Learning nonlinear systems using Gaussian processes, basis functions, and particles

Andreas Svensson

Department of Information Technology, Uppsala University, Sweden
Overview

Gaussian Process State Space model

Particle Gibbs (inference/estimation algorithm)

System Identification

Learning nonlinear systems using GPs, basis functions, and particles
Overview

Gaussian Process
State Space model
(model)

Particle Gibbs
(inference/estimation
algorithm)

System Identification
A versatile nonlinear dynamical model?

Learning nonlinear systems using GPs, basis functions, and particles
A versatile nonlinear dynamical model?

\[ x_{t+1} = f_{\theta}(x_t) + v_t \]

\[ y_t = g(x_t) + e_t \]
A versatile nonlinear dynamical model?

\[ x_{t+1} = f_\theta(x_t) + v_t \]
\[ y_t = g(x_t) + e_t \]

\[ f_\theta(x_t) = \sin(\alpha x_t) \]?
A versatile nonlinear dynamical model?

\[ x_{t+1} = f_\theta(x_t) + v_t \]
\[ y_t = g(x_t) + e_t \]

\[ f_\theta(x_t) = \sin(\alpha x_t)? \]
\[ f_\theta(x_t) = \gamma \sin(\alpha x_t + \beta)? \]
A versatile nonlinear dynamical model?

\[ x_{t+1} = f_\theta(x_t) + v_t \]
\[ y_t = g(x_t) + e_t \]

\[ f_\theta(x_t) = \sin(\alpha x_t)? \]
\[ f_\theta(x_t) = \gamma \sin(\alpha x_t + \beta)? \]
\[ f_\theta(x_t) = x_t \cos(\alpha x_t) + x_t^\beta \exp \left( \gamma - \frac{x_t^2 + \psi x_t^4}{8} \right) + \delta? \]
A versatile nonlinear dynamical model?

\[ x_{t+1} = f_\theta(x_t) + v_t \]
\[ y_t = g(x_t) + e_t \]

\[ f_\theta(x_t) = \sin(\alpha x_t)? \]
\[ f_\theta(x_t) = \gamma \sin(\alpha x_t + \beta)? \]
\[ f_\theta(x_t) = x_t \cos(\alpha x_t) + x_t^\beta \exp \left( \gamma - \frac{x_t^2 + \psi x_t^4}{8} \right) + \delta? \]

\[ f_\theta(x_t) = \sum_j w^{(j)} \phi^{(j)}(x_t)? \]
A versatile nonlinear dynamical model?

\[ x_{t+1} = f_\theta(x_t) + v_t \]
\[ y_t = g(x_t) + e_t \]

\[ f_\theta(x_t) = \sin(\alpha x_t) \]?
\[ f_\theta(x_t) = \gamma \sin(\alpha x_t + \beta) \]?
\[ f_\theta(x_t) = x_t \cos(\alpha x_t) + x_t^\beta \exp \left( \gamma - \frac{x_t^2 + \psi x_t^4}{8} \right) + \delta \]?

\[ f_\theta(x_t) = \sum_j w^{(j)} \phi^{(j)}(x_t) \]?
\[ f_\theta(x_t) = w^{(1)} x_t^1 + w^{(2)} x_t^2 + w^{(3)} x_t^3 \]?
A versatile nonlinear dynamical model?

\[ x_{t+1} = f_\theta(x_t) + \nu_t \]
\[ y_t = g(x_t) + \epsilon_t \]

\[ f_\theta(x_t) = \sin(\alpha x_t) ? \]
\[ f_\theta(x_t) = \gamma \sin(\alpha x_t + \beta) ? \]
\[ f_\theta(x_t) = x_t \cos(\alpha x_t) + x_t^\beta \exp \left( \gamma - \frac{x_t^2 + \psi x_t^4}{8} \right) + \delta ? \]

\[ f_\theta(x_t) = \sum_j w^{(j)} \phi^{(j)}(x_t) ? \]
\[ f_\theta(x_t) = w^{(1)} x_t^1 + w^{(2)} x_t^2 + w^{(3)} x_t^3 ? \]
\[ f_\theta(x_t) = w^{(1)} x_t^1 + w^{(2)} x_t^2 + w^{(3)} x_t^3 + w^{(4)} x_t^4 + w^{(5)} x_t^5 + w^{(6)} x_t^6 ? \]
Gaussian Processes (GP)

\[ f(x) \sim \mathcal{GP}(0, \kappa(x, x')) \]
Gaussian Processes (GP)

\[ f(x) \sim \mathcal{GP}(0, \kappa(x, x')) \]

\[
\begin{bmatrix}
  f(x_1) \\
  \vdots \\
  f(x_n)
\end{bmatrix}
\sim \mathcal{N}
\begin{pmatrix}
  0, \\
  \begin{bmatrix}
    \kappa(x_1, x_1) & \cdots & \kappa(x_1, x_n) \\
    \vdots & \ddots & \vdots \\
    \kappa(x_n, x_1) & \cdots & \kappa(x_n, x_n)
  \end{bmatrix}
\end{pmatrix}
\begin{pmatrix}
  f_* \\
  K(X_*, X_*)
\end{pmatrix}
\]
Gaussian Processes (GP)

\[ f(x) \sim \mathcal{GP}(0, \kappa(x, x')) \]

\[
\begin{bmatrix}
  f(x_1) \\
  \vdots \\
  f(x_n)
\end{bmatrix}
\sim \mathcal{N}
\begin{pmatrix}
  0, \\
  \kappa(x_1, x_1) & \cdots & \kappa(x_1, x_n) \\
  \vdots & \ddots & \vdots \\
  \kappa(x_n, x_1) & \cdots & \kappa(x_n, x_n)
\end{pmatrix}
\]

\[ K(X_*, X_*) \]

\[ f_* \]

\[ \text{Variance} \]

\[ \text{Mean} \]

\[ \text{Samples} \]

Learning nonlinear systems using GPs, basis functions, and particles
Gaussian Processes (GP)

\[ f(x) \sim \mathcal{GP}(0, \kappa(x, x')) \]

\[
\begin{bmatrix}
    f \\
    f^* 
\end{bmatrix} \sim \mathcal{N}
\left(0, \begin{bmatrix}
    K(X, X) & K(X, X^*) \\
    K(X^*, X) & K(X^*, X^*) 
\end{bmatrix}
\right)
\]
Gaussian Processes (GP)

\[ f(x) \sim \mathcal{GP}(0, \kappa(x, x')) \]

\[
\begin{bmatrix} f \\ f^* \end{bmatrix} \sim \mathcal{N} \left( \begin{bmatrix} K(X, X) & K(X, X^*) \\ K(X^*, X) & K(X^*, X^*) \end{bmatrix} \right) \Rightarrow \\
\begin{bmatrix} f \\ f^* \end{bmatrix} | f \sim \mathcal{N} \left( K(X^*, X) K(X, X)^{-1} f, K(X^*, X^*) - K(X^*, X) K(X, X)^{-1} K(X, X^*) \right)
\]
Gaussian Processes (GP)

\[ f(x) \sim \mathcal{GP}(0, \kappa(x, x')) \]

\[
\begin{bmatrix} f \\ f^* \end{bmatrix} \sim \mathcal{N} \left( 0, \begin{bmatrix} K(X, X) & K(X, X^*) \\ K(X^*, X) & K(X^*, X^*) \end{bmatrix} \right) \Rightarrow
\]

\[ f^* | f \sim \mathcal{N} \left( K(X^*, X)K(X, X)^{-1}f, K(X^*, X^*) - K(X^*, X)K(X, X)^{-1}K(X, X^*) \right) \]
Gaussian Processes (GP)

\[ f(x) \sim \mathcal{GP}(0, \kappa(x, x')) \]

\[
\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N} \left( 0, \begin{bmatrix} K(X, X) & K(X, X*) \\ K(X*, X) & K(X*, X*) \end{bmatrix} \right) \Rightarrow
\]

\[ f_* \mid f \sim \mathcal{N} \left( K(X*, X)K(X, X)^{-1}f, K(X*, X*) - K(X*, X)K(X, X)^{-1}K(X, X*) \right) \]
Gaussian Processes (GP)

\[ f(x) \sim \mathcal{GP}(0, \kappa(x, x')) \]

\[
\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N} \left( 0, \begin{bmatrix} K(X, X) & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix} \right) \quad \Rightarrow
\]

\[
f_* \mid f \sim \mathcal{N} \left( K(X_*, X)K(X, X)^{-1}f, K(X_*, X_*) - K(X_*, X)K(X, X)^{-1}K(X, X_*) \right)
\]
Gaussian Process State Space Model

\[ x_{t+1} = f(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \]
\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R) \]
\[ f \sim \mathcal{GP}(0, \kappa(x, x')) \]
Gaussian Process State Space Model

\[ x_{t+1} = f(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \]
\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R) \]
\[ f \sim \mathcal{GP}(0, \kappa(x, x')) \]

- Nonparametric
- Probabilistic
- Versatile and flexible

Computationally feasible?

\[ x_{t+1} = f(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \]
\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R) \]
\[ f \sim \mathcal{GP}(0, \kappa(x, x')) \]
Computationally feasible?

\[ x_{t+1} = f(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \]
\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R) \]
\[ f \sim \mathcal{GP}(0, \kappa(x, x')) \]
Computationally feasible?

\[ x_{t+1} = f(x_t) + v_t, \]
\[ y_t = g(x_t) + e_t, \]
\[ f \sim \mathcal{GP}(0, \kappa(x, x')) \]
\[ v_t \sim \mathcal{N}(0, Q) \]
\[ e_t \sim \mathcal{N}(0, R) \]
Approximate GPs (I/II)
Approximate GPs (I/II)

- Inducing variables:
Approximate GPs (I/II)

- Inducing variables:
Approximate GPs (II/II)

\[ f(x) \sim \text{GP}(0, \kappa(x, x')) \]

\[ f(x) \approx \sum_{j=0}^{m} w(j) \phi(j)(x) \]

with prior
\[ w(j) \sim N(0, S(\lambda(j))) \]

For \( x \in [-L, L] \subseteq \mathbb{R} \):
\[ \phi(j)(x) = \frac{1}{\sqrt{L}} \sin \left( \frac{\pi j (x + L)}{2L} \right) \]

\[ m = 4 \]
\[ m = 16 \]
\[ m = 9 \]

Approximate GPs (II/II)

- Reduced-rank GP approximation:

\[
f(x) \sim \mathcal{GP}(0, \kappa(x, x'))
\]
Approximate GPs (II/II)

- Reduced-rank GP approximation:

\[
f(x) \sim \mathcal{GP}(0, \kappa(x, x')) \iff f(x) \approx \sum_{j=0}^{m} w^{(j)} \phi^{(j)}(x)
\]

with prior

\[
w^{(j)} \sim \mathcal{N}(0, \Sigma^{(j)})
\]

For \( x \in [-L, L] \subset \mathbb{R} \):

\[
\phi^{(j)}(x) = \frac{1}{\sqrt{L}} \sin \left( \frac{\pi j (x+L)}{2L} \right).
\]


Approximate GPs (II/II)

- Reduced-rank GP approximation:

\[ f(x) \sim \mathcal{GP}(0, \kappa(x, x')) \Leftrightarrow f(x) \approx \sum_{j=0}^{m} w^{(j)} \phi^{(j)}(x) \]

with prior

\[ w^{(j)} \sim \mathcal{N}(0, S(\lambda^{(j)})) \]

For \( x \in [-L, L] \subset \mathbb{R} \):

\[ \phi^{(j)}(x) = \frac{1}{\sqrt{L}} \sin\left(\frac{\pi j (x+L)}{2L}\right). \]

Computationally feasible GP-SSM

\[ x_{t+1} = f(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \]

\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R) \]

\[ f \sim \mathcal{GP}(0, \kappa(x, x')) \]
Computationally feasible GP-SSM

\[ x_{t+1} = \sum_{j=0}^{m} w^{(j)} \phi^{(j)}(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \]

\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R) \]

\[ w^{(j)} \sim \mathcal{N}(0, S(\lambda^{(j)})) \]
Overview

Gaussian Process State Space model
(model)

Particle Gibbs (inference/estimation algorithm)

System Identification

Learning nonlinear systems using GPs, basis functions, and particles
Overview

Gaussian Process State Space model

Particle Gibbs (inference/estimation algorithm)

System Identification

Learning nonlinear systems using GPs, basis functions, and particles
Gibbs

Of interest: \( p(\theta, x_1:T \mid y_1:T) \) under the model assumption

\[
x_{t+1} = f_\theta(x_t) + v_t \\
y_t = g(x_t) + e_t
\]
Gibbs

Of interest: \( p(\theta, x_{1:T} | y_{1:T}) \) under the model assumption

\[
x_{t+1} = f_{\theta}(x_t) + v_t
\]
\[
y_t = g(x_t) + e_t
\]

0: Initialize \( x_{1:T}[0], \theta[0] \)

for \( k = 1, \ldots, K \)

1: Sample \( x_{1:T}[k] \sim p(x_{1:T} | \theta[k-1], y_{1:T}) \)

2: Sample \( \theta[k] \sim p(\theta | x_{1:T}[k], y_{1:T}) \)

end
Gibbs

Of interest: \( p(\theta, x_{1:T} \mid y_{1:T}) \) under the model assumption

\[
x_{t+1} = f_\theta(x_t) + v_t \\
y_t = g(x_t) + e_t
\]

0: Initialize \( x_{1:T}[0], \theta[0] \)

for \( k = 1, \ldots, K \)

1: Sample \( x_{1:T}[k] \sim p(x_{1:T} \mid \theta[k-1], y_{1:T}) \)

2: Sample \( \theta[k] \sim p(\theta \mid x_{1:T}[k], y_{1:T}) \)

end

\( \rightarrow \) samples \( \{x_{1:T}[k], \theta[k]\}_{k=1}^{K} \) from \( p(\theta, x_{1:T} \mid y_{1:T}) \) as \( K \to \infty \)
Of interest: \( p(\theta, x_{1:T} \mid y_{1:T}) \) under the model assumption

\[
x_{t+1} = f_\theta(x_t) + v_t \\
y_t = g(x_t) + e_t
\]

0: Initialize \( x_{1:T}[0], \theta[0] \)

for \( k = 1, \ldots, K \)

1: Sample \( x_{1:T}[k] \sim p(x_{1:T} \mid \theta[k-1], y_{1:T}) \)

2: Sample \( \theta[k] \sim p(\theta \mid x_{1:T}[k], y_{1:T}) \leftarrow \text{pen & paper} \)

end

\( \rightarrow \) samples \( \{x_{1:T}[k], \theta[k]\}_{k=1}^{K} \) from \( p(\theta, x_{1:T} \mid y_{1:T}) \) as \( K \rightarrow \infty \)
Particle Gibbs

Of interest: \( p(\theta, x_1:T \mid y_1:T) \) under the model assumption

\[
x_{t+1} = f_{\theta}(x_t) + v_t \\
y_t = g(x_t) + e_t
\]

**0:** Initialize \( x_1:T[0], \theta[0] \)

**for** \( k = 1, \ldots, K \)

1: Sample \( x_1:T[k] \sim p(x_1:T \mid \theta[k-1], y_1:T) \leftarrow \text{particle filter} \)

2: Sample \( \theta[k] \sim p(\theta \mid x_1:T[k], y_1:T) \leftarrow \text{pen & paper} \)

**end**

\( \rightarrow \) samples \( \{ x_1:T[k], \theta[k] \}_{k=1}^K \) from \( p(\theta, x_1:T \mid y_1:T) \) as \( K \rightarrow \infty \)
1: Sample $x_{1:T[k]} \sim p(x_{1:T} | \theta[k-1], y_{1:T})$

2: Sample $\theta[k] \sim p(\theta | x_{1:T}[k], y_{1:T})$
1: Sample $x_{1:T}[k] \sim p(x_{1:T} | \theta[k-1], y_{1:T})$

2: Sample $\theta[k] \sim p(\theta | x_{1:T}[k], y_{1:T})$
1: Sample $x_{1:T}[k] \sim p(x_{1:T} | \theta[k-1], y_{1:T})$

2: Sample $\theta[k] \sim p(\theta | x_{1:T}[k], y_{1:T})$

A particle filter provides samples from $p(x_{1:T} | \theta, y_{1:T})$ when $N \to \infty$
1: Sample $x_{1:T}[k] \sim p(x_{1:T} | \theta[k-1], y_{1:T})$

2: Sample $\theta[k] \sim p(\theta | x_{1:T}[k], y_{1:T})$

- A particle filter provides samples from $p(x_{1:T} | \theta, y_{1:T})$ when $N \to \infty$

- A conditional particle filter $\Pi^N_\theta(x_{1:T}[k-1], \cdot) : X^T \leftrightarrow X^T$ provides samples from $p(x_{1:T} | \theta, y_{1:T})$ when $K \to \infty$ ($N$ fix)

1: Sample \( x_{1:T}[k] \sim p(x_{1:T} | \theta[k-1], y_{1:T}) \)

2: Sample \( \theta[k] \sim p(\theta | x_{1:T}[k], y_{1:T}) \)

- A particle filter provides samples from \( p(x_{1:T} | \theta, y_{1:T}) \) when \( N \to \infty \)

- A conditional particle filter \( \Pi^N_{\theta}(x_{1:T}[k-1], \cdot) : X^T \mapsto X^T \)
  provides samples from \( p(x_{1:T} | \theta, y_{1:T}) \) when \( K \to \infty \) (\( N \) fix)

- A conditional particle filter with ancestor sampling is faster mixing


1: Sample $x_{1:T}[k] \sim p(x_{1:T} | \theta[k-1], y_{1:T})$

2: Sample $\theta[k] \sim p(\theta | x_{1:T}[k], y_{1:T})$
1: Sample $x_{1:T}[k] \sim p(x_{1:T} | \theta[k - 1], y_{1:T})$

2: Sample $\theta[k] \sim p(\theta | x_{1:T}[k], y_{1:T})$
1: Sample $x_{1:T[k]} \sim p(x_{1:T} | \theta[k-1], y_{1:T})$

2: Sample $\theta[k] \sim p(\theta | x_{1:T[k]}, y_{1:T})$

\[
p(\theta | x_{1:T[k]}, y_{1:T}) \propto p(x_{1:T}, y_{1:T} | \theta)p(\theta)
\]
1: Sample $x_1:T[k] \sim p(x_1:T | \theta[k-1], y_1:T)$

2: Sample $\theta[k] \sim p(\theta | x_1:T[k], y_1:T)$

$$p(\theta | x_1:T[k], y_1:T) \propto p(x_1:T, y_1:T | \theta) p(\theta)$$

▶ Example 1:

$$\begin{align*}
x_{t+1} &= a x_t + v_t, \quad v_t \sim \mathcal{N}(0, 1) \\
y_t &= x_t + e_t, \quad e_t \sim \mathcal{N}(0, 1) \\
p(a) &= \mathcal{N}(a; 0, \sigma^2)
\end{align*}$$

⇒

Learning nonlinear systems using GPs, basis functions, and particles
1: Sample $x_{1:T[k]} \sim p(x_{1:T} | \theta[k-1], y_{1:T})$

2: Sample $\theta[k] \sim p(\theta | x_{1:T[k]}, y_{1:T})$

$$p(\theta | x_{1:T[k]}, y_{1:T}) \propto p(x_{1:T}, y_{1:T} | \theta)p(\theta)$$

**Example 1:**

\[
\begin{align*}
x_{t+1} &= ax_t + v_t, \quad v_t \sim \mathcal{N}(0, 1) \\
y_t &= x_t + e_t, \quad e_t \sim \mathcal{N}(0, 1)
\end{align*}
\]

\[
p(a) = \mathcal{N}(a; 0, \sigma^2)
\]

$$p(a | x_{1:T[k]}, y_{1:T}) = \mathcal{N}\left(a; \frac{\sum_t x_{t+1}x_t}{1/\sigma^2 + \sum_t x_t x_t}, \frac{1}{1/\sigma^2 + \sum_t x_t x_t}\right)$$
1: Sample $x_{1:T[k]} \sim p(x_{1:T} \mid \theta[k-1], y_{1:T})$

2: Sample $\theta[k] \sim p(\theta \mid x_{1:T[k]}, y_{1:T})$

$$p(\theta \mid x_{1:T[k]}, y_{1:T}) \propto p(x_{1:T}, y_{1:T} \mid \theta)p(\theta)$$

**Example 1:**

$$x_{t+1} = ax_t + v_t, \quad v_t \sim \mathcal{N}(0, 1)$$

$$y_t = x_t + e_t, \quad e_t \sim \mathcal{N}(0, 1)$$

$$\Rightarrow$$

$$p(a) = \mathcal{N}(a; 0, \sigma^2)$$

$$p(a \mid x_{1:T[k]}, y_{1:T}) = \mathcal{N}(a; \frac{\sum_t x_{t+1} x_t}{1/\sigma^2 + \sum_t x_t x_t}, \frac{1}{1/\sigma^2 + \sum_t x_t x_t})$$

**Example 2:**

$$x_{t+1} = A \phi(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q)$$

$$y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R)$$

$$\Rightarrow$$

$$p(A, Q) = \mathcal{MN}(A; 0, Q, V)\mathcal{IW}(Q; \ell, \Lambda)$$
1: Sample $x_{1:T}[k] \sim p(x_{1:T} | \theta[k-1], y_{1:T})$

2: Sample $\theta[k] \sim p(\theta | x_{1:T}[k], y_{1:T})$

\[ p(\theta | x_{1:T}[k], y_{1:T}) \propto p(x_{1:T}, y_{1:T} | \theta)p(\theta) \]

**Example 1:**
\[
\begin{align*}
x_{t+1} &= ax_t + v_t, \quad v_t \sim \mathcal{N}(0, 1) \\
y_t &= x_t + e_t, \quad e_t \sim \mathcal{N}(0, 1)
\end{align*}
\]
\[
p(a) = \mathcal{N}(a; 0, \sigma^2)
\]
\[
p(a | x_{1:T}[k], y_{1:T}) = \mathcal{N}
\left(a; \frac{\sum_t x_{t+1} x_t}{1/\sigma^2 + \sum_t x_t x_t}, \frac{1}{1/\sigma^2 + \sum_t x_t x_t}\right)
\]

**Example 2:**
\[
\begin{align*}
x_{t+1} &= A\phi(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \\
y_t &= g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R)
\end{align*}
\]
\[
p(A, Q) = \mathcal{MN}(A; 0, Q, V)\mathcal{IW}(Q; \ell, \Lambda)
\]
\[
p(A, Q | x_{1:T}[k], y_{1:T}) = \mathcal{MN}(A; \Psi(\Sigma + V^{-1})^{-1}, Q, (\Sigma + V^{-1})^{-1})\mathcal{IW}(Q; \ell + T, \Lambda + \Phi - \Psi(\Sigma + V^{-1})^{-1})
\]
Overview

- Gaussian Process State Space model
  (model)

- Particle Gibbs
  (inference/estimation algorithm)

System Identification

Learning nonlinear systems using GPs, basis functions, and particles
Overview

Gaussian Process State Space model

Particle Gibbs
(inference/estimation algorithm)

System Identification

Learning nonlinear systems using GPs, basis functions, and particles
Learning GP-SSM

Approximate GP-SSM:

\[
\begin{align*}
    x_{t+1} &= \sum_{j=0}^{\infty} w(j) \phi(j)(x_t) + v_t, \\
    y_t &= g(x_t) + e_t,
\end{align*}
\]

\[v_t \sim \mathcal{N}(0, Q)\], \[e_t \sim \mathcal{N}(0, R)\]

\[f \sim \text{GP}(0, \kappa(x, x'))\]

Particle Gibbs with closed-form parameter update:

\[
\begin{align*}
    x_{t+1} &= A \phi(x_t) + v_t, \\
    y_t &= g(x_t) + e_t,
\end{align*}
\]

\[v_t \sim \mathcal{N}(0, Q)\], \[e_t \sim \mathcal{N}(0, R)\]

\[p(A, Q) = MN(A; \Psi(\Sigma+V^{-1})^{-1}, Q, V) IW(Q; \ell, \Lambda)\]

\[p(A, Q | x_1:T[y_1:T]) = MN(A; \Psi(\Sigma+V^{-1})^{-1}, Q, (\Sigma+V^{-1})^{-1}) IW(Q; \ell+T, \Lambda+\Phi - \Psi(\Sigma+V^{-1})^{-1})\]

Learning GP-SSM

Approximate GP-SSM:

\[ x_{t+1} = \sum_{j=0}^{m} w^{(j)} \phi^{(j)}(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \]

\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R) \]

\[ w^{(j)} \sim \mathcal{N}(0, S(\lambda^{(j)})) \]
Learning GP-SSM

▶ Approximate GP-SSM:

\[ x_{t+1} = \sum_{j=0}^{m} w^{(j)} \phi^{(j)}(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \]

\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R) \]

\[ w^{(j)} \sim \mathcal{N}(0, S(\lambda^{(j)})) \]

▶ Particle Gibbs with closed-form parameter update:

\[
\begin{align*}
    x_{t+1} &= A \phi(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \\
    y_t &= g(x_t) + e_t, \quad v_t \sim \mathcal{N}(0, R) \\
\end{align*}
\]

\[
\begin{align*}
    p(A, Q | x_{1:T}[k], y_{1:T}) &= p(A, Q) = \mathcal{MN}(A; 0, Q, V)\mathcal{IW}(Q; \ell, \Lambda) \\
    \mathcal{MN}(A; \Psi(\Sigma + V^{-1})^{-1}, Q, (\Sigma + V^{-1})^{-1})\mathcal{IW}(Q; \ell + T, \Lambda + \Phi - \Psi(\Sigma + V^{-1})^{-1})
\end{align*}
\]
Learning GP-SSM

► Approximate GP-SSM:

\[ x_{t+1} = \sum_{j=0}^{m} w^{(j)} \phi^{(j)}(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \]
\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R) \]
\[ w^{(j)} \sim \mathcal{N}(0, S(\lambda^{(j)})) \]

► Particle Gibbs with closed-form parameter update:

\[ x_{t+1} = A\phi(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \]
\[ y_t = g(x_t) + e_t, \quad v_t \sim \mathcal{N}(0, R) \]
\[ p(A, Q) = \mathcal{MN}(A; 0, Q, V)\mathcal{IW}(Q; \ell, \Lambda) \]
\[ \mathcal{MN}(A; \Psi(\Sigma + V^{-1})^{-1}, Q, (\Sigma + V^{-1})^{-1})\mathcal{IW}(Q; \ell + T, \Lambda + \Phi - \Psi(\Sigma + V^{-1})^{-1}) \]

Only GP-SSMs?

Approximate GP-SSM:

\[
x_{t+1} = \sum_{j=0}^{m} w^{(j)} \phi^{(j)}(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q)
\]

\[
y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R)
\]

\[
w^{(j)} \sim \mathcal{N}(0, S(\lambda^{(j)}))
\]
Only GP-SSMs?

Approximate GP-SSM:

\[ x_{t+1} = \sum_{j=0}^{m} w(j) \psi(j)(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \]

\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R) \]

\[ w(j) \sim \mathcal{N}(0, \xi(j)) \]
Only GP-SSMs?

Approximate GP-SSM:

\[ x_{t+1} = \sum_{j=0}^{m} w(j) \psi(j)(x_t) + v_t, \quad v_t \sim \mathcal{N}(0, Q) \]

\[ y_t = g(x_t) + e_t, \quad e_t \sim \mathcal{N}(0, R) \]

\[ w(j) \sim \mathcal{N}(0, \xi(j)) \]

Toy example

\[ x_{t+1} = 10 \text{sinc} \left( \frac{x_t}{7} \right) + v_t \]  \quad v_t \sim \mathcal{N}(0, 4)  

\[ y_t = x_t + e_t \text{ (known)} \]  \quad e_t \sim \mathcal{N}(0, 4) 

\[ T = 40, \ m = 40 \]
Toy example

\[ x_{t+1} = 10\text{sinc}\left(\frac{x_t}{7}\right) + v_t \quad v_t \sim \mathcal{N}(0, 4) \]

\[ y_t = x_t + e_t \text{ (known)} \quad e_t \sim \mathcal{N}(0, 4) \]

\[ T = 40, \quad m = 40 \]
Toy example

\[ x_{t+1} = 10 \text{sinc} \left( \frac{x_t}{7} \right) + v_t \quad v_t \sim \mathcal{N}(0, 4) \]
\[ y_t = x_t + e_t \quad \text{(known)} \quad e_t \sim \mathcal{N}(0, 4) \]
\[ T = 40, \ m = 40 \]
Toy example

\[ x_{t+1} = 10 \text{sinc} \left( \frac{x_t}{7} \right) + v_t \quad v_t \sim \mathcal{N}(0, 4) \]

\[ y_t = x_t + e_t \text{ (known)} \quad e_t \sim \mathcal{N}(0, 4) \]

\[ T = 40, \quad m = 40 \]
\[ x_{t+1}^1 = \left( \frac{x_t^1}{1+(x_t^1)^2} + 1 \right) \sin(x_t^2) \]

\[ x_{t+1}^2 = x_t^2 \cos(x_t^2) + x_t^1 \exp\left( -\frac{(x_t^1)^2+(x_t^2)^2}{8} \right) \]

\[ + \frac{(u_t)^3}{1+(u_t)^2+0.5 \cos(x_t^1+x_t^2)} \]

\[ y_t = \frac{x_t^1}{1+0.5 \sin(x_t^2)} + \frac{x_t^2}{1+0.5 \sin(x_t^1)} + \epsilon_t \]
Narendra-Li Benchmark

\[ x_{t+1}^1 = \left( \frac{x_t^1}{1 + (x_t^1)^2} + 1 \right) \sin(x_t^2) \]
\[ x_{t+1}^2 = x_t^2 \cos(x_t^2) + x_t^1 \exp \left( -\frac{(x_t^1)^2 + (x_t^2)^2}{8} \right) \]
\[ + \frac{(u_t)^3}{1 + (u_t)^2 + 0.5 \cos(x_t^1 + x_t^2)} \]
\[ y_t = \frac{x_t^1}{1 + 0.5 \sin(x_t^2)} + \frac{x_t^2}{1 + 0.5 \sin(x_t^1)} + \epsilon_t \]

<table>
<thead>
<tr>
<th>Method</th>
<th>RMSE</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>This</td>
<td>0.06</td>
<td>2 000</td>
</tr>
<tr>
<td>Roll et al.</td>
<td>0.43</td>
<td>50 000</td>
</tr>
<tr>
<td>Stenman et al.</td>
<td>0.46</td>
<td>50 000</td>
</tr>
<tr>
<td>Xu et al. (AHH)</td>
<td>0.31</td>
<td>2 000</td>
</tr>
<tr>
<td>Xu et al. (MARS)</td>
<td>0.49</td>
<td>2 000</td>
</tr>
</tbody>
</table>


Learning nonlinear systems using GPs, basis functions, and particles
Water tank with overflow

Pump

Tank 1

Tank 2

Validation data
2nd order linear state space model. RMSE: 0.67
5th order NARX with sigmoidnet. RMSE: 0.73
5th order NARX with wavelets. RMSE: 0.61
"simulation focus. RMSE: 0.64
The proposed model. RMSE: 0.45
Credibility interval for the proposed method.

Learn nonlinear systems using GPs, basis functions, and particles
Water tank with overflow

Validation data
2nd order linear state space model. RMSE: 0.67
5th order NARX with sigmoidnet. RMSE: 0.73
" simulation focus. RMSE: 0.49
5th order NARX with wavelets. RMSE: 0.61
" simulation focus. RMSE: 0.64
The proposed model. RMSE: 0.45
Credibility interval for the proposed method.

Learning nonlinear systems using GPs, basis functions, and particles
That’s it!

Gaussian Process State Space model (model)

Particle Gibbs (inference/estimation algorithm)

System Identification

Learning nonlinear systems using GPs, basis functions, and particles