Universidade do Minho

Experimental quantum characterization

– Bayesian inference and numerical integration

Alexandra Ramôa Alves

January 2022

Presentation outline

1. **Context**
   - Problem statement
   - Bayesian inference
   - A simple example
2. **Open problems**
   - Experimental design (adaptivity)
   - Integration (Monte Carlo)
3. **Characterization of a quantum device**

What to characterize?

We consider the characterization of a **time-dependent function** describing a system's dynamics (e.g. wavefunction coefficient norms, or elements of a density operator)

$$\varphi(t) = a_0(t)|0\rangle + a_1(t)|1\rangle$$

$$P(0|t) = |\langle 0|\varphi(t)\rangle|^2 = |a_0(t)|^2 = ?$$

Characterization is useful for certification, tuning sensing, and predictions.

(the generalization to state or process tomography, phase estimation, etc. is straightforward)

How to characterize?

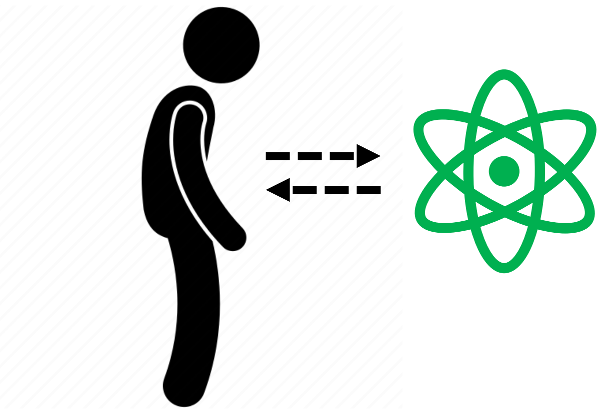
$$P(0|t) = ?$$

We derive empirical knowledge via observation,

i.e. by **measuring the system**.

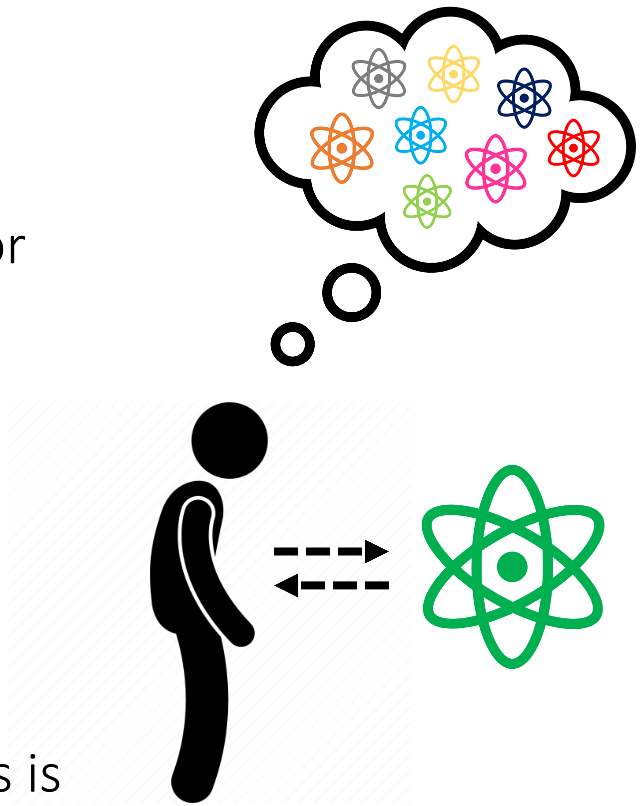
The outcome of the measurement(s) should **depend on the entity to be characterized**. In our case, **this dependence is probabilistic**.

This adds a layer of complexity to metrology – one which can be dealt with using **statistical inference**



How to characterize?

$$P(0|t) = ?$$



Bayesian statistics provides a flexible paradigm for statistical inference.

This strategy ultimately relies on **comparing the behaviour** of the black-box system with that of **potential replicas**.

Having chosen these speculative replicas, the onus is on the **experimental data collection** process, which settles the predictive power

How to characterize?

$$P(0|t) = ?$$

We can exploit some structured knowledge:

- Parametrized **model** for the probability (can be exploratory)
- **Pre-existing information** about the parameters (can be insubstantial)

Along with some other resources:

- Ability to **measure** the system
- Ability to **simulate** the system (given the model)
- Classical processing

How to characterize?

$$P(0|t) = ?$$

We can exploit some structured knowledge:

- Parametrized **model** for the probability $- P(0|\theta; t) = f(\theta, t)$
- **Pre-existing information** about the parameters (can be insubstantial)

Along with some other resources:

- Ability to **measure** the system
- Ability to **simulate** the system (given the model)
- Classical processing

How to characterize?

$$P(0|t) = ?$$

We can exploit some structured knowledge:

- Parametrized **model** for the probability $- P(0|\theta; t) = f(\theta, t)$
- **Pre-existing information** about the parameters $- \ \theta \in \Theta$

Along with some other resources:

- Ability to **measure** the system
- Ability to **simulate** the system (given the model)
- Classical processing

How to characterize?

$$P(0|t) = ?$$

We can exploit some structured knowledge:

- Parametrized **model** for the probability $- P(0|\theta;t) = f(\theta,t)$
- **Pre-existing information** about the parameters $- \ \theta \in \Theta$

Along with some other resources:

- Ability to **measure** the system – **for some** t
- Ability to **simulate** the system (given the model)
- Classical processing

How to characterize?

$$P(0|t) = ?$$

We can exploit some structured knowledge:

- Parametrized **model** for the probability $- P(0|\theta; t) = f(\theta, t)$
- **Pre-existing information** about the parameters $- \ \theta \in \Theta$

Along with some other resources:

- Ability to **measure** the system – **for some** t
- Ability to **simulate** the system – **for some** $\theta, \ t$
- Classical processing

How to characterize?

$$P(0|t) = ?$$

We can exploit some structured knowledge:

- Parametrized **model** for the probability $-P(0|\theta; t) = f(\theta, t)$
- **Pre-existing information** about the parameters $-\ \theta \in \Theta$

Along with some other resources:

- Ability to **measure** the system – **for some** t
- Ability to **simulate** the system – **for some** $\theta,\ t$
- Classical processing – **elementary operations, eventually optimization**

How to characterize?

$$P(0|t) = ? \longrightarrow \boxed{\theta = ?}$$

We can exploit some structured knowledge:

- Parametrized **model** for the probability $- P(0|\theta;t) = f(\theta, t)$
- **Pre-existing information** about the parameters $- \ \theta \in \Theta$

Along with some other resources:

- Ability to **measure** the system – for some t
- Ability to **simulate** the system – for some $\theta, \ t$
- Classical processing – elementary operations, eventually optimization

How to characterize?

$$\theta = ?$$

Bayesian **probabilities** represent **degrees of belief**. Thus, in Bayesian inference we attribute a *probability* to parameter values, which are seen as random variables:

$$P(\theta)$$

This randomness can be ascribed to our own imperfect knowledge: the object of Bayesian statistics is our reality, rather than an idealized version of it.

To **assess the merit of a parameter** instance, we will assign it a degree of belief **according to our observations**

How to characterize?

$$\theta = ?$$

- We start with some **prior probability** distribution over the parameter θ.

$$P(\theta)$$

How to characterize?

$$\theta = ?$$

- We start with some **prior probability** distribution over the parameter θ.
- We then collect an experimental datum D, and **re-weight the prior probability depending on the likelihood** $P(D|\theta)$ that it would have generated D.

$$P(D|\theta)P(\theta)$$

How to characterize?

$$\theta = ?$$

- We start with some **prior probability** distribution over the parameter θ.
- We then collect an experimental datum D, and **re-weight the prior probability depending on the likelihood** $P(D|\theta)$ that it would have generated D.
- Apart from a normalizing constant, we obtain the **posterior probability**, representing our updated knowledge (after observing D).

$$P(\theta|D) \propto P(D|\theta)P(\theta)$$

How to characterize?

$$\theta = ?$$

- We start with some **prior probability** distribution over the parameter θ.
- We then collect an experimental datum D, and **re-weight the prior probability depending on the likelihood** $P(D|\theta)$ that it would have generated D.
- Apart from a normalizing constant, we obtain the **posterior probability**, representing our updated knowledge (after observing D).

Each datum should bring our knowledge slightly closer to reality

$$P(\theta|D) \propto P(D|\theta)P(\theta)$$

How to characterize?

$$\theta = ?$$

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)}$$

Bayes' rule

- $P(\theta|D)$: the **posterior** probability *(our updated knowledge)*
- $P(D|\theta)$: the **likelihood** *(a generative model for the data)*
- $P(\theta)$: the **prior** probability *(can be flat over Θ)*
- $P(D)$: the **marginal** probability *(for a fixed model, this is a normalizing constant)*

The Bayes' rule assesses the **relative merit** of any given **parameter instance**, taken as **the probability of its having generated the observed system behaviour** (up to previous knowledge)

How to characterize?

$$\theta = ?$$

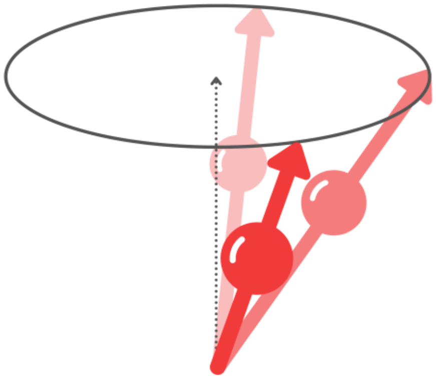
$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)}$$

Bayes' rule

Bayesian inference refers to the systematic use of Bayes' rule to **learn system properties using empirical evidence**. In the quantum case:

- The data $D \in \{0,1\}$ are binary **experimental outcomes**
- The likelihood $P(D|\theta)$ can be calculated using analog **quantum simulation**
- *Experimental controls* may offer **additional degrees of freedom**

A simple example: spin precession



Learn the model Hamiltonian:

$$\hat{H} = \frac{\boldsymbol{\omega}}{2}\hat{\sigma}_z \qquad \boldsymbol{\omega} = ?$$

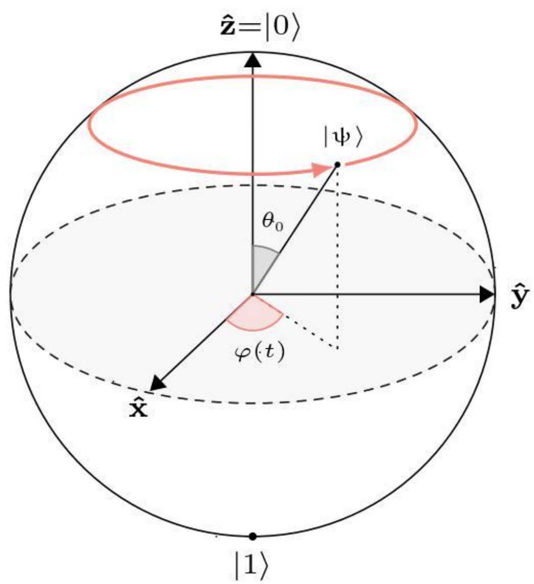
This type of oscillation describes several phenomena of interest - namely **Larmor** precession, **Rabi** flopping, **Ramsey** interferometry, and **phase estimation**.

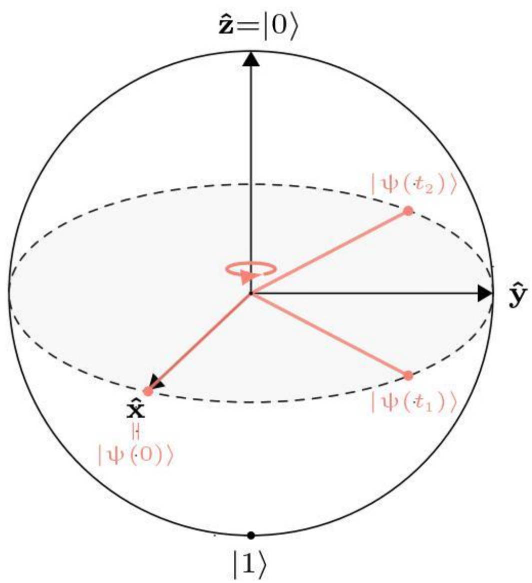
In this context, frequency estimation (learning ω) has applications as important as **magnetic field sensing** and resonant **gate tuning** in superconducting qubits

A simple example: spin precession

In general, this Hamiltonian induces a **precession** about the Bloch sphere's $\hat{\boldsymbol{z}}$ axis.

For simplicity, we consider $|+\rangle$ initialization at *t=0*, and thus oscillations contained in the **equatorial plane**.





To collect the data, we **measure the qubit on the *z* basis** for some chosen **evolution times** t_i (counting from initialization). This evolution time is an experimental control.

A simple example: spin precession

In this case, as in many others, the **likelihood** is physically motivated.

$$P(0|\omega; t) = \cos^2\left({}^{\omega t}/_{2}\right)$$

$$P(1|\omega; t) = 1 - P(0|\omega; t)$$

The **prior** can be chosen to barely influence the results. Generally, we establish a **frequency range**, over which we can lay out a flat prior:

$$P(\omega) = \begin{cases} {}^{1}/_{V_\Omega} & \text{if } \omega \in \Omega \\ 0 & \text{otherwise} \end{cases}$$

All that's left is the marginal, which can be regarded as a normalization constant. Thus, we can evaluate the posterior:

$$P(\omega|D; t) = \frac{P(D|\omega; t)P(\omega)}{P(D; t)}$$

The posterior distribution

$$P(\omega|D;t) = \frac{P(D|\omega;t)P(\omega)}{P(D;t)}$$

The posterior is a **continuous probability distribution** representing our posterior (updated) knowledge on the model parameter(s)**.** From a Bayesian viewpoint, **it holds *all* the available information**.

Conceptually, this solves the problem; but in practice, it creates another one. How do we **extract the information** we want?

The posterior distribution

Taking the single parameter case, the standard **frequentist** approach would take **multiple trials** with **unchanged experimental controls** and **invert the likelihood** to produce a **parameter estimate**.

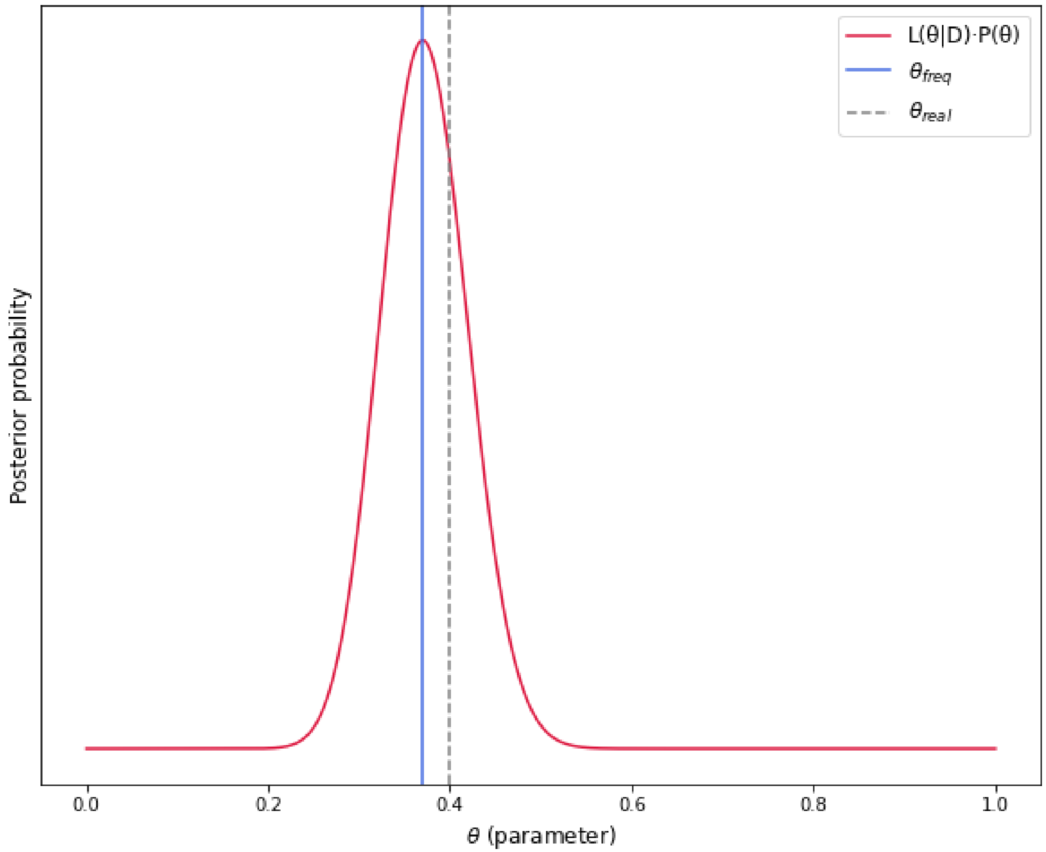
$$P(0|\omega;t) \approx \frac{\#\ 0\ \text{outcomes}}{\#\ \text{trials}} = \cos^2\left({}^{\boldsymbol{\omega} t}\!/_{2}\right) \leftrightarrow \omega = \omega_X$$

By contrast, the Bayesian approach offers *only* a way to compute the probability of parameter estimates of our choice.

$$P(\omega|D;t) = \frac{P(D|\omega;t)P(\omega)}{P(D;t)} \leftrightarrow P(\omega|D;t) = p_X$$

We must then use these **pointwise evaluations** to produce our estimates.

The posterior distribution



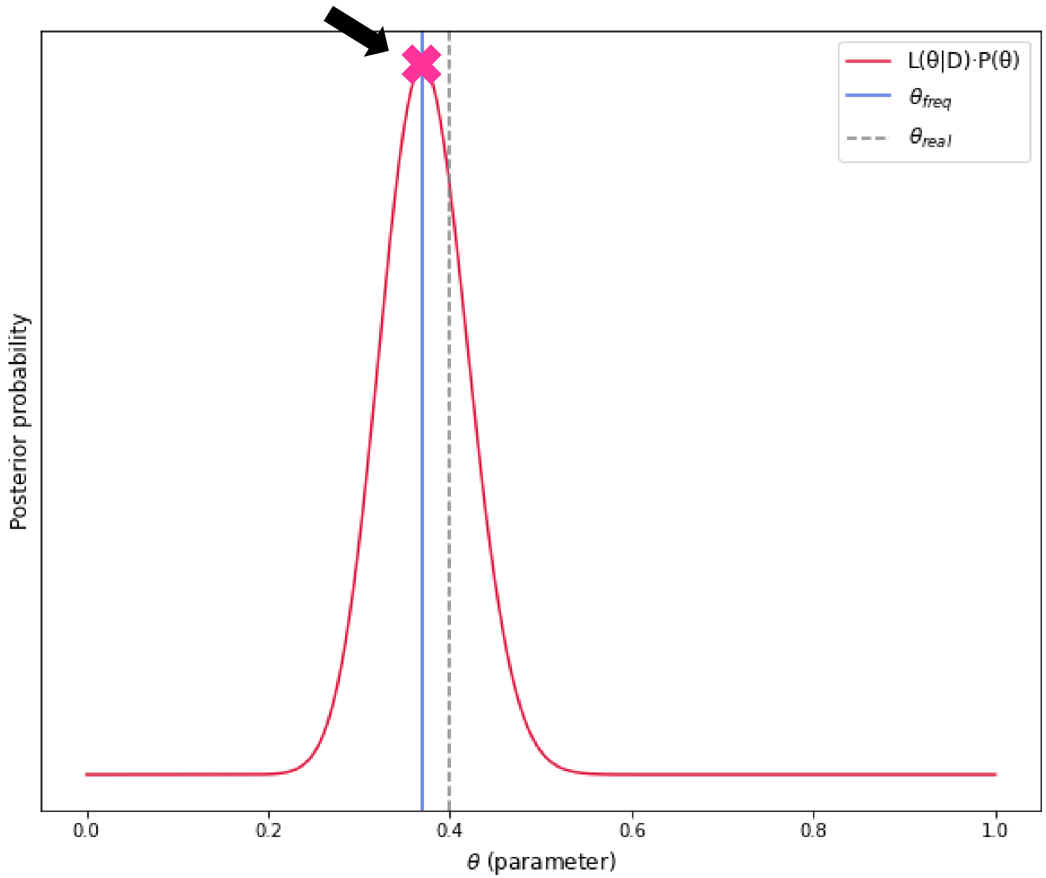
frequentist point estimate:

$$\omega = \omega_X$$

Bayesian posterior:

$$P(\omega|D;t) = p_X$$

The posterior distribution



frequentist point estimate:

$$\omega = \omega_X$$

Bayesian posterior:

$$P(\omega|D;t) = p_X$$

Querying the posterior

There are two known treatment options for the posterior:

Optimization

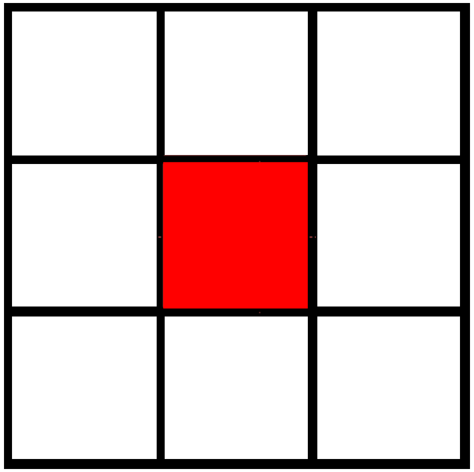
$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, \boldsymbol{P(\theta|D)}$$

Integration

$$\mathbb{E}[f(\theta)] = \int f(\theta) \boldsymbol{P(\theta|D)\, d\theta}$$

The behaviour of a probability distribution is dictated by **probability mass**, $P(\theta|D)\,d\theta$, whereas optimization considers the **density** $P(\theta|D)$ without heeding the **volume** $d\theta$. This blatant disregard is based on a naive, unscalable intuition.

Querying the posterior



As dimensionality increases, the fraction of volume contained within any given region shrinks exponentially. Likewise, **the volume occupied by the highest density points becomes negligible**.

As a consequence, **integration is the only sensible option** for anything other than simple (low-dimensional, convex) models.

Querying the posterior

$$\mathbb{E}[f(\theta)] = \int f(\theta) P(\theta|D)\, d\theta$$

We can then **retrieve any estimate** we want by **integrating over the posterior**. For instance, we can take the **mean** as our **parameter** estimator:

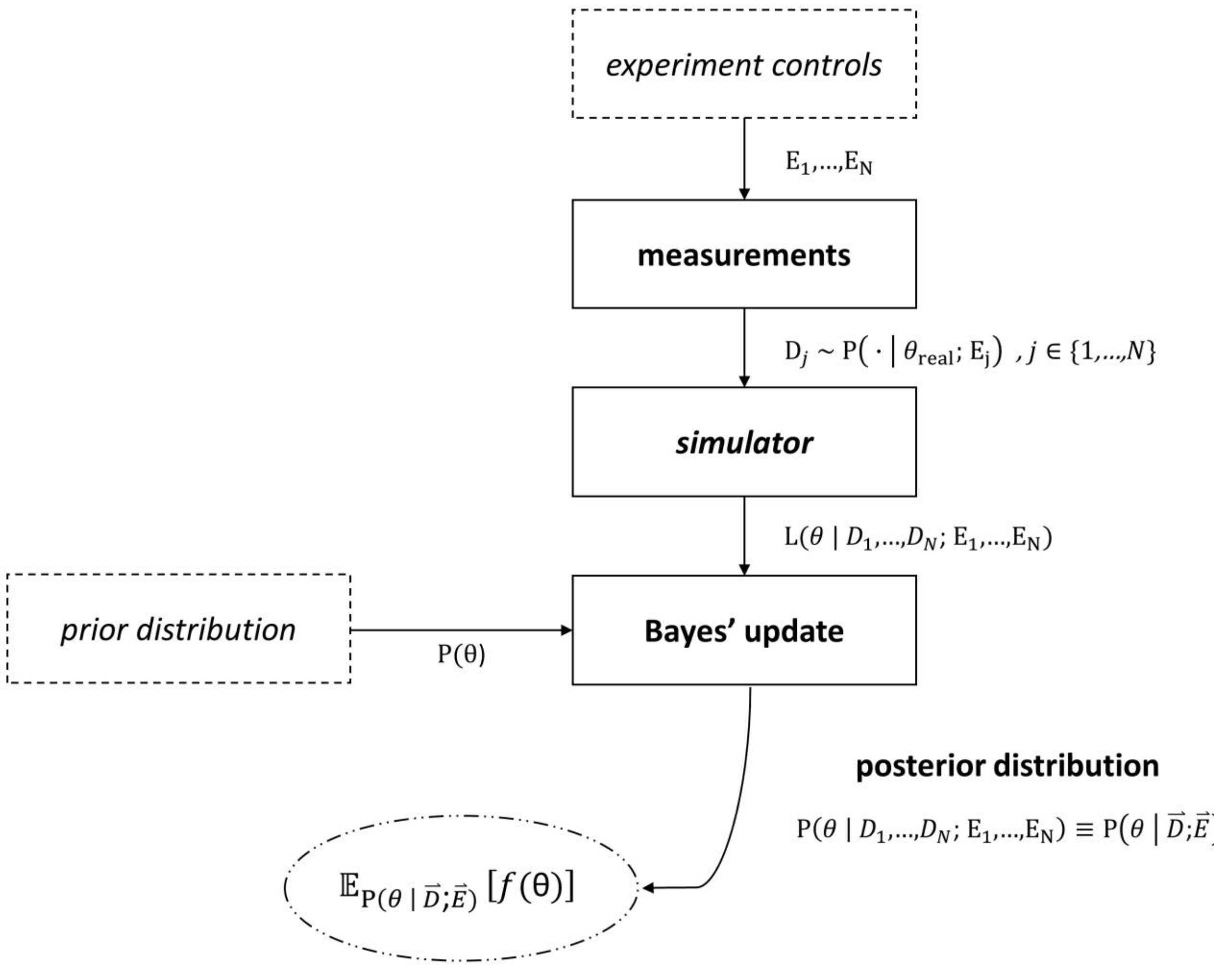
$$\hat{\theta} = \bar{\theta} \equiv \mathbb{E}[\theta]$$

And the **standard deviation** as our **uncertainty** estimator:

$$\sigma^2 = \mathbb{E}[(\theta - \bar{\theta})^2]$$

We can compute many other expectations, be they **statistical assessments** (e.g. interquartile ranges, credible regions) or **empirical predictions** (e.g. probability of finding the qubit at state X when measuring on some basis at a time Y).

Summary: the algorithm



Why Bayesian inference?

This strategy may seem more cumbersome than a simple frequentist guess, but **the framework is also much richer**.

- It is very **general**: it can be applied whenever one can measure a system and simulate parametrized descriptions of it.
- It naturally **estimates the uncertainty** and any other well-defined quantities.
- It is remarkably **robust** and **scalable**.
- It is capable of **online estimation**, and of **learning from small datasets**.

Open problems

querying the system

How to efficiently design informative experiments ?

data → posterior distribution

Finding an efficient ***optimization*** *strategy for the experimental controls*

querying the posterior

How to efficiently capture the information ?

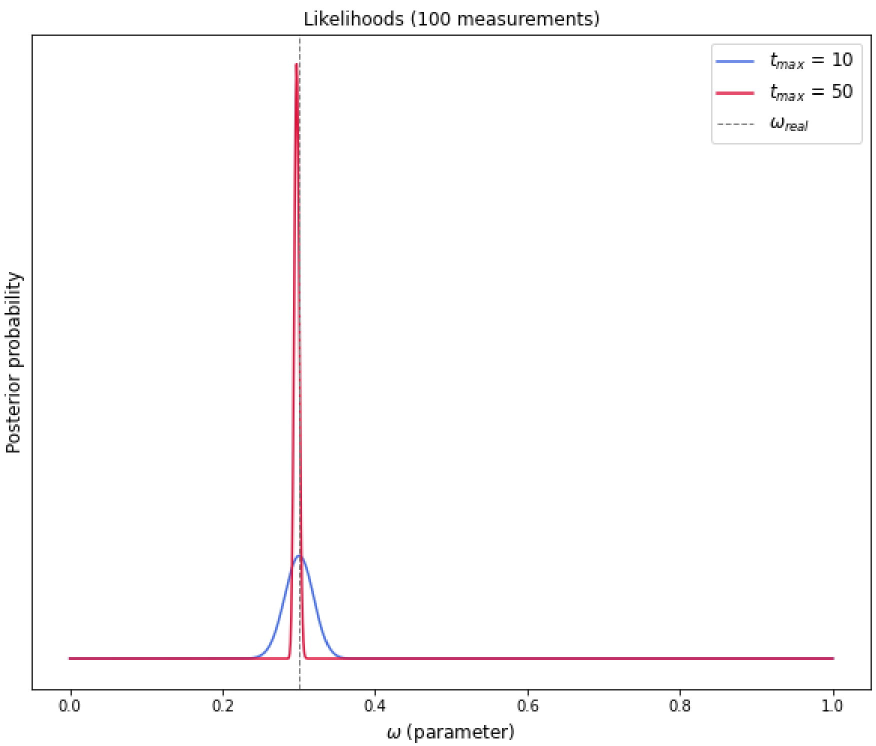
posterior distribution → **predictions**

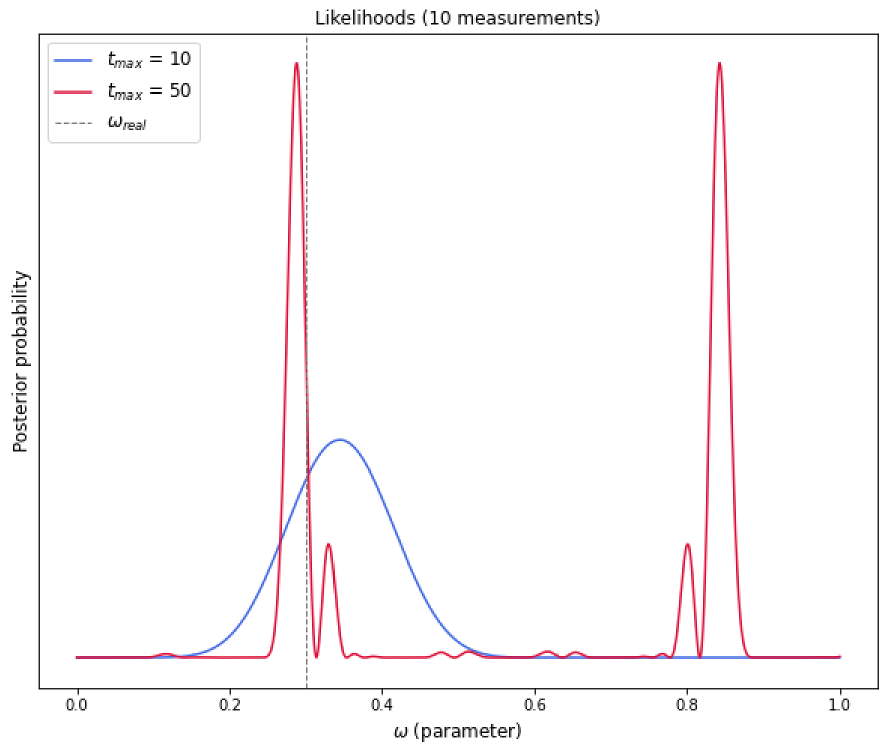
Finding an efficient ***integration*** *strategy for the posterior*

How to efficiently design informative experiments?

The experiments ultimately determine the **sharpness** of the posterior

...and even its **ambiguity**, or lack thereof





How to efficiently design informative experiments?

The Bayesian posterior allows for calculating any statistical expectation.

This includes the **expected utility** of any given **experiment**(s).

We can then use our knowledge to **choose the best possible set of experiments**.

However, this has two problems:

- We incur an **exponential cost** in the number of experiments.
- The knowledge we start out with is limited, and may not be very informative.

How to efficiently design informative experiments?

These two issues can be solved at once, by applying Bayes' rule sequentially:

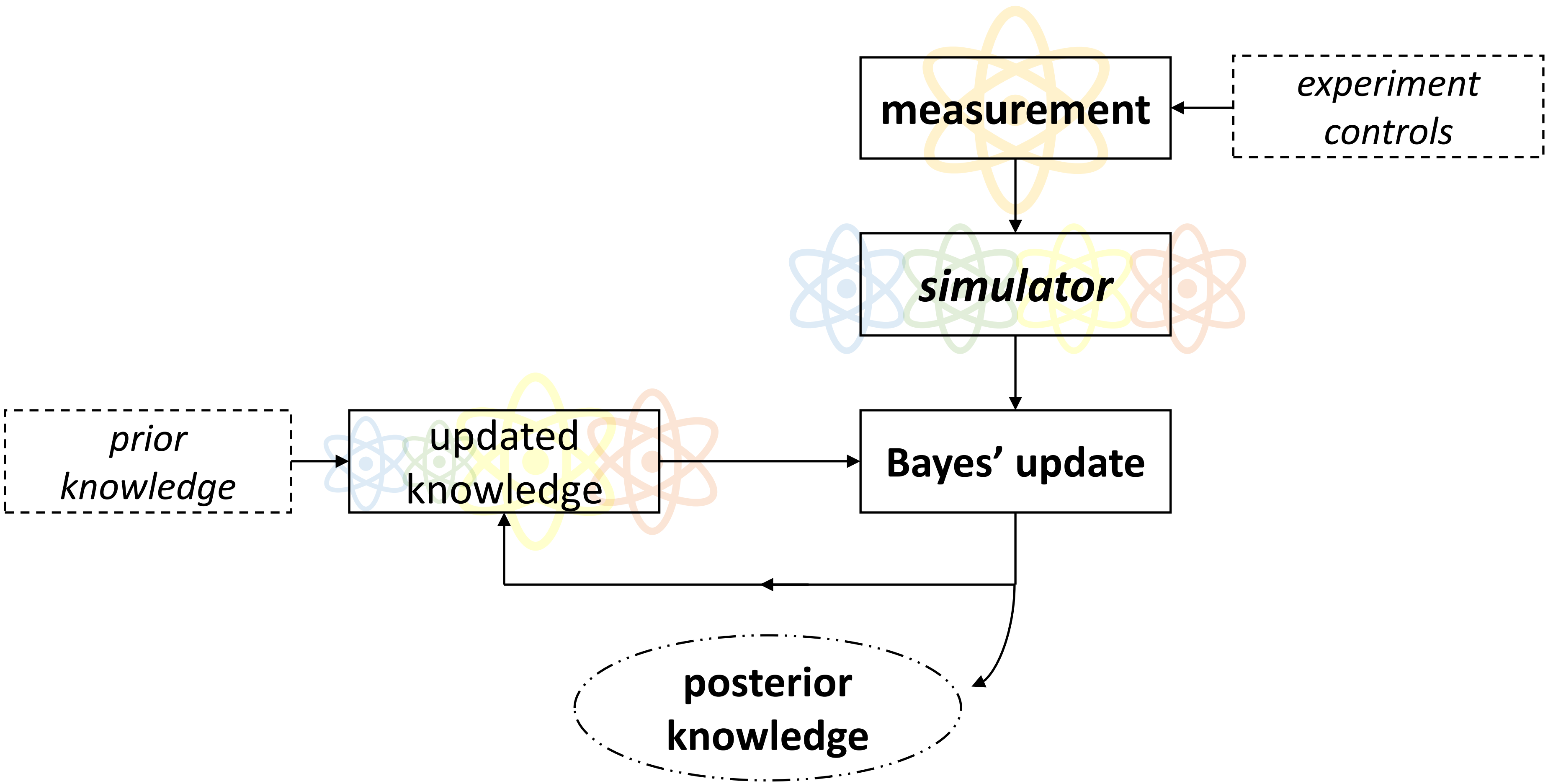
$$P\left(\theta \middle| \vec{D}\right) \propto P\left(\vec{D} \middle| \theta\right) P(\theta) \quad \longleftrightarrow$$

$$P(\theta|D_1) \propto P(D_1|\theta)P(\theta)$$

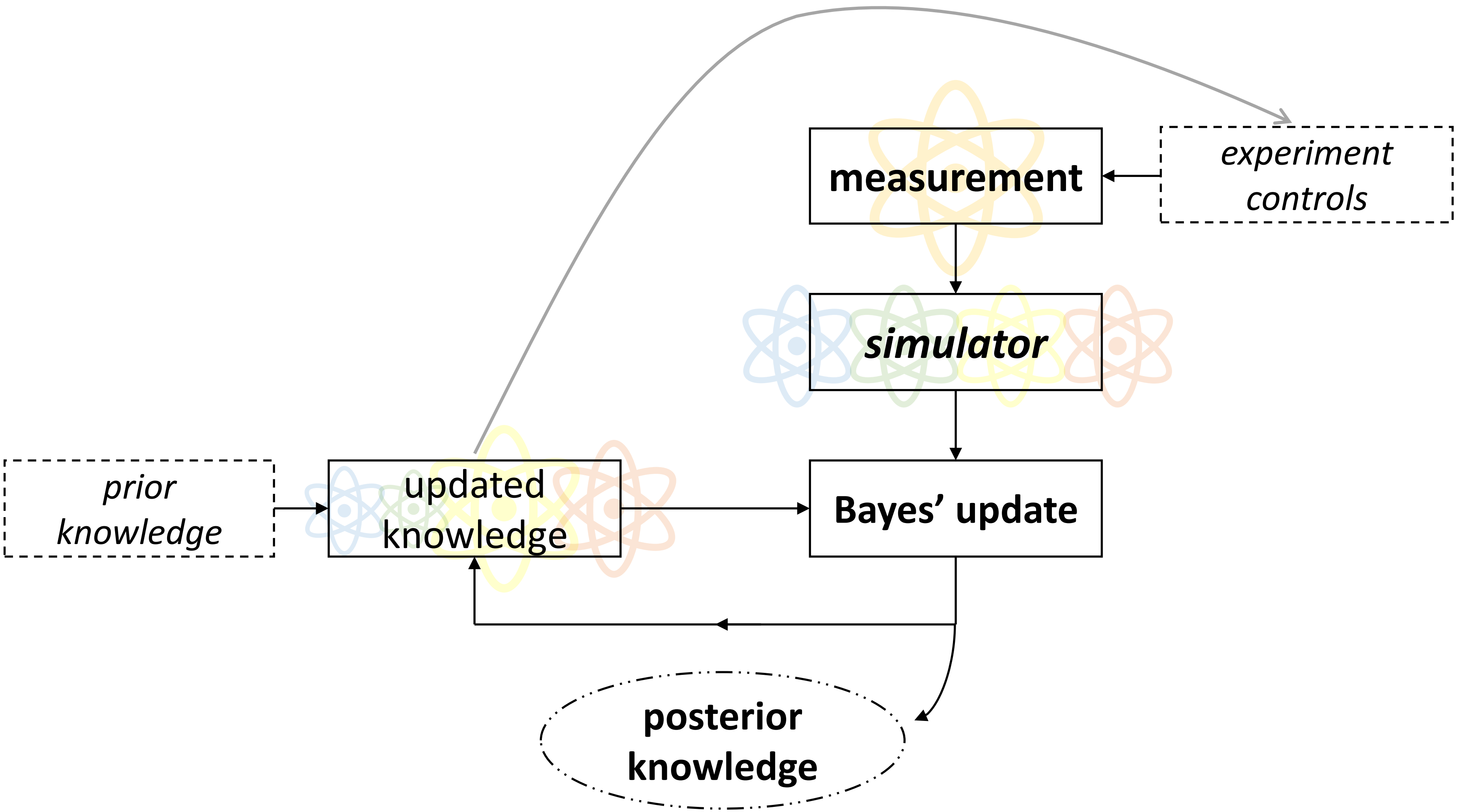
$$P(\theta|\{D_1, D_2\}) \propto P(D_2|\theta)P(\theta|D_1)$$

...

How to efficiently design informative experiments?



How to efficiently design informative experiments?



How to efficiently design informative experiments?

Adaptivity bring two major advantages:

- It brings the cost down to **linear** in the number of experiments.
- The experiments can be chosen according to **the full extent of our** growing **knowledge**

This greatly simplifies the problem, after which we can employ:

- (Local) optimization
- Heuristics (can be analytically/empirically motivated, or use machine learning)
- An in-between strategy

How to efficiently capture the information?

$$\mathbb{E}[f(\theta)] = \int f(\theta) P(\theta|D)\, d\theta$$

We have seen that sound statistical inquiries directed at a probability distribution correspond to integrals.

In general, **the main difficulty of scaling up Bayesian inference is integration.**

How to efficiently capture the information?

$$\mathbb{E}[f(\theta)] = \int f(\theta)P(\theta|D)\,d\theta$$

There are two major options for integration over probability distributions:

- Variational inference
- Numerical integration

How to efficiently capture the information?

$$\mathbb{E}[f(\theta)] = \int f(\theta)P(\theta|D)\, d\theta$$

There are two major options for integration over probability distributions:

- Variational inference
- **Numerical integration**

How to efficiently capture the information?

$$\mathbb{E}[f(\theta)] = \int f(\theta)P(\theta|D)\, d\theta$$

$$\approx \sum_i f(\theta_i) \cdot w_i$$

, with $w_i \propto \frac{P(\theta_i|D)}{\pi(\theta_i)}$, $\boldsymbol{\theta_i} \sim \pi(\cdot)$

Numerical integration

– Monte Carlo

How to efficiently capture the information?

$$\mathbb{E}[f(\theta)] = \int f(\theta)P(\theta|D)\,d\theta$$

The problem of numerical integration is to **choose the evaluation sites**

$$\boldsymbol{\theta_i} \sim \pi(\cdot)$$

Numerical integration

– Monte Carlo

How to efficiently capture the information?

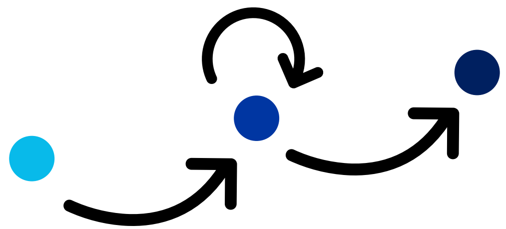
Two Monte Carlo solutions stand out:

- **Markov Chain Monte Carlo (MCMC)**
  - Random walk Metropolis
  - Hamiltonian Monte Carlo
- **Sequential Monte Carlo (SMC)**
  - Sequential importance resampling
  - Tempered likelihood estimation

How to efficiently capture the information?

$\{\boldsymbol{\theta}_{\boldsymbol{i}}\}_{\boldsymbol{i=1}}^{\boldsymbol{N}}$ = ?

It is simple to build a Markov Chain that samples from the posterior, using **random walk Metropolis**:



1. Choose a starting point θ_{curr}
2. For N iterations:
    1. Propose θ_{new} at random.
    2. Replace θ_{curr} with θ_{new} with probability

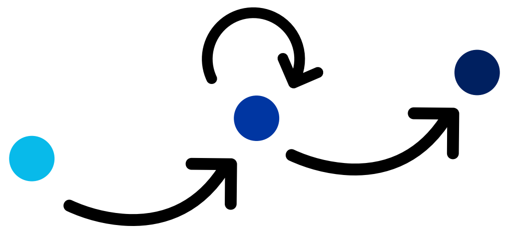
$$\frac{P(\theta_{\text{new}}|D)}{P(\theta_{\text{curr}}|D)}$$

The sequence of θ is distributed according to $P(\cdot\,|D)$ *as* $N \to \infty$

How to efficiently capture the information?

$\{\boldsymbol{\theta}_i\}_{i=1}^{N}$ = ?

It is simple to build a Markov Chain that samples from the posterior, using **random walk Metropolis**:



1. Choose a starting point θ_{curr}
2. For N iterations:
   1. Propose θ_{new} at random.
   2. Replace θ_{curr} with θ_{new} with probability

$$\frac{P(\theta_{\text{new}}|D)}{P(\theta_{\text{curr}}|D)}$$

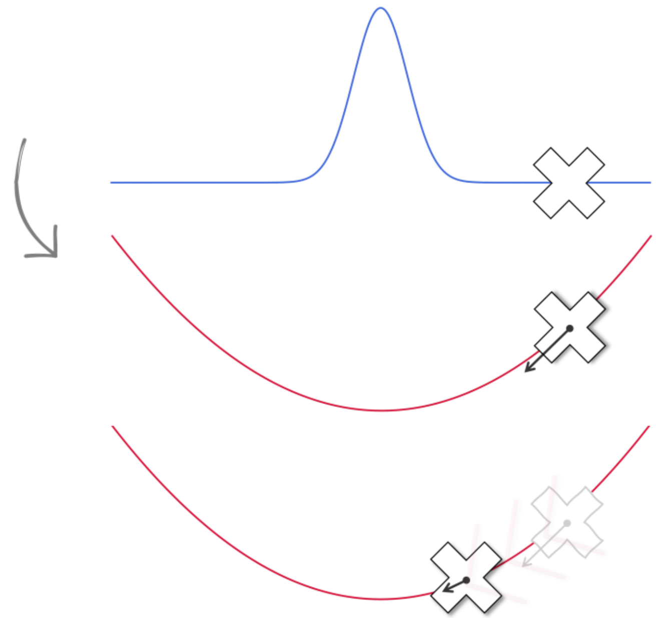
The sequence of θ is distributed according to $P(\cdot\,|D)$ *as* $N \rightarrow \infty$

How to efficiently capture the information?

$\{\boldsymbol{\theta}_i\}_{i=1}^{N}$ = ?

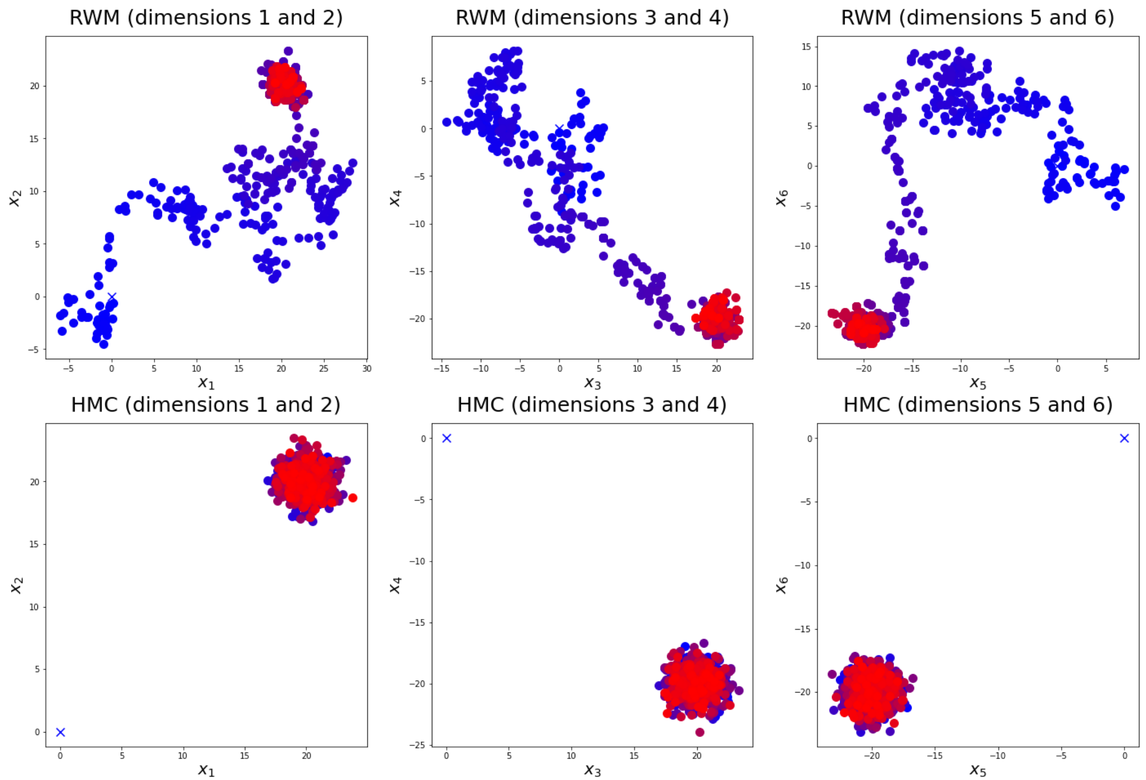
A more efficient approach must be informed by the differential geometry of the target distribution. This is the idea behind

Hamiltonian Monte Carlo

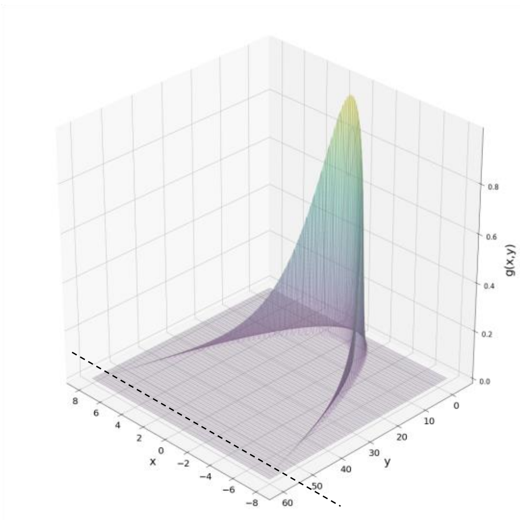


How to efficiently capture the information?

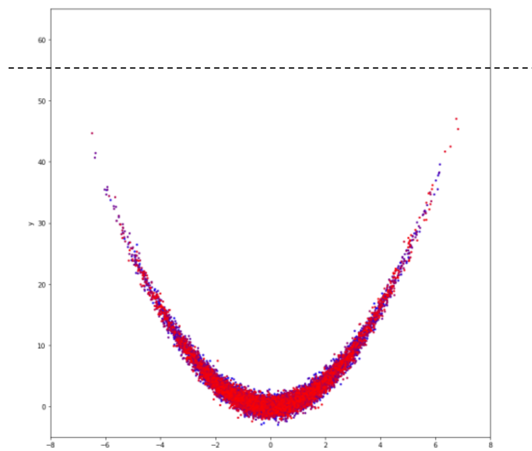
Random walk Metropolis (top) vs. Hamiltonian Monte Carlo (bottom) for a Gaussian:



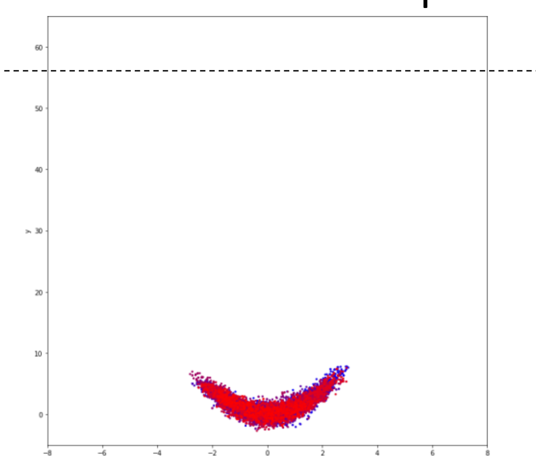
How to efficiently capture the information?



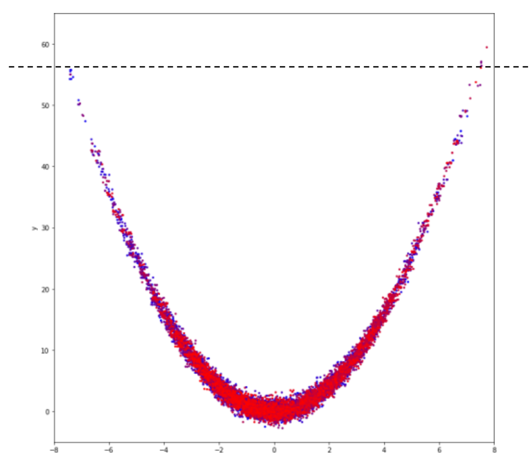
HMC



Random walk Metropolis



NUTS-HMC



Random walk Metropolis vs. Hamiltonian Monte Carlo for a Rosenbrock “smile” function

How to efficiently capture the information?

$\{\boldsymbol{\theta}_i\}_{i=1}^{N}$ = ?

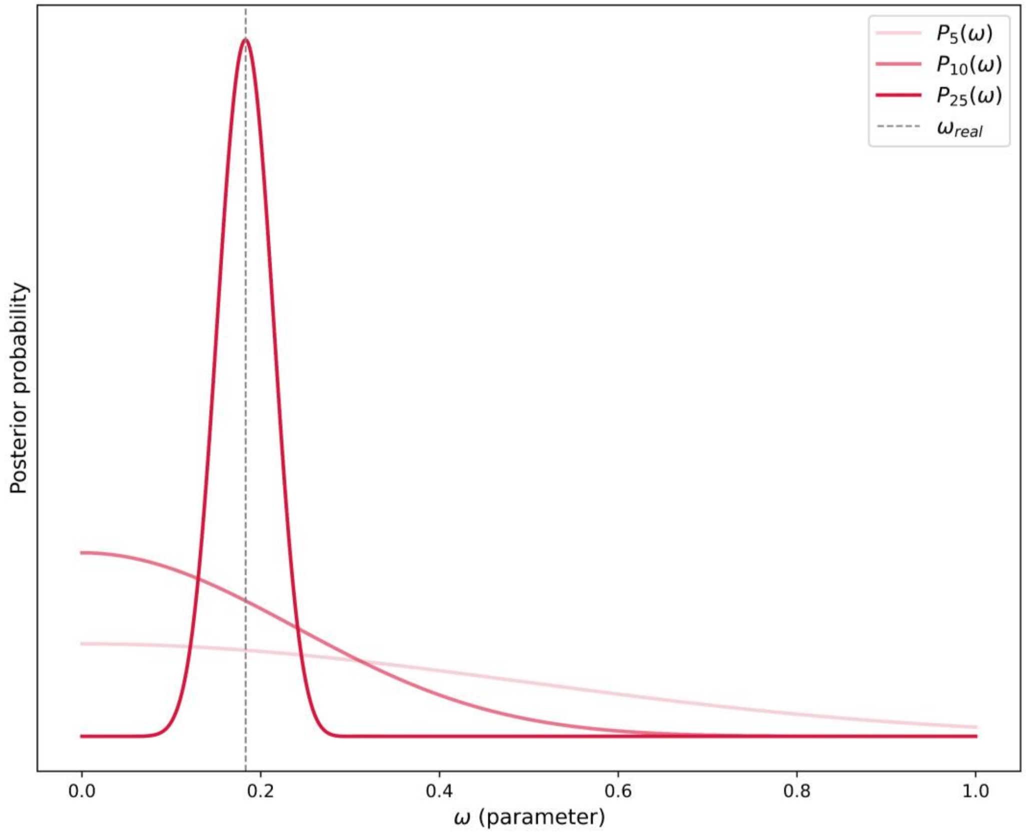
HMC enables a highly efficient exploration for a wide class of functions. However, there are still two problems:

- MCMC is **incompatible with sequential processing**, and thus online sensing and adaptivity.
- Markov chains are **unfit to capture multimodality**.

Often, these two problems can again be solved at once **using adaptivity**.

How to efficiently capture the information?

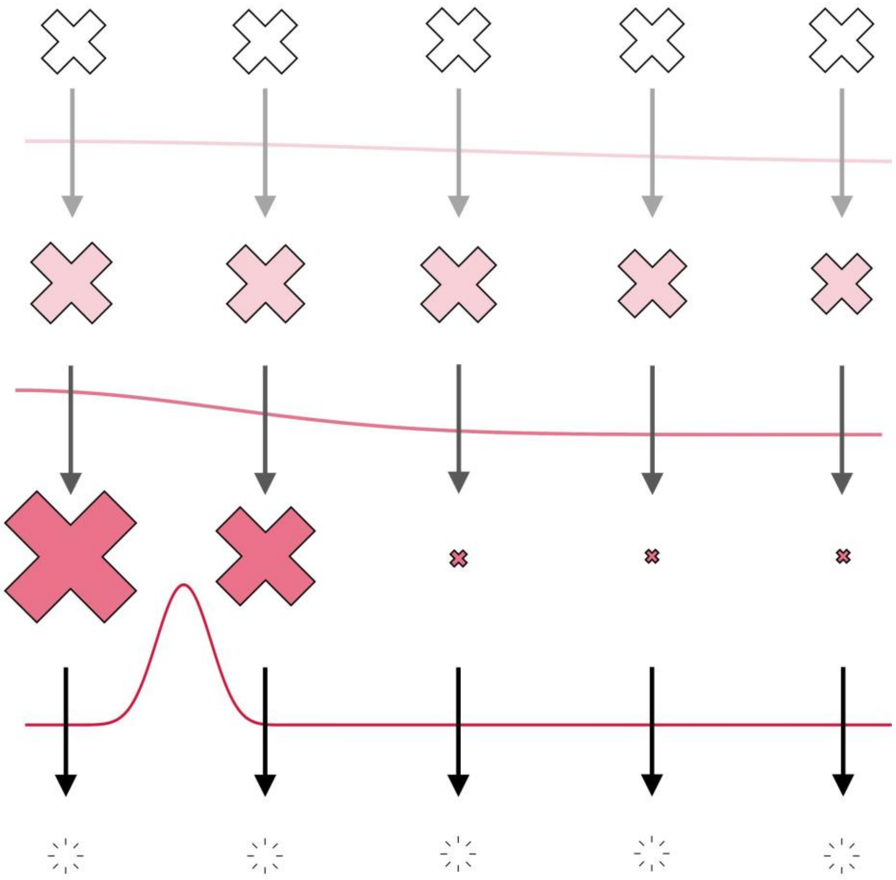
$\{\boldsymbol{\theta}_i\}_{i=1}^{N}$ = ?



Just as the experimental design, **integration may benefit from adaptivity**

How to efficiently capture the information?

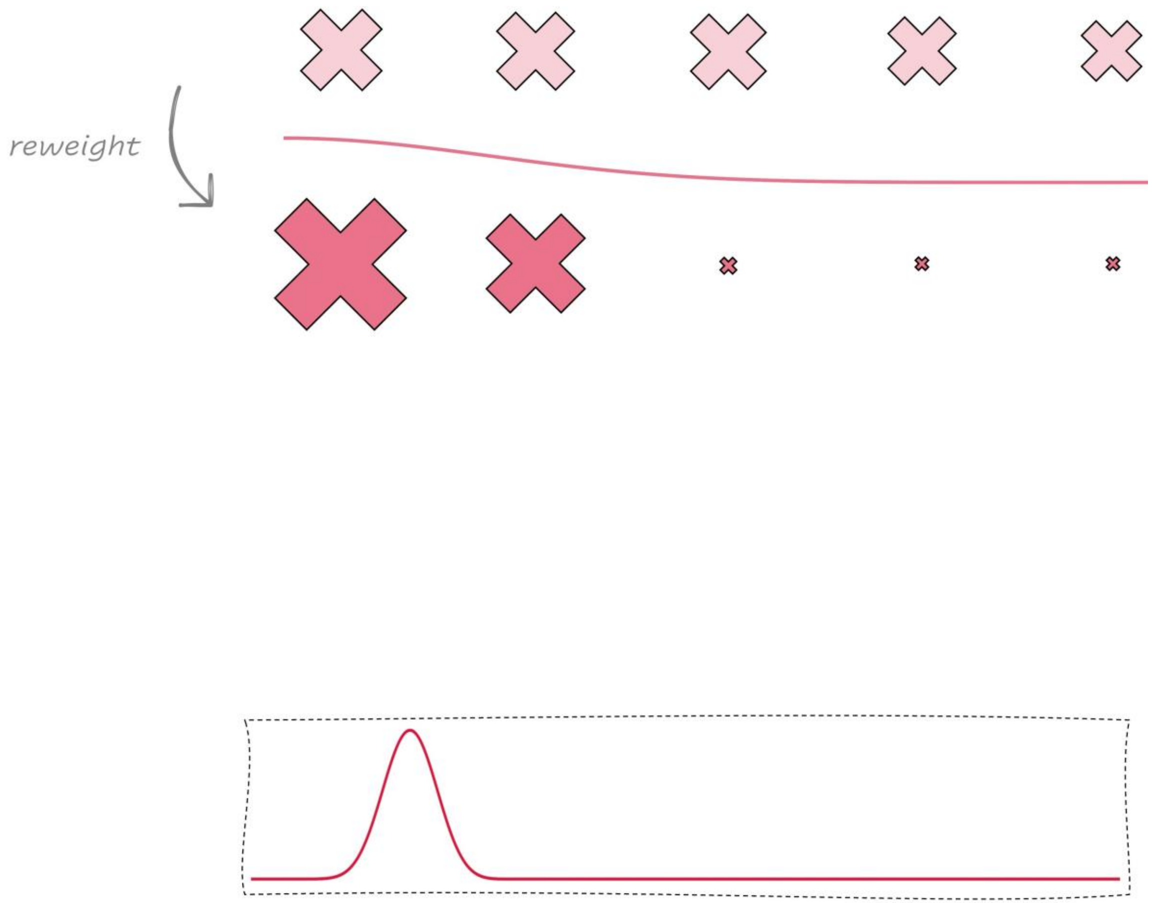
$\{\boldsymbol{\theta}_i\}_{i=1}^{N}$ = ?



A **grid**-based strategy can be made sequential, but it **leads to poor results** and rigid **exponential scaling**.

How to efficiently capture the information?

$\{\boldsymbol{\theta}_i\}_{i=1}^{N}$ = ?

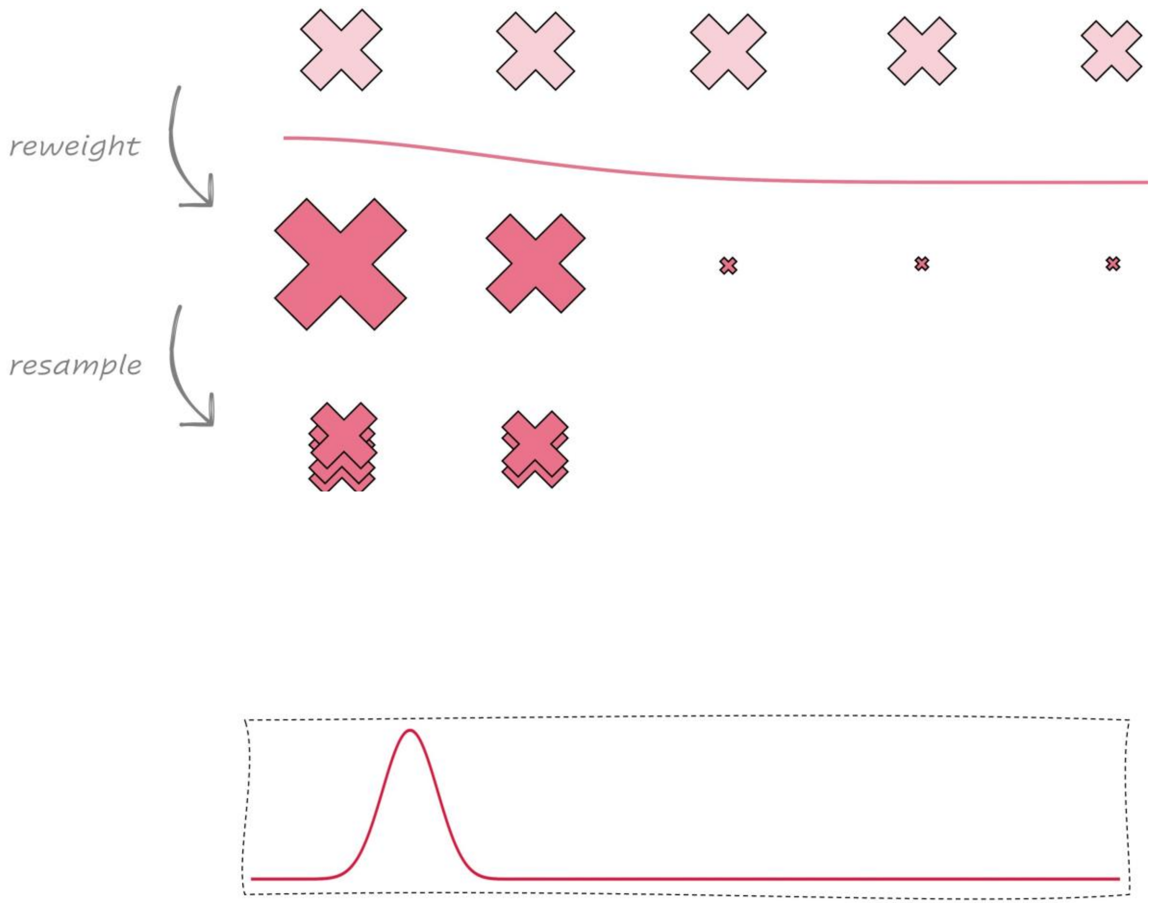


We can correct this by again exploiting the available knowledge, this time to **refocus the computational resources**.

Sequential Monte Carlo guides the construction of such a dynamic grid

How to efficiently capture the information?

$\{\boldsymbol{\theta}_i\}_{i=1}^{N}$ = ?

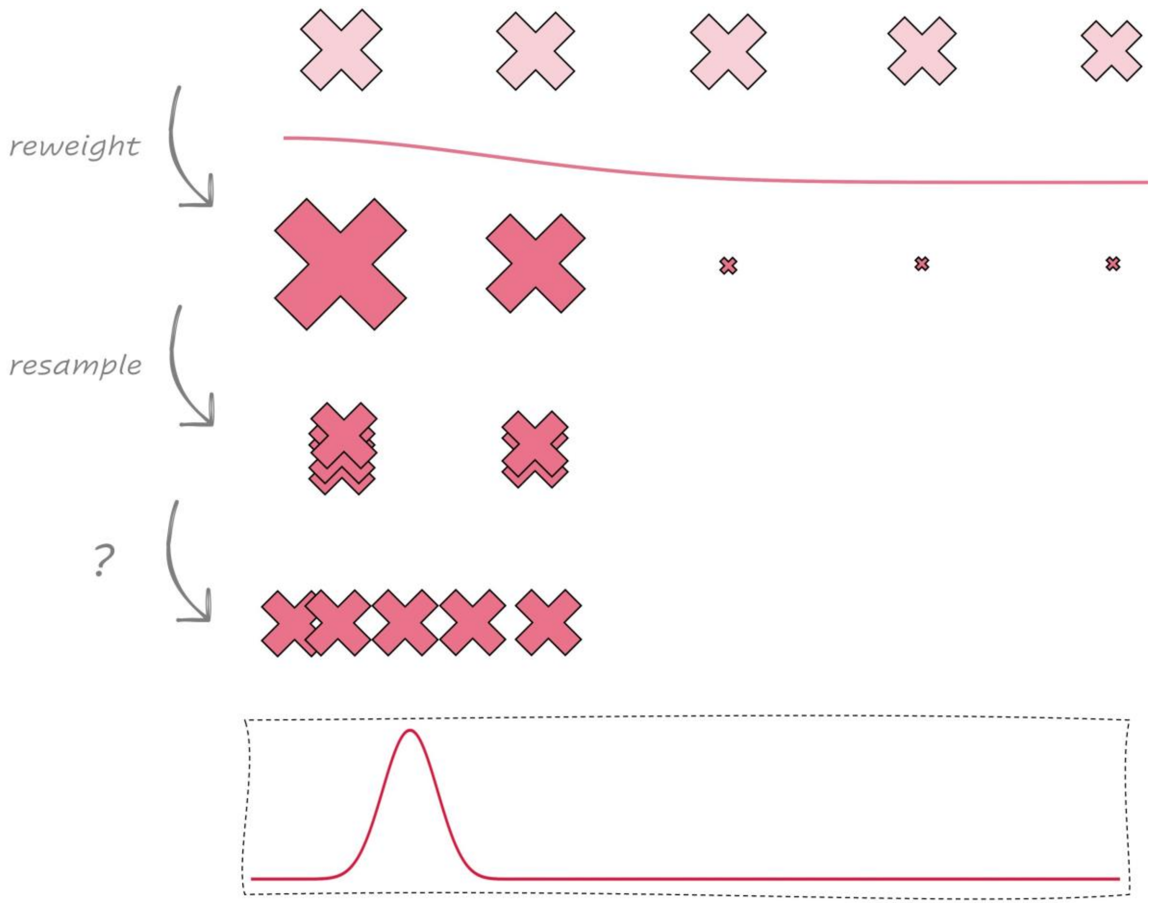


We can correct this by again exploiting the available knowledge, this time to **refocus the computational resources**.

Sequential Monte Carlo guides the construction of such a dynamic grid

How to efficiently capture the information?

$\{\boldsymbol{\theta}_i\}_{i=1}^{N}$ = ?



We can correct this by again exploiting the available knowledge, this time to **refocus the computational resources**.

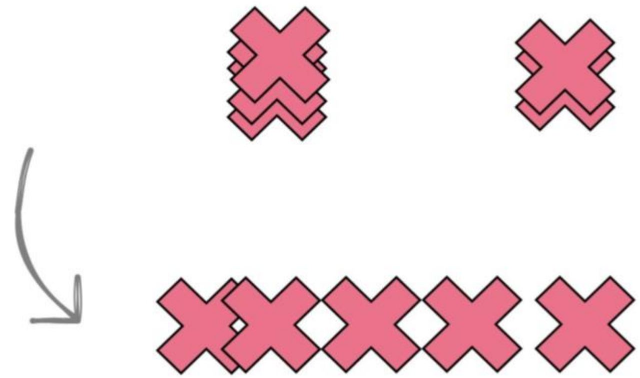
Sequential Monte Carlo guides the construction of such a dynamic grid

How to efficiently capture the information?

$\{\boldsymbol{\theta}_{\boldsymbol{i}}\}_{\boldsymbol{i=1}}^{\boldsymbol{N}}$ = ?

The **move step** is the most crucial. One possibility is to **induce random perturbations**, but we incur information loss. To compensate, we can **pull the points towards the center**.

This is called kernel smoothing and shrinkage, and is done in e.g. **the Liu-West particle filter**, the most (only?) used option in SMC for quantum characterization



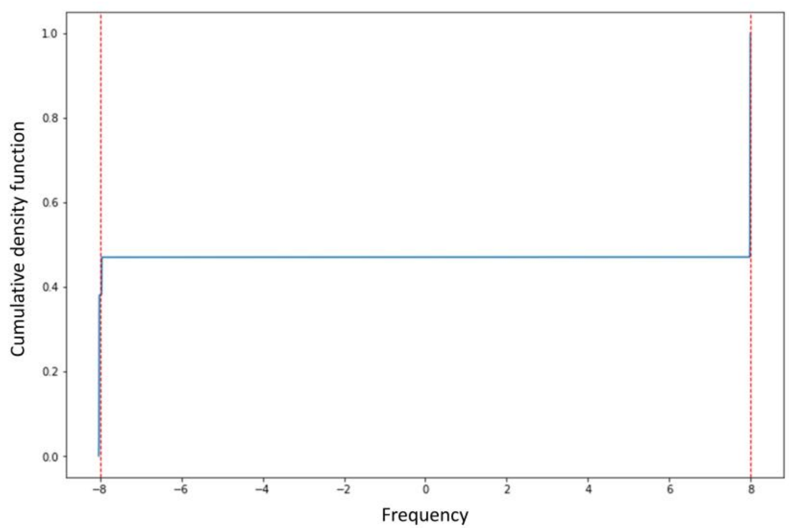
How to efficiently capture the information?

$\{\boldsymbol{\theta}_i\}_{i=1}^{N}$ = ?

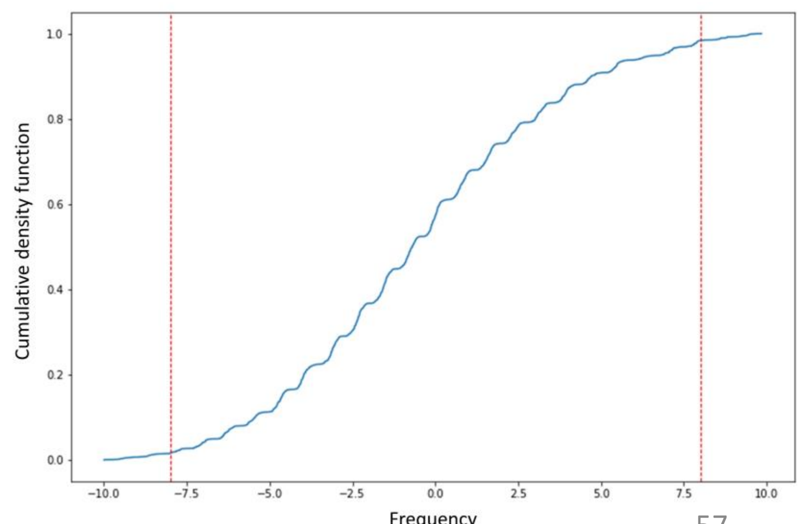
...but it only works perfectly for a Gaussian, and assumes unimodality. It can **bias expectations**, and **get trapped in local minima**.

A more reliable solution is to use **MCMC within SMC**:

with Hamiltonian Monte Carlo kernels

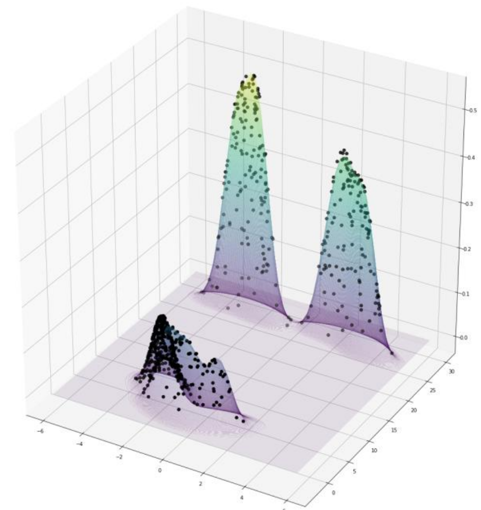


with the Liu-West filter

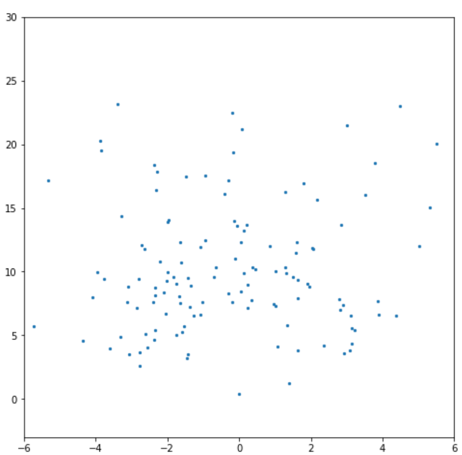


How to efficiently capture the information?

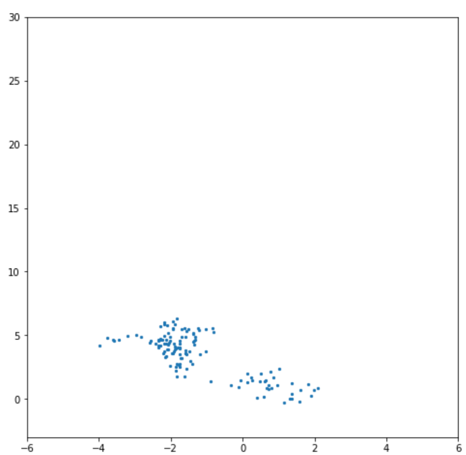
Random walk Metropolis vs. Hamiltonian Monte Carlo vs. their SMC counterparts for a“smiley” function



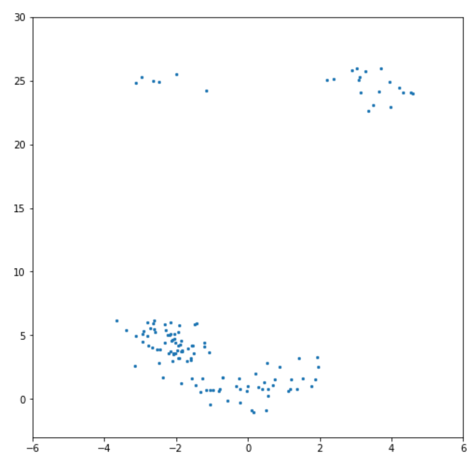
RWM



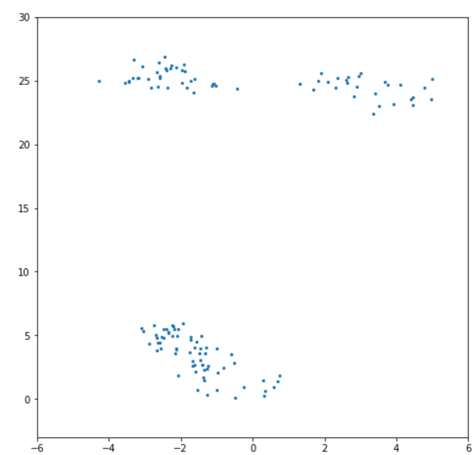
SMC-RWM



HMC

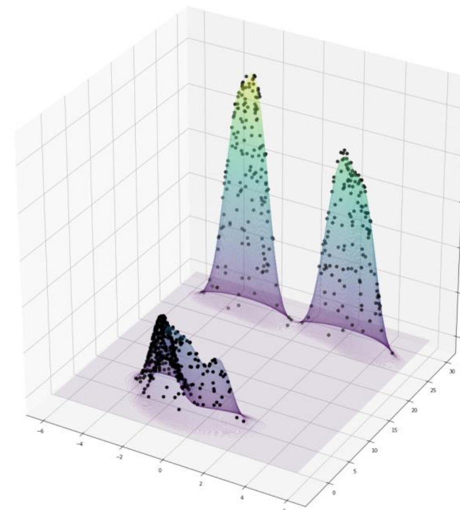


SHMC

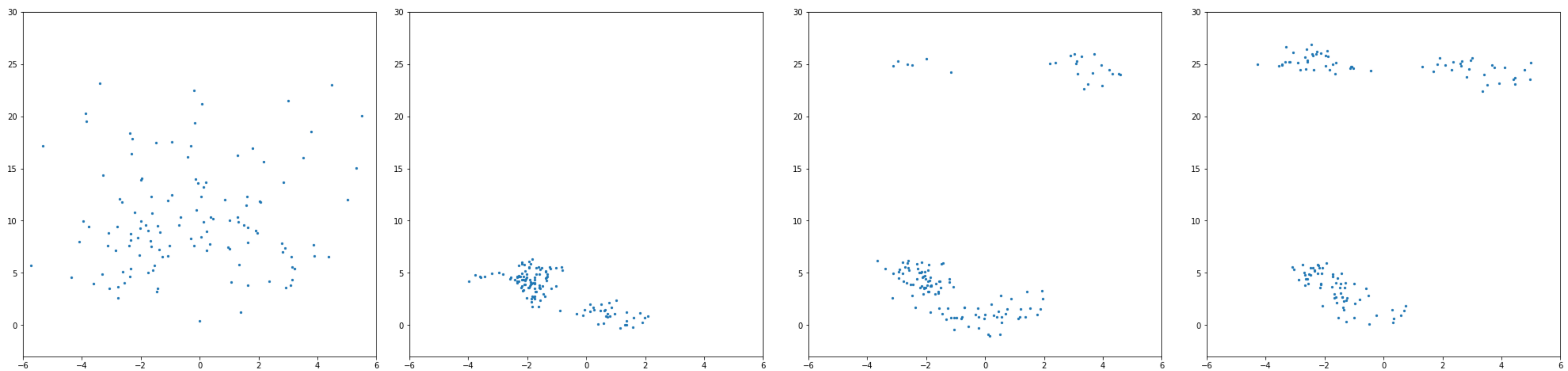


How to efficiently capture the information?

MCMC within SMC concils **online processing** with **robustness** and **correctness**



RWM SMC-RWM HMC SHMC



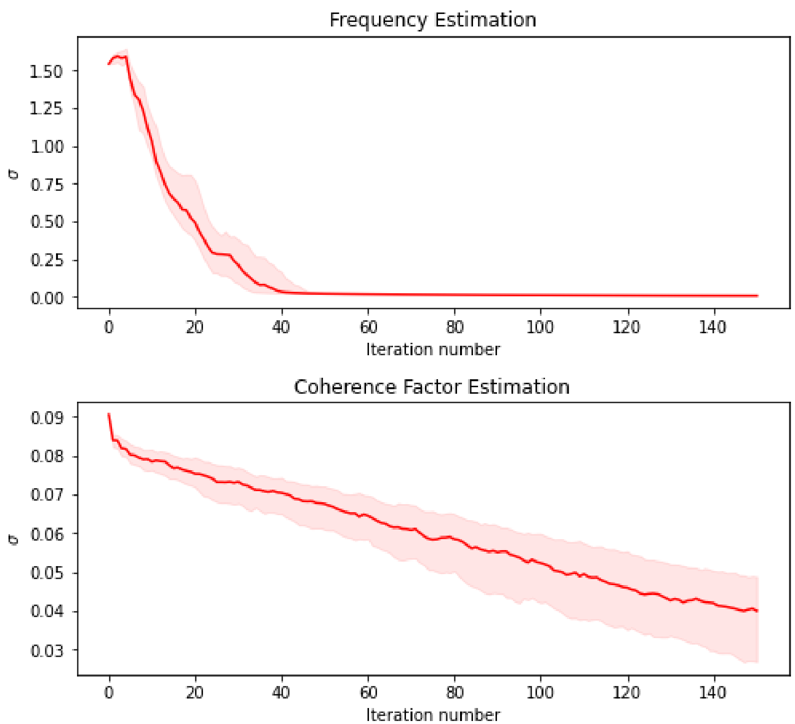
Characterization of an IBMQ qubit

Ramsey experiment on IBMQ device **`ibmq_armonk`**:
joint estimation of ω and T_2^*

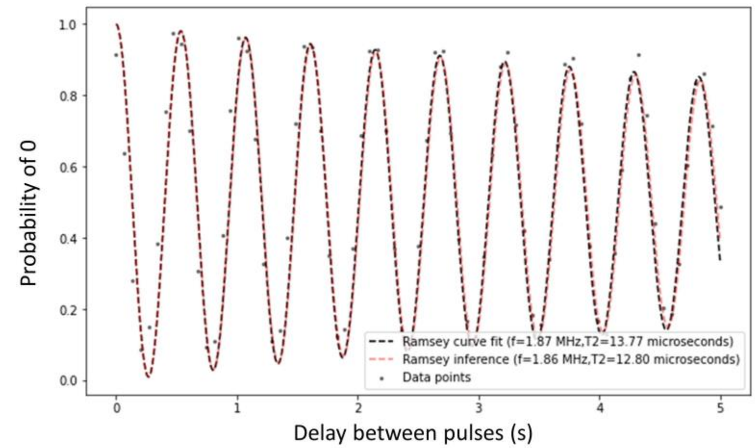
$$P(t) = e^{-t/\boldsymbol{T_2^*}} \cos^2\left({}^{\boldsymbol{\omega} t}/_2\right) + {}^1/_2\left(1 - e^{-t/\boldsymbol{T_2^*}}\right)$$

- Data collection with **Qiskit Pulse**
- Representation via **Sequential Monte Carlo** with **Markov kernels**
- **Each iteration** corresponds to **one datum**
- **Median values** over 100 runs
- Matched against the **default curve fitters**

Characterization of an IBMQ qubit



Evolution of the inference's **uncertainty** with the iterations/data

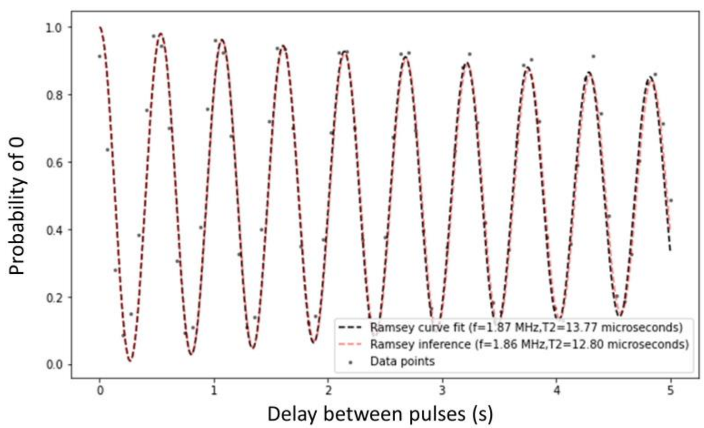


- **Data points**: **512** measurements per time
- **Dashed curve**: obtained by regression on data points
- **Red curve**: Bayesian inference using less than **0.4%** of the data

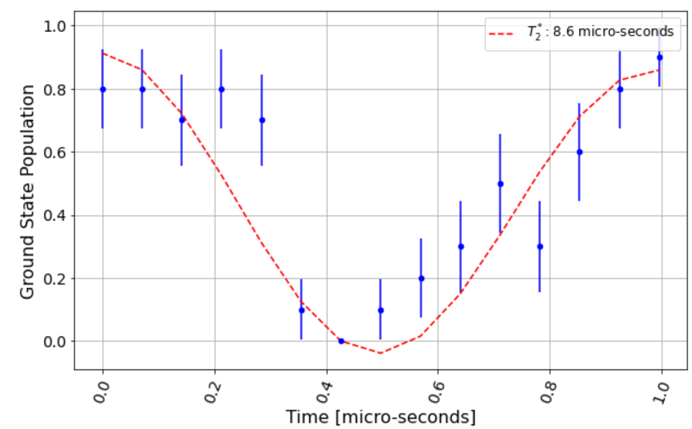
Characterization of an IBMQ qubit

Allowing the fitter as many data as the inference and the same measurement scheme resulted in an **infinite uncertainty** in T_2^*. Finite values could be achieved for fits on shorter intervals, but the best observed value was still **10 times worse**, whereas the frequency estimation performed roughly **40 times worse** (0.4% vs. 15%)

Bayesian inference



Qiskit fitter

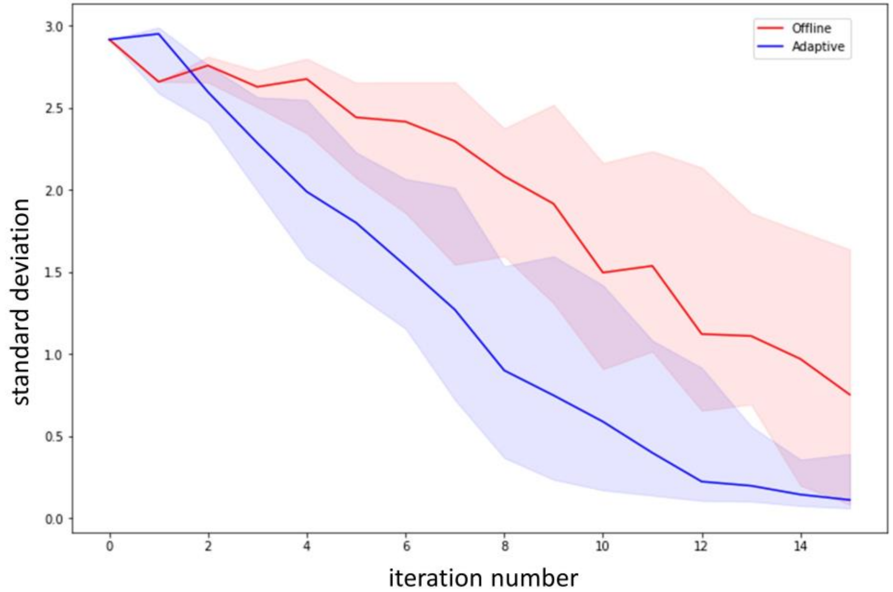


	T_2^* (μs)	**Standard deviation**	**Total shot count**
Bayesian inference	9	3	$75 \cdot 2 = 150$
Qiskit fitter	21	1	$75 \cdot 512 = 38\,400$
Qiskit fitter	9	30	$15 \cdot 10 = 150$

Note: direct comparisons for T_2^* are not viable due to fast fluctuations

Characterization of an IBMQ qubit

Adaptive *Hahn-Ramsey* experiment on IBMQ device **ibmq_armonk**

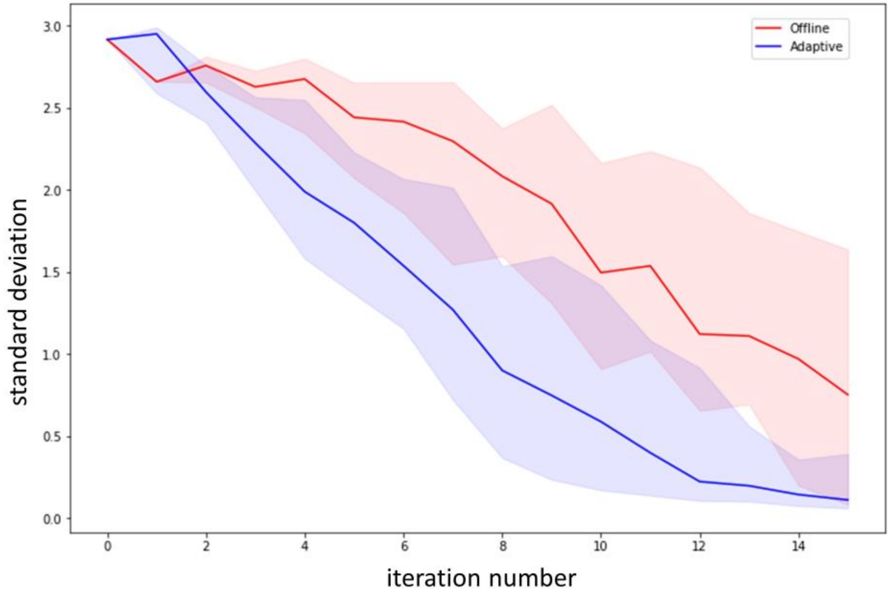


$$P(t) = \cos^2(\omega t/2)$$

	δ (MHz)	**Standard deviation**	**Total shot count**	**Precision** $(\sigma^2 \Delta t_{\text{acc}})$
Adaptive	1.8	0.1	15	0.38
Offline	2.3	0.8	15	9.3

Characterization of an IBMQ qubit

Adaptive *Hahn-Ramsey* experiment on IBMQ device **ibmq_armonk**



$$P(t) = \cos^2(\omega t/_2)$$

	$\delta\ (MHz)$	**Standard deviation**	**Total shot count**	**Precision** $(\sigma^2 \Delta t_{\text{acc}})$
Adaptive	1.8	0.1	15	0.38
Offline	2.3	0.8	15	9.3

Thank you for your attention!

Main bibliographic references

- **Sequential Monte Carlo**: Doucet et al. *An Introduction to Sequential Monte Carlo Methods* (Springer NY, 2001).
    - **Sequential importance resampling:** Granade et al. *Robust Online Hamiltonian Learning* (New J. Phys., 2012).
  - **Tempered likelihood estimation:**
    - Neal. *Annealed importance sampling* (Statistics and Computing, 2001).
    - South et al. *Sequential Monte Carlo Samplers with Independent Markov Chain Monte Carlo Proposals* (Bayesian Analysis, 2019).
  - **Liu-West filtering:** Liu et al. *Combined Parameter and State Estimation in Simulation-Based Filtering* (Springer NY, 2001).
  - **SHMC:** Daviet. *Inference with Hamiltonian Sequential Monte Carlo Simulators* (Elsevier BV, 2016).
  - **Subsampling:**
    - Gunawan et al. *Subsampling sequential Monte Carlo for static Bayesian models* (Springer Science and Business Media LCC, 2020).

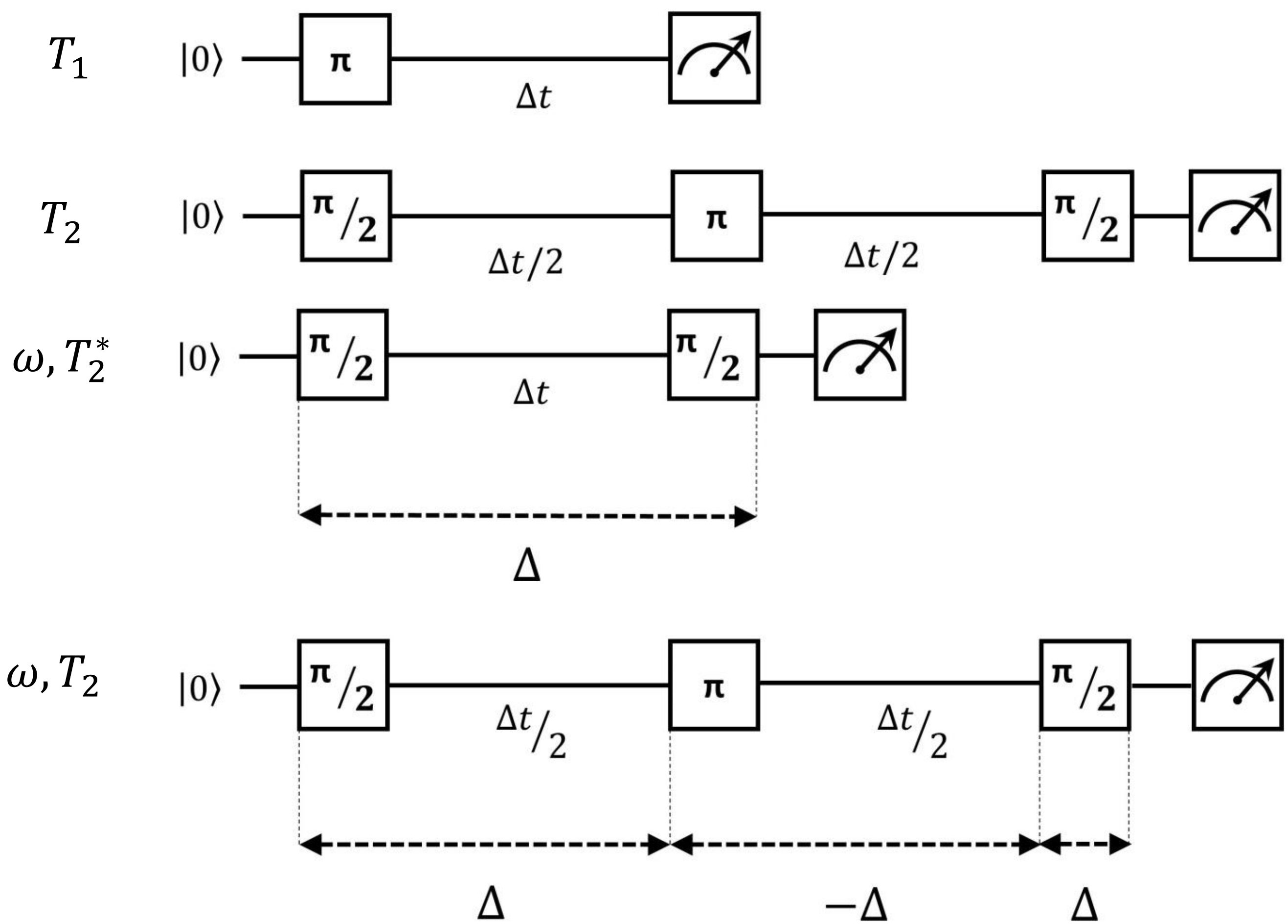
- Hamiltonian Monte Carlo:
  - Foundations:
    - Neal. *MCMC using Hamiltonian dynamics* (Chapman and Hall/CRC, 2011).
    - Betancourt. *A conceptual introduction to Hamiltonian Monte Carlo* (Arxiv e-print, 2018).
  - Variants:
    - Hoffman et al. *The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo* (Journal of Machine Learning Research, 2011).
    - Betancourt. *Adiabatic Monte Carlo* (Arxiv e-print, 2015).
    - Girolami et al. Riemann manifold Langevin and Hamiltonian Monte Carlo methods (Journal of the Royal Statistical Society, 2011).
  - Subsampling:
    - Dang et al. *Hamiltonian Monte Carlo with Energy Conserving Subsampling* (Journal of Machine Learning Research, 2019).
    - Quiroz et al. *Speeding up MCMC by Efficient Data Subsampling* (Journal of the American Statistical Association, 2018).
    - Betancourt. *The Fundamental Incompatibility of Scalable Hamiltonian Monte Carlo and Naive Data Subsampling* (Proceedings of the 32nd International Conference on Machine Learning, 2015).
    - Li et al. *Stochastic gradient Hamiltonian Monte Carlo with variance reduction for Bayesian inference* (Machine Learning, 2019).

- Iterative phase estimation: Svore et. al. *Faster Phase Estimation* (Quant. Inf. Comp 2013).
  - Bayesian adaptive IPE:
    - Higgins et. al. *Entanglement-free Heisenberg-limited phase estimation* (Nature, 2007).
    - Wiebe et. al. *Efficient Bayesian phase estimation* (Phys. Rev. Letters, 2016).
    - Lumino et. al. *Experimental Phase Estimation Enhanced By Machine Learning* (Phys. Rev. Applied, 2017).
  - Experimental implementation: Paesani et. al. *Experimental Bayesian Quantum Phase Estimation on a Silicon Photonic Chip* (Phys. Rev. Letters, 2016).
- Quantum channels:
  - Lidar. *Lecture Notes on the Theory of Open Quantum Systems* (Arxiv e-print, 2020)
  - Minev. *An introduction to open quantum systems* (Qiskit texbook, 2021).
  - Nielsen & Chuang. *Quantum Computation and Quantum Information* (Cambridge University Press, 2011).

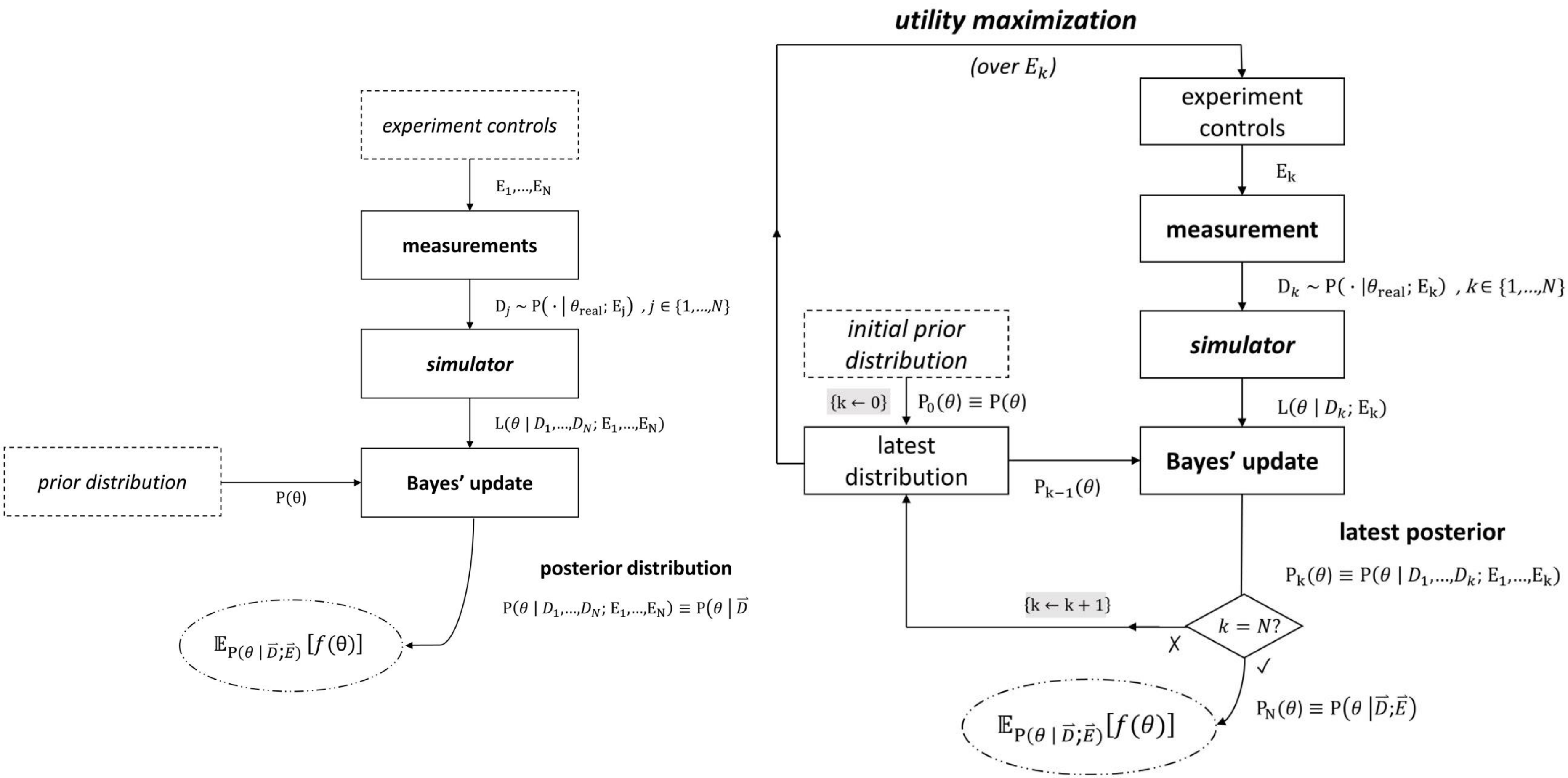
- Precession frequency estimation:
  - **Adaptivity**:
    - Ferrie et al. *Adaptive Hamiltonian Estimation Using Bayesian Experimental Design* (AIP Conf. Proc., 2011).
    - Granade et. al. *Robust Online Hamiltonian Learning* (New J. Phys., 2012).
    - Fiderer et al. *Neural-Network Heuristics for Adaptive Bayesian Quantum Estimation* (PRX Quantum, 2021).
    - Ferrie et al. *How to best sample a periodic probability distribution, or on the accuracy of Hamiltonian finding strategies* (Quantum information processing, 2012).
  - **Experimental implementation**: Wang et. al. *Experimental quantum Hamiltonian learning* (Nature, 2017).

Other graphs

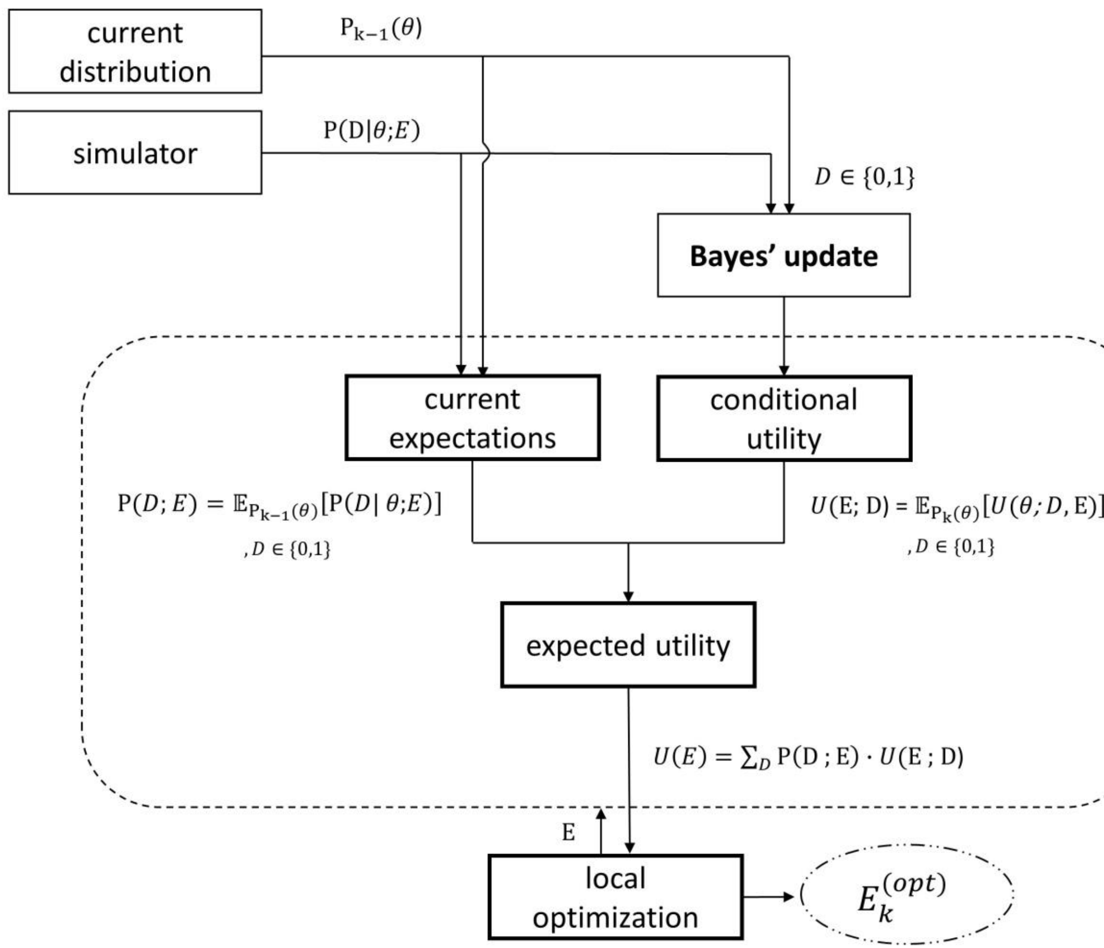
Pulse schedules



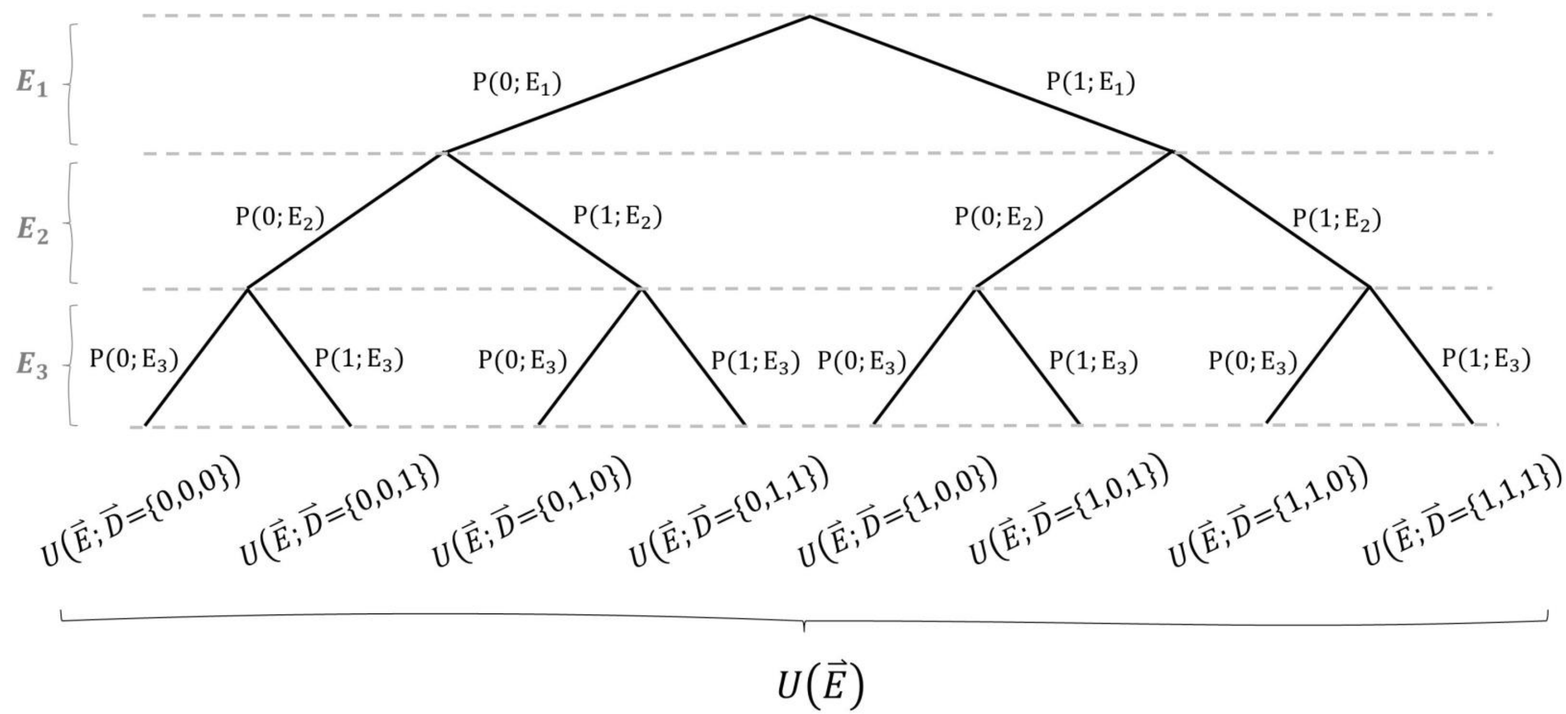
Offline vs. online inference



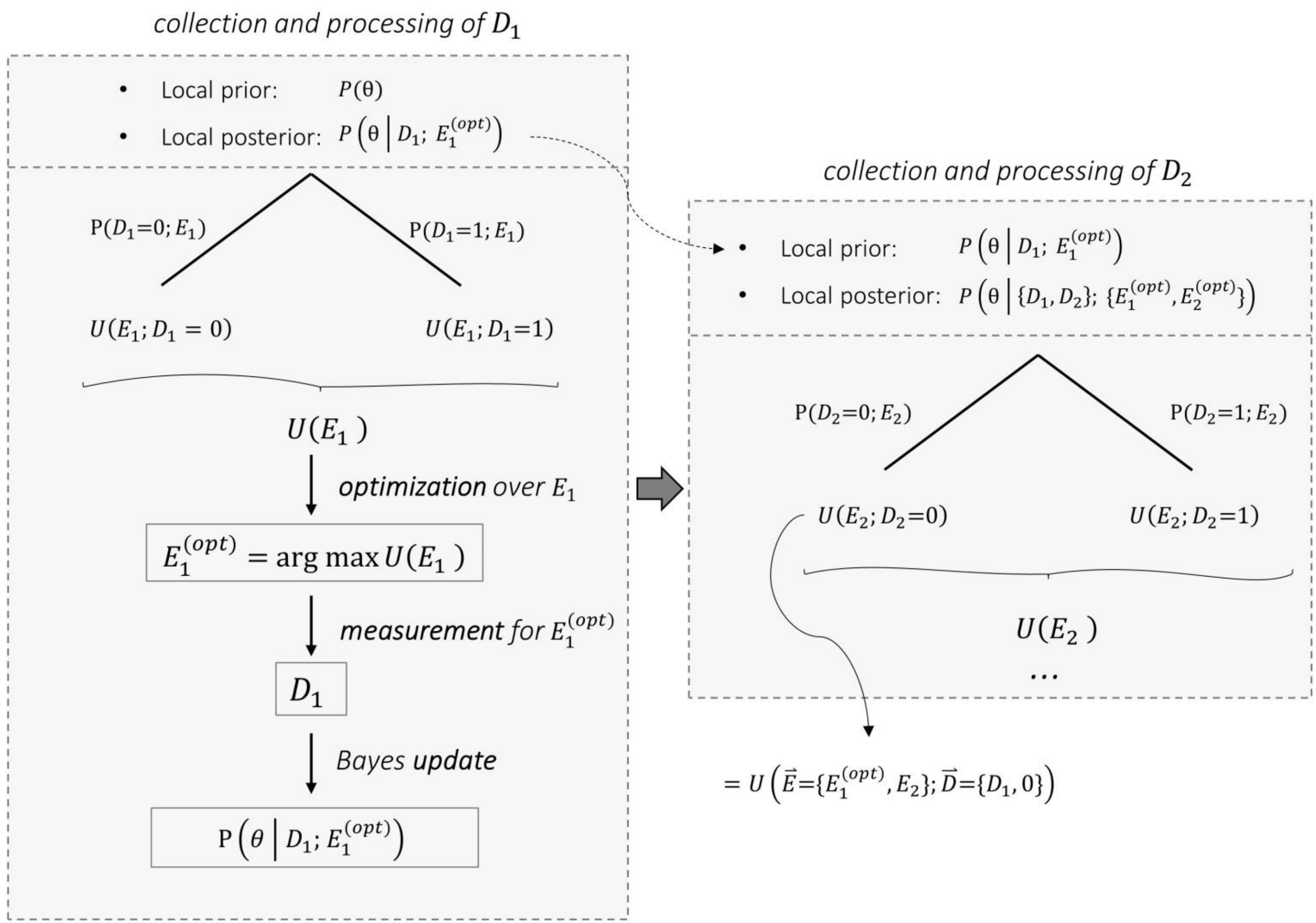
Offline vs. online inference



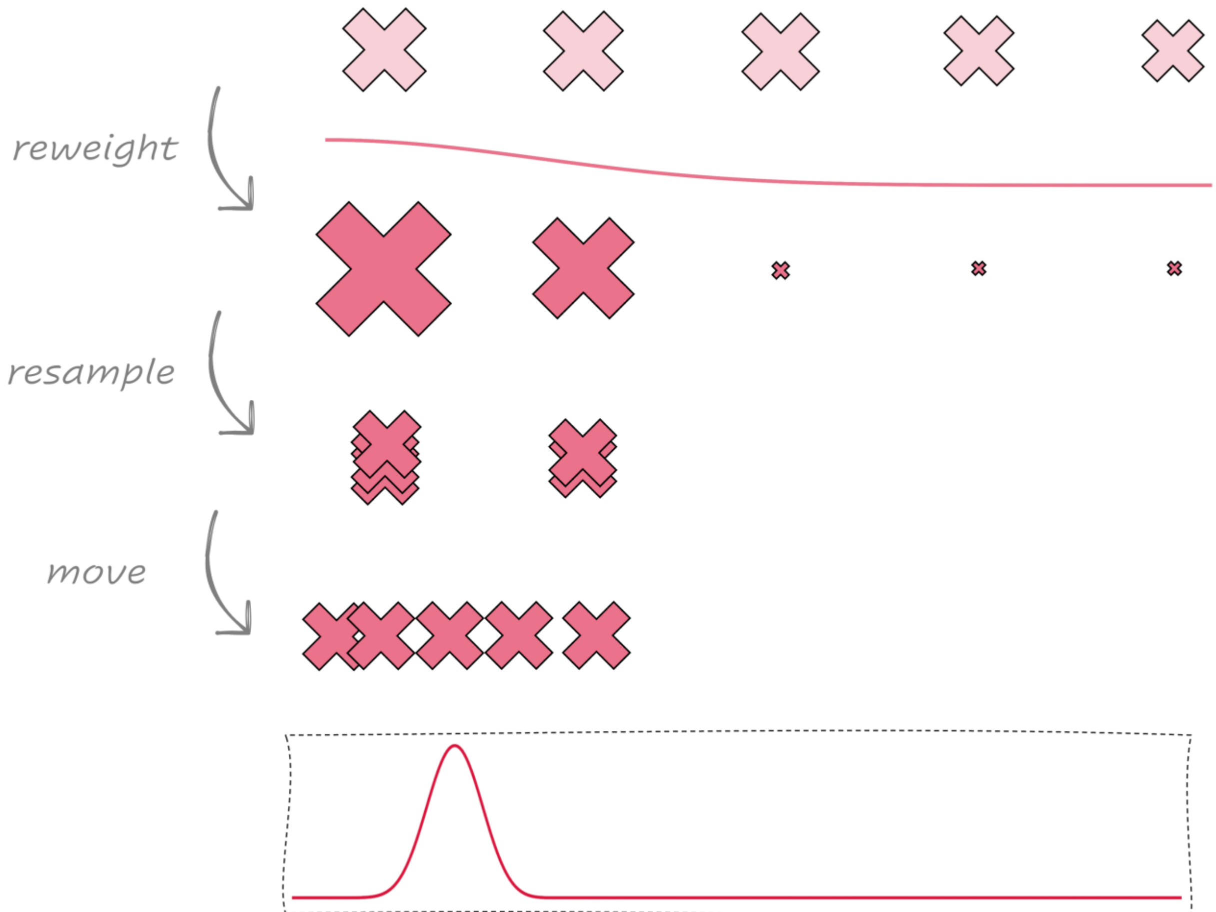
Offline vs. online inference



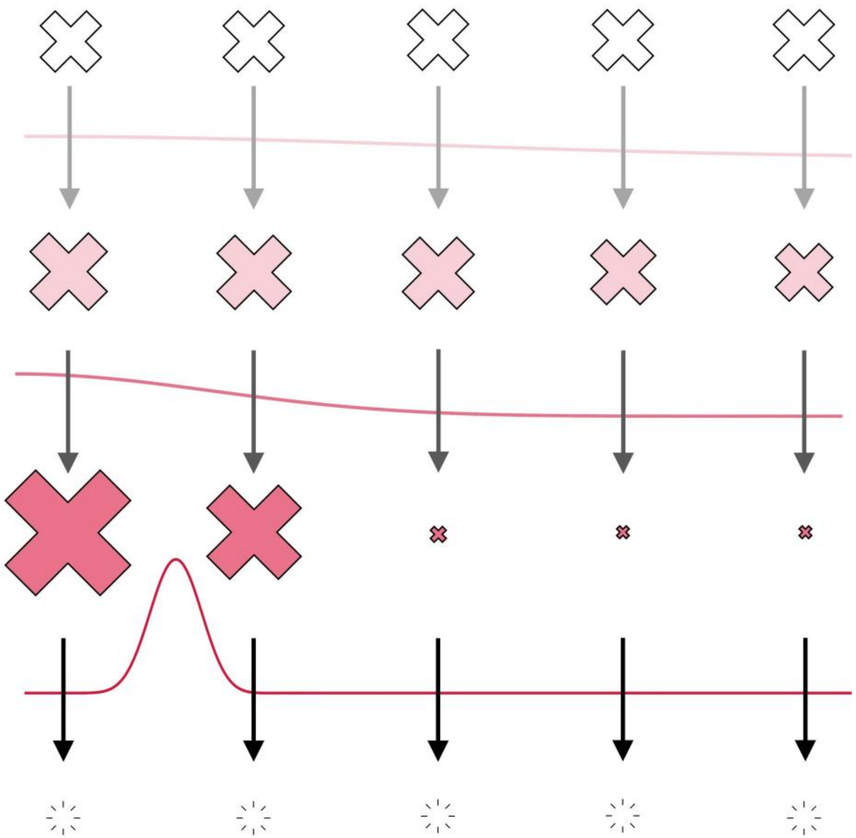
Offline vs. online inference

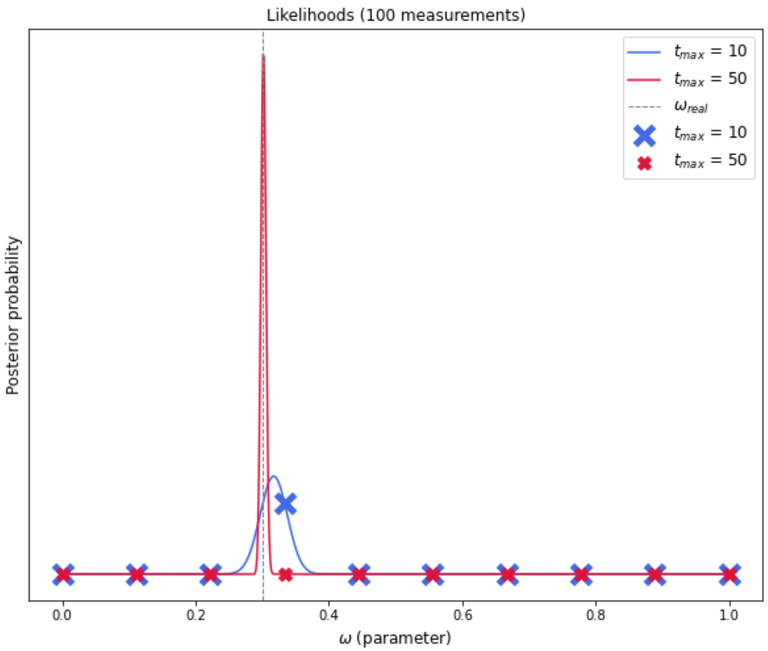


SMC

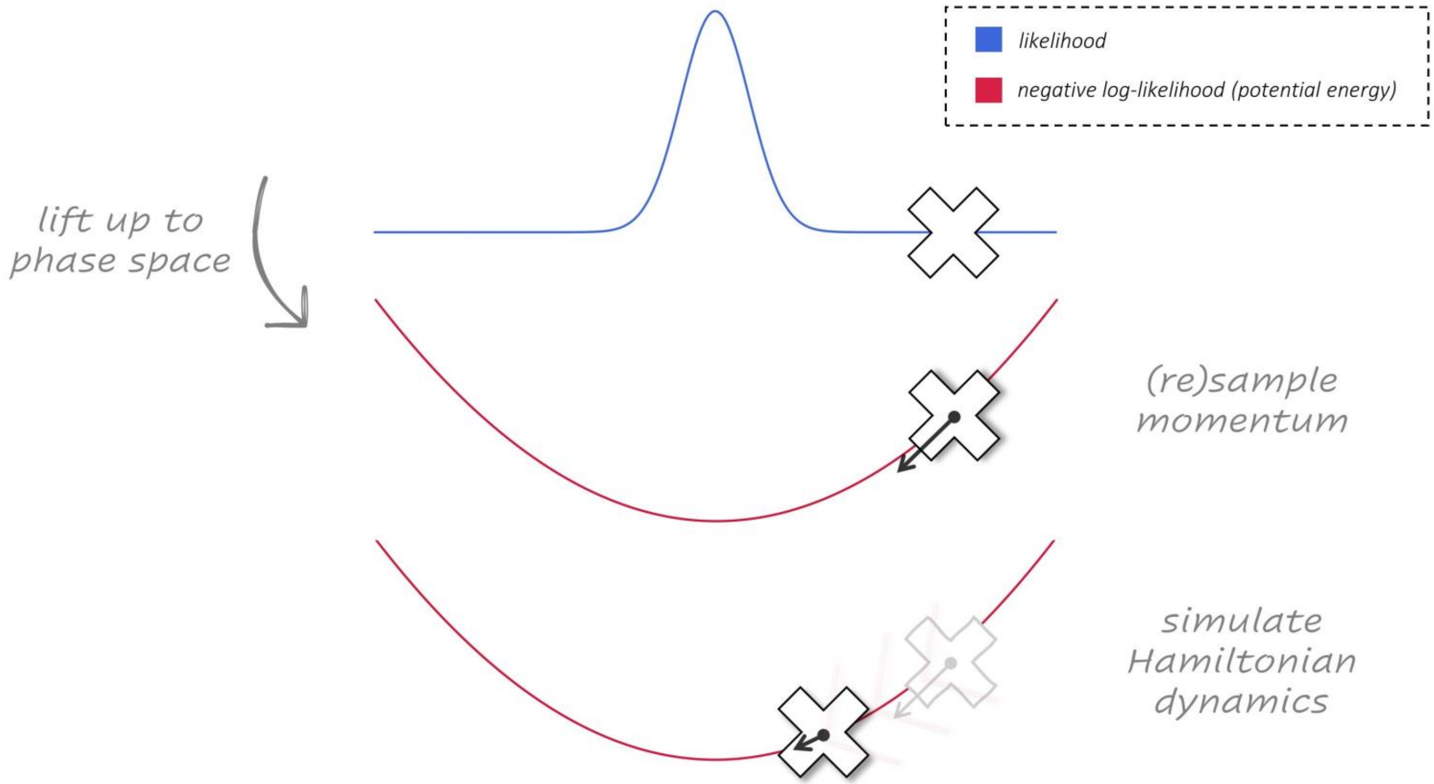


Particle degeneracy in SMC



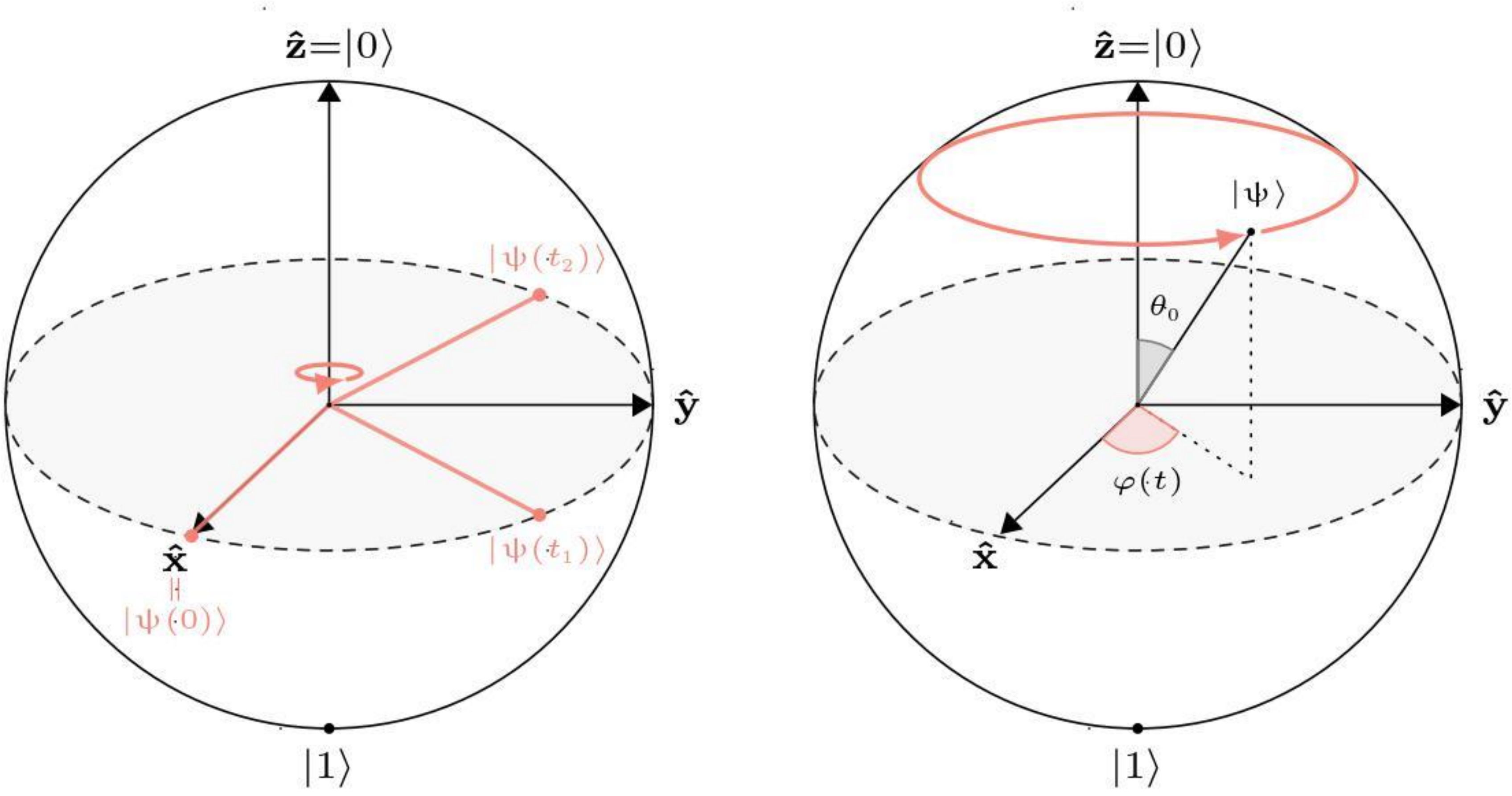


Hamiltonian Monte Carlo

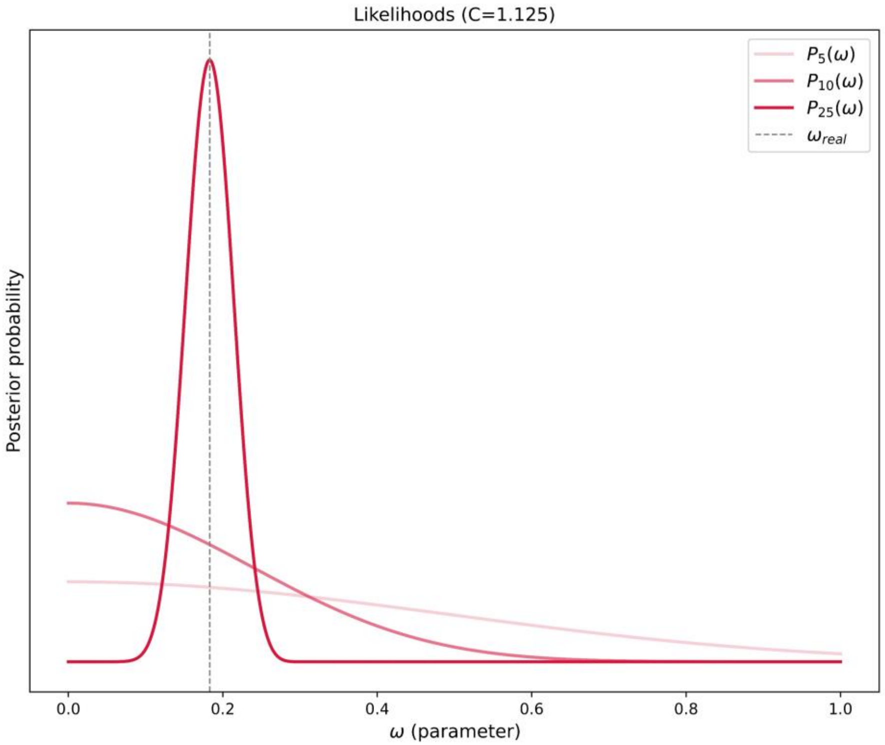


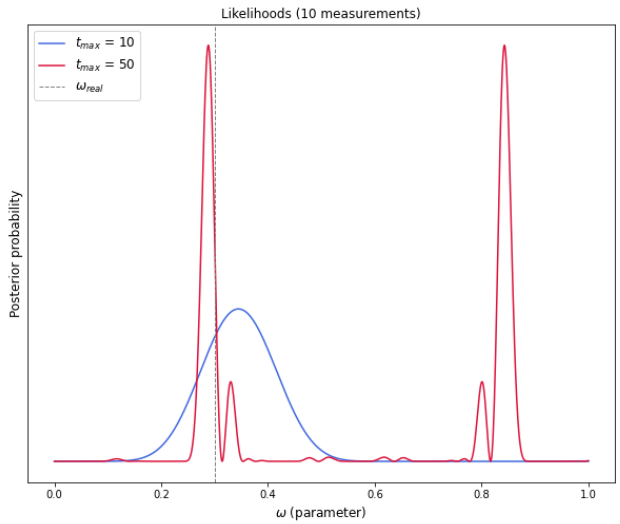
Precession

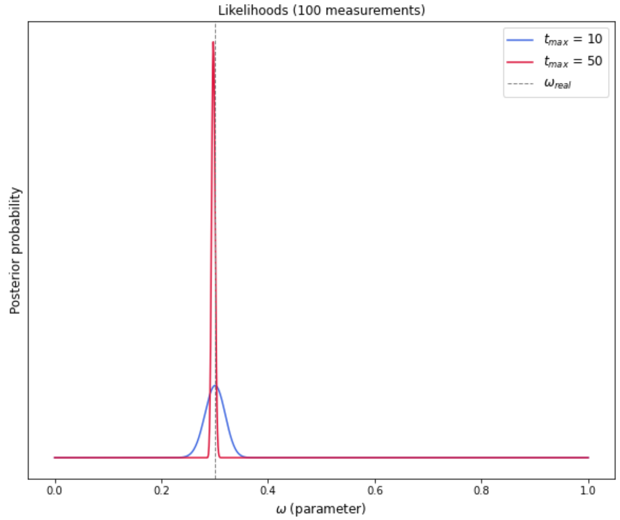
Bloch sphere representation



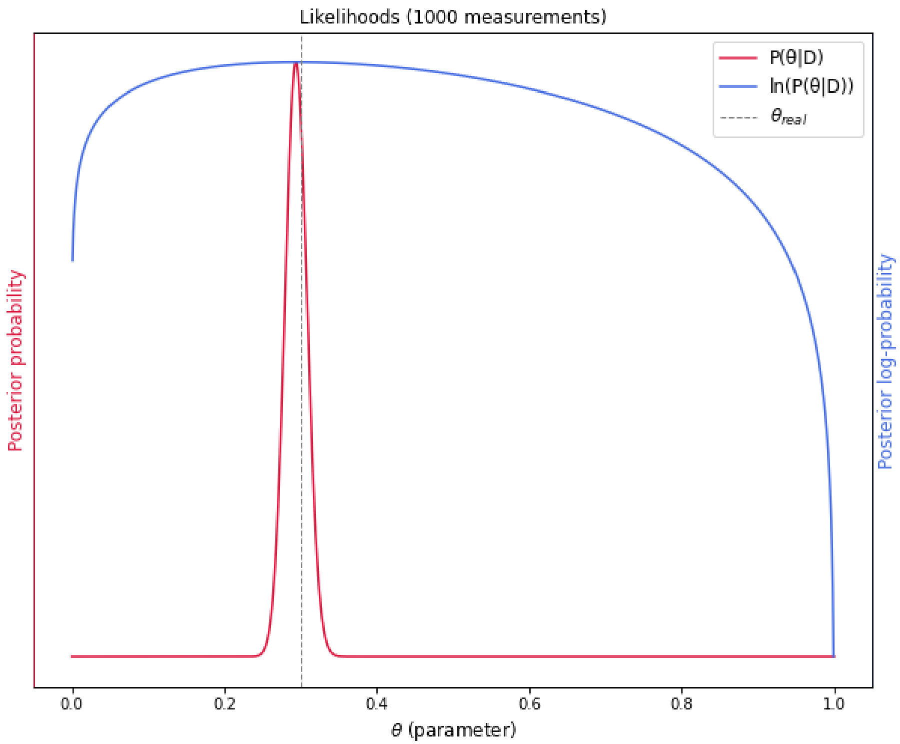
Impact of the evolution times



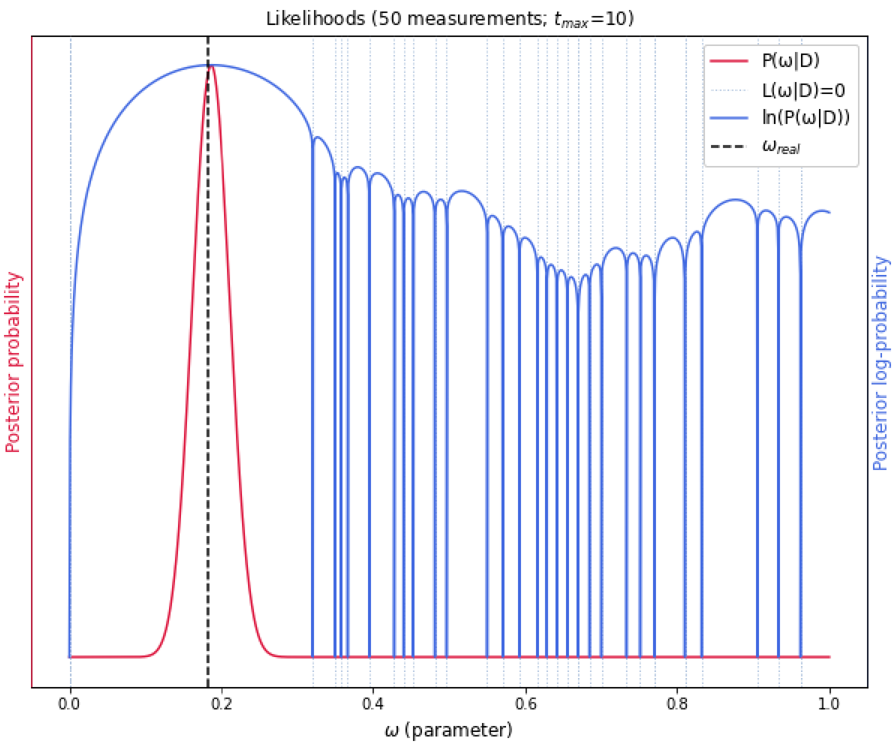




Log-density



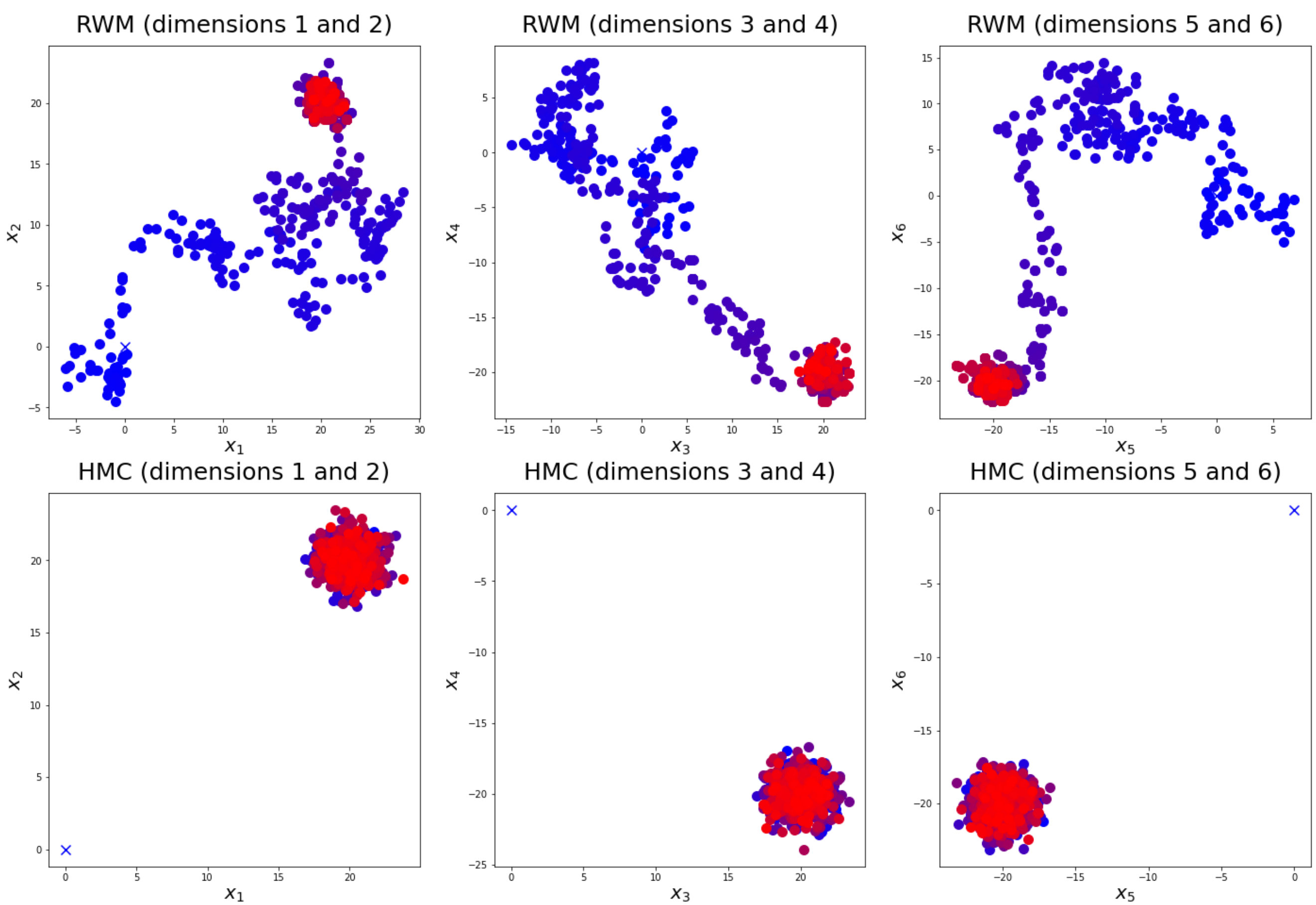
Likelihoods (1000 measurements)
P(θ|D)
ln(P(θ|D))
θ_{real}
Posterior probability
Posterior log-probability
0.0
0.2
0.4
0.6
0.8
1.0
θ (parameter)



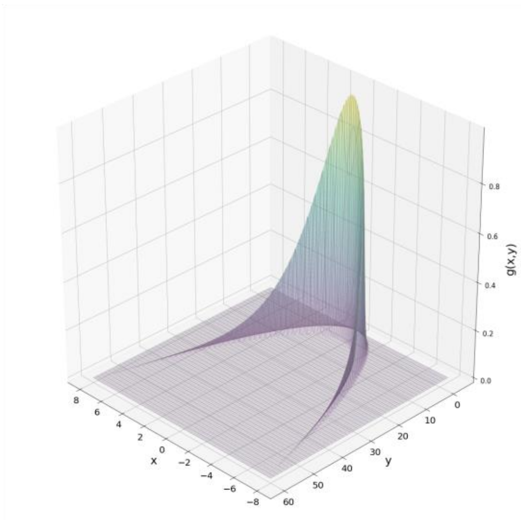
Likelihoods (50 measurements; t_{max}=10)
P(ω|D)
L(ω|D)=0
ln(P(ω|D))
ω_{real}
Posterior probability
Posterior log-probability
0.0
0.2
0.4
0.6
0.8
1.0
ω (parameter)

HMC/SHMC/NUTS vs. RWM/GRF/LW

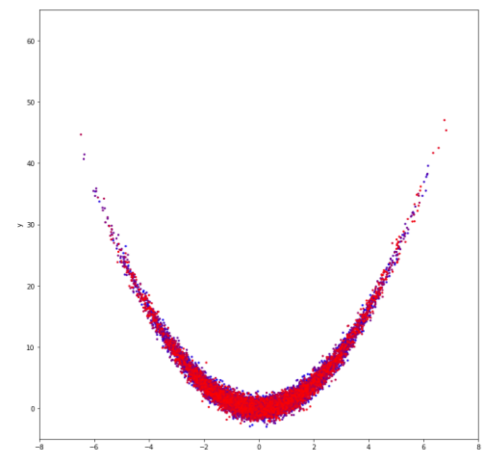
Hamiltonian Monte Carlo



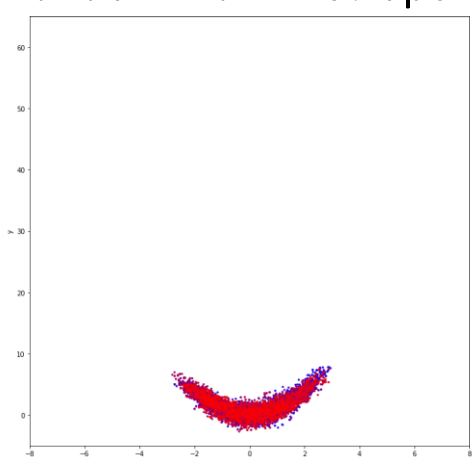
Hamiltonian Monte Carlo



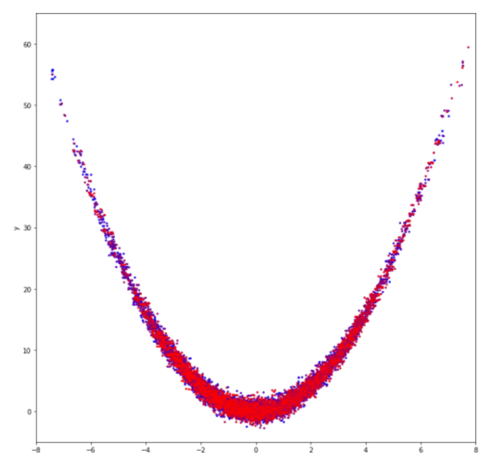
HMC



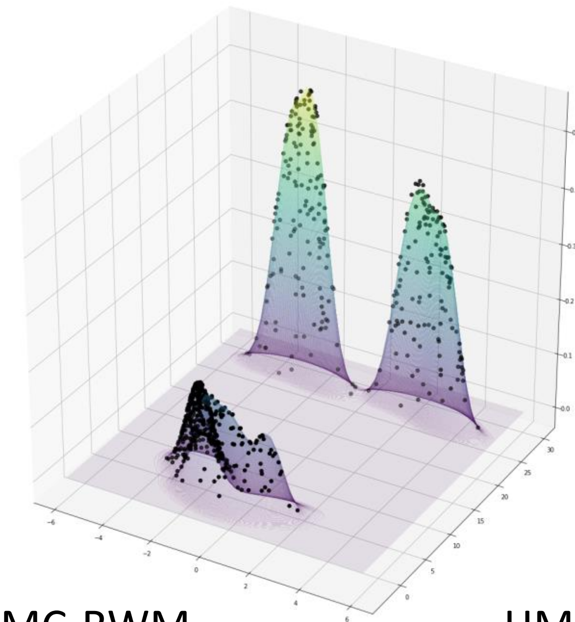
Random walk Metropolis



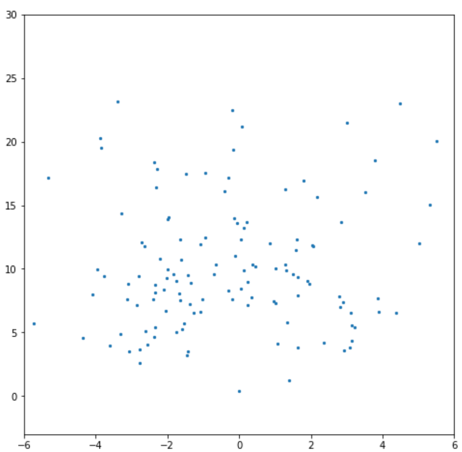
NUTS



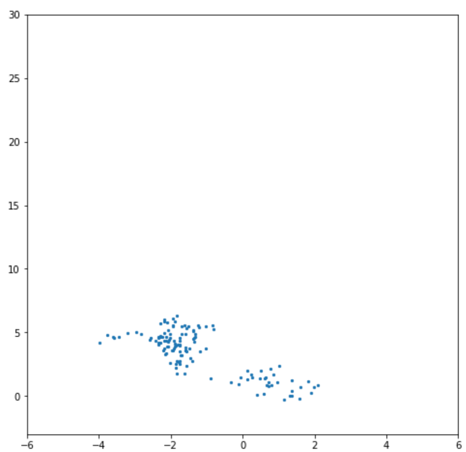
Hamiltonian Monte Carlo



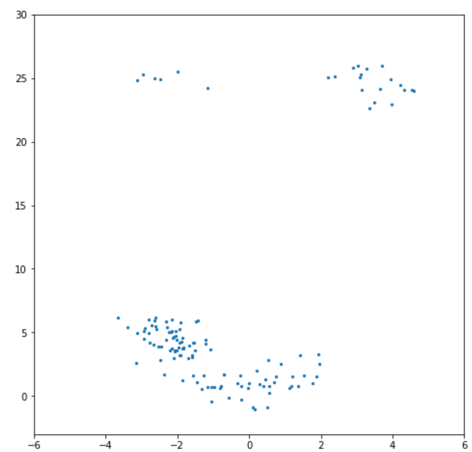
RWM



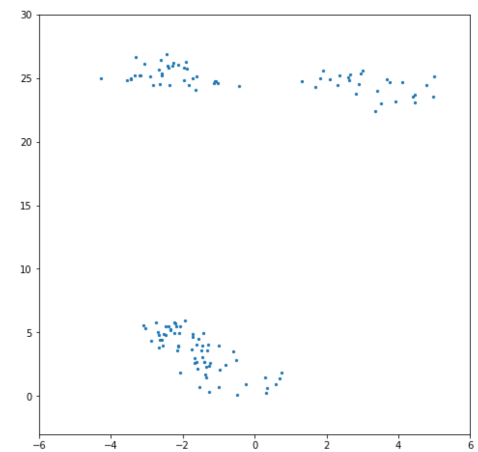
SMC-RWM



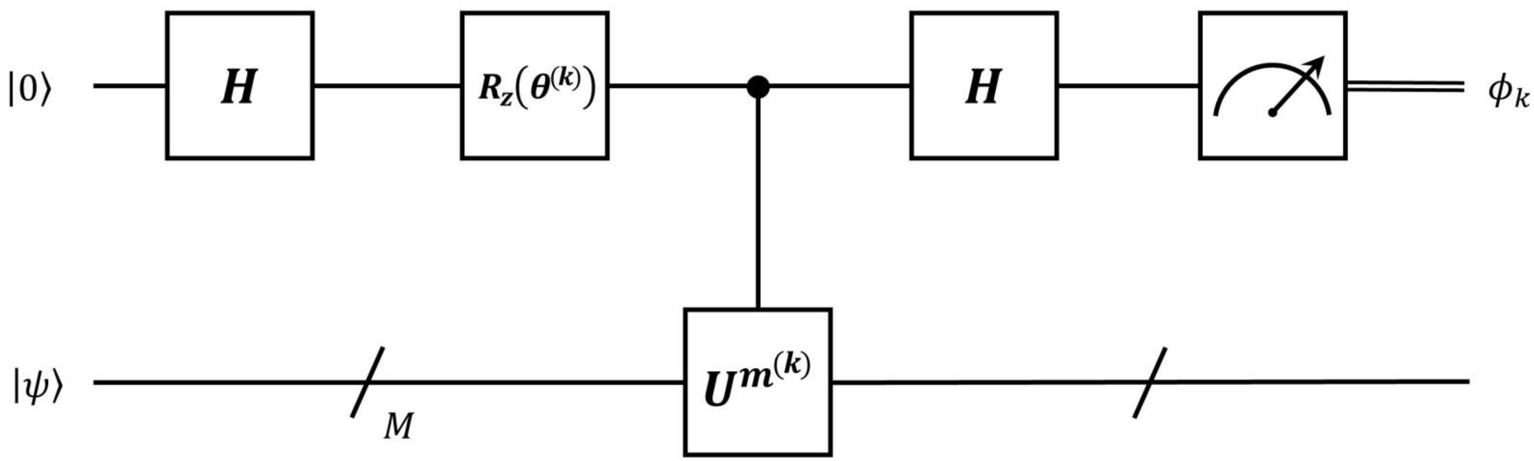
HMC



SHMC

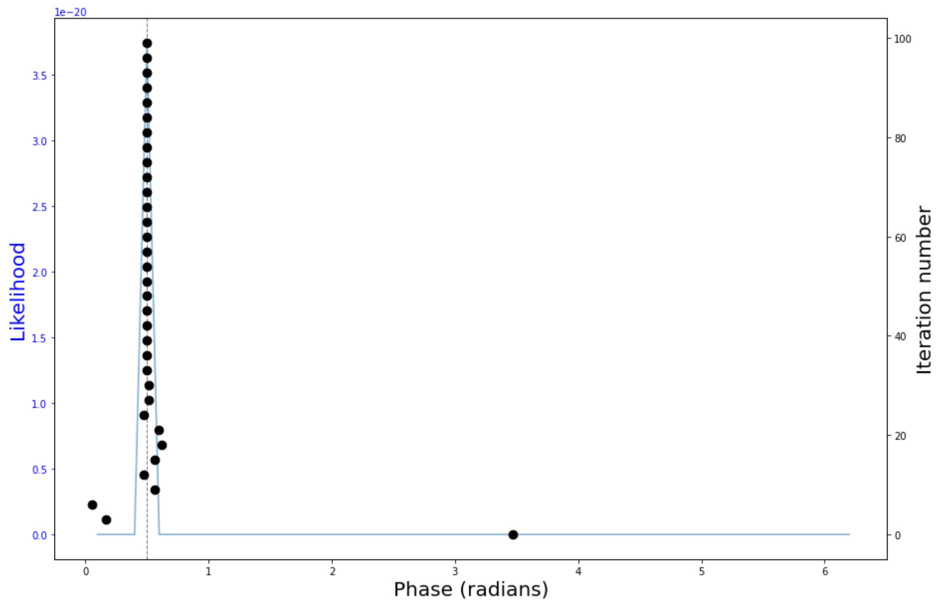


Bayesian phase estimation

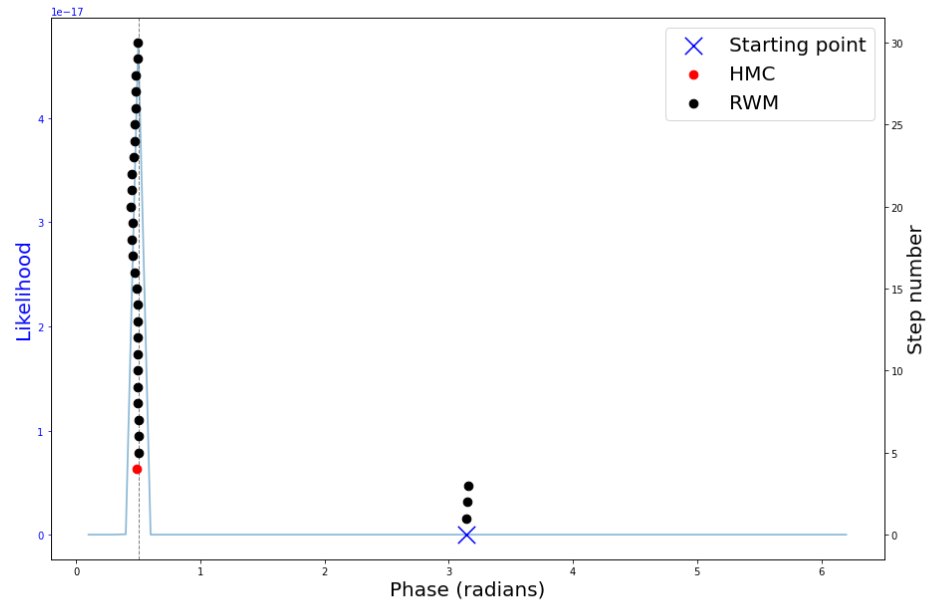


Bayesian phase estimation

Gaussian rejection sampling

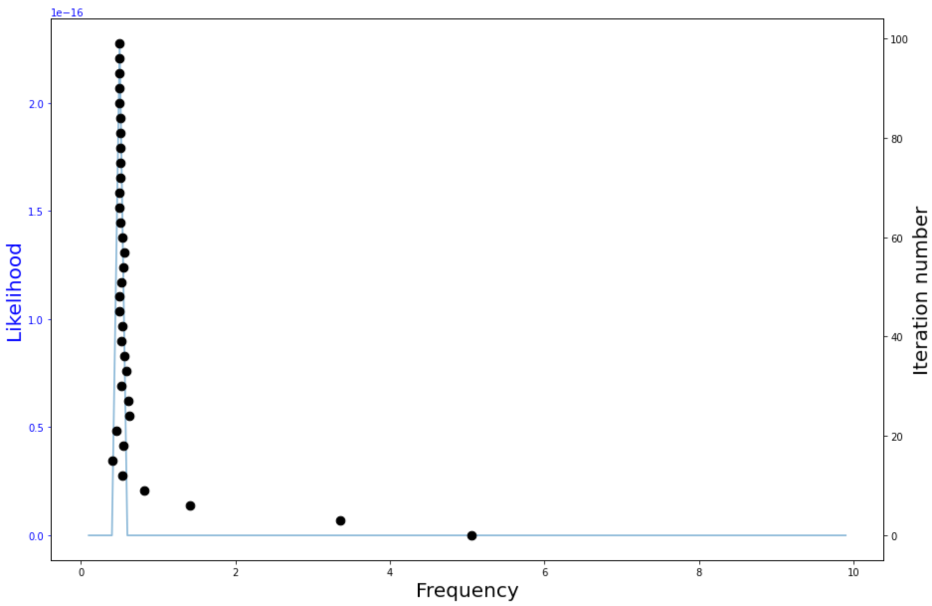


MCMC

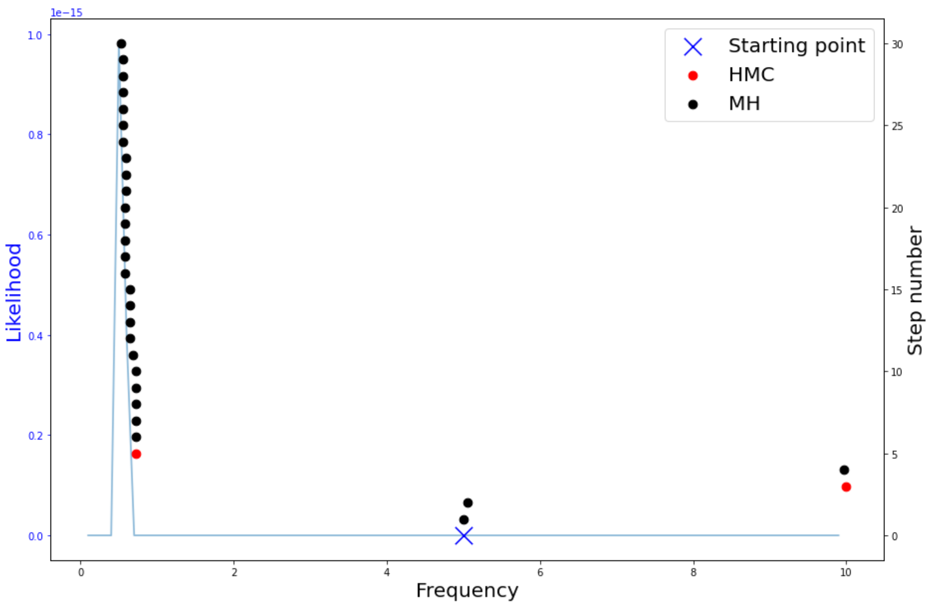


Bayesian phase estimation

SMC-LW



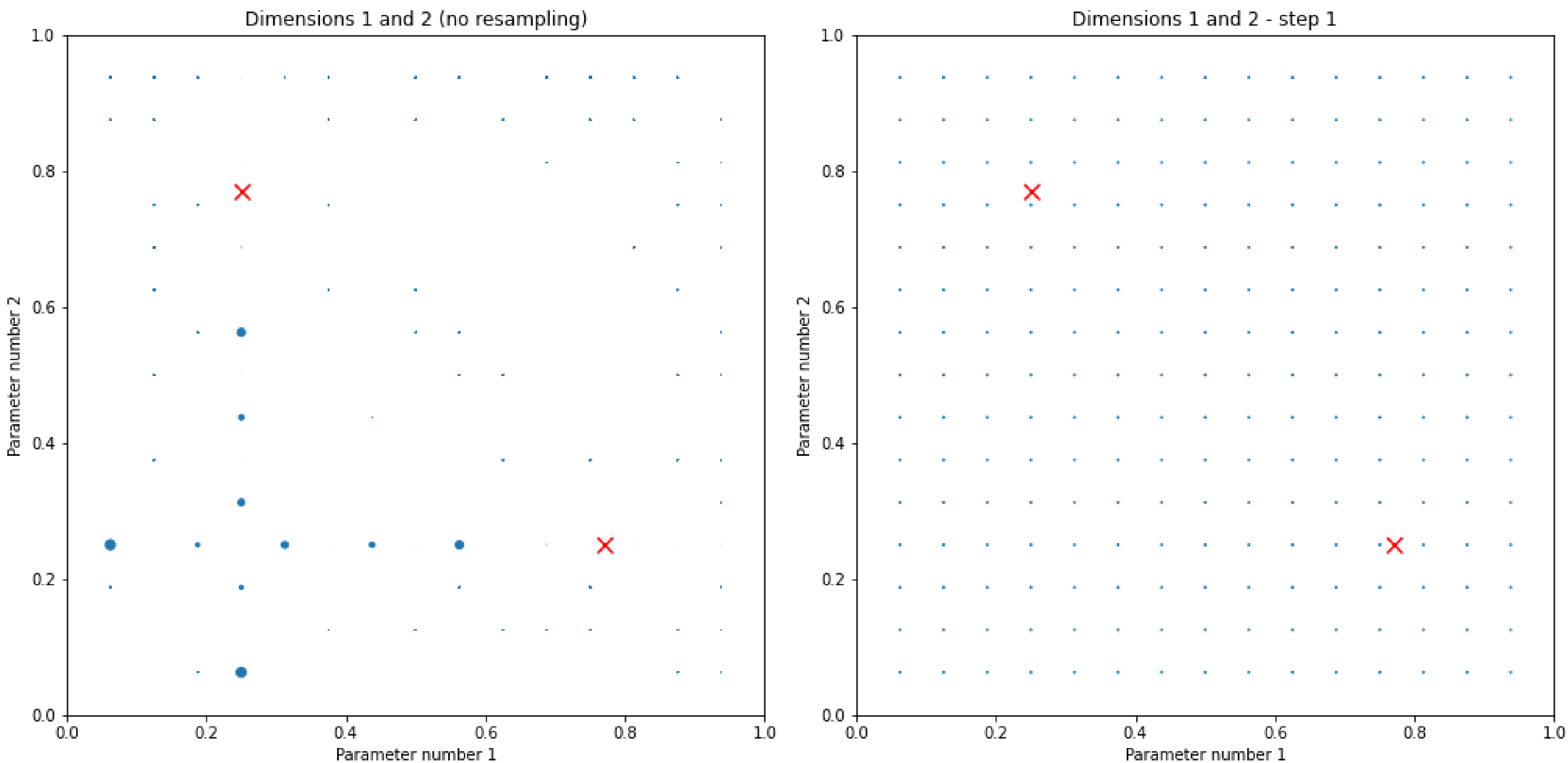
MCMC



Step by step tempered estimation

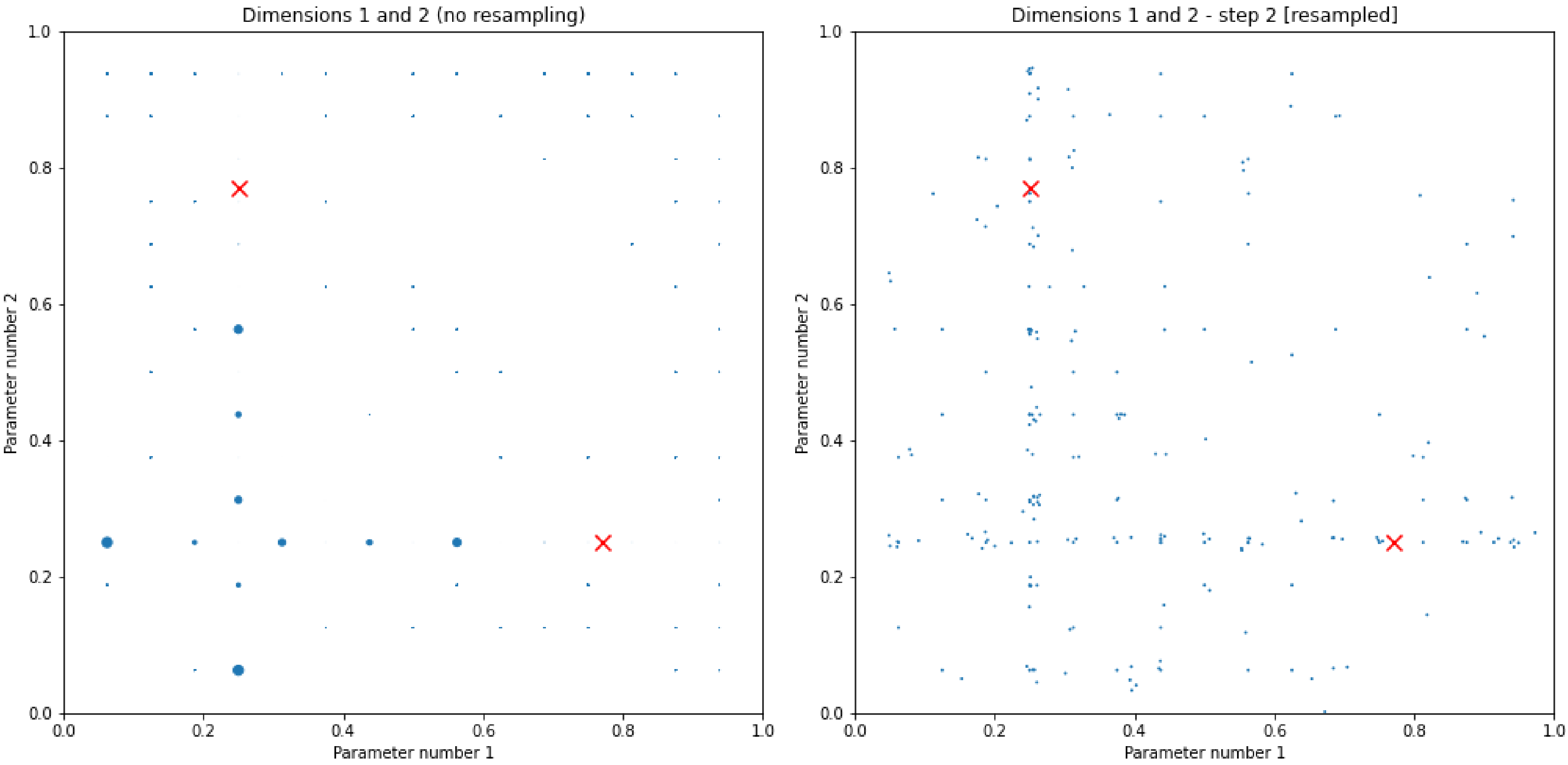
2d tempered estimation step by step

with early redundancy and particle displacement



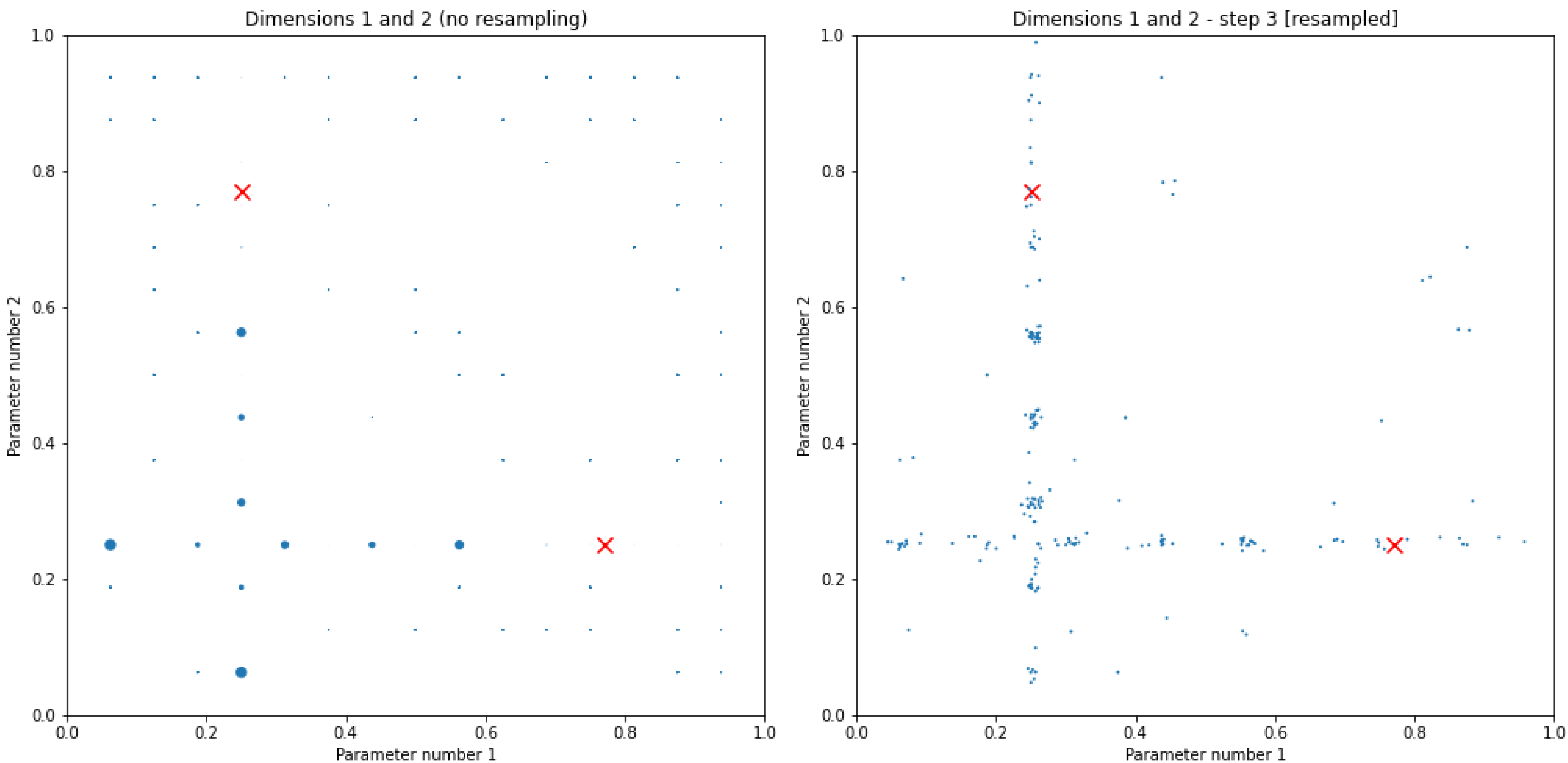
2d tempered estimation step by step

with early redundancy and particle displacement



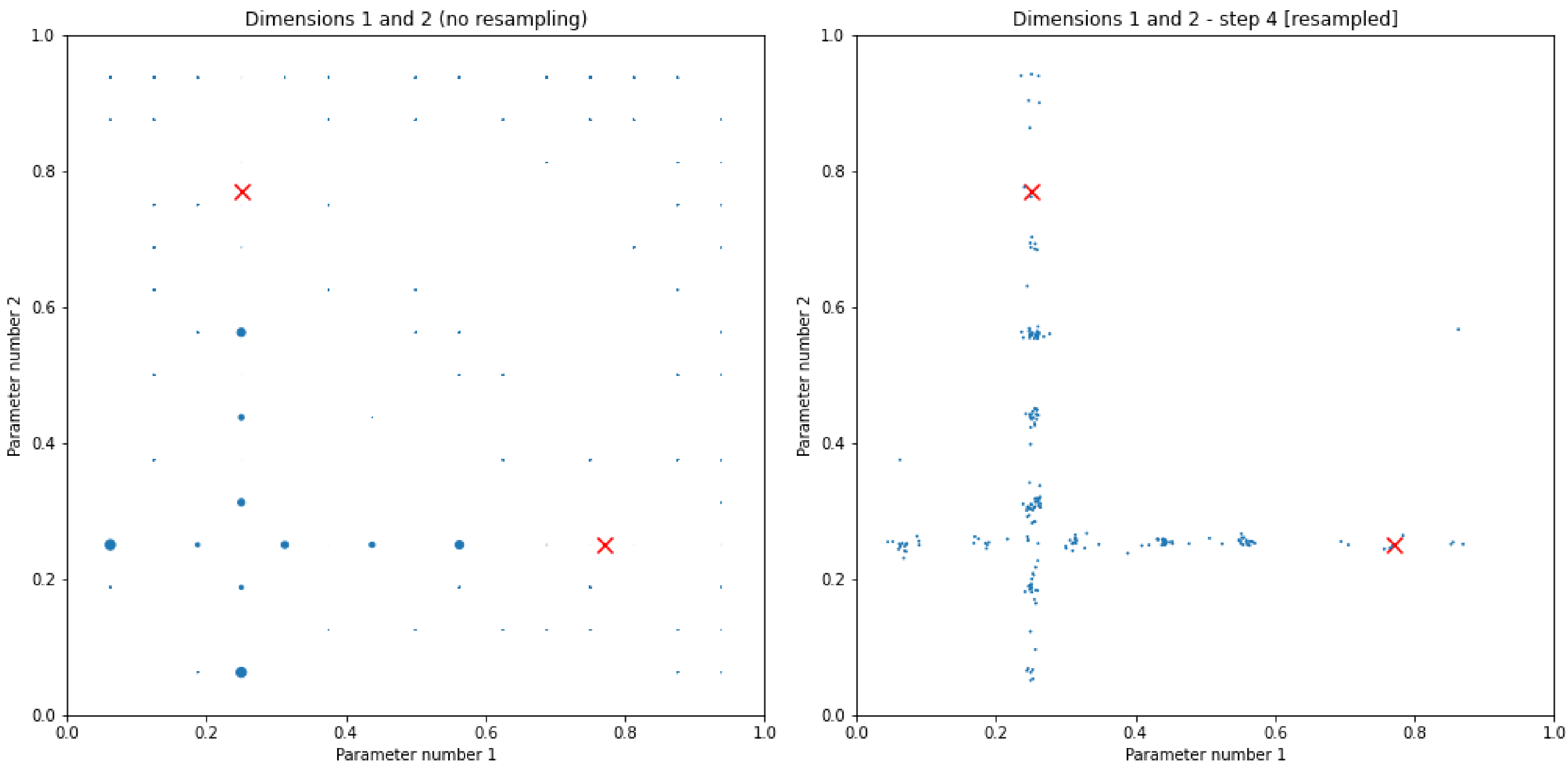
2d tempered estimation step by step

with early redundancy and particle displacement



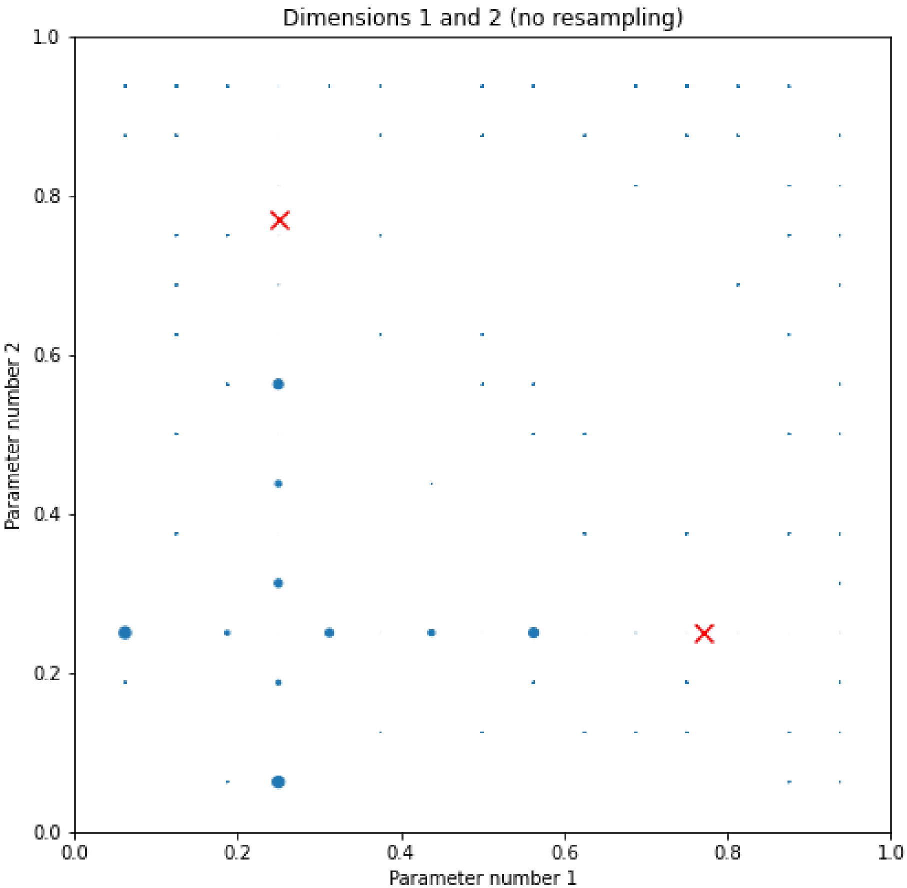
2d tempered estimation step by step

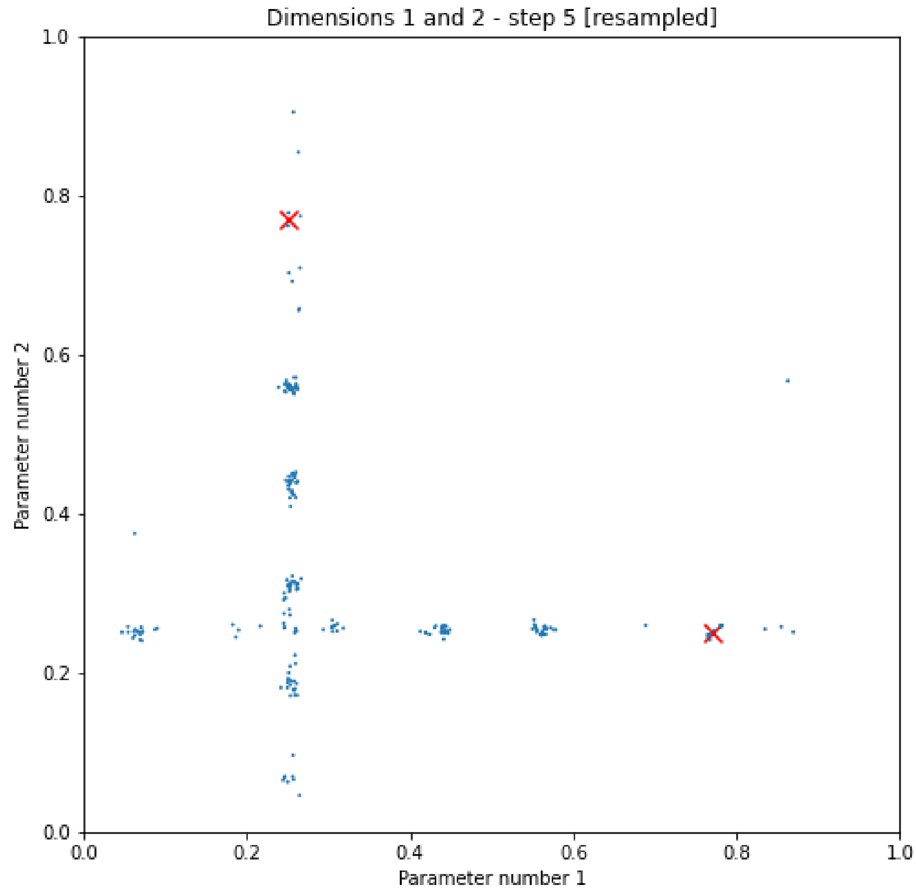
with early redundancy and particle displacement



2d tempered estimation step by step

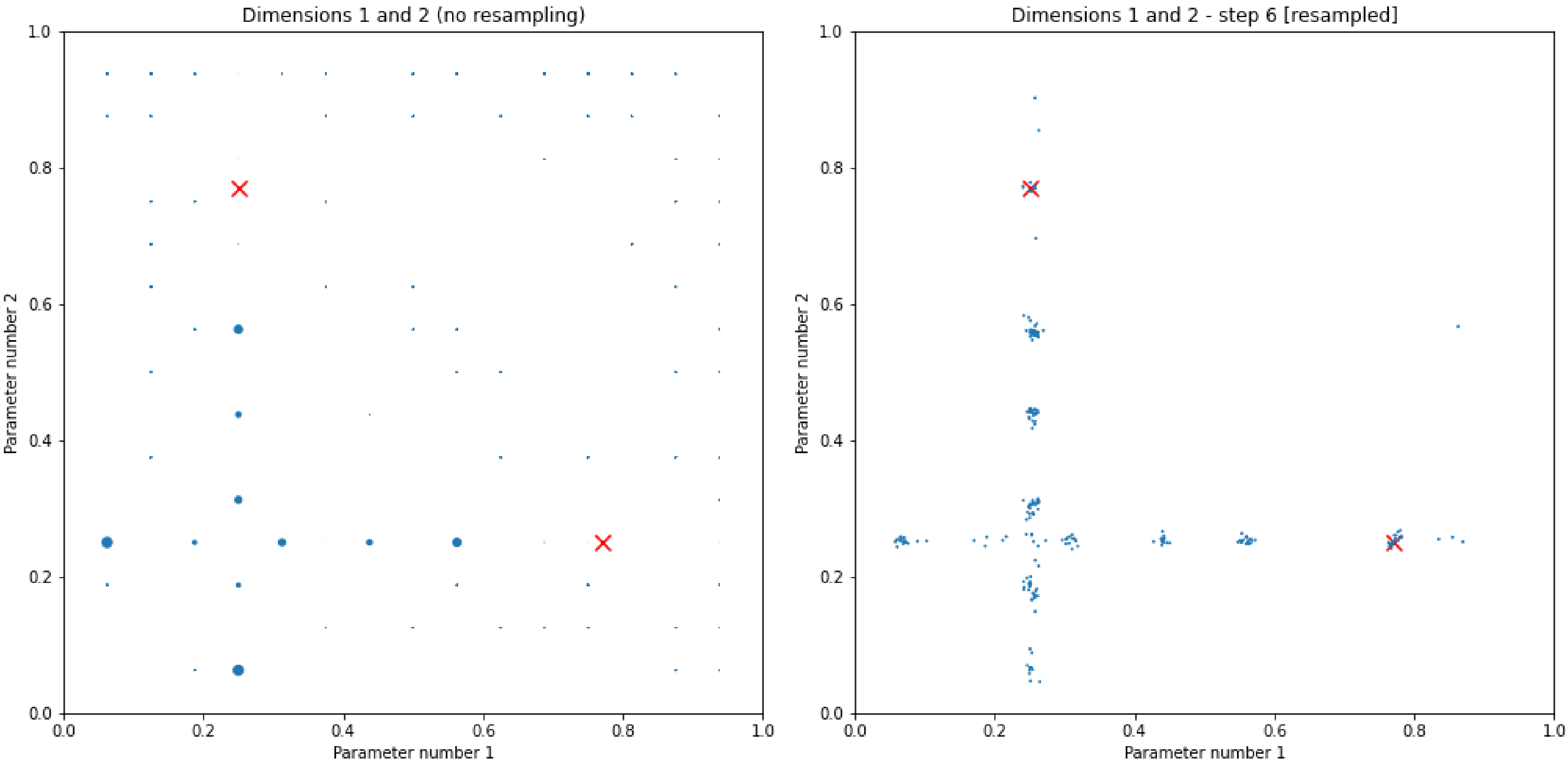
with early redundancy and particle displacement



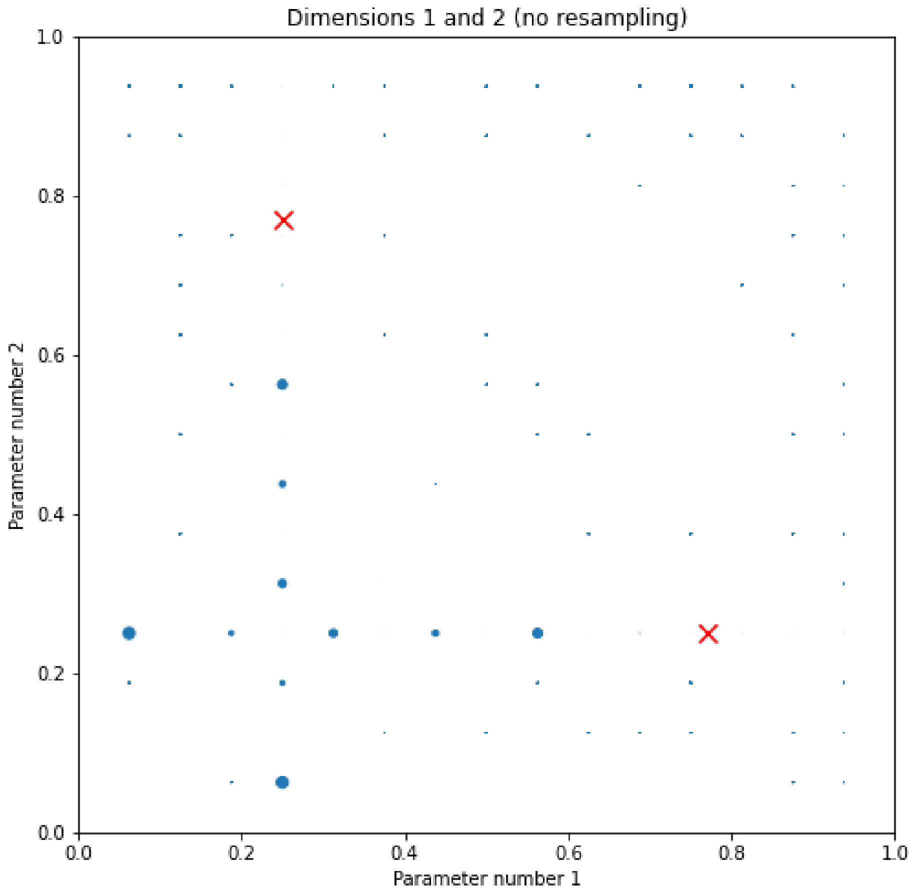


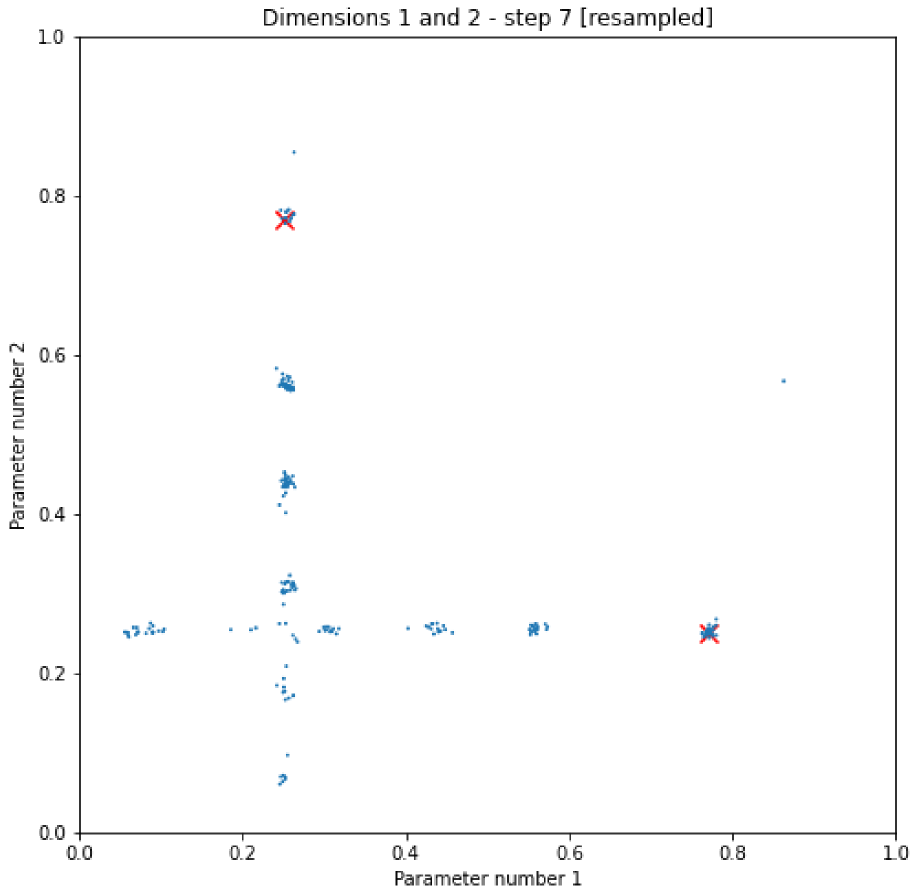
2d tempered estimation step by step

with early redundancy and particle displacement



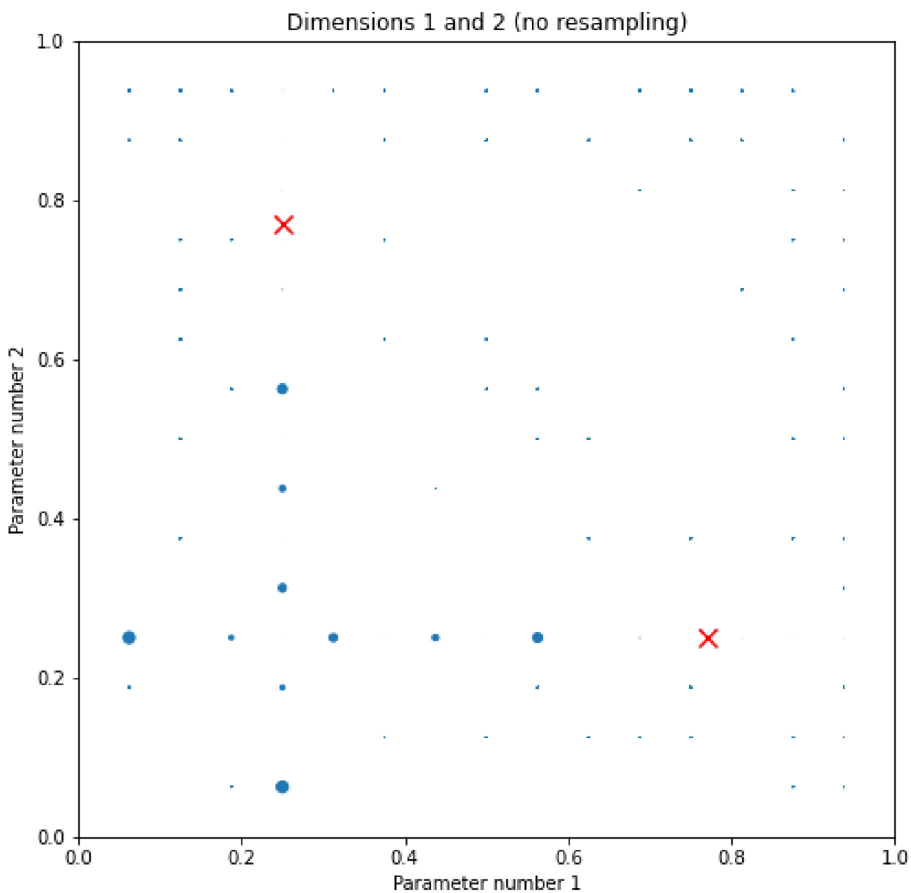
2d tempered estimation step by step

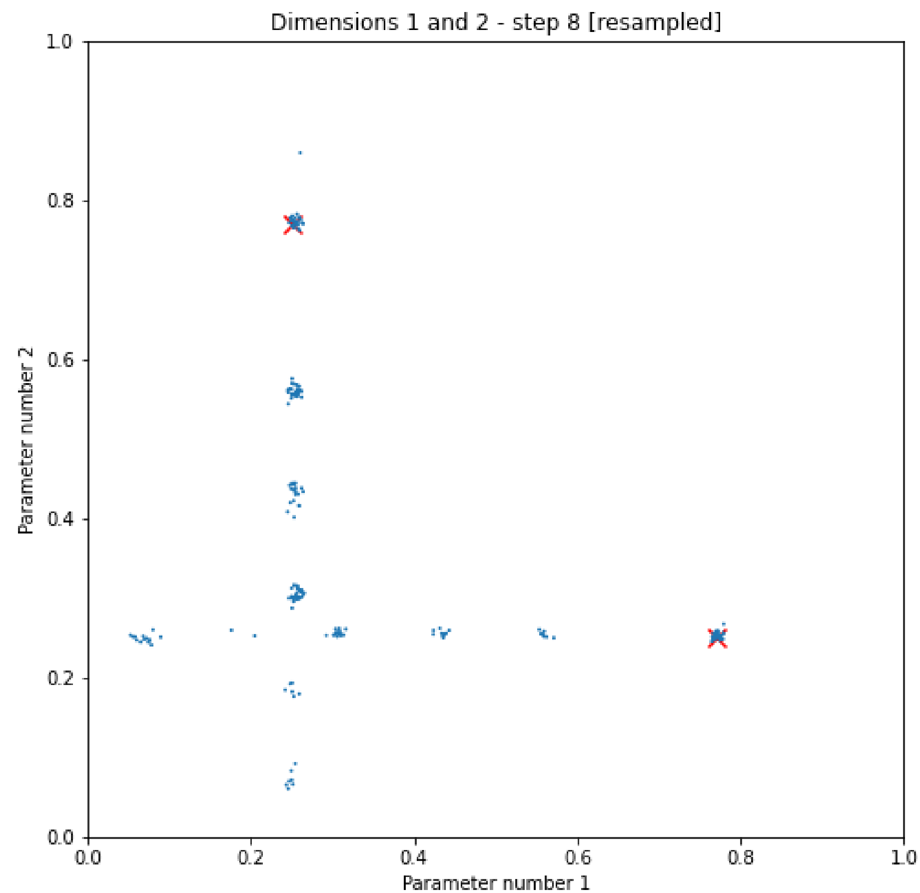




2d tempered estimation step by step

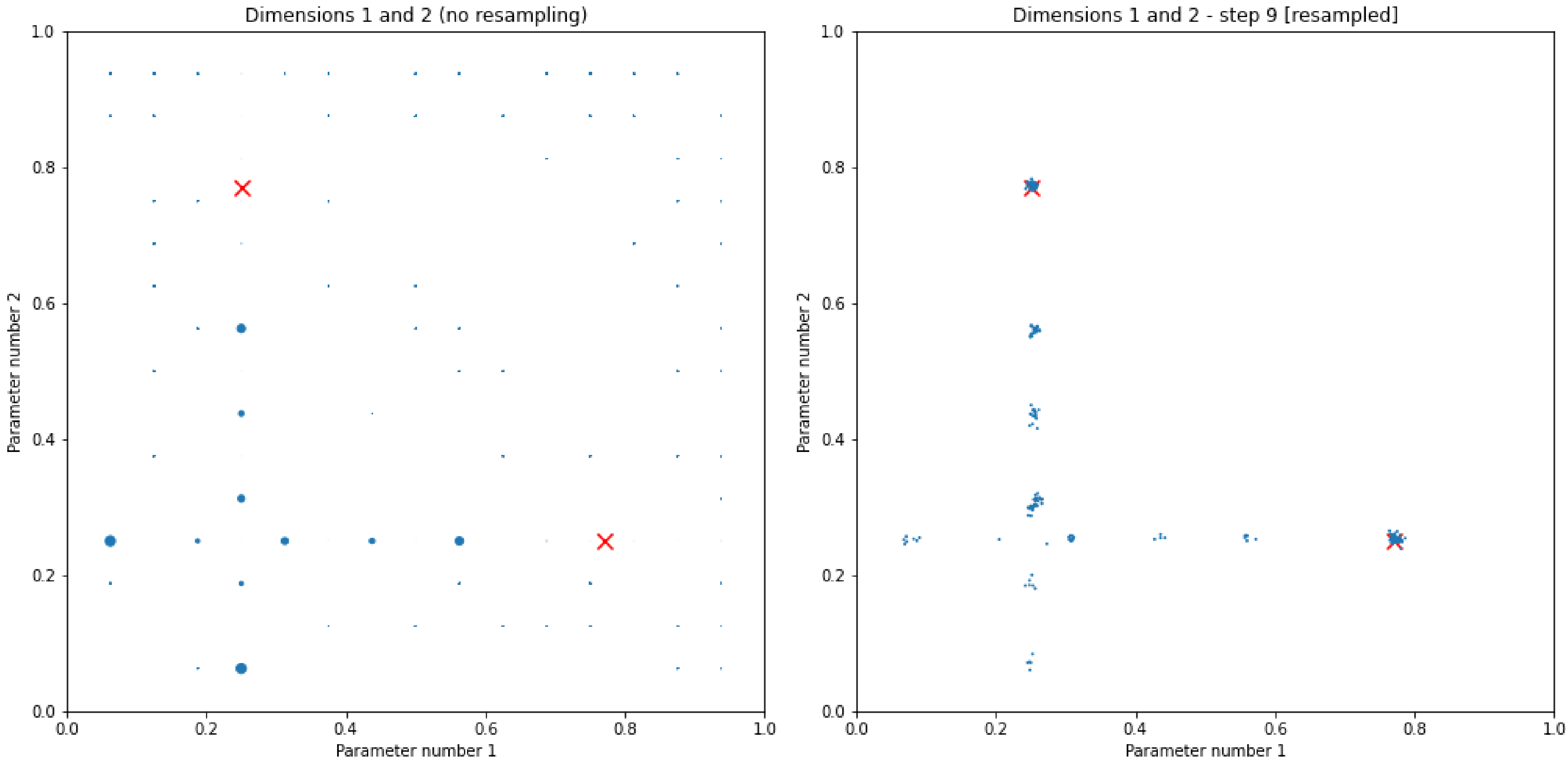
with early redundancy and particle displacement





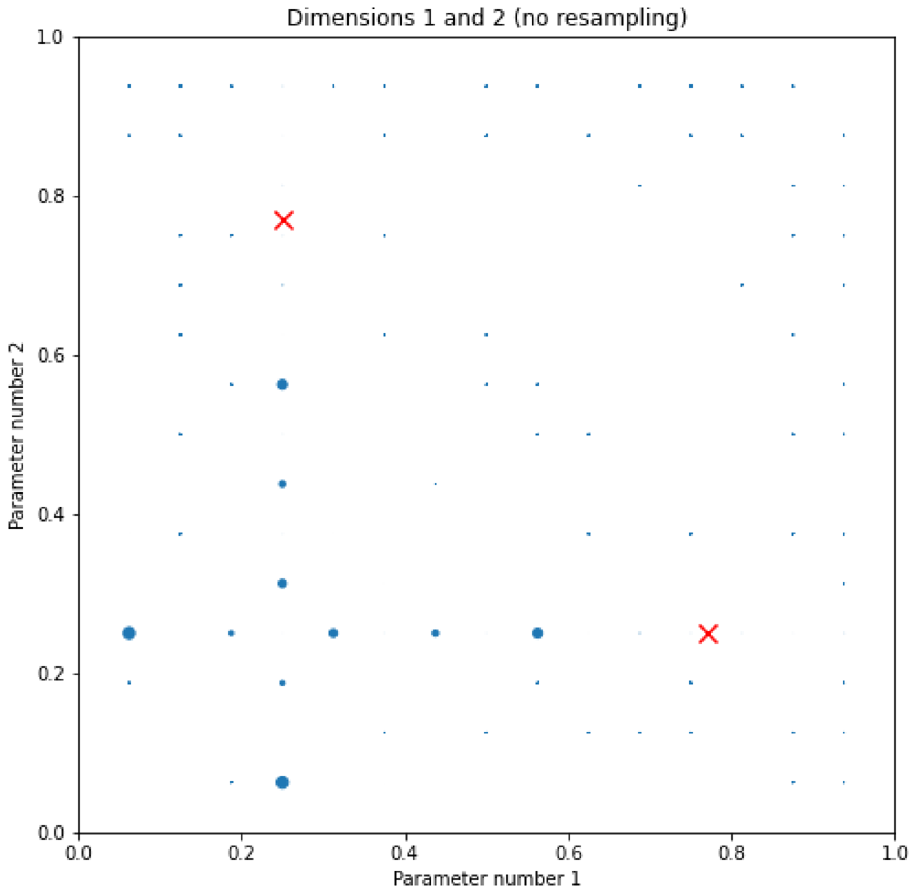
2d tempered estimation step by step

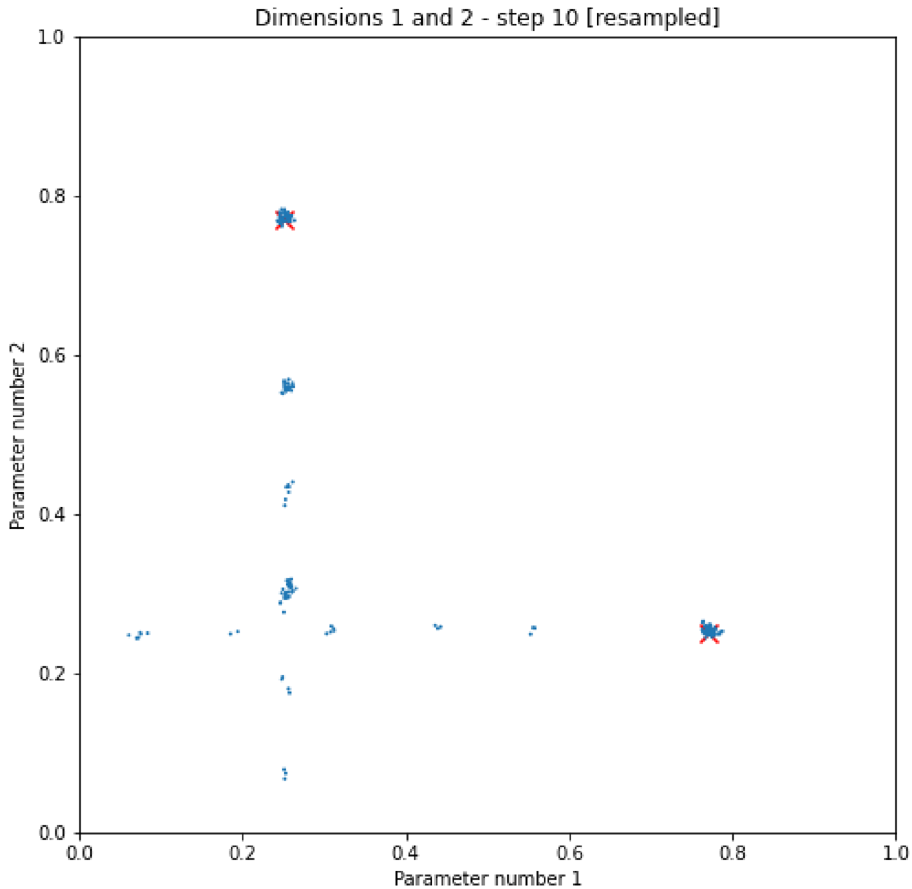
with early redundancy and particle displacement



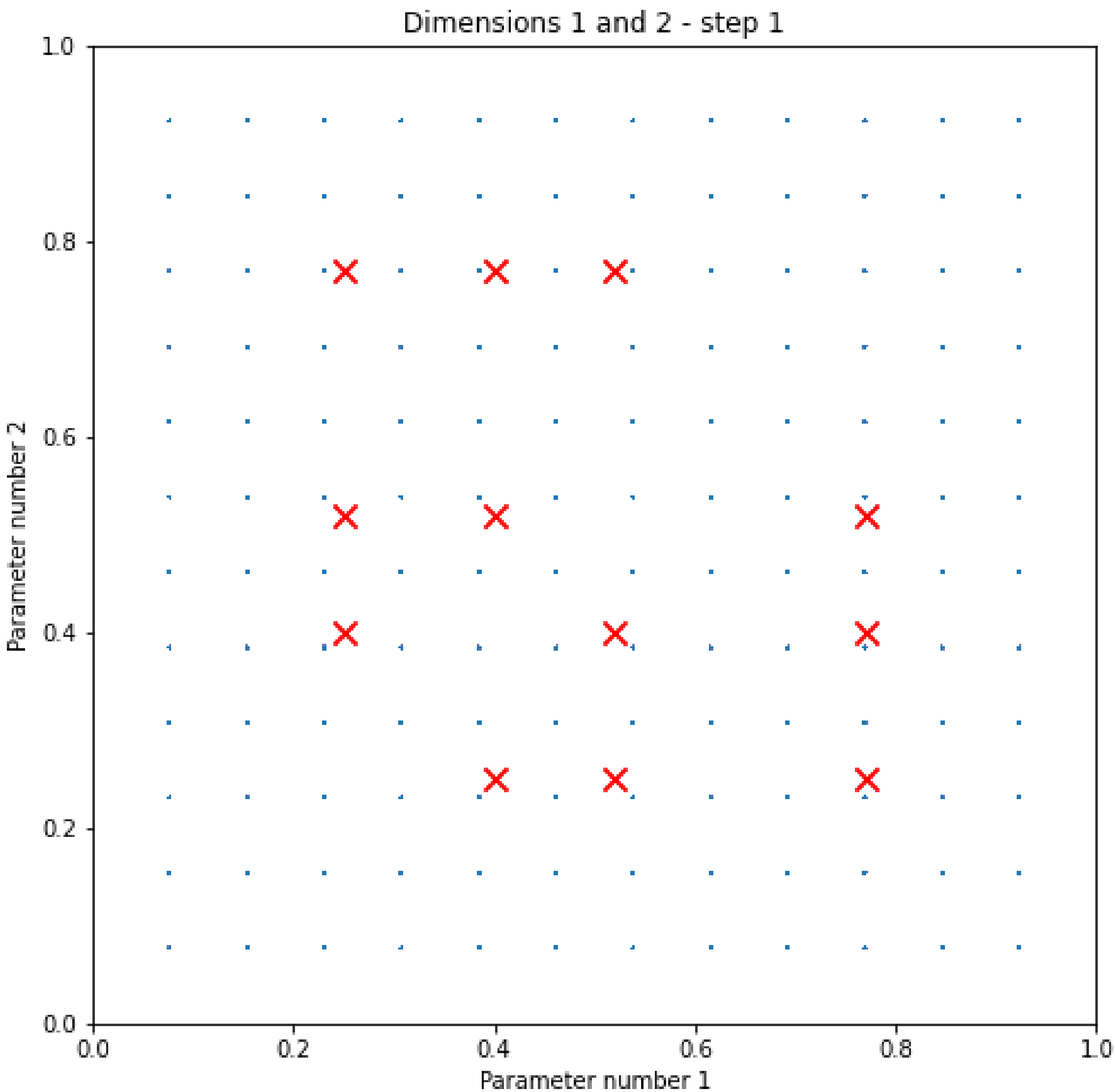
2d tempered estimation step by step

with early redundancy and particle displacement

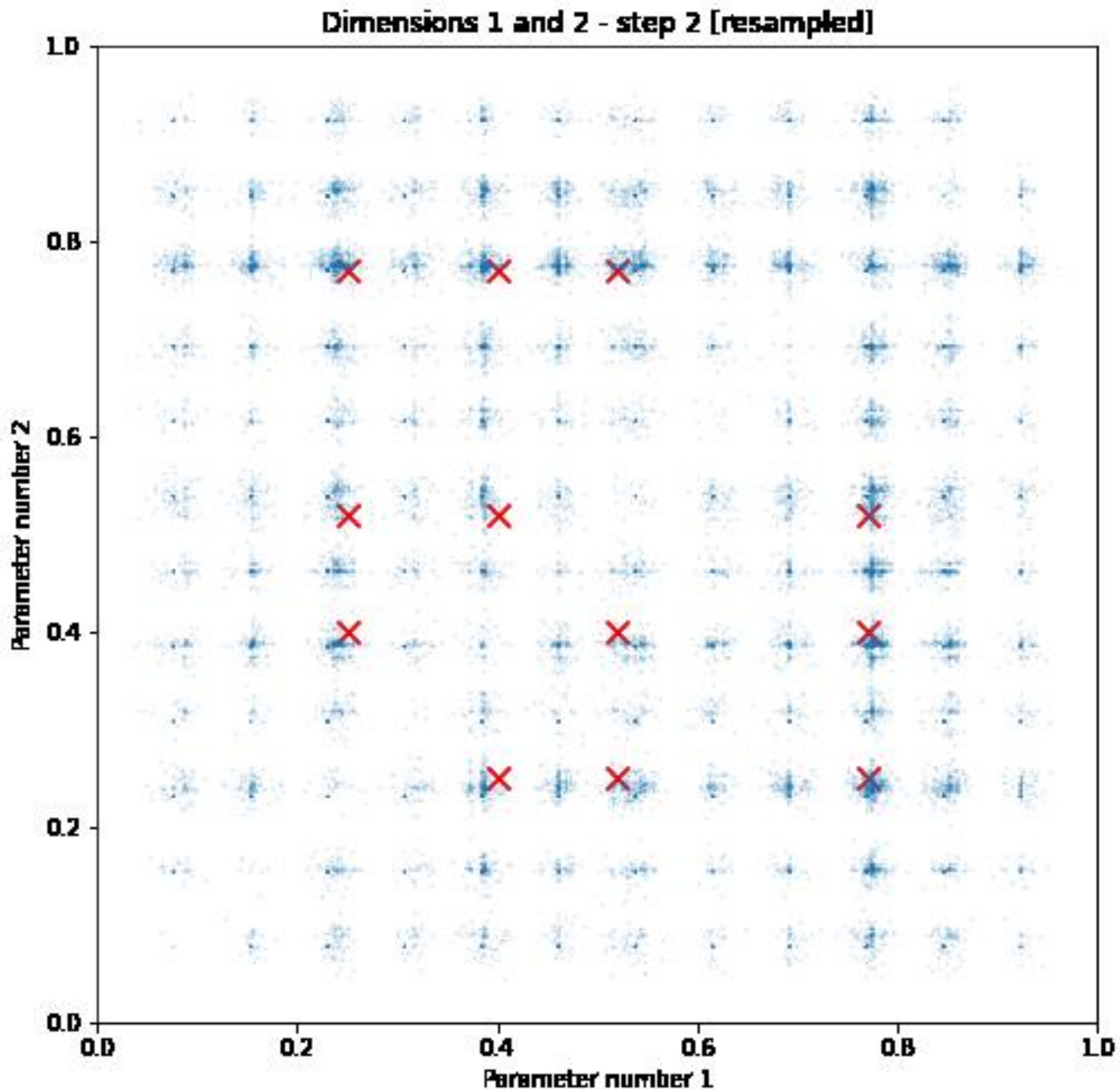




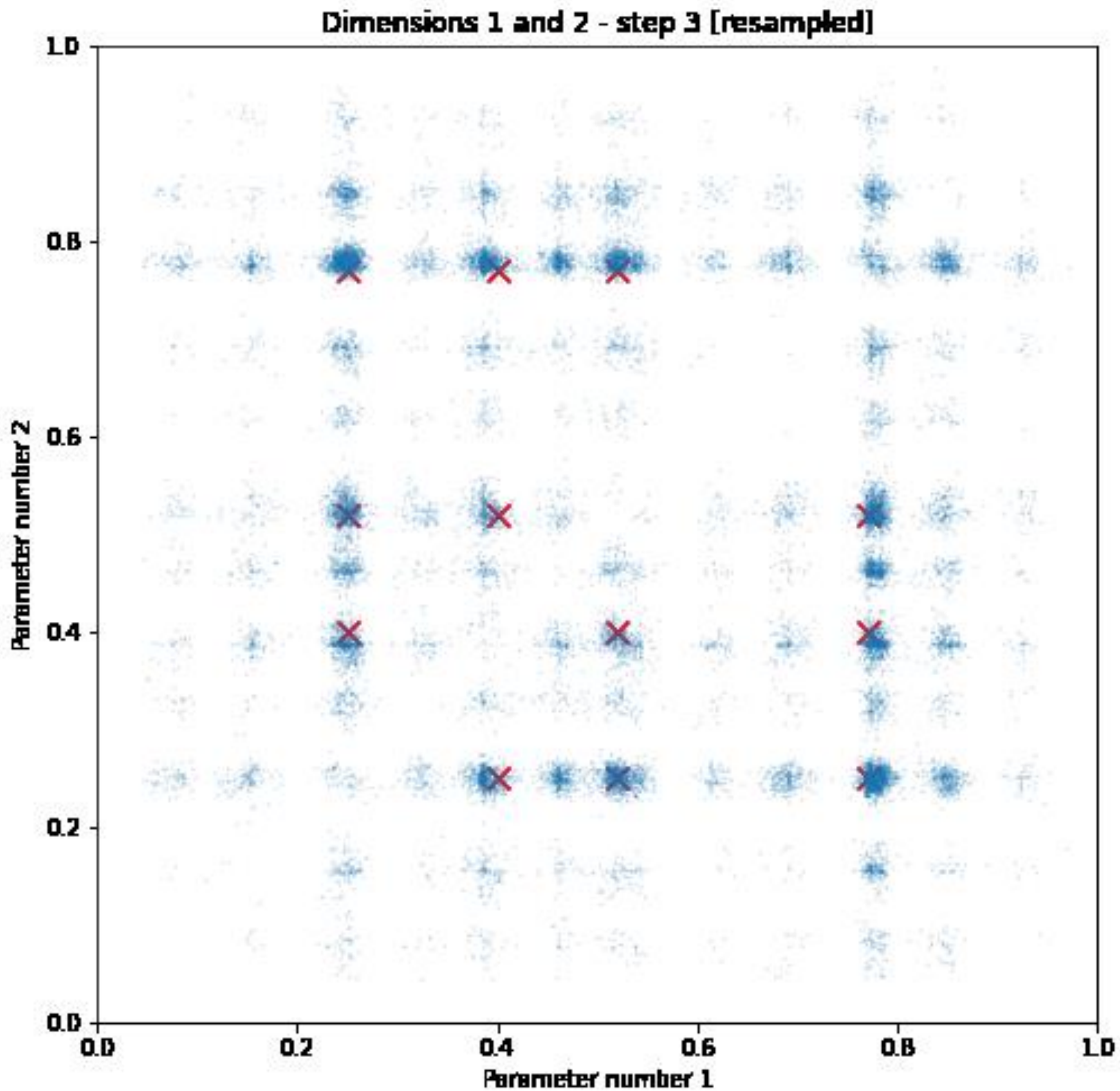
4d tempered estimation step by step



4d tempered estimation step by step



4d tempered estimation step by step



4d tempered estimation step by step