# Adaptive parameter-space exploration for online Bayesian inference

Sara Pérez Vieites

Aalto University
Finnish Center for Artificial Intelligence (FCAI)

BAYSM 2024

29 June

Joint work with Joaquín Míguez (Universidad Carlos III de Madrid) and Víctor Elvira (University of Edinburgh).

# Index

# State-space model

We are interested in systems can be represented by **Markov state-space dynamical models**:

$$
\begin{aligned}
\boldsymbol{x}_t &= \boldsymbol{f}(\boldsymbol{x}_{t-1}, \boldsymbol{\theta}) + \boldsymbol{v}_t, \\
\boldsymbol{y}_t &= \boldsymbol{g}(\boldsymbol{x}_t, \boldsymbol{\theta}) + \boldsymbol{r}_t,
\end{aligned}
$$

- $\boldsymbol{f}$, $\boldsymbol{g}$: state transition function and observation function
- $\boldsymbol{v}_t$, $\boldsymbol{r}_t$: state and observation noises

In terms of a set of **relevant probability density functions (pdfs)**:

- Prior pdfs: $\theta \sim p(\theta)$ and $\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0)$
- Transition pdf of the state: $\boldsymbol{x}_t \sim p(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}, \theta)$
- Conditional pdf of the observation: $\boldsymbol{y}_t \sim p(\boldsymbol{y}_t | \boldsymbol{x}_t, \theta)$

# State-space model

We are interested in systems can be represented by **Markov state-space dynamical models**:

$$\begin{aligned} \boldsymbol{x}_t &= \boldsymbol{f}(\boldsymbol{x}_{t-1}, \boldsymbol{\theta}) + \boldsymbol{v}_t, \\ \boldsymbol{y}_t &= \boldsymbol{g}(\boldsymbol{x}_t, \boldsymbol{\theta}) + \boldsymbol{r}_t, \end{aligned}$$

- $\boldsymbol{f}$, $\boldsymbol{g}$: state transition function and observation function
- $\boldsymbol{v}_t$, $\boldsymbol{r}_t$: state and observation noises

In terms of a set of **relevant probability density functions (pdfs)**:

- Prior pdfs: $\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$ and $\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0)$
- Transition pdf of the state: $\boldsymbol{x}_t \sim p(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}, \boldsymbol{\theta})$
- Conditional pdf of the observation: $\boldsymbol{y}_t \sim p(\boldsymbol{y}_t | \boldsymbol{x}_t, \boldsymbol{\theta})$

# State estimation

Classical filtering methods:

**Bayesian estimation of the state variables**, $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}, \theta^\star)$, assuming $\boldsymbol{\theta}^\star$ is known.

Every time step $t$:

1. Predictive distribution:

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \theta^\star) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \theta^\star)p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1}, \theta^\star)d\boldsymbol{x}_t \quad (1)$$

2. Likelihood: $p(\boldsymbol{y}_t|\boldsymbol{x}_t, \theta^\star)$
3. Posterior/filtering distribution:

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}, \theta^\star) \propto (\boldsymbol{y}_t|\boldsymbol{x}_t, \theta^\star)p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \theta^\star) \quad (2)$$

In practice, $\theta^\star$ is not known. It is needed to **estimate both $\theta$ and $\boldsymbol{x}_t$**, i.e., $p(\boldsymbol{x}_t, \theta|\boldsymbol{y}_{1:t})$.

# State estimation

Classical filtering methods:

**Bayesian estimation of the state variables**, $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}, \boldsymbol{\theta}^\star)$, assuming $\boldsymbol{\theta}^\star$ is known.

Every time step $t$:

1. Predictive distribution:

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}^\star) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{\theta}^\star) p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}^\star) d\boldsymbol{x}_t \quad (1)$$

2. Likelihood: $p(\boldsymbol{y}_t|\boldsymbol{x}_t, \theta^\star)$
3. Posterior/filtering distribution:

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}, \theta^\star) \propto (\boldsymbol{y}_t|\boldsymbol{x}_t, \theta^\star) p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \theta^\star) \quad (2)$$

In practice, $\theta^\star$ is not known. It is needed to **estimate both** $\theta$ **and** $\boldsymbol{x}_t$, i.e., $p(\boldsymbol{x}_t, \theta|\boldsymbol{y}_{1:t})$.

# State estimation

Classical filtering methods:

**Bayesian estimation of the state variables**, $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t},\boldsymbol{\theta}^\star)$, assuming $\boldsymbol{\theta}^\star$ is known.

Every time step $t$:

1. Predictive distribution:

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1},\boldsymbol{\theta}^\star) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1},\boldsymbol{\theta}^\star)p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1},\boldsymbol{\theta}^\star)d\boldsymbol{x}_t \quad (1)$$

2. Likelihood: $p(\boldsymbol{y}_t|\boldsymbol{x}_t,\boldsymbol{\theta}^\star)$
3. Posterior/filtering distribution:

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t},\boldsymbol{\theta}^\star) \propto (\boldsymbol{y}_t|\boldsymbol{x}_t,\boldsymbol{\theta}^\star)p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1},\boldsymbol{\theta}^\star) \quad (2)$$

In practice, $\theta^\star$ is not known. It is needed to **estimate both** $\theta$ **and** $x_t$, i.e., $p(x_t,\theta|y_{1:t})$.

# State estimation

Classical filtering methods:

**Bayesian estimation of the state variables**, $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}, \boldsymbol{\theta}^\star)$, assuming $\boldsymbol{\theta}^\star$ is known.

Every time step $t$:

1. Predictive distribution:

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}^\star) = \int p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{\theta}^\star) p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}^\star) d\boldsymbol{x}_t \quad (1)$$

2. Likelihood: $p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{\theta}^\star)$
3. Posterior/filtering distribution:

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}, \boldsymbol{\theta}^\star) \propto (\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{\theta}^\star) p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}^\star) \quad (2)$$

In practice, $\boldsymbol{\theta}^\star$ is not known. It is needed to **estimate both $\theta$ and $\boldsymbol{x}_t$**, i.e., $p(\boldsymbol{x}_t, \theta|\boldsymbol{y}_{1:t})$.

# State-of-the-art methods

Methods for Bayesian inference of both $\boldsymbol{\theta}$ and $\boldsymbol{x}_t$:

- **particle Markov chain Monte Carlo (PMCMC)**[1]
- **sequential Monte Carlo square (SMC$^2$)**[2]
- **nested particle filters (NPFs)**[3]

⟶ They can quantify the uncertainty or estimation error.
⟶ They can be applied to a broad class of models.
⟶ They provide theoretical guarantees.
⟶ Both PMCMC and SMC$^2$ are batch techniques, while the NPF is a recursive method.

[1] Andrieu, Doucet, and Holenstein 2010.
[2] Chopin, Jacob, and Papaspiliopoulos 2013.
[3] Crisan and Míguez 2018.

# State-of-the-art methods

Methods for Bayesian inference of both $\boldsymbol{\theta}$ and $\boldsymbol{x}_t$:

- **particle Markov chain Monte Carlo (PMCMC)**[1]
- **sequential Monte Carlo square (SMC$^2$)**[2]
- **nested particle filters (NPFs)**[3]

⟶ They can quantify the uncertainty or estimation error.
⟶ They can be applied to a broad class of models.
⟶ They provide theoretical guarantees.
⟶ Both PMCMC and SMC$^2$ are batch techniques, while the NPF is a recursive method.

[1] Andrieu, Doucet, and Holenstein 2010.
[2] Chopin, Jacob, and Papaspiliopoulos 2013.
[3] Crisan and Míguez 2018.

# State-of-the-art methods

Methods for Bayesian inference of both $\boldsymbol{\theta}$ and $\boldsymbol{x}_t$:

- **particle Markov chain Monte Carlo (PMCMC)**[1]
- **sequential Monte Carlo square (SMC$^2$)**[2]
- **nested particle filters (NPFs)**[3]

⟶ They can quantify the uncertainty or estimation error.
⟶ They can be applied to a broad class of models.
⟶ They provide theoretical guarantees.
⟶ Both PMCMC and SMC$^2$ are batch techniques, while the NPF is a recursive method.

[1] Andrieu, Doucet, and Holenstein 2010.
[2] Chopin, Jacob, and Papaspiliopoulos 2013.
[3] Crisan and Míguez 2018.

# Index

# Model inference

We aim at computing the joint posterior pdf $p(\boldsymbol{\theta}, \boldsymbol{x}_t | \boldsymbol{y}_{1:t})$, that can be written as

$$p(\boldsymbol{x}_t, \boldsymbol{\theta} | \boldsymbol{y}_{1:t}) = \underbrace{p(\boldsymbol{x}_t | \boldsymbol{\theta}, \boldsymbol{y}_{1:t})}_{2^{nd}\ \text{layer}} \underbrace{p(\boldsymbol{\theta} | \boldsymbol{y}_{1:t})}_{1^{st}\ \text{layer}}$$

$\longrightarrow$ The **key difficulty** in this class of models is the Bayesian estimation of the parameter vector $\boldsymbol{\theta}$.

# Model inference

At every time step $t$:

1$^{st}$ layer

$\underbrace{p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})}_{\text{Pred. pdf of } \boldsymbol{\theta}}$

2$^{nd}$ layer

$p(\boldsymbol{y}_t|\boldsymbol{\theta}, \boldsymbol{y}_{1:t-1})$

$$\underbrace{p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})}_{\text{Post. pdf of } \boldsymbol{\theta}} \propto p(\boldsymbol{y}_t|\boldsymbol{\theta}, \boldsymbol{y}_{1:t-1})p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})$$

# Model inference

At every time step $t$:

1st layer

$$\underbrace{p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})}_{\text{Pred. pdf of } \boldsymbol{\theta}}$$

2nd layer

$$p(\boldsymbol{y}_t|\boldsymbol{\theta},\boldsymbol{y}_{1:t-1}) = \int p(\boldsymbol{y}_t|\boldsymbol{x}_t,\boldsymbol{\theta})p(\boldsymbol{x}_t|\boldsymbol{\theta},\boldsymbol{y}_{1:t-1})d\boldsymbol{x}_t$$

$$\underbrace{p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})}_{\text{Post. pdf of } \boldsymbol{\theta}} \propto p(\boldsymbol{y}_t|\boldsymbol{\theta},\boldsymbol{y}_{1:t-1})p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})$$

# Model inference

At every time step $t$:

**1st layer**

$\underbrace{p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})}_{\text{Pred. pdf of } \boldsymbol{\theta}}$

Filtering (given $\boldsymbol{\theta}$)

**2nd layer**

Predictive pdf of $\boldsymbol{x}$: $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta})$

Likelihood: $p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{\theta})$

Posterior pdf of $\boldsymbol{x}$: $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}, \boldsymbol{\theta})$

$$p(\boldsymbol{y}_t|\boldsymbol{\theta}, \boldsymbol{y}_{1:t-1}) = \int p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{\theta}) p(\boldsymbol{x}_t|\boldsymbol{\theta}, \boldsymbol{y}_{1:t-1}) d\boldsymbol{x}_t$$

$$\underbrace{p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})}_{\text{Post. pdf of } \boldsymbol{\theta}} \propto p(\boldsymbol{y}_t|\boldsymbol{\theta}, \boldsymbol{y}_{1:t-1}) p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})$$

# Model inference

At every time step $t$:

1st layer

$\underbrace{p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})}_{\text{Pred. pdf of } \boldsymbol{\theta}}$

Filtering (given $\boldsymbol{\theta}$)

Predictive pdf of $\boldsymbol{x}$: $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta})$

Likelihood: $p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{\theta})$

Posterior pdf of $\boldsymbol{x}$: $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}, \boldsymbol{\theta})$

2nd layer

$$p(\boldsymbol{y}_t|\boldsymbol{\theta}, \boldsymbol{y}_{1:t-1}) = \int p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{\theta}) p(\boldsymbol{x}_t|\boldsymbol{\theta}, \boldsymbol{y}_{1:t-1}) d\boldsymbol{x}_t$$

$$\underbrace{p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})}_{\text{Post. pdf of } \boldsymbol{\theta}} \propto p(\boldsymbol{y}_t|\boldsymbol{\theta}, \boldsymbol{y}_{1:t-1}) p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})$$

# Naive importance sampling approximation

Initialisation: Draw $\{\boldsymbol{\theta}^i\}_{i=1}^N$ from $p(\boldsymbol{\theta})$

At $t \geq 1$ and for every $\boldsymbol{\theta}^i$, $i = 1, \ldots, N$:

SMC ($N$ samples) to approximate $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$

For $j = 1, \ldots, M$:

SMC ($M$ samples) to approximate $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}^i)$

- Draw $\bar{\boldsymbol{x}}_t^{i,j} \sim p(\boldsymbol{x}_t|\boldsymbol{\theta}^i, \boldsymbol{y}_{1:t-1})$
- Weights: $\tilde{u}_t^{i,j} \propto p(\boldsymbol{y}_t|\bar{\boldsymbol{x}}_t^{i,j}, \boldsymbol{\theta}^i)$
- Resampling: for $m = 1, \ldots, M$, $\tilde{\boldsymbol{x}}_t^{i,j} = \bar{\boldsymbol{x}}_t^{i,m}$ with prob. $u_t^{i,m} = \frac{\tilde{u}_t^{i,m}}{\sum_{j=1}^M \tilde{u}_t^{i,j}}$

- Likelihood of $\boldsymbol{\theta}^i$: $\tilde{w}_t^i = w_{t-1}^i \left( \frac{1}{M} \sum_{j=1}^M \tilde{u}_t^{i,j} \right)$
- Then, $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t}) = \sum_{i=1}^N w_t^i \delta_{\boldsymbol{\theta}^i}(d\boldsymbol{\theta})$, with $w_t^i = \frac{\tilde{w}_t^i}{\sum_{i=1}^N \tilde{w}_t^i}$.

# Naive importance sampling approximation

Initialisation: Draw $\{\boldsymbol{\theta}^i\}_{i=1}^N$ from $p(\boldsymbol{\theta})$

At $t \geq 1$ and for every $\boldsymbol{\theta}^i$, $i = 1, \ldots, N$:

SMC ($N$ samples) to approximate $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$

For $j = 1, \ldots, M$:

SMC ($M$ samples) to approximate $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}^i)$

- Draw $\bar{\boldsymbol{x}}_t^{i,j} \sim p(\boldsymbol{x}_t|\boldsymbol{\theta}^i, \boldsymbol{y}_{1:t-1})$
- Weights: $\tilde{u}_t^{i,j} \propto p(\boldsymbol{y}_t|\bar{\boldsymbol{x}}_t^{i,j}, \boldsymbol{\theta}^i)$
- Resampling: for $m = 1, \ldots, M$, $\tilde{\boldsymbol{x}}_t^{i,j} = \bar{\boldsymbol{x}}_t^{i,m}$ with prob. $u_t^{i,m} = \frac{\tilde{u}_t^{i,m}}{\sum_{j=1}^M \tilde{u}_t^{i,j}}$

- Likelihood of $\boldsymbol{\theta}^i$: $\tilde{w}_t^i = w_{t-1}^i \left( \frac{1}{M} \sum_{j=1}^M \tilde{u}_t^{i,j} \right)$
- Then, $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t}) = \sum_{i=1}^N w_t^i \delta_{\boldsymbol{\theta}^i}(d\boldsymbol{\theta})$, with $w_t^i = \frac{\tilde{w}_t^i}{\sum_{i=1}^N \tilde{w}_t^i}$.

# Naive importance sampling approximation

Initialisation: Draw $\{\boldsymbol{\theta}^i\}_{i=1}^N$ from $p(\boldsymbol{\theta})$

At $t \geq 1$ and for every $\boldsymbol{\theta}^i$, $i = 1, \ldots, N$:

SMC ($N$ samples) to approximate $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$

For $j = 1, \ldots, M$:

SMC ($M$ samples) to approximate $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}^i)$

- Draw $\bar{\boldsymbol{x}}_t^{i,j} \sim p(\boldsymbol{x}_t|\boldsymbol{\theta}^i, \boldsymbol{y}_{1:t-1})$
- Weights: $\tilde{u}_t^{i,j} \propto p(\boldsymbol{y}_t|\bar{\boldsymbol{x}}_t^{i,j}, \boldsymbol{\theta}^i)$
- Resampling: for $m = 1, \ldots, M$, $\tilde{\boldsymbol{x}}_t^{i,j} = \bar{\boldsymbol{x}}_t^{i,m}$ with prob. $u_t^{i,m} = \frac{\tilde{u}_t^{i,m}}{\sum_{j=1}^M \tilde{u}_t^{i,j}}$

- Likelihood of $\boldsymbol{\theta}^i$: $\tilde{w}_t^i = w_{t-1}^i \left( \frac{1}{M} \sum_{j=1}^M \tilde{u}_t^{i,j} \right)$
- Then, $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t}) = \sum_{i=1}^N w_t^i \delta_{\boldsymbol{\theta}^i}(d\boldsymbol{\theta})$, with $w_t^i = \frac{\tilde{w}_t^i}{\sum_{i=1}^N \tilde{w}_t^i}$.

# Naive importance sampling approximation

Initialisation: Draw $\{\boldsymbol{\theta}^i\}_{i=1}^N$ from $p(\boldsymbol{\theta})$

At $t \geq 1$ and for every $\boldsymbol{\theta}^i$, $i = 1, \ldots, N$:

SMC ($N$ samples) to approximate $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$

For $j = 1, \ldots, M$:

SMC ($M$ samples) to approximate $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}^i)$

- Draw $\bar{\boldsymbol{x}}_t^{i,j} \sim p(\boldsymbol{x}_t|\boldsymbol{\theta}^i, \boldsymbol{y}_{1:t-1})$
- Weights: $\tilde{u}_t^{i,j} \propto p(\boldsymbol{y}_t|\bar{\boldsymbol{x}}_t^{i,j}, \boldsymbol{\theta}^i)$
- Resampling: for $m = 1, \ldots, M$, $\tilde{\boldsymbol{x}}_t^{i,j} = \bar{\boldsymbol{x}}_t^{i,m}$ with prob. $u_t^{i,m} = \frac{\tilde{u}_t^{i,m}}{\sum_{j=1}^M \tilde{u}_t^{i,j}}$

- Likelihood of $\boldsymbol{\theta}^i$: $\tilde{w}_t^i = w_{t-1}^i \left( \frac{1}{M} \sum_{j=1}^M \tilde{u}_t^{i,j} \right)$
- Then, $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t}) = \sum_{i=1}^N w_t^i \delta_{\boldsymbol{\theta}^i}(d\boldsymbol{\theta})$, with $w_t^i = \frac{\tilde{w}_t^i}{\sum_{i=1}^N \tilde{w}_t^i}$.

# Naive importance sampling approximation

Initialisation: Draw $\{\boldsymbol{\theta}^i\}_{i=1}^N$ from $p(\boldsymbol{\theta})$

At $t \geq 1$ and for every $\boldsymbol{\theta}^i$, $i = 1, \ldots, N$:

SMC ($N$ samples) to approximate $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$

For $j = 1, \ldots, M$:

SMC ($M$ samples) to approximate $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}^i)$

- Draw $\bar{\boldsymbol{x}}_t^{i,j} \sim p(\boldsymbol{x}_t|\boldsymbol{\theta}^i, \boldsymbol{y}_{1:t-1})$
- Weights: $\tilde{u}_t^{i,j} \propto p(\boldsymbol{y}_t|\bar{\boldsymbol{x}}_t^{i,j}, \boldsymbol{\theta}^i)$
- Resampling: for $m = 1, \ldots, M$, $\tilde{\boldsymbol{x}}_t^{i,j} = \bar{\boldsymbol{x}}_t^{i,m}$ with prob. $u_t^{i,m} = \frac{\tilde{u}_t^{i,m}}{\sum_{j=1}^M \tilde{u}_t^{i,j}}$

- Likelihood of $\boldsymbol{\theta}^i$: $\tilde{w}_t^i = w_{t-1}^i \left( \frac{1}{M} \sum_{j=1}^M \tilde{u}_t^{i,j} \right)$
- Then, $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t}) = \sum_{i=1}^N w_t^i \delta_{\boldsymbol{\theta}^i}(d\boldsymbol{\theta})$, with $w_t^i = \frac{\tilde{w}_t^i}{\sum_{i=1}^N \tilde{w}_t^i}$.

# Naive importance sampling approximation

Initialisation: Draw $\{\boldsymbol{\theta}^i\}_{i=1}^N$ from $p(\boldsymbol{\theta})$

At $t \geq 1$ and for every $\boldsymbol{\theta}^i$, $i = 1, \ldots, N$:

SMC ($N$ samples) to approximate $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$

For $j = 1, \ldots, M$:

SMC ($M$ samples) to approximate $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}^i)$

- Draw $\bar{\boldsymbol{x}}_t^{i,j} \sim p(\boldsymbol{x}_t|\boldsymbol{\theta}^i, \boldsymbol{y}_{1:t-1})$
- Weights: $\tilde{u}_t^{i,j} \propto p(\boldsymbol{y}_t|\bar{\boldsymbol{x}}_t^{i,j}, \boldsymbol{\theta}^i)$
- Resampling: for $m = 1, \ldots, M$, $\tilde{\boldsymbol{x}}_t^{i,j} = \bar{\boldsymbol{x}}_t^{i,m}$ with prob. $u_t^{i,m} = \frac{\tilde{u}_t^{i,m}}{\sum_{j=1}^M \tilde{u}_t^{i,j}}$

- Likelihood of $\boldsymbol{\theta}^i$: $\tilde{w}_t^i = w_{t-1}^i\left(\frac{1}{M}\sum_{j=1}^M \tilde{u}_t^{i,j}\right)$
- Then, $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t}) = \sum_{i=1}^N w_t^i \delta_{\boldsymbol{\theta}^i}(d\boldsymbol{\theta})$, with $w_t^i = \frac{\tilde{w}_t^i}{\sum_{i=1}^N \tilde{w}_t^i}$.

# Naive importance sampling approximation

Initialisation: Draw $\{\boldsymbol{\theta}^i\}_{i=1}^N$ from $p(\boldsymbol{\theta})$

At $t \geq 1$ and for every $\boldsymbol{\theta}^i$, $i = 1, \ldots, N$:

SMC ($N$ samples) to approximate $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$

For $j = 1, \ldots, M$:

SMC ($M$ samples) to approximate $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}^i)$

- Draw $\bar{\boldsymbol{x}}_t^{i,j} \sim p(\boldsymbol{x}_t|\boldsymbol{\theta}^i, \boldsymbol{y}_{1:t-1})$
- Weights: $\tilde{u}_t^{i,j} \propto p(\boldsymbol{y}_t|\bar{\boldsymbol{x}}_t^{i,j}, \boldsymbol{\theta}^i)$
- Resampling: for $m = 1, \ldots, M$, $\tilde{\boldsymbol{x}}_t^{i,j} = \bar{\boldsymbol{x}}_t^{i,m}$ with prob. $u_t^{i,m} = \frac{\tilde{u}_t^{i,m}}{\sum_{j=1}^M \tilde{u}_t^{i,j}}$

- Likelihood of $\boldsymbol{\theta}^i$: $\tilde{w}_t^i = w_{t-1}^i \left( \frac{1}{M} \sum_{j=1}^M \tilde{u}_t^{i,j} \right)$
- Then, $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t}) = \sum_{i=1}^N w_t^i \delta_{\boldsymbol{\theta}^i}(d\boldsymbol{\theta})$, with $w_t^i = \frac{\tilde{w}_t^i}{\sum_{i=1}^N \tilde{w}_t^i}$.

# Naive importance sampling approximation

- Careful with $p(\boldsymbol{\theta})$: after several time steps the filter degenerates
- Possible solution: drawing $\{\boldsymbol{\theta}_t^i\} \sim p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})$ at each time step $\longrightarrow$ re-running from scratch the filter for $\boldsymbol{x}$ (i.e., not recursive anymore)

- NPF $\longrightarrow$ jittering: $\bar{\boldsymbol{\theta}}_t^i \sim \kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}')$, where

$$\kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}') = (1-\epsilon_N)\delta_{\boldsymbol{\theta}'}(\boldsymbol{\theta}) + \epsilon_N \kappa(d\boldsymbol{\theta}|\boldsymbol{\theta}')$$

  - $0 < \epsilon_N \leq \frac{1}{\sqrt{N}}$
  - $\kappa(d\boldsymbol{\theta}|\boldsymbol{\theta}')$ is an arbitrary Markov kernel with mean $\boldsymbol{\theta}'$ and finite variance, e.g., $\kappa(d\boldsymbol{\theta}|\boldsymbol{\theta}') = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\theta}', \tilde{\sigma}^2 \boldsymbol{I})$, with $\tilde{\sigma}^2 < \infty$.
- Guarantees convergence to the true posterior when $N \longrightarrow \infty$

# Naive importance sampling approximation

- Careful with $p(\boldsymbol{\theta})$: after several time steps the filter degenerates
- Possible solution: drawing $\{\boldsymbol{\theta}_t^i\} \sim p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})$ at each time step $\longrightarrow$ re-running from scratch the filter for $\boldsymbol{x}$ (i.e., not recursive anymore)

- NPF $\longrightarrow$ jittering: $\bar{\boldsymbol{\theta}}_t^i \sim \kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}')$, where

$$\kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}') = (1-\epsilon_N)\delta_{\boldsymbol{\theta}'}(\boldsymbol{\theta}) + \epsilon_N \kappa(d\boldsymbol{\theta}|\boldsymbol{\theta}')$$

  - $0 < \epsilon_N \leq \frac{1}{\sqrt{N}}$
  - $\kappa(d\boldsymbol{\theta}|\boldsymbol{\theta}')$ is an arbitrary Markov kernel with mean $\boldsymbol{\theta}'$ and finite variance, e.g., $\kappa(d\boldsymbol{\theta}|\boldsymbol{\theta}') = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\theta}', \tilde{\sigma}^2 \boldsymbol{I})$, with $\tilde{\sigma}^2 < \infty$.
- Guarantees convergence to the true posterior when $N \longrightarrow \infty$

# Nested particle filter (NPF)[4]

SMC ($N$ samples) to approximate $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$

For $i = 1, \ldots, N$:

- Jittering: Draw $\bar{\boldsymbol{\theta}}_t^i \sim \kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}_{t-1}^i)$

  SMC ($M$ samples) to approximate $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \bar{\boldsymbol{\theta}}_t^i)$

  Given $\bar{\boldsymbol{\theta}}_t^i$, for $j = 1, \ldots, M$:

  - Draw $\bar{\boldsymbol{x}}_t^{i,j} \sim p(\boldsymbol{x}_t|\bar{\boldsymbol{\theta}}_t^i, \boldsymbol{y}_{1:t-1})$
  - Weights: $\tilde{u}_t^{i,j} \propto p(\boldsymbol{y}_t|\bar{\boldsymbol{x}}_t^{i,j}, \bar{\boldsymbol{\theta}}_t^i)$
  - Resampling: for $m = 1, \ldots, M$, $\tilde{\boldsymbol{x}}_t^{i,j} = \bar{\boldsymbol{x}}_t^{i,m}$ with probability $u_t^{i,m} = \frac{\tilde{u}_t^{i,m}}{\sum_{j=1}^M \tilde{u}_t^{i,j}}$

- Likelihood of $\bar{\boldsymbol{\theta}}_t^i$: $\tilde{w}_t^i = \frac{1}{M}\sum_{j=1}^M \tilde{u}_t^{i,j}$
- Resampling: for $l = 1, \ldots, N$, $\{\boldsymbol{\theta}_t^i, \{\boldsymbol{x}_t^{i,j}\}_{1\le j\le M}\} = \{\bar{\boldsymbol{\theta}}_t^l, \{\tilde{\boldsymbol{x}}_t^{l,j}\}_{1\le j\le M}\}$ with prob. $w_t^l$, so that $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t}) = \frac{1}{N}\sum_{i=1}^N \delta_{\boldsymbol{\theta}_t^i}(d\boldsymbol{\theta})$

[4] Crisan and Miguez 2017.

# Family of nested filters

1. **Nested particle filters (NPFs)**[5].
   - Both layers $\longrightarrow$ Sequential Monte Carlo (SMC) methods
     High computational complexity: $N \times M$
2. **Nested hybrid filters (NHFs)**[6].
   - $\theta$-layer $\longrightarrow$ Monte Carlo-based methods (e.g., SMC or SQMC)
   - $x$-layer $\longrightarrow$ Gaussian techniques (e.g., EKFs or EnKFs)
3. **Nested Gaussian filters (NGFs)**[7].
   - $\theta$-layer $\longrightarrow$ Deterministic sampling methods (e.g., UKF).
   - $x$-layer $\longrightarrow$ Gaussian techniques (e.g., EKFs or EnKFs).

[5] Crisan and Míguez 2018.
[6] Pérez-Vieites, Mariño, and Míguez 2017.
[7] Pérez-Vieites and Míguez 2021.

# Family of nested filters

1. **Nested particle filters (NPFs)**[5].
   - Both layers ⟶ Sequential Monte Carlo (SMC) methods
     High computational complexity: $N \times M$
2. **Nested hybrid filters (NHFs)**[6].
   - $\boldsymbol{\theta}$-layer ⟶ Monte Carlo-based methods (e.g., SMC or SQMC)
   - $\boldsymbol{x}$-layer ⟶ Gaussian techniques (e.g., EKFs or EnKFs)
3. **Nested Gaussian filters (NGFs)**[7].
   - $\boldsymbol{\theta}$-layer ⟶ Deterministic sampling methods (e.g., UKF).
   - $\boldsymbol{x}$-layer ⟶ Gaussian techniques (e.g., EKFs or EnKFs).

[5] Crisan and Míguez 2018.
[6] Pérez-Vieites, Mariño, and Míguez 2017.
[7] Pérez-Vieites and Míguez 2021.

# Family of nested filters

1. **Nested particle filters (NPFs)**[5].
   - Both layers ⟶ Sequential Monte Carlo (SMC) methods
     High computational complexity: $N \times M$
2. **Nested hybrid filters (NHFs)**[6].
   - $\boldsymbol{\theta}$-layer ⟶ Monte Carlo-based methods (e.g., SMC or SQMC)
   - $\boldsymbol{x}$-layer ⟶ Gaussian techniques (e.g., EKFs or EnKFs)
3. **Nested Gaussian filters (NGFs)**[7].
   - $\boldsymbol{\theta}$-layer ⟶ Deterministic sampling methods (e.g., UKF).
   - $\boldsymbol{x}$-layer ⟶ Gaussian techniques (e.g., EKFs or EnKFs).

[5] Crisan and Míguez 2018.
[6] Pérez-Vieites, Mariño, and Míguez 2017.
[7] Pérez-Vieites and Míguez 2021.

# Index

# Reducing number of particles online

Problem: Great amount of samples ($N \times M$) and **waste of computational effort** when they are not well chosen.

Possible approach: reducing automatically the number of samples, $N$, when the performance is no longer compromised.

We studied the case for a nested Gaussian filter that implements:

- **Quadrature or cubature rules in the $\theta$-layer**, i.e., we generate $N$ quadrature points such that

$$N = \alpha^{d_\theta}, \quad \text{for} \quad \alpha \in \mathbb{N}, \alpha > 1. \tag{3}$$

The hyperparameter $\alpha$ **will depend on** $t$, so the number of samples is now defined as $N_t = \alpha_t^{d_\theta}$.

# Reducing number of particles online

Problem: Great amount of samples ($N \times M$) and **waste of computational effort** when they are not well chosen.

Possible approach: reducing automatically the number of samples, $N$, when the performance is no longer compromised.

We studied the case for a nested Gaussian filter that implements:

- **Quadrature or cubature rules in the $\theta$-layer**, i.e., we generate $N$ quadrature points such that

$$N = \alpha^{d_\theta}, \quad \text{for} \quad \alpha \in \mathbb{N}, \alpha > 1. \tag{3}$$

The hyperparameter $\alpha$ **will depend on** $t$, so the number of samples is now defined as $N_t = \alpha_t^{d_\theta}$.

# Reducing number of particles online

Problem: Great amount of samples ($N \times M$) and **waste of computational effort** when they are not well chosen.

Possible approach: reducing automatically the number of samples, $N$, when the performance is no longer compromised.

We studied the case for a nested Gaussian filter that implements:

- **Quadrature or cubature rules in the $\theta$-layer**, i.e., we generate $N$ quadrature points such that

$$N = \alpha^{d_\theta}, \quad \text{for} \quad \alpha \in \mathbb{N}, \alpha > 1. \tag{3}$$

The hyperparameter $\alpha$ **will depend on** $t$, so the number of samples is now defined as $N_t = \alpha_t^{d_\theta}$.

# Reducing number of particles online

Problem: Great amount of samples ($N \times M$) and **waste of computational effort** when they are not well chosen.

Possible approach: reducing automatically the number of samples, $N$, when the performance is no longer compromised.

We studied the case for a nested Gaussian filter that implements:

- **Quadrature or cubature rules in the $\theta$-layer**, i.e., we generate $N$ quadrature points such that

$$N = \alpha^{d_\theta}, \quad \text{for} \quad \alpha \in \mathbb{N}, \alpha > 1. \tag{3}$$

The hyperparameter $\alpha$ **will depend on** $t$, so the number of samples is now defined as $N_t = \alpha_t^{d_\theta}$.

# Adaptive reduction rule

New statistic to decide when to reduce $N_t$:

$$\rho_t = \frac{1}{\sum_{i=1}^{N_t} (\bar{s}_t^i)^2} \qquad \text{with} \qquad \bar{s}_t^i = \frac{p(\boldsymbol{y}_t | \boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)}{\sum_{n=1}^{N_t} p(\boldsymbol{y}_t | \boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^n)}$$

The statistic takes

- its **minimum value in** $\rho_t = 1$, which occurs when only one $p(\boldsymbol{y}_t | \boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$ is different from zero; and
- its **maximum value in** $\rho_t = N_t$, when for all $i = 1, \ldots, N_t$, the evaluations $p(\boldsymbol{y}_t | \boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$ are equal.

The adaptive reduction rule. Given $\epsilon$ and $\alpha_{\min}$:

- If $\frac{\rho_t}{N_t} > 1 - \epsilon$ ($\rho_t$ is close to its maximum value),

  Set $\alpha_{t+1} = \max(\alpha_{\min}, \alpha_t - 1)$.

- Otherwise, $\alpha_{t+1} = \alpha_t$.

Set $N_{t+1} = \alpha_{t+1}^{d_\theta}$.

# Adaptive reduction rule

New statistic to decide when to reduce $N_t$:

$$\rho_t = \frac{1}{\sum_{i=1}^{N_t} (\bar{s}_t^i)^2} \qquad \text{with} \qquad \bar{s}_t^i = \frac{p(\boldsymbol{y}_t | \boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)}{\sum_{n=1}^{N_t} p(\boldsymbol{y}_t | \boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^n)}$$

The statistic takes

- its **minimum value in** $\rho_t = 1$, which occurs when only one $p(\boldsymbol{y}_t | \boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$ is different from zero; and
- its **maximum value in** $\rho_t = N_t$, when for all $i = 1, \ldots, N_t$, the evaluations $p(\boldsymbol{y}_t | \boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$ are equal.

The **adaptive reduction rule**. Given $\epsilon$ and $\alpha_{\min}$:

- If $\frac{\rho_t}{N_t} > 1 - \epsilon$ ($\rho_t$ is close to its maximum value),

  Set $\alpha_{t+1} = \max(\alpha_{\min}, \alpha_t - 1)$.

- Otherwise, $\alpha_{t+1} = \alpha_t$.

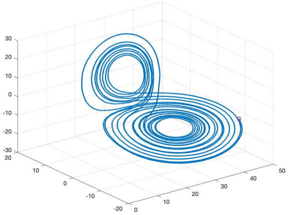
Set $N_{t+1} = \alpha_{t+1}^{d_\theta}$.

# Numerical results - Lorenz 63

We consider a stochastic Lorenz 63 model, whose dynamics are described by

- **state variables $\boldsymbol{x}_t$** of dimension $d_x = 3$,

$$dx_1 = [-S(x_1 - x_2)]d\tau + \sigma dv_1,$$
$$dx_2 = [Rx_1 - x_2 - x_1x_3]d\tau + \sigma dv_2,$$
$$dx_3 = [x_1x_2 - Bx_3]d\tau + \sigma dv_3,$$



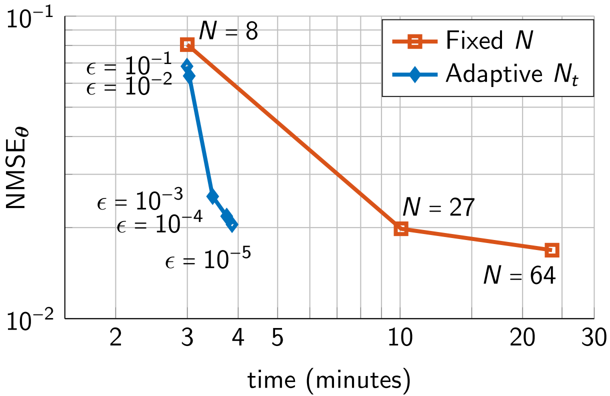

- **static parameters** $\boldsymbol{\theta} = [S, R, B]^\top$, and

- **linear observations** of the form

$$\boldsymbol{y}_t = k_o \begin{bmatrix} x_{1,t} \\ x_{3,t} \end{bmatrix} + \boldsymbol{r}_t,$$

where $k_o$ is a fixed known parameter and $\boldsymbol{r}_t \sim \mathcal{N}(\boldsymbol{r}_t | \mathbf{0}, \sigma_y^2 \boldsymbol{I}_2)$.

# Numerical results[8]




1. Nested Gaussian filter with fixed $N$. Different fixed $\alpha = \{2, 3, 4\}$, i.e., $N = \{8, 27, 64\}$.
2. Nested Gaussian filter with adaptive $N_t$. We set $\alpha_0 = 4$ and $\alpha_{\min} = 2$, i.e., $N_0 = 64$ and $N_{\min} = 8$.

[8] Pérez-Vieites and Elvira 2023.

# Conclusions

1. The nested methodology is online and flexible. It admits different types of filtering techniques in each layer, leading to a **set of algorithms**.
2. For a further reduction of the computational complexity. Automatic reduction of $N$ when points become less informative $\longrightarrow$ reduction of cost for a given performance.

# Conclusions

1. The nested methodology is online and flexible. It admits different types of filtering techniques in each layer, leading to a **set of algorithms**.
2. For a further reduction of the computational complexity. Automatic reduction of $N$ when points become less informative $\longrightarrow$ reduction of cost for a given performance.

# Thank you!


- **Pérez-Vieites, S.**, & Elvira, V. (2023). *Adaptive Gaussian nested filter for parameter estimation and state tracking in dynamical systems.* In 2023 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2023).
- **Pérez-Vieites** & Míguez (2021). *Nested Gaussian filters for recursive Bayesian inference and nonlinear tracking in state space models.* Signal Processing, 189, 108295.
- **Pérez-Vieites**, Mariño & Míguez (2018). *Probabilistic scheme for joint parameter estimation and state prediction in complex dynamical systems.* Physical Review E, 98(6), 063305.
- Crisan & Míguez (2018), *Nested particle filters for online parameter estimation in discrete-time state-space Markov models.* Bernoulli, vol. 24, no. 4A, pp. 3039–3086.


sarapv.github.io

Sara Pérez Vieites