BiiPS

a software for Bayesian inference with interacting Particle Systems

A. Todeschini, F. Caron, P. Legrand, P. Del Moral
Summary

1. Context
2. Particle methods
3. BiiPS software
4. Conclusion
Summary

1. Context

2. Particle methods

3. BiiPS software

4. Conclusion
Bayesian inference

Bayesian inference involves the approximation of a probability distribution of an unknown parameter \( x \in E \) given the observations \( y \):

\[
\pi = p(x|y)
\]

Bayes rule:

\[
p(x|y) = \frac{p(x, y)}{p(y)} = \frac{p(x)p(y|x)}{p(y)} = \frac{\gamma(x)}{Z}
\]

Marginal likelihood:

\[
Z = p(y) = \int_E p(y|x)p(x)dx
\]
Bayesian inference

Delivers both an estimate and the associated uncertainty

\[ \hat{x}_{\text{MMSE}} = \mathbb{E}[x|y] = \int_E x \, p(x|y) \, dx \]

\[ \text{Var}(\hat{x}_{\text{MMSE}}) = \mathbb{E}[(x - \hat{x})^2|Y] = \int_E (x - \hat{x})^2 \, p(x|y) \, dx \]

More generally, we can integrate any test function \( \varphi(x) \)

\[ I = \mathbb{E}[\varphi(x)|y] = \int_E \varphi(x) \, p(x|y) \, dx \]
Bayesian inference

Delivers both an estimate and the associated uncertainty

$$\hat{x}_{\text{MMSE}} = \mathbb{E}[x|y]$$

$$= \int_E x \ p(x|y) \ dx$$

$$\text{Var}(\hat{x}_{\text{MMSE}}) = \mathbb{E}[(x - \hat{x})^2 | Y]$$

$$= \int_E (x - \hat{x})^2 \ p(x|y) \ dx$$

More generally, we can integrate any test function \( \varphi(x) \)

$$I = \mathbb{E}[\varphi(x)|y] = \int_E \varphi(x) \ p(x|y) \ dx$$
The graph displays a factorization of the joint distribution:

$$p(x_{1:3}, y_{1:2})$$

**Figure:** Directed acyclic graph
The graph displays a factorization of the joint distribution:

\[ p(x_{1:3}, y_{1:2}) = p(x_1) \frac{p(x_2 | x_1) p(x_3 | x_2, x_1)}{p(y_1 | x_2, x_3) p(y_2 | x_3)} \]

**Figure:** Directed acyclic graph
The graph displays a factorization of the joint distribution:

\[ p(x_{1:3}, y_{1:2}) = p(x_1) \ p(x_2|x_1) \ p(x_3|x_2, x_1) \ p(y_1|x_2, x_3) \ p(y_2|x_3) \]
The graph displays a factorization of the joint distribution:

\[
p(x_{1:3}, y_{1:2}) = p(x_1) \ p(x_2|x_1) \ p(x_3|x_2, x_1) \\
p(y_1|x_2, x_3) \ p(y_2|x_3)
\]

**Figure:** Directed acyclic graph
Graphical Models / Bayesian networks

The graph displays a factorization of the joint distribution:

\[
p(x_{1:3}, y_{1:2}) = p(x_1) \cdot p(x_2|x_1) \cdot p(x_3|x_2, x_1) \cdot p(y_1|x_2, x_3) \cdot p(y_2|x_3)
\]

Figure: Directed acyclic graph
The graph displays a **factorization** of the joint distribution:

\[
p(x_{1:3}, y_{1:2}) = p(x_1) \ p(x_2 | x_1) \ p(x_3 | x_2, x_1) \\
p(y_1 | x_2, x_3) \ p(y_2 | x_3)
\]

**Figure**: Directed acyclic graph
Hidden Markov Models / state-space models

\[ p(x_0) \quad \text{Initial distribution} \]

\[ \forall t = 1, \ldots, T \]
\[ \begin{align*}
    p(x_{t+1} | x_t) \quad & \text{Evolution model} \\
    p(y_t | x_t) \quad & \text{Measurement model}
\end{align*} \]

where \( x_t \in X \), e.g. \( \mathbb{R}^n \)
Bayesian inference on HMM

\[ X = X_{0:t} \in E = \mathcal{X}^{t+1} \]
\[ Y = Y_{1:t} \]

(1) becomes

\[ p(x_{0:t} | y_{1:t}) = \frac{p(x_{0:t}) p(y_{1:t} | x_{0:t})}{p(y_{1:t})} \]
\[ = p(x_{0:t-1} | y_{1:t-1}) \frac{p(x_t | x_{t-1}) p(y_t | x_t)}{p(y_t | y_{1:t-1})} \]
Bayesian inference

- Many problems (inverse problems, filtering, tracking, deconvolution, etc..) can be formulated in this context
- The posterior distribution is usually not calculable analytically
  - complex non linear models
  - high dimension
- ... hence requiring the use of stochastic simulation techniques
Bayesian inference

- Many problems (inverse problems, filtering, tracking, deconvolution, etc..) can be formulated in this context
- The posterior distribution is usually not calculable analytically
  - complex non linear models
  - high dimension
- ... hence requiring the use of stochastic simulation techniques
BUGS
Bayesian inference Using Gibbs Sampling

- More than 20 years history
  - ‘Classic’ BUGS: started in 1989 at MRC Biostatistics Unit
  - WinBUGS: co-developed with Imperial College School of Medicine
  - OpenBUGS: open source release and experimental development
  - JAGS © Martyn Plummer: open source clone in C++
- Expert system runs MCMC methods (Gibbs, Metropolis, ...) in a ‘black-box’ fashion
  - Iterative algorithms: approximately sampling according to target posterior distribution
- User-friendly
- Very popular among practitioners, applying MCMC methods to a wide range of applications
Summary

1 Context

2 Particle methods

3 BiiPS software

4 Conclusion
Particle methods

- A new generation of stochastic algorithms has emerged in recent years (particle filtering, sequential Monte Carlo methods, etc.).
- Based on interacting particle systems
- Two stochastic mechanisms
  1. **Mutation**: the particles explore their environment randomly and independently
  2. **Selection**: the best suited particles are duplicated, others removed

- Designed to sample from a sequence of distributions \( \pi_k(x_{1:k}) \) e.g.
  \[ \pi_k(x_{1:k}) = p(x_{1:k} | y_{1:k}) \]
- When we can only compute the unnormalized version \( \gamma_k(x_{1:k}) \)

\[
\pi_k(x_{1:k}) = \frac{\gamma_k(x_{1:k})}{Z_k} = \frac{p(x_{1:k}, y_{1:k})}{p(y_{1:k})}
\]
Particle methods

- A new generation of stochastic algorithms has emerged in recent years (particle filtering, sequential Monte Carlo methods, etc.).
- Based on interacting particle systems
- Two stochastic mechanisms
  1. **Mutation**: the particles explore their environment randomly and independently
  2. **Selection**: the best suited particles are duplicated, others removed

- Designed to sample from a sequence of distributions $\pi_k(x_{1:k})$ e.g.
  
  $\pi_k(x_{1:k}) = p(x_{1:k} | y_{1:k})$

- when we can only compute the unnormalized version $\gamma_k(x_{1:k})$

  \[
  \pi_k(x_{1:k}) = \frac{\gamma_k(x_{1:k})}{Z_k} = \frac{p(x_{1:k}, y_{1:k})}{p(y_{1:k})}
  \]
Example

1D linear gaussian state-space model

\[ X_0 \sim \mathcal{N}(0, 1) \]

Pour \( t = 1, \ldots, 20 \)

\[ X_t | X_{t-1} \sim \mathcal{N}(X_{t-1}, 1) \]
\[ Y_t | X_t \sim \mathcal{N}(X_t, 2) \]

- Filtering problem: estimate \( p(x_t|y_{1:t}) \)
- Optimal solution tractable by Kalman filter
Example

$t=0, N=50$

Particles generation

Particle weights

$\text{ESS}_0 = 50.00$
Example

$t=1, N=50$

Particles mutation

Particle weights

ESS$_1$ = 22.27
Example

$t=1, N=50$
Particles resampling

Particle weights
ESS$_1 = 22.27$
Example

t=2, N=50
Particles resampling

Particle weights
ESS$_2$=28.78
Example

$t=3, N=50$
Particles mutation

 Particle weights
ESS$_3=25.50$
Example

$t=3, N=50$

Particles resampling

Particle weights

$\text{ESS}_3 = 25.50$
Example

\textbf{t=4, N=50}

Particles mutation

Particle weights

ESS$_4$ = 3.39
Example

$t=4$, $N=50$

Particles resampling

Particle weights

$\text{ESS}_4 = 3.39$
Example

\[ t=5, N=50 \]

Particles mutation

Particle weights

ESS\(_5\) = 30.92
Example

\[ t=5, N=50 \]

Particles resampling

Particle weights

\[ \text{ESS}_5 = 30.92 \]
Example
Example

$t=6$, $N=50$

Particles resampling

Particle weights

$\text{ESS}_6=33.01$
Example

$t=7, N=50$

Particles mutation

Particle weights

ESS\(_7\) = 31.77
Example

\[ t=7, N=50 \]

Particles resampling

Particle weights

\[ \text{ESS}_7 = 31.77 \]
Example

**Graph: t=8, N=50**

- **Particles mutation**
- **Particle weights**
- **ESS**

$E_{SS} = 33.86$

A. Todeschini et al. - BiIPS
Rencontres R Bordeaux
July 2012- 14/30
Example

$t=8, N=50$

Particles resampling

Particle weights

$ESS_8 = 33.86$
Example

$t=9$, $N=50$

Particles mutation

Particle weights

$\text{ESS}_9 = 35.66$
Example

Particles resampling

ESS$_g$ = 35.66

$t=9, N=50$

A. Todeschini et al. - BiPS
Rencontres R Bordeaux
July 2012 - 14/30
Example

\[ t=10, N=50 \]

Particles mutation

Particle weights

ESS \(_{10}=18.93\)
### SMC algorithm with N particles

- **At time 1:** for $i = 1, \ldots, N$
  - Sample $x_1^{(i)} \sim q_1(x_1)$
  - Compute unnormalized weights $w_1^{(i)} = \frac{\gamma(x_1^{(i)})}{q_1(x_1^{(i)})}$

- **At time $k = 2, \ldots, n$:** for $i = 1, \ldots, N$
  - Resample $\{x_{k-1}^{(i)}, W_{k-1}^{(i)}\}$ and set $W_{k-1}^{(i)} = 1/N$
  - Sample $x_k^{(i)} \sim q_k(x_k|x_1^{1:k-1})$
  - Compute unnormalized weights $w_k^{(i)} = W_{k-1}^{(i)} \frac{\gamma_k(x_1^{1:k})}{\gamma_{k-1}(x_1^{1:k-1}) q_k(x_k^{(i)})}$

**Incremental weight**
Particle methods

Estimates

- At time $k$ we can approximate integrals $I_k = \mathbb{E}_{\pi_k}[\varphi(x_k)]$ by

$$\hat{I}_k = \sum_{i=1}^{N} W_k^{(i)} \varphi(x^{(i)}_n)$$

- We also obtain sequential approximations of the normalizing constant

$$Z_k = Z_{k-1} \frac{1}{N} \sum_{i=1}^{N} \alpha_k(x_{1:k}^{(i)})$$

- Effective Sample Size criterions give quality indicators between 1 and $N$

$$ESS_k \approx \left( \sum_{i=1}^{N} (W_k^{(i)})^2 \right)^{-1}$$
Particle methods

- They are more appropriate than MCMC in several situations (highly correlated variables, multimodality)
- Do not require convergence time to equilibrium, suitable for dynamic estimation problems
- But: no "black box" software allowing the use of these techniques by non-experts
Summary

1. Context

2. Particle methods

3. BiiPS software

4. Conclusion
BiiPS software

Objectives

Develop a "black box" software to make Bayesian inference using interacting Particle Systems.

Figure: Input/Output diagram
**BiiPS software**

**Features**

- Core libraries in C++ (> 25K lines of code) making use of Boost
  - Creates a graphical model and executes a particle algorithm (filtering and smoothing)
  - Selects automatically the order of the variables to be sampled
  - Selects automatically the laws of exploration (conjugate and non-conjugate cases)
  - Module with its extensible set of functions, distributions and samplers
- **BUGS** language compatible: compiler adapted from JAGS © M. Plummer
- **RBiips** interface to R making use of Rcpp package
Example
Financial econometry

Consider inferring the underlying volatility $X_{1:t}$ from observed price or rate data $Y_{1:t}$

$$X_1 \sim \mathcal{N}(0, \frac{\sigma_1^2}{1-\alpha^2})$$

$$X_t | X_{t-1} \sim \mathcal{N}(\alpha x_{t-1}, \frac{\sigma_1^2}{1-\alpha^2}) \quad t > 1$$

$$y_t | X_t \sim \mathcal{N}(0, \beta^2 \exp(x_t)) \quad t > 1$$

BUGS language "volatility.bug"

```
model {
    prec.x <- (1-alpha^2) / sigma^2
    x[1] ~ dnorm(0, prec.x)
    for (t in 2:t.max) {
        x[t] ~ dnorm(alpha * x[t-1], prec.x)
        prec.y[t] <- 1 / (beta^2 * exp(x[t]))
        y[t] ~ dnorm(0, prec.y[t])
    }
}
```
Stochastic volatility simulation
Example

Financial econometry

# Define data
data <- list(t.max=100, sigma=1.0,
               alpha=0.91, beta=0.5,
               y=y)
Example
Financial econometry

# Define data
data <- list(t.max=100, sigma=1.0,
            alpha=0.91, beta=0.5,
            y=y)

# Compile the model and load the data
model <- biips.model("volatility.bug",
data)
Example
Financial econometry

```r
# Define data
data <- list(t.max=100, sigma=1.0,
             alpha=0.91, beta=0.5,
             y=y)

# Compile the model and load the data
model <- biips.model("volatility.bug",
data)

# Run SMC algorithm
out.smc <- smc.samples(model, "x",
                        n.part=1000)
```
Example
Financial econometry

```r
# Summary statistics
x.summ <- summary(out.smc$x,
    fun=c("mean","quantiles"),
    probs=c(.05,.95))
plot(x.summ)
```

Figure: Summary statistics
Example

Financial econometry

# Kernel density estimates
plot(density(out.smc$x, adjust=2))

Figure: Kernel density estimates
Perspectives

Optimization

- Parallelization
- Reduce memory footprint

Software extensions

- More conjugate samplers, distributions and functions
- More advanced particle techniques
- Allow external user defined functions and samplers
- Interfaces: Matlab, Python, standalone
Summary

1. Context
2. Particle methods
3. BiiPS software
4. Conclusion
Conclusion

- **BiiPS** is a general software for Bayesian inference on **graphical models**
- Implements SMC/particle methods in a **black box** fashion
- Easy to use **RBiips** package
References


THANK YOU

http://alea.bordeaux.inria.fr/biips

Centre de Bordeaux
200 avenue de la Vieille Tour
33405 Talence Cedex, France
www.inria.com