



Consistent Parameter Distributions in High-dimensional Spaces: Built One Dimension at a Time

Authors: **Troy Butler** (troy.butler@ucdenver.edu) and Alex Dorio, CU Denver

Contributors/Special Thanks: Timothy Wildey and John D. Jakeman, SNL

Some useful references:

[Combining Push-Forward Measures and Bayes' Rule to Construct Consistent Solutions to Stochastic Inverse Problems, T. Butler, J. Jakeman, T. Wildey, SIAM J. Sci. Comput., 40\(2\), A984-A1011 \(2018\).](#)

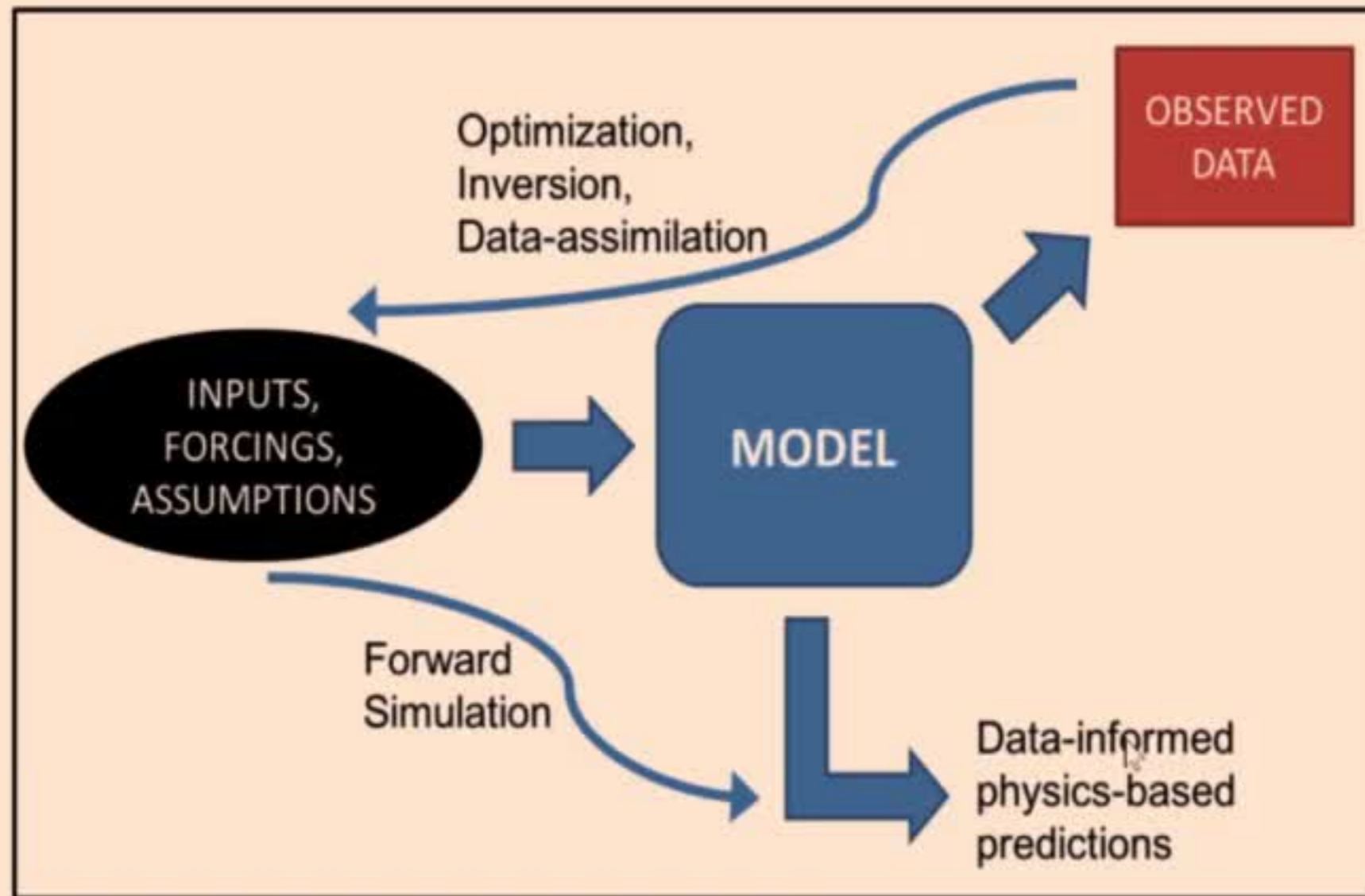
[Optimal \$L_2\$ -norm empirical importance weights for the change of probability measure, S. Amaral, D. Allaire, K. Willcox, Stat. Comput., 27, 625-643 \(2017\).](#)



Some perspective (thanks Mark Twain!)

- Facts are stubborn, but statistics are more pliable.
- Most people use statistics like a drunk man uses a lamppost; more for support than illumination.
- Data is like garbage. You'd better know what you are going to do with it before you collect it.

Data Workflow (cartoon credit: T. Wildey)






The Conclusions! (I may run out of time)

- *Data-consistent* distributions on model inputs have push-forwards that match a distribution on observable model outputs.
- Data-consistent distributions constructed by combining observable and prediction distributions to *update* initial distributions.

I



The Conclusions! (I may run out of time)

- *Data-consistent* distributions on model inputs have push-forwards that match a distribution on observable model outputs.
- Data-consistent distributions constructed by combining observable and prediction distributions to *update* initial distributions.
- Prior work used densities which presents some practical limitations.
 - Expensive models: Computational budgets may prevent sufficient sampling to construct accurate densities.
 - High-dimensional spaces: Slows down convergence rates of naive KDEs.
- The use of optimally weighted empirical CDFs is a promising alternative approximation to data-consistent distributions that implicitly can exploit low-dimensional structure.



Other DCI Presentations

- **CP1 (Monday)** *Sampling Input Parameters of Stochastic Models using Consistent Pull-back Measures*, Tian Yu Yen
- **MS25 (Monday)** *Data-informed Subspace Identification using a Consistent (Data-oriented) Bayesian Method*, Tan Bui-Thanh
- **MS25 (Monday)** *Scalable Approximations for the Consistent Bayes Method*, Brad Marvin
- **MS60 (Monday)** *Exploiting Low-dimensional Structure to Efficiently Perform Stochastic Inference for Prediction*, Timothy Wildey





Notation and Terminology

- $\lambda \in \Lambda$ = model inputs referred to as **model parameters**.
- $Q(\lambda)$ = measurable model outputs referred to as **quantities of interest (QoI)**.
- $D = Q(\Lambda)$ denotes the set of observable data that can be **predicted by the model**.
 - q is used to denote a single datum.



Notation and Terminology

- $\lambda \in \Lambda$ = model inputs referred to as **model parameters**.
- $Q(\lambda)$ = measurable model outputs referred to as **quantities of interest (QoI)**.
- $\mathcal{D} = Q(\Lambda)$ denotes the set of observable data that can be **predicted by the model**.
 - q is used to denote a single datum.

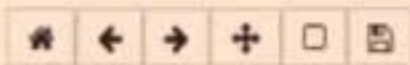
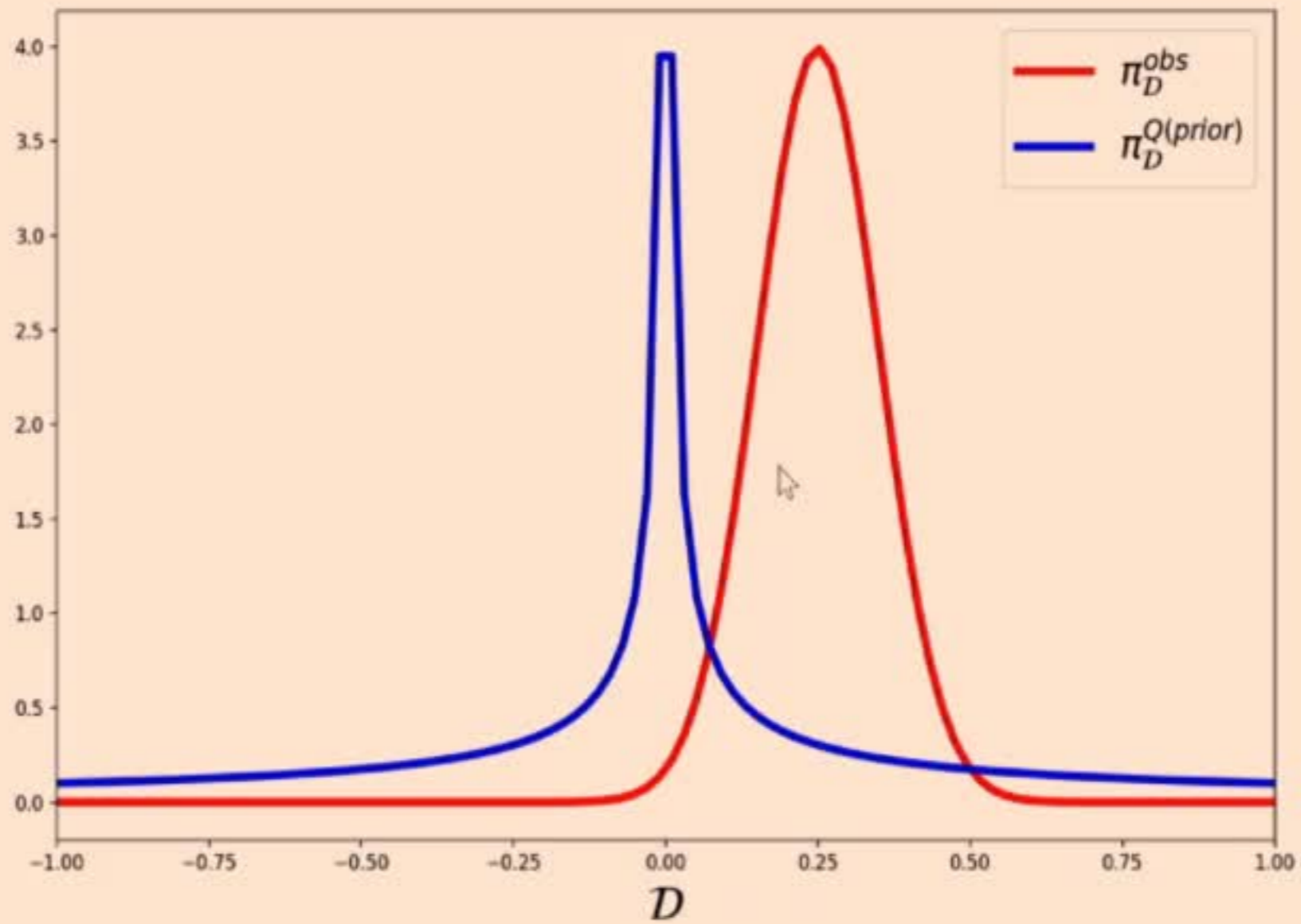




A simple example: The Densities

- $\pi_{\Lambda}^{prior} \sim U([-1, 1])$
- $\pi_D^{obs} \sim N(\mu, \sigma^2)$
 - Initially take $\mu = 0.25$ and $\sigma = 0.1$.
- $\pi_D^{Q(prior)}$
 - Known in this case.

Figure 1

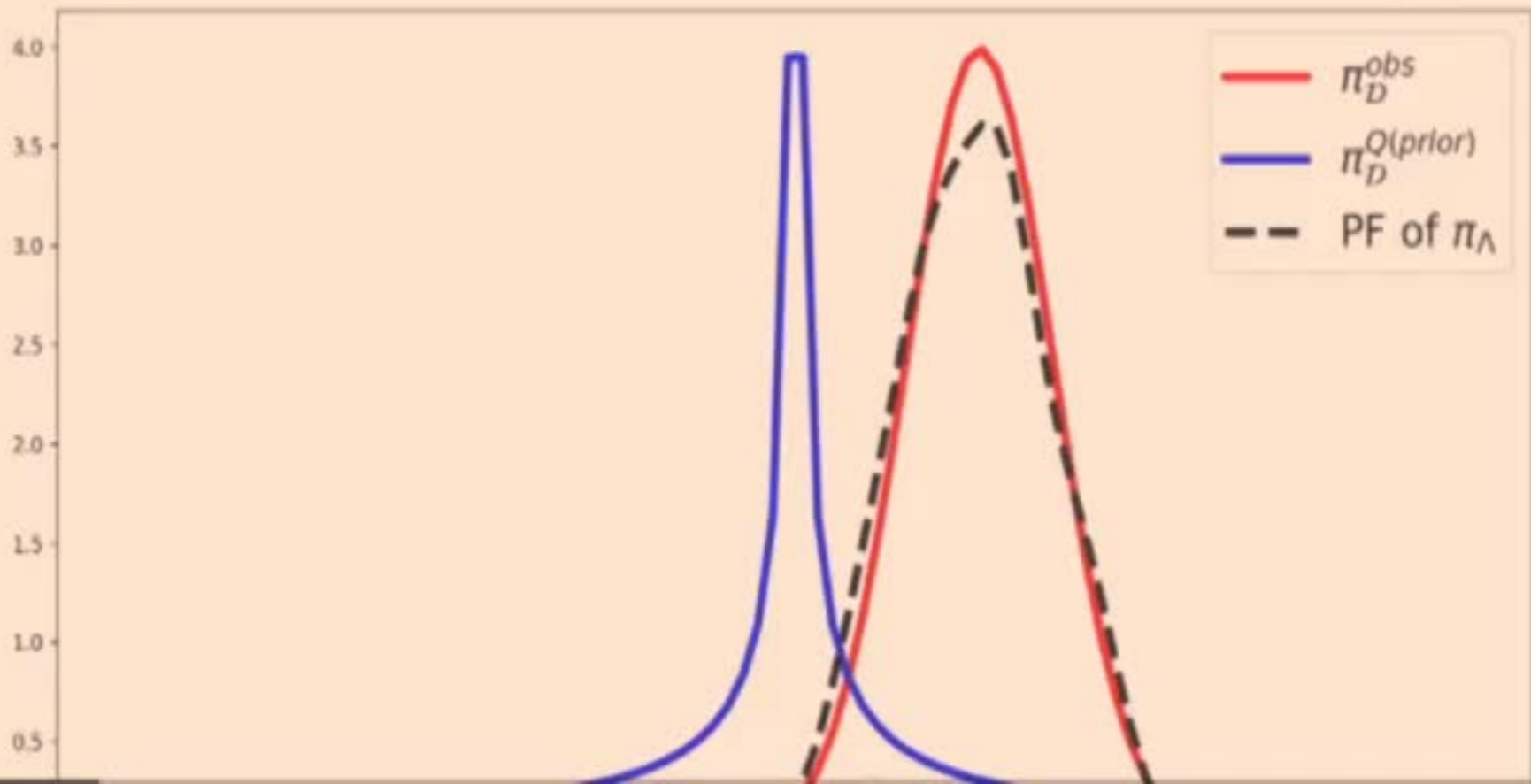


x=0.186679 y=1.7823

```
Out[6]: ((-1, 1),  
Text(0.5,0,' $\mathcal{D}$ '),  
<matplotlib.legend.Legend at 0x1bce7eca8d0>,  
None)
```



Figure 4





A statistical Bayesian perspective

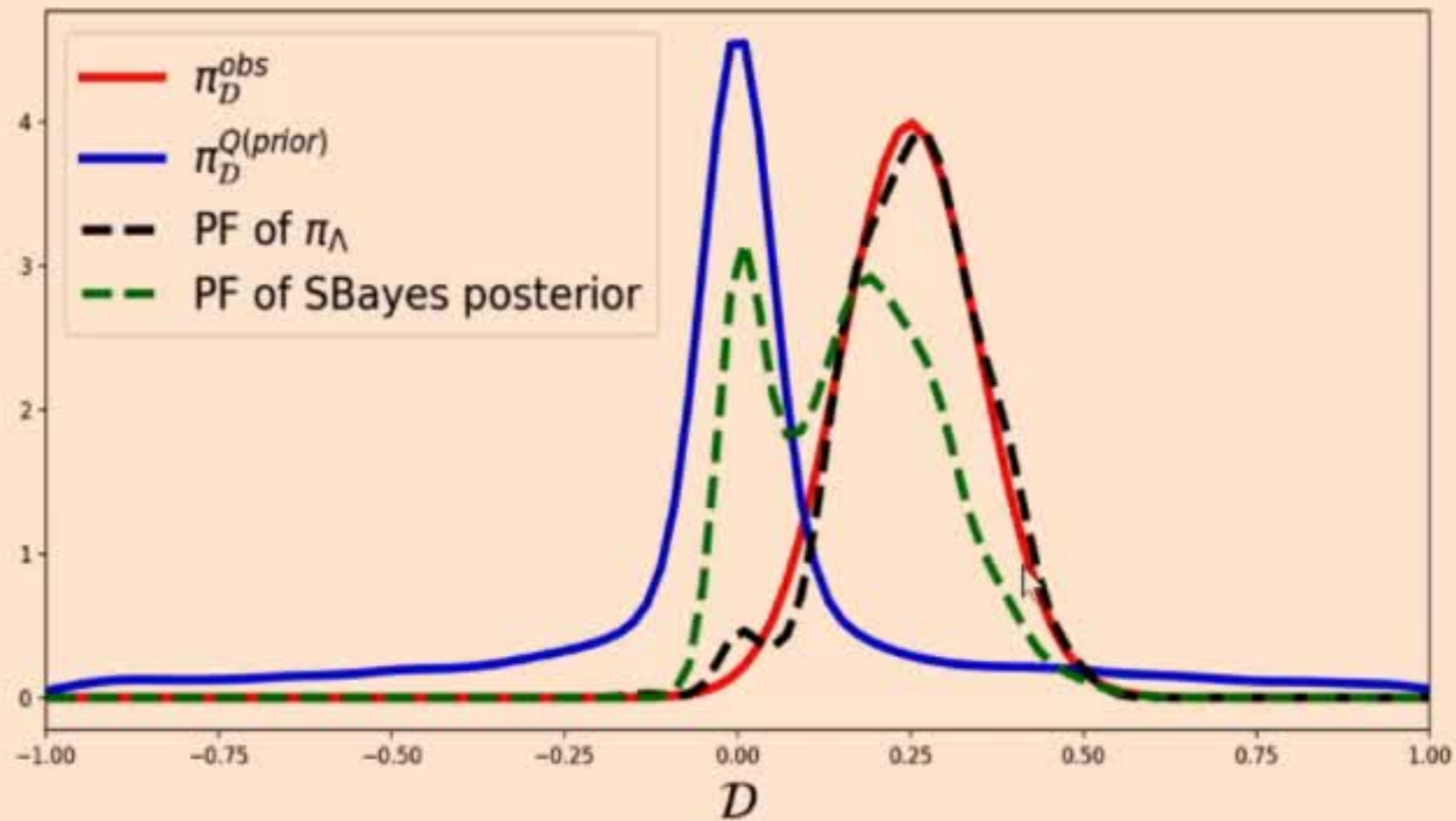
$$\pi_{\Lambda}(\lambda | q) = \pi_{\Lambda}^{prior} \frac{L(q | \lambda)}{\mathcal{C}}$$



A simple example: Comparison to statistical Bayesian

We use the same prior and make the data likelihood function match the observed density.

Figure 7



x=0.410063 y=0.930387



```
Out[21]: ((-1, 1),  
Text(0.5,0, '$\\mathcal{D}$'),  
<matplotlib.legend.Legend at 0x1bce7f89e10>,  
None)
```



Issues with densities

- Convergence rates of naive KDE negatively impacted by dimension.
- May need at least several hundred (if not thousands) of samples in even low dimensions (so iterative updates using rejection sampling may be problematic).

We summarize results for a 100D example from [Convergence of Probability Densities using Approximate Models for Forward and Inverse Problems in Uncertainty Quantification, T. Butler, J. Jakeman, T. Wildey, SIAM J. Sci. Comput., 40\(5\), A3523-A3548. \(2018\)](#)



Discretized PDE Example

Model Setup:

$$\begin{cases} -\nabla \cdot (K(\lambda)\nabla u) = 0, & (x, y) \in \Omega = (0, 1)^2, \\ u = 1, & x = 0, \\ u = 0, & x = 1, \\ K(\lambda)\nabla u \cdot \mathbf{n} = 0, & y = 0 \text{ and } y = 1. \end{cases}$$

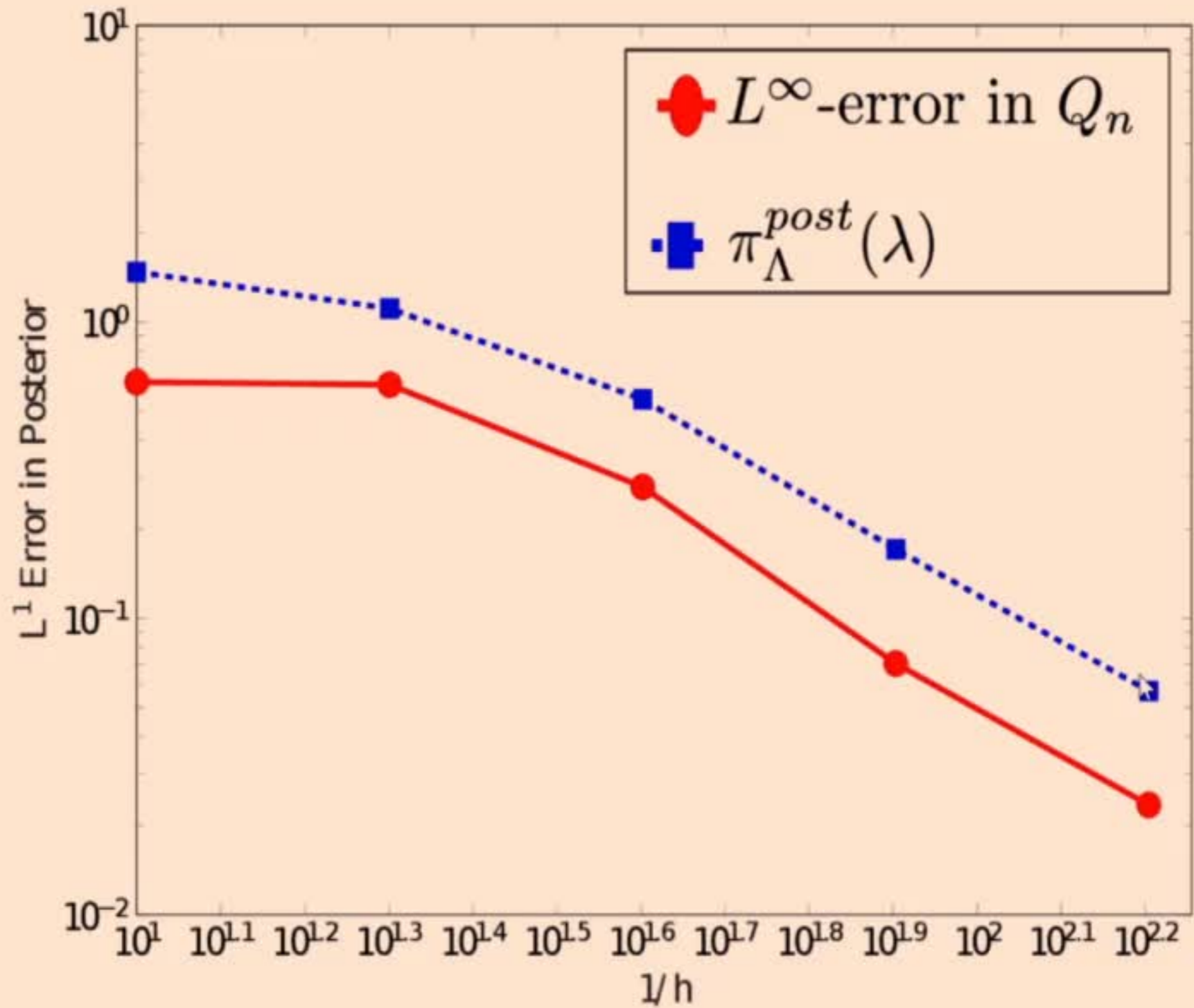
- $Y = \log K$ and $Y(\lambda) = \bar{Y} + \sum_{i=1}^{\infty} \xi_i(\lambda) \sqrt{\eta_i} f_i(x, y)$.
 - Truncate at 100 terms and $\pi_{\Lambda}^{prior} \sim N(0, I)$.


Numerical Setup:

- Continuous piecewise bilinear FEM on uniform spatial grid.
 - $h = 1/10, 1/20, 1/40, 1/80, 1/160$.
- Use 1E3 samples from π_{Λ}^{prior} .




Discretized PDE Example: Posterior Convergence ($\pi_D^{obs} \sim N(0.7, 1E-4)$)






Discretized PDE Example: Wait...accept/reject worked in 100D?

Remember, the computations are really taking place in the 1D data space!





Discretized PDE Example: Wait...accept/reject worked in 100D?

Remember, the computations are really taking place in the 1D data space!


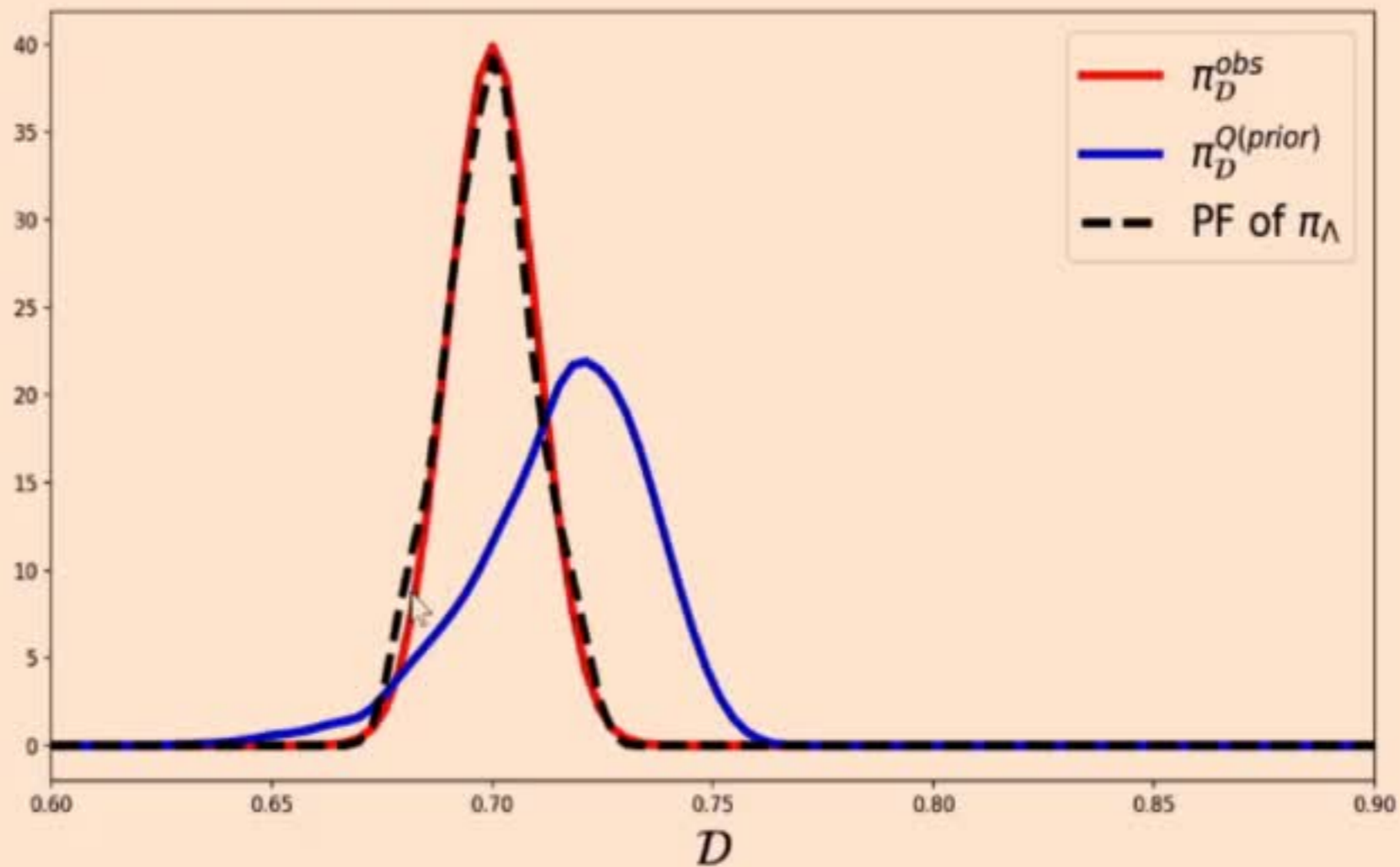


Figure 8

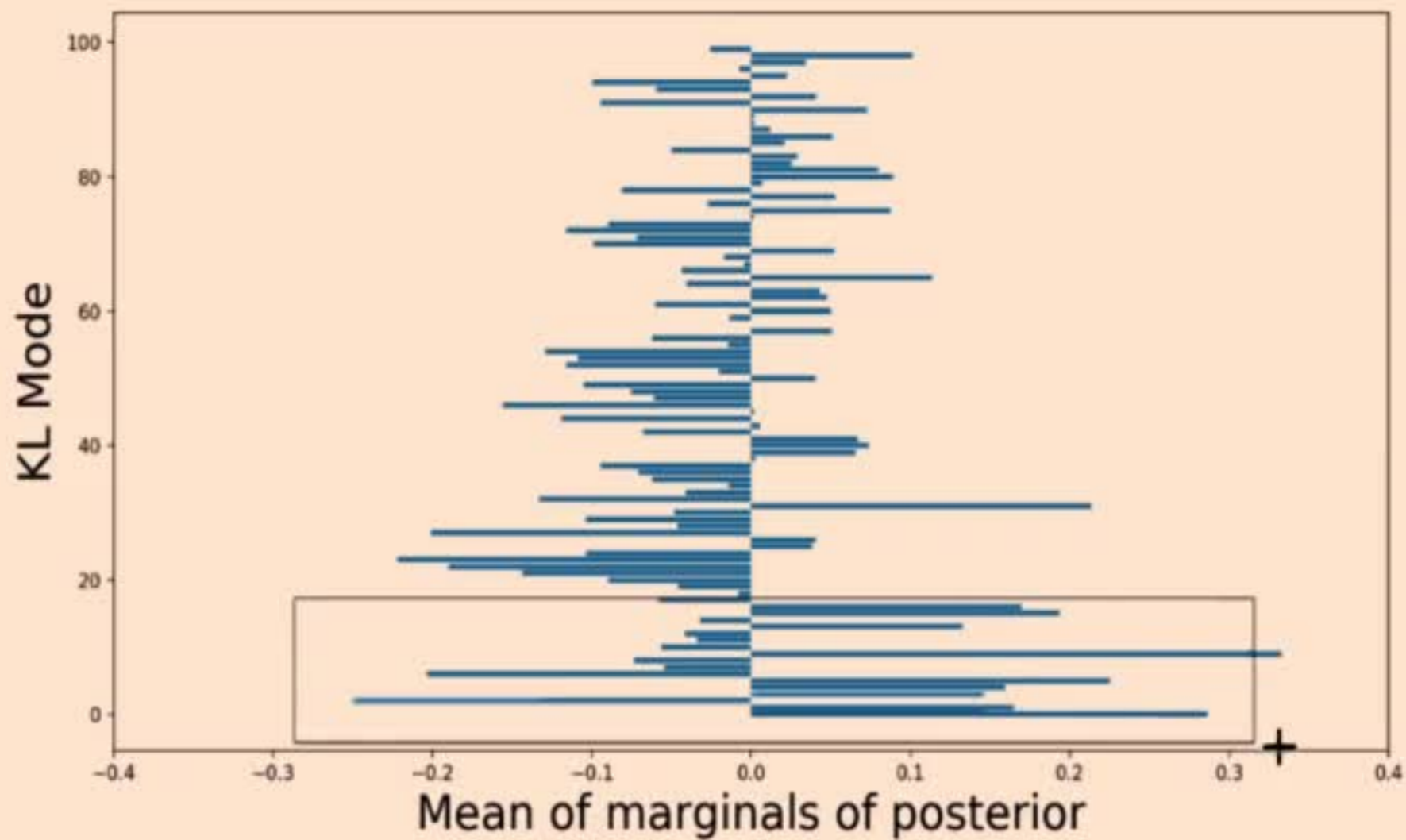


x=0.681448 y=8.82596

```
Out[24]: ((0.6, 0.9),  
Text(0.5,0,' $\mathcal{D}$ '),  
<matplotlib.legend.Legend at 0x1bce8569b00>,  
None)
```



Figure 9



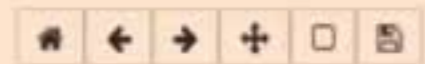
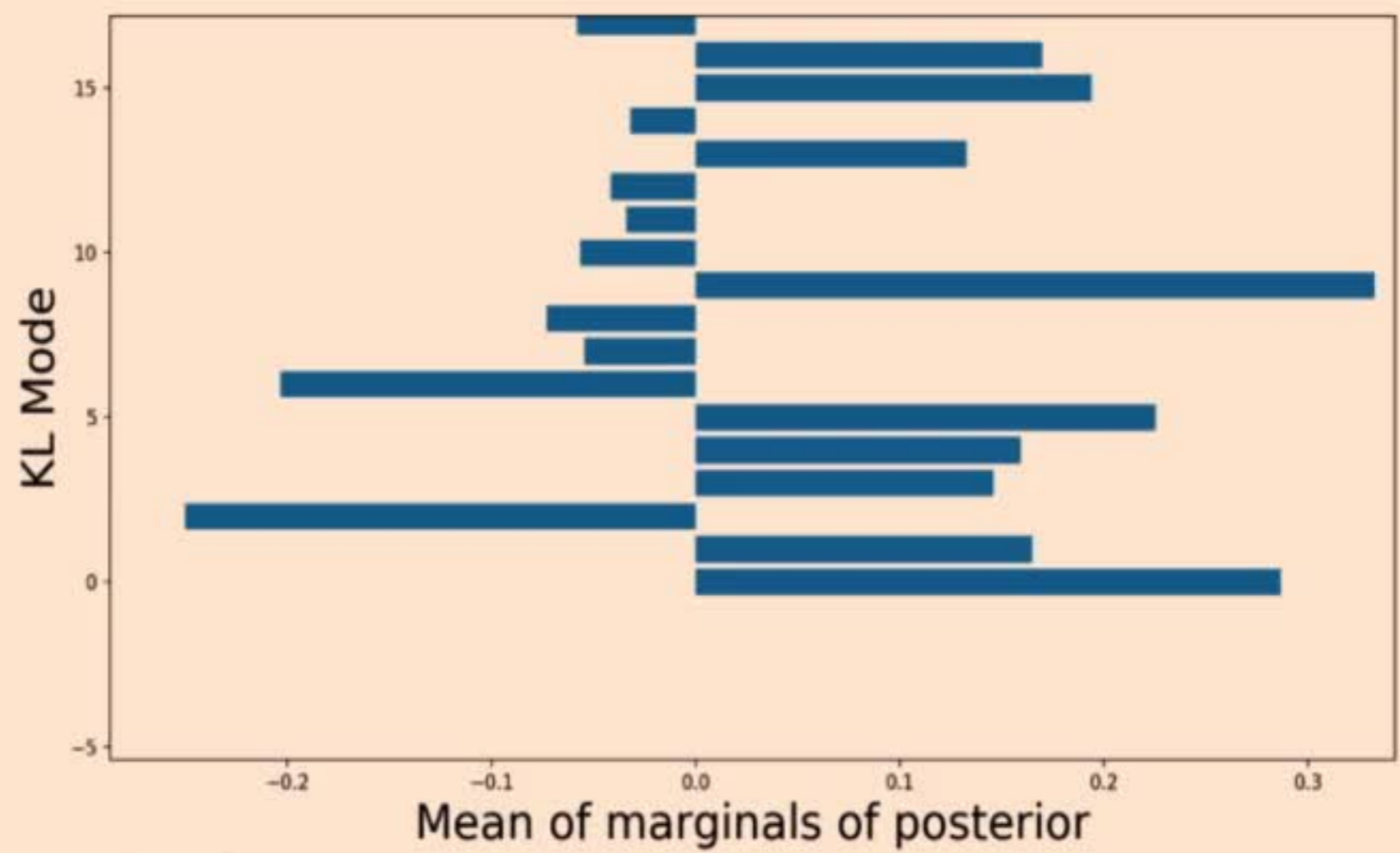
zoom rect. x=0.315649 y=-4.2802

Out[25]: (Text(0,0.5,'KL Mode'), None)





Figure 9



zoom rect

Out[25]: (Text(0,0.5,'KL Mode'), None)





So what exactly is the problem with dimension?

It all depends on which space Λ or \mathcal{D} has the **large** dimension.





Convergence and Dimension: An Example

Setup:

$$Q(\lambda) = (\lambda - \mu)^\top C^{-1}(\lambda - \mu),$$

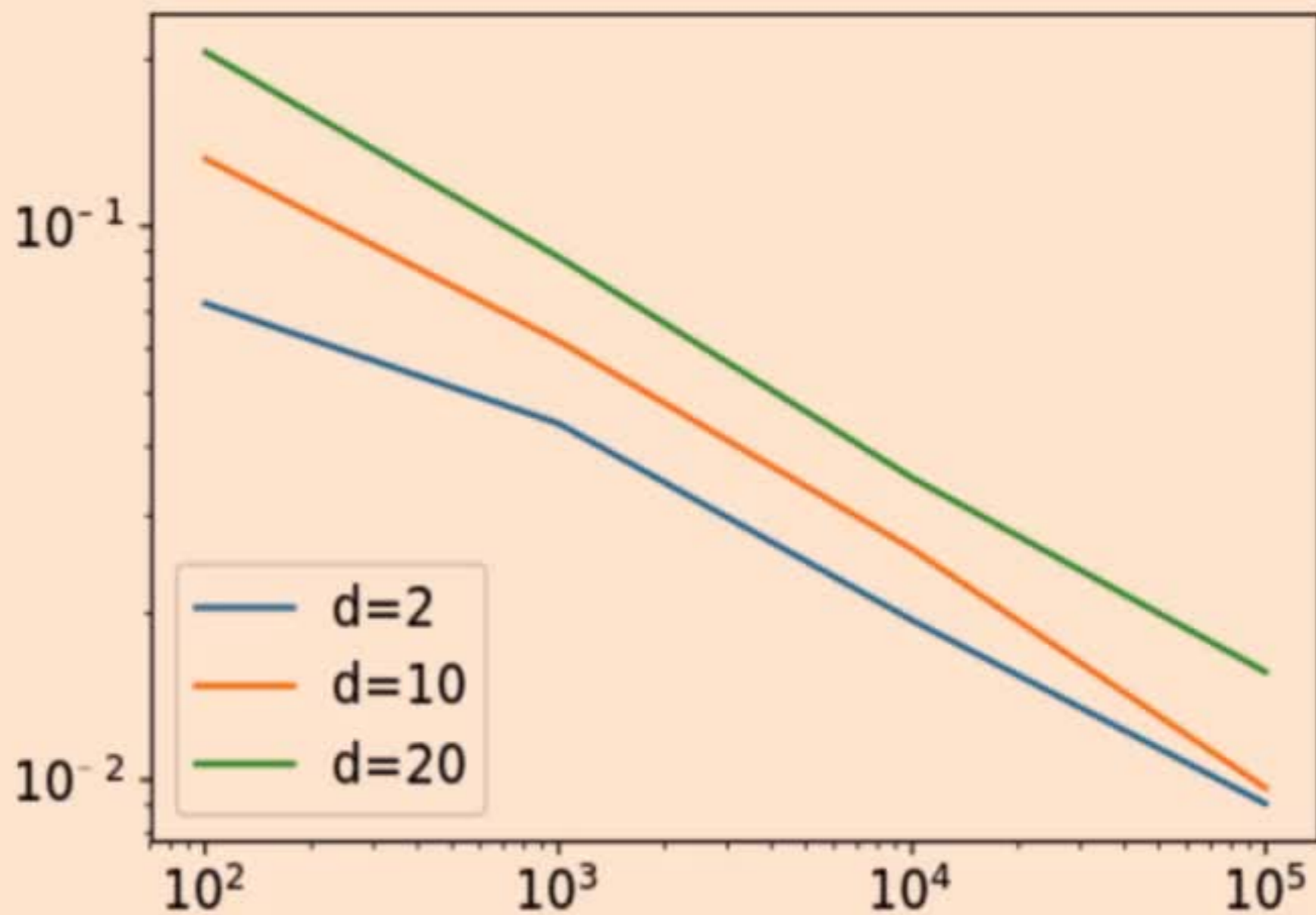
$$\pi_D^{\text{obs}} \sim U([a, b])^*$$

Prior and Push-forward:

$$\boxed{\Lambda = \mathbb{R}^d} \quad \text{and} \quad \pi_\Lambda^{\text{prior}} \sim N(\mu, C) \Rightarrow \pi_D^{Q(\text{prior})} \sim \chi^2(d)$$

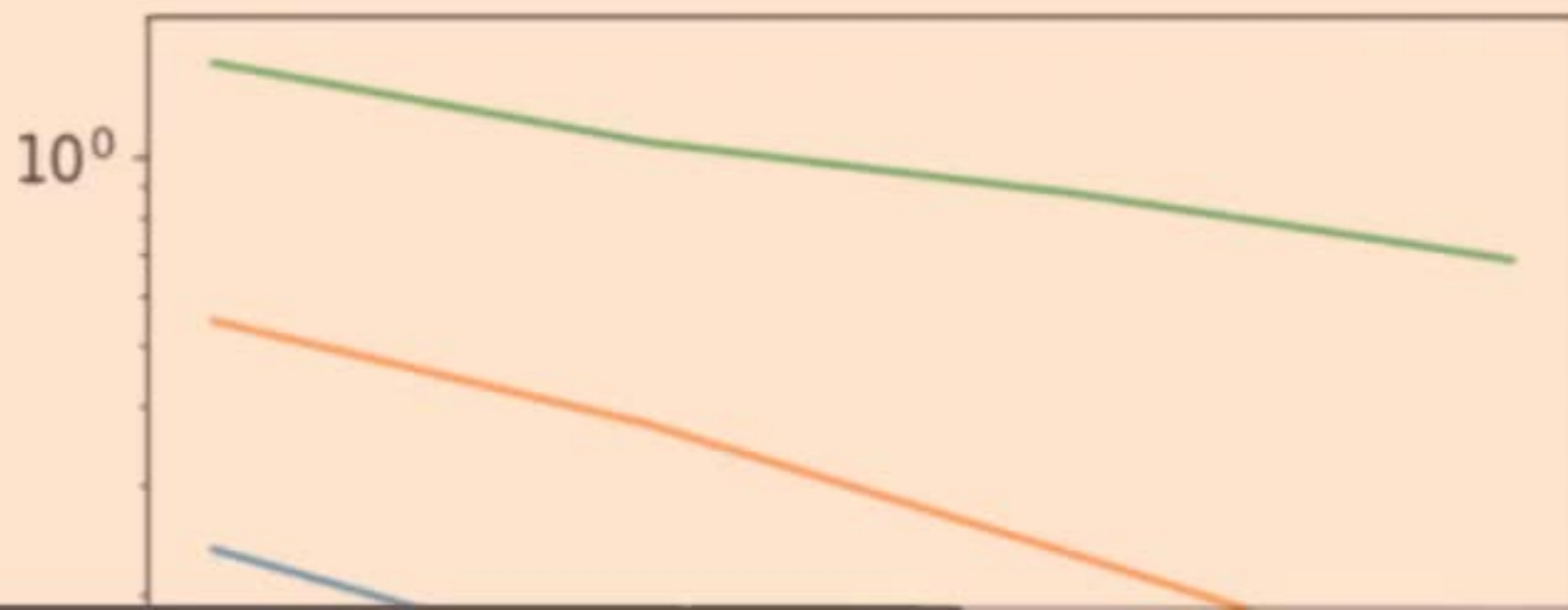
* a and b chosen as the 40th and 60th percentiles of $\pi_D^{Q(\text{prior})}$ for each d

Approx. error in L^1 of updated density vs. Num. Samples used in GKDE.



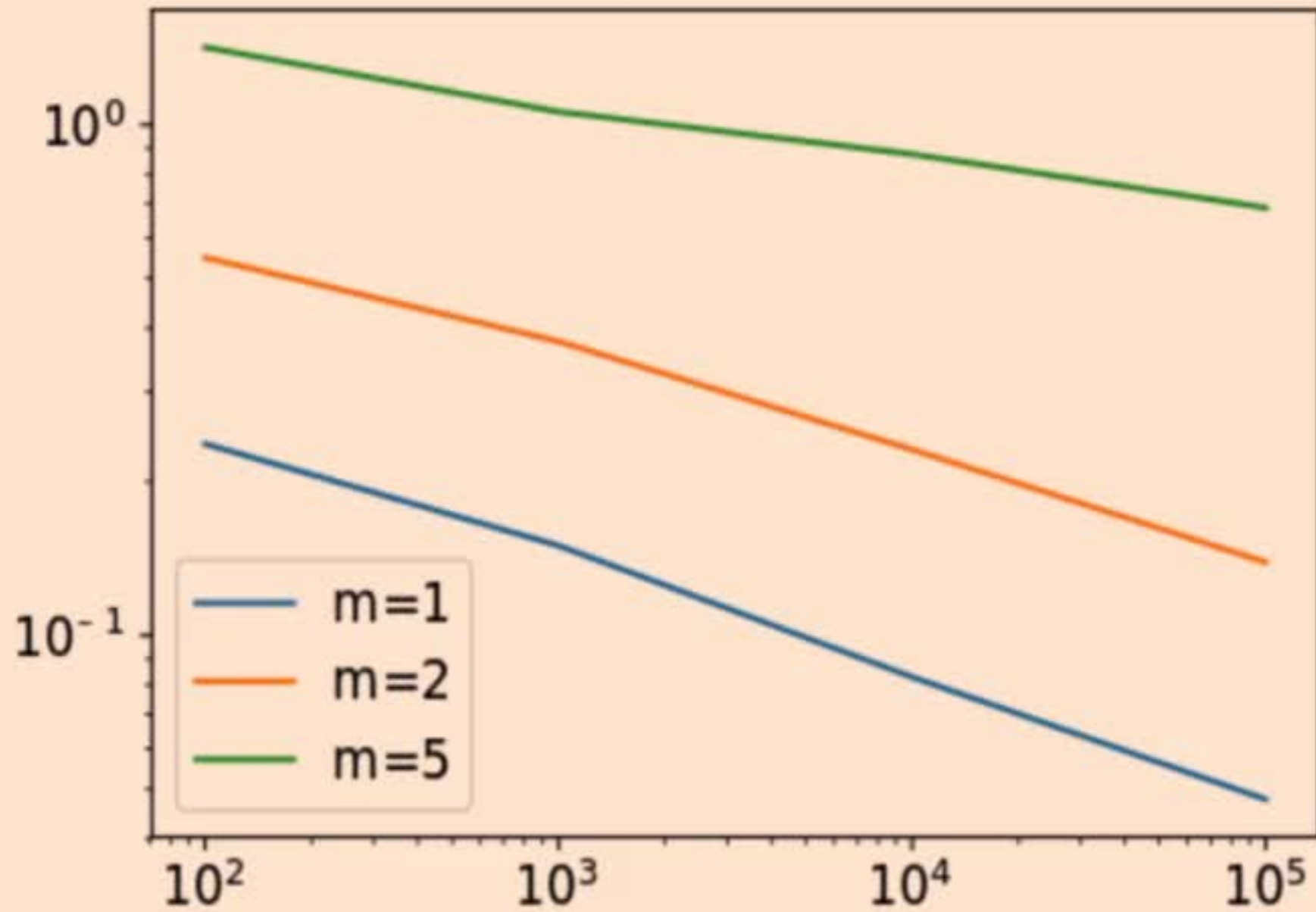
* a and b chosen so Lebesgue measure of support of π_D^{obs} is constant for each m .

Approx. error in L^1 of updated density vs. Num. Samples used in GKDE.



?

Approx. error in L^1 of updated density vs. Num. Samples used in GKDE.





Can we exploit low-dimensional structure?

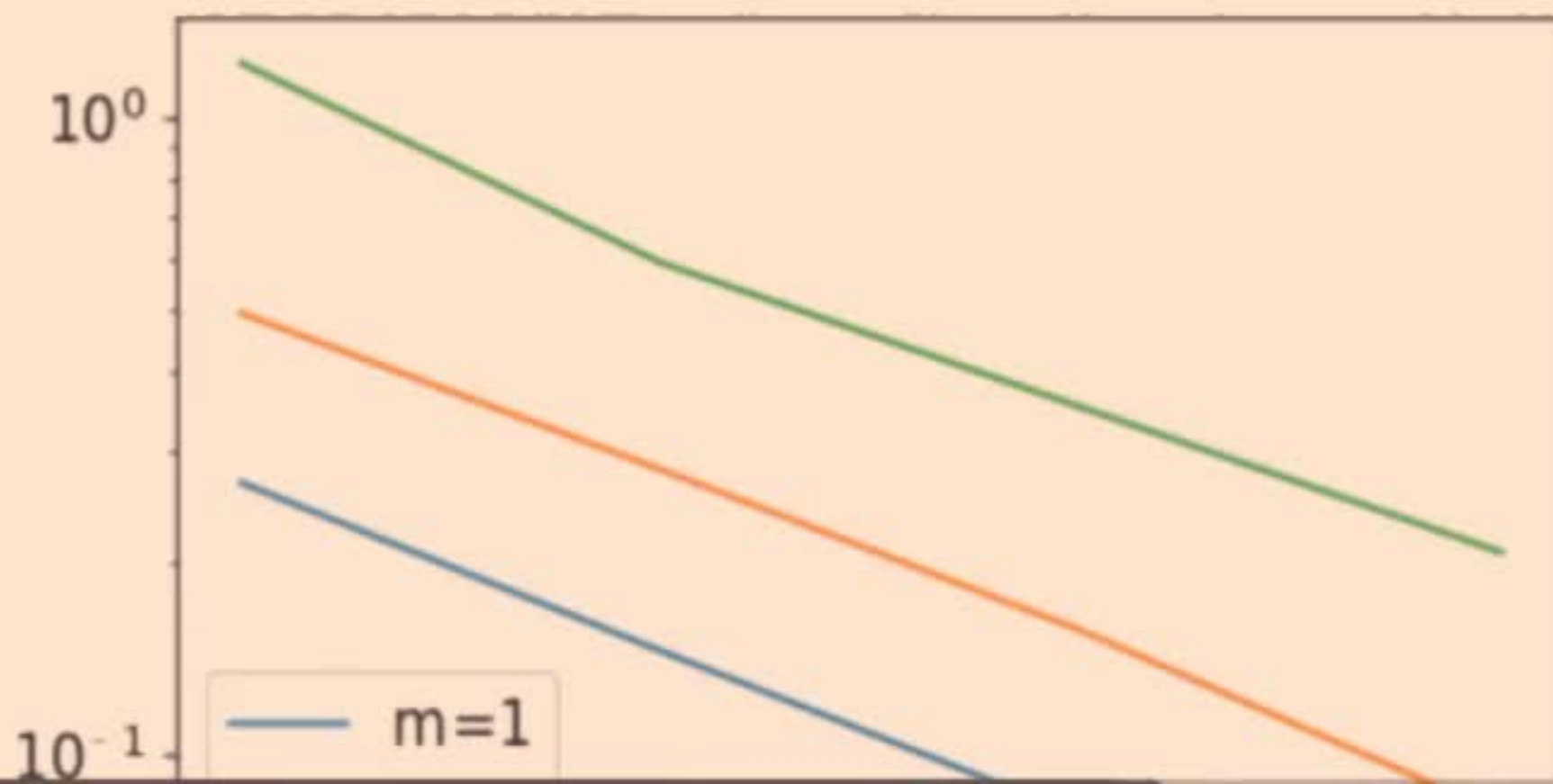
If $\pi_D^{obs}(Q) = \prod_{1 \leq i \leq m} \pi_{D_i}^{obs}(Q_i)$ and $\pi_D^{Q(prior)}(Q) = \prod_{1 \leq i \leq m} \pi_{D_i}^{Q_i(prior)}(Q_i)$

then $\pi_\Lambda(\lambda) = \pi_\Lambda^{prior}(\lambda) \prod_{1 \leq i \leq m} \frac{\pi_{D_i}^{obs}(Q_i(\lambda))}{\pi_{D_i}^{Q_i(prior)}(Q_i(\lambda))}$.



- We can solve the joint problem by solving several lower-dimensional problems.
- [Functional Assimilation] We can **sequentially update** prior weights as distributions on model observables become available.

Approx. error in L^1 of updated density vs. Num. Samples used in GKDE (now exploiting structure)





Avoiding density approximation with CDFs

We build upon the work of [Optimal \$L_2\$ -norm empirical importance weights for the change of probability measure](#), S. Amaral, D. Allaire, K. Willcox, *Stat. Comput.*, 27, 625-643 (2017).


The basic idea is to solve a **forward UQ problem** by constructing a constrained quadratic optimization problem in terms of the empirical **push-forward** CDF and a known, or **observed**, CDF.





Avoiding density approximation with CDFs

We build upon the work of [Optimal \$L_2\$ -norm empirical importance weights for the change of probability measure](#), S. Amaral, D. Allaire, K. Willcox, *Stat. Comput.*, 27, 625-643 (2017).

The basic idea is to solve a **forward UQ problem** by constructing a constrained quadratic optimization problem in terms of the empirical **push-forward** CDF and a known, or **observed**, CDF. 



✘ We build upon the work of [Optimal L2 -norm empirical importance weights for the change of probability measure](#), S. Amaral, D. Allaire, K. Willcox, *Stat. Comput.*, 27, 625-643 (2017).

The basic idea is to solve a *forward UQ problem* by constructing a constrained quadratic optimization problem in terms of the empirical *push-forward* CDF and a known, or *observed*, CDF.

Some definitions/notation

(Empirical Weighted) Proposal:
$$F_{prop;w}^n(\mathbf{t}) = n^{-1} \sum_{i=1}^n w_i \mathbb{1}(\mathbf{x}^{(i)} \leq \mathbf{t})$$

?(Empirical) Target:
$$F_{targ}^m(\mathbf{t}) = m^{-1} \sum_{i=1}^m \mathbb{1}(\mathbf{y}^{(i)} \leq \mathbf{t})$$



Some definitions/notation

(Empirical Weighted) Proposal: $F_{prop; \mathbf{w}}^n(\mathbf{t}) = n^{-1} \sum_{i=1}^n w_i \mathbb{1}(\mathbf{x}^{(i)} \leq \mathbf{t})$

(Empirical) Target: $F_{targ}^m(\mathbf{t}) = m^{-1} \sum_{i=1}^m \mathbb{1}(\mathbf{y}^{(i)} \leq \mathbf{t})$

Misfit functional: $J(\mathbf{w}) = [2(\mu(B_{\mathbf{x}}))^2]^{-1} \left\| F_{prop; \mathbf{w}}^n(\mathbf{t}) - F_{targ}^m(\mathbf{t}) \right\|_{L^2(B)}^2$



The (no details) algorithmic summary

$$\begin{aligned} & \left\| F_{prop; \mathbf{w}}^n(\mathbf{t}) - F_{targ}^m(\mathbf{t}) \right\|_{L^2(B)}^2 \\ &= \boxed{\mathbf{w}^\top H_{prop} \mathbf{w} - 2\mathbf{w}^\top \mathbf{b} + \mathbf{1}_m^\top H_{targ} \mathbf{1}_m} \end{aligned}$$

- $H_{prop} \in \mathbb{R}^{n \times n}$ and $H_{targ} \in \mathbb{R}^{m \times m}$ can be written as Hadamard products of matrices involving only 1D integrals.
- Each component of $\mathbf{b} \in \mathbb{R}^n$ can be written as product of 1D integrals.
- Optimal \mathbf{w} found by minimizing $J(\mathbf{w})$ subject to $w_i \geq 0$ for all i and $\sum w_i = n$.

We use `cvxopt` in Python to solve the optimization problem.





Numerical examples

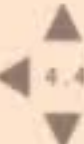
- Verifying alternative approach estimates a consistent solution.
- A high-dimensional example.






Revisiting the Discretized PDE

We compute errors on data space and on the first few KL modes that exhibit greatest differences in means from initial values.





Errors in 1000 samples vs. 240 i.i.d. samples

Errors in data space (absolute and rel. % reduction of weighted vs un-weighted):

un-weighted vs. weighted error: $1.046\text{E-}03$ vs. $6.381\text{E-}06$

rel. improvement: 99.39%


Errors in (certain directions of) parameter space (non-weighted vs. weighted and rel. improvement):

KL mode # 1 CDF errors: $3.329\text{E-}03$ vs. $2.596\text{E-}04$ => 92.20% reduction

KL mode # 3 CDF errors: $1.797\text{E-}03$ vs. $6.762\text{E-}05$ => 96.24% reduction

KL mode # 6 CDF errors: $2.480\text{E-}03$ vs. $1.906\text{E-}04$ => 92.32% reduction

KL mode #10 CDF errors: $7.208\text{E-}03$ vs. $9.611\text{E-}05$ => 98.67% reduction



Errors in 1000 samples vs. 240 i.i.d. samples

Errors in data space (absolute and rel. % reduction of weighted vs un-weighted):

un-weighted vs. weighted error: $1.046\text{E-}03$ vs. $6.381\text{E-}06$

rel. improvement: 99.39%

Errors in (certain directions of) parameter space (non-weighted vs. weighted and rel. improvement):

KL mode # 1 CDF errors: $3.329\text{E-}03$ vs. $2.596\text{E-}04$ => 92.20% reduction

KL mode # 3 CDF errors: $1.797\text{E-}03$ vs. $6.762\text{E-}05$ => 96.24% reduction

KL mode # 6 CDF errors: $2.480\text{E-}03$ vs. $1.906\text{E-}04$ => 92.32% reduction

KL mode #10 CDF errors: $7.208\text{E-}03$ vs. $9.611\text{E-}05$ => 98.67% reduction



Errors in (certain directions of) parameter space (non-weighted vs. weighted and rel. improvement):

λ_1 errors: 6.400E-02 vs. 1.514E-02 \Rightarrow 76.34% reduction

λ_2 errors: 8.314E-02 vs. 1.288E-02 \Rightarrow 84.51% reduction

λ_3 errors: 7.598E-02 vs. 3.435E-02 \Rightarrow 54.79% reduction

λ_4 errors: 6.459E-02 vs. 5.704E-02 \Rightarrow 11.70% reduction

λ_5 errors: 6.286E-02 vs. 5.327E-02 \Rightarrow 15.25% reduction

λ_6 errors: 5.712E-02 vs. 9.002E-02 \Rightarrow 94.24% reduction





Errors in (certain directions of) parameter space (non-weighted vs. weighted and rel. improvement):

λ_1 errors: 0.400E-02 vs. 1.517E-02 \Rightarrow 73.04% reduction

λ_2 errors: 8.314E-02 vs. 1.288E-02 \Rightarrow 84.51% reduction

λ_3 errors: 7.598E-02 vs. 3.435E-02 \Rightarrow 54.79% reduction

λ_4 errors: 6.459E-02 vs. 5.704E-02 \Rightarrow 11.70% reduction

λ_5 errors: 6.286E-02 vs. 5.327E-02 \Rightarrow 15.25% reduction

λ_6 errors: 5.712E-02 vs. 9.003E-03 \Rightarrow 84.24% reduction





Errors in (certain directions of) parameter space (non-weighted vs. weighted and rel. improvement):

λ_6 errors: 5.712E-02 vs. 9.003E-03 \Rightarrow 84.24% reduction

λ_7 errors: 6.713E-02 vs. 2.735E-02 \Rightarrow 59.26% reduction

λ_8 errors: 5.732E-02 vs. 4.150E-02 \Rightarrow 27.59% reduction

λ_9 errors: 7.618E-02 vs. 6.727E-03 \Rightarrow 91.17% reduction

λ_{10} errors: 5.550E-02 vs. 6.385E-03 \Rightarrow 88.50% reduction

