Divide-and-Conquer Fusion: Methods for unifying distributed analyses

Ryan Chan

Murray Pollock (Newcastle), Hongsheng Dai (Essex), Adam Johansen (Warwick), Gareth Roberts (Warwick)

Poster session: Tuesday 28th June (19:00-22:00)

28 June 2022

Outline

Fusion Problem

- Target fusion density:

$$f(\boldsymbol{x}) \propto \prod_{c=1}^{C} f_c(\boldsymbol{x})$$

 where each *sub-posterior*, $f_c(\boldsymbol{x})$, is a density representing one of the C distributed inferences we wish to unify

  - No general analytical approach
  - Monte Carlo: assume we can sample $\boldsymbol{x}^{(c)} \sim f_c(\boldsymbol{x})$

- Applications:
  - Expert elicitation: combining views of multiple experts
  - Big Data (by construction)
    - Partitioning large datasets to make them more manageable
  - Inference in privacy settings

Fusion Problem

- Target fusion density:

$$f(\mathbf{x}) \propto \prod_{c=1}^{C} f_c(\mathbf{x})$$

where each *sub-posterior*, $f_c(\mathbf{x})$, is a density representing one of the C distributed inferences we wish to unify

  - No general analytical approach
  - Monte Carlo: assume we can sample $\mathbf{x}^{(c)} \sim f_c(\mathbf{x})$

- Applications:
  - Expert elicitation: combining views of multiple experts
  - Big Data (by construction)
    - Partitioning large datasets to make them more manageable
  - Inference in privacy settings

Fusion Problem

- Target fusion density:

$$f(\boldsymbol{x}) \propto \prod_{c=1}^{C} f_c(\boldsymbol{x})$$

 where each *sub-posterior*, $f_c(\boldsymbol{x})$, is a density representing one of the C distributed inferences we wish to unify

  - No general analytical approach
  - Monte Carlo: assume we can sample $\boldsymbol{x}^{(c)} \sim f_c(\boldsymbol{x})$

- Applications:
  - Expert elicitation: combining views of multiple experts
  - Big Data (by construction)
    - Partitioning large datasets to make them more manageable
  - Inference in privacy settings

Fusion Problem

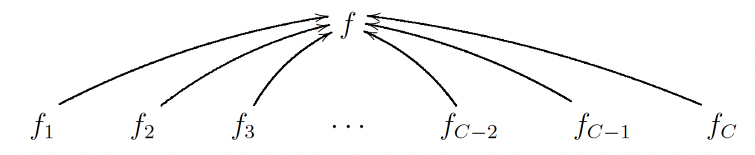
- Target fusion density:

$$f(\boldsymbol{x}) \propto \prod_{c=1}^{C} f_c(\boldsymbol{x})$$

  where each *sub-posterior*, $f_c(\boldsymbol{x})$, is a density representing one of the C distributed inferences we wish to unify
  - No general analytical approach
  - Monte Carlo: assume we can sample $\boldsymbol{x}^{(c)} \sim f_c(\boldsymbol{x})$
- Applications:
  - Expert elicitation: combining views of multiple experts
  - Big Data (by construction)
    - Partitioning large datasets to make them more manageable
  - Inference in privacy settings

Fork-and-join

The fork-and-join approach:



Some Fork-and-Join methods

- Several fork-and-join methods have been developed (typically for Bayesian inference for large dataset applications):
  - Kernel density averaging (KDEMC) [Neiswanger et al., 2014]
  - Weierstrass sampler (WRS) [Wang and Dunson, 2013]
  - Consensus Monte Carlo (CMC) [Scott et al., 2016]
- A primary weakness of these methods is that the recombination is inexact in general and involve approximations
  - CMC is exact if all sub-posteriors are Gaussian
  - All theory is asymptotic in the number of observations
- However, Monte Carlo Fusion [Dai et al., 2019] (and subsequently Bayesian Fusion [Dai et al., 2021]) is exact in the sense it targets the correct fusion density

Some Fork-and-Join methods

- Several fork-and-join methods have been developed (typically for Bayesian inference for large dataset applications):
  - Kernel density averaging (KDEMC) [Neiswanger et al., 2014]
  - Weierstrass sampler (WRS) [Wang and Dunson, 2013]
  - Consensus Monte Carlo (CMC) [Scott et al., 2016]
- A primary weakness of these methods is that the recombination is inexact in general and involve approximations
  - CMC is exact if all sub-posteriors are Gaussian
  - All theory is asymptotic in the number of observations
- However, Monte Carlo Fusion [Dai et al., 2019] (and subsequently Bayesian Fusion [Dai et al., 2021]) is exact in the sense it targets the correct fusion density

Some Fork-and-Join methods

- Several fork-and-join methods have been developed (typically for Bayesian inference for large dataset applications):
  - Kernel density averaging (KDEMC) [Neiswanger et al., 2014]
  - Weierstrass sampler (WRS) [Wang and Dunson, 2013]
  - Consensus Monte Carlo (CMC) [Scott et al., 2016]
- A primary weakness of these methods is that the recombination is inexact in general and involve approximations
  - CMC is exact if all sub-posteriors are Gaussian
  - All theory is asymptotic in the number of observations
- However, Monte Carlo Fusion [Dai et al., 2019] (and subsequently Bayesian Fusion [Dai et al., 2021]) is exact in the sense it targets the correct fusion density

Some Fork-and-Join methods

- Several fork-and-join methods have been developed (typically for Bayesian inference for large dataset applications):
  - Kernel density averaging (KDEMC) [Neiswanger et al., 2014]
  - Weierstrass sampler (WRS) [Wang and Dunson, 2013]
  - Consensus Monte Carlo (CMC) [Scott et al., 2016]
- A primary weakness of these methods is that the recombination is inexact in general and involve approximations
  - CMC is exact if all sub-posteriors are Gaussian
  - All theory is asymptotic in the number of observations
- However, Monte Carlo Fusion [Dai et al., 2019] (and subsequently Bayesian Fusion [Dai et al., 2021]) is exact in the sense it targets the correct fusion density

Some Fork-and-Join methods

- Several fork-and-join methods have been developed (typically for Bayesian inference for large dataset applications):
  - Kernel density averaging (KDEMC) [Neiswanger et al., 2014]
  - Weierstrass sampler (WRS) [Wang and Dunson, 2013]
  - Consensus Monte Carlo (CMC) [Scott et al., 2016]
- A primary weakness of these methods is that the recombination is inexact in general and involve approximations
  - CMC is exact if all sub-posteriors are Gaussian
  - All theory is asymptotic in the number of observations
- However, Monte Carlo Fusion [Dai et al., 2019] (and subsequently Bayesian Fusion [Dai et al., 2021]) is exact in the sense it targets the correct fusion density

An extended target density

Proposition

Suppose that $p_c(\boldsymbol{y}|\boldsymbol{x}^{(c)})$ is the transition density of a stochastic process with stationary distribution $f_c^2(\boldsymbol{x})$. The $(C+1)d$-dimensional (fusion) density proportional to the integrable function

$$g\left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \propto \prod_{c=1}^{C} \left[f_c^2\left(\boldsymbol{x}^{(c)}\right) \cdot p_c\left(\boldsymbol{y} \middle| \boldsymbol{x}^{(c)}\right) \cdot \frac{1}{f_c(\boldsymbol{y})} \right]$$

admits the marginal density f for $\boldsymbol{y}$.

Main idea: If we can sample from g, then we can can obtain a draw from the fusion density ($\boldsymbol{y} \sim f$)

An extended target density

Proposition

Suppose that $p_c(\boldsymbol{y}|\boldsymbol{x}^{(c)})$ is the transition density of a stochastic process with stationary distribution $f_c^2(\boldsymbol{x})$. The $(C+1)d$-dimensional (fusion) density proportional to the integrable function

$$g\left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \propto \prod_{c=1}^{C} \left[f_c^2\left(\boldsymbol{x}^{(c)}\right) \cdot p_c\left(\boldsymbol{y} \middle| \boldsymbol{x}^{(c)}\right) \cdot \frac{1}{f_c\left(\boldsymbol{y}\right)} \right]$$

admits the marginal density f for $\boldsymbol{y}$.

Main idea: If we can sample from g, then we can can obtain a draw from the fusion density ($\boldsymbol{y} \sim f$)

An extended target density

- There are many possible choices for $p_c(\boldsymbol{y}|\boldsymbol{x}^{(c)})$
- Let $p_c(\boldsymbol{y}|\boldsymbol{x}^{(c)}) := p_{T,c}(\boldsymbol{y}|\boldsymbol{x}^{(c)})$, the transition density of the d-dimensional (double) Langevin (DL) diffusion processes $\boldsymbol{X}_t^{(c)}$ from $\boldsymbol{x}^{(c)}$ to $\boldsymbol{y}$ for $c = 1, \ldots, C$, for a pre-defined time $T > 0$ given by

$$\mathrm{d}\boldsymbol{X}_t^{(c)} = \nabla \log f_c\left(\boldsymbol{X}_t^{(c)}\right) \mathrm{d}t + \mathrm{d}\boldsymbol{W}_t^{(c)},$$

(where $\boldsymbol{W}_t^{(c)}$ is d-dimensional Brownian motion and ∇ is the gradient operator over $\boldsymbol{x}$)
  - Has stationary distribution $f_c^2(\boldsymbol{x})$
  - Sample paths of DL diffusions can be simulated exactly using Path Space Rejection Sampling / Exact Algorithm methodology [Beskos et al., 2005, 2006; Pollock et al., 2016]

An extended target density

- There are many possible choices for $p_c(\boldsymbol{y}|\boldsymbol{x}^{(c)})$
- Let $p_c(\boldsymbol{y}|\boldsymbol{x}^{(c)}) := p_{T,c}(\boldsymbol{y}|\boldsymbol{x}^{(c)})$, the transition density of the d-dimensional (double) Langevin (DL) diffusion processes $\boldsymbol{X}_t^{(c)}$ from $\boldsymbol{x}^{(c)}$ to $\boldsymbol{y}$ for $c = 1, \dots, C$, for a pre-defined time $T > 0$ given by

$$\mathrm{d}\boldsymbol{X}_t^{(c)} = \nabla \log f_c \left(\boldsymbol{X}_t^{(c)}\right) \mathrm{d}t + \mathrm{d}\boldsymbol{W}_t^{(c)},$$

(where $\boldsymbol{W}_t^{(c)}$ is d-dimensional Brownian motion and ∇ is the gradient operator over $\boldsymbol{x}$)

  - Has stationary distribution $f_c^2(\boldsymbol{x})$
  - Sample paths of DL diffusions can be simulated exactly using Path Space Rejection Sampling / Exact Algorithm methodology [Beskos et al., 2005, 2006; Pollock et al., 2016]

Constructing a rejection sampler for g

- Extended target density:

$$g\left(\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(C)},\mathbf{y}\right) \propto \prod_{c=1}^{C}\left[f_c^2\left(\mathbf{x}^{(c)}\right)\cdot p_{T,c}\left(\mathbf{y}\middle|\mathbf{x}^{(c)}\right)\cdot\frac{1}{f_c\left(\mathbf{y}\right)}\right]$$

- Consider the proposal density h for the extended target g:

$$h\left(\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(C)},\mathbf{y}\right) \propto \prod_{c=1}^{C}\left[f_c\left(\mathbf{x}^{(c)}\right)\right]\cdot\exp\left(-\frac{C\cdot\|\mathbf{y}-\bar{\mathbf{x}}\|^2}{2T}\right)$$

  - $\bar{\mathbf{x}} = \frac{1}{C}\sum_{c=1}^{C}\mathbf{x}^{(c)}$
  - T is an arbitrary positive constant

Constructing a rejection sampler for g

- Extended target density:

$$g\left(\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(C)},\boldsymbol{y}\right)\propto\prod_{c=1}^{C}\left[f_c^2\left(\boldsymbol{x}^{(c)}\right)\cdot p_{T,c}\left(\boldsymbol{y}\middle|\boldsymbol{x}^{(c)}\right)\cdot\frac{1}{f_c\left(\boldsymbol{y}\right)}\right]$$

- Consider the proposal density h for the extended target g:

$$h\left(\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(C)},\boldsymbol{y}\right)\propto\prod_{c=1}^{C}\left[f_c\left(\boldsymbol{x}^{(c)}\right)\right]\cdot\exp\left(-\frac{C\cdot\left\|\boldsymbol{y}-\bar{\boldsymbol{x}}\right\|^2}{2T}\right)$$

  - $\bar{\boldsymbol{x}}=\frac{1}{C}\sum_{c=1}^{C}\boldsymbol{x}^{(c)}$
  - T is an arbitrary positive constant

Constructing a rejection sampler for g

- Simulation from h is easy:

$$h\left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \propto \prod_{c=1}^{C} \left[f_c\left(\boldsymbol{x}^{(c)}\right)\right] \cdot \exp\left(-\frac{C \cdot \|\boldsymbol{y} - \bar{\boldsymbol{x}}\|^2}{2T}\right)$$

1. Simulate $\boldsymbol{x}^{(c)} \sim f_c(\boldsymbol{x})$ independently
2. Simulate $\boldsymbol{y} \sim \mathcal{N}_d(\bar{\boldsymbol{x}}, \frac{T}{C}\mathbb{I}_d)$
    - This value $\boldsymbol{y}$ ends up being our proposal for f

Constructing a rejection sampler for g

- Simulation from h is easy:

$$h\left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \propto \prod_{c=1}^{C} \left[f_c\left(\boldsymbol{x}^{(c)}\right)\right] \cdot \exp\left(-\frac{C \cdot \|\boldsymbol{y} - \bar{\boldsymbol{x}}\|^2}{2T}\right)$$

1. Simulate $\boldsymbol{x}^{(c)} \sim f_c(\boldsymbol{x})$ independently
2. Simulate $\boldsymbol{y} \sim \mathcal{N}_d(\bar{\boldsymbol{x}}, \frac{T}{C}\mathbb{I}_d)$
    - This value $\boldsymbol{y}$ ends up being our proposal for f

Constructing a rejection sampler for g

- Simulation from h is easy:

$$h\left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \propto \prod_{c=1}^{C} \left[f_c\left(\boldsymbol{x}^{(c)}\right)\right] \cdot \exp\left(-\frac{C \cdot \|\boldsymbol{y} - \bar{\boldsymbol{x}}\|^2}{2T}\right)$$

1. Simulate $\boldsymbol{x}^{(c)} \sim f_c(\boldsymbol{x})$ independently
2. Simulate $\boldsymbol{y} \sim \mathcal{N}_d(\bar{\boldsymbol{x}}, \frac{T}{C}\mathbb{I}_d)$
   - This value $\boldsymbol{y}$ ends up being our proposal for f

Rejection sampling - acceptance probability

- Acceptance probability:

$$\frac{g(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})}{h(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})} \propto \rho_0 \cdot \rho_1$$

where

$$\begin{cases} \rho_0 := e^{-\frac{C\sigma^2}{2T}}, & \sigma^2 = \frac{1}{C}\sum_{c=1}^{C} \left\| \boldsymbol{x}^{(c)} - \bar{\boldsymbol{x}} \right\|^2 \\ \rho_1 := \mathbb{E}_{\bar{\mathbb{W}}}\left(\prod_{c=1}^{C}\left[\exp\left\{-\int_0^T \left(\phi_c\left(\boldsymbol{X}_t^{(c)}\right) - \boldsymbol{\Phi}_c\right)\mathrm{d}t\right\}\right]\right) \end{cases}$$

where $\bar{\mathbb{W}}$ denotes the law of C independent Brownian bridges $\boldsymbol{X}_t^{(1)}, \ldots, \boldsymbol{X}_t^{(C)}$ with $\boldsymbol{X}_0 = \boldsymbol{x}^{(c)}$ and $\boldsymbol{X}_T^{(c)} = \boldsymbol{y}$

- Trade-off with choice of T: as T increases, ρ_0 increases, but this results in ρ_1 to be small (might typically decrease exponentially with T)

Rejection sampling - acceptance probability

- Acceptance probability:

$$\frac{g(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})}{h(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})} \propto \rho_0 \cdot \rho_1$$

  where

$$\begin{cases} \rho_0 := e^{-\frac{C\sigma^2}{2T}}, \qquad \sigma^2 = \frac{1}{C}\sum_{c=1}^{C} \left\|\boldsymbol{x}^{(c)} - \bar{\boldsymbol{x}}\right\|^2 \\ \rho_1 := \mathbb{E}_{\bar{\mathbb{W}}}\left(\prod_{c=1}^{C}\left[\exp\left\{-\int_0^T \left(\phi_c\left(\boldsymbol{X}_t^{(c)}\right) - \boldsymbol{\Phi}_c\right) \mathrm{d}t\right\}\right]\right) \end{cases}$$

  where $\bar{\mathbb{W}}$ denotes the law of C independent Brownian bridges $\boldsymbol{X}_t^{(1)}, \ldots, \boldsymbol{X}_t^{(C)}$ with $\boldsymbol{X}_0 = \boldsymbol{x}^{(c)}$ and $\boldsymbol{X}_T^{(c)} = \boldsymbol{y}$

- Trade-off with choice of T: as T increases, ρ_0 increases, but this results in ρ_1 to be small (might typically decrease exponentially with T)

Rejection sampling - acceptance probability

- Acceptance probability:

$$\frac{g(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})}{h(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})} \propto \rho_0 \cdot \rho_1$$

where

$$\begin{cases} \rho_0 := e^{-\frac{C\sigma^2}{2T}}, \qquad \sigma^2 = \frac{1}{C}\sum_{c=1}^{C} \left\| \boldsymbol{x}^{(c)} - \bar{\boldsymbol{x}} \right\|^2 \\ \rho_1 := \mathbb{E}_{\bar{\mathbb{W}}} \left(\prod_{c=1}^{C} \left[\exp\left\{ - \int_0^T \left(\phi_c \left(\boldsymbol{X}_t^{(c)} \right) - \boldsymbol{\Phi}_c \right) \mathrm{d}t \right\} \right] \right) \end{cases}$$

where $\bar{\mathbb{W}}$ denotes the law of C independent Brownian bridges $\boldsymbol{X}_t^{(1)}, \ldots, \boldsymbol{X}_t^{(C)}$ with $\boldsymbol{X}_0 = \boldsymbol{x}^{(c)}$ and $\boldsymbol{X}_T^{(c)} = \boldsymbol{y}$

- Trade-off with choice of T: as T increases, ρ_0 increases, but this results in ρ_1 to be small (might typically decrease exponentially with T)

ρ_1 Acceptance Probability

$$\rho_1 := \mathbb{E}_{\bar{\mathbb{W}}} \left(\prod_{c=1}^{C} \left[\exp \left\{ - \int_0^T \left(\phi_c \left(\boldsymbol{X}_t^{(c)} \right) - \boldsymbol{\Phi}_c \right) \mathrm{d}t \right\} \right] \right)$$

where

- $\phi_c(\boldsymbol{x}) = \frac{1}{2} \left(\|\nabla \log f_c(\boldsymbol{x})\|^2 + \Delta \log f_c(\boldsymbol{x}) \right)$
- Φ_c are constants such that for all $\boldsymbol{x}$, $\phi_c(\boldsymbol{x}) \geq \Phi_c$ for $c \in \{1, \ldots, C\}$
- Events of probability ρ_1 can be simulated using Poisson thinning and methodology called Path-space Rejection Sampling (PSRS) or the Exact Algorithm (Beskos et al. [2005], Beskos et al. [2006], Pollock et al. [2016])

ρ_1 Acceptance Probability

$$\rho_1 := \mathbb{E}_{\bar{\mathbb{W}}}\left(\prod_{c=1}^{C}\left[\exp\left\{-\int_0^T\left(\phi_c\left(\boldsymbol{X}_t^{(c)}\right)-\boldsymbol{\Phi}_c\right)\mathrm{d}t\right\}\right]\right)$$

where

- $\phi_c(\boldsymbol{x}) = \frac{1}{2}\left(\|\nabla \log f_c(\boldsymbol{x})\|^2 + \Delta \log f_c(\boldsymbol{x})\right)$
- Φ_c are constants such that for all $\boldsymbol{x}$, $\phi_c(\boldsymbol{x}) \geq \Phi_c$ for $c \in \{1, \dots, C\}$
- Events of probability ρ_1 can be simulated using Poisson thinning and methodology called Path-space Rejection Sampling (PSRS) or the Exact Algorithm (Beskos et al. [2005], Beskos et al. [2006], Pollock et al. [2016])

ρ_1 Acceptance Probability

$$\rho_1 := \mathbb{E}_{\bar{\mathbb{W}}} \left(\prod_{c=1}^{C} \left[\exp \left\{ - \int_0^T \left(\phi_c \left(\boldsymbol{X}_t^{(c)} \right) - \boldsymbol{\Phi}_c \right) \mathrm{d}t \right\} \right] \right)$$

where

- $\phi_c(\boldsymbol{x}) = \frac{1}{2} \left(\|\nabla \log f_c(\boldsymbol{x})\|^2 + \Delta \log f_c(\boldsymbol{x}) \right)$
- Φ_c are constants such that for all $\boldsymbol{x}$, $\phi_c(\boldsymbol{x}) \geq \Phi_c$ for $c \in \{1, \ldots, C\}$
- Events of probability ρ_1 can be simulated using Poisson thinning and methodology called Path-space Rejection Sampling (PSRS) or the Exact Algorithm (Beskos et al. [2005], Beskos et al. [2006], Pollock et al. [2016])

Monte Carlo Fusion - Summary

- Aim: Sample from g (admits marginal density f for $\boldsymbol{y}$)
- Proposal:

$$h\left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \propto \prod_{c=1}^{C}\left[f_c\left(\boldsymbol{x}^{(c)}\right)\right] \cdot \exp\left(-\frac{C \cdot \|\boldsymbol{y} - \bar{\boldsymbol{x}}\|^2}{2T}\right)$$

- Accept $\boldsymbol{y}$ as a draw from fusion density f with probability:

$$\frac{g(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})}{h(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})} \propto \rho_0 \cdot \rho_1$$

Monte Carlo Fusion - Summary

- Aim: Sample from g (admits marginal density f for $\boldsymbol{y}$)
- Proposal:

$$h\left(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \propto \prod_{c=1}^{C} \left[f_c\left(\boldsymbol{x}^{(c)}\right)\right] \cdot \exp\left(-\frac{C \cdot \|\boldsymbol{y} - \bar{\boldsymbol{x}}\|^2}{2T}\right)$$

- Accept $\boldsymbol{y}$ as a draw from fusion density f with probability:

$$\frac{g(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(C)}, \boldsymbol{y})}{h(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(C)}, \boldsymbol{y})} \propto \rho_0 \cdot \rho_1$$

Monte Carlo Fusion - Summary

- Aim: Sample from g (admits marginal density f for $\boldsymbol{y}$)
- Proposal:

$$h\left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \propto \prod_{c=1}^{C} \left[f_c\left(\boldsymbol{x}^{(c)}\right)\right] \cdot \exp\left(-\frac{C \cdot \|\boldsymbol{y} - \bar{\boldsymbol{x}}\|^2}{2T}\right)$$

- Accept $\boldsymbol{y}$ as a draw from fusion density f with probability:

$$\frac{g(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})}{h(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})} \propto \rho_0 \cdot \rho_1$$

Limitations of Monte Carlo Fusion

- Robustness: there is a lack of robustness when:
  - sub-posterior correlation increases
  - C increases
  - d increases
  - combining conflicting sub-posteriors
- Aim: To construct a fusion algorithm / framework to alleviate some of these limitations (see Dai et al. [2021]; Chan et al. [2021] for full details)

Limitations of Monte Carlo Fusion

- Robustness: there is a lack of robustness when:
  - sub-posterior correlation increases
  - C increases
  - d increases
  - combining conflicting sub-posteriors
- Aim: To construct a fusion algorithm / framework to alleviate some of these limitations (see Dai et al. [2021]; Chan et al. [2021] for full details)

The Generalised Monte Carlo Fusion (GMCF) approach

Problem: Scalability with sub-posterior correlation

- Recall we have the flexibility to choose different p_c (transition density of stochastic process with f_c^2 invariant density)
- Now, we choose p_c to be the transition density of the d-dimensional (double) Langevin (DL) diffusion processes $\boldsymbol{X}_t^{(c)}$ with covariance matrix, $\boldsymbol{\Lambda}_c$ from $\boldsymbol{x}^{(c)}$ to $\boldsymbol{y}$ for $c = 1, \ldots, C$, over $[0, T]$ given by

$$\mathrm{d}\boldsymbol{X}_t^{(c)} = \boldsymbol{\Lambda}_c \nabla \log f_c \left(\boldsymbol{X}_t^{(c)}\right) \mathrm{d}t + \boldsymbol{\Lambda}_c^{1/2} \mathrm{d}\boldsymbol{W}_t^{(c)},$$

  - Has stationary density proportional to $f_c^2(\boldsymbol{x})$
  - $\boldsymbol{\Lambda}_c$ is the *preconditioning matrix* (enables incorporation of covariance / correlation structure into our algorithm)

The Generalised Monte Carlo Fusion (GMCF) approach

Problem: Scalability with sub-posterior correlation

- Recall we have the flexibility to choose different p_c (transition density of stochastic process with f_c^2 invariant density)
- Now, we choose p_c to be the transition density of the d-dimensional (double) Langevin (DL) diffusion processes $\boldsymbol{X}_t^{(c)}$ with covariance matrix, $\boldsymbol{\Lambda}_c$ from $\boldsymbol{x}^{(c)}$ to $\boldsymbol{y}$ for $c = 1, \ldots, C$, over $[0, T]$ given by

$$\mathrm{d}\boldsymbol{X}_t^{(c)} = \boldsymbol{\Lambda}_c \nabla \log f_c \left(\boldsymbol{X}_t^{(c)} \right) \mathrm{d}t + \boldsymbol{\Lambda}_c^{1/2} \mathrm{d}\boldsymbol{W}_t^{(c)},$$

  - Has stationary density proportional to $f_c^2(\boldsymbol{x})$
  - $\boldsymbol{\Lambda}_c$ is the *preconditioning matrix* (enables incorporation of covariance / correlation structure into our algorithm)

Constructing an importance sampler

- Switch to importance sampler for the extended target density $g(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y})$:
  - Rejection sampling can be wasteful
  - We will subsequently embed this approach within a SMC algorithm
- Consider an alternative proposal density h for the extended target g:

$$h\left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \propto \prod_{c=1}^{C} \left[f_c\left(\boldsymbol{x}^{(c)}\right)\right] \cdot \exp\left\{ -\frac{(\boldsymbol{y}-\tilde{\boldsymbol{x}})^\intercal \boldsymbol{\Lambda}^{-1} (\boldsymbol{y}-\tilde{\boldsymbol{x}})}{2T} \right\},$$

where

$$\tilde{\boldsymbol{x}} := \left(\sum_{c=1}^{C} \boldsymbol{\Lambda}_c^{-1} \right)^{-1} \left(\sum_{c=1}^{C} \boldsymbol{\Lambda}_c^{-1} \boldsymbol{x}^{(c)} \right), \qquad \boldsymbol{\Lambda}^{-1} := \sum_{c=1}^{C} \boldsymbol{\Lambda}_c^{-1}.$$

Constructing an importance sampler

- Switch to importance sampler for the extended target density $g(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(C)}, \boldsymbol{y})$:
  - Rejection sampling can be wasteful
  - We will subsequently embed this approach within a SMC algorithm
- Consider an alternative proposal density h for the extended target g:

$$h\left(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(C)}, \boldsymbol{y}\right) \propto \prod_{c=1}^{C} \left[f_c \left(\boldsymbol{x}^{(c)} \right) \right] \cdot \exp \left\{ -\frac{(\boldsymbol{y} - \tilde{\boldsymbol{x}})^{\intercal} \boldsymbol{\Lambda}^{-1} (\boldsymbol{y} - \tilde{\boldsymbol{x}})}{2T} \right\},$$

where

$$\tilde{\boldsymbol{x}} := \left(\sum_{c=1}^{C} \boldsymbol{\Lambda}_c^{-1} \right)^{-1} \left(\sum_{c=1}^{C} \boldsymbol{\Lambda}_c^{-1} \boldsymbol{x}^{(c)} \right), \qquad \boldsymbol{\Lambda}^{-1} := \sum_{c=1}^{C} \boldsymbol{\Lambda}_c^{-1}.$$

Importance weights

- Importance weights:

$$\frac{g(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(C)}, \boldsymbol{y})}{h(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(C)}, \boldsymbol{y})} \propto \rho_0 \cdot \rho_1$$

where

$$\begin{cases} \rho_0 := \exp\left\{ -\sum_{c=1}^{C} \frac{(\tilde{\boldsymbol{x}} - \boldsymbol{x}^{(c)})^\intercal \boldsymbol{\Lambda}_c^{-1} (\tilde{\boldsymbol{x}} - \boldsymbol{x}^{(c)})}{2T} \right\} \\ \rho_1 := \prod_{c=1}^{C} \mathbb{E}_{\mathbb{W}_{\boldsymbol{\Lambda}_c}} \left[\exp\left\{ -\int_0^T \left(\phi_c\left(\boldsymbol{X}_t^{(c)}\right) - \boldsymbol{\Phi}_c \right) \mathrm{d}t \right\} \right] \end{cases}$$

where $\phi_c(\boldsymbol{x}) := \frac{1}{2}\left(\nabla \log f_c(\boldsymbol{x})^\intercal \boldsymbol{\Lambda}_c \nabla \log f_c(\boldsymbol{x}) + \mathrm{Tr}\left(\boldsymbol{\Lambda}_c \nabla^2 \log f_c(\boldsymbol{x})\right)\right)$, with $\mathbb{W}_{\boldsymbol{\Lambda}_c}$ denoting the law of a Brownian bridge $\{\boldsymbol{X}_t^{(c)}, t \in [0, T]\}$ with $\boldsymbol{X}_0^{(c)} := \boldsymbol{x}^{(c)}$, $\boldsymbol{X}_T^{(c)} := \boldsymbol{y}$ and covariance matrix $\boldsymbol{\Lambda}_c$

Scalability with sub-posterior correlation

In our *Generalised* Monte Carlo Fusion setting:

- Able to incorporate covariance / correlation information within our proposals and through p_c and h (in MCF $\mathbf{\Lambda}_c = \mathbb{I}_d$ for $c = 1, \ldots, C$)
- Unfortunately no longer have i.i.d. draws from f but now have weighted samples to approximate f (later embed within divide-and-conquer SMC [Lindsten et al., 2017] framework)

Scalability with sub-posterior correlation

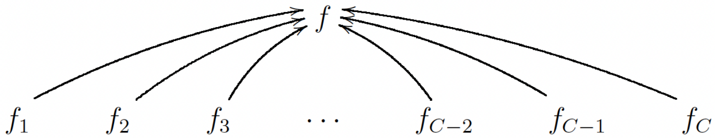
In our *Generalised* Monte Carlo Fusion setting:

- Able to incorporate covariance / correlation information within our proposals and through p_c and h (in MCF $\mathbf{\Lambda}_c = \mathbb{I}_d$ for $c = 1, \ldots, C$)
- Unfortunately no longer have i.i.d. draws from f but now have weighted samples to approximate f (later embed within divide-and-conquer SMC [Lindsten et al., 2017] framework)

Divide-and-Conquer Monte Carlo Fusion

Problem: Scalability with C

The (Generalised) Monte Carlo Fusion algorithm implies a fork-and-join approach:

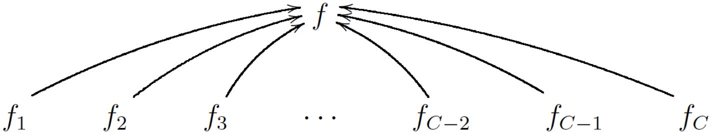


- Not necessarily the most efficient way to combine sub-posteriors
- For MCF, acceptance probabilities typically decrease geometrically with C

Divide-and-Conquer Monte Carlo Fusion

Problem: Scalability with C

The (Generalised) Monte Carlo Fusion algorithm implies a fork-and-join approach:

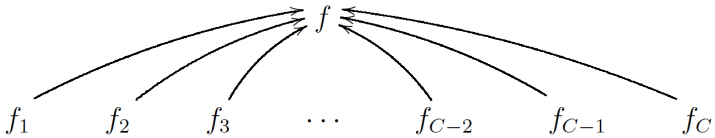


- Not necessarily the most efficient way to combine sub-posteriors
- For MCF, acceptance probabilities typically decrease geometrically with C

Divide-and-Conquer Monte Carlo Fusion

Problem: Scalability with C

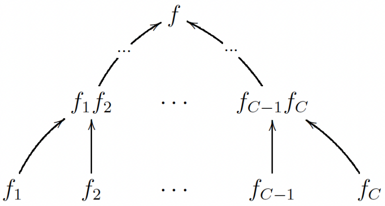
The (Generalised) Monte Carlo Fusion algorithm implies a fork-and-join approach:

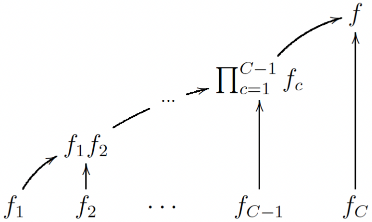


- Not necessarily the most efficient way to combine sub-posteriors
- For MCF, acceptance probabilities typically decrease geometrically with C

Divide-and-Conquer Monte Carlo Fusion

- Solution: Divide-and-Conquer Monte Carlo Fusion
  - We could perform fusion in a proper divide-and-conquer framework
    - i.e. a fork-and-join method is recursively applied
  - Two possible choices are balanced-binary (left) and progressive (right) trees

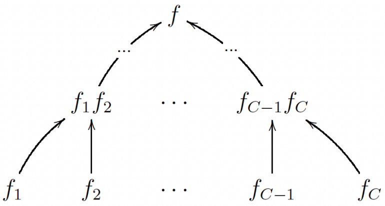


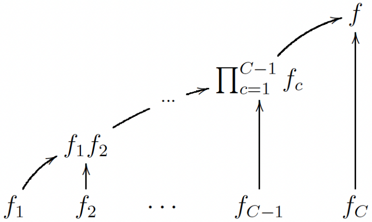


Note: Other trees are possible

Divide-and-Conquer Monte Carlo Fusion

- Solution: Divide-and-Conquer Monte Carlo Fusion
  - We could perform fusion in a proper divide-and-conquer framework
    - i.e. a fork-and-join method is recursively applied
  - Two possible choices are balanced-binary (left) and progressive (right) trees

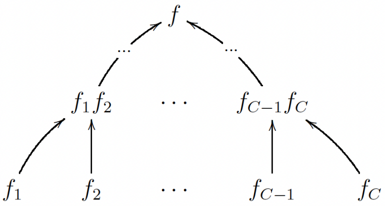


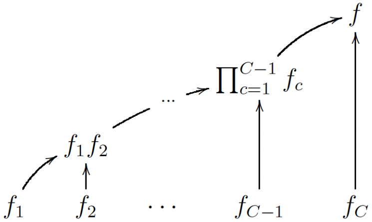


Note: Other trees are possible

Divide-and-Conquer Monte Carlo Fusion

- Solution: Divide-and-Conquer Monte Carlo Fusion
  - We could perform fusion in a proper divide-and-conquer framework
    - i.e. a fork-and-join method is recursively applied
  - Two possible choices are balanced-binary (left) and progressive (right) trees



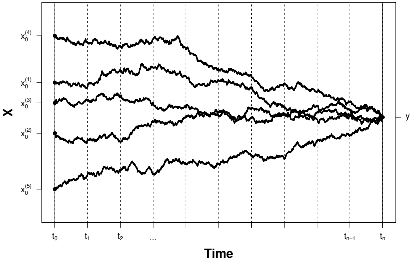


Note: Other trees are possible

Generalised Bayesian Fusion

Problem: Robustness to conflicting sub-posteriors

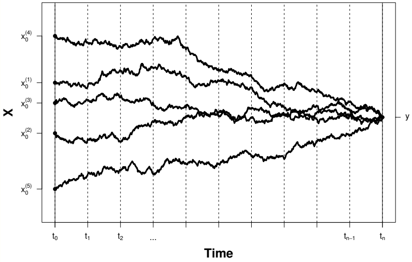
- Generalising the Bayesian Fusion approach of Dai et al. [2021]
- Recall choosing a value $T > 0$ for MCF can be hard:
  - Want to make T large so that ρ_0 is large - but this makes ρ_1 smaller (since we have to simulate a diffusion over a longer time horizon T)
- Solution: Introduce temporal partition of T
  - Have the flexibility to choose T large enough for initialisation, while being able to have small intervals in the partition



Generalised Bayesian Fusion

Problem: Robustness to conflicting sub-posteriors

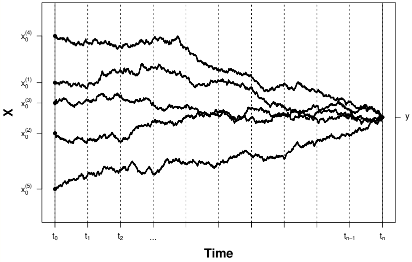
- Generalising the Bayesian Fusion approach of Dai et al. [2021]
- Recall choosing a value $T > 0$ for MCF can be hard:
    - Want to make T large so that ρ_0 is large - but this makes ρ_1 smaller (since we have to simulate a diffusion over a longer time horizon T)
- Solution: Introduce temporal partition of T
    - Have the flexibility to choose T large enough for initialisation, while being able to have small intervals in the partition



Generalised Bayesian Fusion

Problem: Robustness to conflicting sub-posteriors

- Generalising the Bayesian Fusion approach of Dai et al. [2021]
- Recall choosing a value $T > 0$ for MCF can be hard:
  - Want to make T large so that ρ_0 is large - but this makes ρ_1 smaller (since we have to simulate a diffusion over a longer time horizon T)
- Solution: Introduce temporal partition of T
  - Have the flexibility to choose T large enough for initialisation, while being able to have small intervals in the partition



Examples

- We compare our methodology with the approximate methodologies KDEMC [Neiswanger et al., 2014], WRS [Wang and Dunson, 2013] and CMC [Scott et al., 2016]
- To compare methods we calculate the integrated absolute distance metric

$$IAD = \frac{1}{2d} \sum_{j=1}^{d} \int \left| \hat{f}(\boldsymbol{x}_j) - f(\boldsymbol{x}_j) \right| \mathrm{d}\boldsymbol{x}_j \in [0, 1]$$

  where $\hat{f}(\boldsymbol{x}_j)$ is the marginal density for $\boldsymbol{x}_j$ based on the method applied (computed using a kernel density estimate) and $f(\boldsymbol{x}_j)$ is target marginal density
  - Gives a measure of how accurate our samples are to our target (lower is better)

Examples

- We compare our methodology with the approximate methodologies KDEMC [Neiswanger et al., 2014], WRS [Wang and Dunson, 2013] and CMC [Scott et al., 2016]
- To compare methods we calculate the integrated absolute distance metric

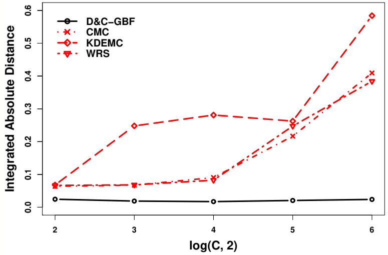
$$IAD = \frac{1}{2d} \sum_{j=1}^{d} \int \left| \hat{f}(\boldsymbol{x}_j) - f(\boldsymbol{x}_j) \right| \mathrm{d}\boldsymbol{x}_j \in [0, 1]$$

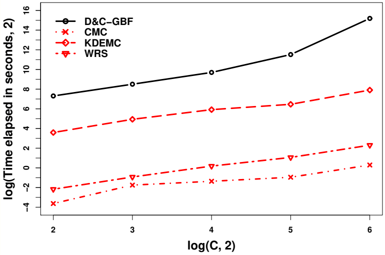
where $\hat{f}(\boldsymbol{x}_j)$ is the marginal density for $\boldsymbol{x}_j$ based on the method applied (computed using a kernel density estimate) and $f(\boldsymbol{x}_j)$ is target marginal density

  - Gives a measure of how accurate our samples are to our target (lower is better)

Logistic regression

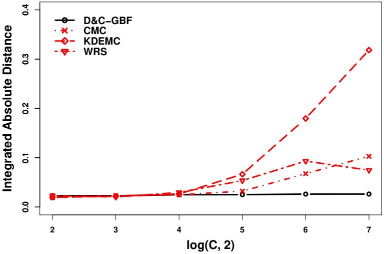
- Simulated data example with $n = 1000$ and $d = 5$ (and set $N = 10000$)
  - Small data size means that large data assumptions will fail
- We split the data into $C = 4, 8, 16, 32, 64$ and apply D&C-GBF (using a balanced binary tree approach)

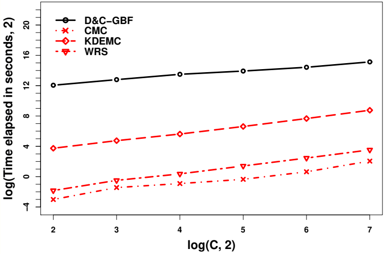




Negative Binomial regression

- Using the Bike sharing dataset ($n = 17379$, $d = 10$) (and set $N = 10000$)
- We split the data into $C = 4, 8, 16, 32, 64, 128$ and apply D&C-GBF (using a balanced binary tree approach)





Ongoing research questions

- Reducing the computational cost of the Fusion approach
  - Exactness comes at a cost
- Practical implementation considerations for specific applications:
  - Big data setting: evaluating ϕ_c has $\mathcal{O}(m_c)$ cost - can perhaps employ sub-sampling methods to reduce this cost
  - Confidential fusion (Con-fusion): where sharing information/data between cores is not permitted
- Scalability with dimension
  - Performance with regards to dimension has improved since MCF, but not been explicitly addressed

References

Beskos, A., Papaspiliopoulos, O., Roberts, G. O., and Fearnhead, P. (2006). Exact and computationally efficient likelihood-based estimation for discretely observed diffusion processes (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):333–382.

Beskos, A., Roberts, G. O., et al. (2005). Exact simulation of diffusions. *The Annals of Applied Probability*, 15(4):2422–2444.

Chan, R. S., Pollock, M., Johansen, A. M., and Roberts, G. O. (2021). Divide-and-Conquer Monte Carlo Fusion. Statistics e-print 2110.07265, arXiv.

Dai, H., Pollock, M., and Roberts, G. O. (2019). Monte Carlo Fusion. *Journal of Applied Probability*, 56(1):174–191.

Dai, H., Pollock, M., and Roberts, G. O. (2021). Bayesian Fusion: Scalable unification of distributed statistical analyses. Statistics e-print 2102.02123, arXiv.

Lindsten, F., Johansen, A. M., Naesseth, C. A., Kirkpatrick, B., Schön, T. B., Aston, J. A., and Bouchard-Côté, A. (2017). Divide-and-Conquer with Sequential Monte Carlo. *Journal of Computational and Graphical Statistics*, 26(2):445–458.

Neiswanger, W., Wang, C., and Xing, E. P. (2014). Asymptotically Exact, Embarrassingly Parallel MCMC. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, UAI'14, page 623–632, Arlington, Virginia, USA. AUAI Press.

Pollock, M., Johansen, A. M., Roberts, G. O., et al. (2016). On the exact and ε-strong simulation of (jump) diffusions. *Bernoulli*, 22(2):794–856.

Scott, S. L., Blocker, A. W., Bonassi, F. V., Chipman, H. A., George, E. I., and McCulloch, R. E. (2016). Bayes and Big Data: The Consensus Monte Carlo Algorithm. *International Journal of Management Science and Engineering Management*, 11(2):78–88.

Wang, X. and Dunson, D. B. (2013). Parallelizing MCMC via Weierstrass Sampler. Statistics e-print 1312.4605, arXiv.

Poster session: Tuesday 28th June (19:00-22:00)