Bayesian Methods for Simulation Optimization

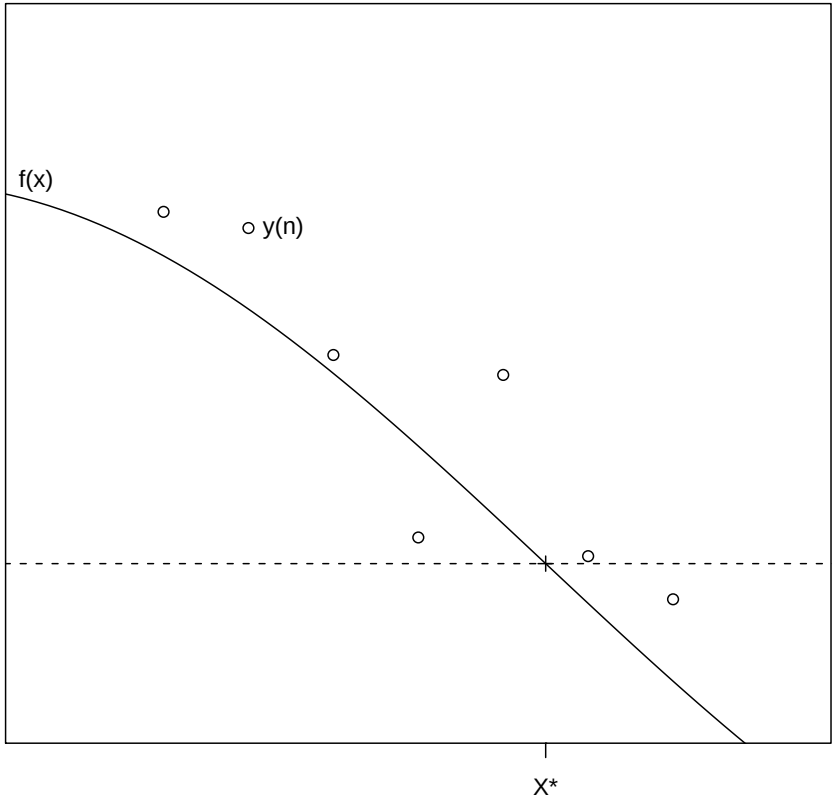
Peter I. Frazier

Operations Research & Information Engineering, Cornell University

Tuesday October 12, 2010
Industrial Engineering and Management Sciences
Northwestern University

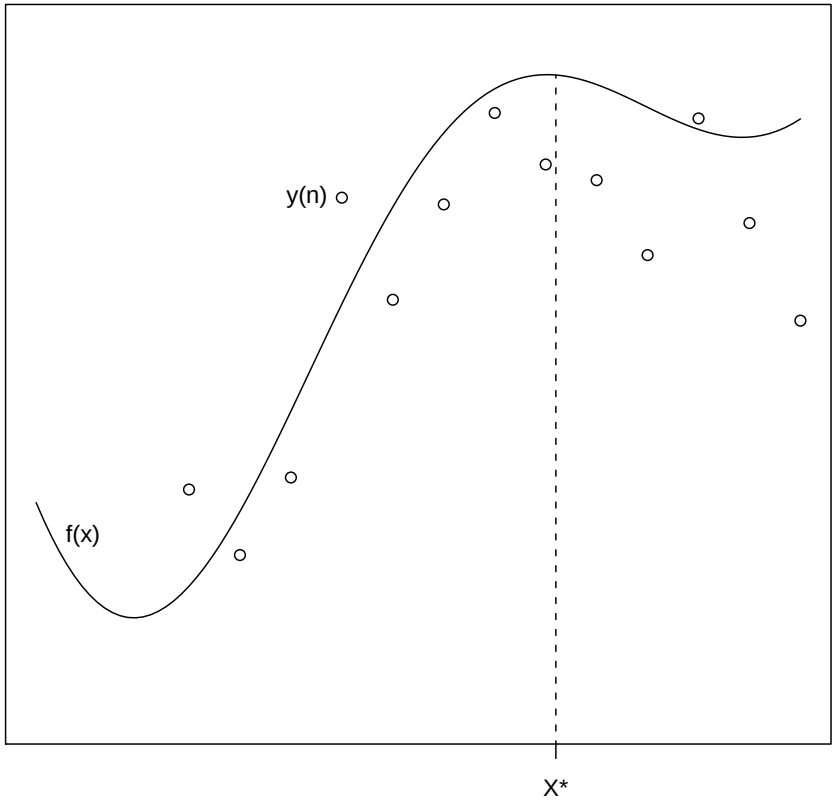
Stochastic Root Finding

Find the root of a monotone function whose values can only be measured with noise. Use as few function evaluations as possible.



Stochastic Global Optimization

Find the maximum of a non-concave function in 1 or more dimensions whose values can only be measured with noise. Use as few function evaluations as possible.



Overview

- Simulation optimization problems can be formulated as problems in sequential Bayesian design of experiments.
- When formulated in this way, they can be analyzed using dynamic programming.
- Novel results:
  - We solve the DP for an idealized version of stochastic root-finding and apply this to the real stochastic root-finding problem.
  - We present a myopic policy motivated by the DP for stochastic global optimization.

Outline

General Framework

1. Before we begin, nature fixes some true function f.
   - e.g., f is the function whose root we are trying to find.
2. For time $n = 1$ up to N, we:
   1. Choose some type of measurement x_n to perform.
   2. Observe some value y_n whose distribution depends on x_n and f.
3. After the final measurement, we make some decision $\widehat{x_*}$ based on the data $x_1, \ldots, x_N$, $y_1, \ldots, y_N$.
4. We pay some penalty $L(\widehat{x_*}, f)$.

Our goal is to choose $x_1, \ldots, x_N, \widehat{x_*}$ adaptively to minimize $L(\widehat{x_*}, f)$.

General Framework

- $\mathbb{E}^{\pi}[L(\widehat{x_*}, f) \mid f]$ is the expected loss when we use policy π and the truth is f.
  - A policy π is a method for choosing each x_{n+1} as a function of $x_{1:n}, y_{1:n}$, and choosing $\widehat{x_*}$ as a function of $x_{1:N}, y_{1:N}$.
- We cannot solve $\inf_{\pi} \mathbb{E}^{\pi}[L(\widehat{x_*}, f) \mid f]$ because the answer depends on f, which we do not know.
- In this talk, we are interested in **average case performance**.

Bayesian General Framework

- The average-case loss under a probability distribution $\mathbb{P}_0$ is

$$\int \mathbb{E}^{\pi}\left[L(\widehat{x_*}, f) \mid f\right] \mathbb{P}_0(df) = \mathbb{E}^{\pi}\left[L(\widehat{x_*}, f)\right]$$

- We call $\mathbb{P}_0$ our "prior".
- Goal: minimize the average-case loss

$$\inf_{\pi} \mathbb{E}^{\pi}\left[L(\widehat{x_*}, f)\right]$$

Bayesian General Framework

- Associated with the prior is a sequence of "posterior" distributions

$$\mathbb{P}_n(f \in \cdot) = \mathbb{P}_0\{f \in \cdot \mid x_{1:n}, y_{1:n}\}.$$

- The problem $\inf_\pi \mathbb{E}^\pi[L(\widehat{x}_*, f)]$ can be viewed as a **Markov decision process**, where the Markov process being controlled is the sequence of posterior distributions $(\mathbb{P}_n : n \geq 0)$.
- As a Markov decision process, its solution is characterized by the **dynamic programming equations**.

Dynamic Programming

- $V_n(\mathbb{P}_n)$ is the **value function**. $\mathbb{P}_n$ at time n.

$$V_n(\mathbb{P}_n) = \inf_{\pi} \mathbb{E}_n^{\pi} \left[L(\widehat{x}_*, f) \right]$$

- The value function satisfies the **dynamic programming equations**:

$$V_N(\mathbb{P}_N) = \inf_{\widehat{x}_*} \mathbb{E}_N \left[L(\widehat{x}_*, f) \right]$$

$$V_n(\mathbb{P}_n) = \inf_{x} \mathbb{E}_n \left[V_{n+1}(\mathbb{P}_{n+1}) \mid x_{n+1} = x \right].$$

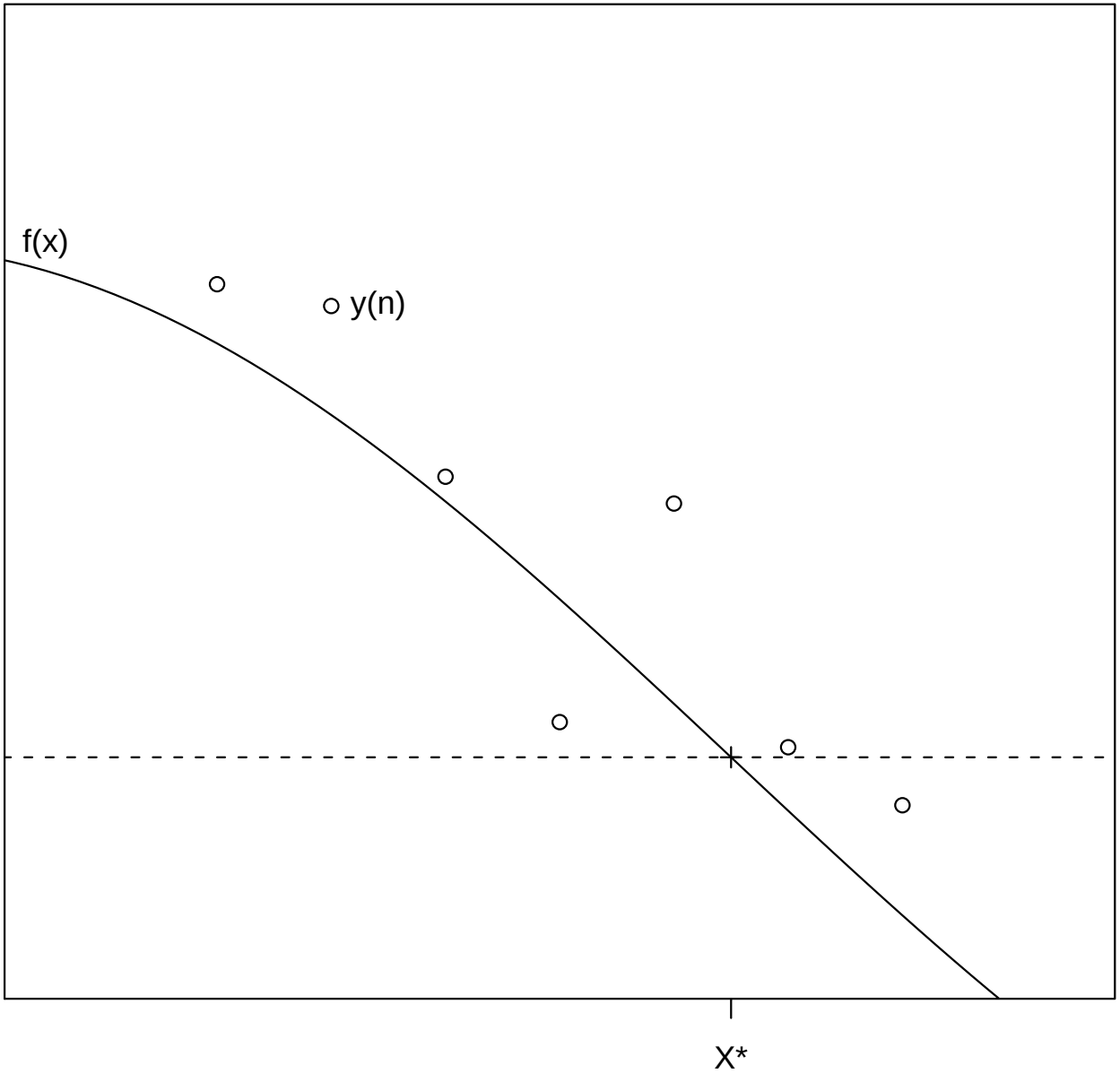
- Any policy whose decisions x_n, $\widehat{x}_*$ attain the infimum is optimal.
- **If we can find the value function, we can find the optimal policy**.

Dynamic Programming

- To find the value function numerically, we would have to compute V_n for every possible value of $(n, \mathbb{P}_n)$.
- This is usually **much too large to solve numerically**.
- Finding analytic solutions to dynamic programs is usually also difficult.
- In this talk, dynamic programming is used as a theoretical tool.
  - We solve one dynamic program analytically.
  - We solve another in a special case, motivating a heuristic solution.

Outline

Stochastic Root-Finding

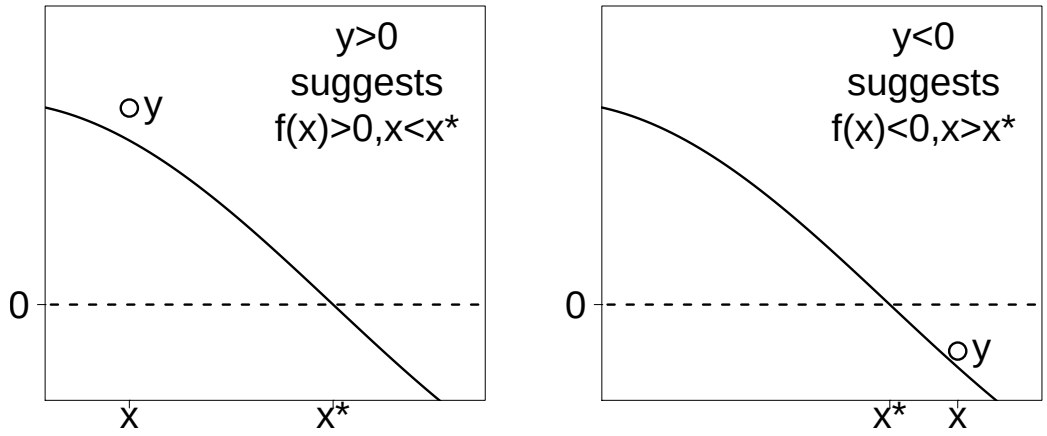


Stochastic Root-Finding

- $f : \mathbb{R} \mapsto \mathbb{R}$ is a decreasing function.
- The only way to evaluate f is via stochastic simulation.
- We observe $y_n = f(x_n) + \varepsilon_n$, where ε_n is independent noise.
- Our goal is to find a root x_*, i.e., a point x_* such that $f(x_*) = 0$.
- Central Question: Given a budget of N measurements, $x_1, \ldots, x_N$, how should we place them to find x_* as accurately as possible?

Stochastic Root-Finding: Model

- Assume we observe only whether y_n is larger or smaller than 0.



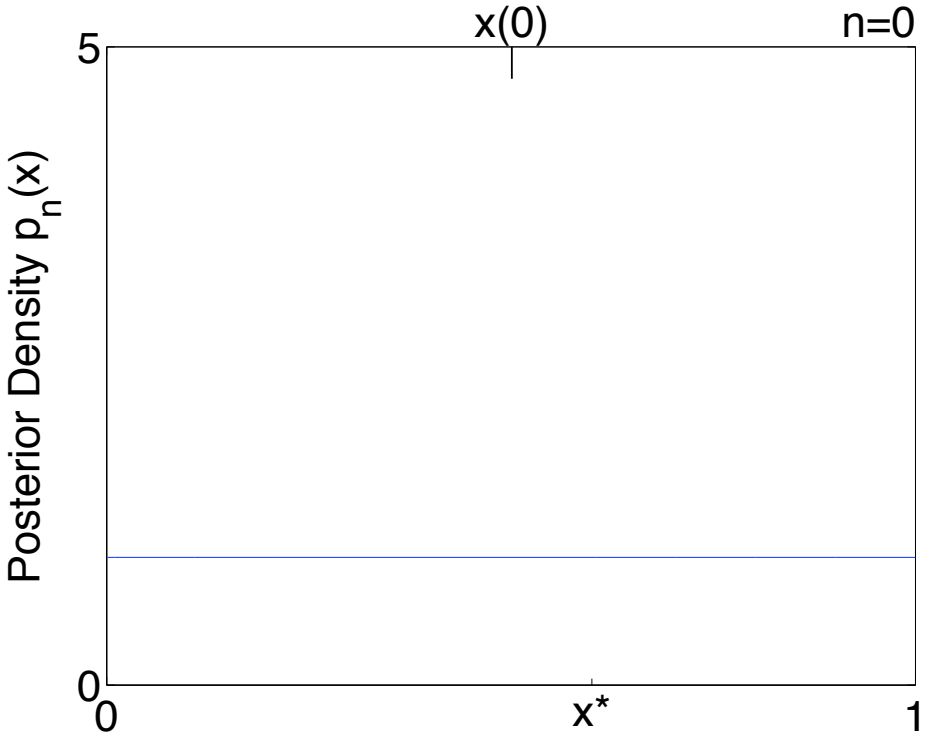
- Assume nature gives the incorrect sign with fixed known probability q:

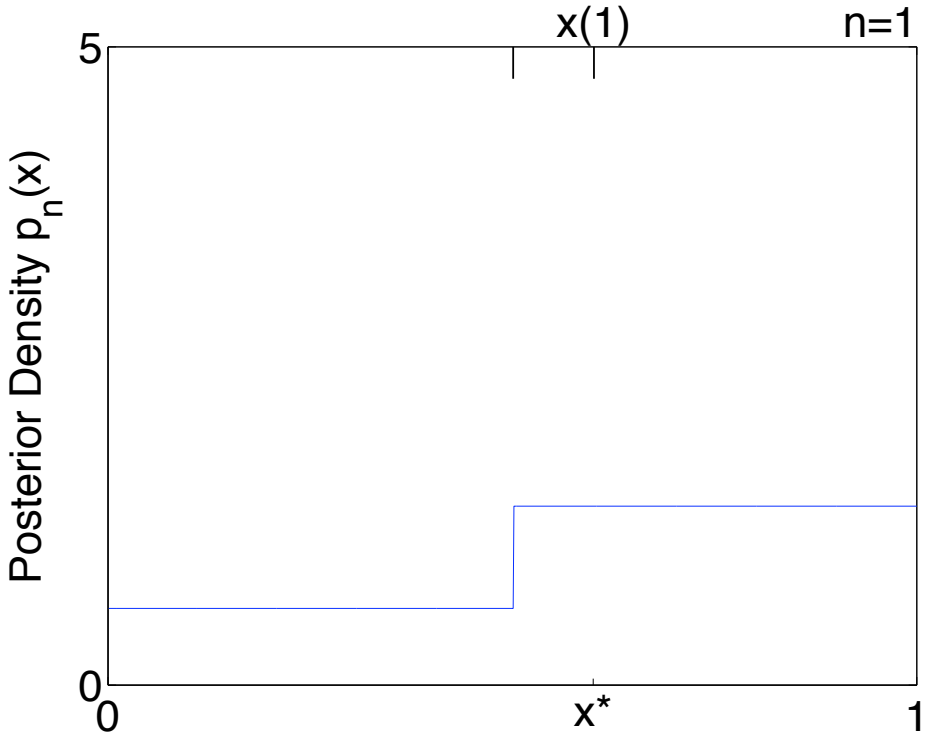
$$\operatorname{sgn}(y_n) = \begin{cases} -\operatorname{sgn}(f(x_n)), & \text{with probability } q, \\ \operatorname{sgn}(f(x_n)), & \text{with probability } 1-q \end{cases}$$

Posterior Distributions

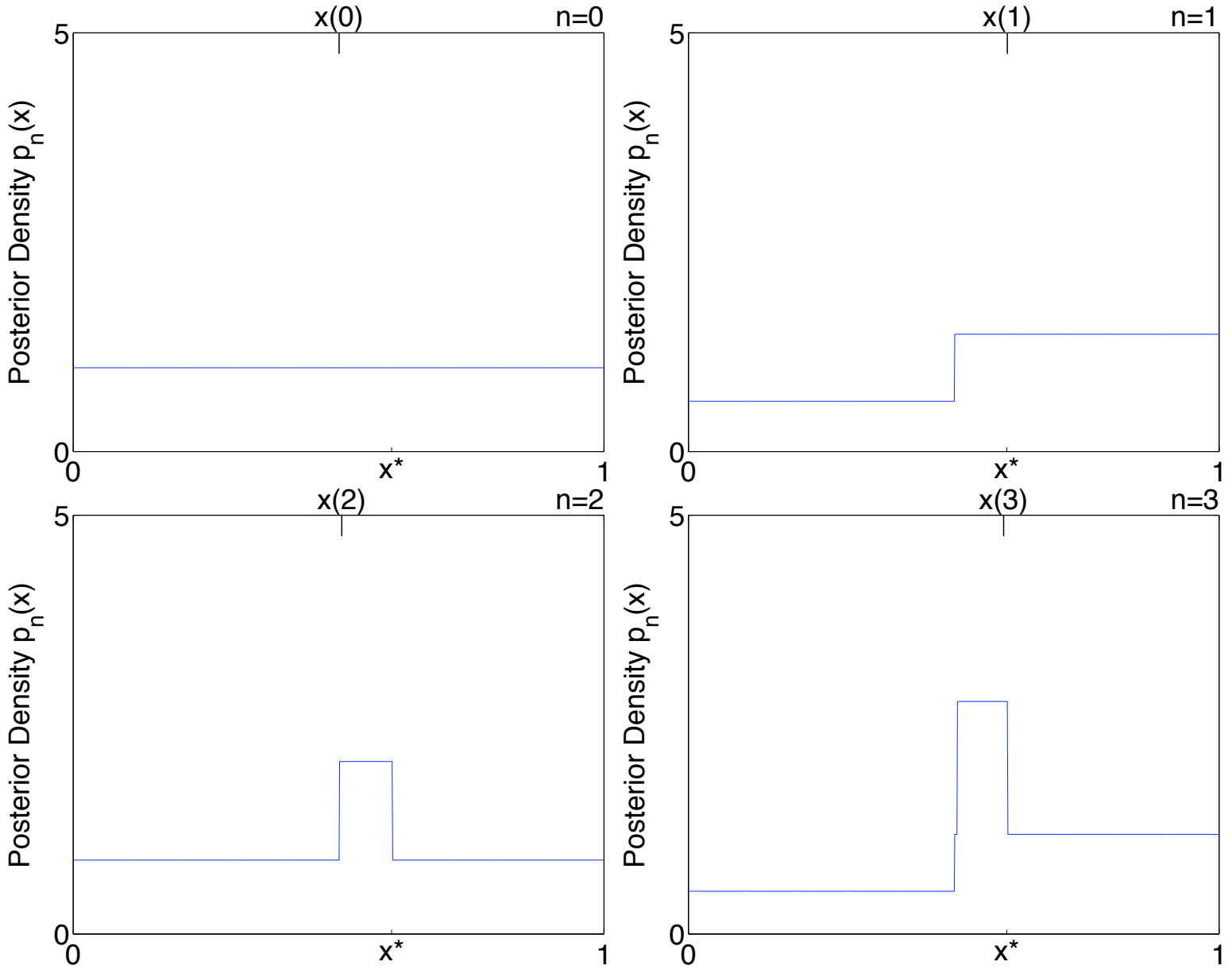
- Place a prior density p_0 on the root x_*, e.g., uniform on $[0,1]$.
- Each measurement x_n produces a new posterior density p_n on x_*:

$$p_n(x) = \mathbb{P}\{X_* \in dx \mid x_{1:n-1}, y_{1:n-1}\}$$





Posterior Distributions



Stochastic Root-Finding

- Our goal is to minimize the final entropy

$$H(p_N) = -\int p_N(x) \log p_N(x)\, dx.$$

- The value function is

$$V_n(p_n) = \inf_{\pi} \mathbb{E}^{\pi} \left[H(p_N) \mid p_n \right],$$

  and the stochastic optimization problem we need to solve is to find the policy π that attains the infimum defining $V_0(p_0)$.
- Bellman's recursion can be written,
  $V_n(p_n) = \inf_{x_n \in [0,1]} \mathbb{E}\left[V_{n+1}(p_{n+1}) \mid p_n \right]$.

Main Result: Bayes Optimality

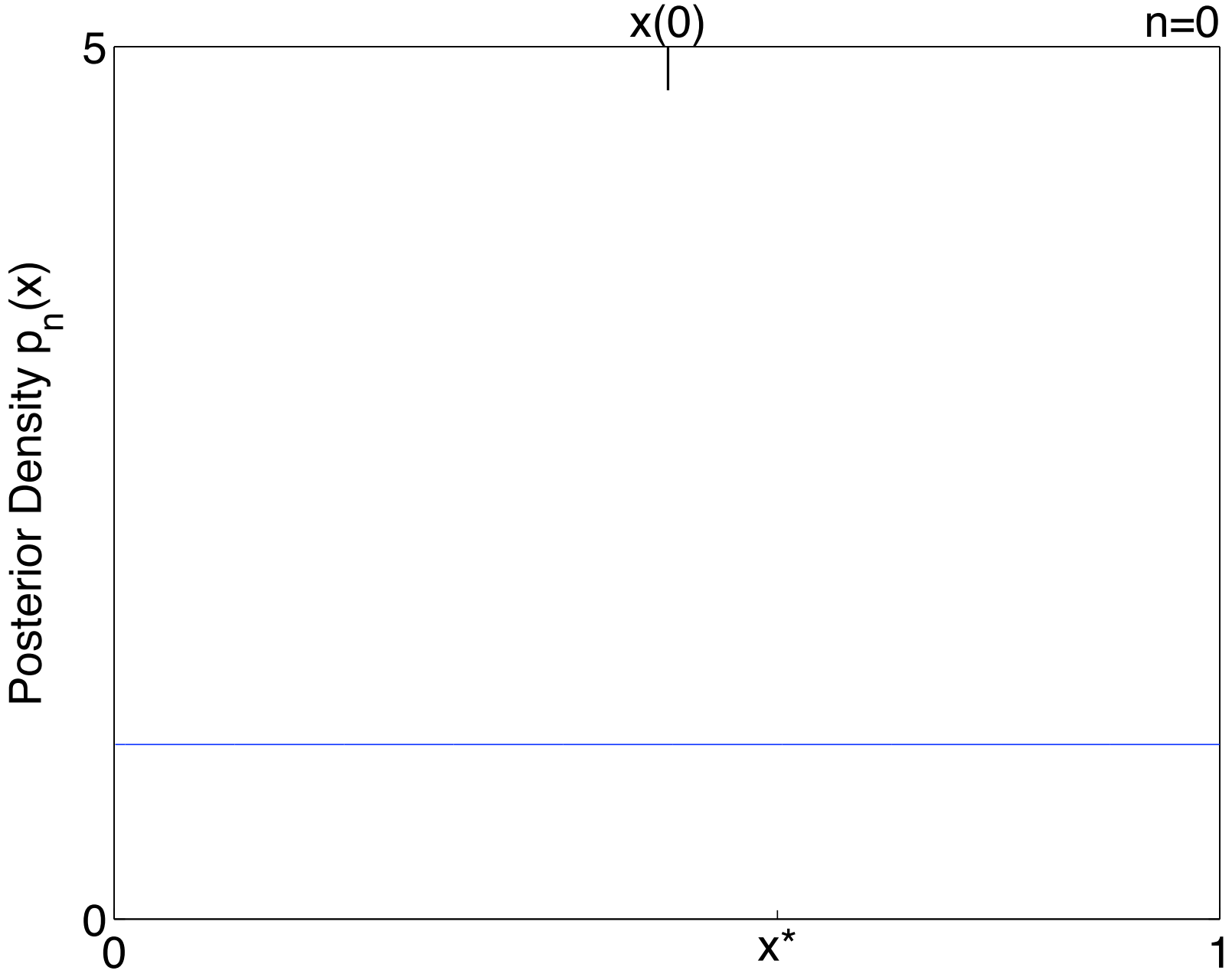
Theorem

The value function can be written explicitly as

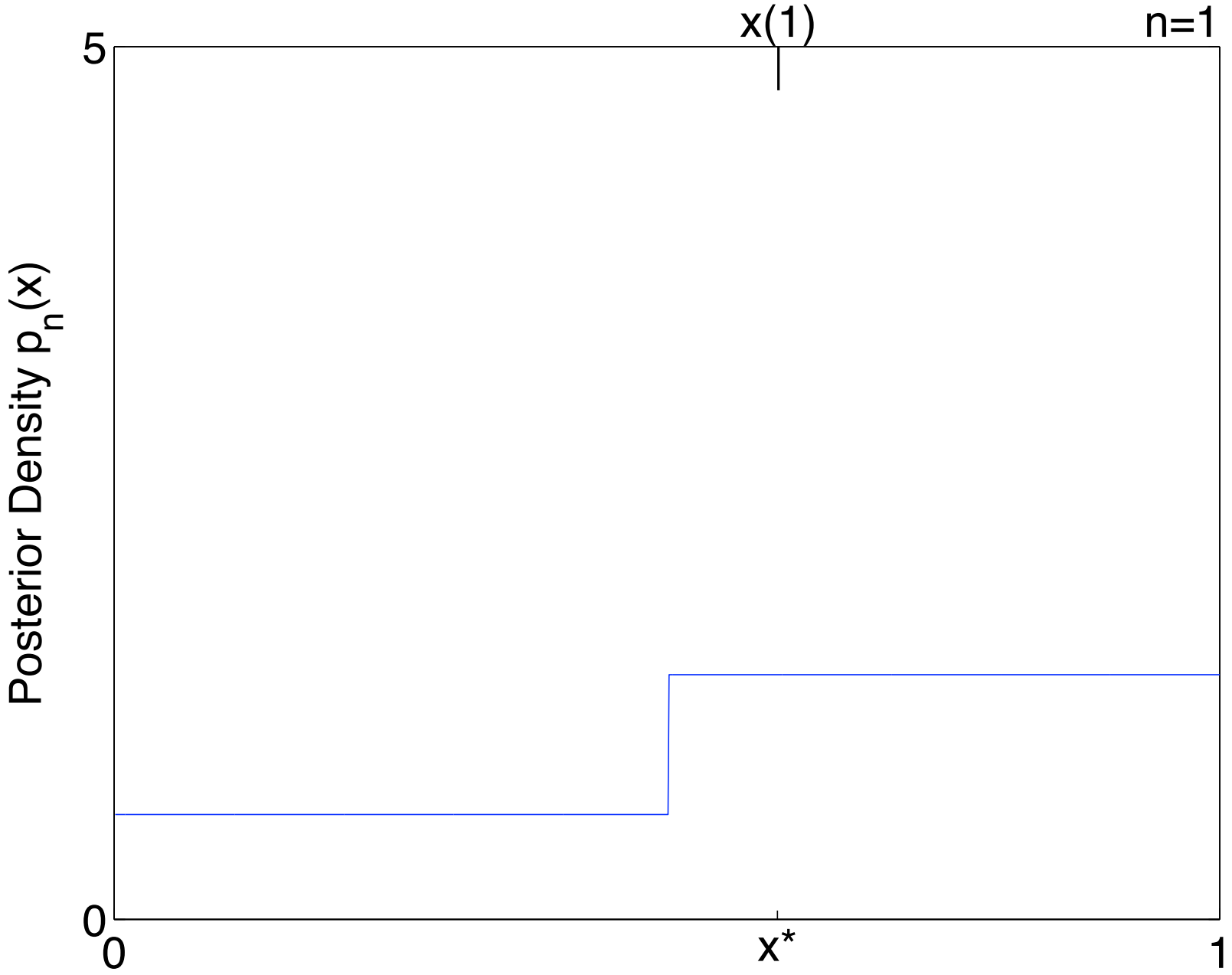
$$V(p_n) = H(p_n) - (N-n)\left[-q\log_2(q) - (1-q)\log_2(1-q)\right],$$

and **the policy that chooses x_n at the median of p_n is optimal**.

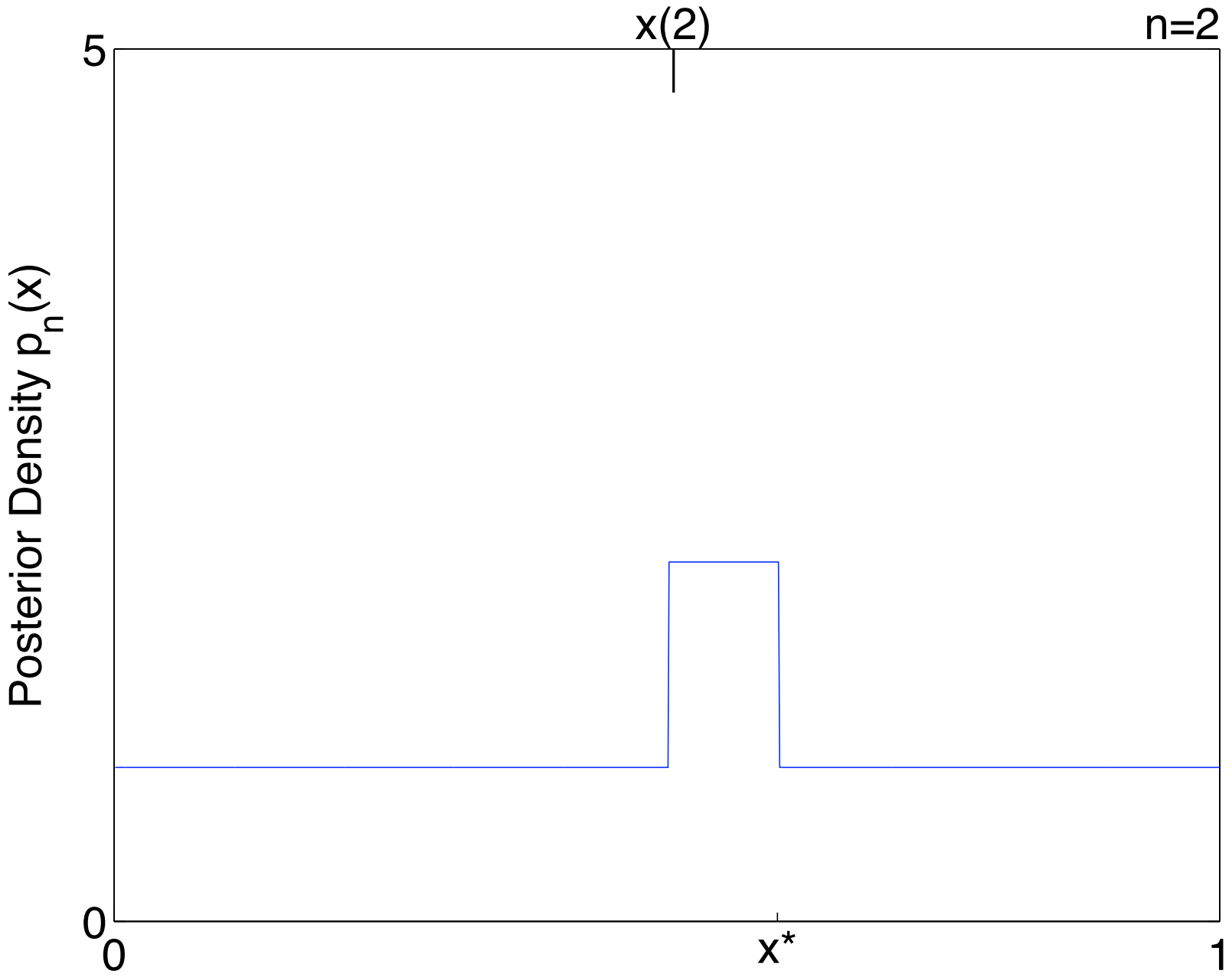
Example



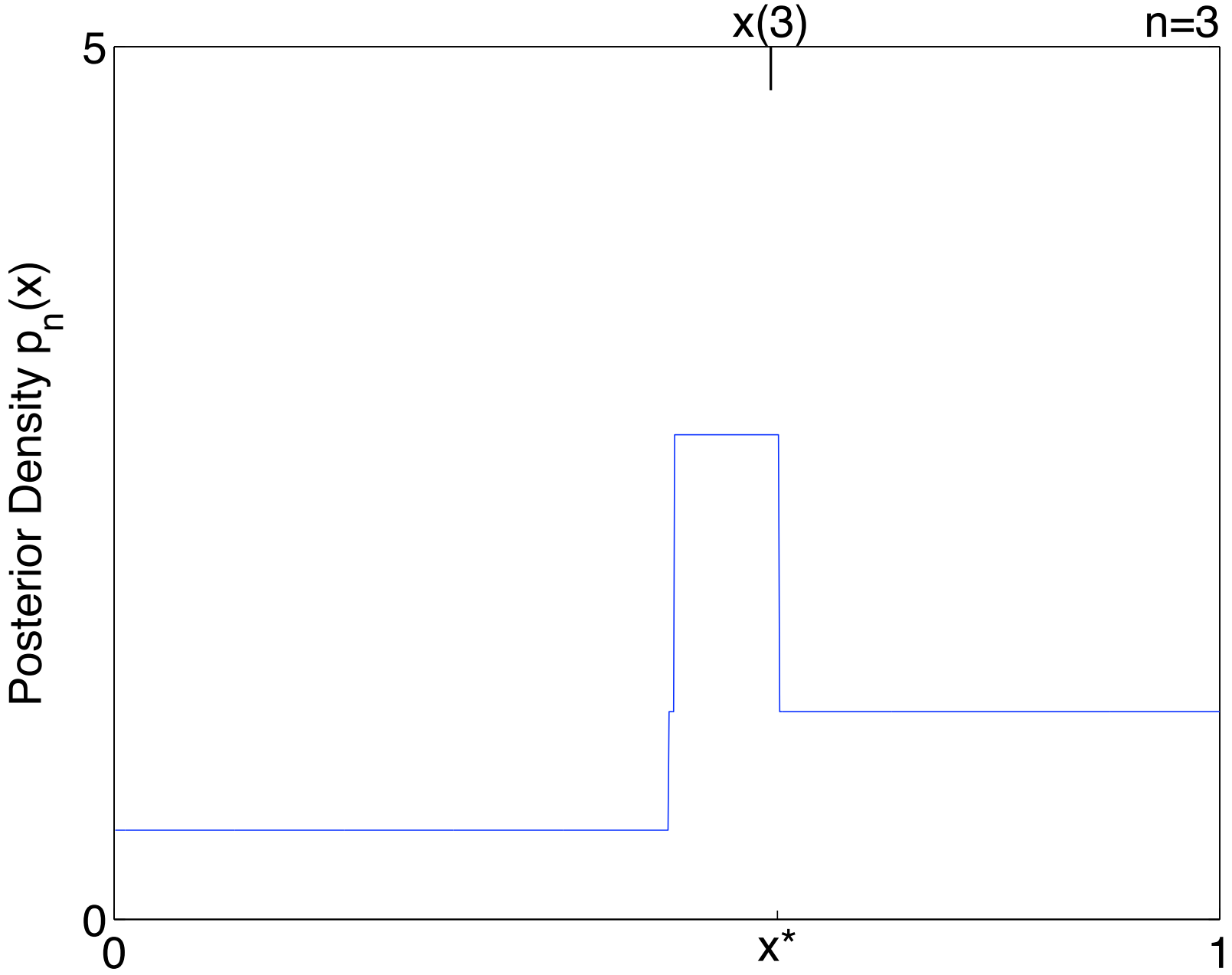
Example



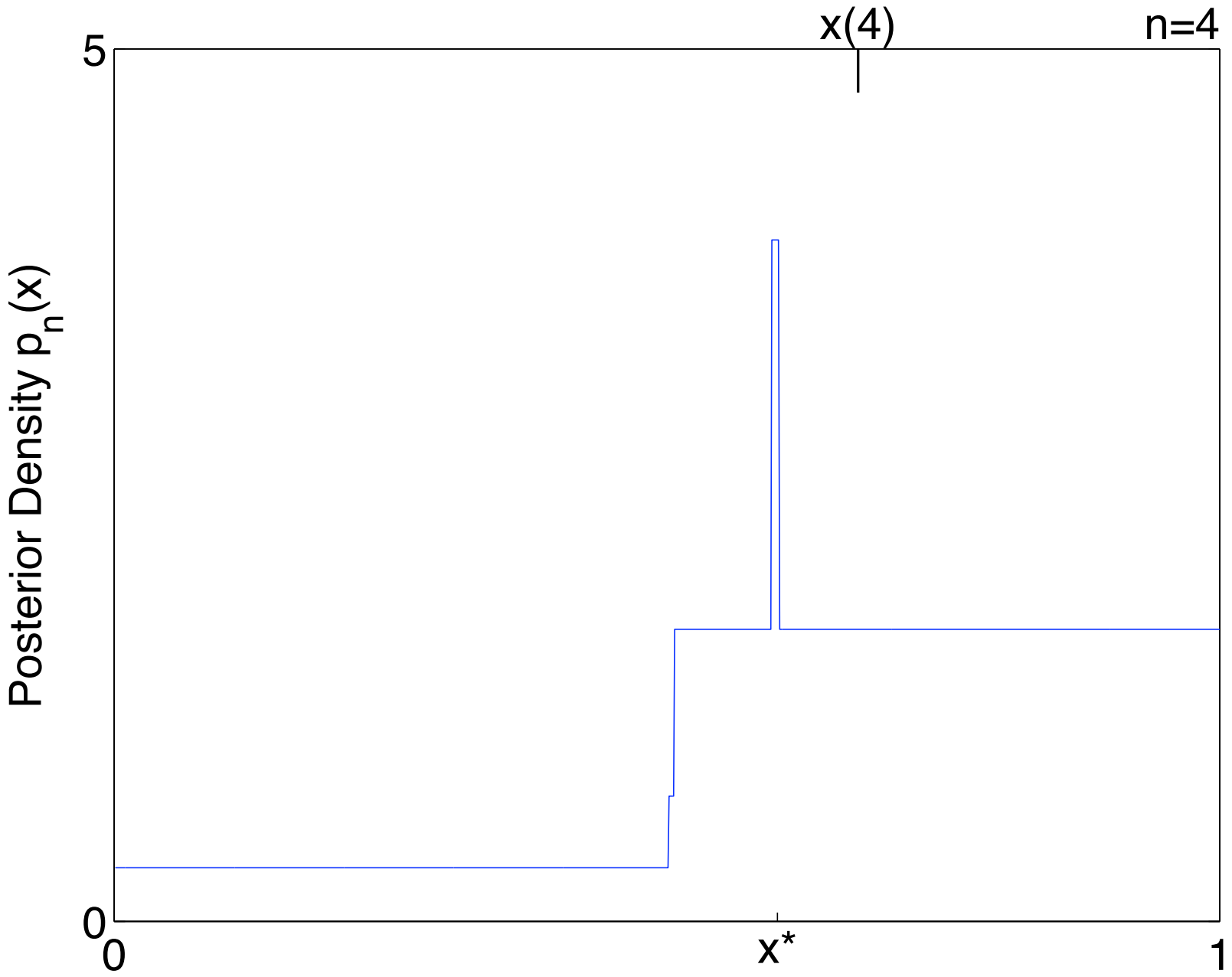
Example



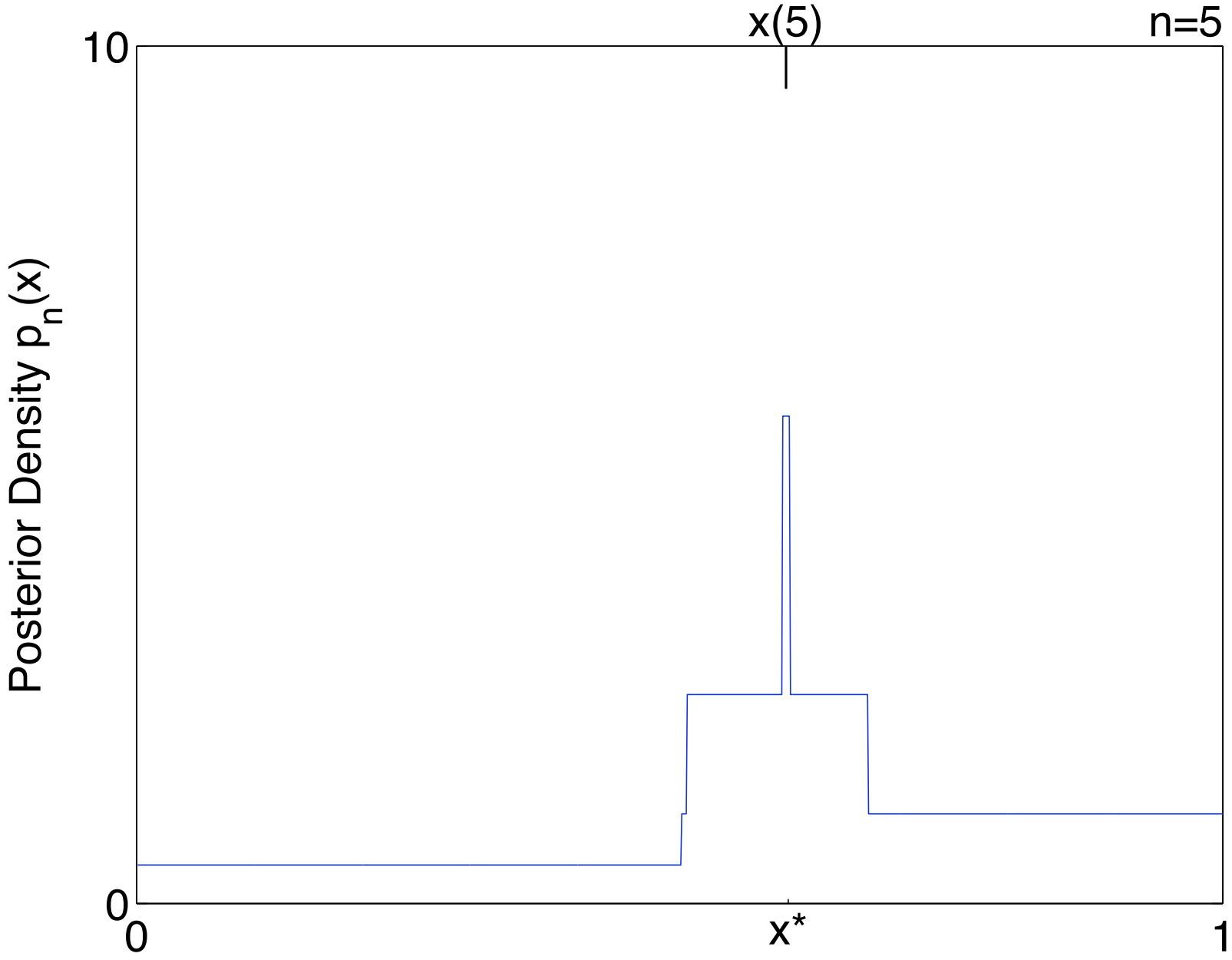
Example



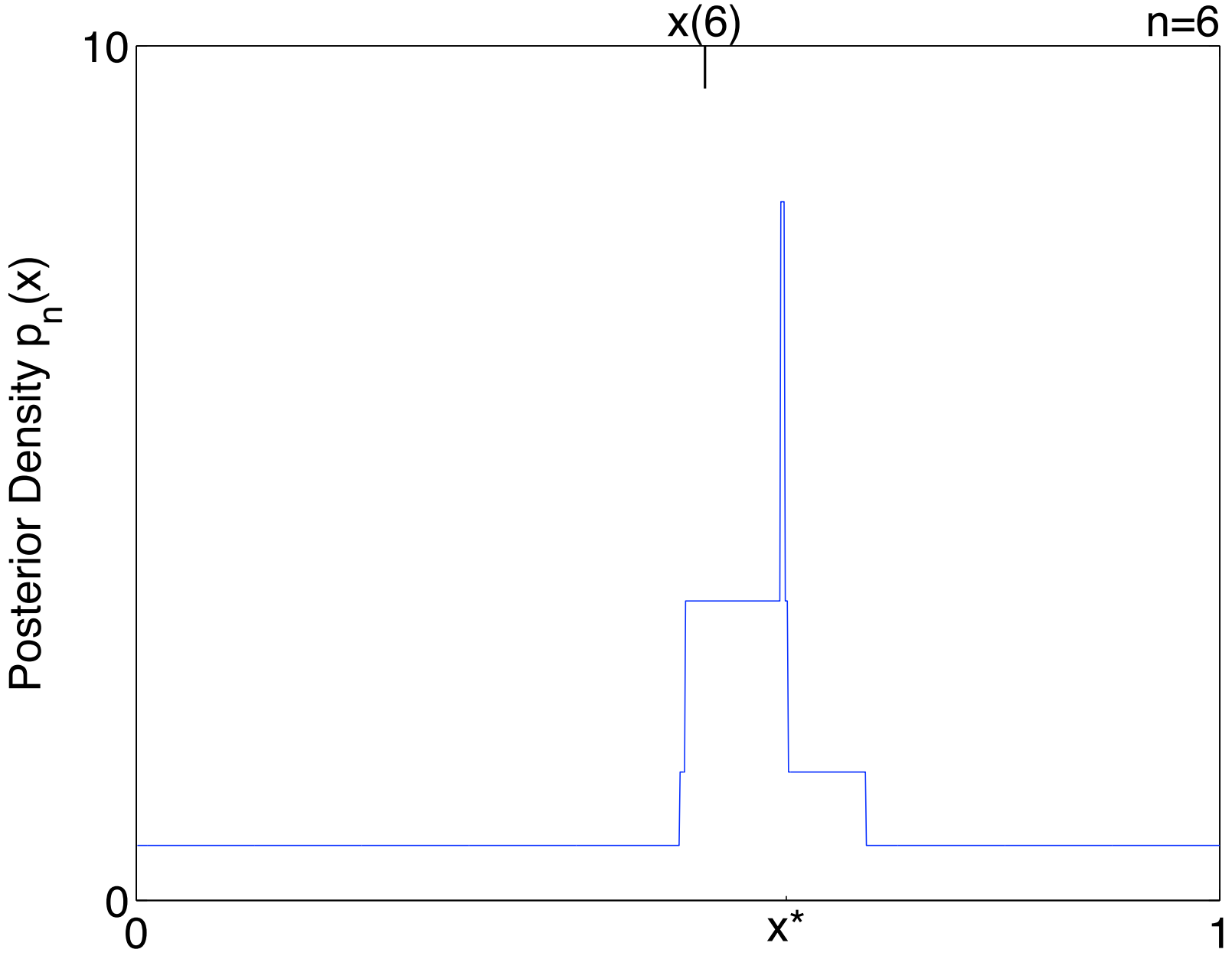
Example



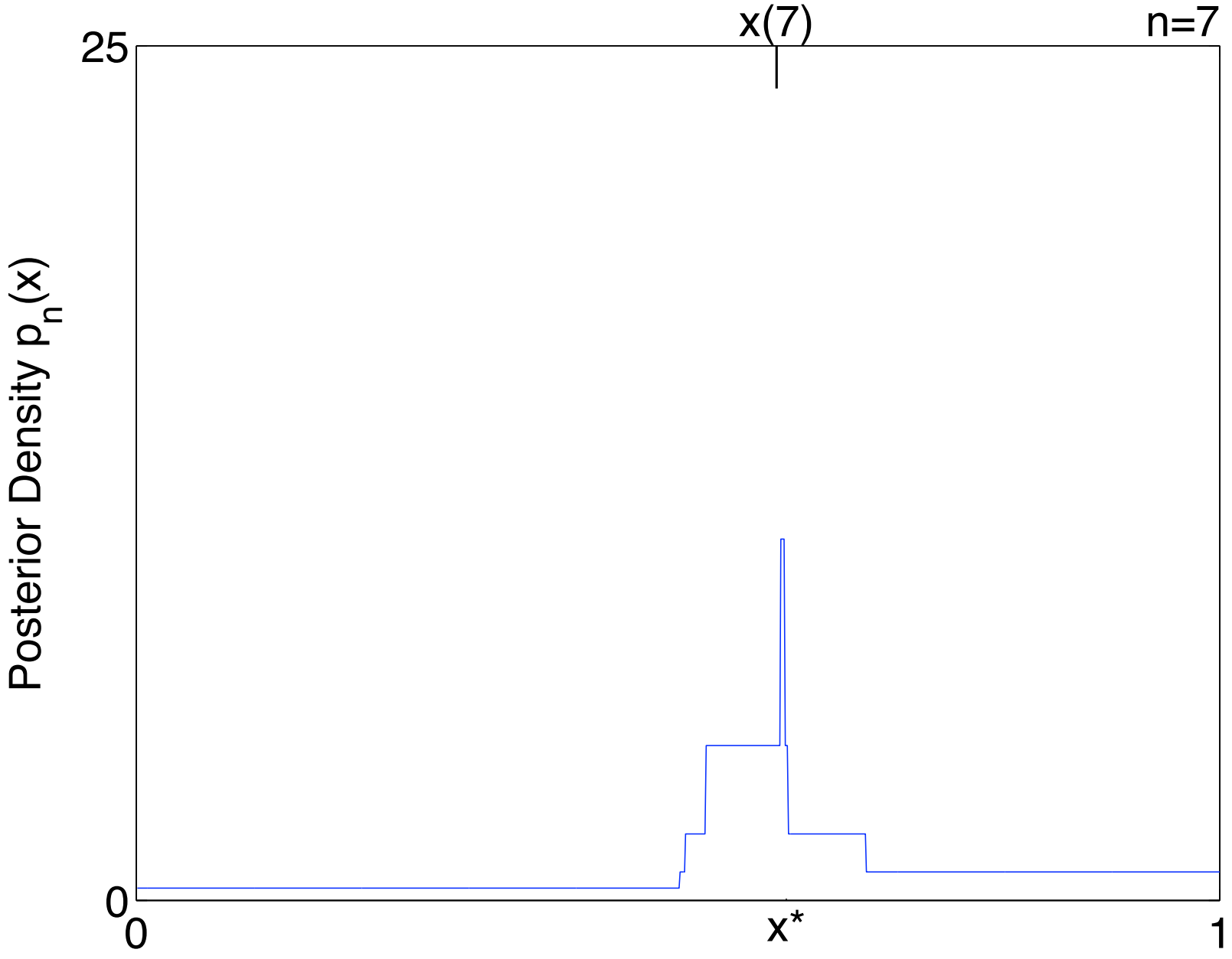
Example



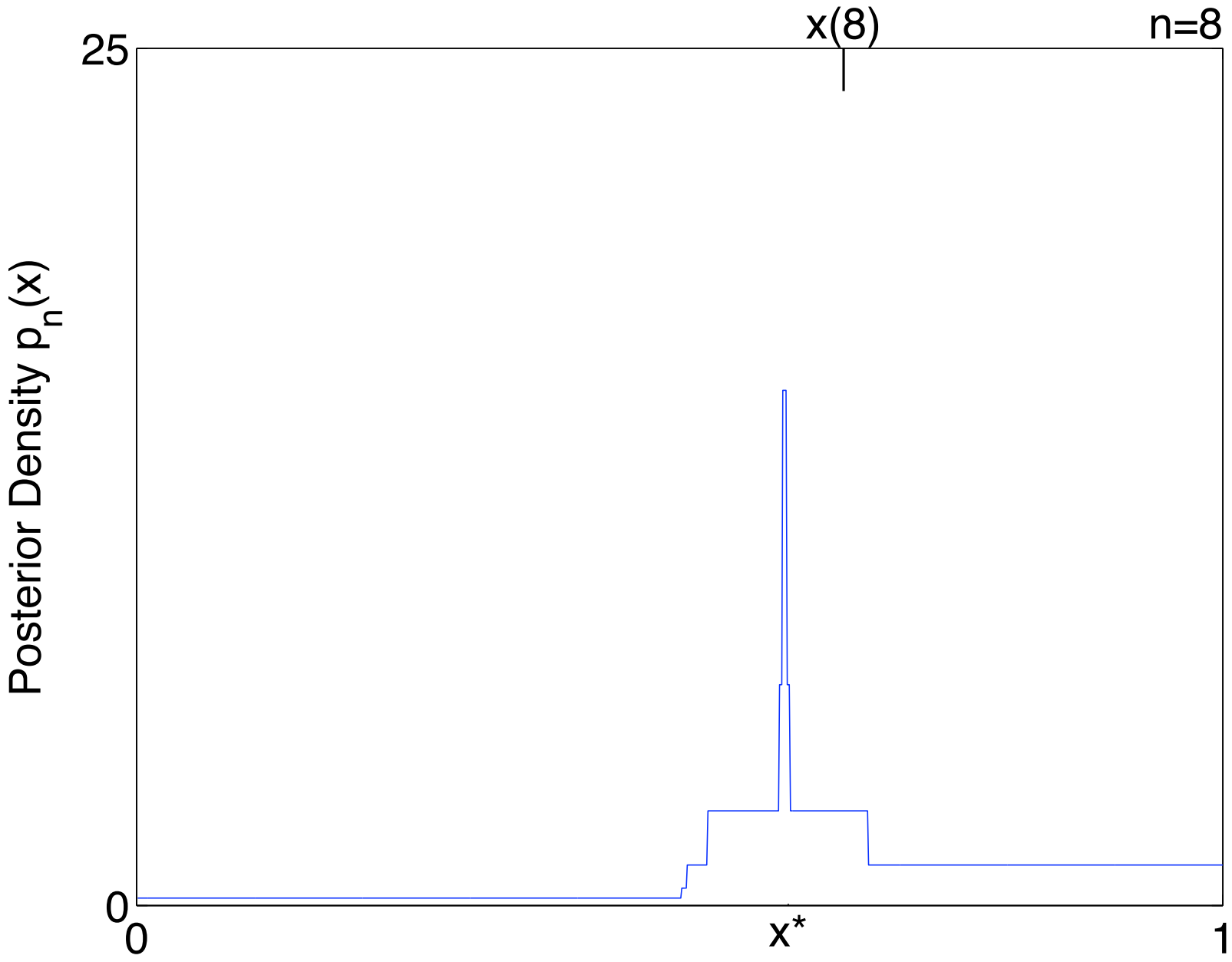
Example



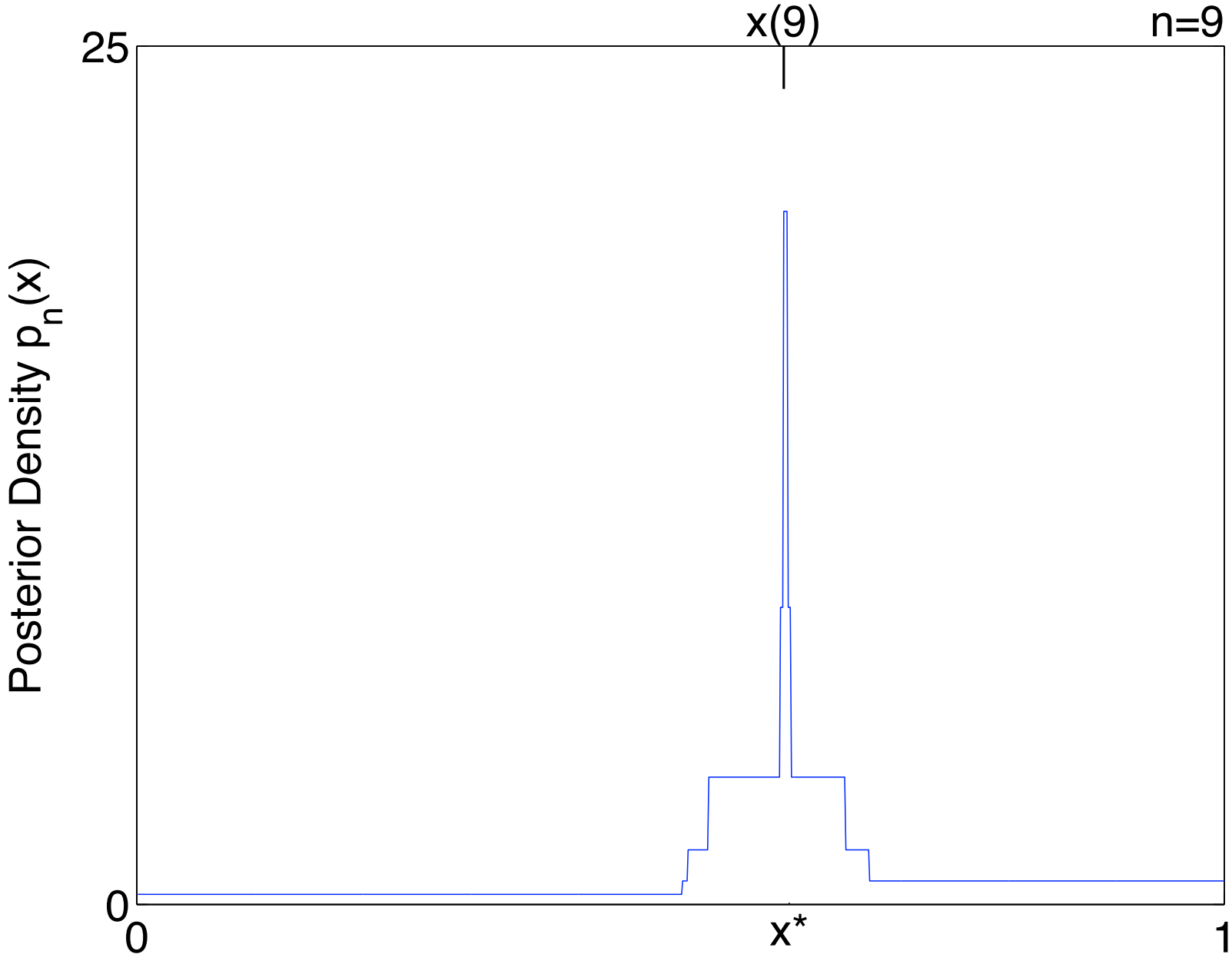
Example



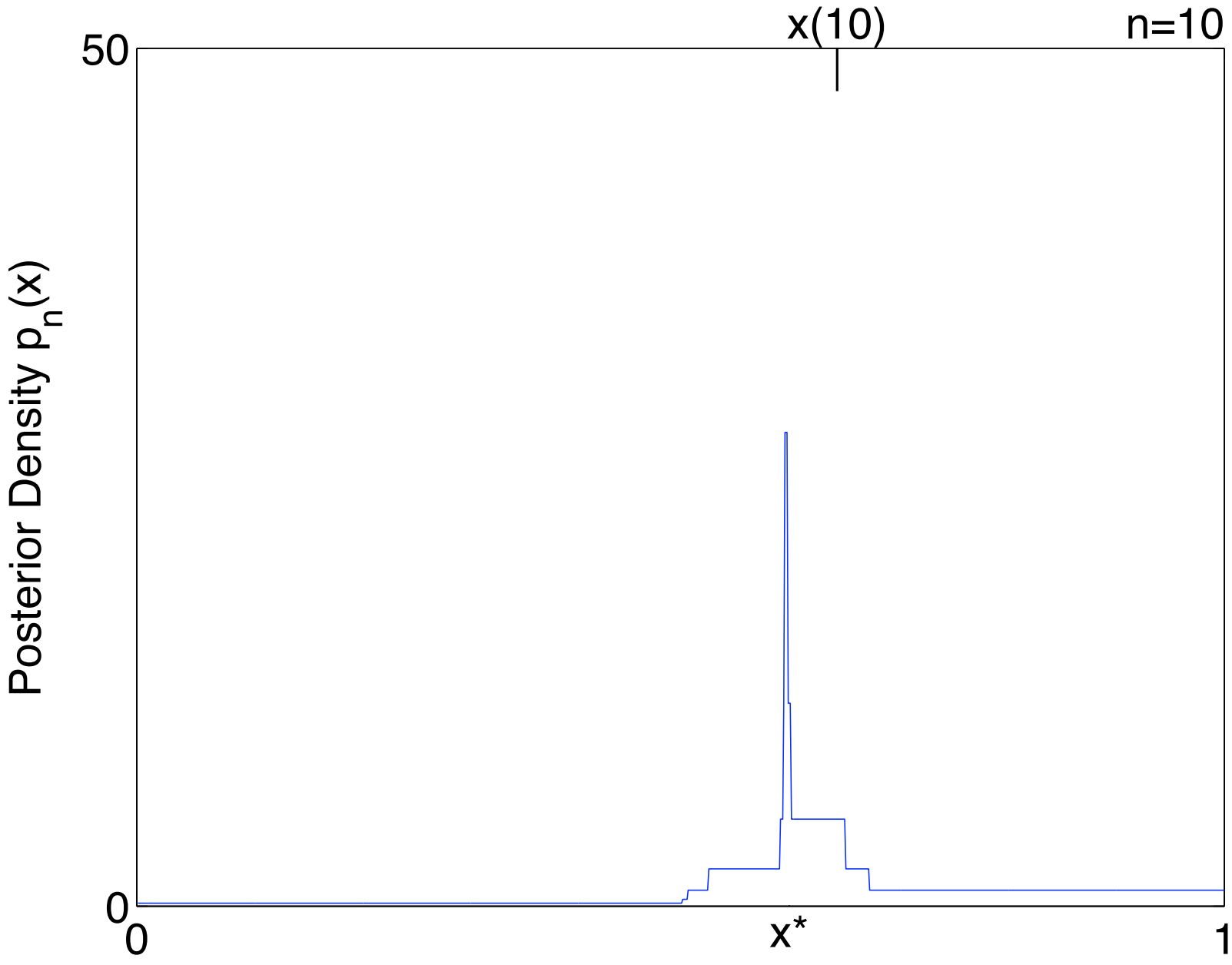
Example



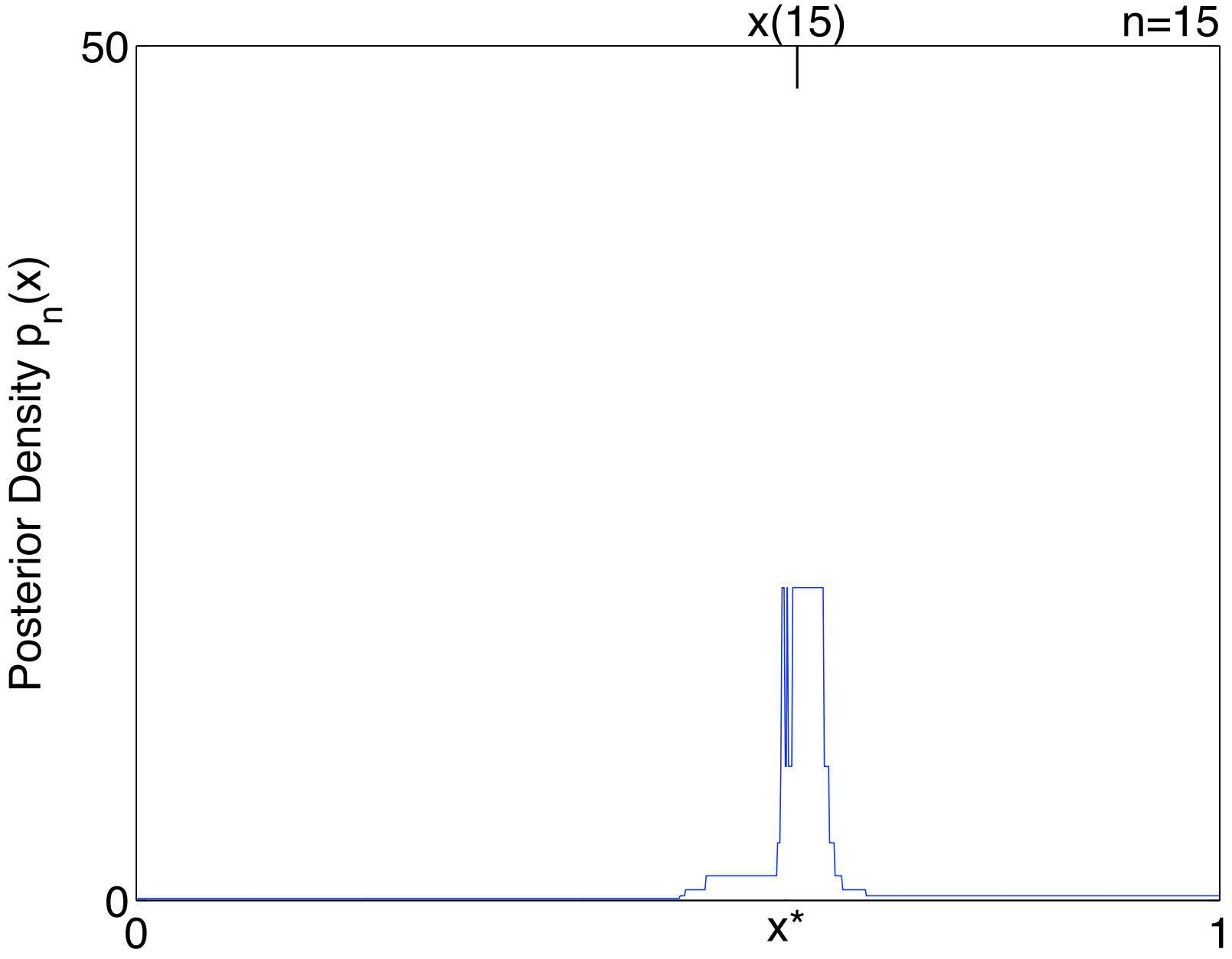
Example



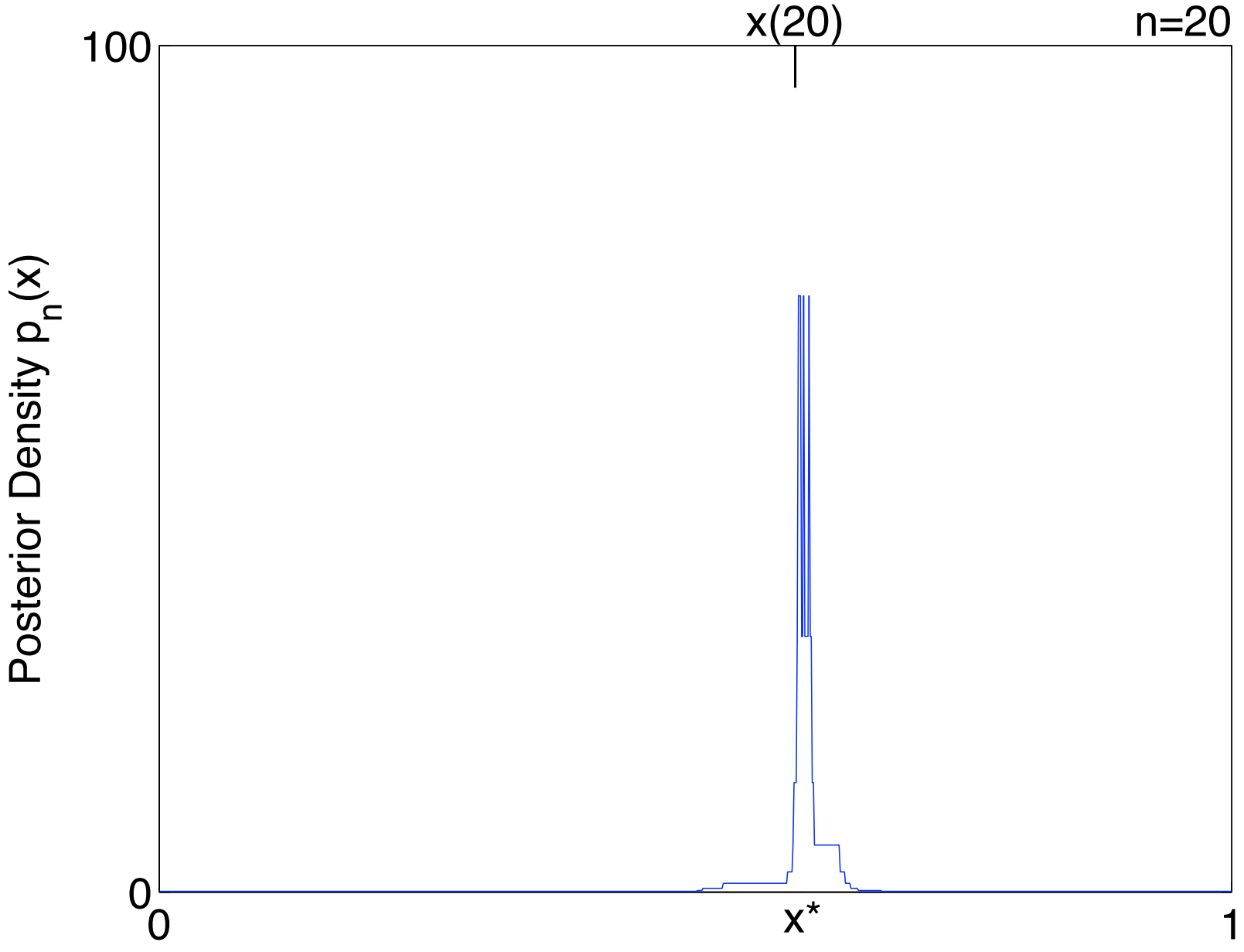
Example



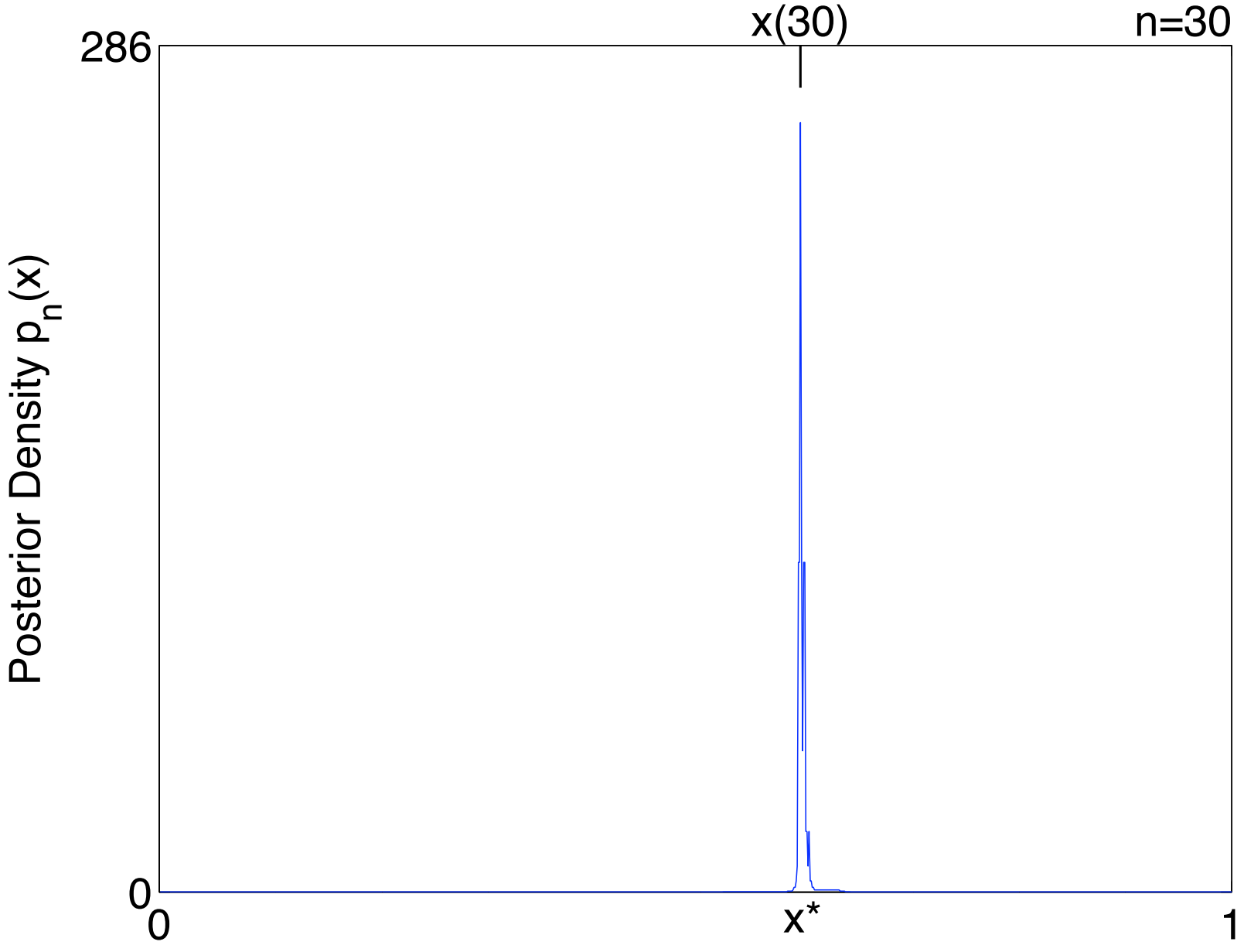
Example



Example



Example



Consistency

- Allow $q(x)$ to vary with x.
- Continue to assume that updates are done with the correct $q(x)$.
- Then we have convergence to the root.

Theorem

$x_n \longrightarrow x_*$ almost surely as $n \longrightarrow \infty$.

[The following results are joint work with Shane Henderson]

Geometric Convergence

- q is fixed.
- Updates are done using q.
- Then x_n converges geometrically to the root x_*.

Theorem

Fix $\varepsilon > 0$ *and let* $A_\varepsilon = [x_* - \varepsilon, x_* + \varepsilon]$.
Let $\tau = \inf\{n : x_n \in A_\varepsilon\}$.
Let $c(q) = 1 + q\log_2(q) + (1-q)\log_2(1-q)$. *Then*

$$\mathbb{E}[\tau] \leq -\log(2\varepsilon)/c(q)$$

Unknown $q(x)$

- In practice, the probability of error $q(x)$ varies with x and is unknown.
- We can estimate $q(x_n)$ by performing multiple measurements at the same x_n, and then use this estimate in our update.
- Alternatively, we can sample sequentially to achieve an error probability $q(x_n)$ **bounded above by a constant**, call it $\widetilde{q}$.
  - Assume ε_n is normal with known variance.
  - We extend standard results from Siegmund 1985 for hypothesis testing of the mean of a normal random variable using curved boundaries.
  - Sampling the sign of $f(x_n)$ can be seen as deciding between three hypotheses: $f(x_n) > 0$; $f(x_n) < 0$; $f(x_n) = 0$.
  - $q(x_n)$ is the probability that the test chooses the wrong hypothesis.
  - Given any $0 < \widetilde{q} < \frac{1}{2}$, a test may be constructed whose $q(x_n) \leq \widetilde{q}$ whenever $P(y_n > 0) \neq \frac{1}{2}$.
  - When $P(y_n > 0)$ is close to $\frac{1}{2}$, the test takes more samples to decide. When $P(y_n > 0) = \frac{1}{2}$, the test may sample forever.

Convergence Rate

- Allow $q(x)$ to vary with x.
- Updates are done with $\widetilde{q}$, which is an upper bound on $q(x_n)$.

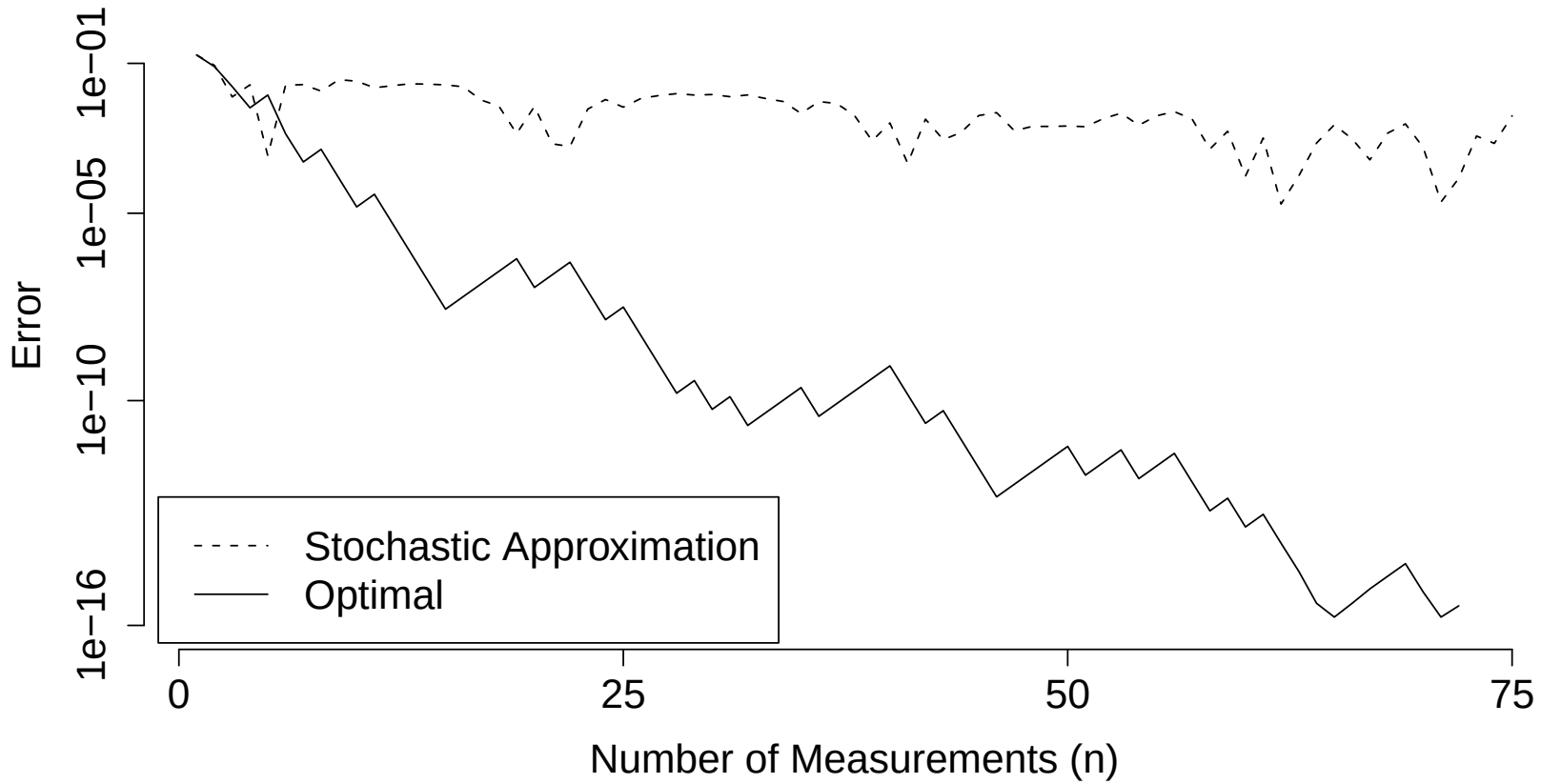
Theorem

Fix $\varepsilon > 0$ and let $A_\varepsilon = [x_* - \varepsilon, x_* + \varepsilon]$.
Let $\tau = \inf\{n : x_n \in A_\varepsilon\}$.
Let $c(\widetilde{q}) = 1 + \widetilde{q}\log_2(\widetilde{q}) + (1-\widetilde{q})\log_2(1-\widetilde{q})$. Then

$$\mathbb{E}[\tau] \leq -\log(2\varepsilon)/c(\widetilde{q})$$

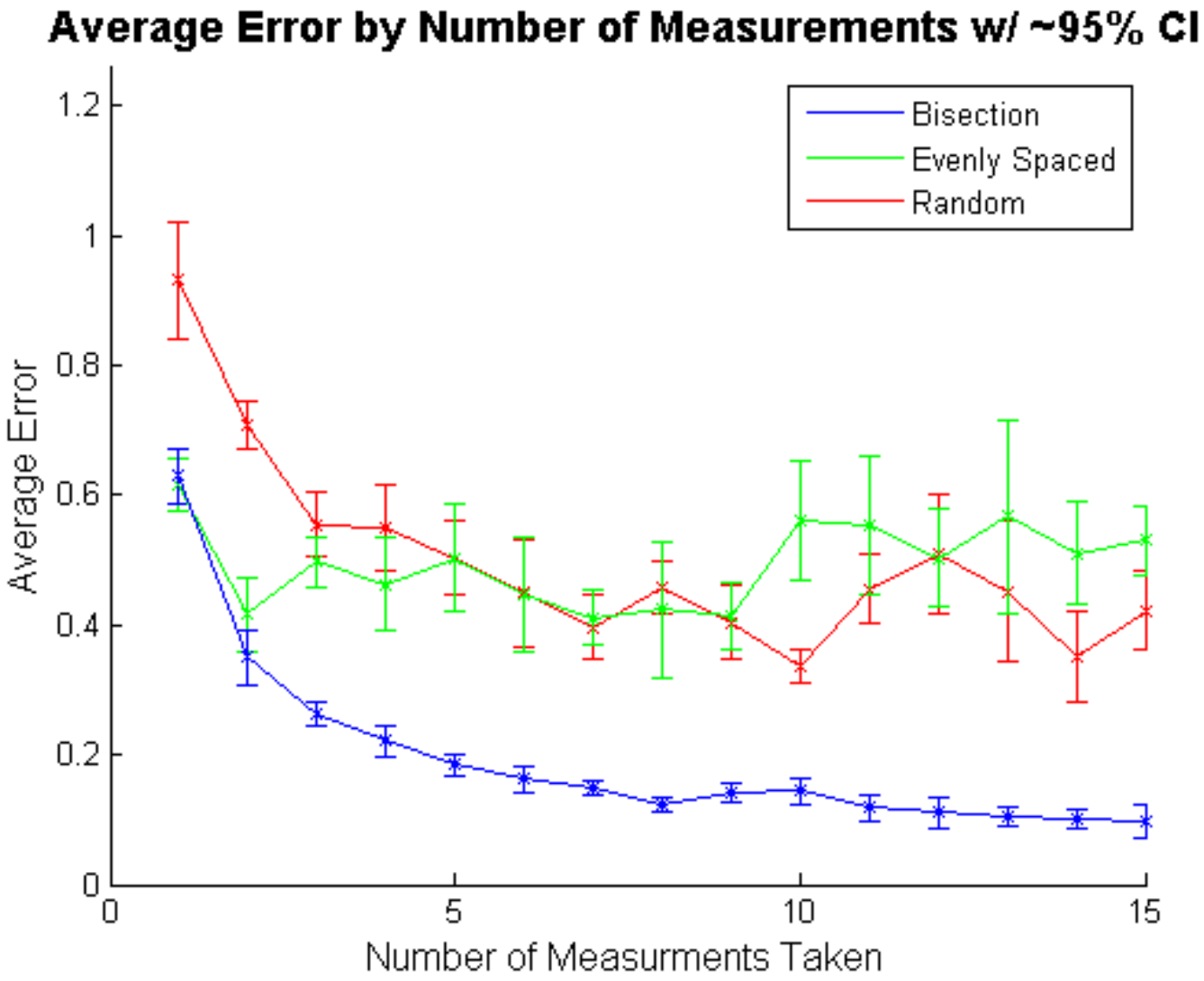
We still have geometric convergence in the number of different x_n sampled, but it may take more time to sample one x_n when x_n is closer to x_*.

Experimental Results (Simulation Problem)



Performance on a problem with $f(x) = \exp(x) - \exp(1/3)$ and domain $[0,1]$. Error is $|x_n - x_*|$ on one sample path. Stochastic approximation used stepsize $1/n$.

Experimental Results (Laboratory Problem)

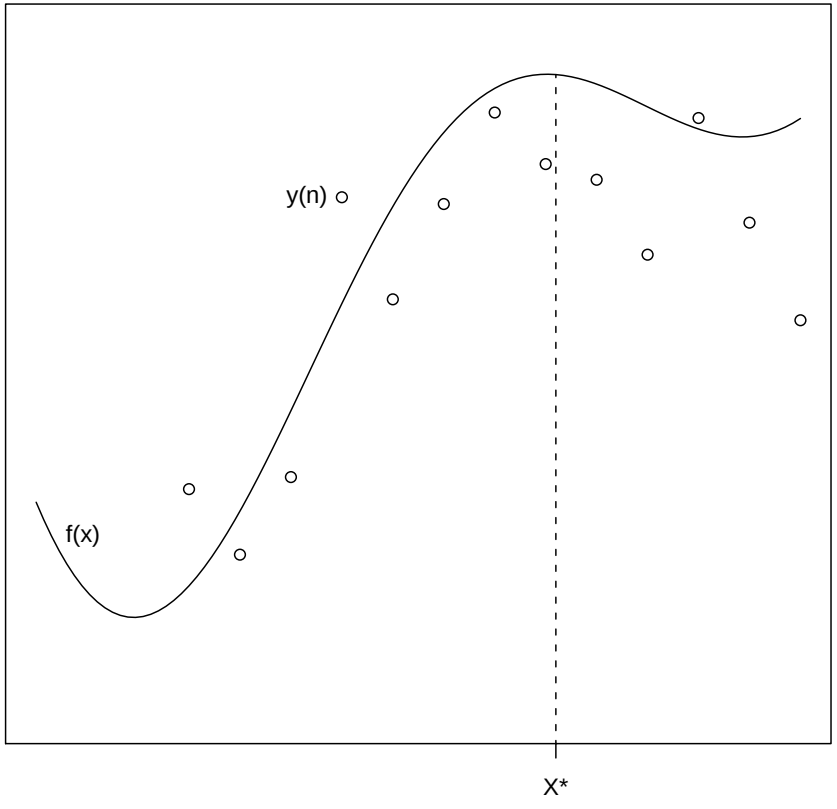


Joint work with Zach Owen, Thorsten Joachims, and Rodrigo Bicalho.

Outline

Global Optimization with Noise

We now consider a different problem:
Given a fixed budget of function evaluations, locate the global maximum of a continuous function.



[Joint work with Warren Powell & Savas Dayanik]

Global Optimization with Noise

- $f : [0,1]^d \mapsto \mathbb{R}$, not necessarily monotone or concave.
- $y_n = f(x_n) + \varepsilon_n$ where $\varepsilon_n \sim \mathcal{N}(0, \sigma^2(x_n))$ is independent noise.
- Our goal is to choose $\widehat{x_*}$ to make $f(\widehat{x_*})$ as large as possible.
- Central Question: Given a budget of N measurements, $x_1, \ldots, x_N$, how should we place them to choose x_* as well as possible?

Global Optimization with Noise

- For any fixed function f, and any fixed policy π, the expected performance is

$$\mathbb{E}^{\pi}\left[f(\widehat{x}_*) \mid f\right].$$

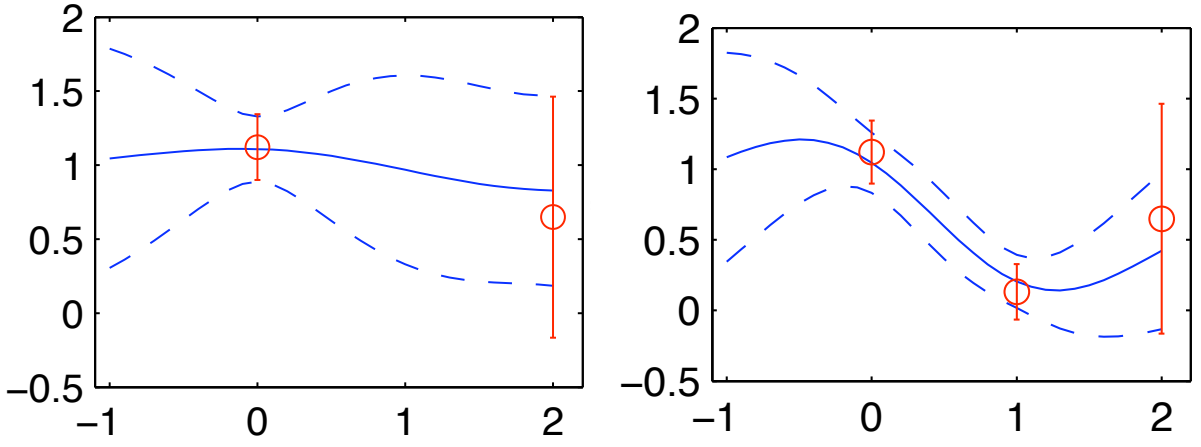
- Our goal is to solve

$$\sup_{\pi} \mathbb{E}\left[f(\widehat{x}_*)\right],$$

where the expectation over f is taken with respect to the prior $\mathbb{P}_0$.

Gaussian Process Prior

- Our prior $\mathbb{P}_0$ on f is a **Gaussian Process Prior (GPP)**.
- GPPs are commonly used in Bayesian spatial statistics to model continuous functions.
- GPPs are analytically convenient:
  they can be updated in closed-form.



Dynamic Programming

- The optimal policy is described by the dynamic programming equations.

$$V_N(\mathbb{P}_N) = \sup_{x \in [0,1]^d} \mathbb{E}_N [f(x)],$$
$$V_n(\mathbb{P}_n) = \sup_{x \in [0,1]^d} \mathbb{E}_n [V_{n+1}(\mathbb{P}_{n+1})].$$

- These DP equations are **extremely difficult to solve numerically** because the state space is so large.
  - The state space is the space of all possible posterior probability distributions on f.
  - It has $2N$ continuous dimensions.

Knowledge-Gradient Policy

- We construct a one-step approximation to the optimal policy.
- Recall $V_N(P)$ is the expected value of the best $\widehat{x}_*$ we can choose based on a posterior P.
- For each x, we define the **Knowledge-Gradient (KG) Factor**

$$KG_n(x) = \mathbb{E}_n \left[V_N(\mathbb{P}_{n+1}) - V_N(P_n) \mid x_n = x \right].$$

- $KG_n(x)$ is the expected difference in value between our knowledge $\mathbb{P}_n$ at time n, and our knowledge $\mathbb{P}_{n+1}$ at time $n+1$.
- $KG_n(x)$ is the (myopic) value of what we learn from sampling x.

Knowledge-Gradient Policy

- Recall the KG factor is defined as

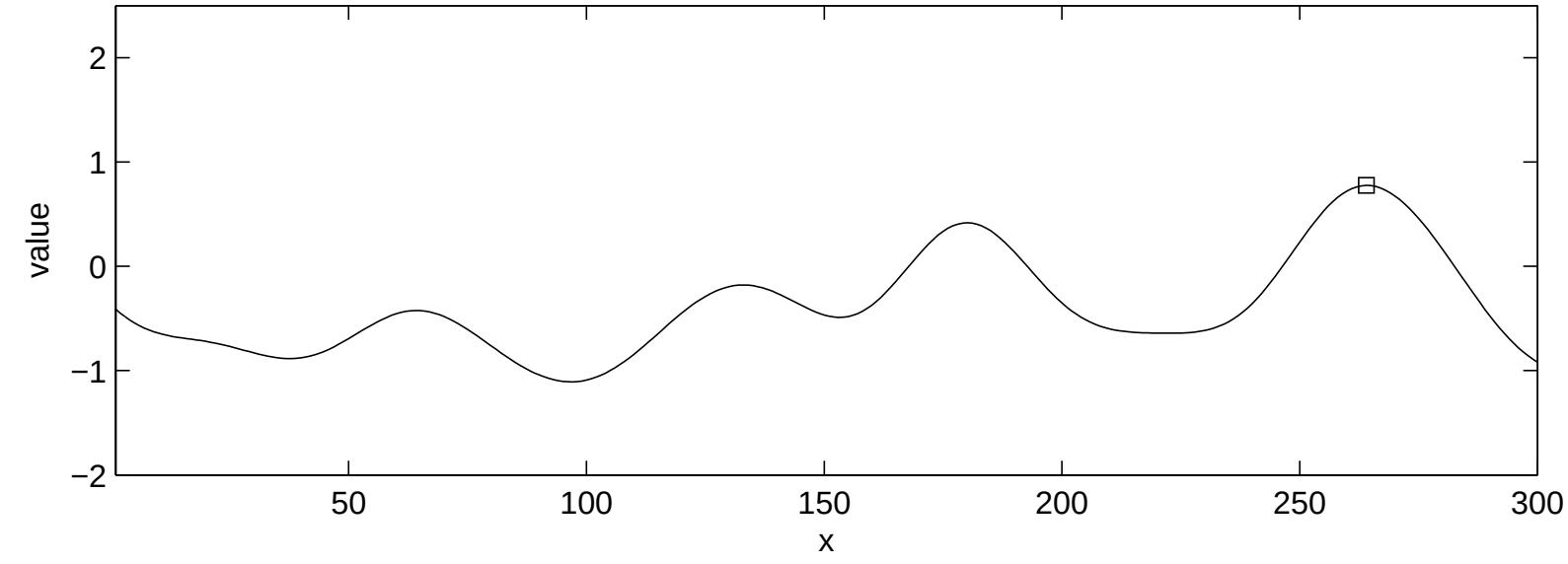
$$KG_n(x) = \mathbb{E}_n \left[V_N(\mathbb{P}_{n+1}) - V_N(P_n) \mid x_n = x \right].$$

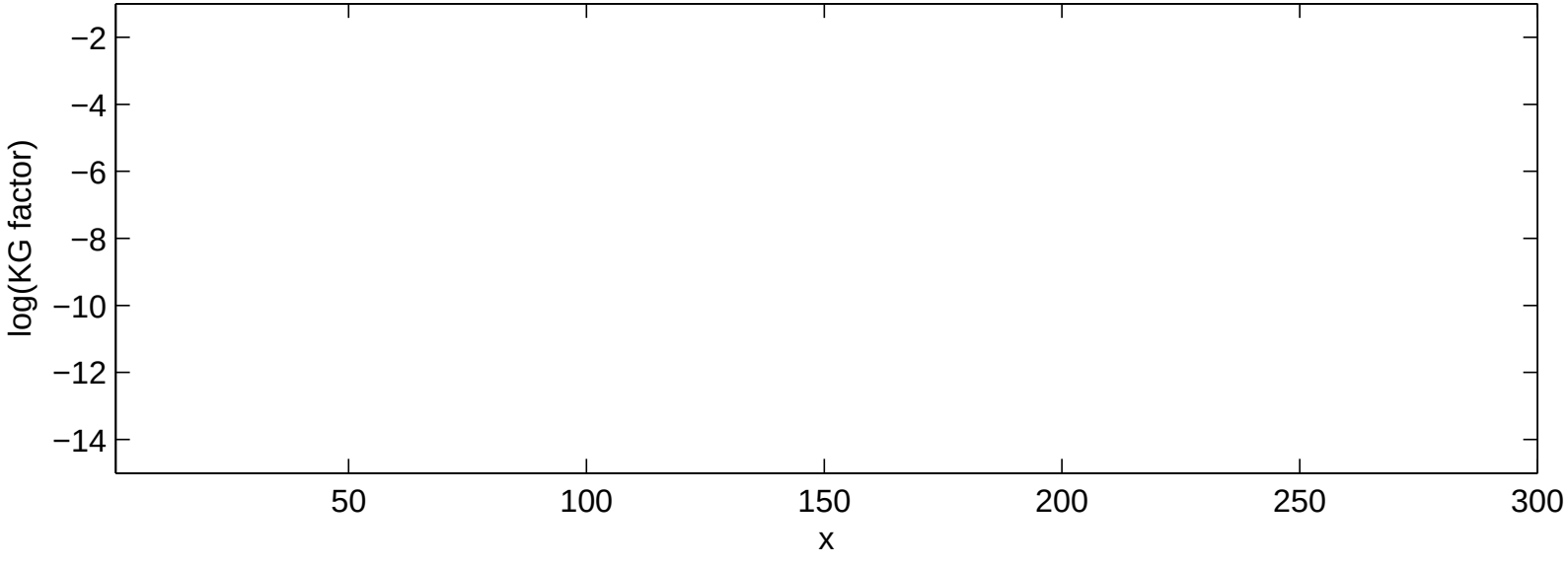
- The **knowledge-gradient (KG) policy** is

$$x_n \in \arg\max_x KG_n(x).$$

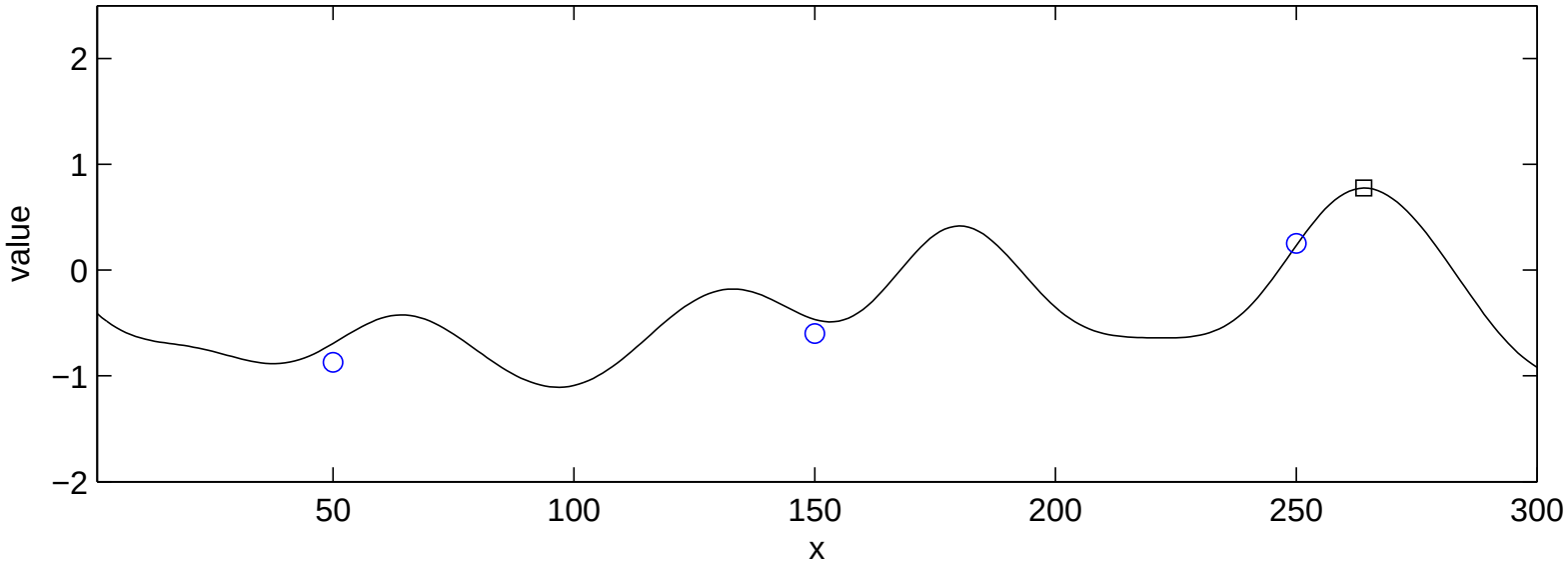
- The KG policy is optimal when $n = N - 1$, and so is a one-step lookahead policy.

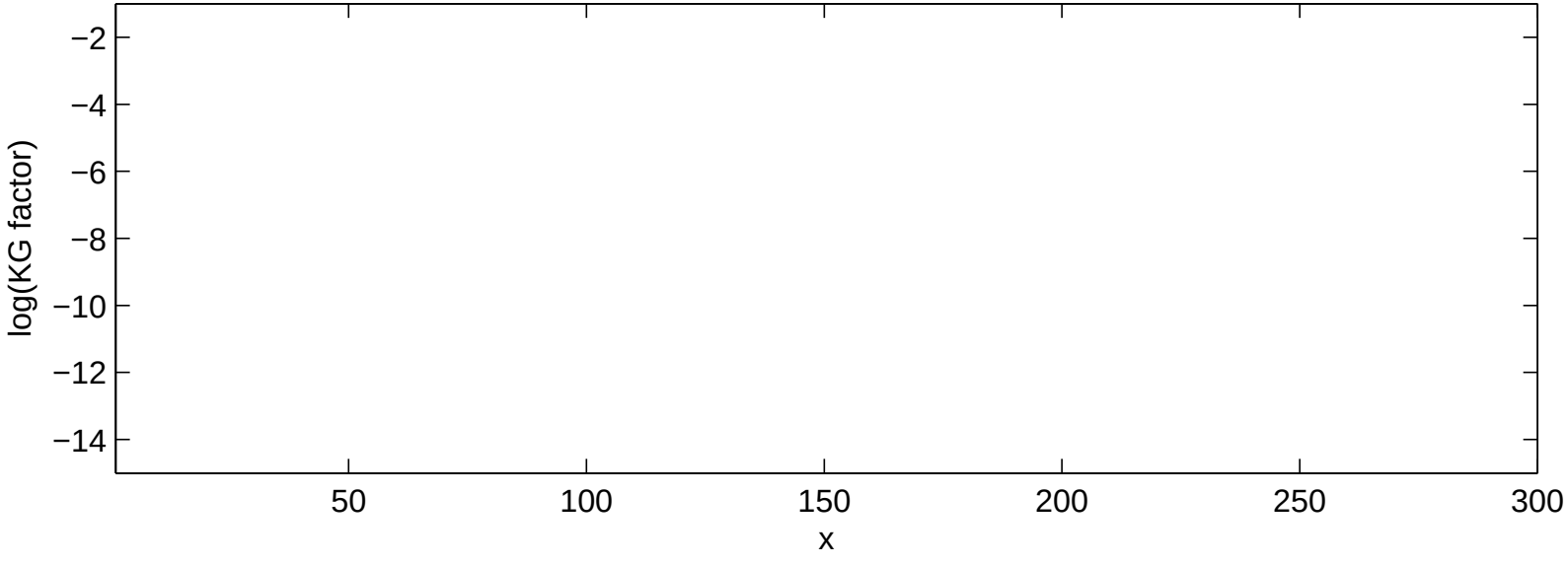
Global Optimization Example



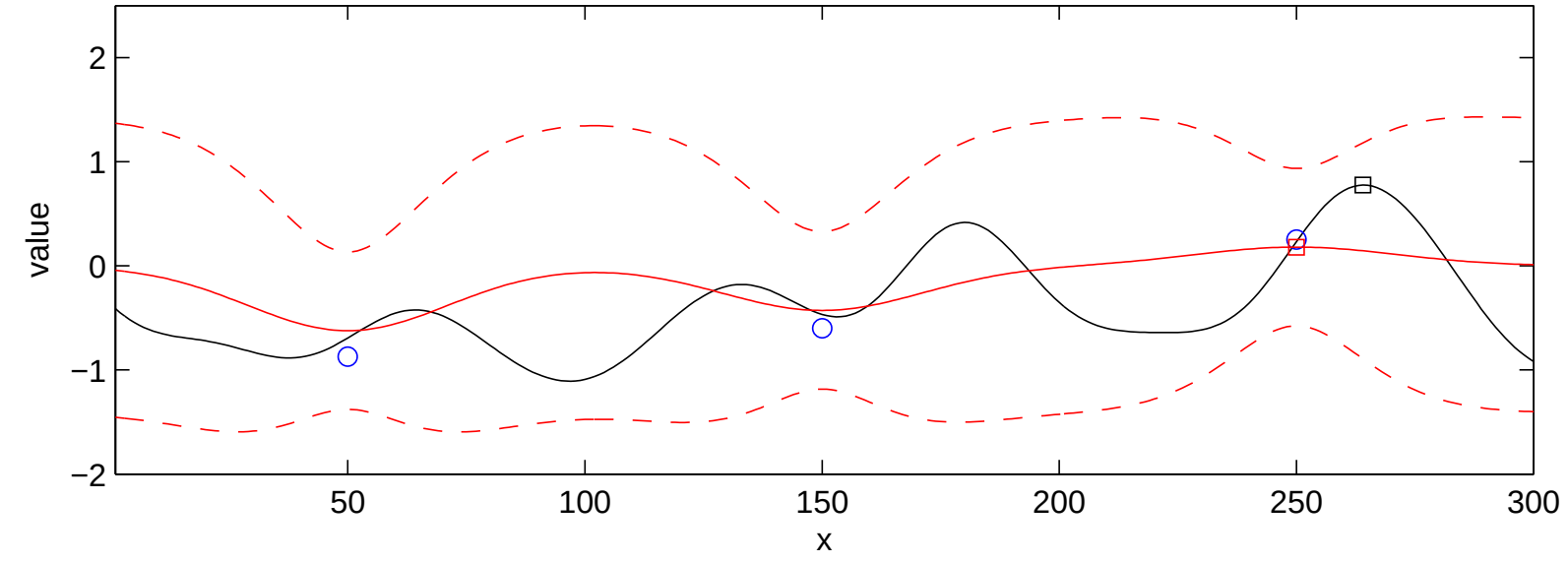


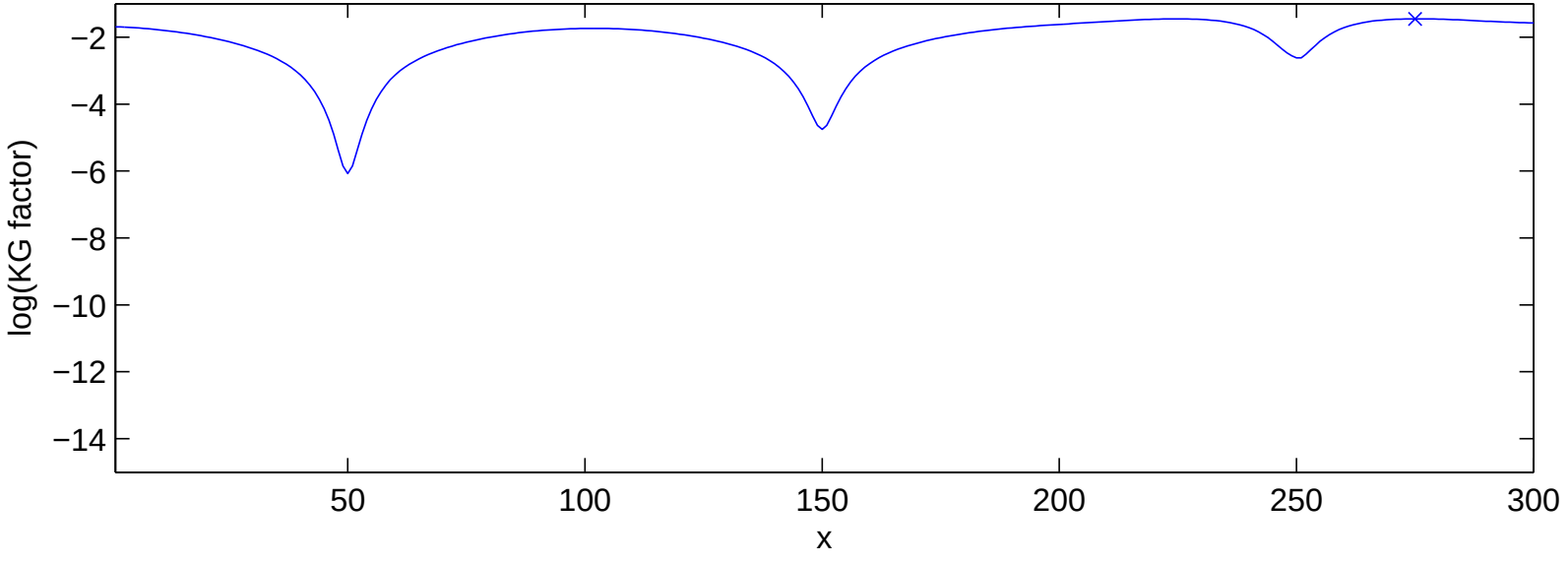
Global Optimization Example



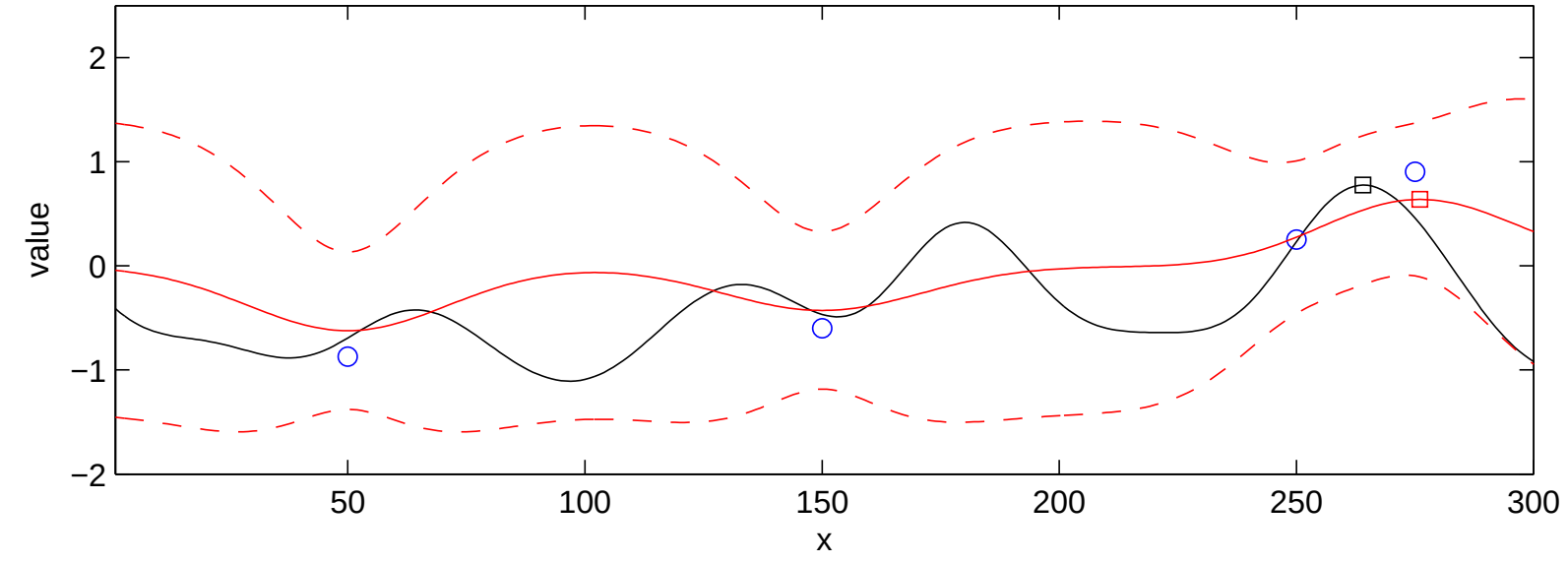


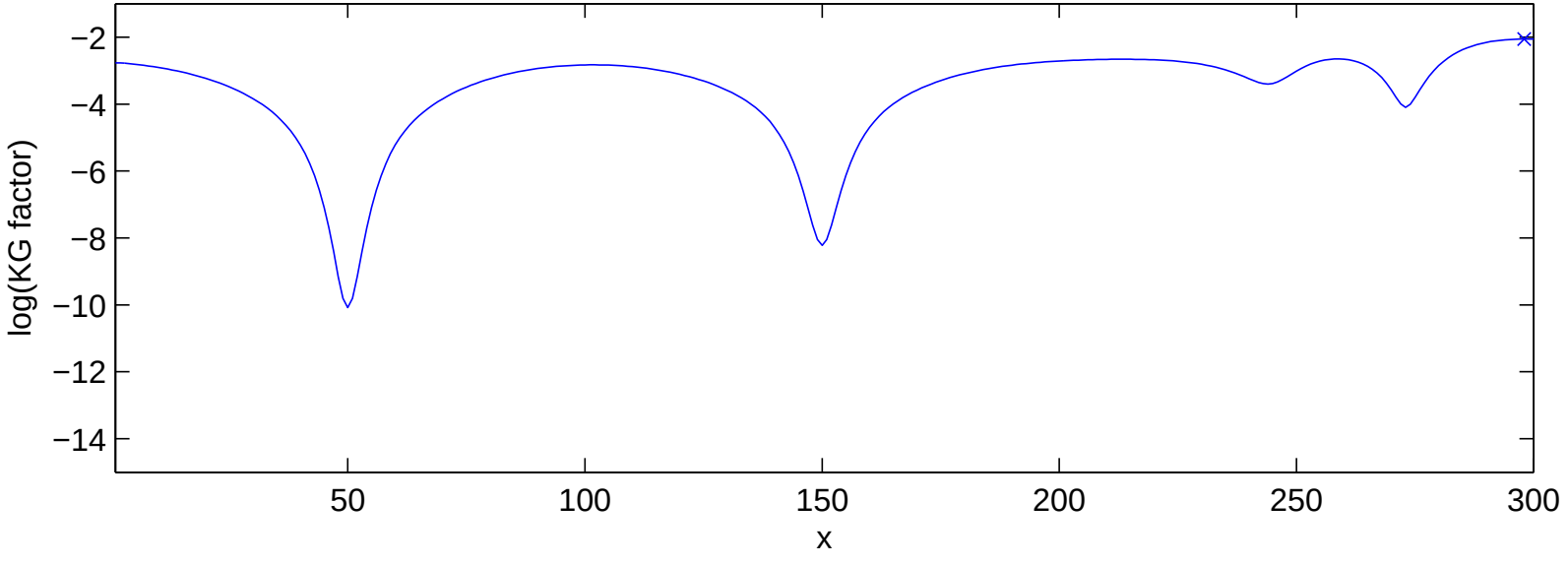
Global Optimization Example



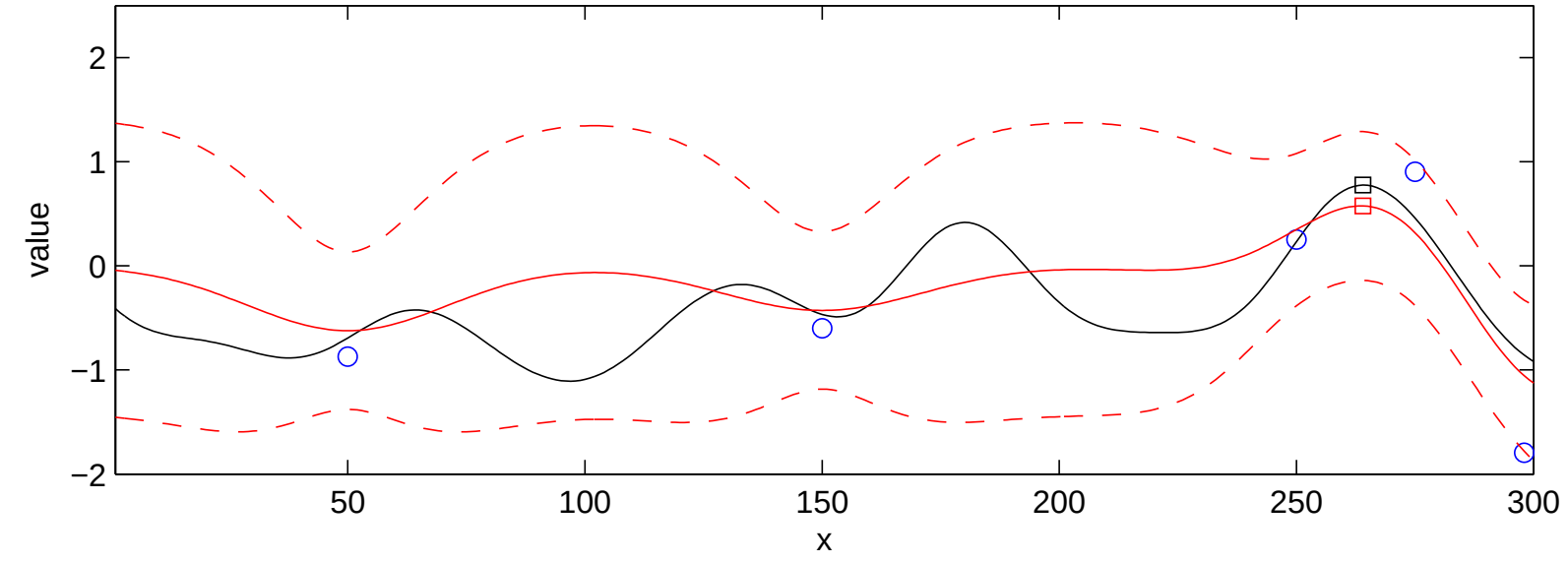


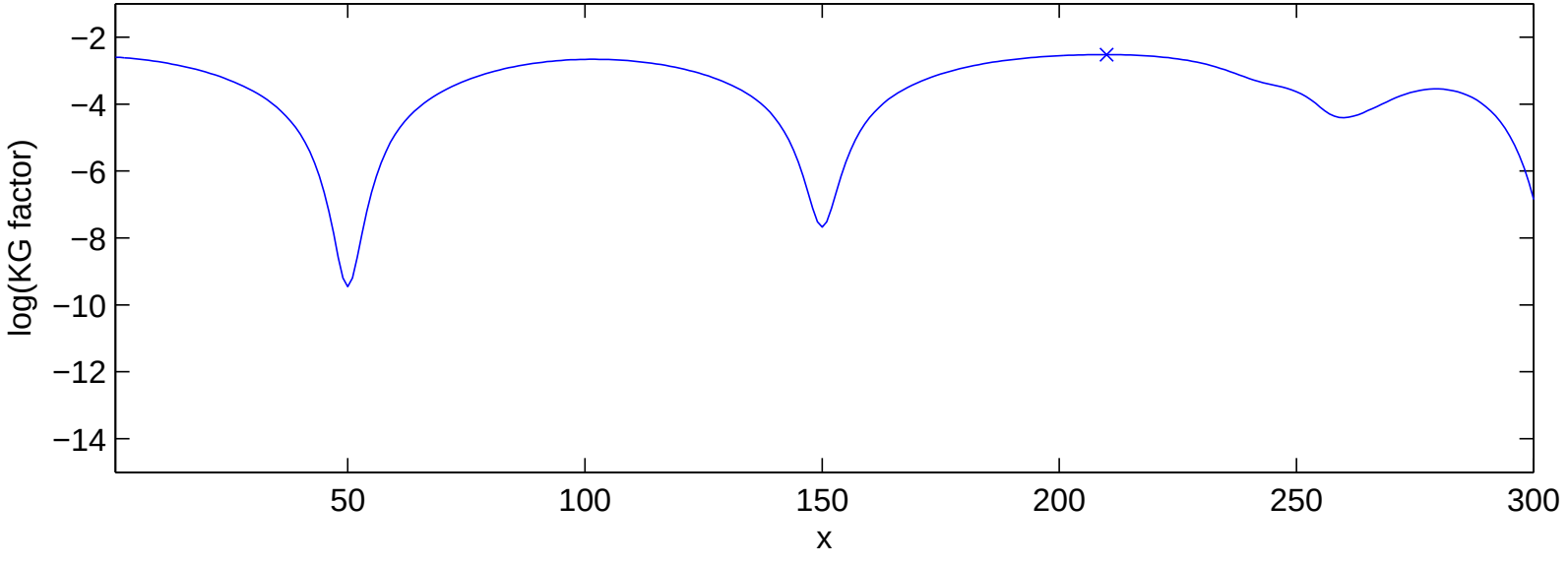
Global Optimization Example



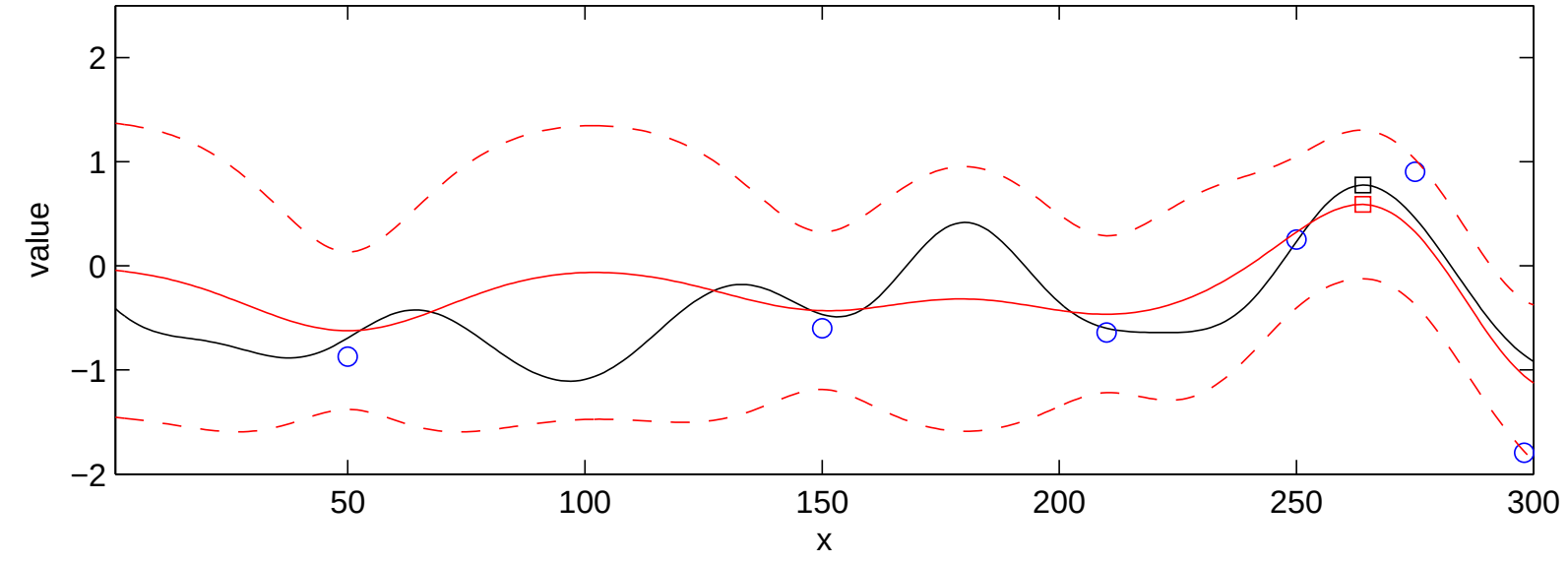


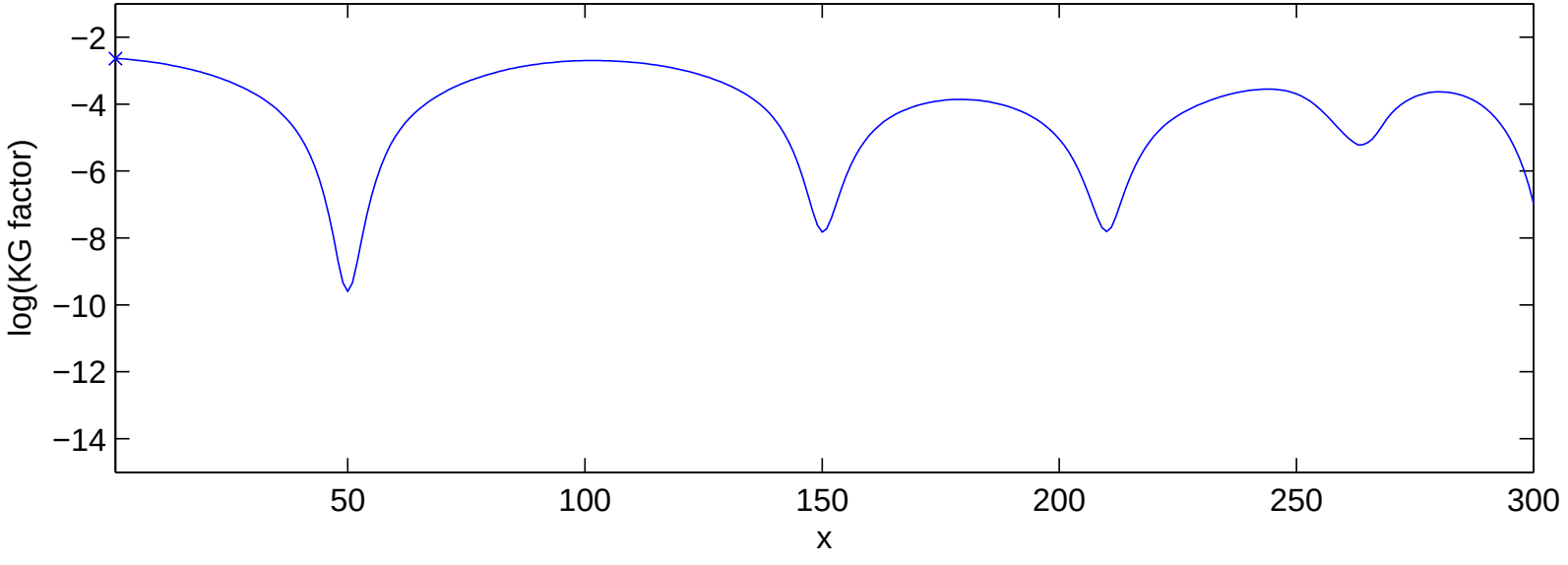
Global Optimization Example



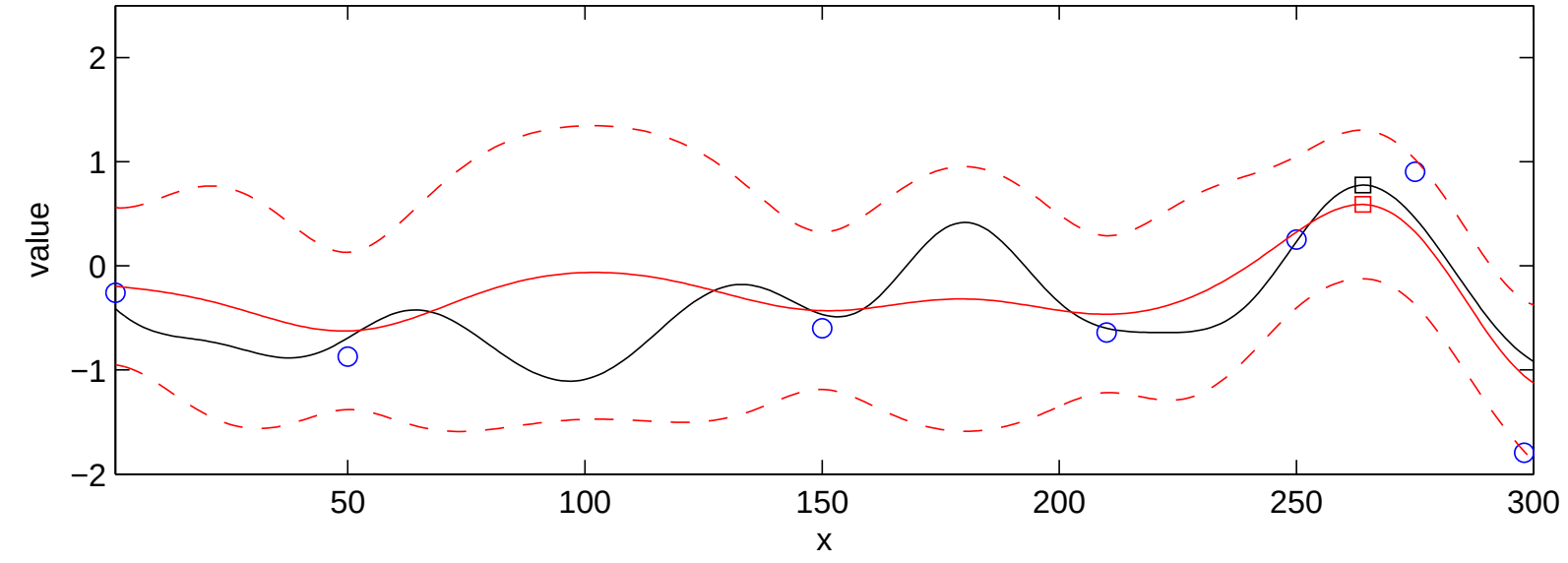


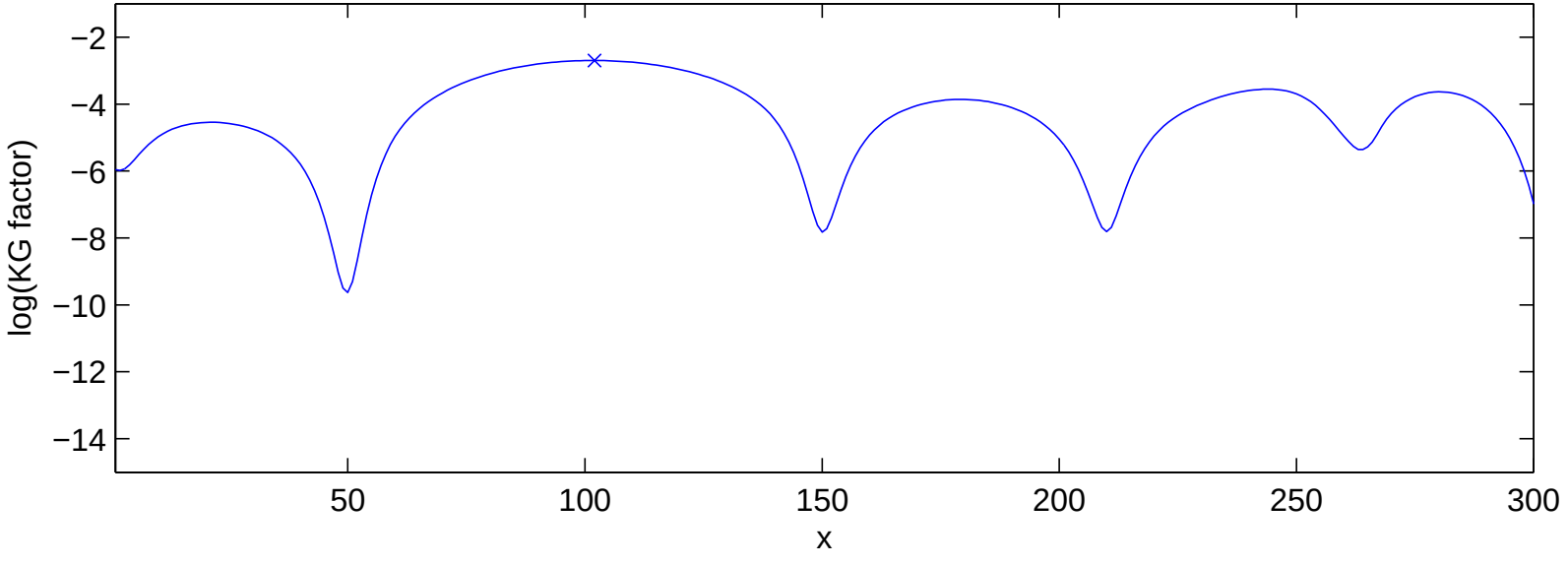
Global Optimization Example



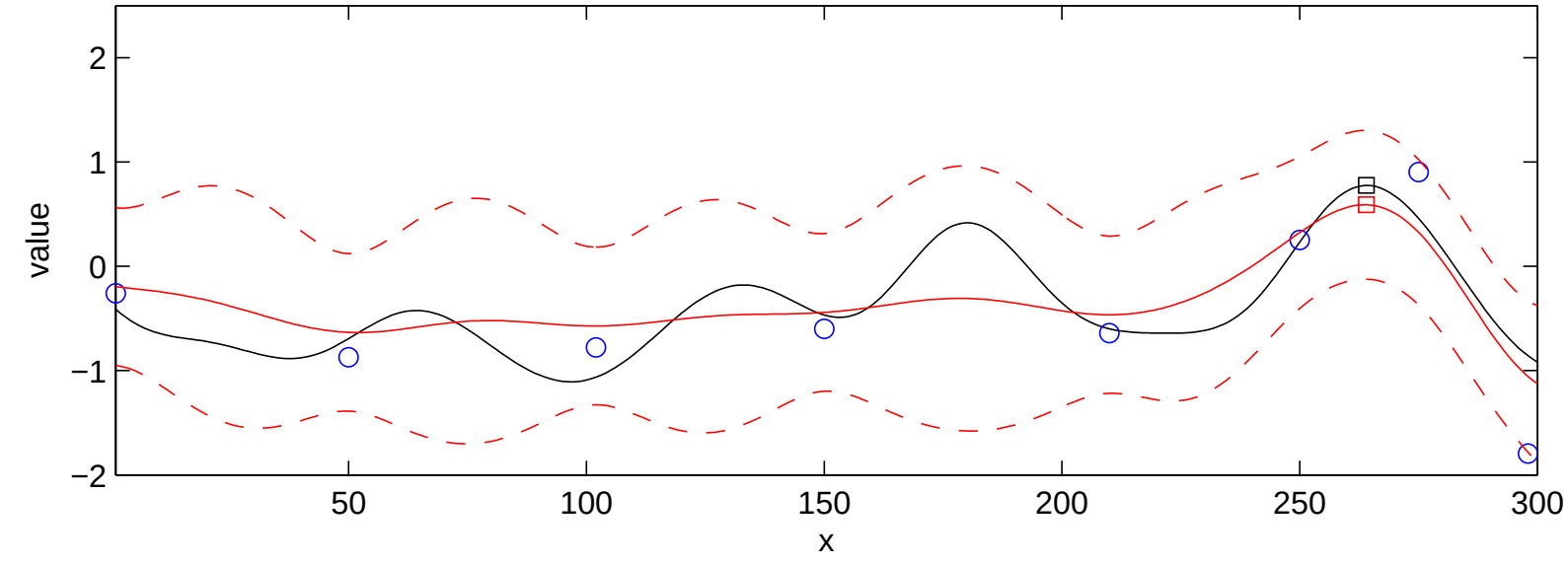


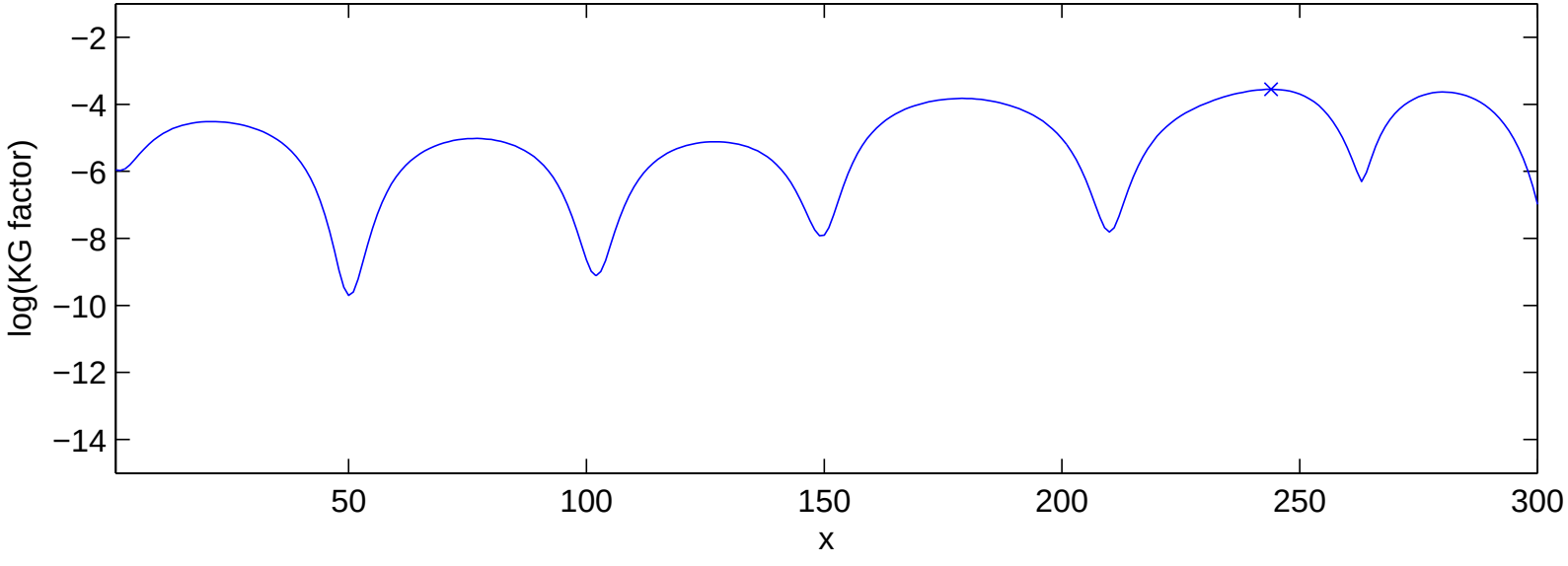
Global Optimization Example



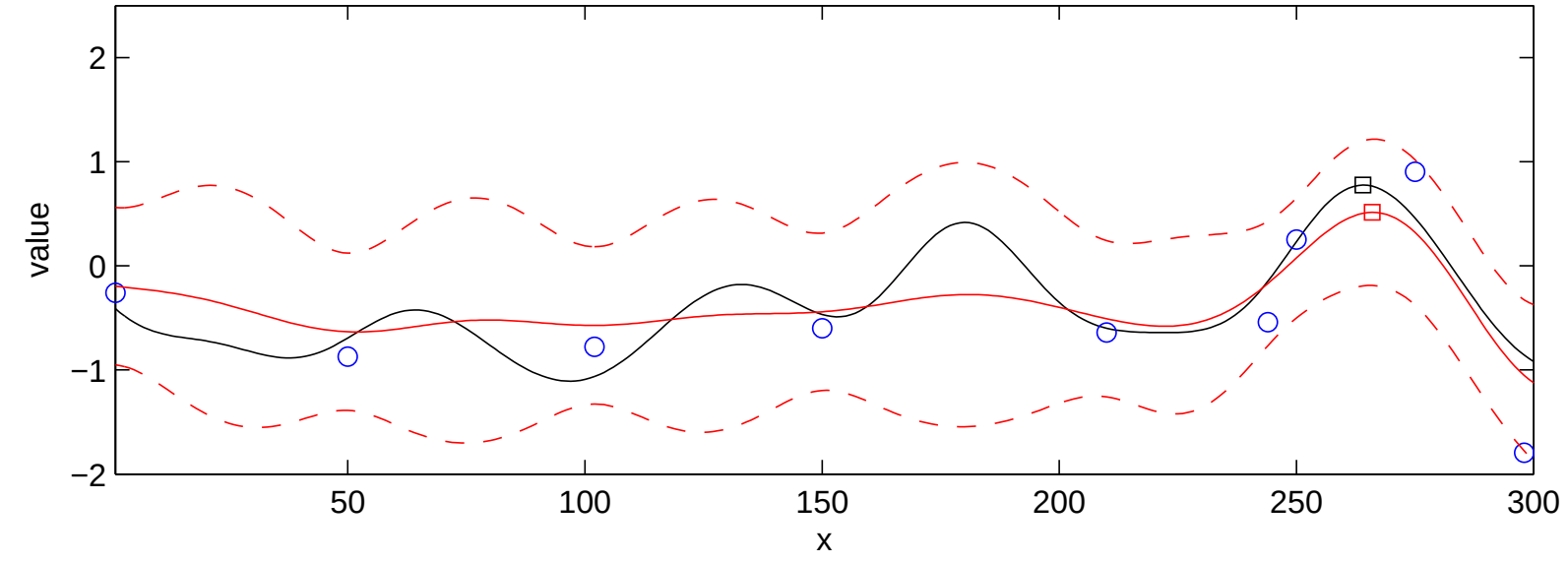


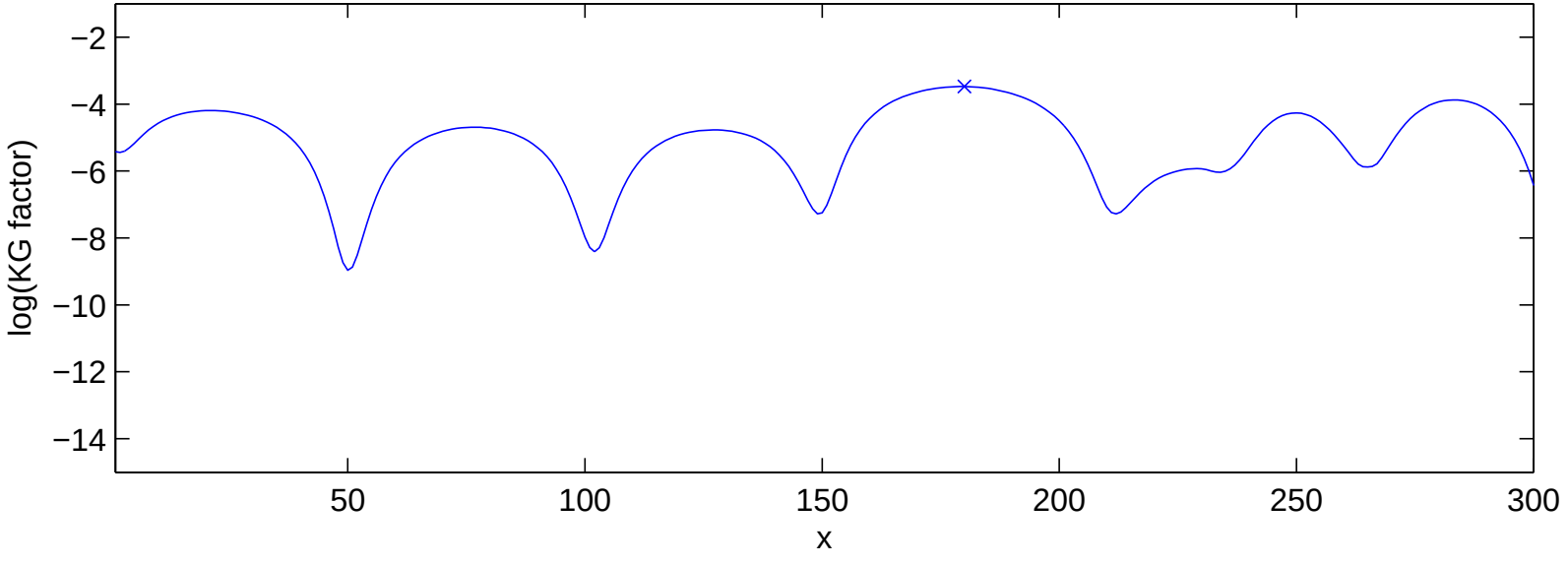
Global Optimization Example



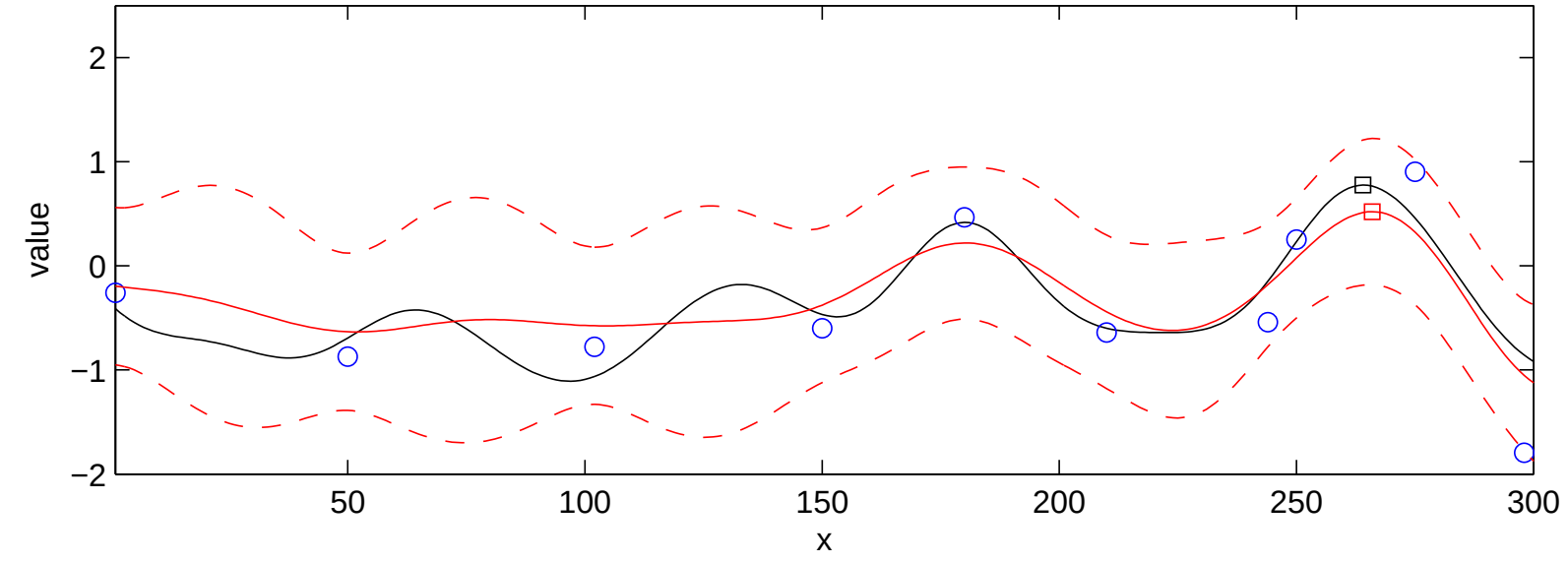


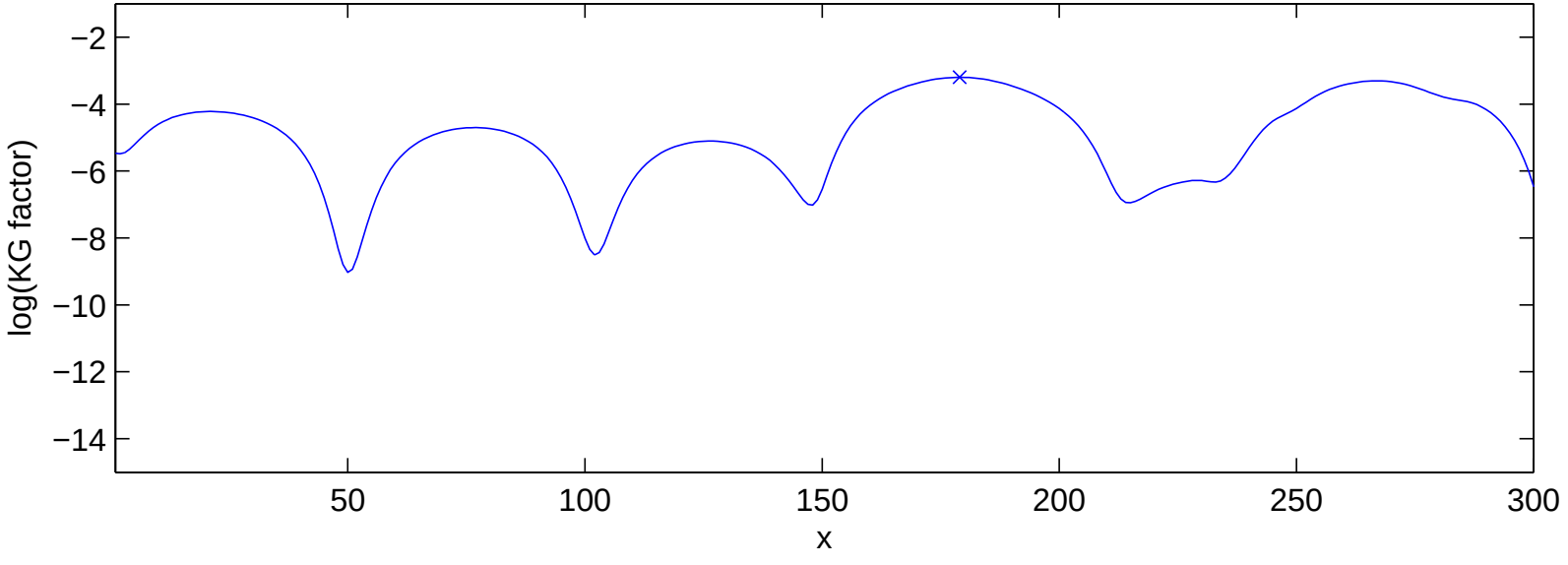
Global Optimization Example



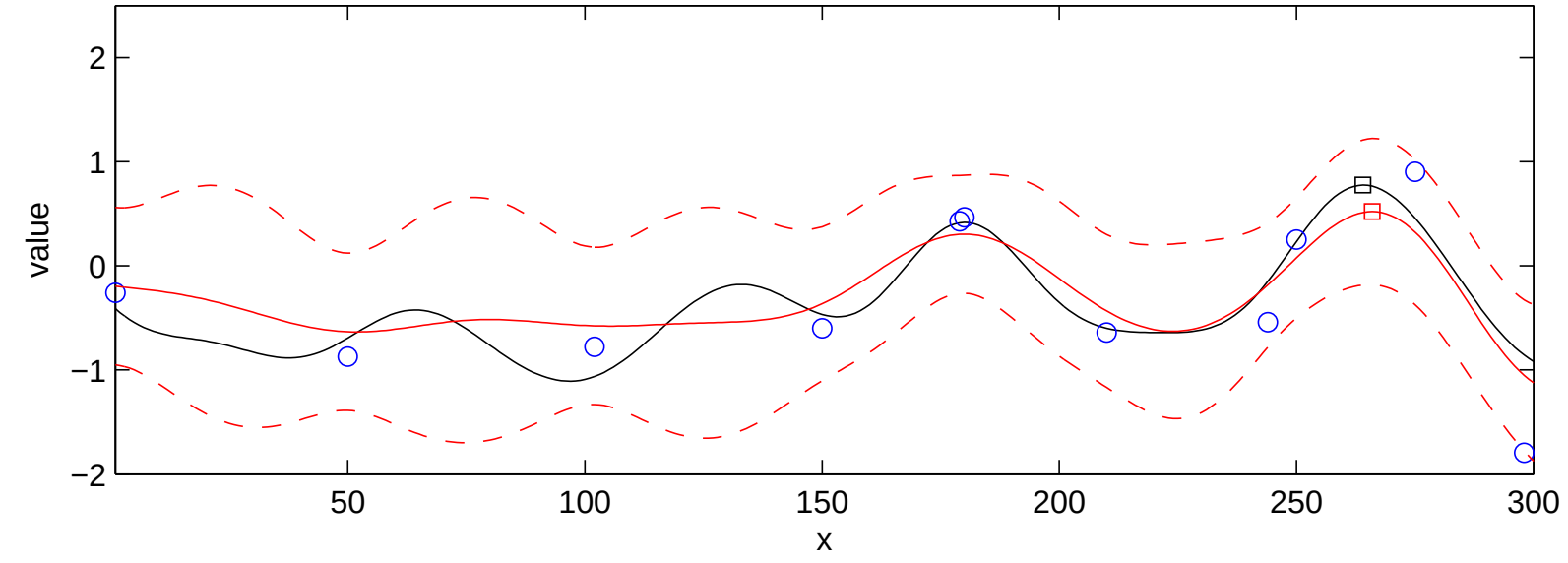


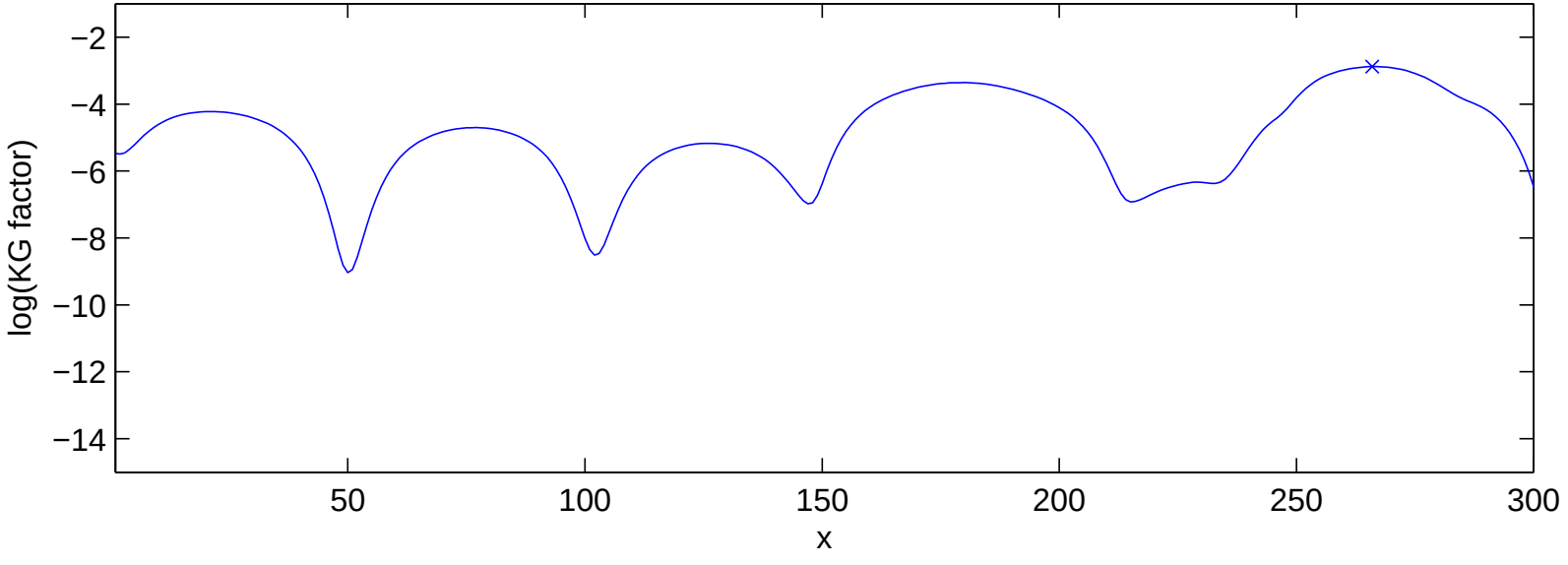
Global Optimization Example



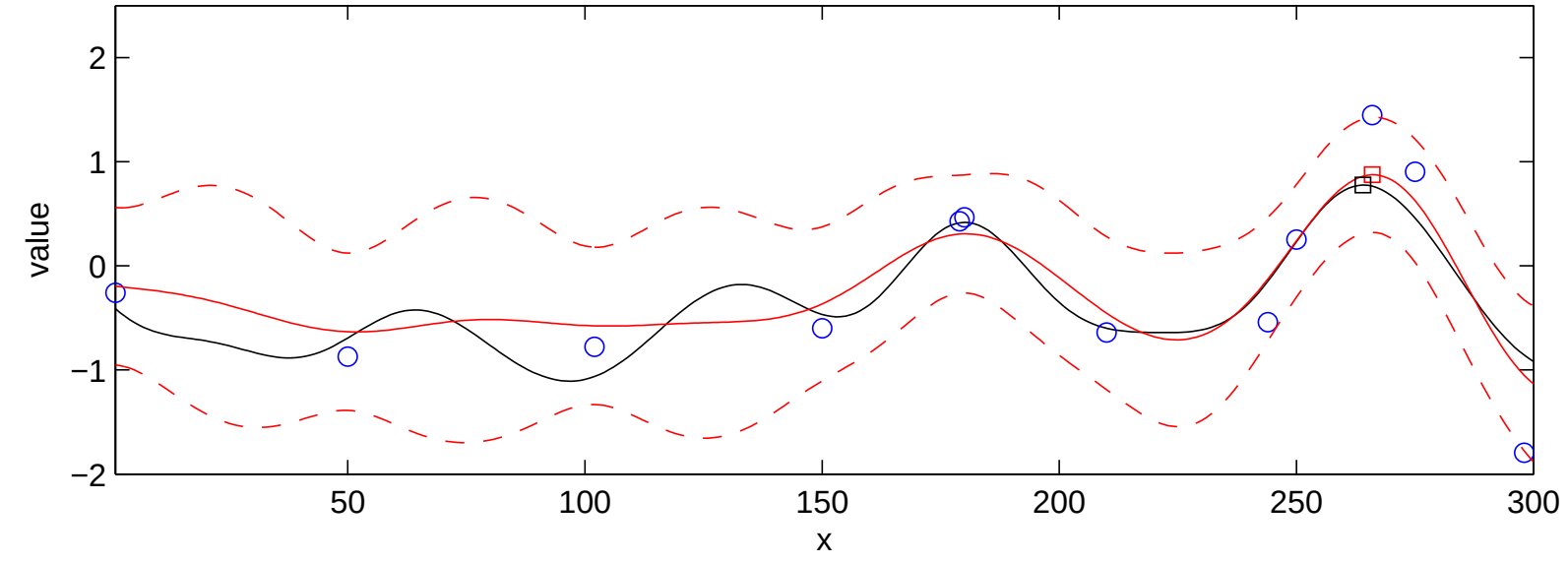


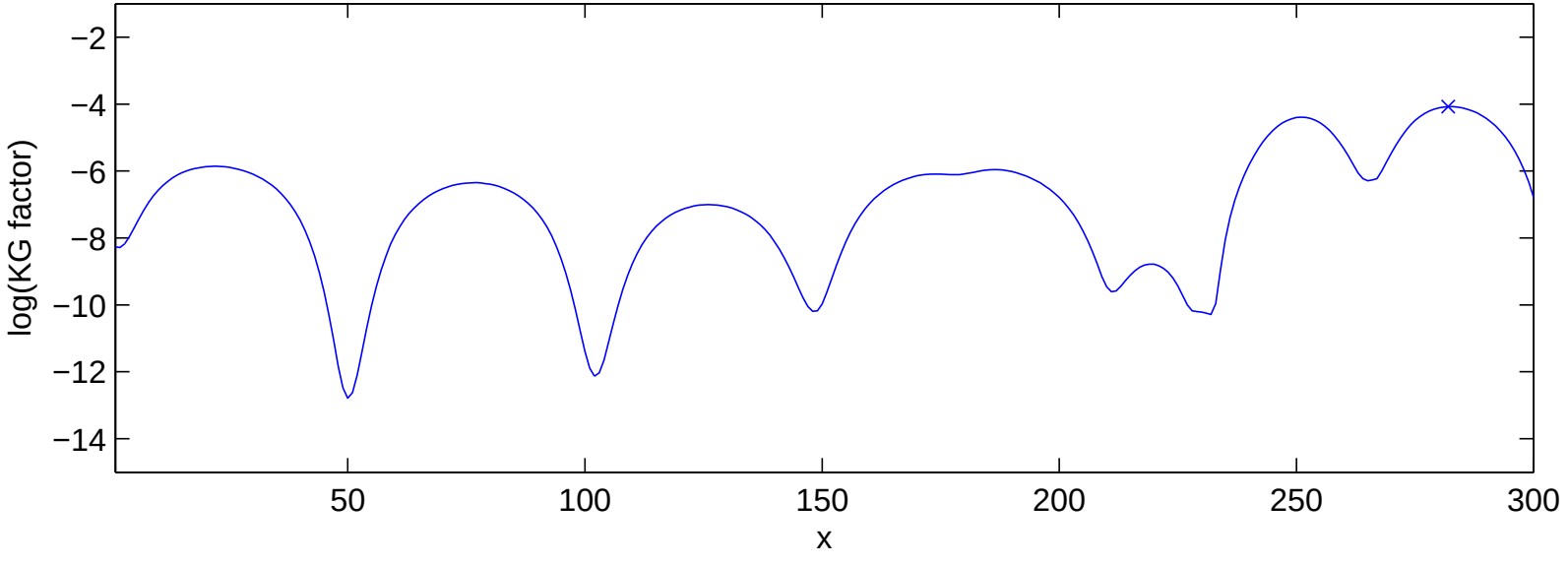
Global Optimization Example



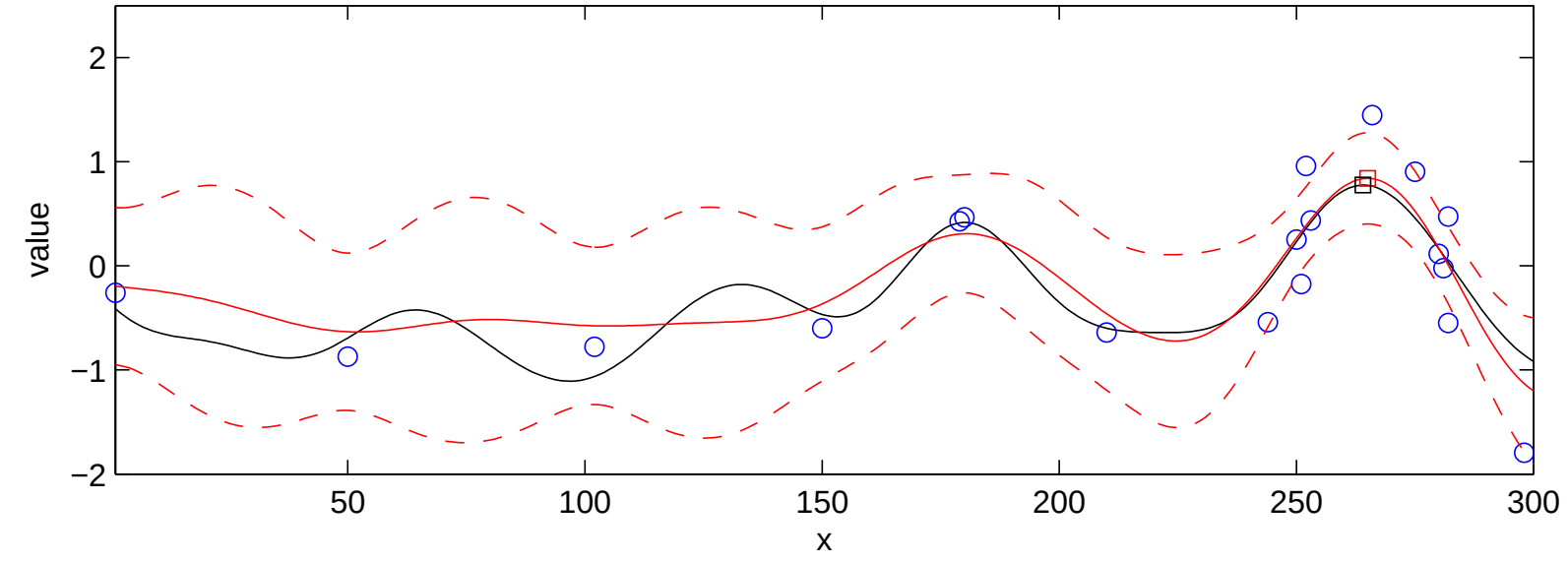


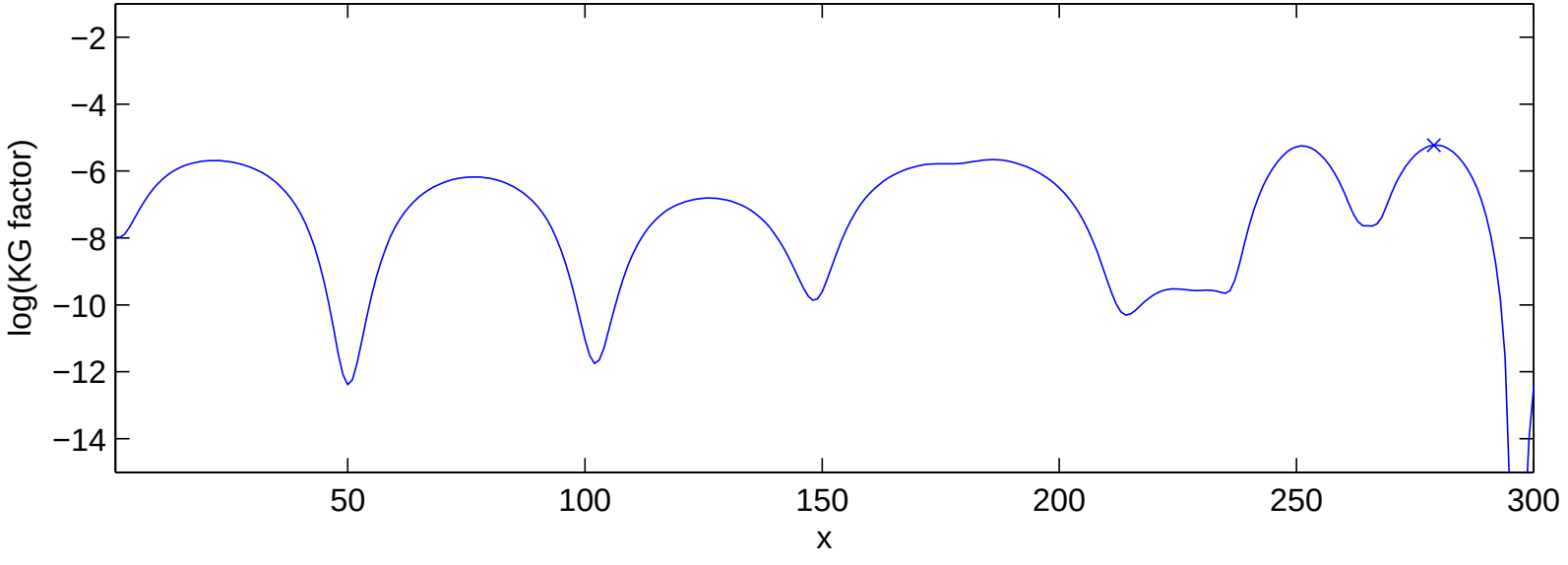
Global Optimization Example





Global Optimization Example





Computing KG Factors

- To compute the KG factor, we must (approximately) compute

$$\mathbb{E}_n \left[\sup_{x \in [0,1]^d} \mu_{n+1}(x) \mid x_n = x \right].$$

- We choose a finite set $A \subset [0,1]^d$ and approximate this as

$$\mathbb{E}_n \left[\max_{x \in A} \mu_{n+1}(x) \mid x_n = x \right].$$

- We can evaluate this expectation because $\mu_{n+1}(x) = \mu_N(x) + s_n(x)Z$, where Z is a **univariate** standard normal random variable, and the function s_n depends on x_n.
- As A gets bigger, the approximation becomes more accurate.

Computing KG Factors

In general, to compute the approximate KG factor for a candidate measurement:

- Let $A' = (x^1, \ldots, x^m) \subseteq A$ contain those points that are best under the time-$n+1$ posterior with nonzero probability.
- Sort A' in order of decreasing $s_n(x^i)$.
- The approximate KG factor is

$$\sum_{i=1}^{m-1} (s_n(x^{i+1}) - s_n(x^i)) g\left(\frac{-|\mu_n(x^{i+1}) - \mu_n(x^i)|}{s_n(x^{i+1}) - s_n(x^i)}\right),$$

where $g(z) = \varphi(z) + z\Phi(z)$,
φ is the normal pdf and Φ is the normal cdf.

Consistency

Consider a discretized problem in which x_n and $\widehat{x}_*$ are restricted to a finite set $\mathscr{X} \subset [0,1]^d$. Then:

Theorem

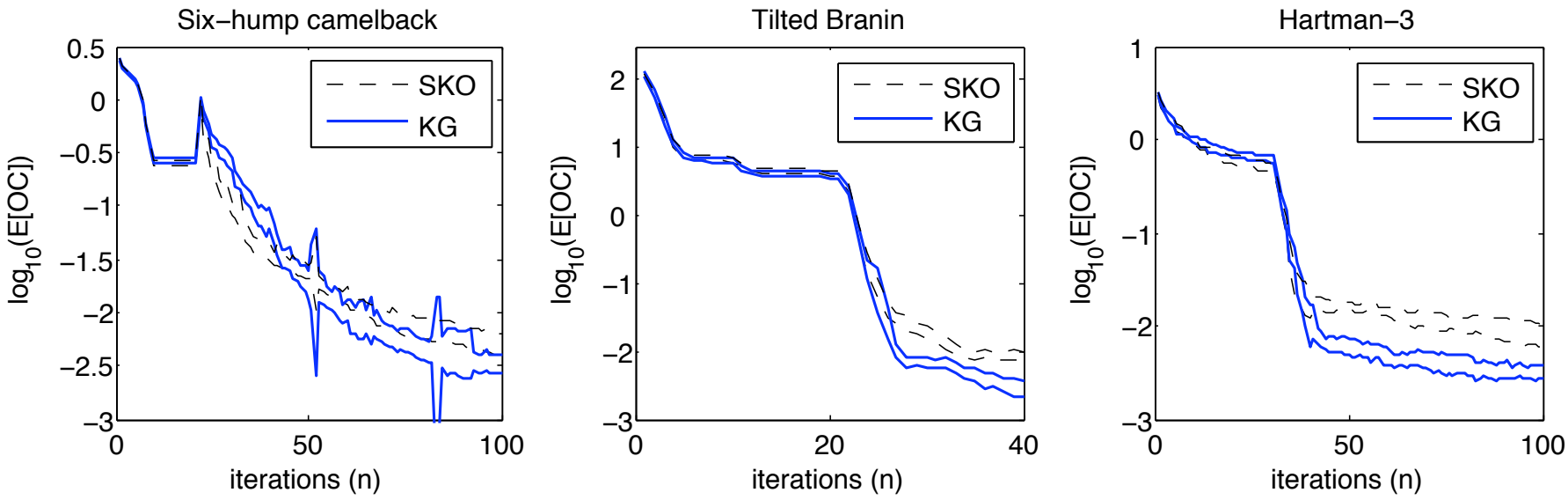
$$\lim_{N\to\infty} \arg\max_{x\in\mathscr{X}} \mu_N(x) = \arg\max_{x\in\mathscr{X}} f(x)$$

almost surely under $\mathbb{P}_0$.

This states that the global optimum is eventually discovered.
The proof uses a martingale convergence argument to show that every $x \in \mathscr{X}$ is evaluated infinitely often.

Global Optimization: Numerical Results

Sequential Kriging Optimization (SKO) is a recent method that uses Bayesian methods and spends a great deal of computational effort deciding where to sample next in order to minimize the number of function evaluations required.



Other Approaches to Continuous Global Optimization

- Many other derivative-free noise-tolerant global optimization methods exist, e.g.,
  - pattern search, e.g., Nelder-Mead
  - stochastic approximation, e.g., SPSA [Spall 1992].
  - evolutionary algorithms, simulated annealing, tabu search
- The KG method is a Bayesian global optimization (BGO) method because it places a Bayesian prior distribution on the underlying but unknown function. (See [Jones et al. 1998]).
- The posterior may understood as a surrogate model for the unknown function, and in this way BGO methods are similar to radial basis function methods [Gutmann 2001, Regis & Shoemaker 2007] and response surface methods [Myers & Montgomery 2002].
- Surrogate-based methods require more computation to decide where to evaluate next, but often require fewer evaluations to find global extrema.

Conclusion

- **We solved the dynamic program** for an idealized version of the stochastic root-finding problem. Other problems where preliminary results suggest that the DP can be solved:
  - Finding level sets of 1-dimensional continuous functions.
  - Multiple comparisons with a standard / finding level sets.
  - A search problem arising in computer vision.
- We used dynamic programming to find a **good heuristic policy** for stochastic global optimization. This technique is general and seems reasonably likely to work well in many problems.
- Dynamic programming is a **useful theoretical tools** for developing algorithms for simulation optimization and other applications requiring heavy computation.

Thank You

Any questions?