



# Population Based Particle Filter

Sergio Hernandez

School of Mathematics, Statistics and Computer Science.

2/5/08

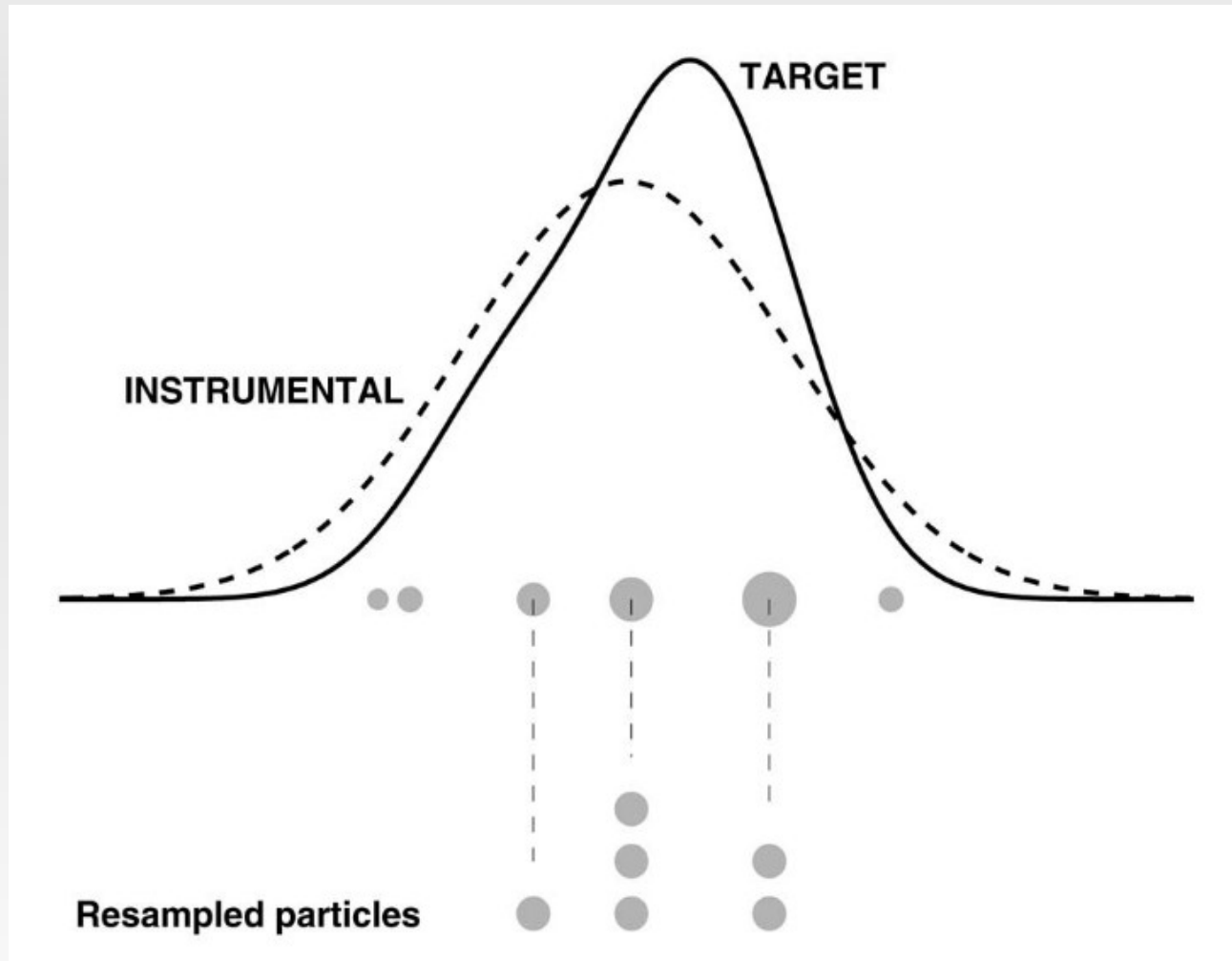
ubuntu

# Particle Filter



- Proposed by Gordon (1993) as a method for approximating the filtering distribution of a dynamic Bayesian network (i.e. hidden Markov model).
- A set samples ("particles") are propagated in time using sampling and resampling methods.
- **Intuitively** is easy to understand the method in terms of randomised search algorithms, but convergence is provided in terms of the law of large numbers and the central limit theorem.

# Particle Filter



"An Overview of Existing Methods and Recent Advances in Sequential Monte Carlo" Cappe et al., 2007.

# Population-based Monte Carlo for static inference



- Static inference is usually performed in parameter estimation problems.
- Idea : Generate a collection of random variables in parallel:
  - MCMC methodology : Parallel tempering (Geyer, 1991), Simulated tempering (Marinari et al., 1992), Adaptive direction sampling (Gilks et al., 1994), Evolutionary Monte Carlo (Liang et al., 2000)
  - Importance sampling : Annealed importance sampling (Neal, 2001), Population Monte Carlo (Iba, 2000), Population Monte Carlo (Cappe et al., 2003).

# Evolutionary Monte Carlo



- Mutation :

$$K(x_{1:N}, dx'_{1:N}) = \prod_{n=1}^N K_n(x_n, dx'_n),$$

- Crossover:

$$x'_n = (x_{1n}, \dots, x_{(l-1)n}, x_{lq}, \dots, x_{dq}),$$

$$x'_q = (x_{1q}, \dots, x_{(l-1)q}, x_{ln}, \dots, x_{dn}).$$

- Exchange:

$$A = \frac{\pi_n(x_q)\pi_q(x_n)}{\pi_n(x_n)\pi_q(x_q)},$$

- Snooker moves: Adapt the future movement of one chain along the direction generated by other chain.

# Population-based Monte Carlo for dynamic models



- Dynamic inference is performed for state estimation in dynamical systems (i.e. tracking), but it also can be used in on-line joint parameter and state estimation.
- Idea : Perform sampling/resampling and add an additional step for particle rejuvenation:
  - Resample-Move (Gilks et.al, 2001)
  - Annealed particle filter (Godsill, 2001), Sequential Monte Carlo samplers (Del Moral, 2006), Population particle filter (Bhaskar et al, 2008).
  - Genetic particle filter (Higuchi, 1997), Metaheuristic particle filter (Pantrigo et al., 2005).

# Sequential Monte Carlo Samplers (Del Moral et.al, 2006)



- Define a sequence of target measures  $\{\pi_n\}$  that contains the target density  $\pi$ .
- Algorithm
  - 1: Build a sequence of distributions  $\{\pi_n\}$ ,  $n=1, \dots, p$ , such that  $\pi_1$  is easy to sample from and  $\pi$  is one of its marginals.
  - 2: Build a sequence of MCMC transition kernels  $\{K_n\}$  such that  $K_n$  is  $\pi_n$  invariant or  $K_n$  is an approximate Gibbs move of invariant distribution  $\pi_n$ .
  - 3: On the basis of  $\{\pi_n\}$  and  $\{K_n\}$ , build a sequence of artificial backward Markov kernels  $\{L_n\}$  approximating the optimal backward kernel (minimizing the variance of the importance weights).
  - 4: Use the SMC algorithm to approximate  $\{\pi_n\}$  and to estimate  $\{Z_n\}$ .

# Population-based Particle Filter (Bhaskar 2008)



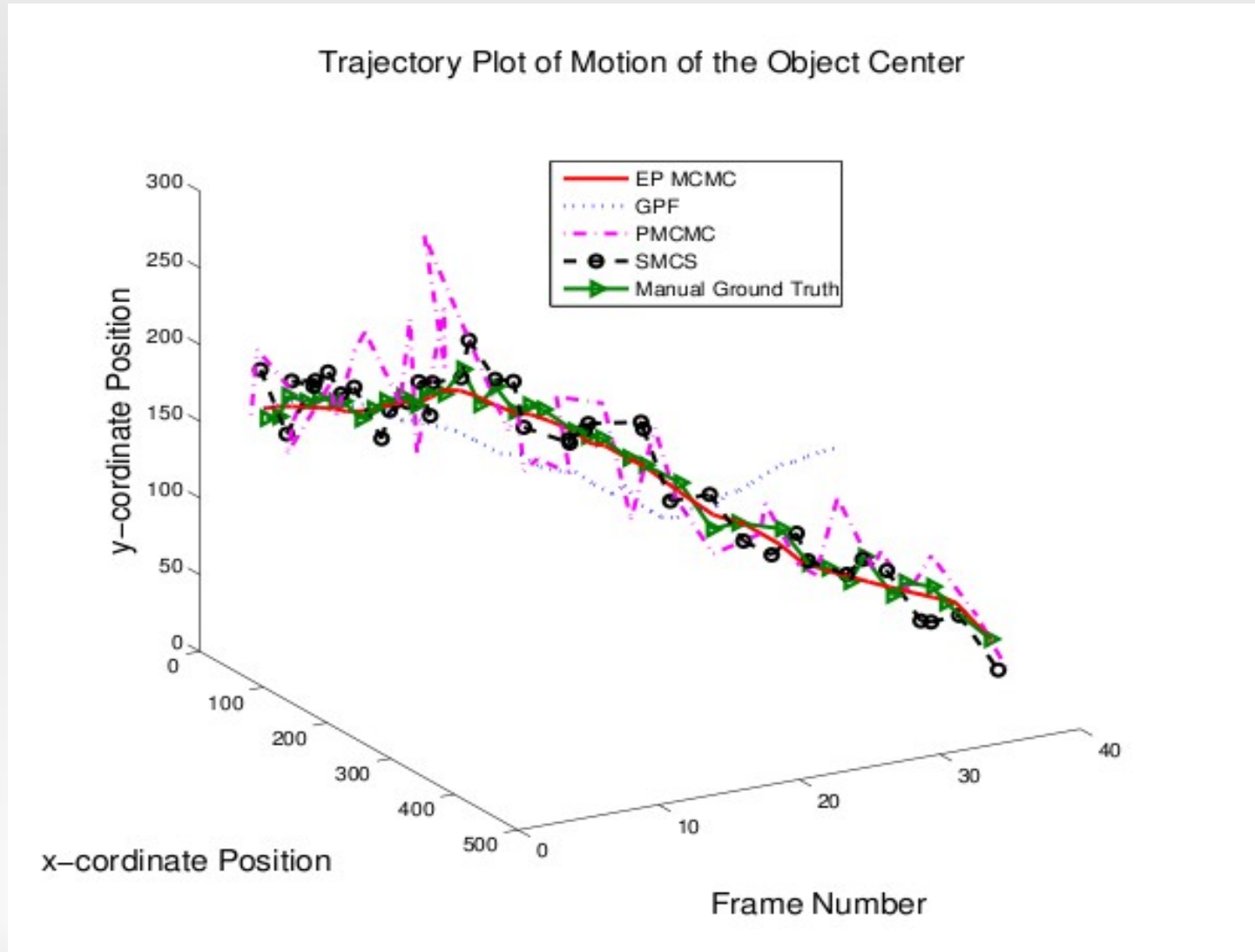
- Uses a method named Evolving Population MCMC (EP MCMC).
- The algorithm takes ideas from sequential Monte Carlo samplers, constructing samples from a sequence of target distributions which are related to the posterior.
- Idea : Generate blocks of sub-populations and apply crossover and mutation operators.

# Population-based Particle Filter (Bhaskar 2008)



- Algorithm:
  - 1: At time  $k=1$  Draw a  $n=1$  populations of  $N$  samples  $\{X_k\}$
  - Iterate steps 2-3 until  $k=T$ 
    - 2 : For each population  $j=1..n$ 
      - Resample each member of the population using ESS.
      - Perform mutation, crossover and exchange
    - 3 : Increment time step  $k=k+1$ 
      - For each sample  $i=1..N$  draw a new population  $\{X_k\}^i$  using mutation, crossover and exchange.
      - Evaluate and normalize weights  $\{W_k\}^i$

# Population-based Particle Filter (Bhaskar 2008)



# Particle swarm optimization particle filter (Hernandez, 2007)



- Mutate particles using a velocity factor computed from the sample population.

$$v_k^{i,j+1} = v_0 + c_1 r_1 (x_k^{i,\text{best}} - x_k^{i,j}) + c_2 r_2 (x_k^{\text{best},j} - x_k^{i,j}) \quad (3)$$

$$x_k^{i,\text{best}} = \operatorname{argmax}_{x_i} (w_{k-1}^i p(y_k | x_k^{i,j}))$$
$$x_k^{\text{best},j} = \operatorname{argmax}_{x_j} (w_{k-1}^i p(y_k | x_k^{i,j}))$$

- Adapt particle positions according to their velocity:

$$x_k^{i,j+1} = x_k^{i,j} + v_k^{i,j+1}$$

# Parameter estimation with PSO particle filter

