INSEA

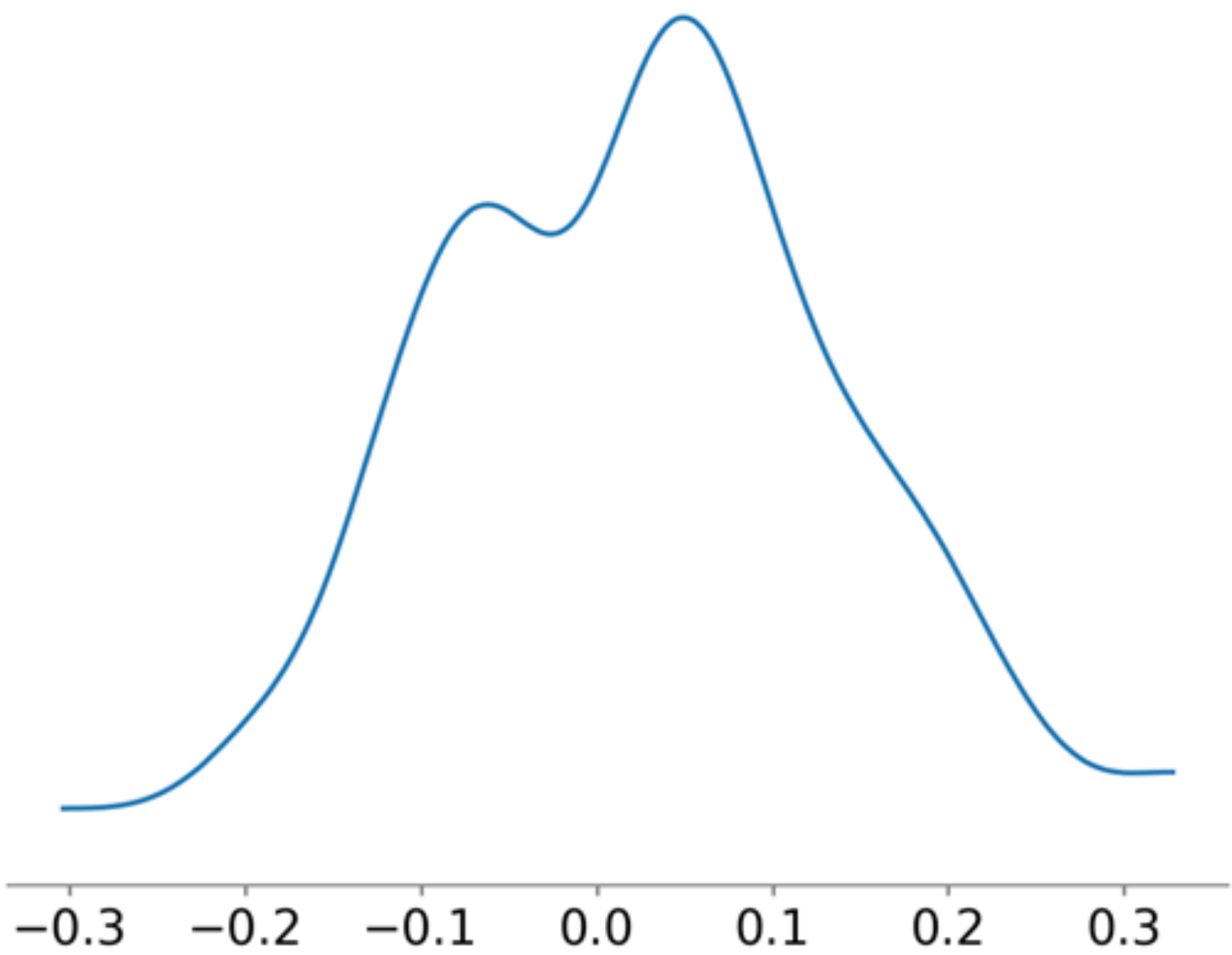
69

MCMC diagnostics in 1D

HDI: High density intervals

High density interval (HDI) ou *Credible Interval* (CI):
équivalent de l'intervalle de confiance en statistiques bayésiennes

HDI de niveau $1-\alpha$ = Le plus petit intervalle I telle que $\mathbb{P}_f(X \in I) \geq 1-\alpha$


−0.3
−0.2
−0.1
0.0
0.1
0.2
0.3

Comment définir un intervalle à haute densité de niveau 90% ?

Un intervalle I tel que $\mathbb{P}_f(X \in I) \geq 0.9$

$$\int_I f \geq 0.9$$

On peut en trouver plusieurs !

Lequel faut-il choisir ?

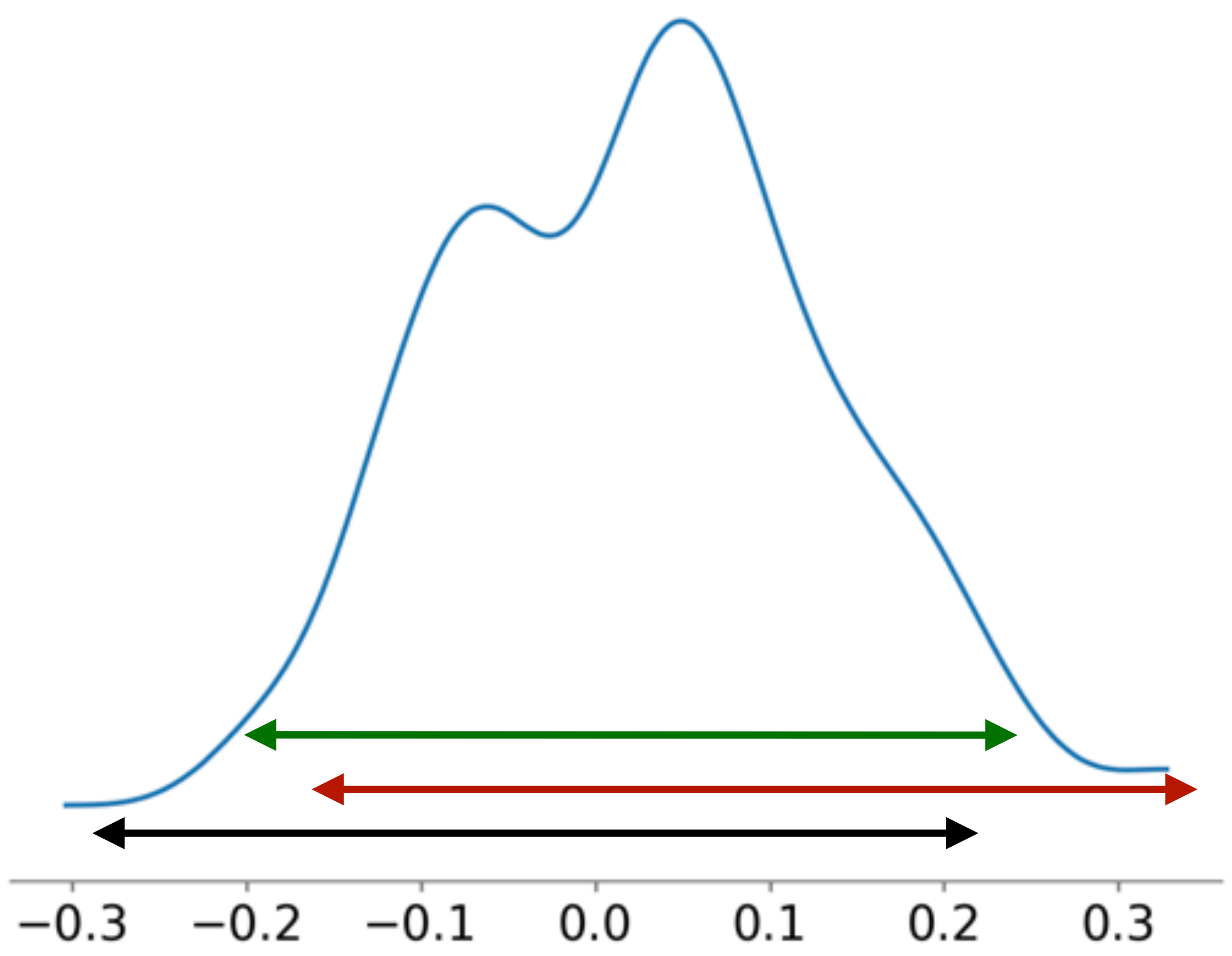
Comment peut-on les comparer aux intervalles de confiance d'un point de vue statistique ?

High density interval (HDI) ou *Credible Interval* (CI):
équivalent de l'intervalle de confiance en statistiques bayésiennes

Comment définir un intervalle à haute densité de niveau 90% ?

Un intervalle I tel que $\mathbb{P}_f(X \in I) \geq 0.9$

$$\int_I f \geq 0.9$$



On peut en trouver plusieurs !

Lequel faut-il choisir ?

$$\text{HDI de niveau } 1-\alpha = \text{Le plus petit intervalle } I \text{ telle que } \mathbb{P}_f(X \in I) \geq 1-\alpha$$

Comment peut-on les comparer aux intervalles de confiance d'un point de vue statistique ?

1. Pourquoi Monte-Carlo ? (Exemple de modèle hiérarchique)
2. Introduction à la méthode Monte-Carlo (historique, PRNG)
3. Algorithmes de simulation i.i.d (PRNG, transformation, rejet)
4. Méthodes MCMC (Gibbs, Metropolis)
5. Diagonstics de convergence MCMC
6. Méthodes MCMC avancées (Langevin, HMC, NUTS)

INSEA

Application (MCMC Diagnostics)

Soit $Z|\mu, \sigma^2 \sim \mathcal{N}(\mu, \sigma^2)$. μ et σ^2 ont des densités a priori indépendantes f_μ et f_{σ^2}.

1. Dessiner le graphe probabiliste du modèle.
2. Déterminez la loi a posteriori jointe $(\mu, \sigma^2)|Z$.
3. Déterminez les lois conditionnelles $\mu|\sigma^2, Z$ et $\sigma^2|\mu, Z$.
4. Quelles lois a priori f_μ et f_{σ^2} devrait-on prendre pour avoir des lois conditionnelles usuelles ?

Simuler 4 chaînes en numpy (Gibbs) et faîtes le diagnostique avec Arviz:

```
import arviz as az
chains_data_1d = .... # np array de taille (n_chains, n_samples)
i_data = az.convert_to_inference_data(dict(var_name=chains_data_1d)) # on construit l'objet inf_data nécessaire pour arviz

az.plot_trace(i_data) # visualiser les chaines / densités
az.plot_autocorr(i_data) # visualiser auto_corr
print(az.summary(kind="diagnostics")) # afficher les métriques ESS, R

az.plot_posterior(i_data) # on visualise la densité a posteriori
```