Monotonic Alpha-Divergence Variational Inference

Kamélia Daudel

Research collaboration day – 27/04/2022

Joint work with Randal Douc and François Roueff

Introduction

- Bayesian Inference : complex posterior density $p(y|\mathscr{D})$ only known up to a constant

- Variational Inference :
  1. Posit a **simpler** variational family $\mathcal{Q}$
  2. Find the best approximation to the posterior density belonging to $\mathcal{Q}$: solve an optimisation problem involving a measure of dissimilarity D

$$\inf_{q \in \mathcal{Q}} D(q \,||\, p(\cdot|\mathscr{D}))$$

- Typically, D is the *exclusive* Kullback-Leibler (KL) divergence and $\mathcal{Q}$ is parametric

$$\mathcal{Q} = \{q : y \mapsto k(\theta, y) \ : \ \theta \in \mathsf{T}\}$$

with θ for example being optimised via stochastic gradient descent.

- Yet, the exclusive KL has some known limitations (e.g. posterior variance underestimation, cannot capture multimodality) and $\mathcal{Q}$ might also be too restrictive

Introduction

- Bayesian Inference : complex posterior density $p(y|\mathscr{D})$ only known up to a constant

- Variational Inference :
  1. Posit a **simpler** variational family $\mathcal{Q}$
  2. Find the best approximation to the posterior density belonging to $\mathcal{Q}$: solve an optimisation problem involving a measure of dissimilarity D

$$\inf_{q \in \mathcal{Q}} D(q \,||\, p(\cdot|\mathscr{D}))$$

- Typically, D is the *exclusive* Kullback-Leibler (KL) divergence and $\mathcal{Q}$ is parametric

$$\mathcal{Q} = \{q : y \mapsto k(\theta, y) \ : \ \theta \in \mathsf{T}\}$$

  with θ for example being optimised via stochastic gradient descent.

- Yet, the exclusive KL has some known limitations (e.g. posterior variance underestimation, cannot capture multimodality) and $\mathcal{Q}$ might also be too restrictive

Introduction

- Bayesian Inference : complex posterior density $p(y|\mathscr{D})$ only known up to a constant

- Variational Inference :
  1. Posit a **simpler** variational family $\mathcal{Q}$
  2. Find the best approximation to the posterior density belonging to $\mathcal{Q}$: solve an optimisation problem involving a measure of dissimilarity D

$$\inf_{q \in \mathcal{Q}} D(q \,||\, p(\cdot|\mathscr{D}))$$

- Typically, D is the *exclusive* Kullback-Leibler (KL) divergence and $\mathcal{Q}$ is parametric

$$\mathcal{Q} = \{q : y \mapsto k(\theta, y) \ : \ \theta \in \mathsf{T}\}$$

  with θ for example being optimised via stochastic gradient descent.

- Yet, the exclusive KL has some known limitations (e.g. posterior variance underestimation, cannot capture multimodality) and $\mathcal{Q}$ might also be too restrictive

Introduction

- Bayesian Inference : complex posterior density $p(y|\mathscr{D})$ only known up to a constant

- Variational Inference :
  1. Posit a **simpler** variational family $\mathcal{Q}$
  2. Find the best approximation to the posterior density belonging to $\mathcal{Q}$: solve an optimisation problem involving a measure of dissimilarity D

$$\inf_{q \in \mathcal{Q}} D(q \mid\mid p(\cdot|\mathscr{D}))$$

- Typically, D is the *exclusive* Kullback-Leibler (KL) divergence and $\mathcal{Q}$ is parametric

$$\mathcal{Q} = \{q : y \mapsto k(\theta, y) \; : \; \theta \in \mathsf{T}\}$$

  with θ for example being optimised via stochastic gradient descent.

- Yet, the exclusive KL has some known limitations (e.g. posterior variance underestimation, cannot capture multimodality) and $\mathcal{Q}$ might also be too restrictive

An idea

Instead of the exclusive KL divergence and $\mathcal{Q}$ of the form

$$\mathcal{Q} = \{q : y \mapsto k(\theta, y) \ : \ \theta \in \mathsf{T}\} \ ,$$

consider the α-divergence and choose $\mathcal{Q}$ of the form

$$\mathcal{Q} = \left\{ q : y \mapsto \mu_{\boldsymbol{\lambda},\Theta} k(y) := \sum_{j=1}^{J} \lambda_j k(\theta_j, y) \ : \ \boldsymbol{\lambda} \in \mathcal{S}_J, \Theta \in \mathsf{T}^J \right\} \ .$$

$\rightarrow$ Why is that a good idea?

1. The α-divergence family is more general and permits to bypass some issues of the exclusive KL divergence when $\alpha < 1$
2. Optimising w.r.t. $\boldsymbol{\lambda}$ and Θ expands the traditional parametric variational family $\rightarrow$ Optimising w.r.t. $\boldsymbol{\lambda}$ enables to select the mixture components according to their overall importance in the set of component parameters Θ

An idea

Instead of the exclusive KL divergence and $\mathcal{Q}$ of the form

$$\mathcal{Q} = \{q : y \mapsto k(\theta, y) \ : \ \theta \in \mathsf{T}\} \ ,$$

consider the α-divergence and choose $\mathcal{Q}$ of the form

$$\mathcal{Q} = \left\{ q : y \mapsto \mu_{\boldsymbol{\lambda},\Theta} k(y) := \sum_{j=1}^{J} \lambda_j k(\theta_j, y) \ : \ \boldsymbol{\lambda} \in \mathcal{S}_J, \Theta \in \mathsf{T}^J \right\} .$$

→ Why is that a good idea?

1. The α-divergence family is more general and permits to bypass some issues of the exclusive KL divergence when $\alpha < 1$
2. Optimising w.r.t. $\boldsymbol{\lambda}$ and Θ expands the traditional parametric variational family → Optimising w.r.t. $\boldsymbol{\lambda}$ enables to select the mixture components according to their overall importance in the set of component parameters Θ

An idea

Instead of the exclusive KL divergence and $\mathcal{Q}$ of the form

$$\mathcal{Q} = \{q : y \mapsto k(\theta, y) \ : \ \theta \in \mathsf{T}\} \ ,$$

consider the α-divergence and choose $\mathcal{Q}$ of the form

$$\mathcal{Q} = \left\{ q : y \mapsto \mu_{\boldsymbol{\lambda},\Theta} k(y) := \sum_{j=1}^{J} \lambda_j k(\theta_j, y) \ : \ \boldsymbol{\lambda} \in \mathcal{S}_J, \Theta \in \mathsf{T}^J \right\} \ .$$

$\rightarrow$ Why is that a good idea?

1. The α-divergence family is more general and permits to bypass some issues of the exclusive KL divergence when $\alpha < 1$
2. Optimising w.r.t. $\boldsymbol{\lambda}$ and Θ expands the traditional parametric variational family $\rightarrow$ Optimising w.r.t. $\boldsymbol{\lambda}$ enables to select the mixture components according to their overall importance in the set of component parameters Θ

An idea

Instead of the exclusive KL divergence and $\mathcal{Q}$ of the form

$$\mathcal{Q} = \{q : y \mapsto k(\theta, y) \ : \ \theta \in \mathsf{T}\} \ ,$$

consider the α-divergence and choose $\mathcal{Q}$ of the form

$$\mathcal{Q} = \left\{ q : y \mapsto \mu_{\boldsymbol{\lambda},\Theta} k(y) := \sum_{j=1}^{J} \lambda_j k(\theta_j, y) \ : \ \boldsymbol{\lambda} \in \mathcal{S}_J, \Theta \in \mathsf{T}^J \right\} \ .$$

$\rightarrow$ Why is that a good idea?

1. The α-divergence family is more general and permits to bypass some issues of the exclusive KL divergence when $\alpha < 1$
2. Optimising w.r.t. $\boldsymbol{\lambda}$ and Θ expands the traditional parametric variational family $\rightarrow$ Optimising w.r.t. $\boldsymbol{\lambda}$ enables to select the mixture components according to their overall importance in the set of component parameters Θ

An idea - cont'd

Instead of the exclusive KL divergence and $\mathcal{Q}$ of the form

$$\mathcal{Q} = \{q : y \mapsto k(\theta, y) \ : \ \theta \in \mathsf{T}\} \ ,$$

consider the α-divergence and choose $\mathcal{Q}$ of the form

$$\mathcal{Q} = \left\{ q : y \mapsto \mu_{\boldsymbol{\lambda},\Theta} k(y) := \sum_{j=1}^{J} \lambda_j k(\theta_j, y) \ : \ \boldsymbol{\lambda} \in \mathcal{S}_J, \Theta \in \mathsf{T}^J \right\} \ .$$

$\rightarrow$ What are the challenges?

1. The optimisation w.r.t $\boldsymbol{\lambda}$ is over a constrained space (the simplex)
2. How do we establish theoretical guarantees?

Monotonic Alpha-divergence Minimisation for Variational Inference.
K. Daudel, R. Douc and F. Roueff (2021). https://arxiv.org/abs/2103.05684

Goal : Propose valid update formulas for $(\boldsymbol{\lambda}, \Theta)$ that ensures a **systematic decrease in the α-divergence** $D_\alpha(\mu_{\boldsymbol{\lambda},\Theta} k \,||\, p(\cdot|\mathscr{D}))$ at each step, with $\alpha \in [0, 1)$.

An idea - cont'd

Instead of the exclusive KL divergence and $\mathcal{Q}$ of the form

$$\mathcal{Q} = \{q : y \mapsto k(\theta, y) \ : \ \theta \in \mathsf{T}\} \ ,$$

consider the α-divergence and choose $\mathcal{Q}$ of the form

$$\mathcal{Q} = \left\{ q : y \mapsto \mu_{\boldsymbol{\lambda},\Theta} k(y) := \sum_{j=1}^{J} \lambda_j k(\theta_j, y) \ : \ \boldsymbol{\lambda} \in \mathcal{S}_J, \Theta \in \mathsf{T}^J \right\} \ .$$

$\rightarrow$ What are the challenges?

1. The optimisation w.r.t $\boldsymbol{\lambda}$ is over a constrained space (the simplex)
2. How do we establish theoretical guarantees?

Monotonic Alpha-divergence Minimisation for Variational Inference.
K. Daudel, R. Douc and F. Roueff (2021). https://arxiv.org/abs/2103.05684

Goal : Propose valid update formulas for $(\boldsymbol{\lambda}, \Theta)$ that ensures a **systematic decrease in the α-divergence** $D_\alpha(\mu_{\boldsymbol{\lambda},\Theta} k \ || \ p(\cdot|\mathscr{D}))$ at each step, with $\alpha \in [0, 1)$.

An idea - cont'd

Instead of the exclusive KL divergence and $\mathcal{Q}$ of the form

$$\mathcal{Q} = \{q : y \mapsto k(\theta, y) \ : \ \theta \in \mathsf{T}\} \ ,$$

consider the α-divergence and choose $\mathcal{Q}$ of the form

$$\mathcal{Q} = \left\{ q : y \mapsto \mu_{\boldsymbol{\lambda},\Theta} k(y) := \sum_{j=1}^{J} \lambda_j k(\theta_j, y) \ : \ \boldsymbol{\lambda} \in \mathcal{S}_J, \Theta \in \mathsf{T}^J \right\} \ .$$

$\rightarrow$ What are the challenges?

1. The optimisation w.r.t $\boldsymbol{\lambda}$ is over a constrained space (the simplex)
2. How do we establish theoretical guarantees?

Monotonic Alpha-divergence Minimisation for Variational Inference.
K. Daudel, R. Douc and F. Roueff (2021). https://arxiv.org/abs/2103.05684

Goal : Propose valid update formulas for $(\boldsymbol{\lambda}, \Theta)$ that ensures a **systematic decrease in the α-divergence** $D_\alpha(\mu_{\boldsymbol{\lambda},\Theta} k \,||\, p(\cdot|\mathscr{D}))$ at each step, with $\alpha \in [0, 1)$.

An idea - cont'd

Instead of the exclusive KL divergence and $\mathcal{Q}$ of the form

$$\mathcal{Q} = \{q : y \mapsto k(\theta, y) \ : \ \theta \in \mathsf{T}\} \ ,$$

consider the α-divergence and choose $\mathcal{Q}$ of the form

$$\mathcal{Q} = \left\{ q : y \mapsto \mu_{\boldsymbol{\lambda},\Theta} k(y) := \sum_{j=1}^{J} \lambda_j k(\theta_j, y) \ : \ \boldsymbol{\lambda} \in \mathcal{S}_J, \Theta \in \mathsf{T}^J \right\} \ .$$

$\rightarrow$ What are the challenges?

1. The optimisation w.r.t $\boldsymbol{\lambda}$ is over a constrained space (the simplex)
2. How do we establish theoretical guarantees?

Monotonic Alpha-divergence Minimisation for Variational Inference.
K. Daudel, R. Douc and F. Roueff (2021). https://arxiv.org/abs/2103.05684

Goal : Propose valid update formulas for $(\boldsymbol{\lambda}, \Theta)$ that ensures a **systematic decrease in the** α**-divergence** $D_\alpha(\mu_{\boldsymbol{\lambda},\Theta} k \mid\mid p(\cdot|\mathscr{D}))$ at each step, with $\alpha \in [0, 1)$.

Monotonic Alpha-Divergence Minimisation

- Goal : Given $(\boldsymbol{\lambda}_n, \Theta_n)$ find $(\boldsymbol{\lambda}_{n+1}, \Theta_{n+1})$ such that

$$D_\alpha(\mu_{\boldsymbol{\lambda}_{n+1},\Theta_{n+1}} k \mid\mid p(\cdot|\mathscr{D})) \leqslant D_\alpha(\mu_{\boldsymbol{\lambda}_n,\Theta_n} k \mid\mid p(\cdot|\mathscr{D}))$$

- Core step : simplify the problem by writing conditions enabling separate (and simultaneous!) updates for $\boldsymbol{\lambda}_{n+1}$ and Θ_{n+1}

$$\int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{\lambda_{j,n+1}}{\lambda_{j,n}}\right) \nu(\mathrm{d}y) \geqslant 0 \qquad \text{(Weights)}$$

$$\int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{k(\theta_{j,n+1}, y)}{k(\theta_{j,n}, y)}\right) \nu(\mathrm{d}y) \geqslant 0 \qquad \text{(Components)}$$

where $\varphi_{j,n}^{(\alpha)}(y) = k(\theta_{j,n}, y)\left(\frac{\mu_{\boldsymbol{\lambda}_n,\Theta_n} k(y)}{p(y,\mathscr{D})}\right)^{\alpha-1}$

Monotonic Alpha-Divergence Minimisation

- Goal : Given $(\boldsymbol{\lambda}_n, \Theta_n)$ find $(\boldsymbol{\lambda}_{n+1}, \Theta_{n+1})$ such that

$$D_\alpha(\mu_{\boldsymbol{\lambda}_{n+1},\Theta_{n+1}}k \,||\, p(\cdot|\mathscr{D})) \leqslant D_\alpha(\mu_{\boldsymbol{\lambda}_n,\Theta_n}k \,||\, p(\cdot|\mathscr{D}))$$

- Core step : simplify the problem by writing conditions enabling separate (and simultaneous!) updates for $\boldsymbol{\lambda}_{n+1}$ and Θ_{n+1}

$$\int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{\lambda_{j,n+1}}{\lambda_{j,n}}\right) \nu(\mathrm{d}y) \geqslant 0 \qquad \text{(Weights)}$$

$$\int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{k(\theta_{j,n+1}, y)}{k(\theta_{j,n}, y)}\right) \nu(\mathrm{d}y) \geqslant 0 \qquad \text{(Components)}$$

where $\varphi_{j,n}^{(\alpha)}(y) = k(\theta_{j,n}, y) \left(\frac{\mu_{\boldsymbol{\lambda}_n,\Theta_n}k(y)}{p(y,\mathscr{D})}\right)^{\alpha-1}$

Updating the mixture weights $\boldsymbol{\lambda}_{n+1}$

$$\int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{\lambda_{j,n+1}}{\lambda_{j,n}}\right) \nu(\mathrm{d}y) \geqslant 0 \qquad \text{(Weights)}$$

$\to$ (Weights) holds for $\boldsymbol{\lambda}_{n+1}$ such that

$$\boldsymbol{\lambda}_{n+1} = \mathrm{argmax}_{\boldsymbol{\lambda} \in \mathcal{S}_J} \int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{\lambda_{j}}{\lambda_{j,n}}\right) \nu(\mathrm{d}y)$$

Updating the mixture weights $\boldsymbol{\lambda}_{n+1}$

$$\int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{\lambda_{j,n+1}}{\lambda_{j,n}}\right) \nu(\mathrm{d}y) \geqslant 0 \tag{Weights}$$

$\rightarrow$ (Weights) holds for $\boldsymbol{\lambda}_{n+1}$ such that

$$\boldsymbol{\lambda}_{n+1} = \mathrm{argmax}_{\boldsymbol{\lambda} \in \mathcal{S}_J} \int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{\lambda_j}{\lambda_{j,n}}\right) \nu(\mathrm{d}y)$$

Updating the mixture weights $\boldsymbol{\lambda}_{n+1}$

$$\int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{\lambda_{j,n+1}}{\lambda_{j,n}}\right) \nu(\mathrm{d}y) \geqslant 0 \qquad \text{(Weights)}$$

$\rightarrow$ (Weights) holds for $\boldsymbol{\lambda}_{n+1}$ such that

$$\lambda_{j,n+1} = \frac{\lambda_{j,n} \int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y)}{\sum_{\ell=1}^{J} \lambda_{\ell,n} \int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y)}\ , \quad j = 1 \ldots J$$

Updating the mixture weights $\boldsymbol{\lambda}_{n+1}$

$$\int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{\lambda_{j,n+1}}{\lambda_{j,n}}\right) \nu(\mathrm{d}y) \geqslant 0 \qquad \text{(Weights)}$$

$\rightarrow$ (Weights) holds for $\boldsymbol{\lambda}_{n+1}$ such that

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^{J} \lambda_{\ell,n}\left[\int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}, \quad j = 1 \ldots J$$

where $\eta_n \in (0,1]$ and κ_n is such that $(\alpha-1)\kappa_n \geqslant 0$

Updating the mixture components parameters Θ_{n+1}

$$\int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{k(\theta_{j,n+1},y)}{k(\theta_{j,n},y)}\right) \nu(\mathrm{d}y) \geqslant 0 \qquad \text{(Components)}$$

- Maximisation approach : for all $j = 1 \dots J$,

$$\theta_{j,n+1} = \mathrm{argmax}_{\theta \in \mathsf{T}} \int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{k(\theta,y)}{k(\theta_{j,n},y)}\right) \nu(\mathrm{d}y)$$

- Gradient-based approach : for all $j = 1 \dots J$, $\gamma_{j,n} \in (0,1]$

$$\theta_{j,n+1} = \theta_{j,n} - \frac{\gamma_{j,n}}{\beta_{j,n}} \nabla g_{j,n}(\theta)|_{\theta=\theta_{j,n}}$$

where $g_{j,n}$ is assumed to be $\beta_{j,n}$-smooth on $\mathsf{T} = \mathbb{R}^d$ with

$$g_{j,n}(\theta) = \int_{\mathsf{Y}} \frac{\varphi_{j,n}^{(\alpha)}(y)}{\alpha - 1} \log\left(\frac{k(\theta,y)}{k(\theta_{j,n},y)}\right) \nu(\mathrm{d}y) \,.$$

Updating the mixture components parameters Θ_{n+1}

$$\int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log \left(\frac{k(\theta_{j,n+1}, y)}{k(\theta_{j,n}, y)} \right) \nu(\mathrm{d}y) \geqslant 0 \qquad \text{(Components)}$$

- Maximisation approach : for all $j = 1 \ldots J$,

$$\theta_{j,n+1} = \mathrm{argmax}_{\theta \in \mathsf{T}} \int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \log \left(\frac{k(\theta, y)}{k(\theta_{j,n}, y)} \right) \nu(\mathrm{d}y)$$

- Gradient-based approach : for all $j = 1 \ldots J$, $\gamma_{j,n} \in (0, 1]$

$$\theta_{j,n+1} = \theta_{j,n} - \frac{\gamma_{j,n}}{\beta_{j,n}} \nabla g_{j,n}(\theta)|_{\theta=\theta_{j,n}}$$

where $g_{j,n}$ is assumed to be $\beta_{j,n}$-smooth on $\mathsf{T} = \mathbb{R}^d$ with

$$g_{j,n}(\theta) = \int_{\mathsf{Y}} \frac{\varphi_{j,n}^{(\alpha)}(y)}{\alpha - 1} \log \left(\frac{k(\theta, y)}{k(\theta_{j,n}, y)} \right) \nu(\mathrm{d}y) .$$

Updating the mixture components parameters Θ_{n+1}

$$\int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{k(\theta_{j,n+1}, y)}{k(\theta_{j,n}, y)}\right) \nu(\mathrm{d}y) \geqslant 0 \qquad \text{(Components)}$$

- Maximisation approach : for all $j = 1 \dots J$, $b_{j,n} \geqslant 0$ and

$$\theta_{j,n+1} = \operatorname{argmax}_{\theta \in \mathsf{T}} \int_{\mathsf{Y}} \left[\varphi_{j,n}^{(\alpha)}(y) + b_{j,n} k(\theta_{j,n}, y)\right] \log\left(\frac{k(\theta, y)}{k(\theta_{j,n}, y)}\right) \nu(\mathrm{d}y)$$

- Gradient-based approach : for all $j = 1 \dots J$, $\gamma_{j,n} \in (0, 1]$

$$\theta_{j,n+1} = \theta_{j,n} - \frac{\gamma_{j,n}}{\beta_{j,n}} \nabla g_{j,n}(\theta)|_{\theta=\theta_{j,n}}$$

where $g_{j,n}$ is assumed to be $\beta_{j,n}$-smooth on $\mathsf{T} = \mathbb{R}^d$ with

$$g_{j,n}(\theta) = \int_{\mathsf{Y}} \frac{\varphi_{j,n}^{(\alpha)}(y)}{\alpha - 1} \log\left(\frac{k(\theta, y)}{k(\theta_{j,n}, y)}\right) \nu(\mathrm{d}y) .$$

Updating the mixture components parameters Θ_{n+1}

$$\int_{\mathsf{Y}} \sum_{j=1}^{J} \lambda_{j,n} \varphi_{j,n}^{(\alpha)}(y) \log\left(\frac{k(\theta_{j,n+1}, y)}{k(\theta_{j,n}, y)}\right) \nu(\mathrm{d}y) \geqslant 0 \qquad \text{(Components)}$$

- Maximisation approach : for all $j = 1 \ldots J$, $b_{j,n} \geqslant 0$ and

$$\theta_{j,n+1} = \operatorname{argmax}_{\theta \in \mathsf{T}} \int_{\mathsf{Y}} \left[\varphi_{j,n}^{(\alpha)}(y) + b_{j,n} k(\theta_{j,n}, y)\right] \log\left(\frac{k(\theta, y)}{k(\theta_{j,n}, y)}\right) \nu(\mathrm{d}y)$$

- Gradient-based approach : for all $j = 1 \ldots J$, $\gamma_{j,n} \in (0, 1]$

$$\theta_{j,n+1} = \theta_{j,n} - \frac{\gamma_{j,n}}{\beta_{j,n}} \nabla g_{j,n}(\theta)|_{\theta=\theta_{j,n}}$$

where $g_{j,n}$ is assumed to be $\beta_{j,n}$-smooth on $\mathsf{T} = \mathbb{R}^d$ with

$$g_{j,n}(\theta) = \int_{\mathsf{Y}} \frac{\varphi_{j,n}^{(\alpha)}(y)}{\alpha - 1} \log\left(\frac{k(\theta, y)}{k(\theta_{j,n}, y)}\right) \nu(\mathrm{d}y) .$$

An example : GMMs, $k(\theta_j, y) = \mathcal{N}(y; m_j, \Sigma_j)$

- Maximisation approach with $\theta_j = (m_j, \Sigma_j)$: for all $j = 1 \ldots J$,

$$m_{j,n+1} = (1-\gamma_{j,n})m_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \; y \; \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

$$\Sigma_{j,n+1} = (1-\gamma_{j,n})\tilde{\Sigma}_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)(y - m_{j,n+1})(y - m_{j,n+1})^T \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

where $\tilde{\Sigma}_{j,n} = \Sigma_{j,n} + (m_{j,n+1} - m_{j,n})(m_{j,n+1} - m_{j,n})^T$ and $\gamma_{j,n}$ depends on $b_{j,n}$

Considering all possible values of $b_{j,n}$, we have $\gamma_{j,n} \in (0, 1]$

- Gradient-based approach with $\theta_j = m_j$: for all $j = 1 \ldots J$

$$m_{j,n+1} = (1-\gamma_{j,n})m_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \; y \; \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

where $\gamma_{j,n} \in (0, 1]$, and $\Sigma_j = \sigma_j^2 \boldsymbol{I}_d$ with $\sigma_j > 0$ fixed

(here $g_{j,n}$ is $\beta_{j,n}$-smooth with $\beta_{j,n} = \sigma_j^{-2}(1-\alpha)^{-1} \int \varphi_{j,n}^{(\alpha)} \mathrm{d}\nu$)

An example : GMMs, $k(\theta_j, y) = \mathcal{N}(y; m_j, \Sigma_j)$

- Maximisation approach with $\theta_j = (m_j, \Sigma_j)$: for all $j = 1 \dots J$,

$$m_{j,n+1} = (1-\gamma_{j,n}) m_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\ y\ \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

$$\Sigma_{j,n+1} = (1-\gamma_{j,n}) \tilde{\Sigma}_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)(y - m_{j,n+1})(y - m_{j,n+1})^T \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

where $\tilde{\Sigma}_{j,n} = \Sigma_{j,n} + (m_{j,n+1} - m_{j,n})(m_{j,n+1} - m_{j,n})^T$ and $\gamma_{j,n}$ depends on $b_{j,n}$

Considering all possible values of $b_{j,n}$, we have $\gamma_{j,n} \in (0, 1]$

- Gradient-based approach with $\theta_j = m_j$: for all $j = 1 \dots J$

$$m_{j,n+1} = (1-\gamma_{j,n}) m_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\ y\ \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

where $\gamma_{j,n} \in (0, 1]$, and $\Sigma_j = \sigma_j^2 \boldsymbol{I}_d$ with $\sigma_j > 0$ fixed

(here $g_{j,n}$ is $\beta_{j,n}$-smooth with $\beta_{j,n} = \sigma_j^{-2}(1-\alpha)^{-1} \int \varphi_{j,n}^{(\alpha)} \mathrm{d}\nu$)

An example : GMMs, $k(\theta_j, y) = \mathcal{N}(y; m_j, \Sigma_j)$

- Maximisation approach with $\theta_j = (m_j, \Sigma_j)$: for all $j = 1 \dots J$,

$$m_{j,n+1} = (1-\gamma_{j,n}) m_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \; y \; \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

$$\Sigma_{j,n+1} = (1-\gamma_{j,n}) \tilde{\Sigma}_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) (y - m_{j,n+1})(y - m_{j,n+1})^T \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

where $\tilde{\Sigma}_{j,n} = \Sigma_{j,n} + (m_{j,n+1} - m_{j,n})(m_{j,n+1} - m_{j,n})^T$ and $\gamma_{j,n}$ depends on $b_{j,n}$

Considering all possible values of $b_{j,n}$, we have $\gamma_{j,n} \in (0,1]$

- Gradient-based approach with $\theta_j = m_j$: for all $j = 1 \dots J$

$$m_{j,n+1} = (1-\gamma_{j,n}) m_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \; y \; \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

where $\gamma_{j,n} \in (0,1]$, and $\Sigma_j = \sigma_j^2 \boldsymbol{I}_d$ with $\sigma_j > 0$ fixed

(here $g_{j,n}$ is $\beta_{j,n}$-smooth with $\beta_{j,n} = \sigma_j^{-2} (1-\alpha)^{-1} \int \varphi_{j,n}^{(\alpha)} \mathrm{d}\nu$)

At this stage...

Goal : Given $(\boldsymbol{\lambda}_n, \Theta_n)$ find $(\boldsymbol{\lambda}_{n+1}, \Theta_{n+1})$ such that

$$D_\alpha(\mu_{\boldsymbol{\lambda}_{n+1},\Theta_{n+1}} k \mid\mid p(\cdot|\mathscr{D})) \leqslant D_\alpha(\mu_{\boldsymbol{\lambda}_n,\Theta_n} k \mid\mid p(\cdot|\mathscr{D}))$$

We found:

1. Updates for the mixture weights $\boldsymbol{\lambda}_{n+1}$

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^{J} \lambda_{\ell,n}\left[\int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}, \quad j = 1 \ldots J$$

where $\eta_n \in (0,1]$ and κ_n is such that $(\alpha-1)\kappa_n \geqslant 0$

2. Updates for the mixture components parameters Θ_{n+1}
   - Maximisation approach
   - Gradient-based approach

   that are applicable to GMMs [maximisation approach providing covariance matrices updates].

Question : Related work?

At this stage...

Goal : Given $(\boldsymbol{\lambda}_n, \Theta_n)$ find $(\boldsymbol{\lambda}_{n+1}, \Theta_{n+1})$ such that

$$D_\alpha(\mu_{\boldsymbol{\lambda}_{n+1},\Theta_{n+1}}k \mid\mid p(\cdot|\mathscr{D})) \leqslant D_\alpha(\mu_{\boldsymbol{\lambda}_n,\Theta_n}k \mid\mid p(\cdot|\mathscr{D}))$$

We found:

1. Updates for the mixture weights $\boldsymbol{\lambda}_{n+1}$

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^J \lambda_{\ell,n}\left[\int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}\;, \quad j = 1 \ldots J$$

where $\eta_n \in (0,1]$ and κ_n is such that $(\alpha-1)\kappa_n \geqslant 0$

2. Updates for the mixture components parameters Θ_{n+1}
   - Maximisation approach
   - Gradient-based approach

   that are applicable to GMMs [maximisation approach providing covariance matrices updates].

Question : Related work?

At this stage...

Goal : Given $(\boldsymbol{\lambda}_n, \Theta_n)$ find $(\boldsymbol{\lambda}_{n+1}, \Theta_{n+1})$ such that

$$D_\alpha(\mu_{\boldsymbol{\lambda}_{n+1},\Theta_{n+1}} k \mid\mid p(\cdot|\mathscr{D})) \leqslant D_\alpha(\mu_{\boldsymbol{\lambda}_n,\Theta_n} k \mid\mid p(\cdot|\mathscr{D}))$$

We found:

❶ Updates for the mixture weights $\boldsymbol{\lambda}_{n+1}$

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^{J} \lambda_{\ell,n}\left[\int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}, \quad j = 1 \ldots J$$

where $\eta_n \in (0,1]$ and κ_n is such that $(\alpha-1)\kappa_n \geqslant 0$

❷ Updates for the mixture components parameters Θ_{n+1}

- Maximisation approach
- Gradient-based approach

that are applicable to GMMs [maximisation approach providing covariance matrices updates].

Question : Related work?

At this stage...

Goal : Given $(\boldsymbol{\lambda}_n, \Theta_n)$ find $(\boldsymbol{\lambda}_{n+1}, \Theta_{n+1})$ such that

$$D_\alpha(\mu_{\boldsymbol{\lambda}_{n+1},\Theta_{n+1}} k \mid\mid p(\cdot|\mathscr{D})) \leqslant D_\alpha(\mu_{\boldsymbol{\lambda}_n,\Theta_n} k \mid\mid p(\cdot|\mathscr{D}))$$

We found:

❶ Updates for the mixture weights $\boldsymbol{\lambda}_{n+1}$

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^J \lambda_{\ell,n}\left[\int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}} \ , \quad j = 1 \ldots J$$

where $\eta_n \in (0,1]$ and κ_n is such that $(\alpha-1)\kappa_n \geqslant 0$

❷ Updates for the mixture components parameters Θ_{n+1}

- Maximisation approach
- Gradient-based approach

that are applicable to GMMs [maximisation approach providing covariance matrices updates].

Question : Related work?

At this stage...

Goal : Given $(\boldsymbol{\lambda}_n, \Theta_n)$ find $(\boldsymbol{\lambda}_{n+1}, \Theta_{n+1})$ such that

$$D_\alpha(\mu_{\boldsymbol{\lambda}_{n+1},\Theta_{n+1}} k \,||\, p(\cdot|\mathscr{D})) \leqslant D_\alpha(\mu_{\boldsymbol{\lambda}_n,\Theta_n} k \,||\, p(\cdot|\mathscr{D}))$$

We found:

1. Updates for the mixture weights $\boldsymbol{\lambda}_{n+1}$

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^J \lambda_{\ell,n}\left[\int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}, \quad j = 1 \dots J$$

where $\eta_n \in (0,1]$ and κ_n is such that $(\alpha-1)\kappa_n \geqslant 0$

2. Updates for the mixture components parameters Θ_{n+1}
   - Maximisation approach
   - Gradient-based approach

   that are applicable to GMMs [maximisation approach providing covariance matrices updates].

Question : Related work?

1) GD for (Rényi's) α-divergence minimisation

Rényi divergence variational inference. Y. Li and R. E Turner (2016). NeurIPS

Variational inference via χ-upper bound minimization. A. Dieng, D. Tran, R. Ranganath, J. Paisley, and D. Blei (2017). NeurIPS

$\rightarrow$ Main focus on mixture component parameters optimisation (via GD)

Recall that in the GMM case, we obtained: for all $j = 1 \ldots J$,

$$m_{j,n+1} = (1-\gamma_{j,n})m_{j,n} + \gamma_{j,n}\frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\ y\ \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y)}$$

where $\gamma_{j,n} \in (0,1]$

Core insights :

1. We recover GD steps for mixture components optimisation by Rényi's α-divergence minimisation for a well-chosen $\gamma_{j,n}$.

   $\rightarrow$ Compatibility between GD steps and mixture weights updates

2. Same update on the means for maximisation and gradient-based approaches

   $\rightarrow$ Compatibility between GD steps, mixture weights updates **and covariance matrices updates**

1) GD for (Rényi's) α-divergence minimisation

Rényi divergence variational inference. Y. Li and R. E Turner (2016). NeurIPS

Variational inference via χ-upper bound minimization. A. Dieng, D. Tran, R. Ranganath, J. Paisley, and D. Blei (2017). NeurIPS

$\rightarrow$ Main focus on mixture component parameters optimisation (via GD)

Recall that in the GMM case, we obtained: for all $j = 1 \ldots J$,

$$m_{j,n+1} = (1 - \gamma_{j,n}) m_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \; y \; \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

where $\gamma_{j,n} \in (0, 1]$

Core insights :

1. We recover GD steps for mixture components optimisation by Rényi's α-divergence minimisation for a well-chosen $\gamma_{j,n}$.

   $\rightarrow$ Compatibility between GD steps and mixture weights updates

2. Same update on the means for maximisation and gradient-based approaches

   $\rightarrow$ Compatibility between GD steps, mixture weights updates **and covariance matrices updates**

1) GD for (Rényi's) α-divergence minimisation

Rényi divergence variational inference. Y. Li and R. E Turner (2016). NeurIPS

Variational inference via χ-upper bound minimization. A. Dieng, D. Tran, R. Ranganath, J. Paisley, and D. Blei (2017). NeurIPS

$\rightarrow$ Main focus on mixture component parameters optimisation (via GD)

Recall that in the GMM case, we obtained: for all $j = 1 \ldots J$,

$$m_{j,n+1} = (1-\gamma_{j,n})m_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \; y \; \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

where $\gamma_{j,n} \in (0,1]$

Core insights :

1. We recover GD steps for mixture components optimisation by Rényi's α-divergence minimisation for a well-chosen $\gamma_{j,n}$.

   $\rightarrow$ Compatibility between GD steps and mixture weights updates

2. Same update on the means for maximisation and gradient-based approaches

   $\rightarrow$ Compatibility between GD steps, mixture weights updates **and covariance matrices updates**

1) GD for (Rényi's) α-divergence minimisation

Rényi divergence variational inference. Y. Li and R. E Turner (2016). NeurIPS

Variational inference via χ-upper bound minimization. A. Dieng, D. Tran, R. Ranganath, J. Paisley, and D. Blei (2017). NeurIPS

$\rightarrow$ Main focus on mixture component parameters optimisation (via GD)

Recall that in the GMM case, we obtained: for all $j = 1 \ldots J$,

$$m_{j,n+1} = (1 - \gamma_{j,n}) m_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\ y\ \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

where $\gamma_{j,n} \in (0, 1]$

Core insights :

1. We recover GD steps for mixture components optimisation by Rényi's α-divergence minimisation for a well-chosen $\gamma_{j,n}$.

   $\rightarrow$ Compatibility between GD steps and mixture weights updates

2. Same update on the means for maximisation and gradient-based approaches

   $\rightarrow$ Compatibility between GD steps, mixture weights updates **and covariance matrices updates**

1) GD for (Rényi's) α-divergence minimisation

Rényi divergence variational inference. Y. Li and R. E Turner (2016). NeurIPS

Variational inference via χ-upper bound minimization. A. Dieng, D. Tran, R. Ranganath, J. Paisley, and D. Blei (2017). NeurIPS

$\rightarrow$ Main focus on mixture component parameters optimisation (via GD)

Recall that in the GMM case, we obtained: for all $j = 1 \dots J$,

$$m_{j,n+1} = (1 - \gamma_{j,n}) m_{j,n} + \gamma_{j,n} \frac{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \; y \; \nu(\mathrm{d}y)}{\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y) \nu(\mathrm{d}y)}$$

where $\gamma_{j,n} \in (0, 1]$

Core insights :

1. We recover GD steps for mixture components optimisation by Rényi's α-divergence minimisation for a well-chosen $\gamma_{j,n}$.

   $\rightarrow$ Compatibility between GD steps and mixture weights updates

2. Same update on the means for maximisation and gradient-based approaches

   $\rightarrow$ Compatibility between GD steps, mixture weights updates **and covariance matrices updates**

2) The Power Descent algorithm

Infinite-dimensional gradient-based descent for alpha-divergence minimisation.
K. Daudel, R. Douc and F. Portier. Ann. Statist. 49 (4) 2250 - 2270, August 2021.
https://doi.org/10.1214/20-AOS2035.

→ Main focus on mixture weights optimisation

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^{J} \lambda_{\ell,n}\left[\int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}, \quad j = 1 \ldots J$$

$$\Theta_{n+1} = \Theta_n$$

where $\eta_n \in (0,1]$ and κ_n is such that $(\alpha-1)\kappa_n \geqslant 0$

Core insights :

1. Same update on the mixture weights as the Power Descent
   → Compatibility between mixture weights updates and mixture components parameters updates
2. The mixture weights update is gradient-based, η_n plays the role of a learning rate

2) The Power Descent algorithm

Infinite-dimensional gradient-based descent for alpha-divergence minimisation.
K. Daudel, R. Douc and F. Portier. Ann. Statist. 49 (4) 2250 - 2270, August 2021.
https://doi.org/10.1214/20-AOS2035.

$\rightarrow$ Main focus on mixture weights optimisation

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^{J} \lambda_{\ell,n}\left[\int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}, \quad j = 1 \dots J$$

$$\Theta_{n+1} = \Theta_n$$

where $\eta_n \in (0,1]$ and κ_n is such that $(\alpha-1)\kappa_n \geqslant 0$

Core insights :

1. Same update on the mixture weights as the Power Descent
   $\rightarrow$ Compatibility between mixture weights updates and mixture components parameters updates
2. The mixture weights update is gradient-based, η_n plays the role of a learning rate

2) The Power Descent algorithm

Infinite-dimensional gradient-based descent for alpha-divergence minimisation.
K. Daudel, R. Douc and F. Portier. Ann. Statist. 49 (4) 2250 - 2270, August 2021.
https://doi.org/10.1214/20-AOS2035.

→ Main focus on mixture weights optimisation

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^{J} \lambda_{\ell,n}\left[\int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}\,, \quad j = 1\ldots J$$

$$\Theta_{n+1} = \Theta_n$$

where $\eta_n \in (0,1]$ and κ_n is such that $(\alpha-1)\kappa_n \geqslant 0$

Core insights :

1. Same update on the mixture weights as the Power Descent
   → Compatibility between mixture weights updates and mixture components parameters updates
2. The mixture weights update is gradient-based, η_n plays the role of a learning rate

2) The Power Descent algorithm

Infinite-dimensional gradient-based descent for alpha-divergence minimisation.
K. Daudel, R. Douc and F. Portier. Ann. Statist. 49 (4) 2250 - 2270, August 2021.
https://doi.org/10.1214/20-AOS2035.

$\rightarrow$ Main focus on mixture weights optimisation

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^{J} \lambda_{\ell,n}\left[\int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}\ , \quad j = 1\ldots J$$
$$\Theta_{n+1} = \Theta_n$$

where $\eta_n \in (0,1]$ and κ_n is such that $(\alpha-1)\kappa_n \geqslant 0$

Core insights :

1. Same update on the mixture weights as the Power Descent
   $\rightarrow$ Compatibility between mixture weights updates and mixture components parameters updates
2. The mixture weights update is gradient-based, η_n plays the role of a learning rate

2) The Power Descent algorithm

Infinite-dimensional gradient-based descent for alpha-divergence minimisation.
K. Daudel, R. Douc and F. Portier. Ann. Statist. 49 (4) 2250 - 2270, August 2021.
https://doi.org/10.1214/20-AOS2035.

→ Main focus on mixture weights optimisation

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^{J} \lambda_{\ell,n}\left[\int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}\,, \quad j = 1\ldots J$$

$$\Theta_{n+1} = \Theta_n$$

where $\eta_n \in (0,1]$ and κ_n is such that $(\alpha-1)\kappa_n \geqslant 0$

Core insights :

1. Same update on the mixture weights as the Power Descent

   → Compatibility between mixture weights updates and mixture components parameters updates

2. The mixture weights update is gradient-based, η_n plays the role of a learning rate

2) The Power Descent algorithm

Infinite-dimensional gradient-based descent for alpha-divergence minimisation.
K. Daudel, R. Douc and F. Portier. Ann. Statist. 49 (4) 2250 - 2270, August 2021.
https://doi.org/10.1214/20-AOS2035.

$\rightarrow$ Main focus on mixture weights optimisation

$$\lambda_{j,n+1} = \frac{\lambda_{j,n} \left[\int_{\mathsf{Y}} \varphi_{j,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^J \lambda_{\ell,n} \left[\int_{\mathsf{Y}} \varphi_{\ell,n}^{(\alpha)}(y)\nu(\mathrm{d}y) + (\alpha-1)\kappa_n\right]^{\eta_n}}, \quad j = 1 \ldots J$$
$$\Theta_{n+1} = \Theta_n$$

where $\eta_n \in (0,1]$ and κ_n is such that $(\alpha-1)\kappa_n \geqslant 0$

Core insights :

1. Same update on the mixture weights as the Power Descent
   $\rightarrow$ Compatibility between mixture weights updates and mixture components parameters updates
2. The mixture weights update is gradient-based, η_n plays the role of a learning rate

3) The M-PMC algorithm a.k.a 'Integrated EM'

Adaptive importance sampling in general mixture classes. O. Cappé, R. Douc, A. Guillin, J-M Marin and C. P Robert (2008). Statistics and Computing, 18(4):447–459

We recover this algorithm by setting :
- $\alpha = 0$
- $\eta_n = 1$, $\kappa_n = 0$ in the mixture weights update
- $b_{j,n} = 0$ in the maximisation approach

Core insights :

We have generalised an integrated EM algorithm for mixture models optimisation

1. We extend the systematic decrease property to $\alpha \in [0, 1)$
2. We introduce η_n and κ_n, where η_n acts as a learning rate
3. In the GMM case, we introduce $\gamma_{j,n}$ via $b_{j,n}$, which acts as a learning rate

NB : Why do the learning rate aspects matter? In pratice, Monte Carlo approximations!

3) The M-PMC algorithm a.k.a 'Integrated EM'

Adaptive importance sampling in general mixture classes. O. Cappé, R. Douc, A. Guillin, J-M Marin and C. P Robert (2008). Statistics and Computing, 18(4):447–459

We recover this algorithm by setting :
- $\alpha = 0$
- $\eta_n = 1$, $\kappa_n = 0$ in the mixture weights update
- $b_{j,n} = 0$ in the maximisation approach

Core insights :

We have generalised an integrated EM algorithm for mixture models optimisation

1. We extend the systematic decrease property to $\alpha \in [0, 1)$
2. We introduce η_n and κ_n, where η_n acts as a learning rate
3. In the GMM case, we introduce $\gamma_{j,n}$ via $b_{j,n}$, which acts as a learning rate

NB : Why do the learning rate aspects matter? In pratice, Monte Carlo approximations!

3) The M-PMC algorithm a.k.a 'Integrated EM'

Adaptive importance sampling in general mixture classes. O. Cappé, R. Douc, A. Guillin, J-M Marin and C. P Robert (2008). Statistics and Computing, 18(4):447–459

We recover this algorithm by setting :
- $\alpha = 0$
- $\eta_n = 1$, $\kappa_n = 0$ in the mixture weights update
- $b_{j,n} = 0$ in the maximisation approach

Core insights :

We have generalised an integrated EM algorithm for mixture models optimisation

1. We extend the systematic decrease property to $\alpha \in [0, 1)$
2. We introduce η_n and κ_n, where η_n acts as a learning rate
3. In the GMM case, we introduce $\gamma_{j,n}$ via $b_{j,n}$, which acts as a learning rate

NB : Why do the learning rate aspects matter? In pratice, Monte Carlo approximations!

3) The M-PMC algorithm a.k.a 'Integrated EM'

Adaptive importance sampling in general mixture classes. O. Cappé, R. Douc, A. Guillin, J-M Marin and C. P Robert (2008). Statistics and Computing, 18(4):447–459

We recover this algorithm by setting :
- $\alpha = 0$
- $\eta_n = 1$, $\kappa_n = 0$ in the mixture weights update
- $b_{j,n} = 0$ in the maximisation approach

Core insights :

We have generalised an integrated EM algorithm for mixture models optimisation

1. We extend the systematic decrease property to $\alpha \in [0, 1)$
2. We introduce η_n and κ_n, where η_n acts as a learning rate
3. In the GMM case, we introduce $\gamma_{j,n}$ via $b_{j,n}$, which acts as a learning rate

NB : Why do the learning rate aspects matter? In pratice, Monte Carlo approximations!

3) The M-PMC algorithm a.k.a 'Integrated EM'

Adaptive importance sampling in general mixture classes. O. Cappé, R. Douc, A. Guillin, J-M Marin and C. P Robert (2008). Statistics and Computing, 18(4):447–459

We recover this algorithm by setting :
- $\alpha = 0$
- $\eta_n = 1$, $\kappa_n = 0$ in the mixture weights update
- $b_{j,n} = 0$ in the maximisation approach

Core insights :

We have generalised an integrated EM algorithm for mixture models optimisation

1. We extend the systematic decrease property to $\alpha \in [0, 1)$
2. We introduce η_n and κ_n, where η_n acts as a learning rate
3. In the GMM case, we introduce $\gamma_{j,n}$ via $b_{j,n}$, which acts as a learning rate

NB : Why do the learning rate aspects matter? In pratice, Monte Carlo approximations!

3) The M-PMC algorithm a.k.a 'Integrated EM'

Adaptive importance sampling in general mixture classes. O. Cappé, R. Douc, A. Guillin, J-M Marin and C. P Robert (2008). Statistics and Computing, 18(4):447–459

We recover this algorithm by setting :
- $\alpha = 0$
- $\eta_n = 1$, $\kappa_n = 0$ in the mixture weights update
- $b_{j,n} = 0$ in the maximisation approach

Core insights :

We have generalised an integrated EM algorithm for mixture models optimisation

1. We extend the systematic decrease property to $\alpha \in [0, 1)$
2. We introduce η_n and κ_n, where η_n acts as a learning rate
3. In the GMM case, we introduce $\gamma_{j,n}$ via $b_{j,n}$, which acts as a learning rate

NB : Why do the learning rate aspects matter? In pratice, Monte Carlo approximations!

3) The M-PMC algorithm a.k.a 'Integrated EM'

Adaptive importance sampling in general mixture classes. O. Cappé, R. Douc, A. Guillin, J-M Marin and C. P Robert (2008). Statistics and Computing, 18(4):447–459

We recover this algorithm by setting :
- $\alpha = 0$
- $\eta_n = 1$, $\kappa_n = 0$ in the mixture weights update
- $b_{j,n} = 0$ in the maximisation approach

Core insights :

We have generalised an integrated EM algorithm for mixture models optimisation

1. We extend the systematic decrease property to $\alpha \in [0, 1)$
2. We introduce η_n and κ_n, where η_n acts as a learning rate
3. In the GMM case, we introduce $\gamma_{j,n}$ via $b_{j,n}$, which acts as a learning rate

NB : Why do the learning rate aspects matter? In pratice, Monte Carlo approximations!

3) The M-PMC algorithm a.k.a 'Integrated EM'

Adaptive importance sampling in general mixture classes. O. Cappé, R. Douc, A. Guillin, J-M Marin and C. P Robert (2008). Statistics and Computing, 18(4):447–459

We recover this algorithm by setting :
- $\alpha = 0$
- $\eta_n = 1$, $\kappa_n = 0$ in the mixture weights update
- $b_{j,n} = 0$ in the maximisation approach

Core insights :

We have generalised an integrated EM algorithm for mixture models optimisation

1. We extend the systematic decrease property to $\alpha \in [0, 1)$
2. We introduce η_n and κ_n, where η_n acts as a learning rate
3. In the GMM case, we introduce $\gamma_{j,n}$ via $b_{j,n}$, which acts as a learning rate

NB : Why do the learning rate aspects matter? In pratice, Monte Carlo approximations!

Monte Carlo approximations

Algorithm 1: Gaussian Mixture Models optimisation

At iteration n,

1. Draw independently M samples $(Y_{m,n})_{1\leqslant m\leqslant M}$ from the proposal q_n.
2. For all $j = 1 \dots J$, set:

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\sum_{m=1}^{M}\hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n}) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^{J}\lambda_{\ell,n}\left[\sum_{m=1}^{M}\hat{\varphi}_{\ell,n}^{(\alpha)}(Y_{m,n}) + (\alpha-1)\kappa_n\right]^{\eta_n}}$$

$$(MG) \quad m_{j,n+1} = (1-\gamma_n)\, m_{j,n} + \gamma_n \frac{\sum_{m=1}^{M}\hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n})\cdot Y_{m,n}}{\sum_{m=1}^{M}\hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n})}$$

$$(RGD) \quad m_{j,n+1} = m_{j,n} + \gamma_n \frac{\lambda_{j,n}\sum_{m=1}^{M}\hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n})\cdot(Y_{m,n}-\theta_{j,n})}{\sum_{j=1}^{J}\sum_{m=1}^{M}\lambda_{j,n}\hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n})}$$

→ Here $\hat{\varphi}_{j,n}^{(\alpha)}(y) = \frac{\varphi_{j,n}^{(\alpha)}(y)}{q_n(y)}$, $\gamma_{j,n} := \gamma_n \in (0,1]$ (simultaneity matters!)

→ RGD : updates derived from GD steps w.r.t. Θ applied to Rényi's α-divergence

→ 2 possible samplers : $q_n = \mu_{\boldsymbol{\lambda}_n,\Theta_n}$ (IS-n) and $q_n = J^{-1}\sum_{j=1}^{J}k(\theta_{j,n},\cdot)$ (IS-unif).

Monte Carlo approximations

Algorithm 1: Gaussian Mixture Models optimisation

At iteration n,

1. Draw independently M samples $(Y_{m,n})_{1 \leqslant m \leqslant M}$ from the proposal q_n.
2. For all $j = 1 \dots J$, set:

$$\lambda_{j,n+1} = \frac{\lambda_{j,n}\left[\sum_{m=1}^M \hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n}) + (\alpha-1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^J \lambda_{\ell,n}\left[\sum_{m=1}^M \hat{\varphi}_{\ell,n}^{(\alpha)}(Y_{m,n}) + (\alpha-1)\kappa_n\right]^{\eta_n}}$$

$$(MG) \quad m_{j,n+1} = (1-\gamma_n)\, m_{j,n} + \gamma_n \frac{\sum_{m=1}^M \hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n}) \cdot Y_{m,n}}{\sum_{m=1}^M \hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n})}$$

$$(RGD) \quad m_{j,n+1} = m_{j,n} + \gamma_n \frac{\lambda_{j,n} \sum_{m=1}^M \hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n}) \cdot (Y_{m,n} - \theta_{j,n})}{\sum_{j=1}^J \sum_{m=1}^M \lambda_{j,n} \hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n})}$$

→ Here $\hat{\varphi}_{j,n}^{(\alpha)}(y) = \frac{\varphi_{j,n}^{(\alpha)}(y)}{q_n(y)}$, $\gamma_{j,n} := \gamma_n \in (0,1]$ (simultaneity matters!)

→ RGD : updates derived from GD steps w.r.t. Θ applied to Rényi's α-divergence

→ 2 possible samplers : $q_n = \mu_{\boldsymbol{\lambda}_n, \Theta_n}$ (IS-n) and $q_n = J^{-1} \sum_{j=1}^J k(\theta_{j,n}, \cdot)$ (IS-unif).

Monte Carlo approximations

Algorithm 1: Gaussian Mixture Models optimisation

At iteration n,

1. Draw independently M samples $(Y_{m,n})_{1 \leqslant m \leqslant M}$ from the proposal q_n.
2. For all $j = 1 \dots J$, set:

$$\lambda_{j,n+1} = \frac{\lambda_{j,n} \left[\sum_{m=1}^M \hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n}) + (\alpha - 1)\kappa_n\right]^{\eta_n}}{\sum_{\ell=1}^J \lambda_{\ell,n} \left[\sum_{m=1}^M \hat{\varphi}_{\ell,n}^{(\alpha)}(Y_{m,n}) + (\alpha - 1)\kappa_n\right]^{\eta_n}}$$

$$(MG) \quad m_{j,n+1} = (1 - \gamma_n)\, m_{j,n} + \gamma_n \frac{\sum_{m=1}^M \hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n}) \cdot Y_{m,n}}{\sum_{m=1}^M \hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n})}$$

$$(RGD) \quad m_{j,n+1} = m_{j,n} + \gamma_n \frac{\lambda_{j,n} \sum_{m=1}^M \hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n}) \cdot (Y_{m,n} - \theta_{j,n})}{\sum_{j=1}^J \sum_{m=1}^M \lambda_{j,n} \hat{\varphi}_{j,n}^{(\alpha)}(Y_{m,n})}$$

→ Here $\hat{\varphi}_{j,n}^{(\alpha)}(y) = \frac{\varphi_{j,n}^{(\alpha)}(y)}{q_n(y)}$, $\gamma_{j,n} := \gamma_n \in (0, 1]$ (simultaneity matters!)

→ RGD : updates derived from GD steps w.r.t. Θ applied to Rényi's α-divergence

→ 2 possible samplers : $q_n = \mu_{\boldsymbol{\lambda}_n, \Theta_n}$ (IS-n) and $q_n = J^{-1} \sum_{j=1}^J k(\theta_{j,n}, \cdot)$ (IS-unif).

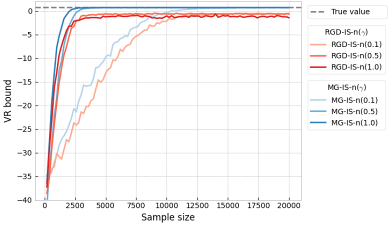
Comparing RGD to MG (fixed $\boldsymbol{\lambda}$)

$$\text{Target}: \quad p(y) = 2 \times \left[0.5\mathcal{N}(\boldsymbol{y}; -2\boldsymbol{u_d}, \boldsymbol{I_d}) + 0.5\mathcal{N}(\boldsymbol{y}; 2\boldsymbol{u_d}, \boldsymbol{I_d})\right]$$

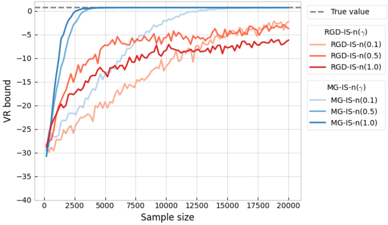
- MC estimate of the VR Bound averaged over 30 trials for RGD and MG.

[Here, $\alpha = 0.2$, $d = 16$, $M = 200$, $\kappa_n = 0$, $\eta_n = 0$. and $q_n = \mu_n k$.]

$J = 10$



$J = 50$



- LogMSE averaged over 30 trials for RGD and MG.

	$J = 10$			$J = 50$		
	$\gamma = 0.1$	$\gamma = 0.5$	$\gamma = 1.0$	$\gamma = 0.1$	$\gamma = 0.5$	$\gamma = 1.0$
RGD-IS-n(γ)	-0.081	-0.076	-0.218	-1.640	-1.673	-1.560
MG-IS-n(γ)	**-3.702**	**-1.875**	**-2.711**	**-2.760**	**-2.771**	**-2.788**

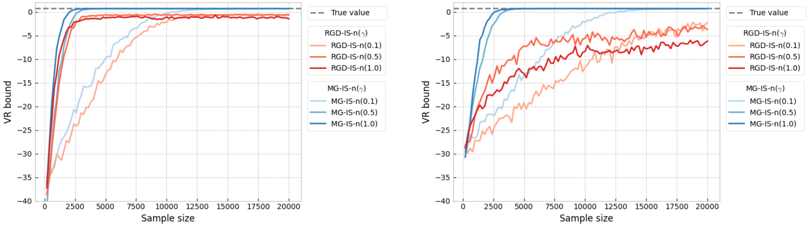
Comparing RGD to MG (fixed $\boldsymbol{\lambda}$)

Target : $p(y) = 2 \times [0.5\mathcal{N}(\boldsymbol{y}; -2\boldsymbol{u_d}, \boldsymbol{I_d}) + 0.5\mathcal{N}(\boldsymbol{y}; 2\boldsymbol{u_d}, \boldsymbol{I_d})]$

- MC estimate of the VR Bound averaged over 30 trials for RGD and MG.

[Here, $\alpha = 0.2$, $d = 16$, $M = 200$, $\kappa_n = 0$, $\eta_n = 0$. and $q_n = \mu_n k$.]

$J = 10$ $\quad$ $J = 50$



- LogMSE averaged over 30 trials for RGD and MG.

	$J = 10$			$J = 50$		
	$\gamma = 0.1$	$\gamma = 0.5$	$\gamma = 1.0$	$\gamma = 0.1$	$\gamma = 0.5$	$\gamma = 1.0$
RGD-IS-n(γ)	-0.081	-0.076	-0.218	-1.640	-1.673	-1.560
MG-IS-n(γ)	**-3.702**	**-1.875**	**-2.711**	**-2.760**	**-2.771**	**-2.788**

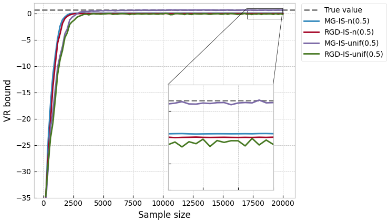
Comparing RGD to MG (varying $\boldsymbol{\lambda}$)

$$\text{Target}: \quad p(y) = 2 \times [0.5\mathcal{N}(\boldsymbol{y}; -2\boldsymbol{u_d}, \boldsymbol{I_d}) + 0.5\mathcal{N}(\boldsymbol{y}; 2\boldsymbol{u_d}, \boldsymbol{I_d})]$$

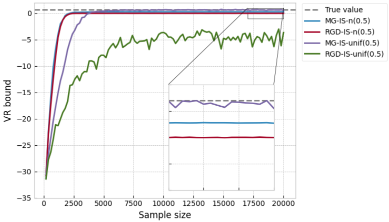
- MC estimate of the VR Bound averaged over 30 trials for RGD and MG.

[Here, $\alpha = 0.2$, $d = 16$, $M = 200$, $\eta = 0.1$, $\kappa_n = 0$.]

$J = 10$



$J = 50$



- LogMSE averaged over 30 trials for RGD and MG.

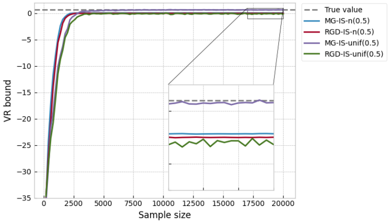
	$J=10$			$J=50$		
	$\gamma=0.1$	$\gamma=0.5$	$\gamma=1.0$	$\gamma=0.1$	$\gamma=0.5$	$\gamma=1.0$
RGD-IS-n(γ)	0.372	0.510	0.384	-0.616	-0.713	-0.778
MG-IS-n(γ)	1.104	1.074	0.387	1.135	-0.077	-0.060
RGD-IS-unif(γ)	0.359	0.469	0.458	-0.688	-0.670	-0.583
MG-IS-unif(γ)	**-0.200**	**-0.229**	**-0.515**	**-1.500**	**-1.462**	**-1.246**

Comparing RGD to MG (varying $\boldsymbol{\lambda}$)

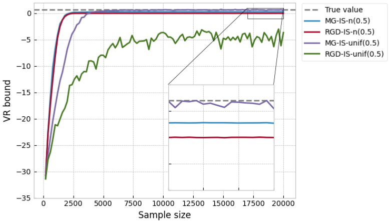
Target : $\quad p(y) = 2 \times [0.5\mathcal{N}(\boldsymbol{y}; -2\boldsymbol{u_d}, \boldsymbol{I_d}) + 0.5\mathcal{N}(\boldsymbol{y}; 2\boldsymbol{u_d}, \boldsymbol{I_d})]$

- MC estimate of the VR Bound averaged over 30 trials for RGD and MG. [Here, $\alpha = 0.2$, $d = 16$, $M = 200$, $\eta = 0.1$, $\kappa_n = 0$.]

$J = 10$



$J = 50$



- LogMSE averaged over 30 trials for RGD and MG.

	$J=10$			$J=50$		
	$\gamma=0.1$	$\gamma=0.5$	$\gamma=1.0$	$\gamma=0.1$	$\gamma=0.5$	$\gamma=1.0$
RGD-IS-n(γ)	0.372	0.510	0.384	-0.616	-0.713	-0.778
MG-IS-n(γ)	1.104	1.074	0.387	1.135	-0.077	-0.060
RGD-IS-unif(γ)	0.359	0.469	0.458	-0.688	-0.670	-0.583
MG-IS-unif(γ)	**-0.200**	**-0.229**	**-0.515**	**-1.500**	**-1.462**	**-1.246**

Comparing RGD to MG (varying $\boldsymbol{\lambda}$) - 2

Target : $p(y) = 2 \times [0.5\mathcal{N}(\boldsymbol{y}; -2\boldsymbol{u_d}, \boldsymbol{I_d}) + 0.5\mathcal{N}(\boldsymbol{y}; 2\boldsymbol{u_d}, \boldsymbol{I_d})]$

• LogMSE averaged over 30 trials for RGD and MG.
[Here, $\alpha = 0.2$, $d = 16$, $M = 200$, $\gamma = 0.5$, $\kappa_n = 0$.]

	$J = 10$			$J = 50$		
	$\eta = 0.05$	$\eta = 0.1$	$\eta = 0.5$	$\eta = 0.05$	$\eta = 0.1$	$\eta = 0.5$
RGD-IS-n(γ)	0.045	0.510	1.299	-1.355	-0.713	0.924
MG-IS-n(γ)	0.087	1.074	1.343	-1.205	-0.077	1.329
RGD-IS-unif(γ)	-0.018	0.469	1.328	-1.385	-0.670	0.928
MG-IS-unif(γ)	**-1.244**	**-0.229**	**1.100**	**-2.524**	**-1.462**	**0.309**

Comparing RGD to MG (varying $\boldsymbol{\lambda}$) - 2

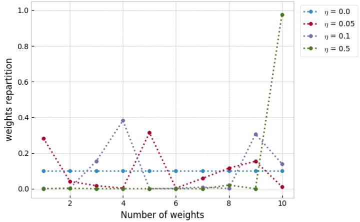
Target : $p(y) = 2 \times [0.5\mathcal{N}(\boldsymbol{y}; -2\boldsymbol{u_d}, \boldsymbol{I_d}) + 0.5\mathcal{N}(\boldsymbol{y}; 2\boldsymbol{u_d}, \boldsymbol{I_d})]$

- LogMSE averaged over 30 trials for RGD and MG.

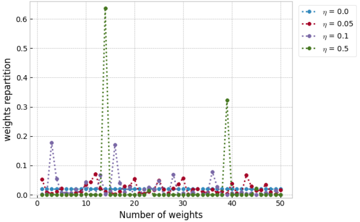
[Here, $\alpha = 0.2$, $d = 16$, $M = 200$, $\gamma = 0.5$, $\kappa_n = 0$.]

	$J = 10$			$J = 50$		
	$\eta = 0.05$	$\eta = 0.1$	$\eta = 0.5$	$\eta = 0.05$	$\eta = 0.1$	$\eta = 0.5$
RGD-IS-n(γ)	0.045	0.510	1.299	-1.355	-0.713	0.924
MG-IS-n(γ)	0.087	1.074	1.343	-1.205	-0.077	1.329
RGD-IS-unif(γ)	-0.018	0.469	1.328	-1.385	-0.670	0.928
MG-IS-unif(γ)	**-1.244**	**-0.229**	**1.100**	**-2.524**	**-1.462**	**0.309**

$J = 10$



$J = 50$



Conclusion

Novel framework for monotonic alpha-divergence minimisation

- applicable to mixture models optimisation with theoretical guarantees
- mixture weights and mixture components parameters can be updated simultaneously
- links with gradient-based approaches and with an Integrated EM algorithm
- Encouraging empirical benefits of our general framework

Some perspectives

- Additional convergence results
- Hyperparameters tuning...

Conclusion

Novel framework for monotonic alpha-divergence minimisation

- applicable to mixture models optimisation with theoretical guarantees
- mixture weights and mixture components parameters can be updated simultaneously
- links with gradient-based approaches and with an Integrated EM algorithm
- Encouraging empirical benefits of our general framework

Some perspectives

- Additional convergence results
- Hyperparameters tuning...

Conclusion

Novel framework for monotonic alpha-divergence minimisation

- applicable to mixture models optimisation with theoretical guarantees
- mixture weights and mixture components parameters can be updated simultaneously
- links with gradient-based approaches and with an Integrated EM algorithm
- Encouraging empirical benefits of our general framework

Some perspectives

- Additional convergence results
- Hyperparameters tuning...

Thank you for your attention!