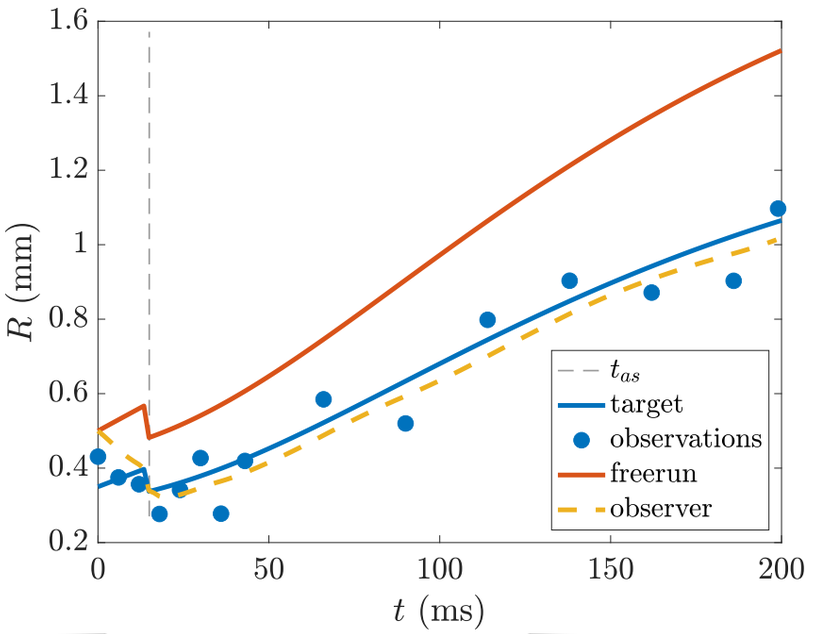
- Synthetic case - State observer - No observation error

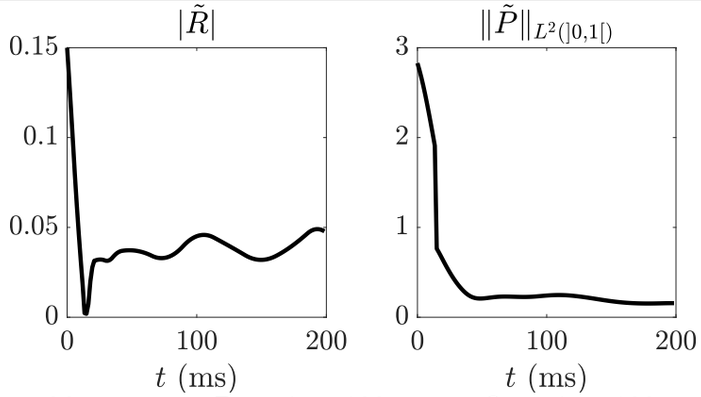


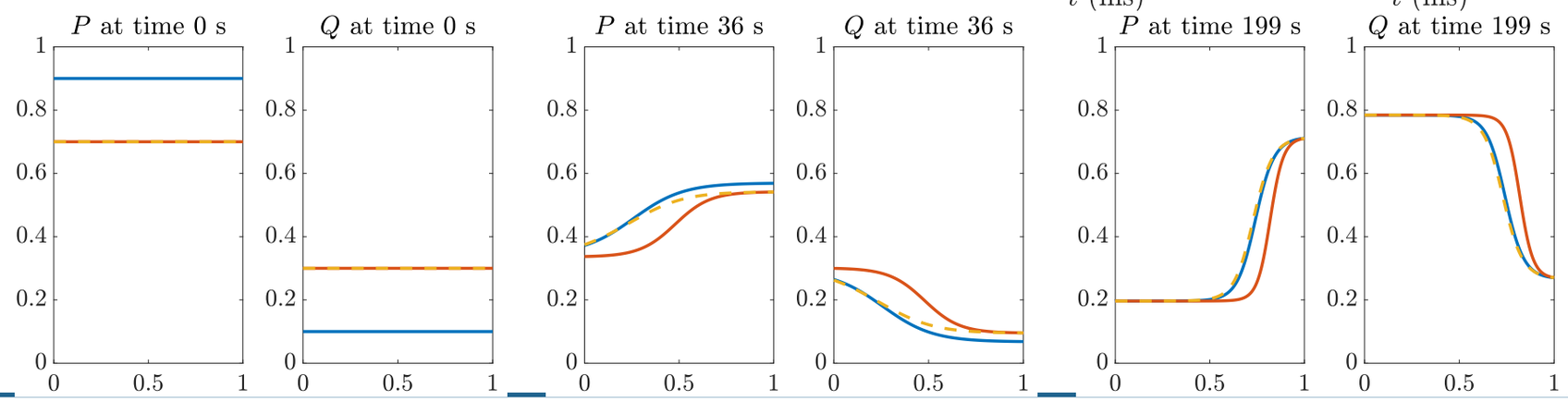




- Synthetic case - State observer - Observation error (std = 0.08)

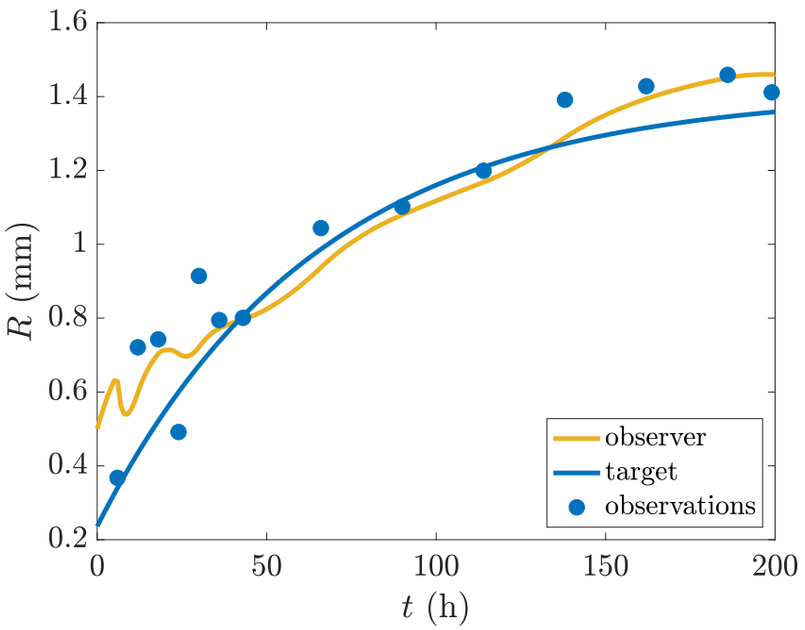


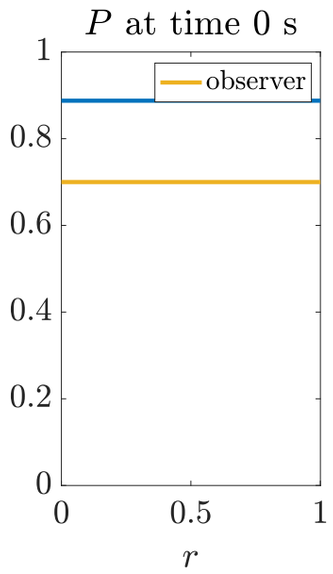


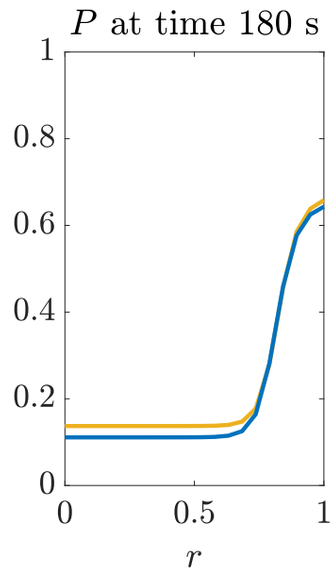


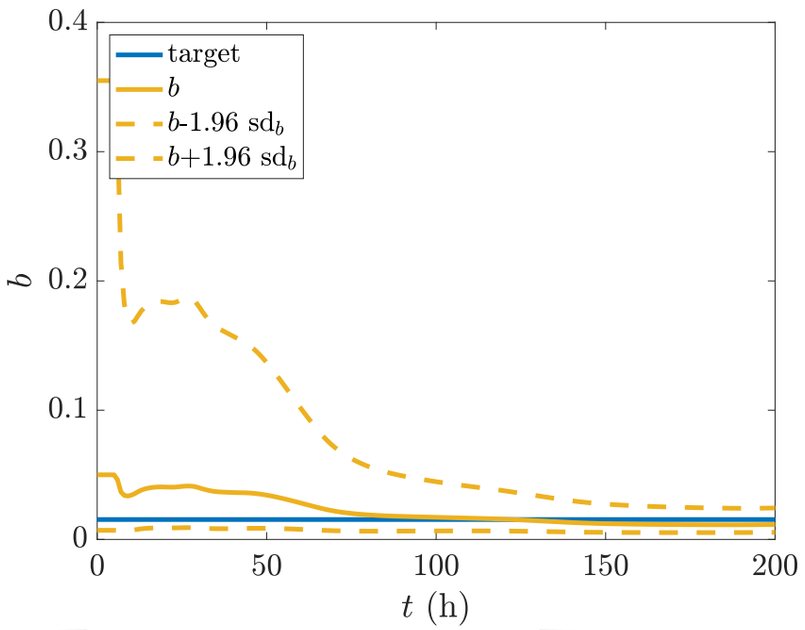
Electroporation - Joint state and parameter observer

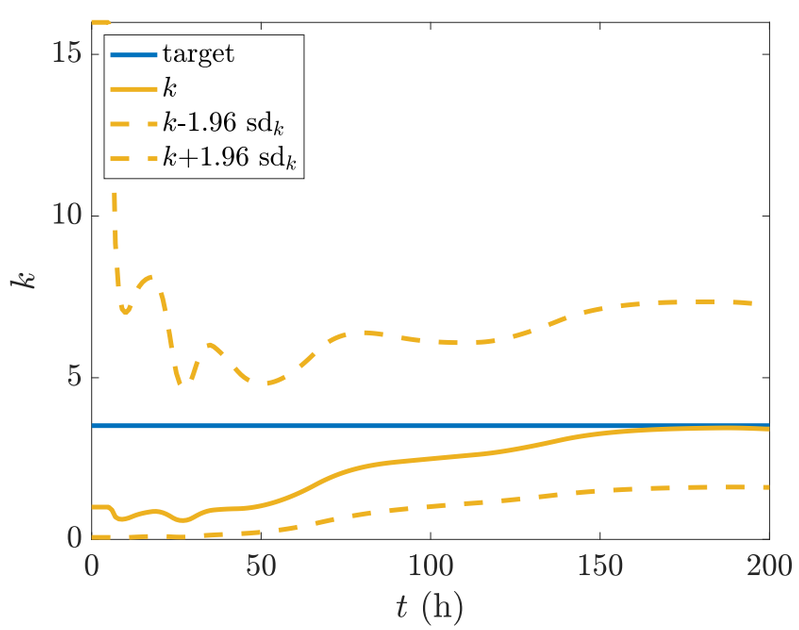
- Synthetic case (only free growth)
- Scenario: weak priors & false initial conditions with state observer











Luenberger state observer coupled with a Reduced-order Unscented Kalman Filter

Time t^n

Generate p particles around the mean value using the covariance reduced to parametric space

Prediction

Apply the stabilized model to this particle to compute one time step

Innovation

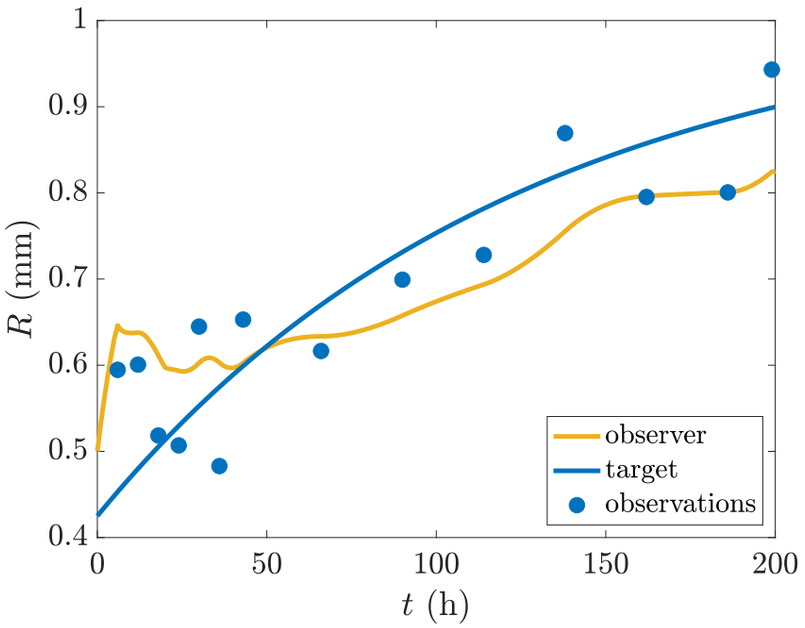
Compute the discrepancy w.r.t the observations for each particle

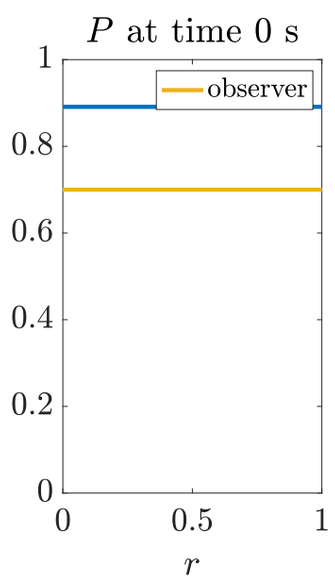
Correction

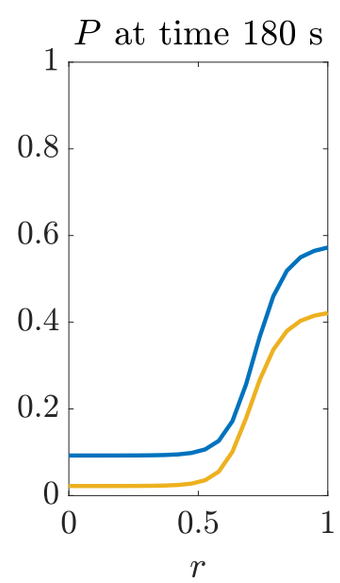
Gather the errors the parameter sensitivity hence the feedback correction

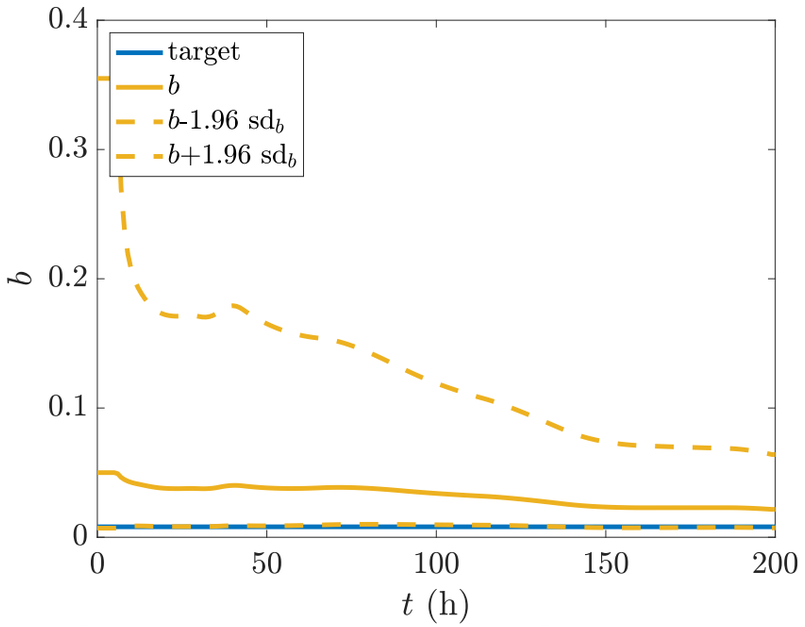
Electroporation - Joint state and parameter observer

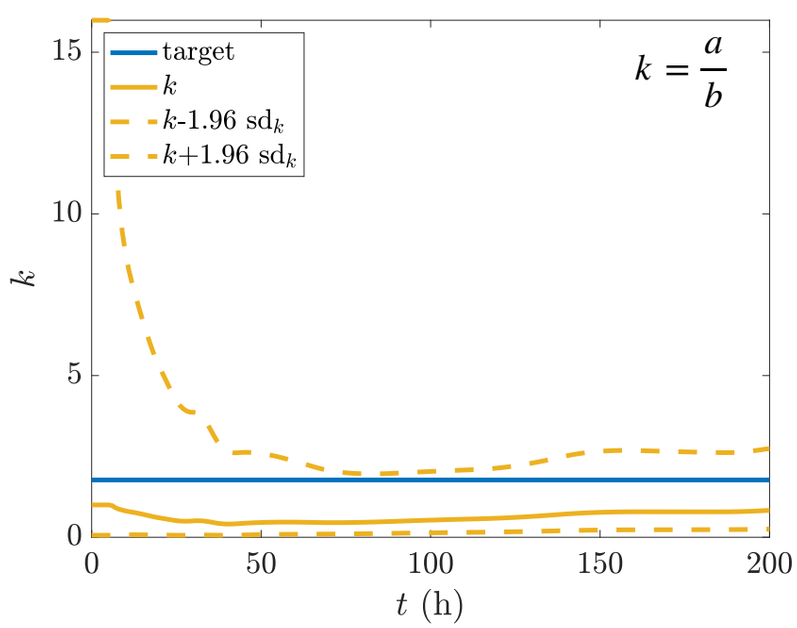
- Synthetic case (only free growth)
- Scenario: weak priors & false initial conditions with state observer











Luenberger state observer coupled with a Reduced-order Unscented Kalman Filter

Time t^n

Generate p particles around the mean value using the covariance reduced to parametric space

Prediction

Apply the stabilized model to this particle to compute one time step

Innovation

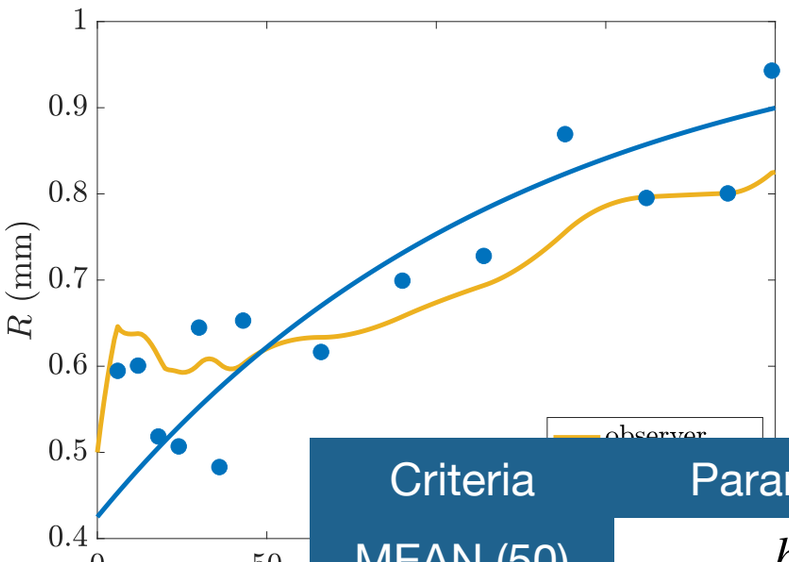
Compute the discrepancy w.r.t the observations for each particle

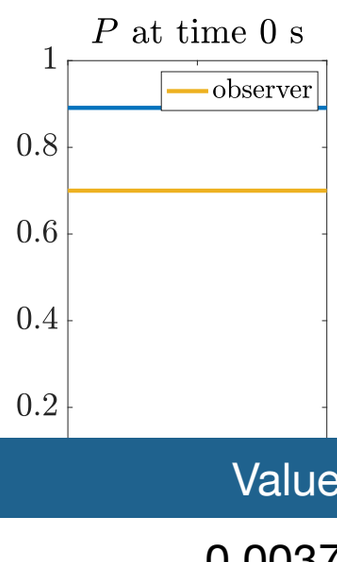
Correction

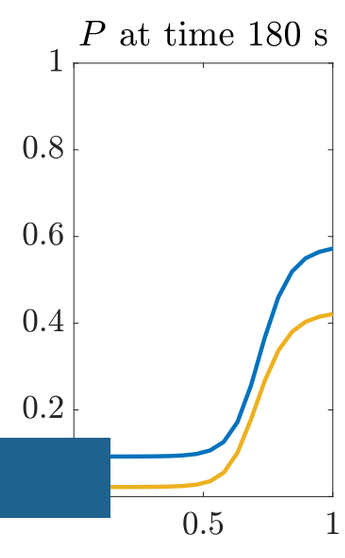
Gather the errors the parameter sensitivity hence the feedback correction

Electroporation - Joint state and parameter observer

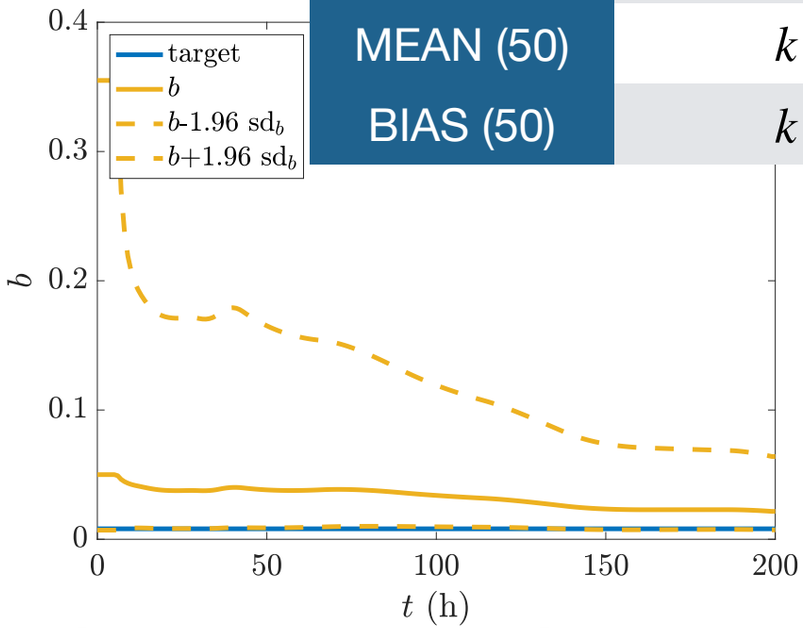
- Synthetic case (only free growth)
- Scenario: weak priors & false initial conditions with state observer

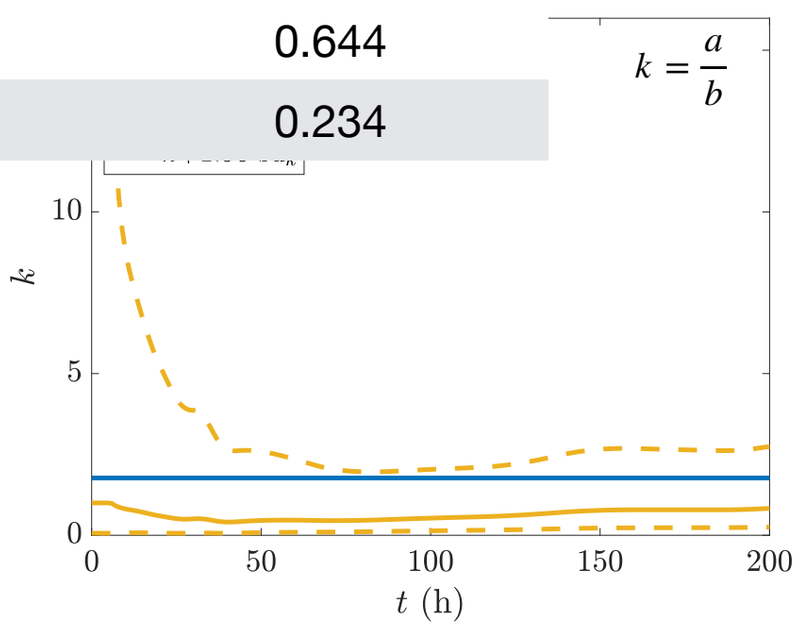






Criteria	Parameter	Value
MEAN (50)	b	0.00377
BIAS (50)	b	-0.00279
MEAN (50)	k	0.644
BIAS (50)	k	0.234





Luenberger state observer coupled with a Reduced-order Unscented Kalman Filter

Time t^n

Generate p particles around the mean value using the covariance reduced to parametric space

Prediction

Apply the stabilized model to this particle to compute one time step

Innovation

Compute the discrepancy w.r.t the observations for each particle

Correction

Gather the errors the parameter sensitivity hence the feedback correction

Population Kalman observer

- Population approach: compensate sparse sampling times and measurement uncertainties by constraining the variability of the parameters in the population.

- Classical strategy: mixed effect approach (in this context for a population of 5 spheroids, not so small space step, 2 parameters: ~80 minutes in an "equivalent" context).

A. Collin, M. Prague, and P. Moireau. Estimation for dynamical systems using a population-based Kalman filter–Applications in computational biology. MathematicS In Action, 2022.

Population Kalman observer

- Population approach: compensate sparse sampling times and measurement uncertainties by constraining the variability of the parameters in the population.
- Classical strategy: mixed effect approach (in this context for a population of 5 spheroids, not so small space step, 2 parameters: ~80 minutes in an "equivalent" context).
- Objective: develop a population Kalman observer inspiring from mixed effect approach.
- A population made of groups indexed by i: $\xi^i = \xi^{pop} + \tilde{\xi}^i$
- Corresponds to least-square minimisation when Gaussian laws are considered

$$\min_{\xi^{pop},\xi,\nu} J_T(\xi^{pop},\xi,\nu) = \frac{1}{2}\left[\xi^{pop}, M\,\xi^{pop}\right]$$

$$+\sum_{i=1}^{N_{pop}}\left[\frac{1}{2}\left[(\xi^i-\xi^{pop}),(\hat{P}_0)^{-1}(\xi^i-\xi^{pop})\right]+\frac{1}{2}\int_0^T\left[\nu^i(t),(Q(t)^i)^{-1}\nu^i(t)\right]dt+\frac{1}{2}\sum_{k=0}^{N_{T,obs}}\left[y_k^i-h(z^i(t_k)),(W_k^i)^{-1}(y_k^i-h(z^i(t_k))\right]\right]$$

$$\xi=\begin{pmatrix}\xi^1\\ \vdots\\ \xi^{N_{pop}}\end{pmatrix}$$

$$\dot{z}^i = F(t,z^i)+B\nu^i(t)$$

$$z^i(0) = z_0^i+\xi^i$$

A. Collin, M. Prague, and P. Moireau. Estimation for dynamical systems using a population-based Kalman filter–Applications in computational biology. MathematicS In Action, 2022.

Population Kalman observer

- Assuming that $\xi^{pop} = \sum \alpha^i \xi^i$ with $\sum \alpha^i = 1$, one can prove that

$$\min_{\xi^{pop},\xi,\nu} J_T(\xi^{pop},\xi,\nu) = \frac{1}{2}\left[\xi^{pop}, M\,\xi^{pop}\right] \qquad \xi = \begin{pmatrix} \xi^1 \\ \vdots \\ \xi^{N_{pop}} \end{pmatrix} \quad \cdots$$

$$+\sum_{i=1}^{N_{pop}} \left[\frac{1}{2}\left[(\xi^i - \xi^{pop}), (\hat{P}_0)^{-1}\,(\xi^i - \xi^{pop})\right] + \frac{1}{2}\int_0^T \left[\nu^i(t), (Q(t)^i)^{-1}\nu^i(t)\right] dt + \frac{1}{2}\sum_{k=0}^{N_{T,obs}} \left[y_k^i - h(z^i(t_k)), (W_k^i)^{-1}(y_k^i - h(z^i(t_k))\right]\right]$$

is equivalent to

$$\min_{\xi^{pop},\xi,\nu} J_T(\xi^{pop},\xi,\nu) = \left[\frac{1}{2}\left[\xi, (P_0^{pop})^{-1}\,\xi\right] + \frac{1}{2}\int_0^T \left[\nu(t), Q(t)^{-1}\nu(t)\right] dt + \frac{1}{2}\sum_{k=0}^{N_{T,obs}} \left[y^k - h(z(t_k)), (W_k)^{-1}(y_k - h(z(t_k))\right]\right]$$

where P_0^{pop} is an invertible matrix of dimension $N_{pop} \times N_z$ defined by

$$P_0^{pop} = \left(\frac{1}{N_{pop}^2}\overrightarrow{1}_{N_{pop}}\overrightarrow{1}_{N_{pop}}^T \otimes M + [I_{N_{pop}} - \frac{1}{N_{pop}}\overrightarrow{1}_{N_{pop}}\overrightarrow{1}_{N_{pop}}^T] \otimes P_0^{-1}\right)^{-1},$$

with $\overrightarrow{1}_{N_{pop}} = (1 \cdots 1)^T \in \mathbb{R}^{N_{pop}}$.

Population Kalman observer

- Assuming that $\xi^{pop} = \sum \alpha^i \xi^i$ with $\sum \alpha^i = 1$, one can prove that

$$\min_{\xi^{pop},\xi,\nu} J_T(\xi^{pop},\xi,\nu) = \frac{1}{2}\left[\xi^{pop}, M\,\xi^{pop}\right] \qquad \xi = \begin{pmatrix} \xi^1 \\ \vdots \\ \xi^{N_{pop}} \end{pmatrix} \quad \cdots$$

$$+\sum_{i=1}^{N_{pop}} \left[\frac{1}{2}\left[(\xi^i - \xi^{pop}), (\hat{P}_0)^{-1}\,(\xi^i - \xi^{pop})\right] + \frac{1}{2}\int_0^T \left[\nu^i(t), (Q(t)^i)^{-1}\nu^i(t)\right] dt + \frac{1}{2}\sum_{k=0}^{N_{T,obs}} \left[y_k^i - h(z^i(t_k)), (W_k^i)^{-1}(y_k^i - h(z^i(t_k))\right]\right]$$

is equivalent to

$$\min_{\xi^{pop},\xi,\nu} J_T(\xi^{pop},\xi,\nu) = \left[\frac{1}{2}\left[\xi, (P_0^{pop})^{-1}\,\xi\right] + \frac{1}{2}\int_0^T \left[\nu(t), Q(t)^{-1}\nu(t)\right] dt + \frac{1}{2}\sum_{k=0}^{N_{T,obs}} \left[y^k - h(z(t_k)), (W_k)^{-1}(y_k - h(z(t_k))\right]\right]$$

where P_0^{pop} is an invertible matrix of dimension $N_{pop} \times N_z$ defined by

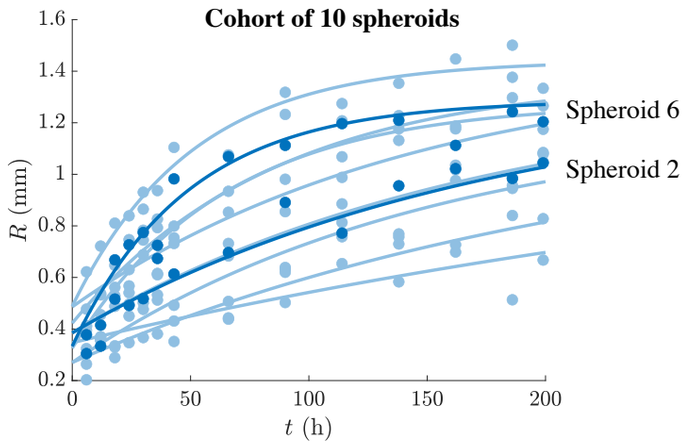
$$P_0^{pop} = \left(\frac{1}{N_{pop}^2}\overrightarrow{1}_{N_{pop}}\overrightarrow{1}_{N_{pop}}^T \otimes M + [I_{N_{pop}} - \frac{1}{N_{pop}}\overrightarrow{1}_{N_{pop}}\overrightarrow{1}_{N_{pop}}^T] \otimes P_0^{-1}\right)^{-1},$$

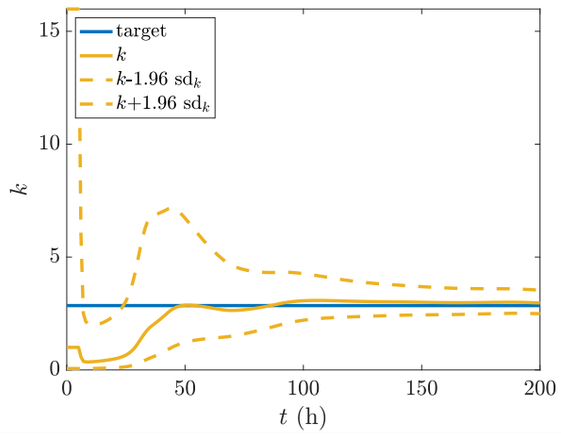
with $\overrightarrow{1}_{N_{pop}} = (1 \cdots 1)^T \in \mathbb{R}^{N_{pop}}$.

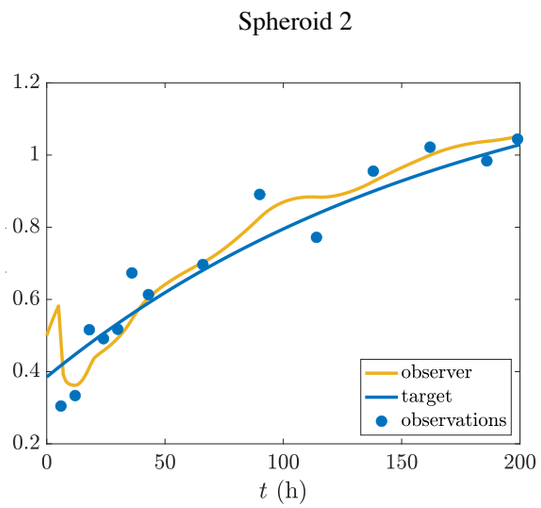
- The key of our uncertainty modeling is that P_0^{pop} couples the population members since

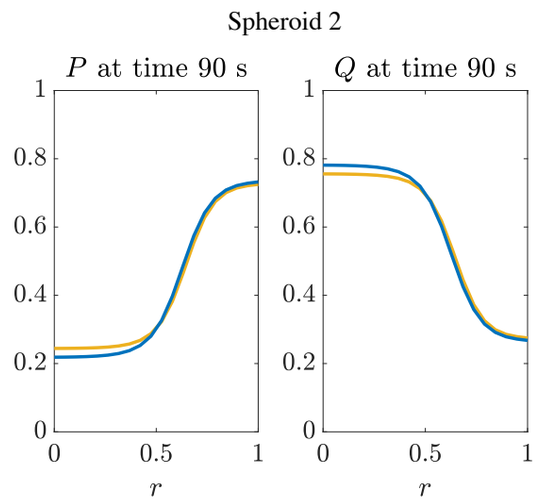
$$(P_0^{pop})^{-1} \neq \begin{pmatrix} P_0^{-1} & & \\ & \ddots & \\ & & P_0^{-1} \end{pmatrix}.$$

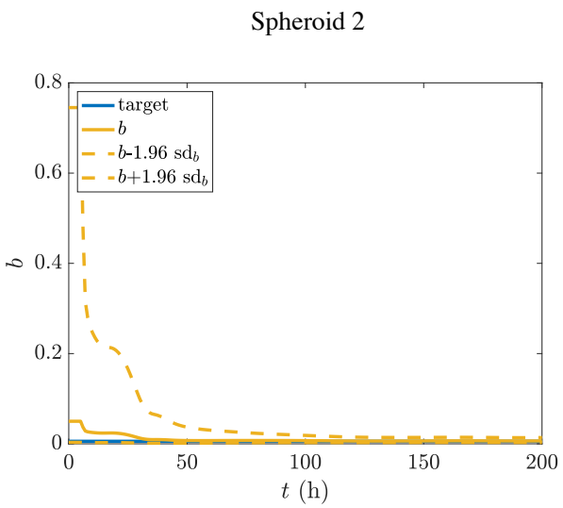
Validation on synthetic data

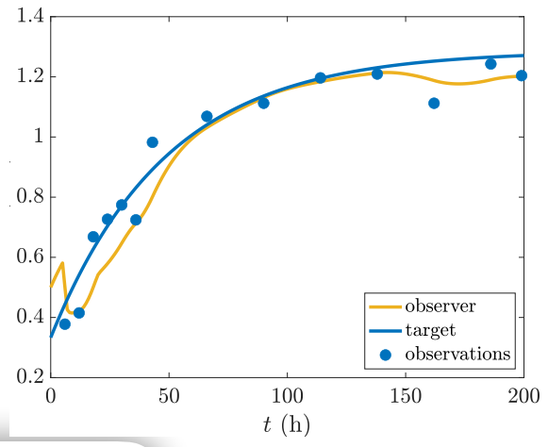


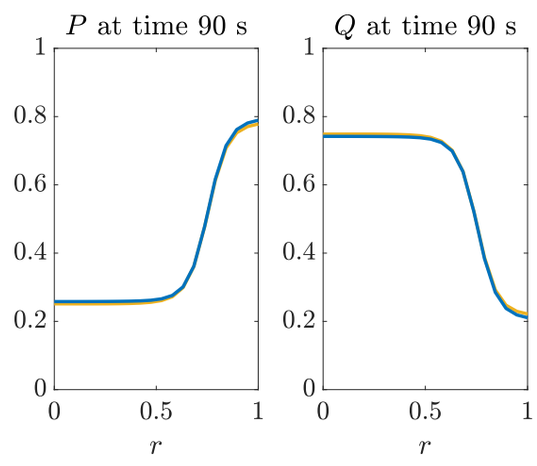


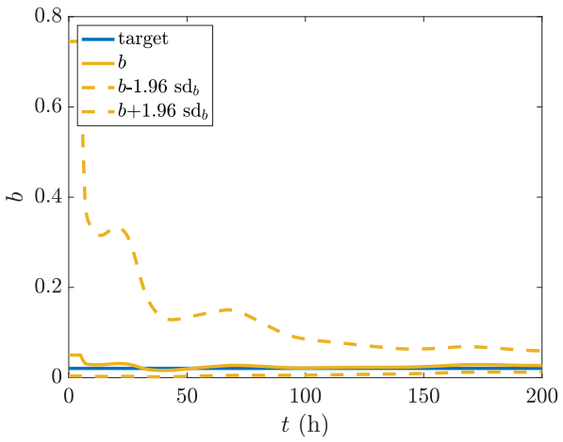






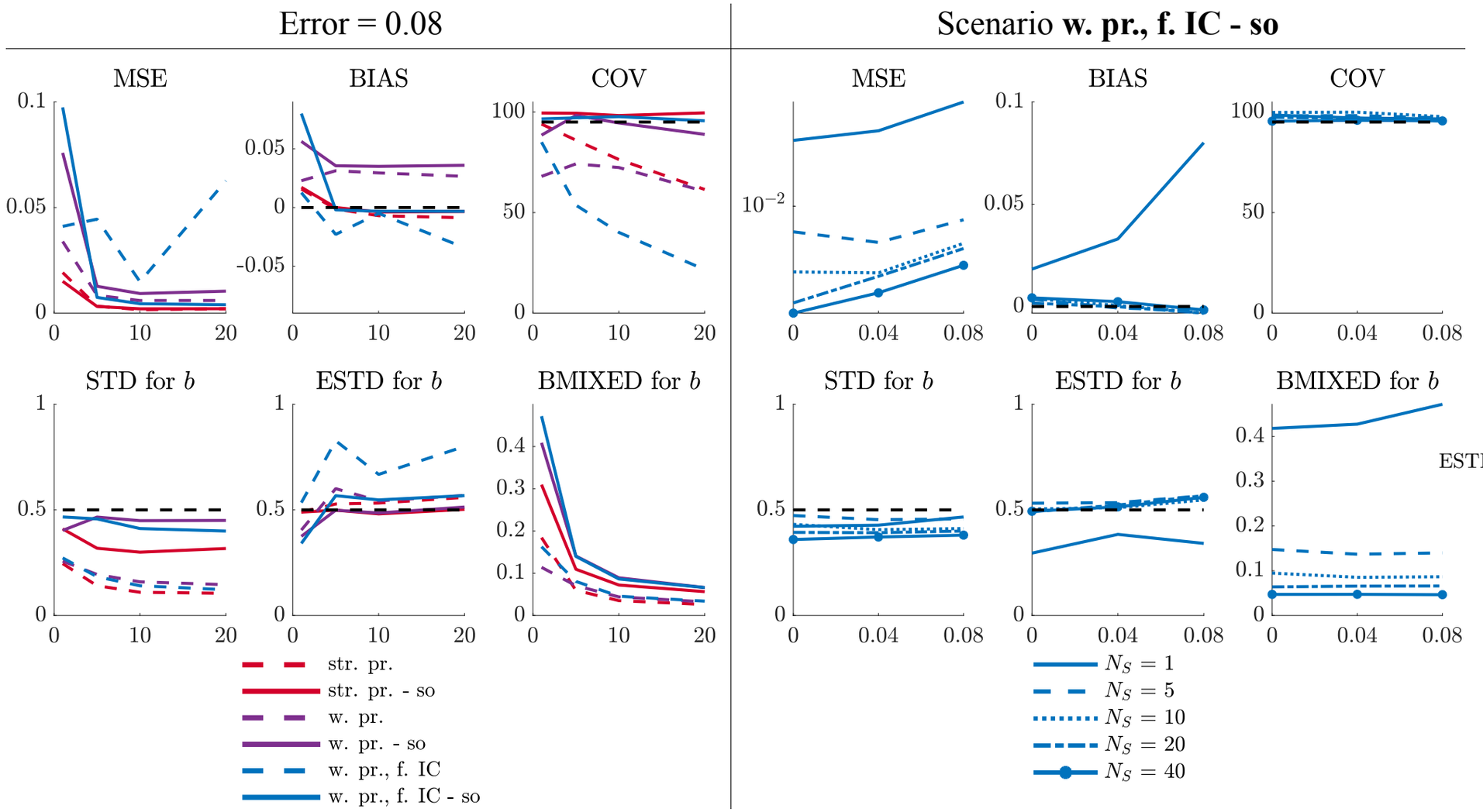






Validation on synthetic data

- Synthetic case
- 6 scenarios: strong priors (str. pr.), strong priors with state observer (str. pr. - so), weak priors (w. pr.), weak priors with state observer (w. pr. - so), weak priors & false initial conditions (w. pr., f. IC), weak priors & false initial conditions with state observer (w. pr., f. IC - so)



$$\text{MSE} = \frac{1}{N_R N_S} \sum_{r=1}^{N_R} \sum_{i=1}^{N_S} \left(\frac{\theta_t(r,i) - \theta_e(r,i)}{\theta_t(r,i)} \right)^2$$

$$\text{BIAS} = \frac{1}{N_R N_S} \sum_{r=1}^{N_R} \sum_{i=1}^{N_S} \frac{\theta_t(r,i) - \theta_e(r,i)}{|\theta_t(r,i)|}$$

$$\text{STD} = \frac{1}{N_R N_S} \sum_{r=1}^{N_R} \sum_{i=1}^{N_S} \sigma_e^{\theta}(r,i)$$

$$\text{ESTD} = \sqrt{\frac{1}{N_R N_S} \sum_{r=1}^{N_R} \sum_{i=1}^{N_S} \left(\theta_e(r,i) - \frac{1}{N_R N_S} \sum_{n=1}^{N_R} \sum_{i=1}^{N_S} \theta_e(r,i) \right)^2}$$

$$\text{BMIXED} = \frac{1}{N_R N_S} \sum_{r=1}^{N_R} \sum_{i=1}^{N_S} \left(\sigma_{e,m}^{\theta}(r,i) - \sigma_t^{\theta}(r,i) \right)$$

Comparing with existing strategy

- Computational times on synthetic data

Algo	Δ_x	$N_S = 1$	$N_S = 5$	$N_S = 10$	$N_S = 20$	$N_S = 40$
Algorithm	0.05	0.03	0.3	1	4	12
Algorithm	0.01	x	x	3.5	x	x
Algorithm	0.001	x	x	60	x	x
Function *nlmefitsa*	0.05	x	80	x	x	x

- Parameters estimation on real data

Case		EF500 (12 spheroids)	EF1000-B (5 spheroids)
λ	Function *nlmefitsa*	0.22 (mean) - 0.092 (std)	0.37 (mean) - 0.012 (std)
	Algo	0.37 (mean) - 0.035 (std)	0.36 (mean) - 0.045 (std)
p	Function *nlmefitsa*	0.89 (mean) - 0.014 (std)	0.82 (mean) - 0.086 (std)
	Algo	0.90 (mean) - 0.0028 (std)	0.86 (mean) - 0.012 (std)
m	Function *nlmefitsa*	1.11 (mean) - 0.024 (std)	1.40 (mean) - 0.16 (std)
	Algo	1.14 (mean) - 0.18 (std)	1.21 (mean) - 0.16 (std)

Case		EF1000-A (6 spheroids)	EF-2000 (14 spheroids)
λ	Function *nlmefitsa*	0.77 (mean) - 0.10 (std)	0.97 (mean) - 0.065 (std)
	Algo	0.69 (mean) - 0.04 (std)	0.92 (mean) - 0.0015 (std)
p	Function *nlmefitsa*	0.17 (mean) - 0.010 (std)	0.0016 (mean) - 9×10^{-5} (std)
	Algo	0.11 (mean) - 0.015 (std)	0.00097 (mean) - 2.1×10^{-6} (std)
m	Function *nlmefitsa*	5.19 (mean) - 1.02 (std)	x
	Algo	5.3 (mean) - 0.77 (std)	x

- λ percent of cells whose the metabolism is modified
- p percent of cells whose are not destroyed
- m boost of resumption

Electroporation - Results

t = 0h t = 21h t = 42.1h t = 63.1h t = 84.2h t = 105h t = 126h t = 147h t = 168h t = 189h t = 210h

Free growth

t = 0h t = 21h t = 42.1h t = 63.1h t = 84.2h t = 105h t = 126h t = 147h t = 168h t = 189h t = 210h

EF = 500 V.cm-1

t = 0h t = 21h t = 42.1h t = 63.1h t = 84.2h t = 105h t = 126h t = 147h t = 168h t = 189h t = 210h

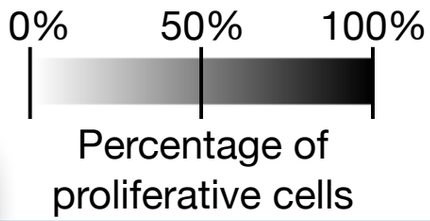
EF = 1000 V.cm-1 (Group B)

t = 0h t = 21h t = 42.1h t = 63.1h t = 84.2h t = 105h t = 126h t = 147h t = 168h t = 189h t = 210h

EF = 1000 V.cm-1 (Group A)

t = 0h t = 21h t = 42.1h t = 63.1h t = 84.2h t = 105h t = 126h t = 147h t = 168h t = 189h t = 210h

EF = 2000 V.cm-1



Collin, A et al. Spatial mechanistic modeling for prediction of 3D multicellular spheroids behavior upon exposure to high intensity pulsed electric fields. Bioengineering 2022.

Thank you for your attention!