Title: Rejection and Particle Filtering for Hamiltonian Learning

Date: Aug 08, 2016 11:45 AM

URL: http://pirsa.org/16080003

Abstract: Many tasks in quantum information rely on accurate knowledge of a system's Hamiltonian, including calibrating control, characterizing devices, and verifying quantum simulators. In this talk, we pose the problem of learning Hamiltonians as an instance of parameter estimation. We then solve this problem with Bayesian inference, and describe how rejection and particle filtering provide efficient numerical algorithms for learning Hamiltonians. Finally, we discuss how filtering can be combined with quantum resources to verify quantum systems beyond the reach of classical simulators.

Rejection and Particle Filtering for Hamiltonian Learning

Christopher E. Granade
Centre for Engineered Quantum Systems
University of Sydney

www.cgranade.com/research/talks/qml/2016

What is a Hamiltonian?

Lots of meanings and applications in condensed matter, quantum info, etc. In this talk, we consider precisely one:

What is a Hamiltonian?

Lots of meanings and applications in condensed matter, quantum info, etc. In this talk, we consider precisely one:

A Hamiltonian *generates dynamics*.

What is a Hamiltonian?

Lots of meanings and applications in condensed matter, quantum info, etc. In this talk, we consider precisely one:

$$|\psi(t)\rangle = \mathrm{e}^{\mathrm{i}Ht}|\psi(0)\rangle$$

A Hamiltonian *generates dynamics*.

Learning Hamiltonians is critical to a range of tasks:

Metrology
Learning magnetic fields, etc.

Calibration
Static field / pulse power / crosstalk, etc.

Debugging/Diagnosis
T_2 estimation, other noise finding

Verification/Validation
Analog and digital quantum simulation

Example: Ramsey Estimation

Suppose $H = \omega\sigma_z/2$ for some unknown ω.

Traditional approach:

- Prepare $|+\rangle \propto |0\rangle + |1\rangle$, measure "click" w/ pr.: $|\langle +|e^{i\omega t\sigma_z/2}|+\rangle|^2 = \cos^2(\omega t/2)$.
- Repeat for many "shots" to estimate click pr.
- Repeat for many times to estimate signal.

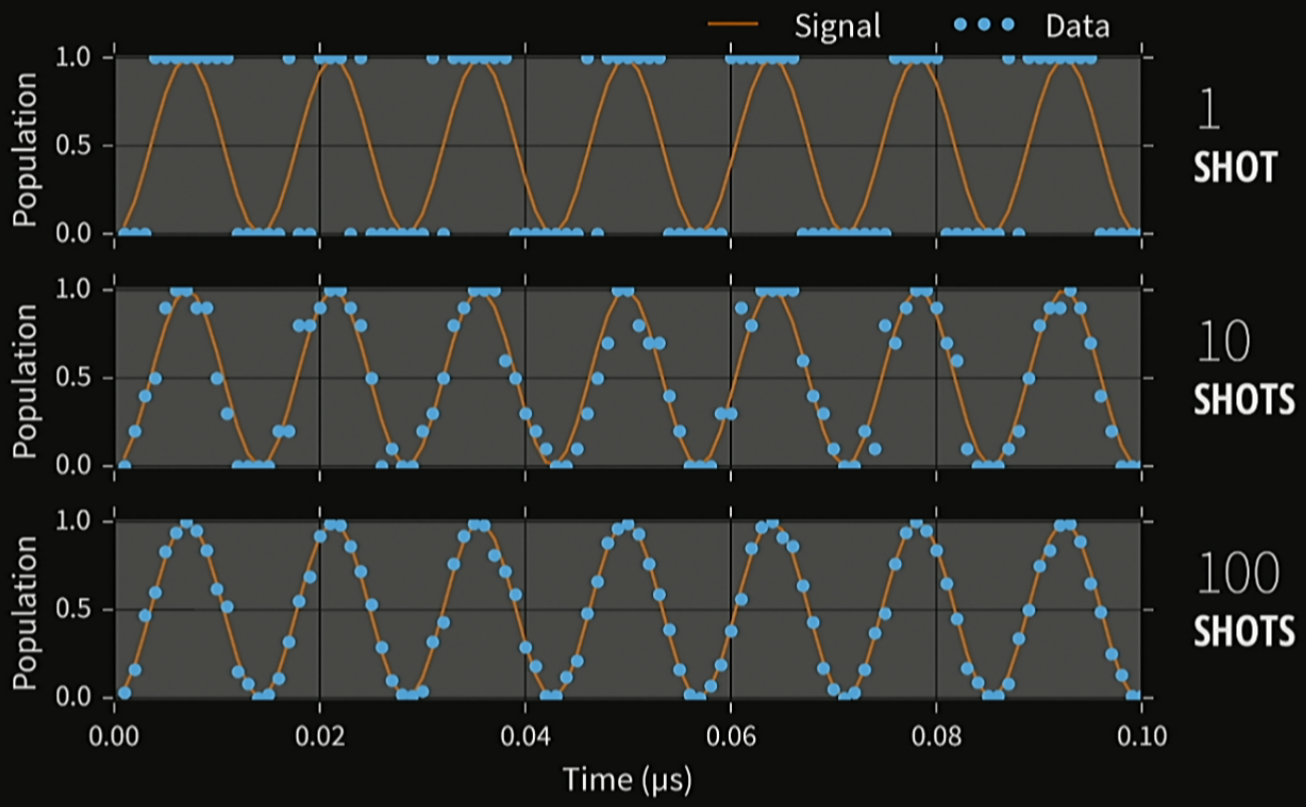
Example: Ramsey Estimation

Suppose $H = \omega\sigma_z/2$ for some unknown ω.

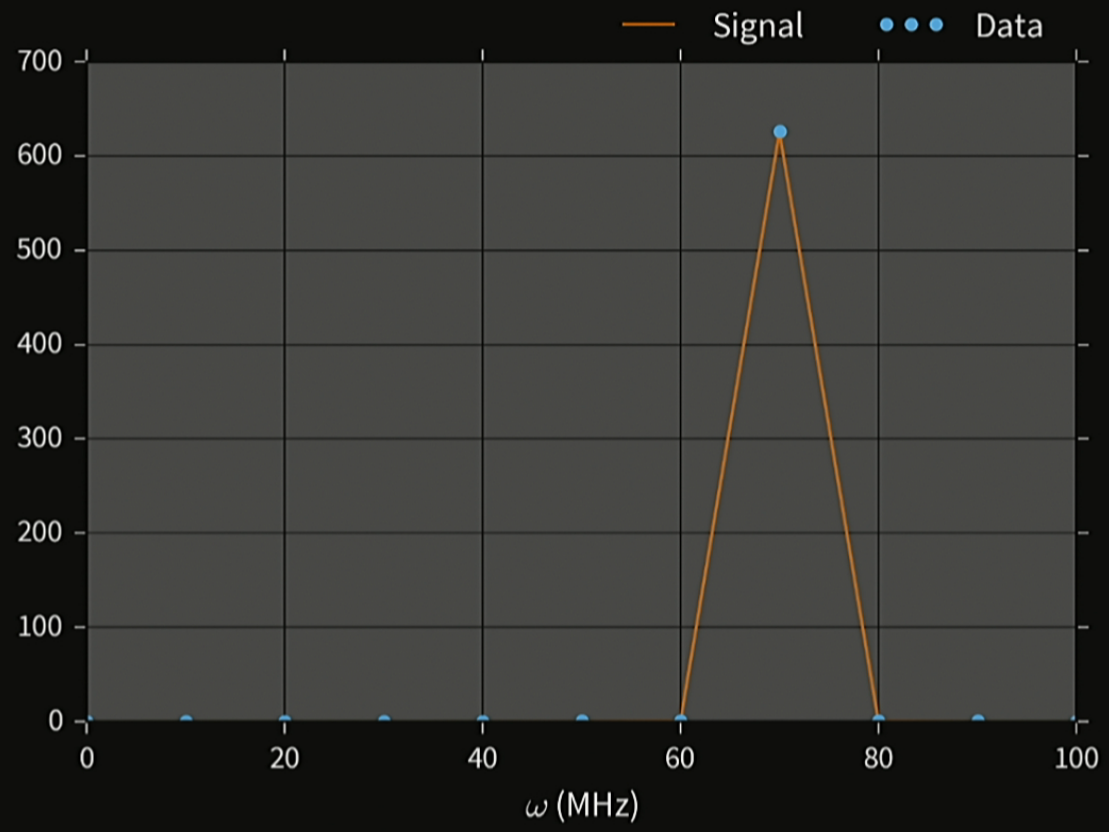
Traditional approach:

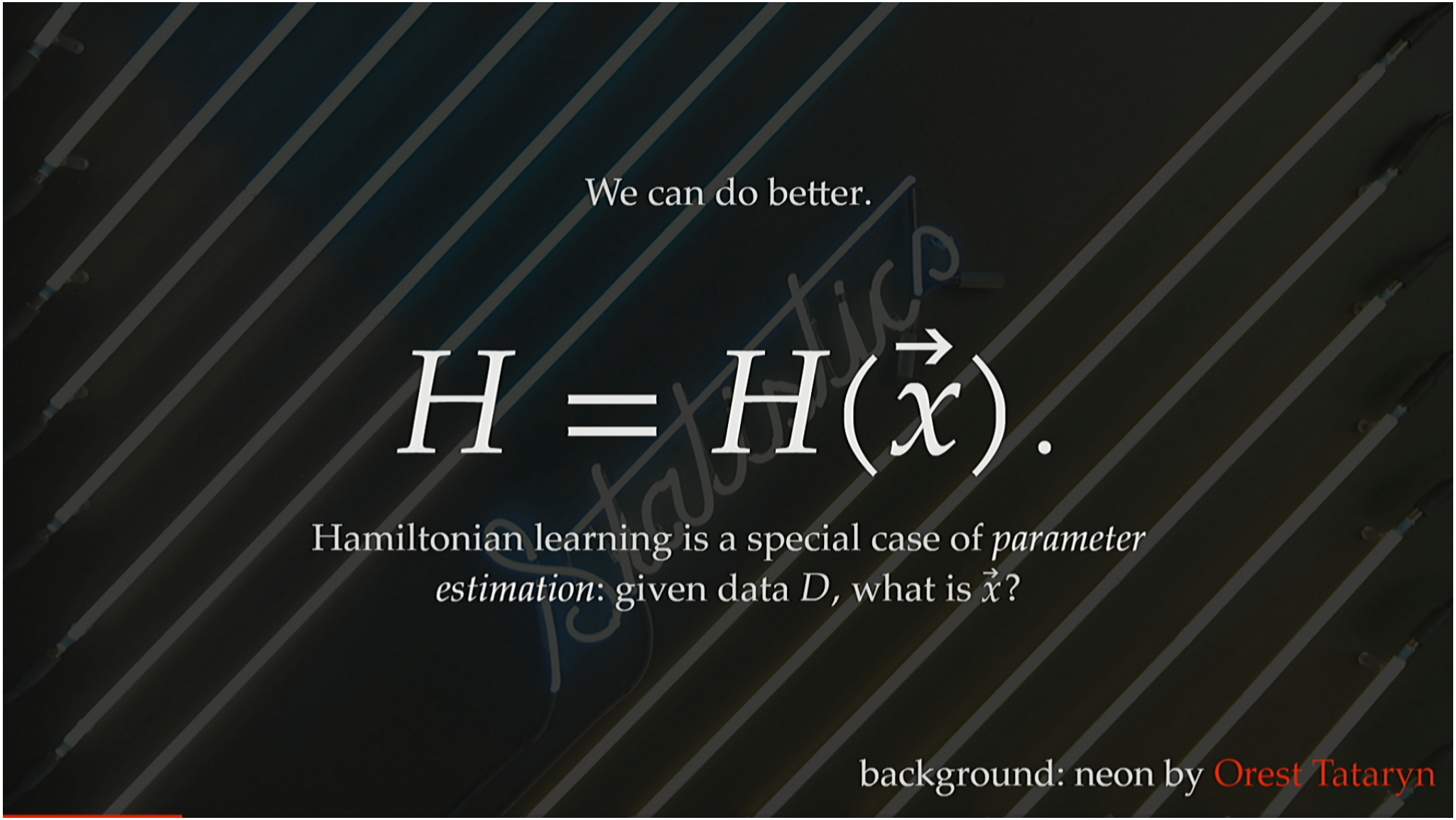
- Prepare $|+\rangle \propto |0\rangle + |1\rangle$, measure "click" w/ pr.: $|\langle +|e^{i\omega t\sigma_z/2}|+\rangle|^2 = \cos^2(\omega t/2)$.
- Repeat for many "shots" to estimate click pr.
- Repeat for many times to estimate signal.

You'll get something that looks a bit like this:



What's ω? Fourier transform and look at the peak.




We can do better.
$H = H(\vec{x})$.
Hamiltonian learning is a special case of *parameter estimation*: given data D, what is $\vec{x}$?
background: neon by Orest Tataryn

We want an approach that can work for small and large quantum devices alike.

We want an approach that can work for small and large quantum devices alike.

Punchline: our algorithm can characterize 50-qubit devices.

$$H = \sum_{i=1}^{50} \sum_{j=1}^{50} \omega_{ij} \sigma_z^{(i)} \sigma_z^{(j)},$$

$$\omega_{i,j} \sim \text{Uniform}\left(0, 10^{-2(|i-j|-1)}\right).$$

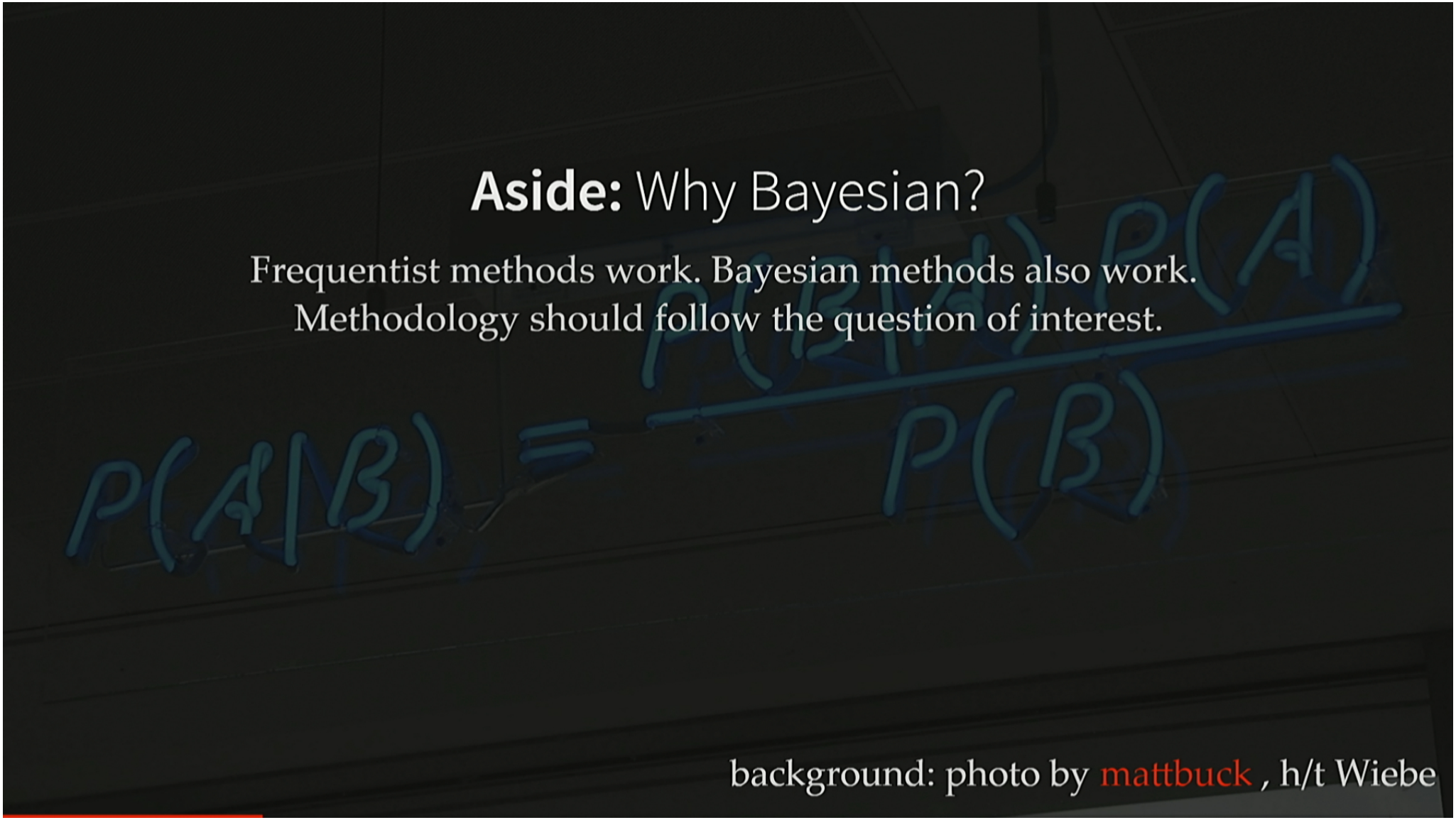
To get there, we consider several different approaches to parameter estimation.

Analytic	Sergeevich *et al.* DOI 10/c4vv95
	Ferrie, Granade, and Cory DOI 10/tfx
Rejection filter	Wiebe and Granade DOI 10/bk9d
Particle filter	Doucet *et al.* DOI 10/bmch
Likelihood-free particle filter	Ferrie and Granade DOI 10/tdj
Quantum bootstrapping	Wiebe, Granade and Cory DOI 10/7nx

The Likelihood Function

$|\langle +|e^{i\omega t\sigma_z/2}|+\rangle|^2$ defines probability $\Pr(d|\omega;t)$ for every outcome d, model ω and experiment t.

Basis for both maximum-likelihood and Bayesian methods.


Aside: Why Bayesian?
Frequentist methods work. Bayesian methods also work.
Methodology should follow the question of interest.
P(A|B) = P(B|A) P(A) / P(B)
background: photo by mattbuck , h/t Wiebe

Aside: Why Bayesian?

Frequentist methods work. Bayesian methods also work.
Methodology should follow the question of interest.

Example

What should I believe the properties of my system are, given my experience and a set of observations?

Bayesian question, hence Bayesian answer.

background: photo by mattbuck , h/t Wiebe

Bayesian Parameter Estimation

The likelihood tells us what we learn from data:

$$\Pr(\vec{x}|d;e) = \frac{\Pr(d|\vec{x};e)}{\Pr(d|e)} \Pr(\vec{x})$$

$$\text{Estimate } \hat{x} = \mathbb{E}[\vec{x}|d;e] = \int \vec{x} \Pr(\vec{x}|d;e) \mathrm{d}\vec{x}\,.$$

- **Optimal** for mean—squared error.

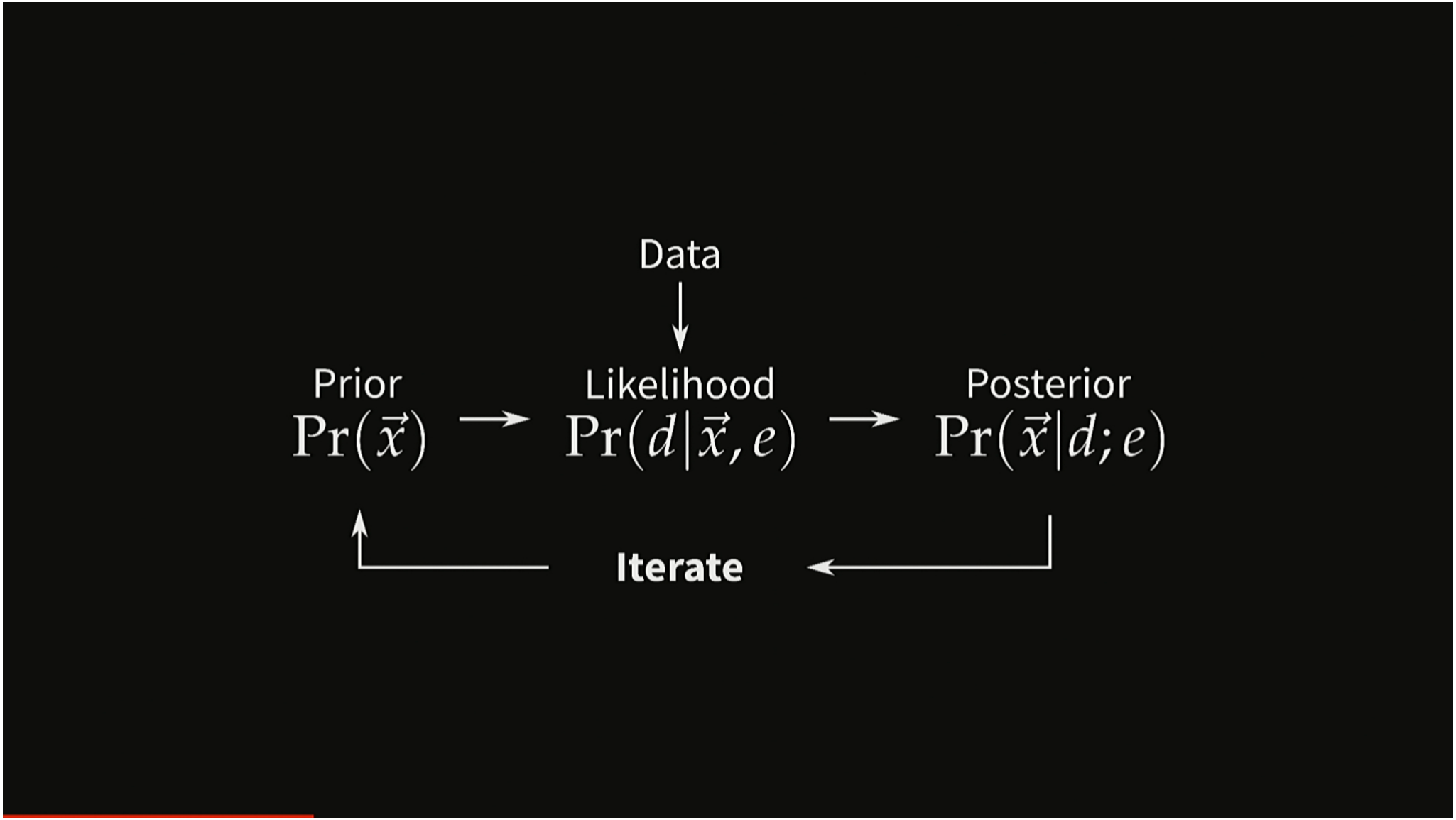
Bayesian Parameter Estimation

The likelihood tells us what we learn from data:

$$\Pr(\vec{x}|d;e) = \frac{\Pr(d|\vec{x};e)}{\Pr(d|e)} \Pr(\vec{x})$$

$$\text{Estimate } \hat{x} = \mathbb{E}[\vec{x}|d;e] = \int \vec{x} \Pr(\vec{x}|d;e) \mathrm{d}\vec{x}\,.$$

- **Optimal** for mean—squared error.


Data
Prior
$\Pr(\vec{x})$
Likelihood
$\Pr(d|\vec{x},e)$
Posterior
$\Pr(\vec{x}|d;e)$
Iterate

Inference as an Iterative Algorithm

Input: Prior $\Pr(\vec{x})$, data set D, likelihood $\Pr(d|\vec{x};e)$

- $p(\vec{x}) \leftarrow \Pr(\vec{x})$
- **For** each datum $d \in D$ and experiment e:
  - Update based on d:
    $p(\vec{x}) \leftarrow \Pr(d|\vec{x};e)p(\vec{x}) / \Pr(d)$.
- **Return** $\hat{x} = \int \vec{x}\, p(\vec{x})\, \mathrm{d}\vec{x}$.

Inference as an Iterative Algorithm

Input: Prior $\Pr(\vec{x})$, data set D, likelihood $\Pr(d|\vec{x};e)$

- $p(\vec{x}) \leftarrow \Pr(\vec{x})$
- **For** each datum $d \in D$ and experiment e:
  - Update based on d:
    $p(\vec{x}) \leftarrow \Pr(d|\vec{x};e)p(\vec{x}) / \Pr(d)$.
- **Return** $\hat{x} = \int \vec{x}\, p(\vec{x})\, \mathrm{d}\vec{x}$.

At each step, $p(\vec{x})$ also encodes our uncertainty.

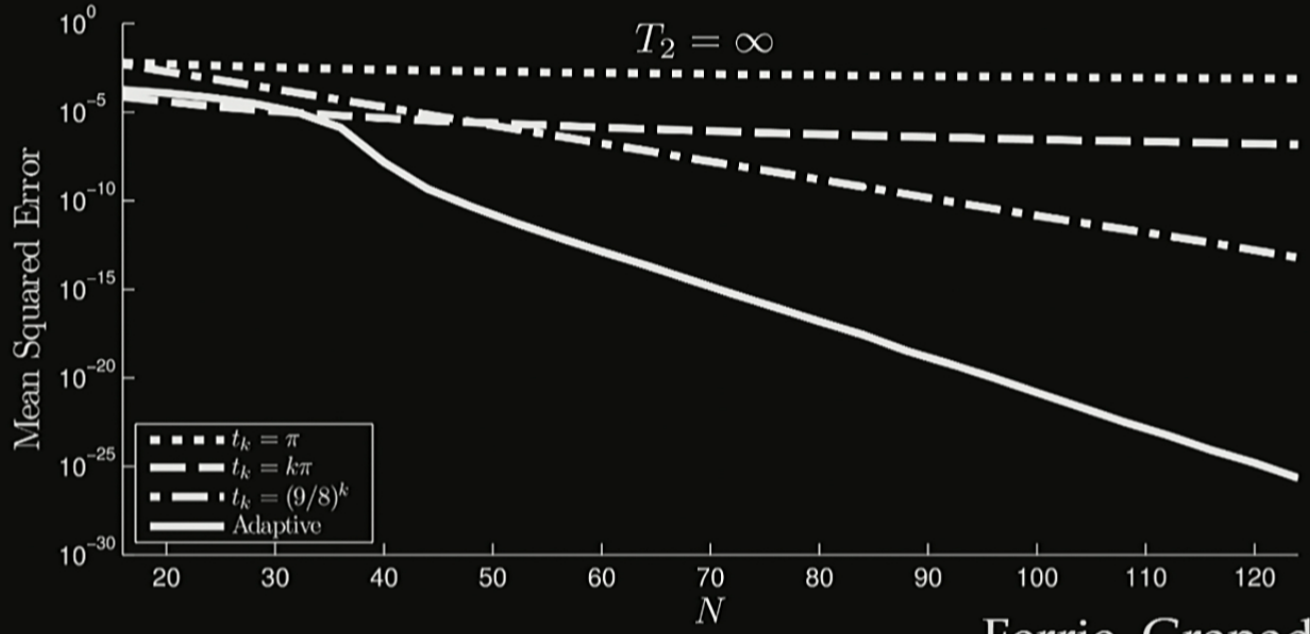
$$\mathbb{E}[(x - \hat{x})^2 | d;e] = \mathbb{V}[x \mid d;e].$$

We can use our posteriors to make *adaptive* decisions:

$$e_* = \arg\min_e \mathbb{E}_d[\mathbb{V}(x|d;e)]$$

Example: $x = (\omega)$

Can analytically find posterior for Gaussian priors, use to adaptively choose t_k .



Ferrie, Granade, Cory DOI 10/tfx

Problem: may be intractable to analytically compute

$$\hat{x} := \int \Pr(\vec{x}|d;e)\,\mathrm{d}\vec{x} = \int \frac{\Pr(d|\vec{x};e)}{\int \Pr(d|\vec{x};e)\,\Pr(\vec{x})\,\mathrm{d}\vec{x}}\,\Pr(\vec{x})\,\mathrm{d}\vec{x}.$$

Problem: may be intractable to analytically compute

$$\hat{x} := \int \Pr(\vec{x}|d;e)\,\mathrm{d}\vec{x} = \int \frac{\Pr(d|\vec{x};e)}{\int \Pr(d|\vec{x};e)\,\Pr(\vec{x})\,\mathrm{d}\vec{x}}\,\Pr(\vec{x})\,\mathrm{d}\vec{x}.$$

Problem: may be intractable to analytically compute

$$\hat{x} := \int \Pr(\vec{x}|d;e)\,\mathrm{d}\vec{x} = \int \frac{\Pr(d|\vec{x};e)}{\int \Pr(d|\vec{x};e)\,\Pr(\vec{x})\,\mathrm{d}\vec{x}}\,\Pr(\vec{x})\,\mathrm{d}\vec{x}.$$

Answer: numerically approximate $\int f(\vec{x})\,\Pr(\vec{x}|d)\,\mathrm{d}\vec{x}$.

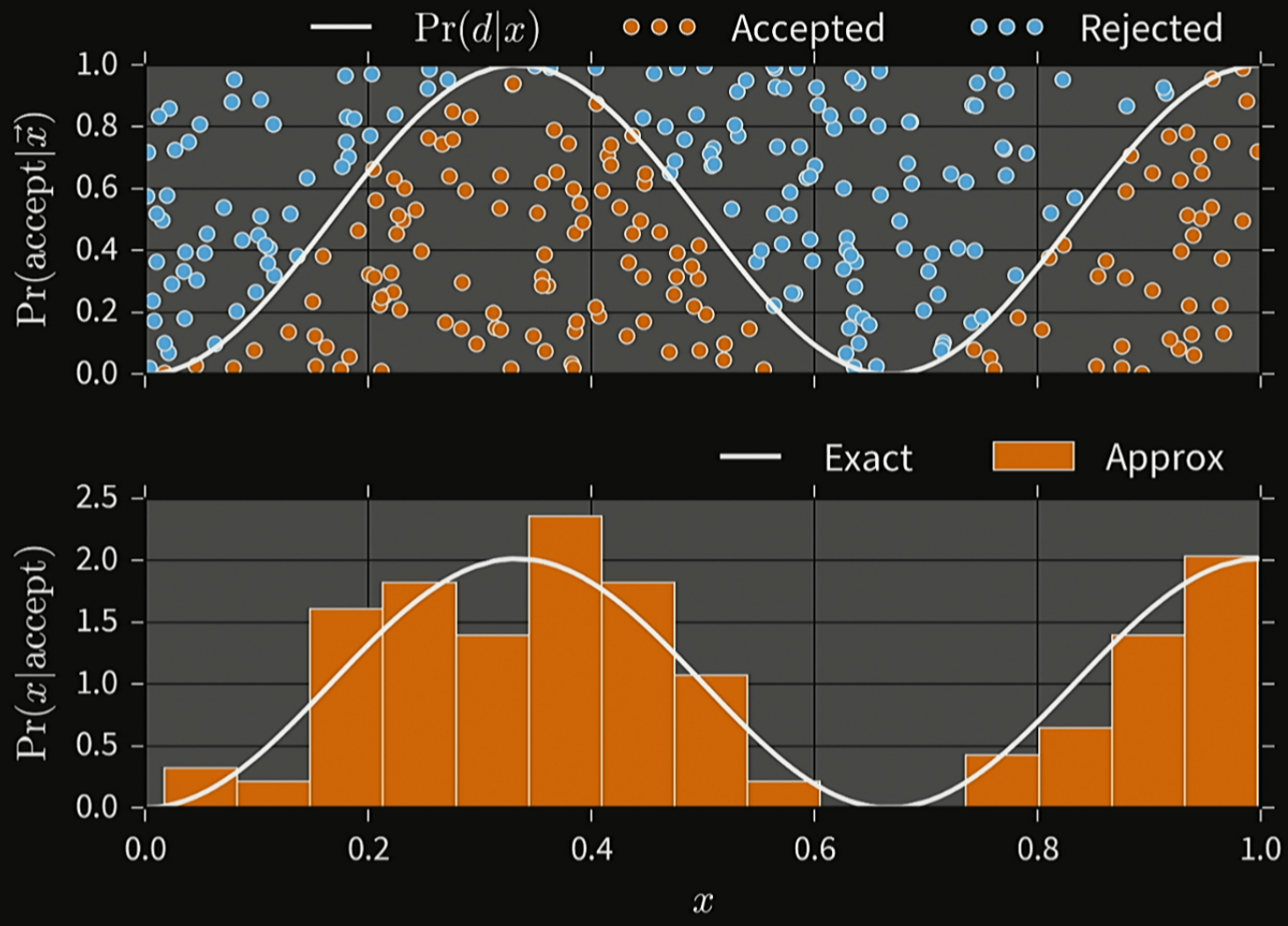
Efficient to use Monte Carlo integration if we can sample from the posterior $\Pr(\vec{x}|d;e)$.

Rejection Sampling

Given samples from $\Pr(\vec{x})$ and likelihood function $\Pr(d|\vec{x};e)$, how do we sample from posterior for datum d?

- Draw $\vec{x} \sim \Pr(\vec{x})$, accept $\vec{x}$ w/ $\Pr(d|\vec{x};e)$.

Accepted samples are distributed according to posterior.


$\Pr(d|x)$
Accepted
Rejected
$\Pr(\text{accept}|\vec{x})$
1.0
0.8
0.6
0.4
0.2
0.0
Exact
Approx
$\Pr(x|\text{accept})$
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
x

Rejection Sampling Isn't Enough

Let $D = d_1, \ldots, d_k$ be a set of data.

$$\Pr(\text{accept}|\vec{x}) = \Pr(D|\vec{x}) = \prod_{d \in D} \Pr(d|\vec{x}) \overset{k \to \infty}{\longrightarrow} 0.$$

Example: Biased Coin $x = (p)$

$$\Pr(H|p) = p, d \in \{H, T\}.$$

$$p \approx 0.5 \implies \Pr(d_1, \ldots, d_k|p) \approx 1/2^k .$$

We will accept exponentially few samples!

Gaussian Resampling

For each datum d, use rejection sampling to approximate posterior moments:

- $\bar{x} \leftarrow \mathbb{E}[\vec{x}|d]$.
- $\Sigma \leftarrow \mathrm{Cov}[\vec{x}|d] = \mathbb{E}[\vec{x}\vec{x}^{\mathrm{T}}|d] - \bar{x}\bar{x}^{\mathrm{T}}$.

Gaussian Resampling

For each datum d, use rejection sampling to approximate posterior moments:

- $\bar{x} \leftarrow \mathbb{E}[\vec{x}|d]$.
- $\Sigma \leftarrow \mathrm{Cov}[\vec{x}|d] = \mathbb{E}[\vec{x}\vec{x}^{\mathrm{T}}|d] - \bar{x}\bar{x}^{\mathrm{T}}$.

At the next iteration, draw prior samples from Gaussian with these moments:

$$\vec{x} \mid d \sim \mathrm{N}(\bar{x}, \Sigma)$$

Keeps $\Pr(\text{accept}) \approx \text{constant}$.

Can compute $\bar{x}$, Σ from one sample at a time by accumulating

$$x_\Sigma = \sum x \text{ and } x_\Sigma^2 = \sum x^2.$$

$$\bar{x} = x_\Sigma / n_{\text{accept}}$$

$$\Sigma = x_\Sigma^2 / n_{\text{accept}} - \bar{x}^2.$$

Welford's algorithm: numerically-stable modification.

Rejection Filtering (RejF)

Input: Prior mean $\bar{x}$, prior covariance Σ, number of samples m to accept.

- **For** each datum d and experiment e:
  - $n, \bar{x}', M_2 \leftarrow 0$ // Initialize Welford.
  - **While** $n < m$:
    - Draw $\vec{x} \sim \mathrm{N}(\bar{x}, \Sigma)$. // Sample f/ prior.
    - Accept $\vec{x}$ w/ $\Pr(d|\vec{x}; e)$.
    - **If** accepted, update n, $\bar{x}'$, M_2.
  - $\bar{x} \leftarrow \bar{x}'$, $\Sigma \leftarrow M_2/(n-1)$. // Est. moments.

Wiebe and Granade DOI 10/bk9d

Rejection Filtering (RejF)

Input: Prior mean $\bar{x}$, prior covariance Σ, number of samples m to accept.

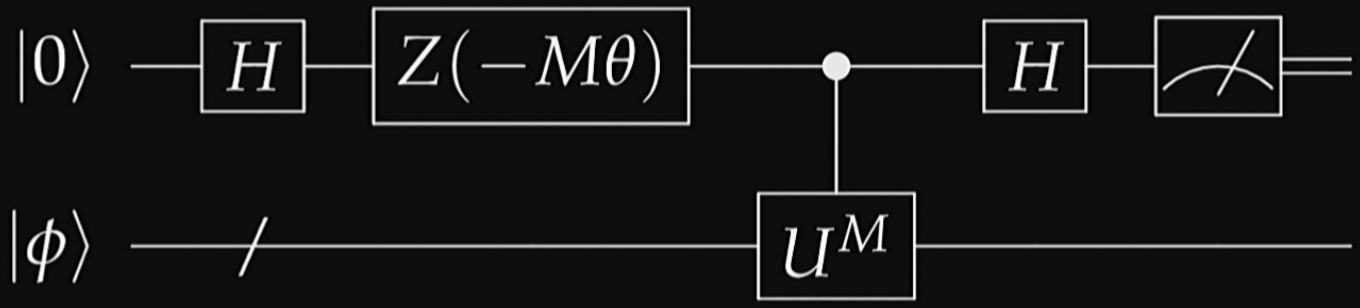
- **For** each datum d and experiment e:
  - $n, \bar{x}', M_2 \leftarrow 0$ // Initialize Welford.
  - **While** $n < m$:
    - Draw $\vec{x} \sim \mathrm{N}(\bar{x}, \Sigma)$. // Sample f/ prior.
    - Accept $\vec{x}$ w/ $\Pr(d|\vec{x}; e)$.
    - **If** accepted, update n, $\bar{x}'$, M_2.
  - $\bar{x} \leftarrow \bar{x}'$, $\Sigma \leftarrow M_2/(n-1)$. // Est. moments.

Easy to implement and embed in control hardware.

Wiebe and Granade DOI 10/bk9d

Example: Phase Estimation, $x = (\phi)$

Prepare state $|\phi\rangle$ s. t. $U|\phi\rangle = e^{i\phi}|\phi\rangle$, measure to learn ϕ.



$$\Pr(1|\phi; M, \theta) = \cos^2(M[\phi - \theta])$$

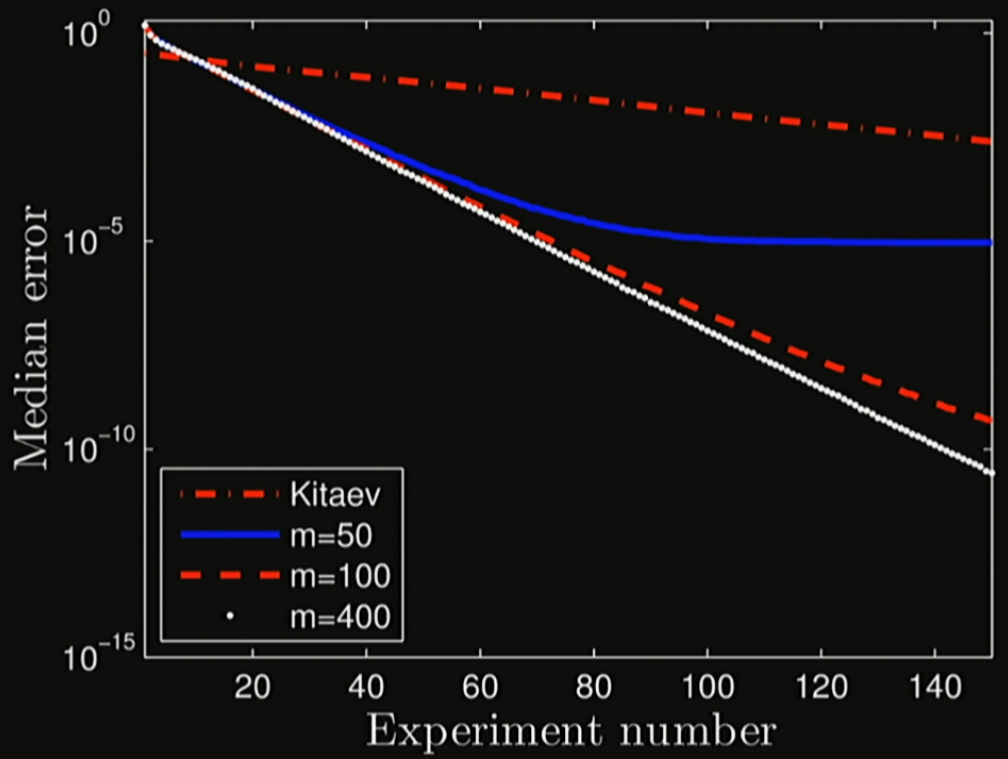
Applications

Interferometry / metrology
Gate calibration / robust PE
Quantum simulation and chemistry

Higgins *et al.* DOI 10/crwd6w
Kimmel *et al.* DOI 10/bmrg
Reiher *et al.* 1605.03590

Svore *et al.* 1304.0741

Example: Phase Estimation, $x = (\phi)$



Wiebe and Granade DOI 10/bk9d

Drawback: RejF requires posterior after each datum to be ≈ Gaussian.

Drawback: RejF requires posterior after each datum to be ≈ Gaussian.

Drawback: RejF requires posterior after each datum to be ≈ Gaussian.

We can solve this by using a more general approach:

- Weaken Gaussian assumption.
- Generalize the rejection sampling step.

Liu-West Resampler

If we remember each sample $\vec{x}$, we can use them to relax RejF assumptions.

Input: $a, h \in [0,1]$ s.t. $a^2 + h^2 = 1$, distribution $p(\vec{x})$.

- Approximate $\bar{x} \leftarrow \mathbb{E}[\vec{x}]$, $\Sigma \leftarrow \mathrm{Cov}(\vec{x})$
- Draw *parent* $\vec{x}$ from $p(\vec{x})$.
- Draw $\vec{\epsilon} \sim \mathrm{N}(0, \Sigma)$.
- **Return** new sample $\vec{x}' \leftarrow a\vec{x} + (1-a)\bar{x} + h\vec{\epsilon}$.

Liu and West DOI 10/8c2

Why Does Liu-West Work?

$$\vec{x}' \leftarrow a\vec{x} + (1-a)\bar{x} + h\vec{\epsilon}$$
$$\mathbb{E}[\vec{x}'] = [a + (1-a)]\bar{x}$$
$$\mathrm{Cov}(\vec{x}') = (a^2 + h^2)\mathrm{Cov}(\vec{x}).$$
$$\implies a^2 + h^2 = 1 \text{ preserves } \mathbb{E}[\vec{x}], \mathrm{Cov}(\vec{x}).$$

Mixes original approximation with $1 - a$ of a Gaussian matching moments.

- $a \to 0$: RejF (assumed density) approx
- $a \to 1$: Bootstrap
- $a = 0.98$: typical case (2% Gaussian).

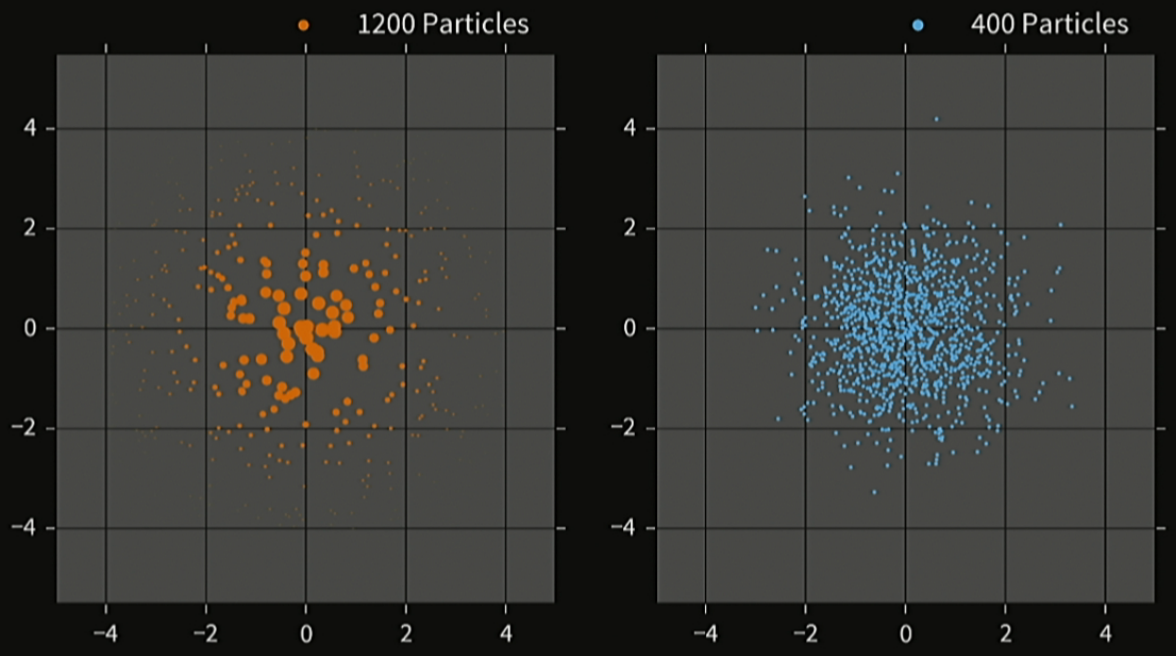
From Samples to Particles

Define a particle $(w_i, \vec{x}_i)$ as a sample $\vec{x}_i$ and a weight $w_i \in [0, 1]$.

- $\mathbb{E}[\vec{x}] = \sum_i w_i \vec{x}_i$
- $\mathrm{Cov}(\vec{x}) = \sum_i w_i \vec{x}_i \vec{x}_i^{\mathrm{T}} - \mathbb{E}[\vec{x}]\mathbb{E}^{\mathrm{T}}[\vec{x}]$
- $\mathbb{E}[f(\vec{x})] = \sum_i w_i f(\vec{x}_i)$

Corresponds to $p(\vec{x}) \approx \sum_i w_i \delta(\vec{x} - \vec{x}_i)$.

Particles can represent distributions using either weights or positions.



Particle Filter

- Draw N initial samples $\vec{x}_i$ from the prior $\Pr(\vec{x})$ w/ uniform weights.
- Instead of rej. sampling, update weights by

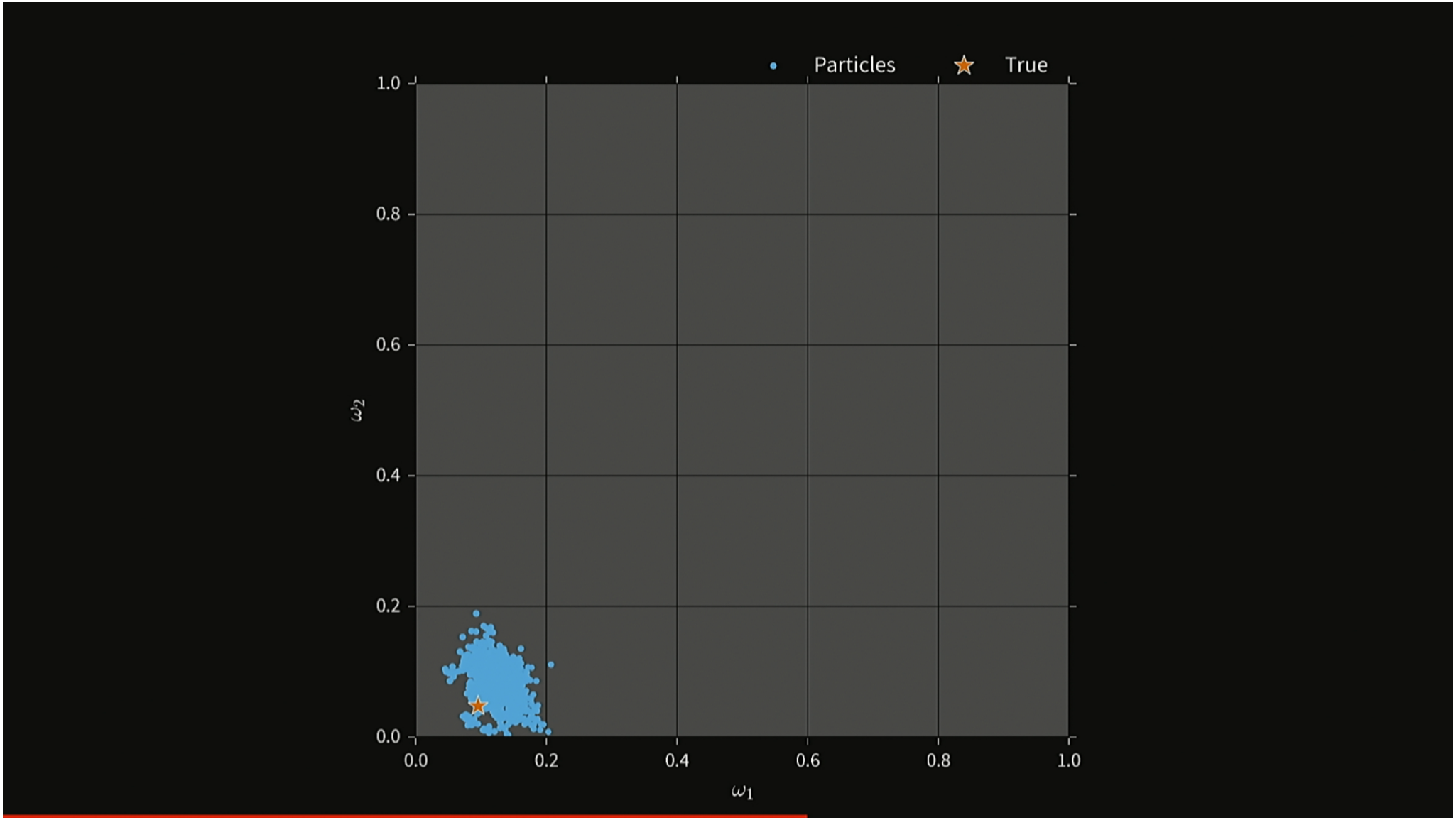
$$\tilde{w}_i = w_i \times \Pr(d|\vec{x}_i; e)$$

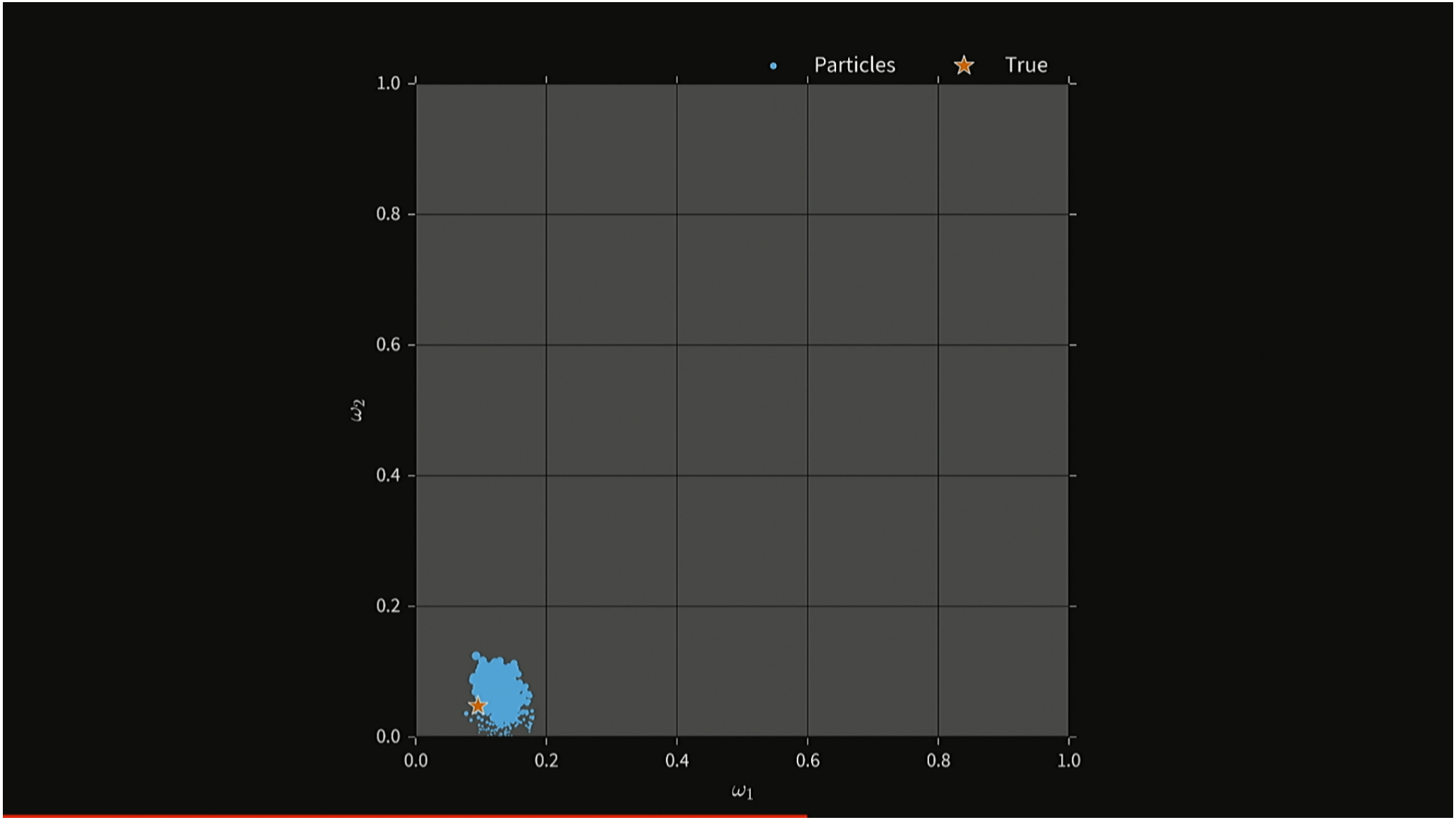
- Renormalize.

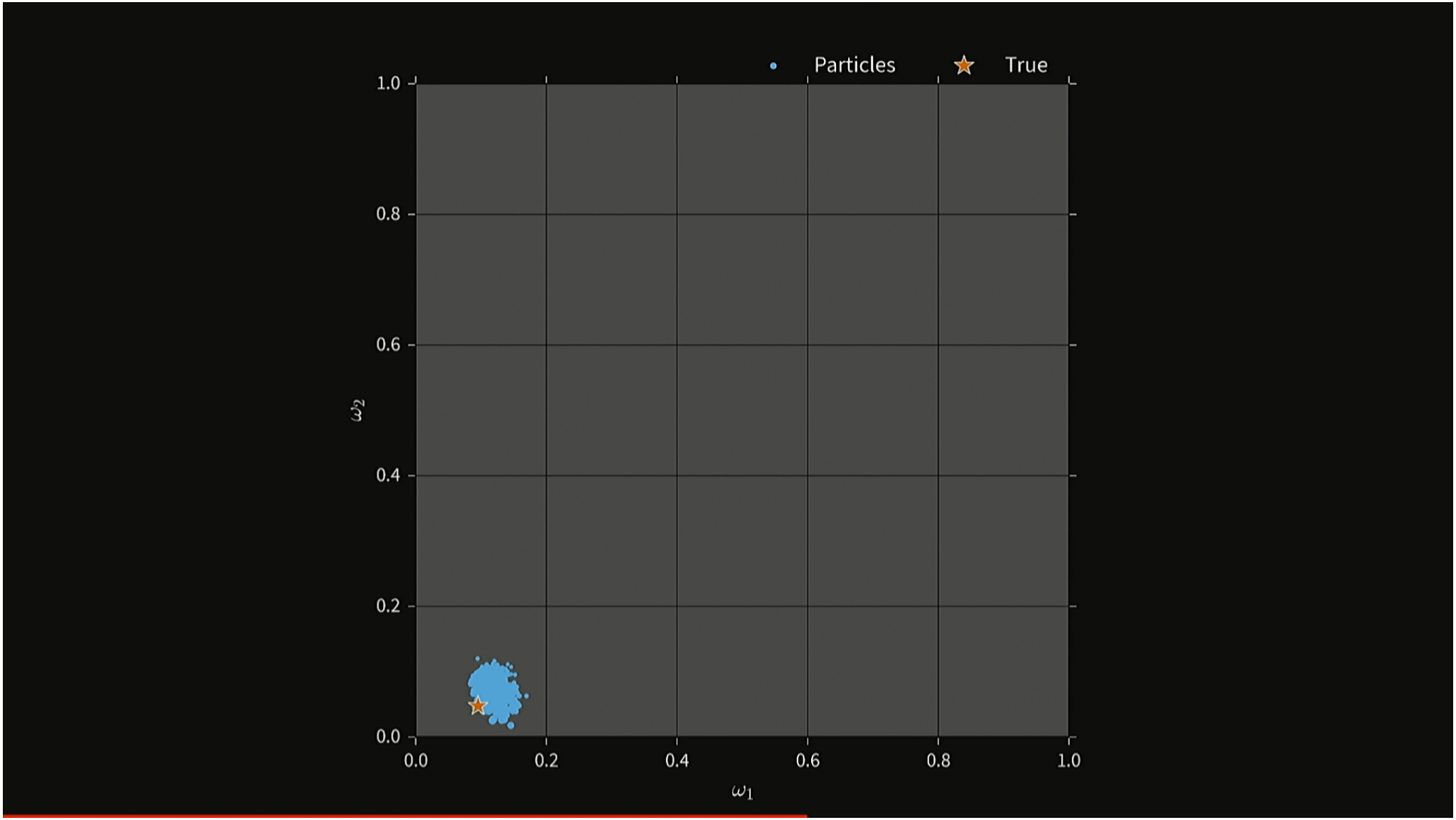
$$w_i \mapsto \tilde{w}_i / \sum_i \tilde{w}_i$$

- Periodically use Liu-West to draw new $\{\vec{x}_i\}$ with uniform weights. // Store posterior in positions.

Doucet *et al.* DOI 10/bmch


Particles
True
1.0
0.8
0.6
0.4
0.2
0.0
ω_2
0.0
0.2
0.4
0.6
0.8
1.0
ω_1


Particles
True
1.0
0.8
0.6
0.4
0.2
0.0
ω_2
0.0
0.2
0.4
0.6
0.8
1.0
ω_1


Particles
True
1.0
0.8
0.6
0.4
0.2
0.0
ω_2
0.0
0.2
0.4
0.6
0.8
1.0
ω_1

Useful for Hamiltonian models...

Rabi/Ramsey/Phase est.	Granade *et al.* DOI 10/s87
Swap spectroscopy	Stenberg *et al.* DOI 10/7nw

...as well as other QIP tasks.

Tomography	Huszár and Holsby DOI 10/s86
	Struchalin *et al.* DOI 10/bmg5
	Ferrie DOI 10/7nt
	Granade *et al.* DOI 10/bhdw, 1605.05039
Randomized benchmarking	Granade *et al.* DOI 10/zmz
Continuous measurement	Chase and Geremia DOI 10/chk4q7
Interferometry/metrology	Granade HDL 10012/9217

Estimation in Practice

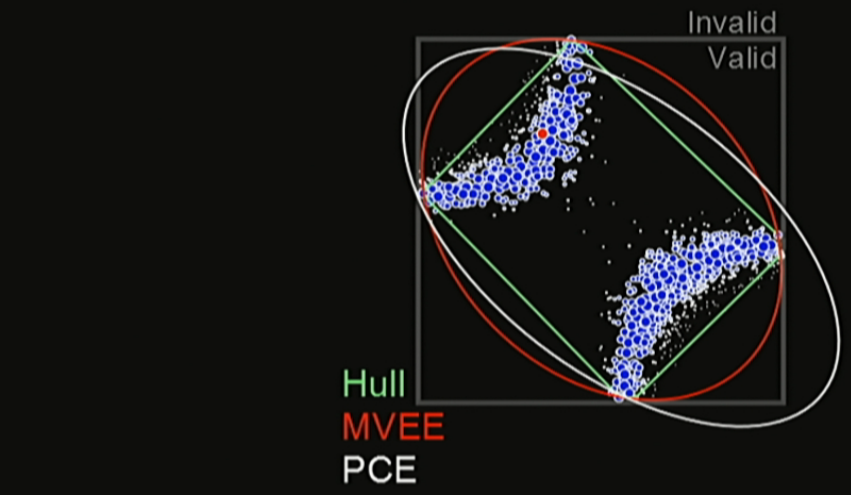
We need a bit more to make particle filtering a *practical* solution.

- Error bars — How well do we know $\vec{x}$?
- Time-dependence — $\vec{x} = \vec{x}(t)$
- Software impl. — Off-the-shelf.

Dealing with each in turn...

Error Bars

Particle filters report *credible regions* X_α s.t.
$\Pr(\vec{x} \in X_\alpha | d; e) \geq \alpha.$



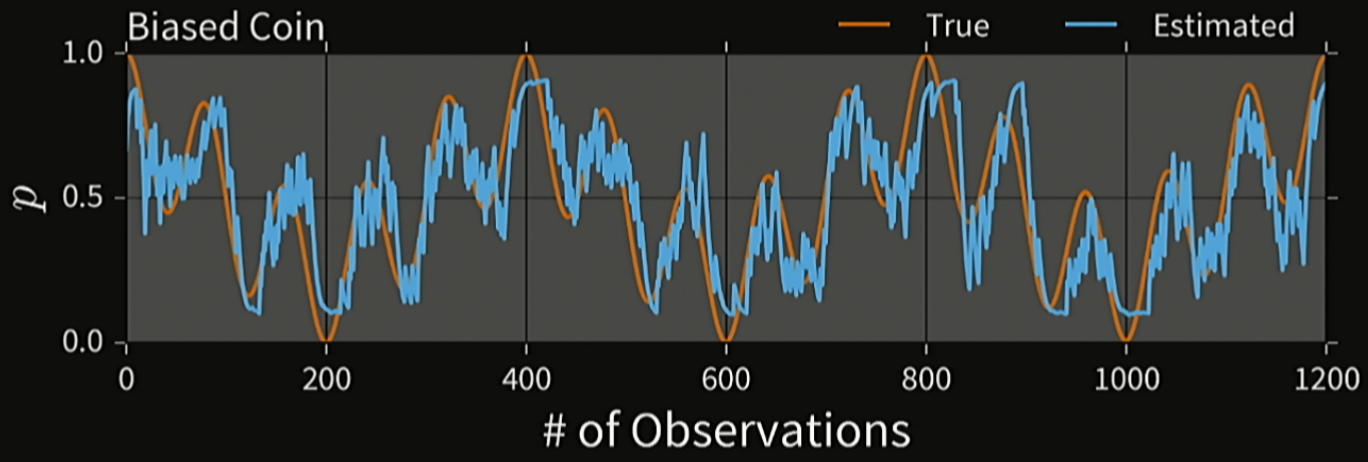
E.g.: Posterior covariance ellipse, convex hull, MVEE.

figure: Ferrie DOI 10/tb4

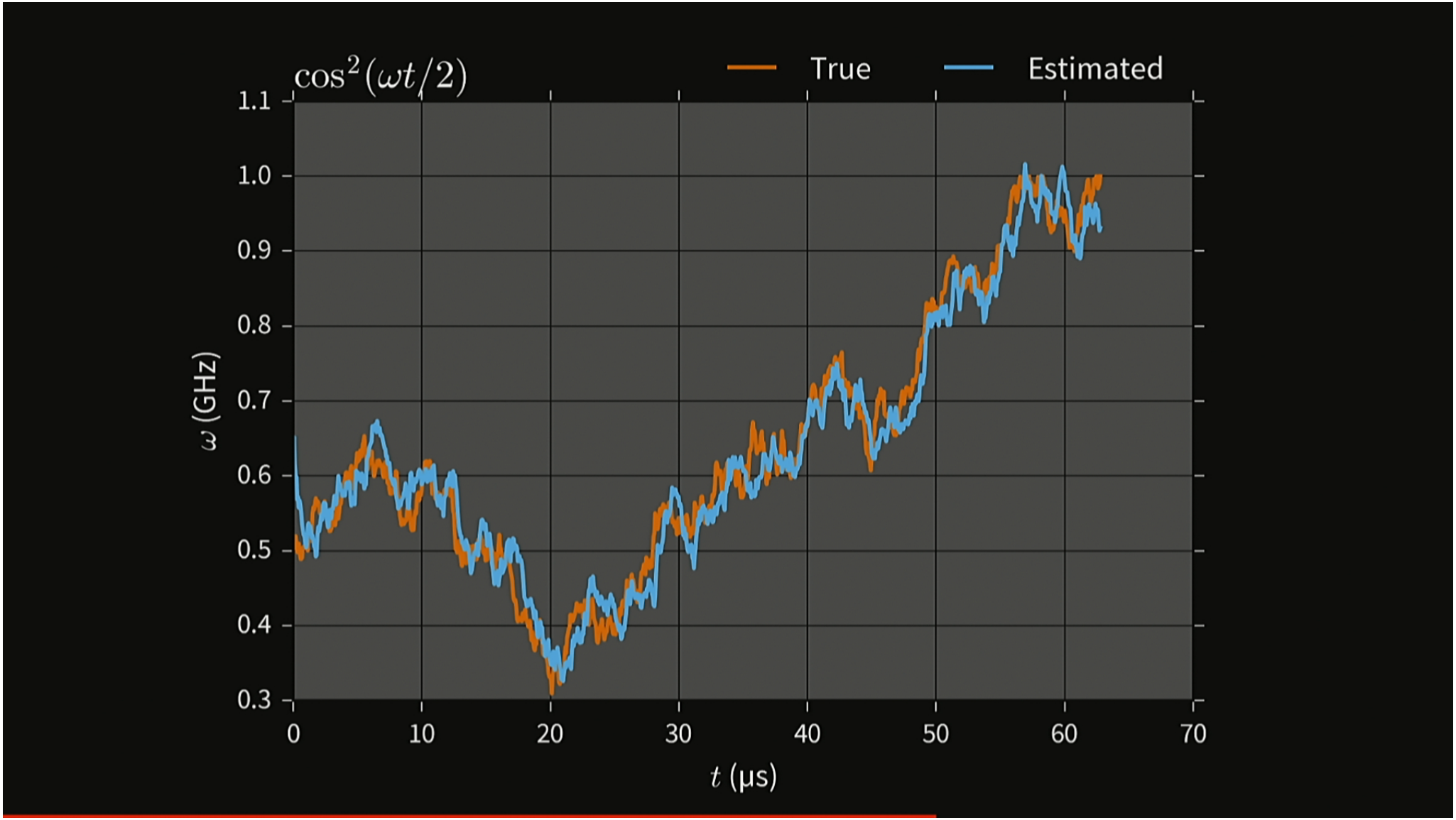
Time-Dependence

In addition to updating particle weights, move each particle stochastically:

$$\vec{x}(t_{k+1}) = \vec{x}(t_k) + \vec{\eta}, \qquad \vec{\eta} \sim \mathrm{N}(0, (t_{k+1} - t_k)\Sigma)$$



Isard and Blake DOI 10/cc76f6


$\cos^2(\omega t/2)$
True
Estimated
1.1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
ω (GHz)
0
10
20
30
40
50
60
70
t (μs)

Software Implementation

We provide **QInfer**, an open-source implementation for use in quantum information.

Written in Python, works with MATLAB and Julia.

qinfer.org

Software Implementation

We provide **QInfer**, an open-source implementation for use in quantum information.

Written in Python, works with MATLAB and Julia.

Puts focus on algorithms and experiments, not implementations.

qinfer.org

Bigger and Better

We've seen that filtering is useful for estimating small quantum models. Now let's push on to bigger systems.

What challenges do we face for large systems?

Simulation Costs

$$\tilde{w}_i = w_i \times \Pr(d|\vec{x}_i; e)$$

$$w_i \mapsto \tilde{w}_i / \sum_i \tilde{w}_i$$

- $\Pr(d|\vec{x}_i; e)$ is a *simulation*.
- Need to simulate for each particle (~1000s calls/datum).
- Infeasible for large quantum models.

Let's do better: use *quantum* simulation instead.

What challenges do we face for large systems?

Simulation Costs

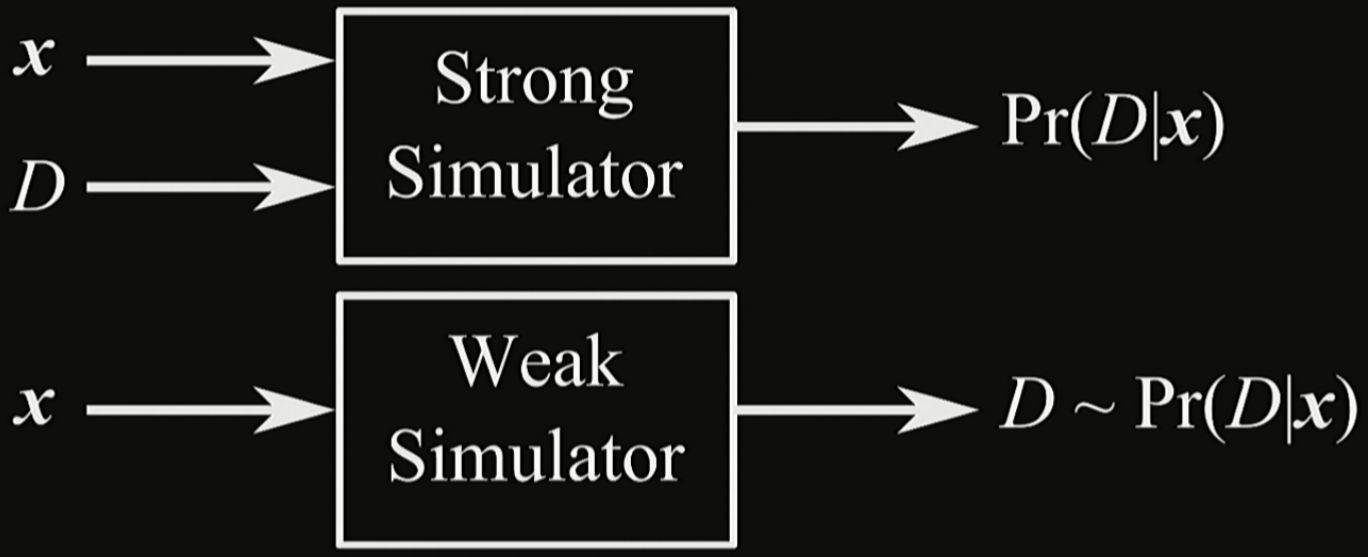
$$\tilde{w}_i = w_i \times \Pr(d|\vec{x}_i; e)$$

$$w_i \mapsto \tilde{w}_i / \sum_i \tilde{w}_i$$

- $\Pr(d|\vec{x}_i; e)$ is a *simulation*.
- Need to simulate for each particle (~1000s calls/datum).
- Infeasible for large quantum models.

Let's do better: use *quantum* simulation instead.

Two Kinds of Simulation

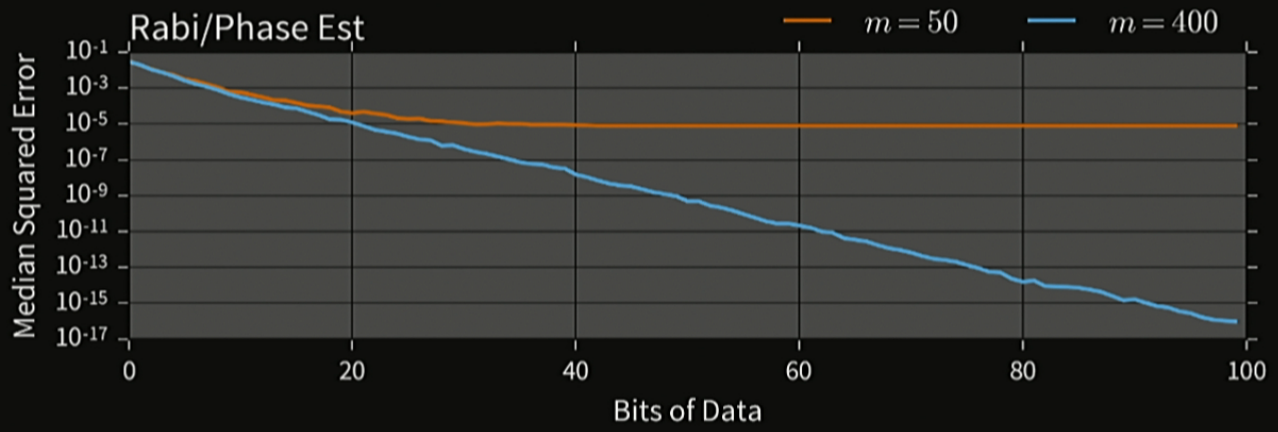


Weak simulators produce *plausible datasets*.

Likelihood-Free RejF

Replace rej. sampling step by drawing datum from likelihood instead of computing exact value:

- Draw datum d' from $\Pr(d|\vec{x};e)$.
- Accept $\vec{x}$ if $d = d'$.



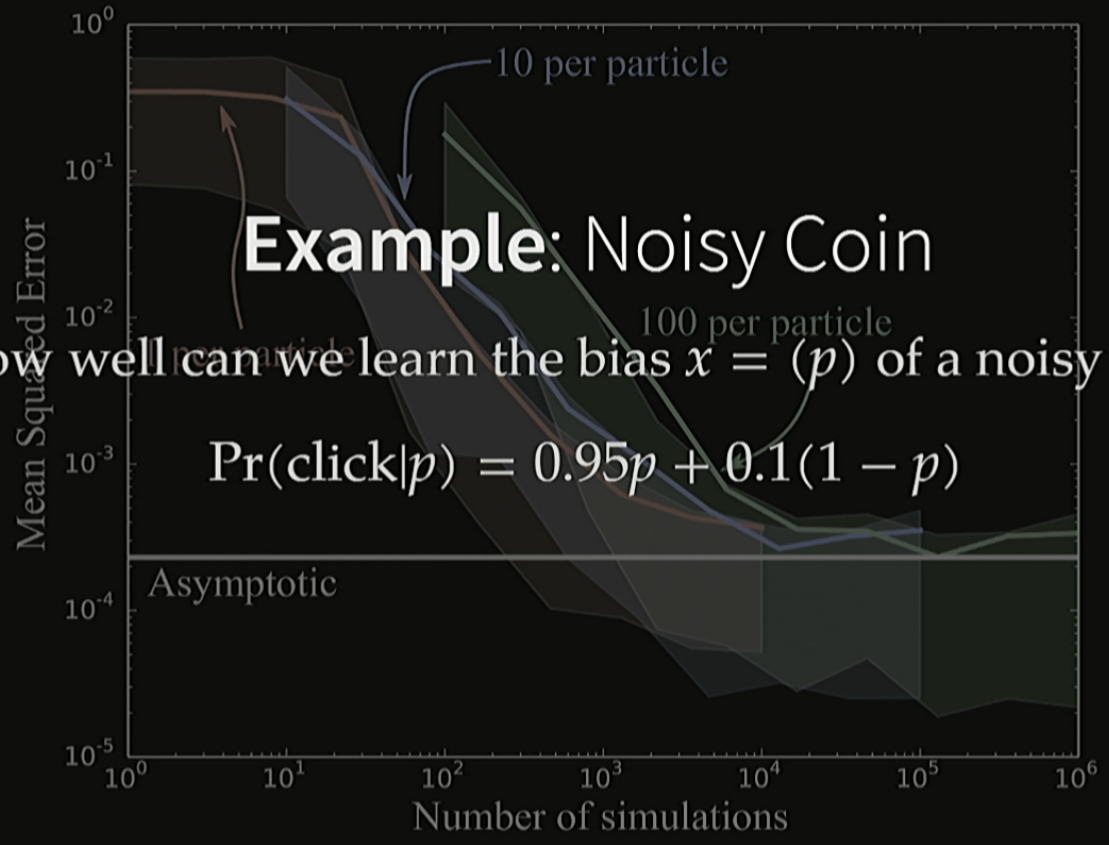
Likelihood-Free Particle Filtering

Can also use frequencies f from weak simulation to approximate likelihoods in particle filtering.

$$\widehat{\Pr}(d|\vec{x};e) = \frac{f(d|\vec{x};e)}{k}$$

k: number of weak simulation calls used.

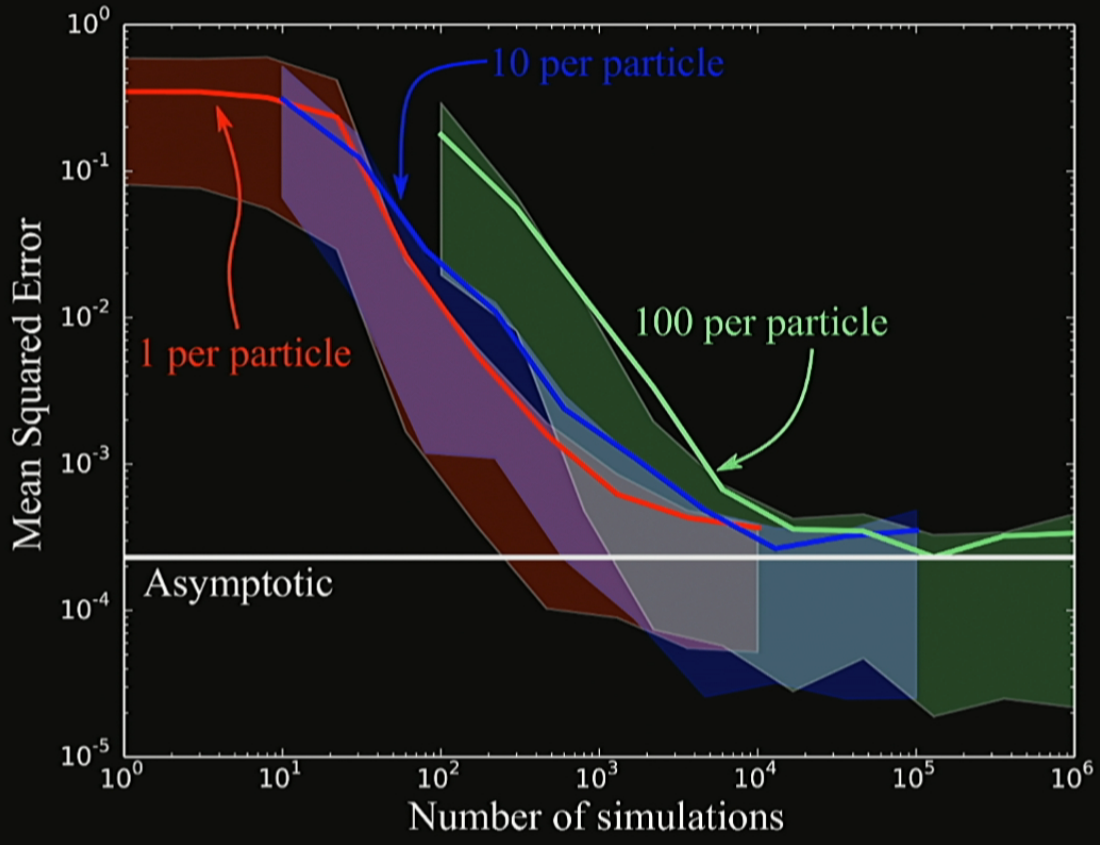
Ferrie and Granade DOI 10/tdj

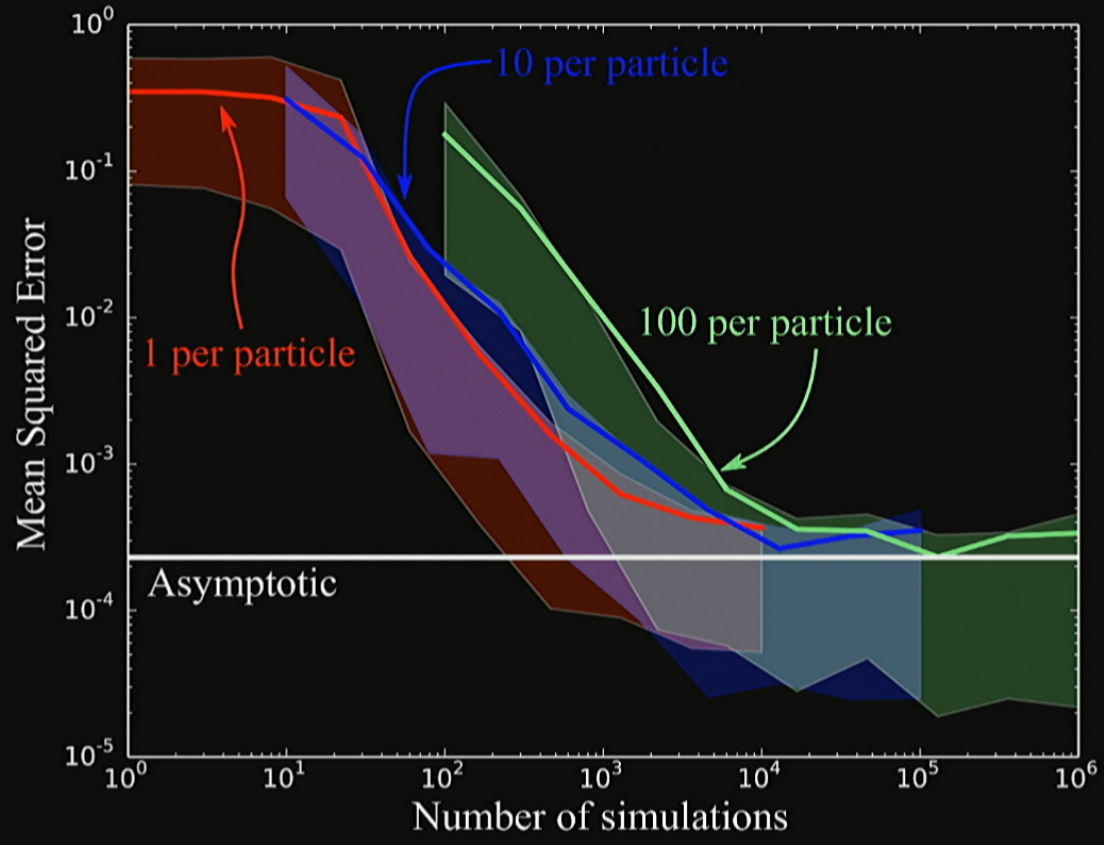


Example: Noisy Coin

How well can we learn the bias $x = (p)$ of a noisy coin?

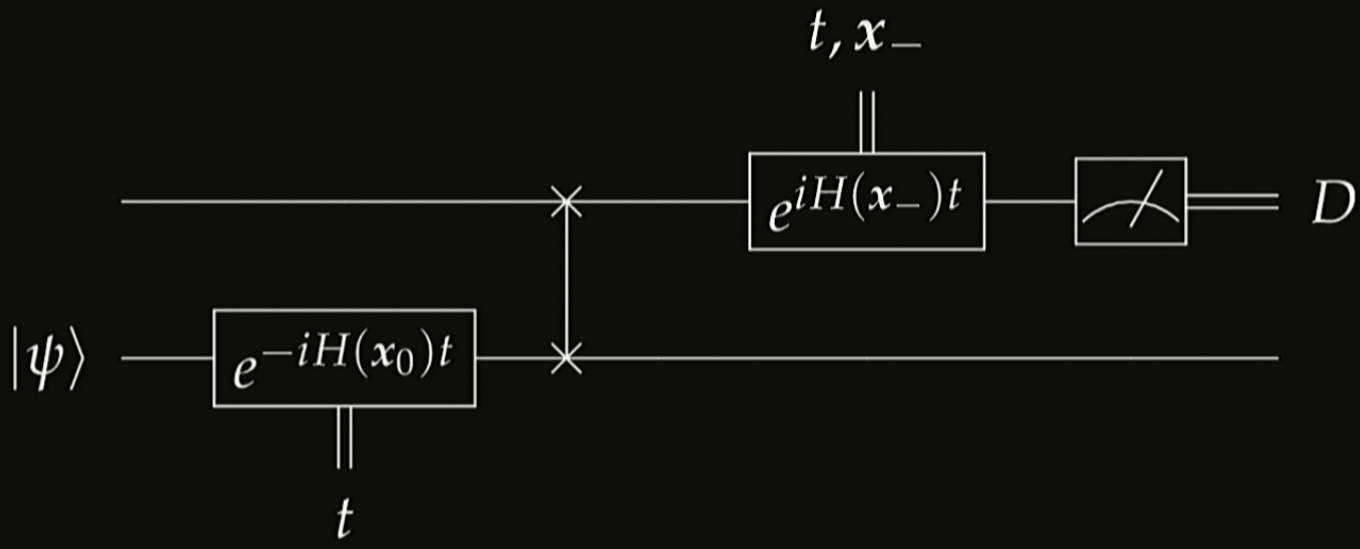
$$\Pr(\text{click}|p) = 0.95p + 0.1(1 - p)$$


10 per particle
1 per particle
100 per particle
Asymptotic
Mean Squared Error
Number of simulations
10^0
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^0
10^1
10^2
10^3
10^4
10^5
10^6


10 per particle
1 per particle
100 per particle
Asymptotic
Mean Squared Error
Number of simulations
10^0
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^0
10^1
10^2
10^3
10^4
10^5
10^6

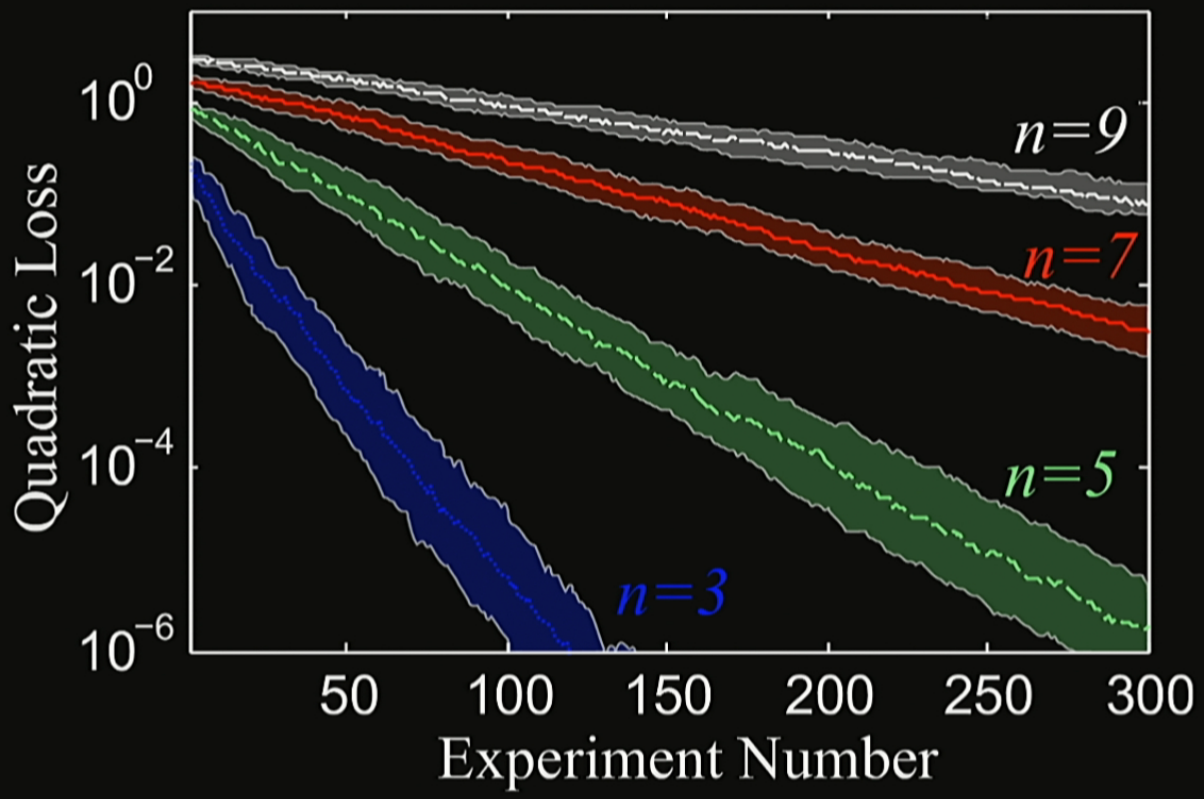
Quantum Hamiltonian Learning

We can learn large Hamiltonians by using likelihood-free filtering w/ interactivity.

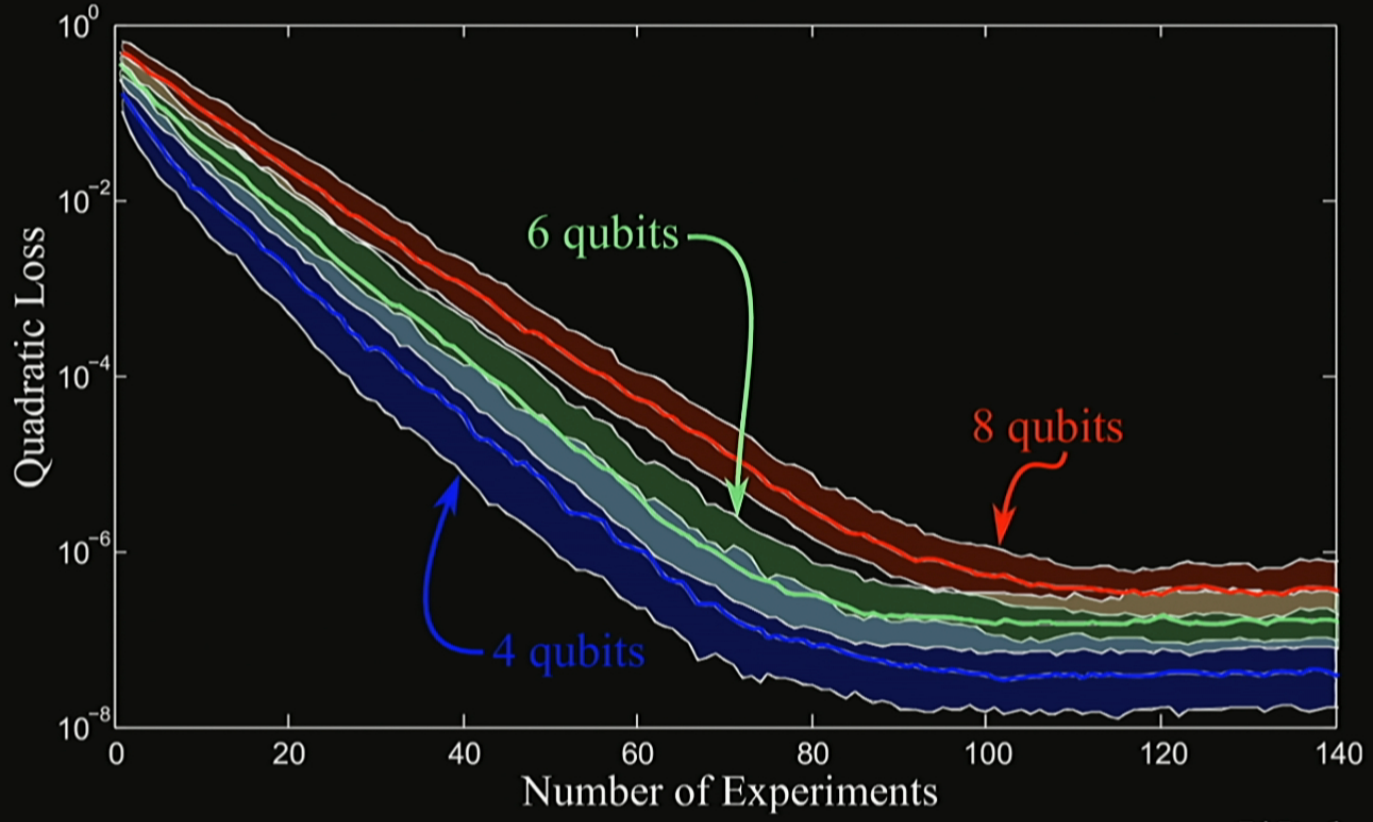


Perform weak simulations on trusted device only.

Example: Ising on Complete Graph



Robust even to wrong model. (0.5 NN + 10^{-4} Complete)



Wiebe *et al.* DOI 10/tdk

Quantum Bootstrapping

One important approximation f/ physical insight:
information locality.

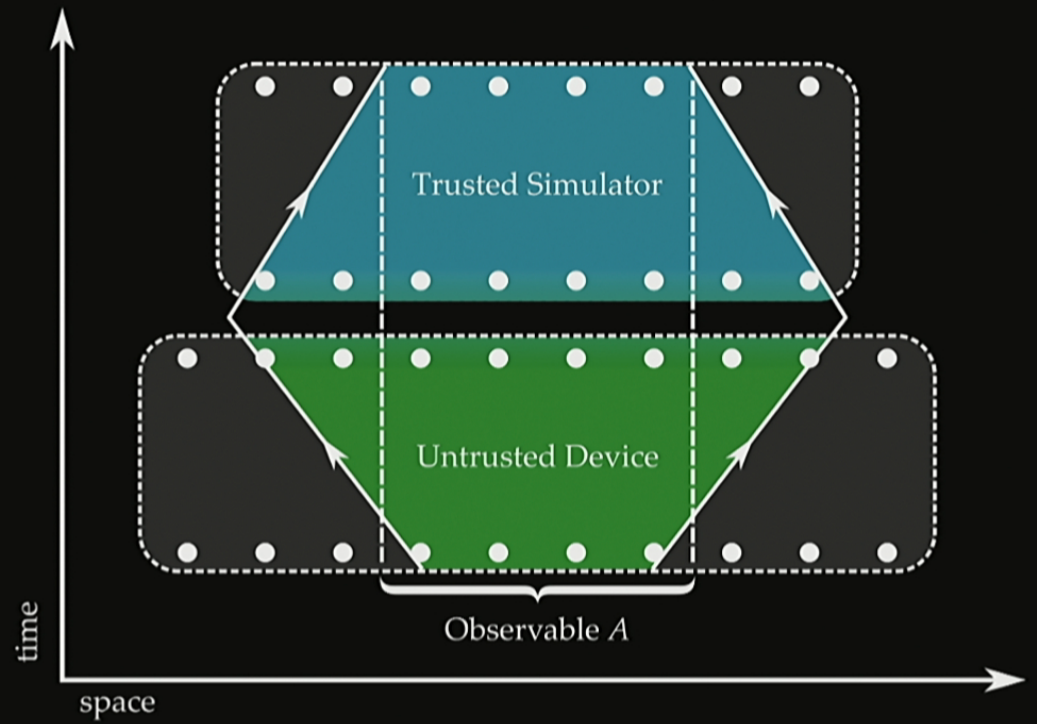
Wiebe *et al.* DOI 10/7nx

Quantum Bootstrapping

One important approximation f/ physical insight: *information locality*.

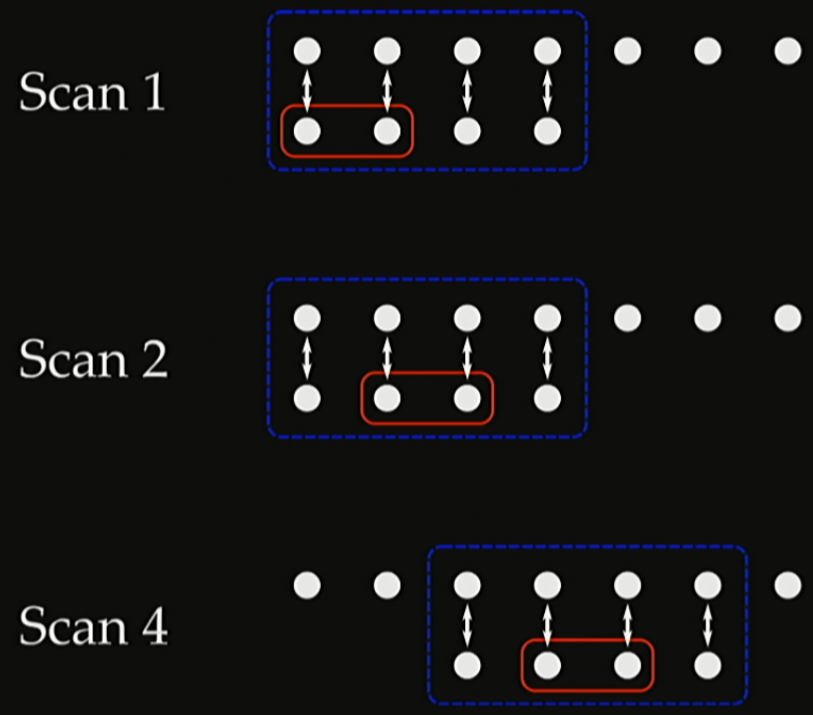
Allows using *small* trusted device to learn large Hamiltonians.

Wiebe *et al.* DOI 10/7nx



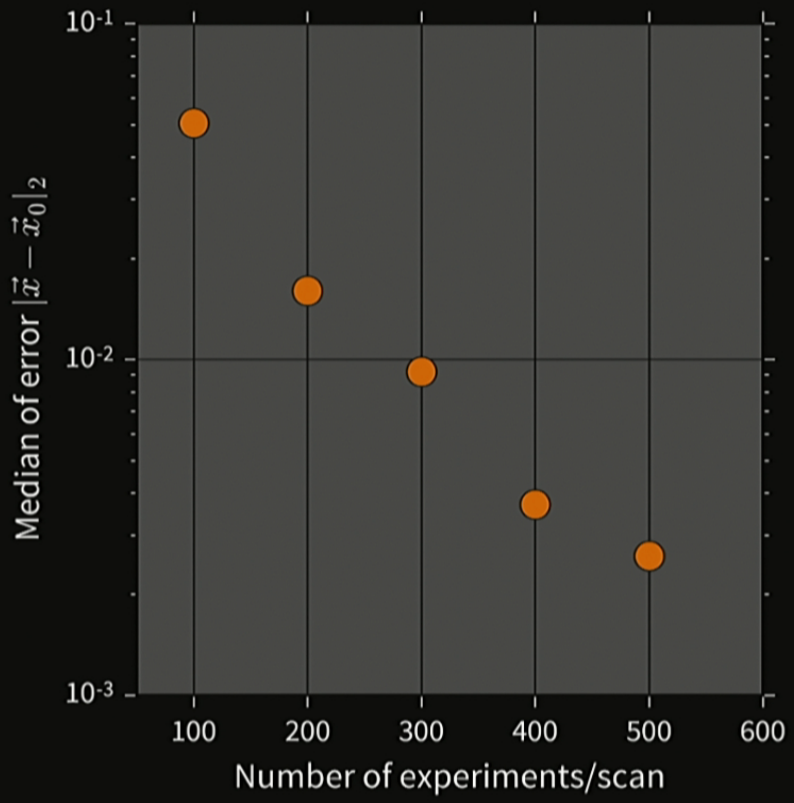
Approximation quality can be bounded if Lieb-Robinson velocity is finite.

Scan trusted device across untrusted.



Run particle filter *only* on supported parameters.

50 qubit Ising chain, 8 qubit simulator, 4 qubit observable



Filtering

- Practical solution for current experimental tasks.
- Enables learning large Hamiltonians using quantum resources.
- Physical insight gives new statistical algorithm for even larger systems.

Going Further

Hyperparameterization Granade *et al.* DOI 10/s87

$\Pr(d|y) = \mathbb{E}_x[\Pr(d|x)\Pr(x|y)]$.

Allows composing w/ noise, inhomogeneity, etc.

Model selection Ferrie DOI 10/7nt

Using acceptance ratio or normalizations enables comparing models.

Quantum filtering Wiebe and Granade 1512.03145

Rejection filtering is a dequantization of quantum filtering using Harrow *et al.* DOI 10/bcz3hc.

Thank you!