Probabilistic Numerics

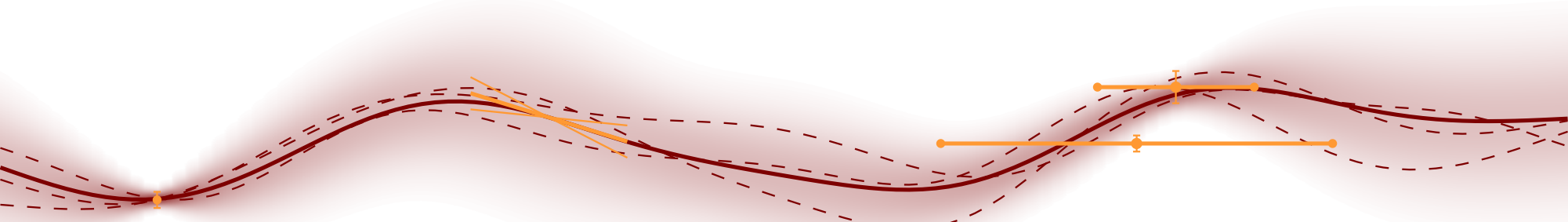
V – Nonlinear Optimization

Philipp Hennig

Dobbiaco Summer School
22 June 2017

Research Group for Probabilistic Numerics
Max Planck Institute for Intelligent Systems
Tübingen, Germany

Some of the presented work was supported by
the Emmy Noether Programme of the DFG



Today:

- some “probabilistic” challenges in **nonlinear optimization**
- parameter adaptation in stochastic optimization problems
- **global optimization**

Big Picture

Formulation of Computation as Inference

Estimate z from computations c, under model m.

Prior: structural knowledge reduces complexity

Likelihood: modeling imprecise computation reduces cost

$$p(z \mid c, m) = \frac{p(z \mid m)p(c \mid z, m)}{\int p(z \mid m)p(c \mid z, m)\, dz}$$

Posterior: adaptive evaluations for efficient convergence

Evidence:

- so far: classic methods can be interpreted as MAP estimators
- today: new functionality, constructed from the probabilistic viewpoint

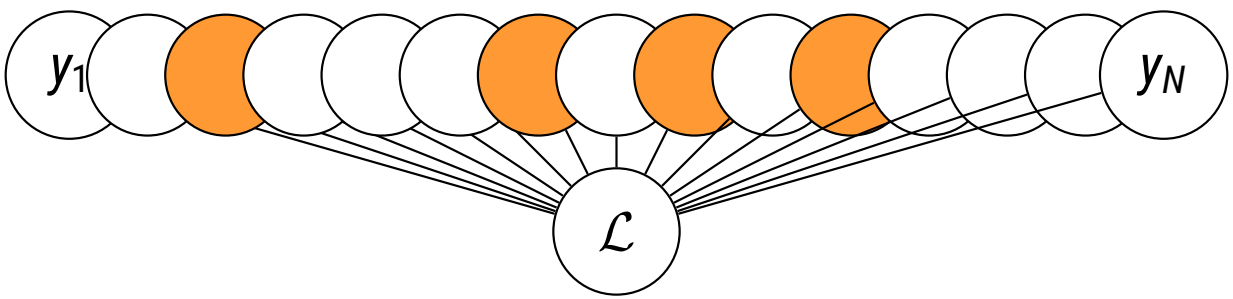
Attack of the Stochastic Gradients

A fundamental challenge in Big Data settings

In Big Data setting, batching introduces (Gaussian) noise

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \ell(y_i;\theta) \approx \frac{1}{M}\sum_{j=1}^{M} \ell(y_j;\theta) =: \hat{\mathcal{L}}(\theta) \qquad M \ll N$$

$$p(\hat{\mathcal{L}} \mid \mathcal{L}) \approx \mathcal{N}\left(\hat{\mathcal{L}}; \mathcal{L}, \mathcal{O}\left(\frac{N-M}{M}\right)\right)$$



Problems for Existing Optimizers

Stability, Identifiability

- BFGS is unstable
- even GD needs step size

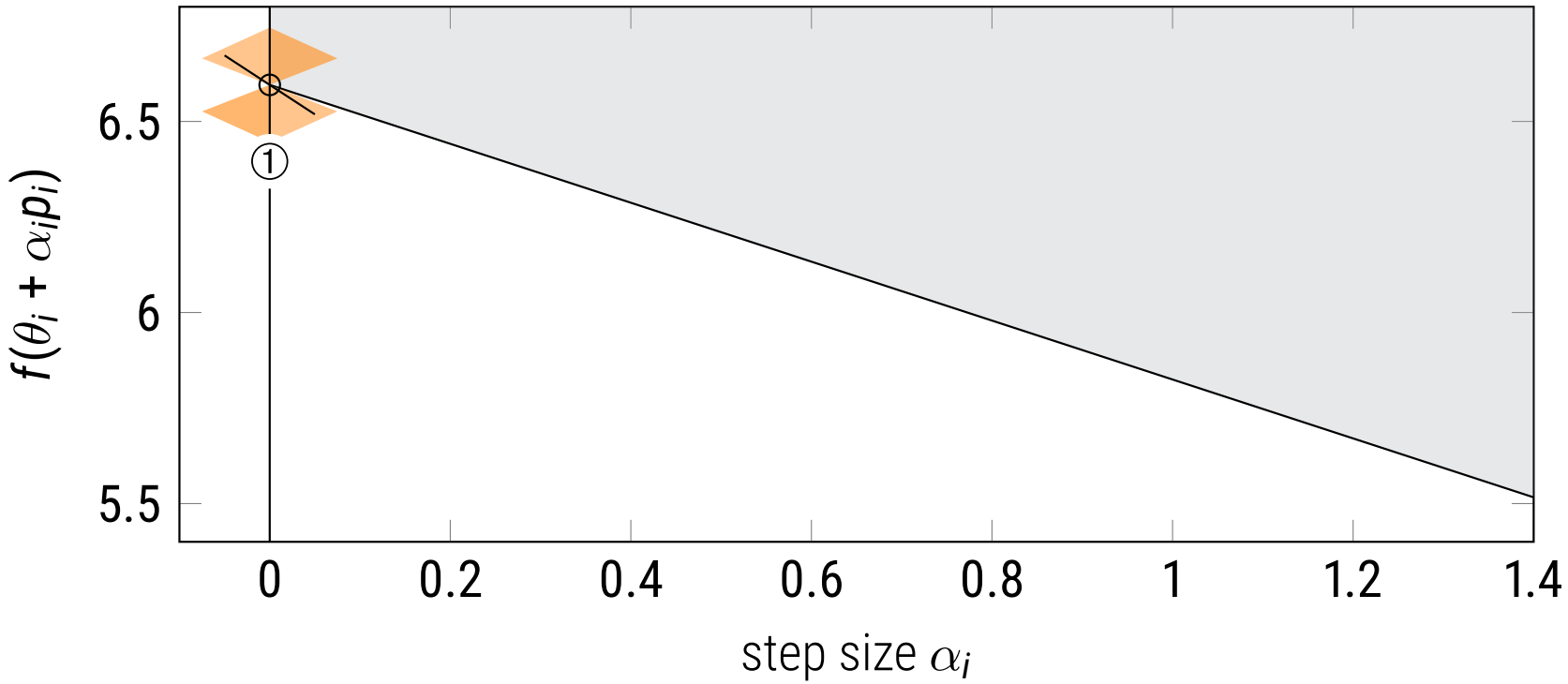
$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \alpha_i \boldsymbol{p}_i = \boldsymbol{\theta}_i - \alpha_i \nabla \hat{\mathcal{L}}(\theta_i) \qquad \text{set } \alpha_i \text{ “well”}$$

Can we infer the missing degrees of freedom?

Line Searches

a case study

Nocedal & Wright, Alg. 3.5



The (weak) Wolfe conditions:

SIREV, 1969

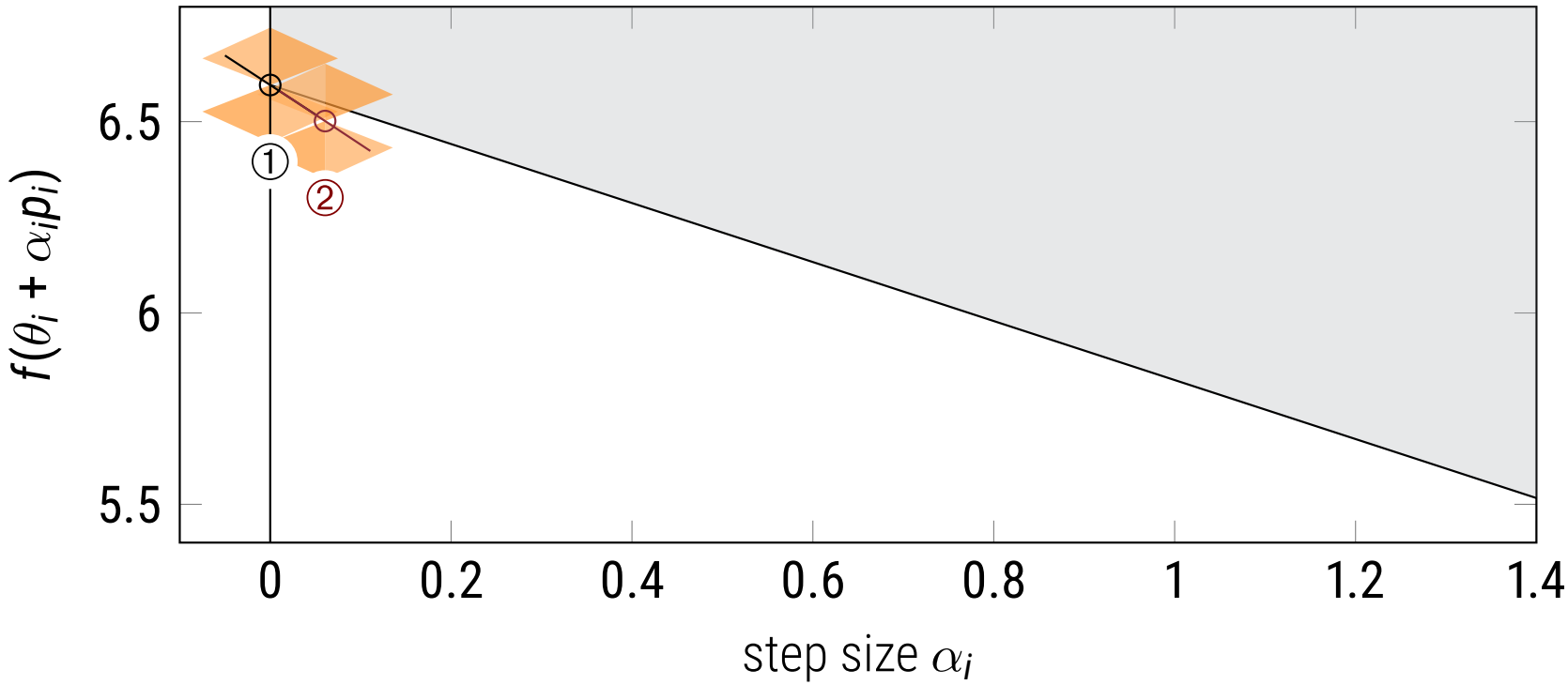
$$f(x_i + \alpha_i p_i) \leq f(x_i) + c_1 \alpha_i \nabla f(x_i)^\intercal p_i \qquad \text{Armijo condition (1966)}$$

$$\nabla f(x_i + \alpha_i p_i)^\intercal p_i \geq c_2 \nabla f(x_i)^\intercal p_i \qquad \text{curvature condition}$$

Line Searches

a case study

Nocedal & Wright, Alg. 3.5



The **strong** Wolfe conditions: SIREV, 1969

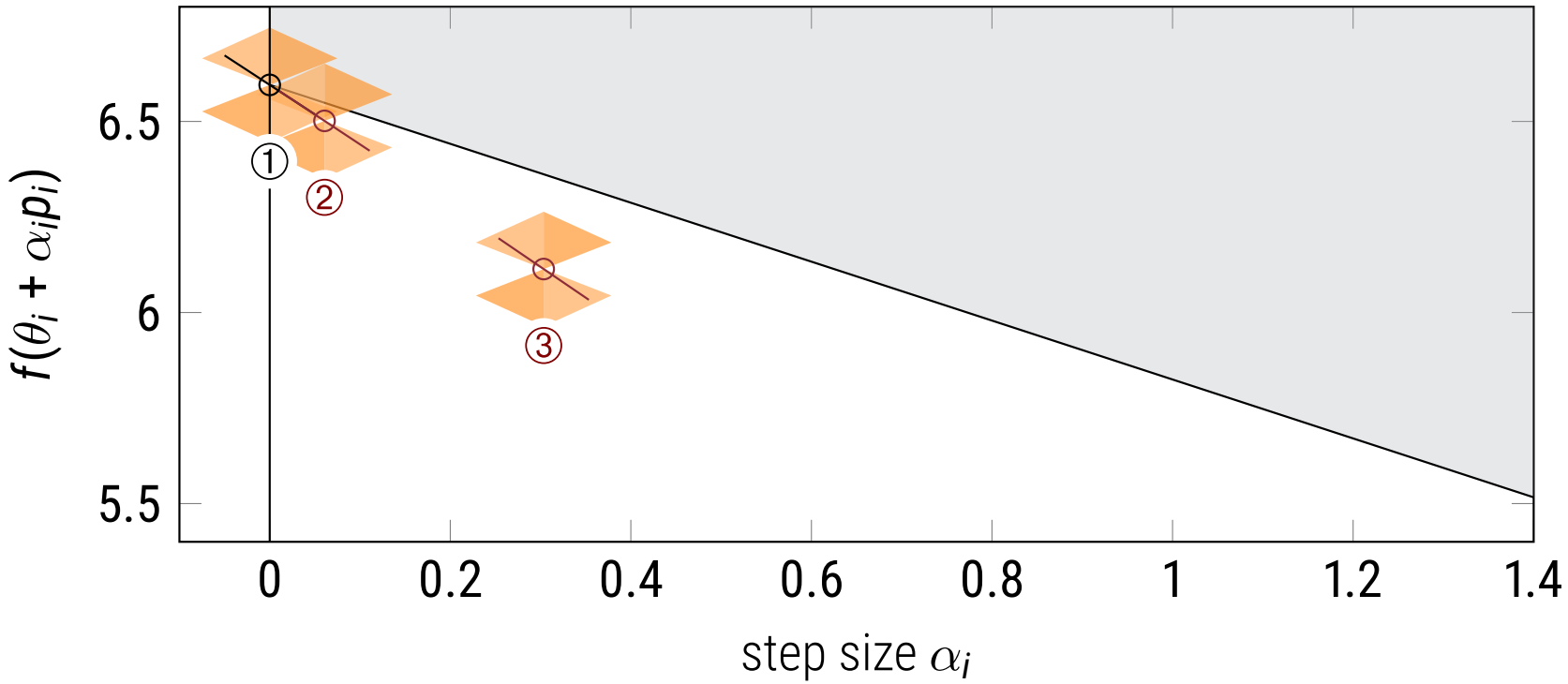
$$f(x_i + \alpha_i p_i) \leq f(x_i) + c_1 \alpha_i \nabla f(x_i)^\intercal p_i \qquad \text{Armijo condition (1966)}$$

$$|\nabla f(x_i + \alpha_i p_i)^\intercal p_i| \leq c_2 |\nabla f(x_i)^\intercal p_i| \qquad \text{curvature condition}$$

Line Searches

a case study

Nocedal & Wright, Alg. 3.5



The **strong** Wolfe conditions: SIREV, 1969

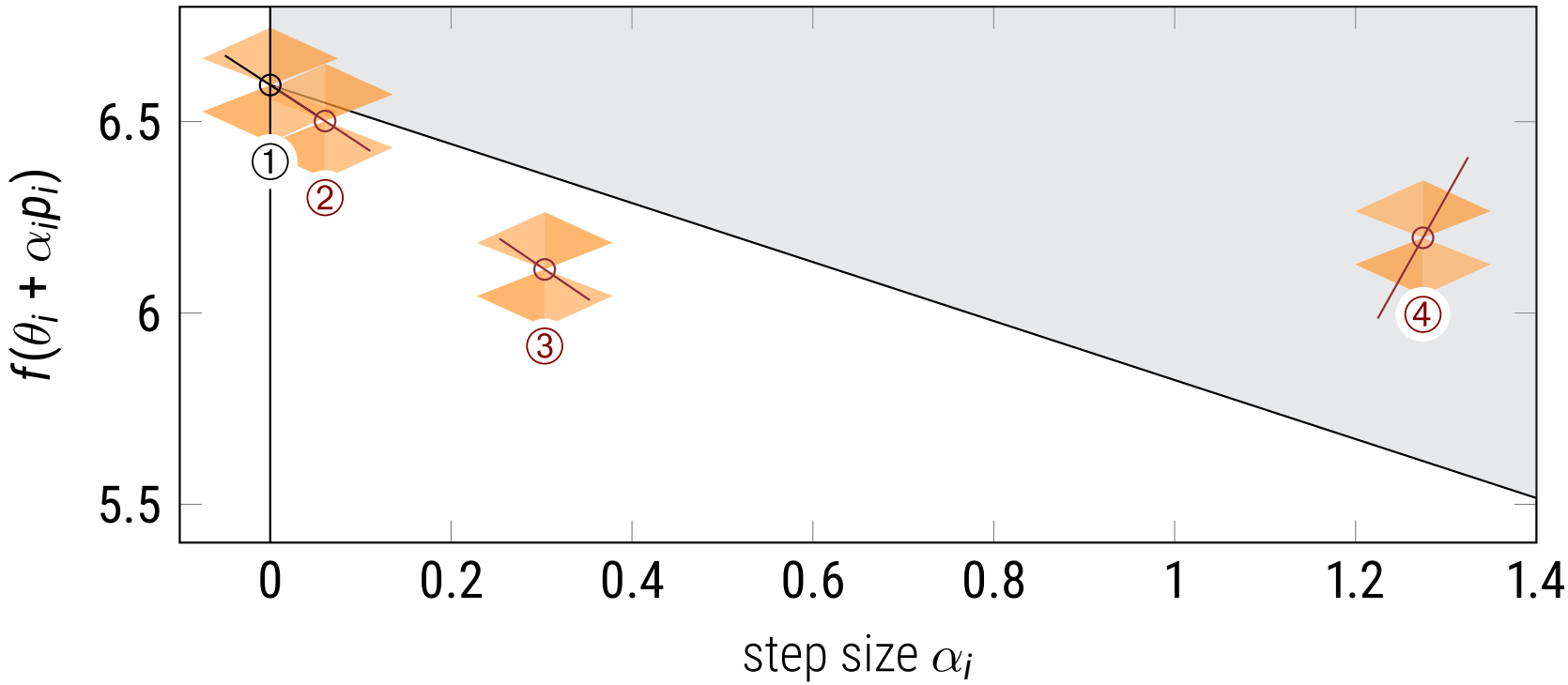
$$f(x_i + \alpha_i p_i) \leq f(x_i) + c_1 \alpha_i \nabla f(x_i)^\intercal p_i \qquad \text{Armijo condition (1966)}$$

$$|\nabla f(x_i + \alpha_i p_i)^\intercal p_i| \leq c_2 |\nabla f(x_i)^\intercal p_i| \qquad \text{curvature condition}$$

Line Searches

a case study

Nocedal & Wright, Alg. 3.5



The **strong** Wolfe conditions: SIREV, 1969

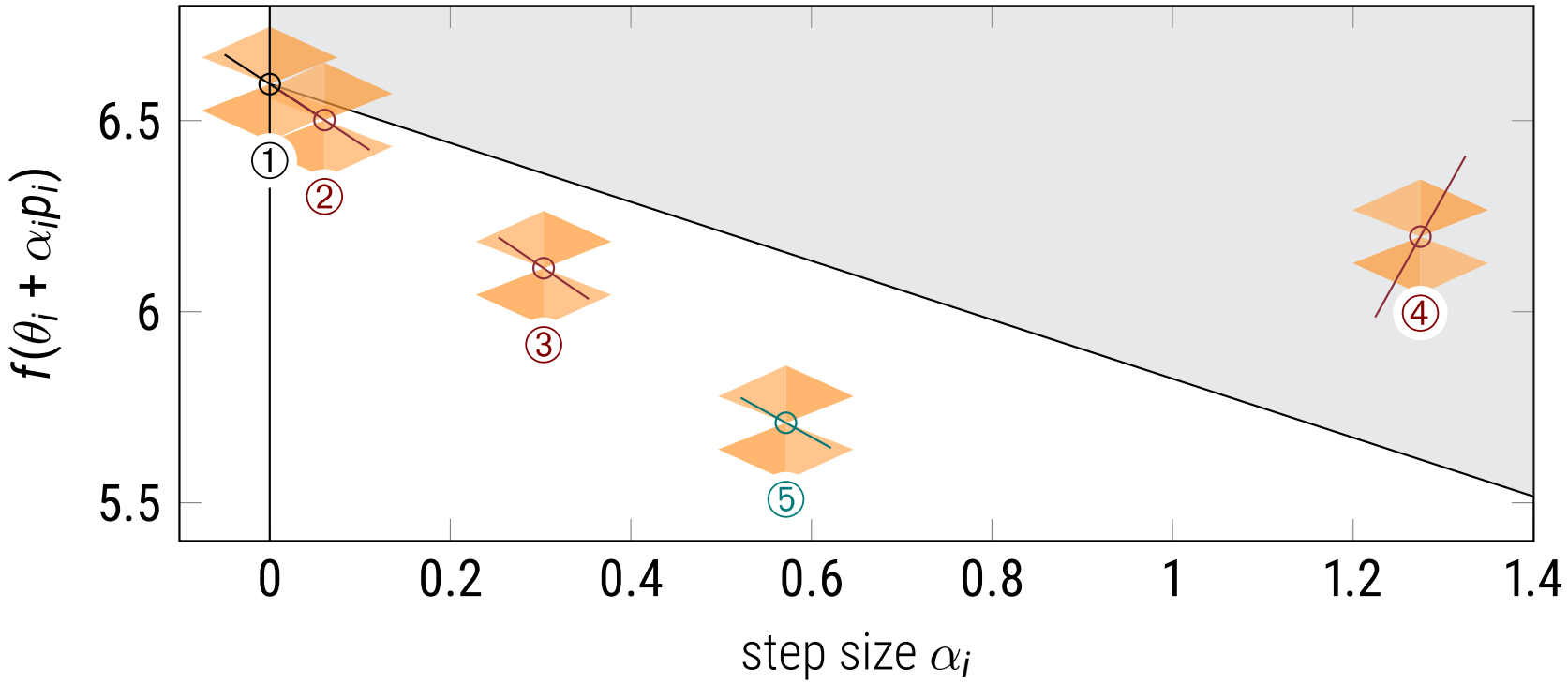
$$f(x_i + \alpha_i p_i) \leq f(x_i) + c_1 \alpha_i \nabla f(x_i)^\intercal p_i \qquad \text{Armijo condition (1966)}$$

$$|\nabla f(x_i + \alpha_i p_i)^\intercal p_i| \leq c_2 |\nabla f(x_i)^\intercal p_i| \qquad \text{curvature condition}$$

Line Searches

a case study

Nocedal & Wright, Alg. 3.5



The **strong** Wolfe conditions: SIREV, 1969

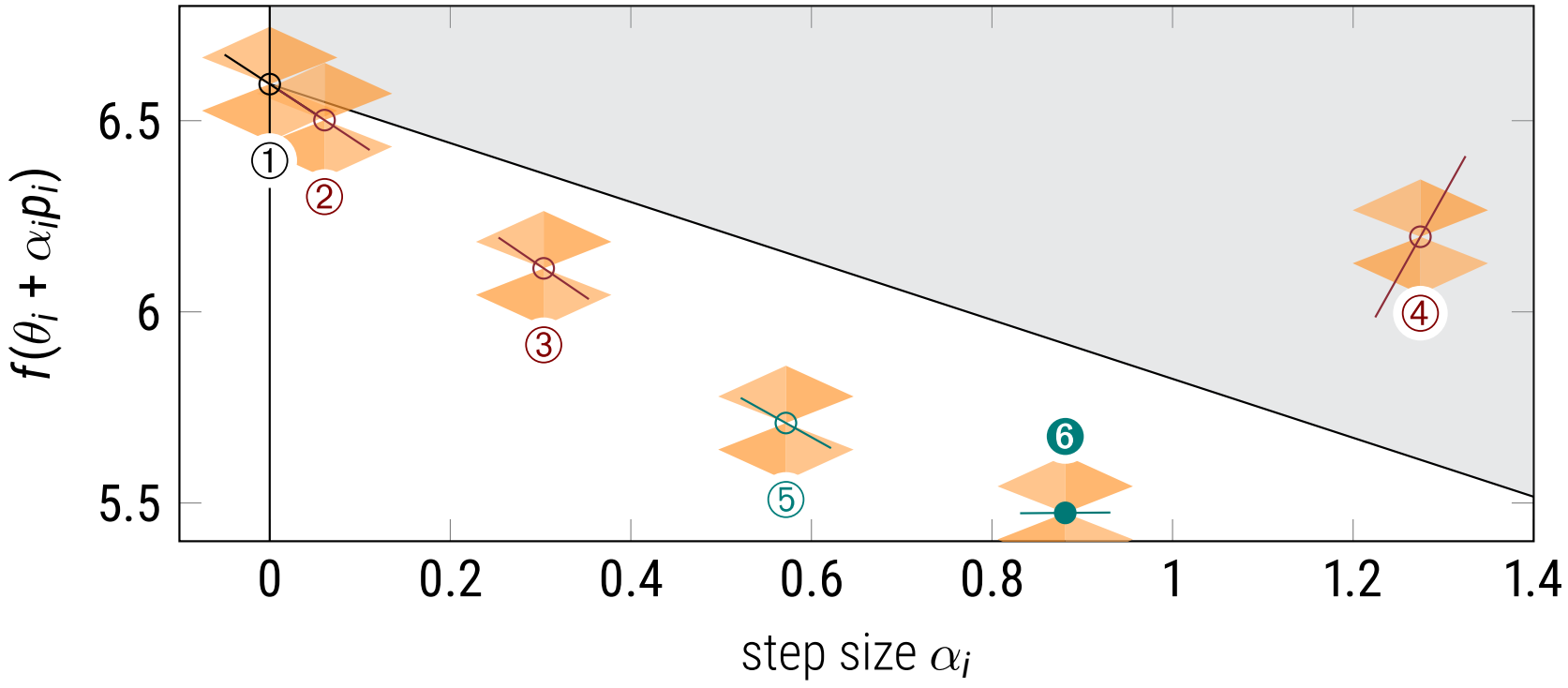
$$f(x_i + \alpha_i p_i) \leq f(x_i) + c_1 \alpha_i \nabla f(x_i)^\intercal p_i \qquad \text{Armijo condition (1966)}$$

$$|\nabla f(x_i + \alpha_i p_i)^\intercal p_i| \leq c_2 |\nabla f(x_i)^\intercal p_i| \qquad \text{curvature condition}$$

Line Searches

a case study

Nocedal & Wright, Alg. 3.5



The **strong** Wolfe conditions: SIREV, 1969

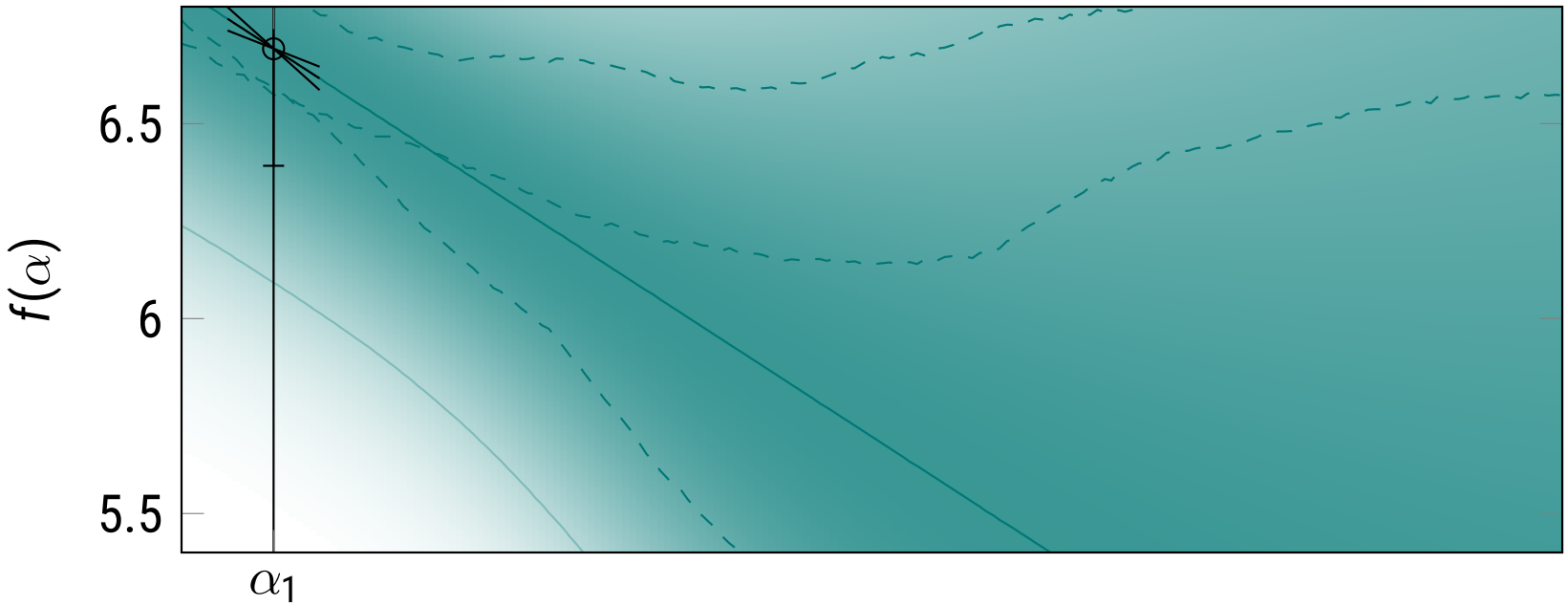
$$f(x_i + \alpha_i p_i) \leq f(x_i) + c_1 \alpha_i \nabla f(x_i)^\intercal p_i \qquad \text{Armijo condition (1966)}$$

$$|\nabla f(x_i + \alpha_i p_i)^\intercal p_i| \leq c_2 |\nabla f(x_i)^\intercal p_i| \qquad \text{curvature condition}$$

Probabilistic Line Searches

model

Mahsereci & Hennig, NIPS 2015



- $p(f) = \mathcal{GP}(0, k)$ with ($\tilde{\alpha} = \alpha - \alpha_0$, set $\alpha_0 = 10$)

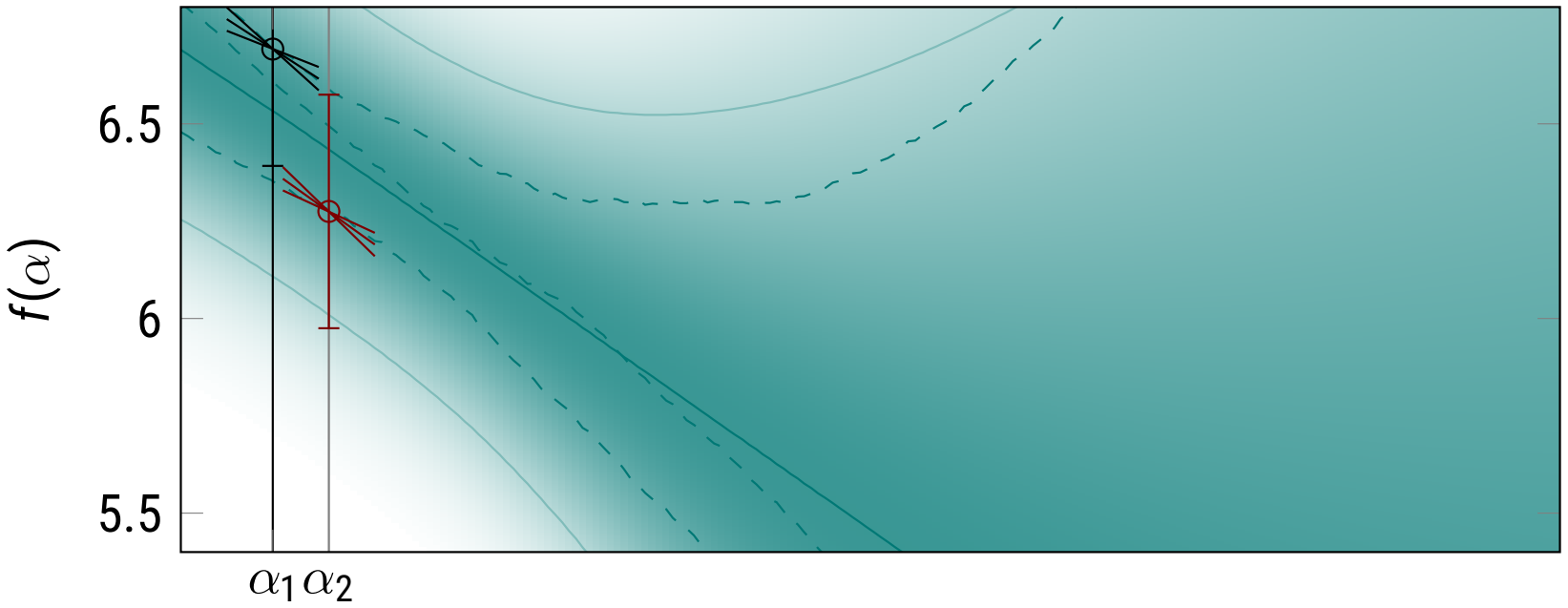
$$k(\alpha, \alpha') = \theta^2 \left[1/3 \min^3(\tilde{\alpha}, \tilde{\alpha}') + 1/2 |\tilde{\alpha} - \tilde{\alpha}'| \min^2(\tilde{\alpha}, \tilde{\alpha}') \right].$$

- gives **piecewise cubic spline interpolants**

Probabilistic Line Searches

model

Mahsereci & Hennig, NIPS 2015



- $p(f) = \mathcal{GP}(0, k)$ with ($\tilde{\alpha} = \alpha - \alpha_0$, set $\alpha_0 = 10$)

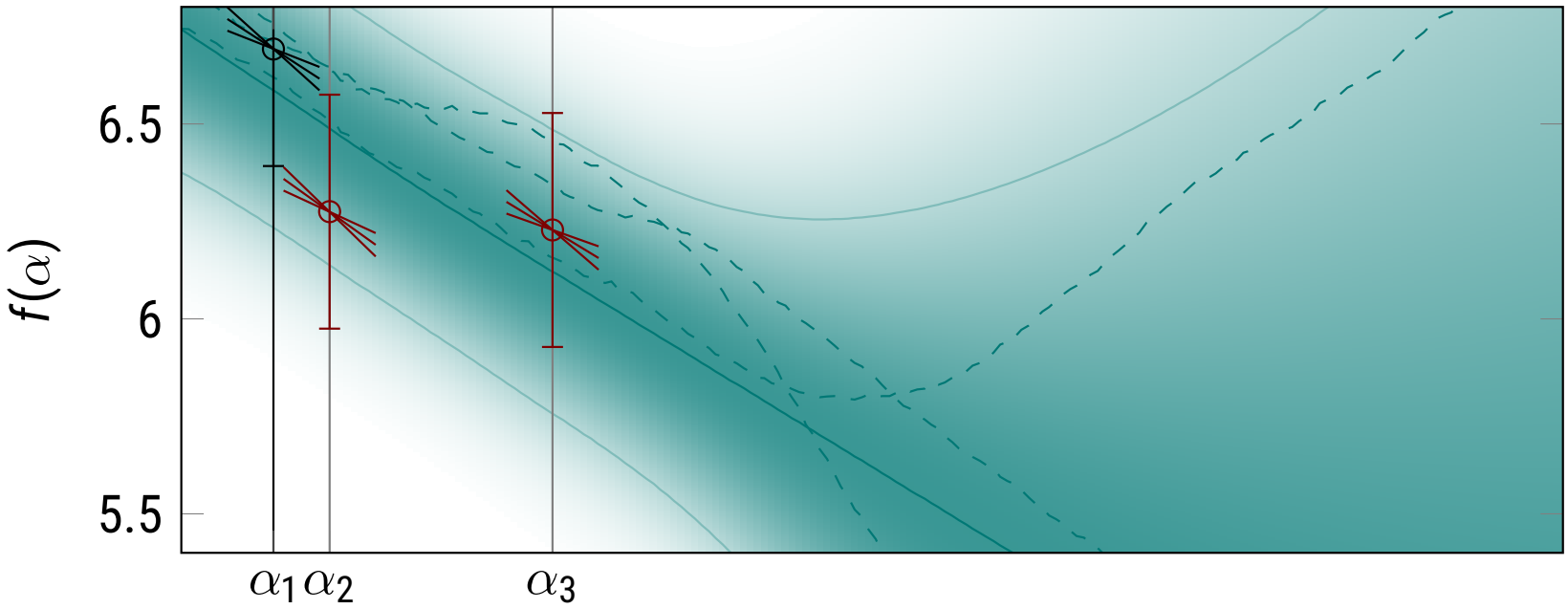
$$k(\alpha, \alpha') = \theta^2 \left[1/3 \min^3(\tilde{\alpha}, \tilde{\alpha}') + 1/2 |\tilde{\alpha} - \tilde{\alpha}'| \min^2(\tilde{\alpha}, \tilde{\alpha}') \right].$$

- gives **piecewise cubic spline interpolants**

Probabilistic Line Searches

model

Mahsereci & Hennig, NIPS 2015



- $p(f) = \mathcal{GP}(0, k)$ with ($\tilde{\alpha} = \alpha - \alpha_0$, set $\alpha_0 = 10$)

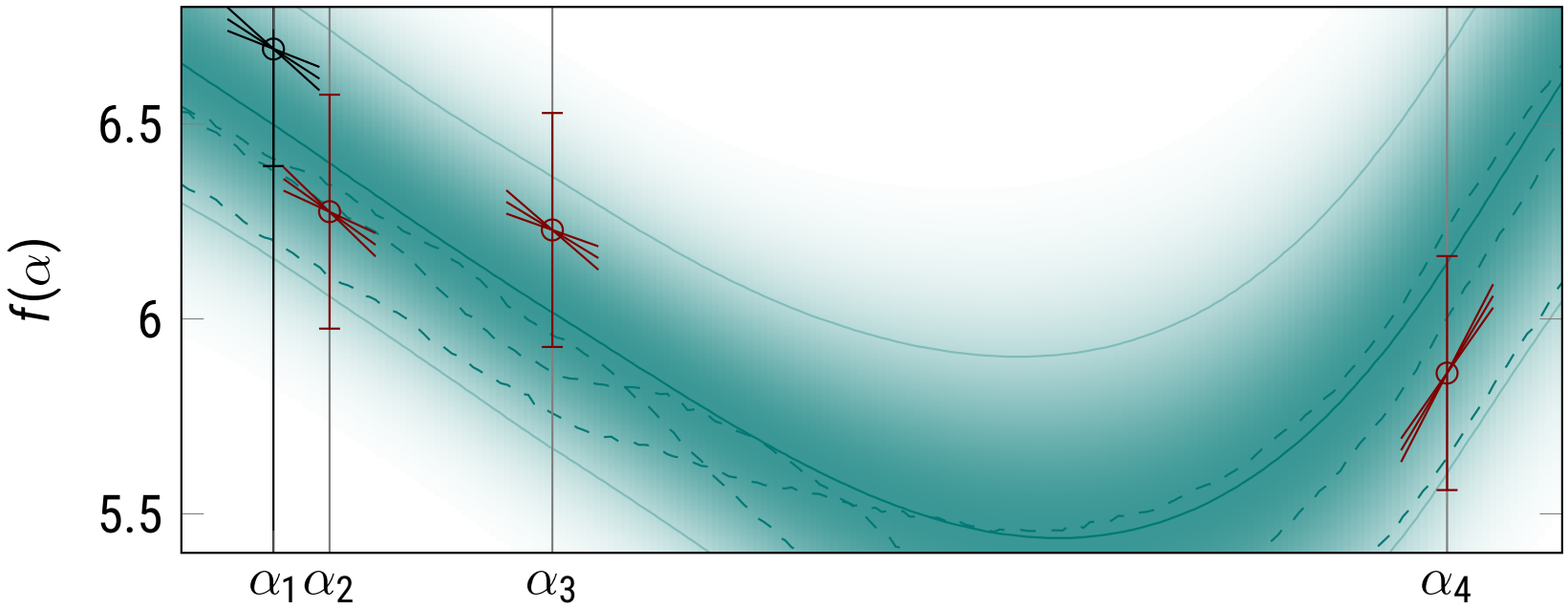
$$k(\alpha, \alpha') = \theta^2 \left[1/3 \min^3(\tilde{\alpha}, \tilde{\alpha}') + 1/2 |\tilde{\alpha} - \tilde{\alpha}'| \min^2(\tilde{\alpha}, \tilde{\alpha}') \right].$$

- gives **piecewise cubic spline interpolants**

Probabilistic Line Searches

model

Mahsereci & Hennig, NIPS 2015



- $p(f) = \mathcal{GP}(0, k)$ with ($\tilde{\alpha} = \alpha - \alpha_0$, set $\alpha_0 = 10$)

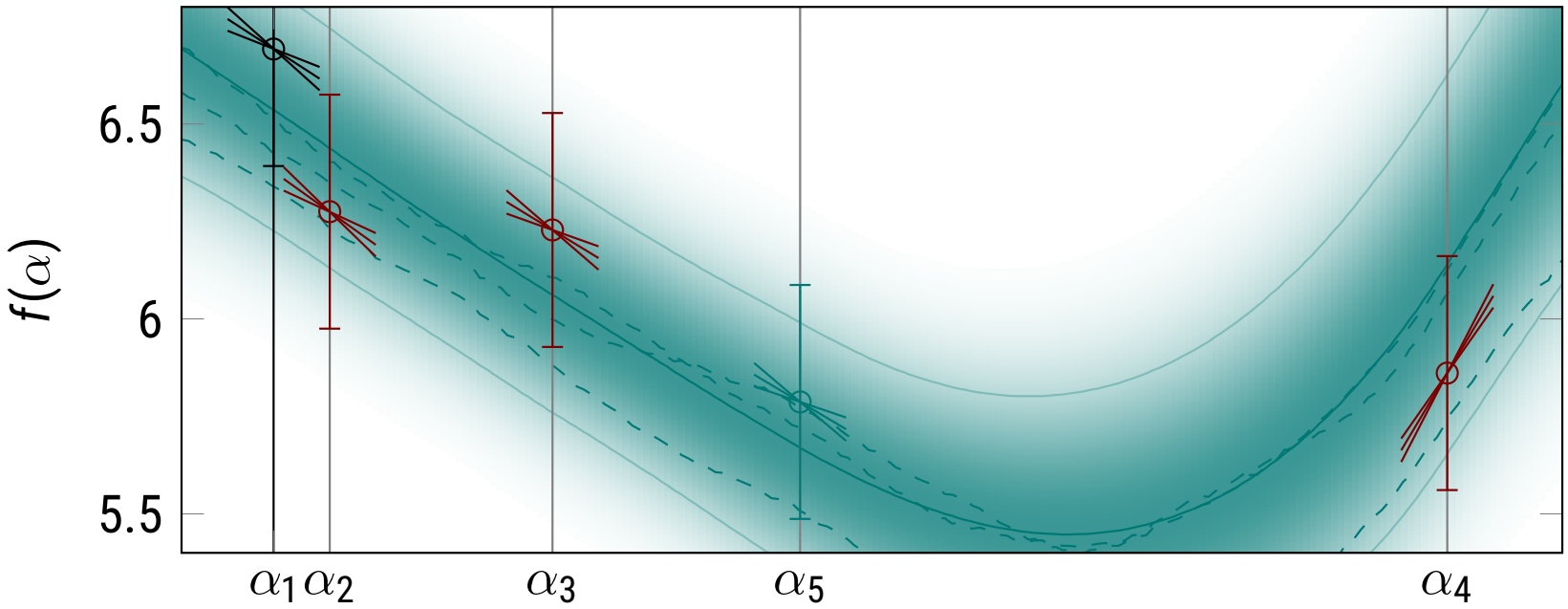
$$k(\alpha, \alpha') = \theta^2 \left[1/3 \min^3(\tilde{\alpha}, \tilde{\alpha}') + 1/2 |\tilde{\alpha} - \tilde{\alpha}'| \min^2(\tilde{\alpha}, \tilde{\alpha}') \right].$$

- gives **piecewise cubic spline interpolants**

Probabilistic Line Searches

model

Mahsereci & Hennig, NIPS 2015



- $p(f) = \mathcal{GP}(0, k)$ with ($\tilde{\alpha} = \alpha - \alpha_0$, set $\alpha_0 = 10$)

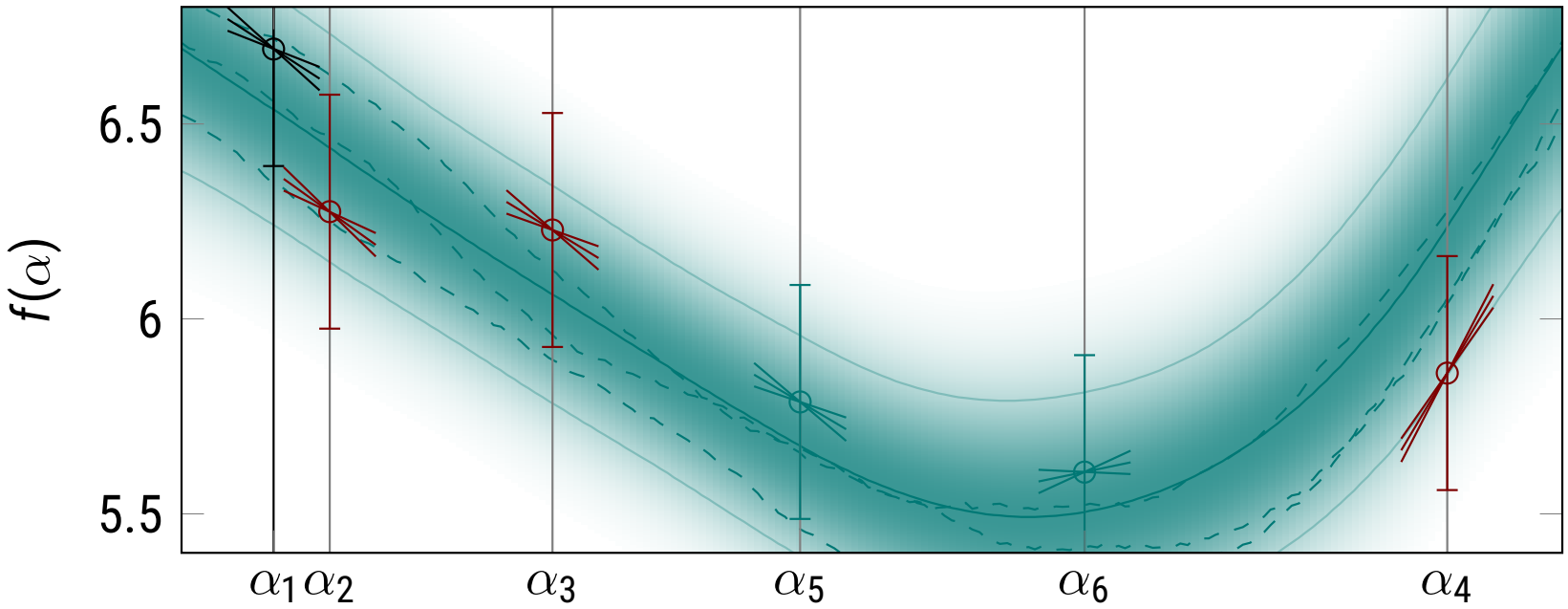
$$k(\alpha, \alpha') = \theta^2 \left[1/3 \min^3(\tilde{\alpha}, \tilde{\alpha}') + 1/2 |\tilde{\alpha} - \tilde{\alpha}'| \min^2(\tilde{\alpha}, \tilde{\alpha}') \right].$$

- gives **piecewise cubic spline interpolants**

Probabilistic Line Searches

model

Mahsereci & Hennig, NIPS 2015



- $p(f) = \mathcal{GP}(0, k)$ with ($\tilde{\alpha} = \alpha - \alpha_0$, set $\alpha_0 = 10$)

$$k(\alpha, \alpha') = \theta^2 \left[1/3 \min^3(\tilde{\alpha}, \tilde{\alpha}') + 1/2 |\tilde{\alpha} - \tilde{\alpha}'| \min^2(\tilde{\alpha}, \tilde{\alpha}') \right].$$

- gives **piecewise cubic spline interpolants**

Probabilistic Line Searches

search

Mahsereci & Hennig, NIPS 2015

- after M evaluations (f_i, f_i'), posterior mean

$$m(\alpha) = \begin{bmatrix} k_{\alpha,\boldsymbol{\alpha}} & k_{\alpha,\partial\boldsymbol{\alpha}} \end{bmatrix} \begin{bmatrix} k_{\boldsymbol{\alpha},\boldsymbol{\alpha}} & k_{\boldsymbol{\alpha},\partial\boldsymbol{\alpha}} \\ k_{\partial\boldsymbol{\alpha},\boldsymbol{\alpha}} & k_{\partial\boldsymbol{\alpha},\partial\boldsymbol{\alpha}} \end{bmatrix}^{-1} \begin{bmatrix} Y_f \\ Y_{f'} \end{bmatrix}$$

  is a piece-wise cubic spline
- hence, in each cell $[\alpha_{i-1}, \alpha_i]$, there is a unique local minimum. If it's not at the corner, keep it in a list $I = \{\hat{\alpha}_1, \ldots, \hat{\alpha}_M\}$ (also add an extrapolation point)
- pick point $\alpha_* = \arg\max_{\hat{\alpha}_i} \{u(\hat{\alpha}_i)\}$
- for utilities u, see later today
- note: all this costs at most $\mathcal{O}(M)$
- but it requires the noise variance / SNR!
- which can be inferred empirically, from

$$R(\theta) = \frac{1}{M-1} \left(\frac{1}{M} \sum_{j=1}^{M} (\nabla \ell(y_j, \theta)).^2 - (\nabla \hat{\mathcal{L}}(\theta))^2 \right)$$

Probabilistic Line Searches

termination: probabilistic Wolfe conditions

Mahsereci & Hennig, NIPS 2015

$$f(\alpha) \leq f(0) + c_1 \alpha f'(0) \quad \text{(W-I)} \qquad \text{and}$$
$$f'(\alpha) \geq c_2 f'(0) \quad \text{(W-II)}$$

$$\begin{bmatrix} a_\alpha \\ b_\alpha \end{bmatrix} = \begin{bmatrix} 1 & c_1\alpha & -1 & 0 \\ 0 & -c_2 & 0 & 1 \end{bmatrix} \begin{bmatrix} f(0) \\ f'(0) \\ f(\alpha) \\ f'(\alpha) \end{bmatrix} \geq 0.$$

$$p(a_\alpha, b_\alpha) = \mathcal{N}\left(\begin{bmatrix} a_\alpha \\ b_\alpha \end{bmatrix}; \begin{bmatrix} m_\alpha^a \\ m_\alpha^b \end{bmatrix}, \begin{bmatrix} C_\alpha^{aa} & C_\alpha^{ab} \\ C_\alpha^{ba} & C_\alpha^{bb} \end{bmatrix} \right),$$

with $m_\alpha^a = \mu(0) - \mu(\alpha) + c_1 \alpha \mu'(0)$ and $m_\alpha^b = \mu'(\alpha) - c_2 \mu'(0)$

and $C_\alpha^{aa} = k_{00} + (c_1\alpha)^2 k_{\partial 0, \partial 0} + k_{\alpha\alpha} + 2[c_1\alpha(k_{0,\partial 0} - k_{\partial 0, t}) - k_{0\alpha}]$

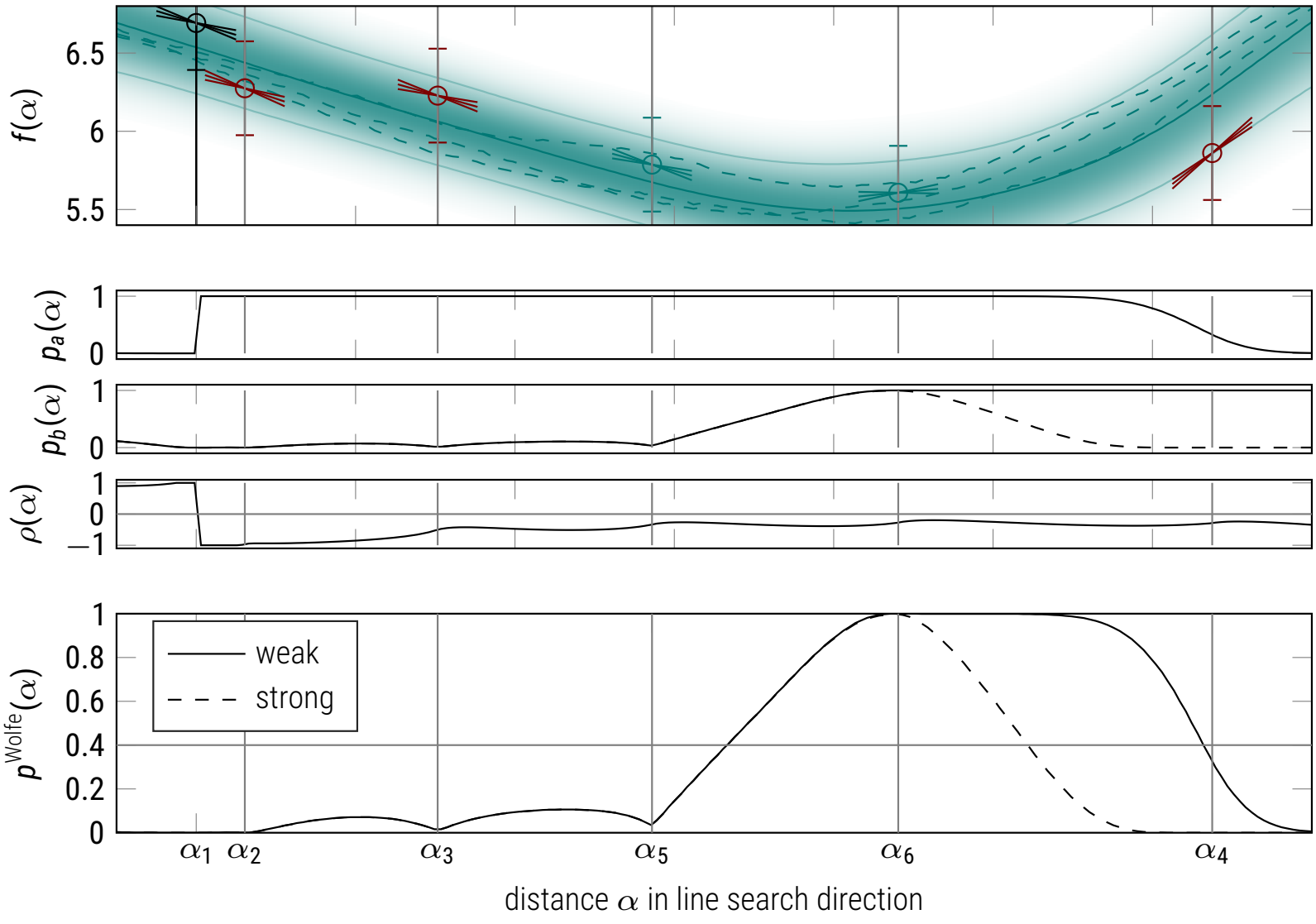
$C_\alpha^{bb} = c_2^2 k_{\partial 0, \partial 0} - 2c_2 k_{\partial 0, \partial \alpha} + k_{\partial\alpha, \partial\alpha}$

$C_\alpha^{ab} = C_t^{ba} = -c_2(k_{0,\partial 0} + c_1\alpha k_{\partial 0, \partial 0}) + (1 + c_2) k_{\partial 0, \alpha} + c_1 \alpha k_{\partial 0, \partial\alpha} - k_{\alpha, \partial\alpha}.$

Probabilistic Line Searches

termination: probabilistic Wolfe conditions

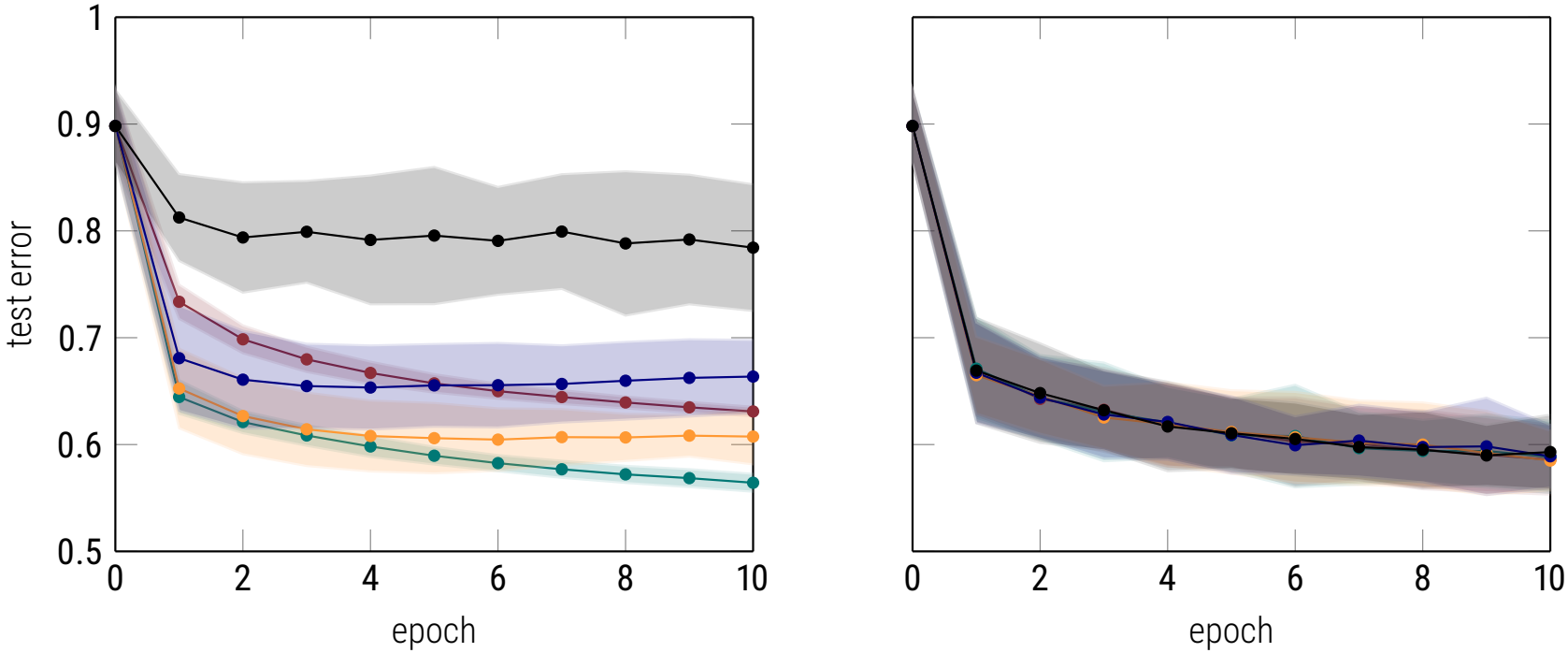
Mahsereci & Hennig, NIPS 2015



Probabilistic Line Searches

experiments

Mahsereci & Hennig, NIPS 2015



two-layer feed-forward perceptron on CIFAR 10. Details, additional results in Mahsereci & Hennig, NIPS 2015.

https://github.com/ProbabilisticNumerics/probabilistic_line_search

- nontrivial likelihoods can be used to **regularize**, **stabilize** classic methods
- this can be used to build inference rules for **algorithmic parameters**
- doing so does not necessarily raise computational cost (significantly)

code: `https://github.com/ProbabilisticNumerics/probabilistic_line_search`

paper: Mahsereci & Hennig
Probabilistic Line Searches for Stochastic Optimization
Advances in Neural Information Processing Systems (NIPS) 2015

long form: `https://arxiv.org/abs/1703.10034`

Other Examples

probabilistic parameter adaptation

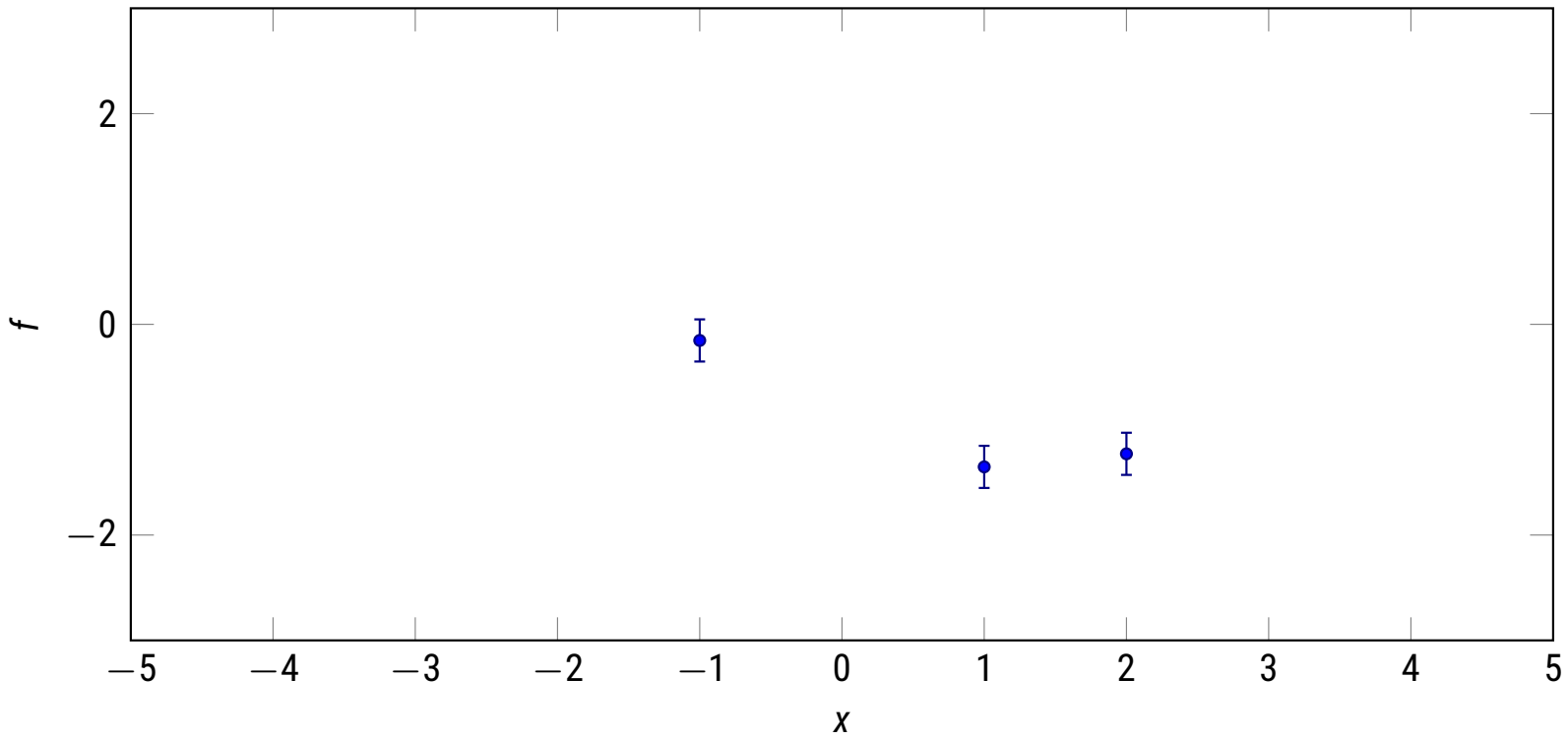
- **batch sizes**
  Balles, Romero, Hennig arXiv:1612.05086
  Coupling Adaptive Batch Sizes with Learning Rates
  `https://github.com/ProbabilisticNumerics/cabs`
- **early stopping**
  Mahsereci, Balles, Lassner, Hennig arXiv:1703.09580
  Early Stopping without a Validation Set
- **search directions**
  Balles & Hennig arXiv:1705.07774
  Follow the Signs for Robust Stochastic Optimization
  `https://github.com/ProbabilisticNumerics/sodas`

- big data gives computational “noise”
- tricky to model in second-order methods
- even first-order **stochastic** methods have free parameters
- “explicit likelihood” can be used to infer parameters

What if the function itself is really valuable?

Bayesian Optimization (BO)

Problem Setting

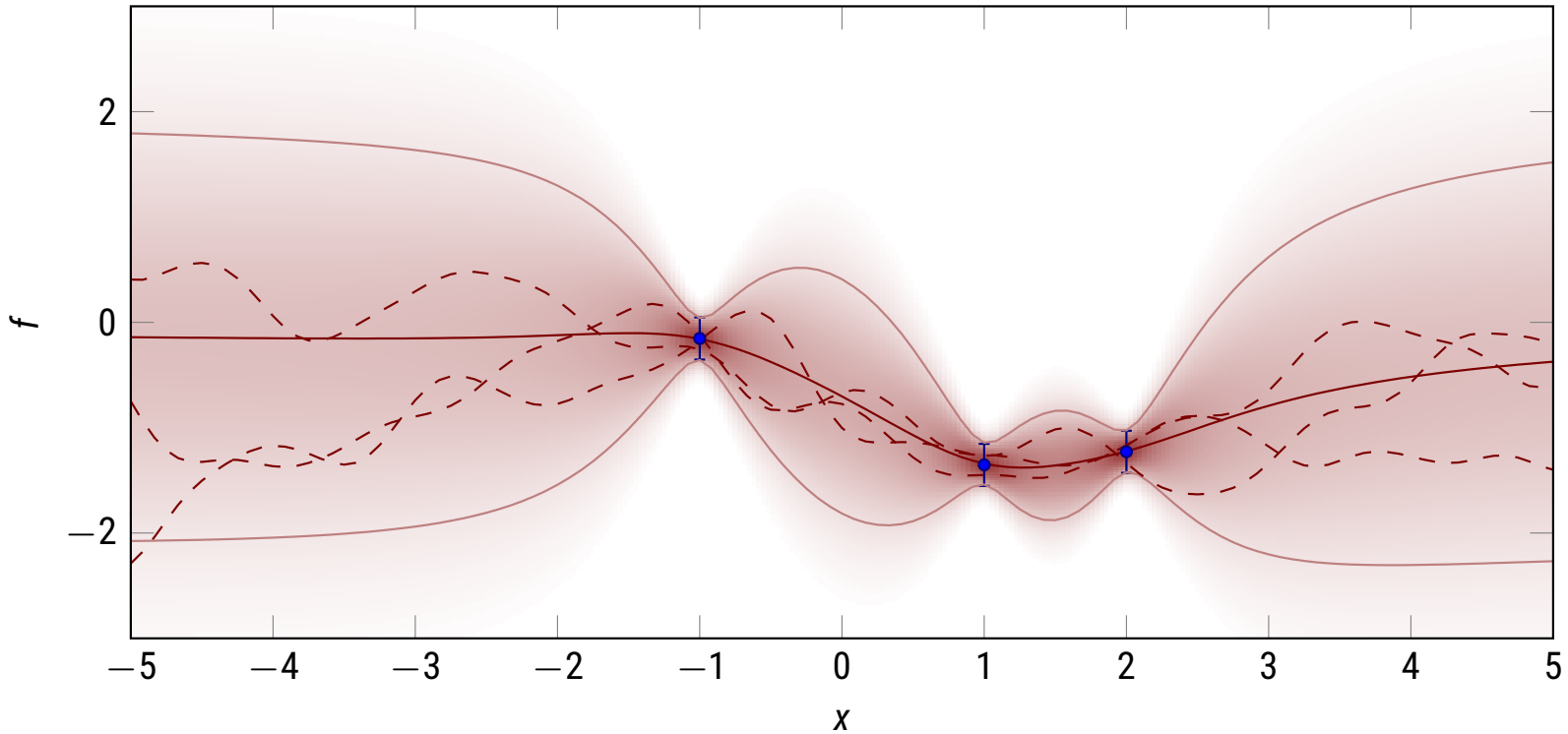


example use cases / motivation

- biomedical / physical experiments
- prototyping a product
- high-level model selection in machine learning

General Recipe

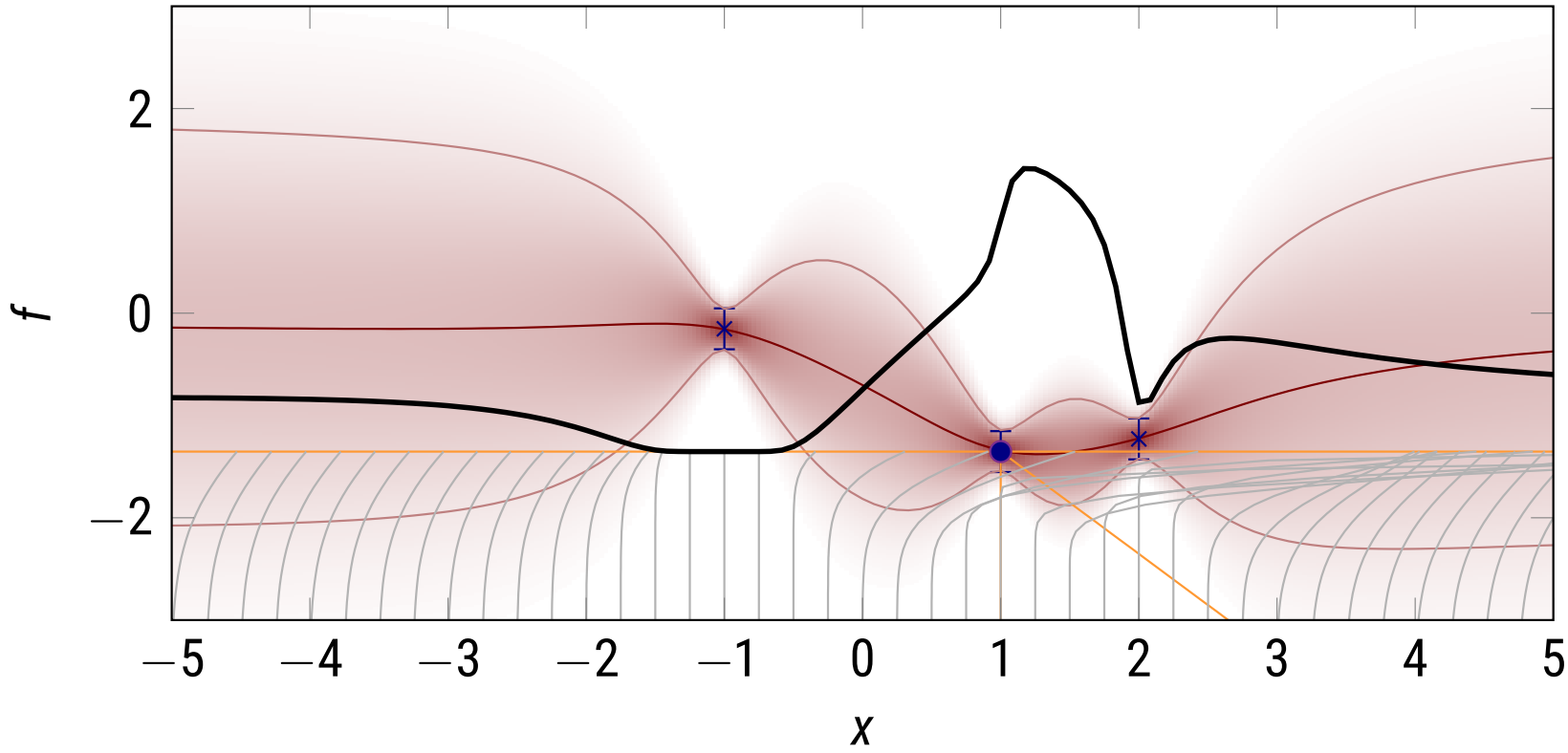
surrogate-based optimization



- assume $f \sim \mathcal{GP}(m, k)fa$

Probability of Improvement

Lizotte, 2008

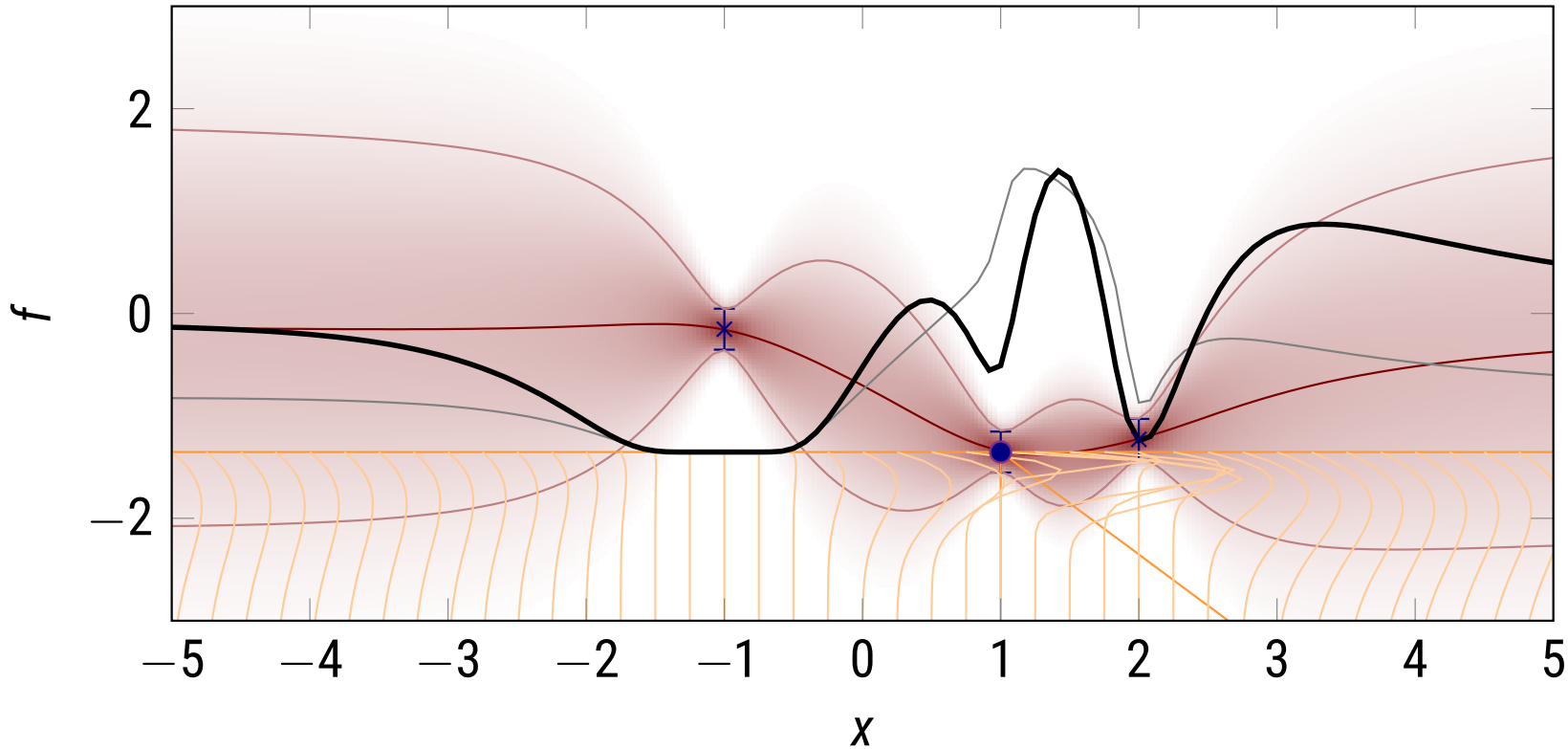


$$u_{\mathrm{PI}}(x) = p[f(x) < \eta] = \int_{-\infty}^{\eta} \mathcal{N}(f(x); \mu(x), \sigma(x)^2) f(x) = \Phi\left(\frac{\eta - \mu(x)}{\sigma(x)}\right)$$

where $\Phi(z) = 1/2[1 + \mathrm{erf}(z/\sqrt{2})]$, $\phi(x) = \mathcal{N}(x; 0, 1)$

Expected Improvement

Jones, Schonlau, Welch, 1998

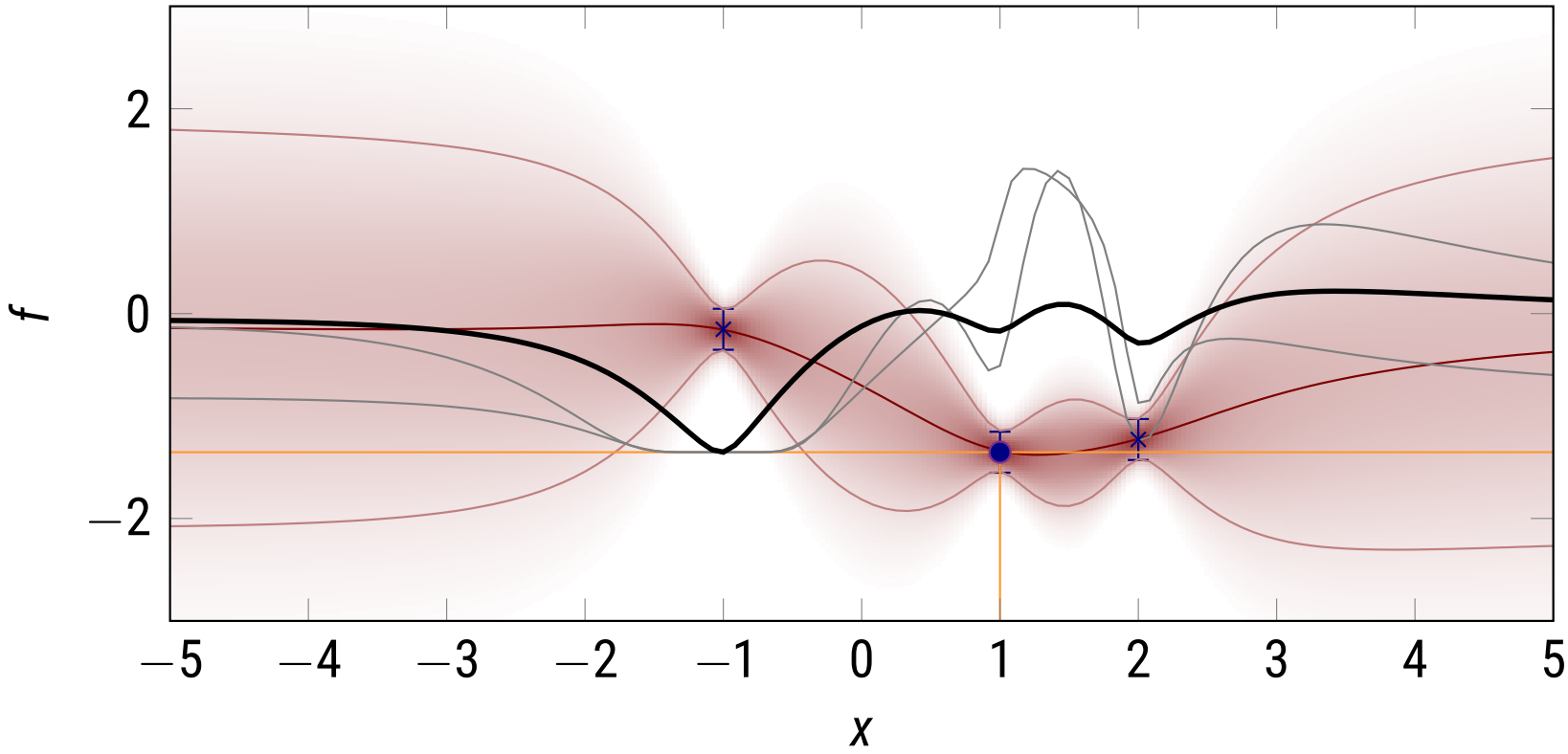


$$u_{\mathrm{EI}}(x) = \mathbb{E}[\min\{0, (\eta - f(x))\}] = (\eta - \mu(x))\Phi\left(\frac{\eta - \mu(x)}{\sigma(x)}\right) + \sigma\phi\left(\frac{\eta - \mu(x)}{\sigma(x)}\right)$$

where $\Phi(z) = 1/2[1 + \mathrm{erf}(z/\sqrt{2})]$, $\phi(x) = \mathcal{N}(x; 0, 1)$

GP Upper Confidence Bound

Srinivas et al., 2009

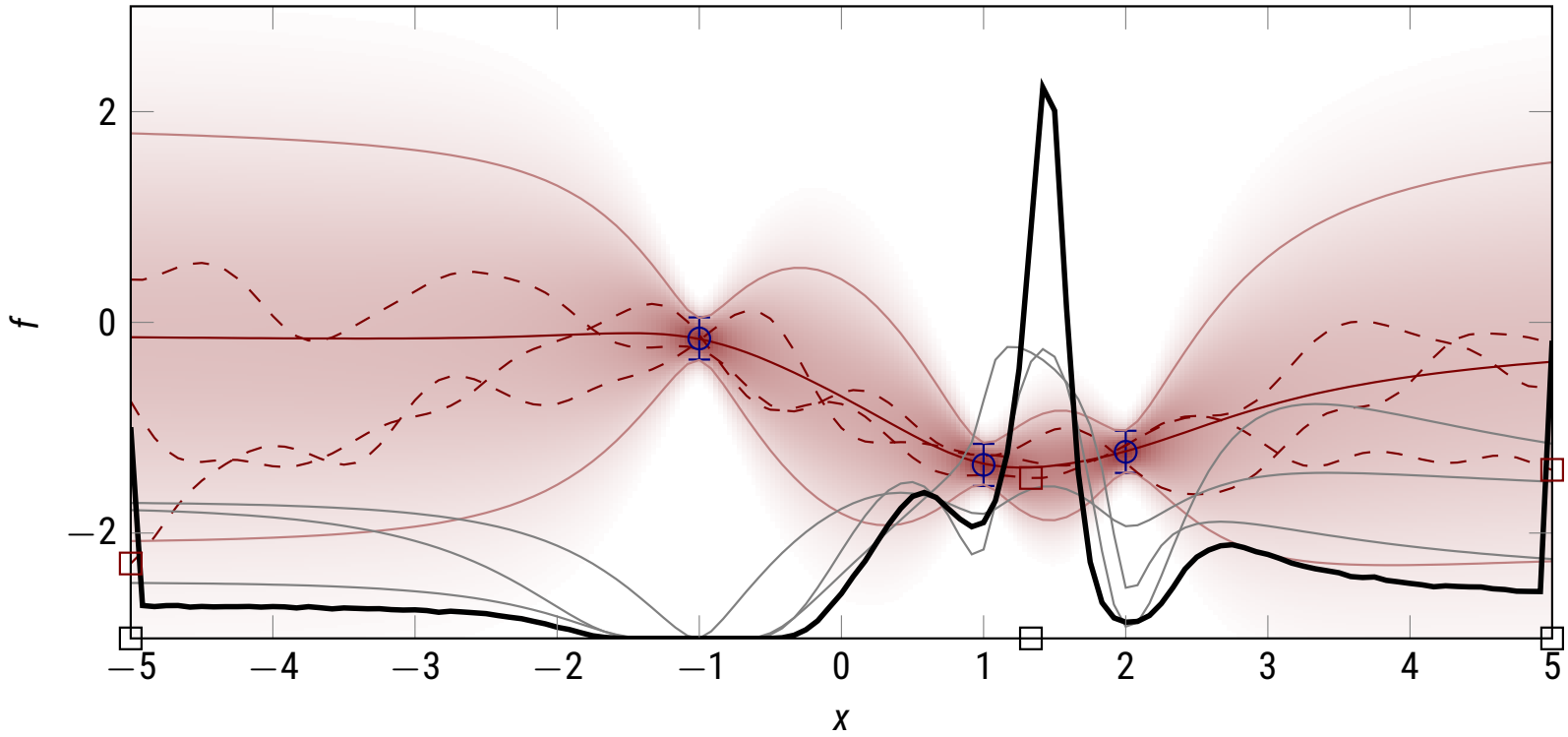


$$u_{\text{GP-UCB}}(x) = -\mu(x) + \sqrt{\beta_i \operatorname{var}(f(x))}$$

where $\beta_i = \frac{2}{5}\log(|D|i^2\pi^2/6\delta$, and $\delta \in [0, 1]$

Beyond Regret: Thompson Sampling

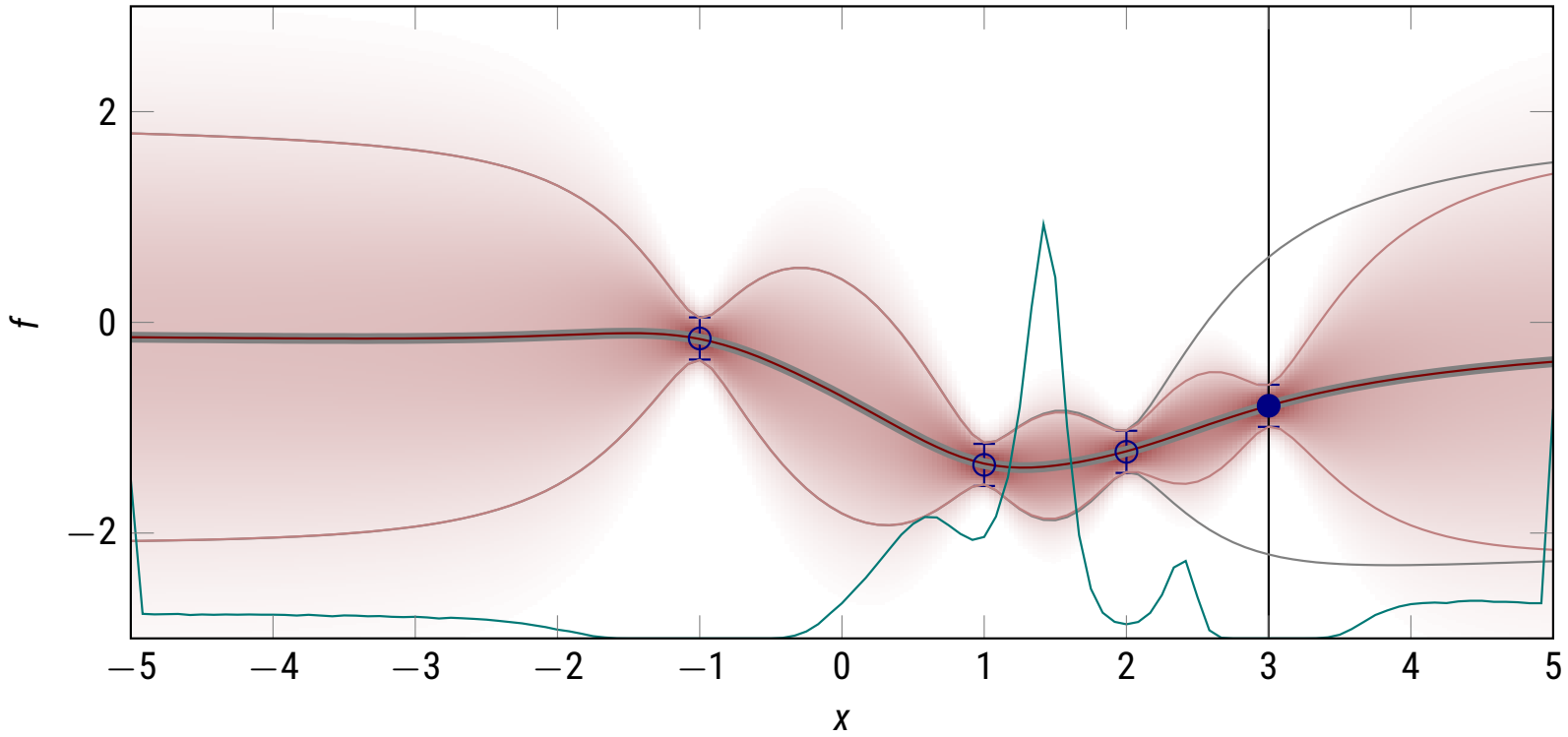
Thompson, 1933



- the distribution of the minium can be found by sampling and building a histogram, although that's obviously not the most efficient way
- the minimum of a single sample can also be used as an evaluation point — **Thompson sampling**

Beyond Regret: Entropy Search

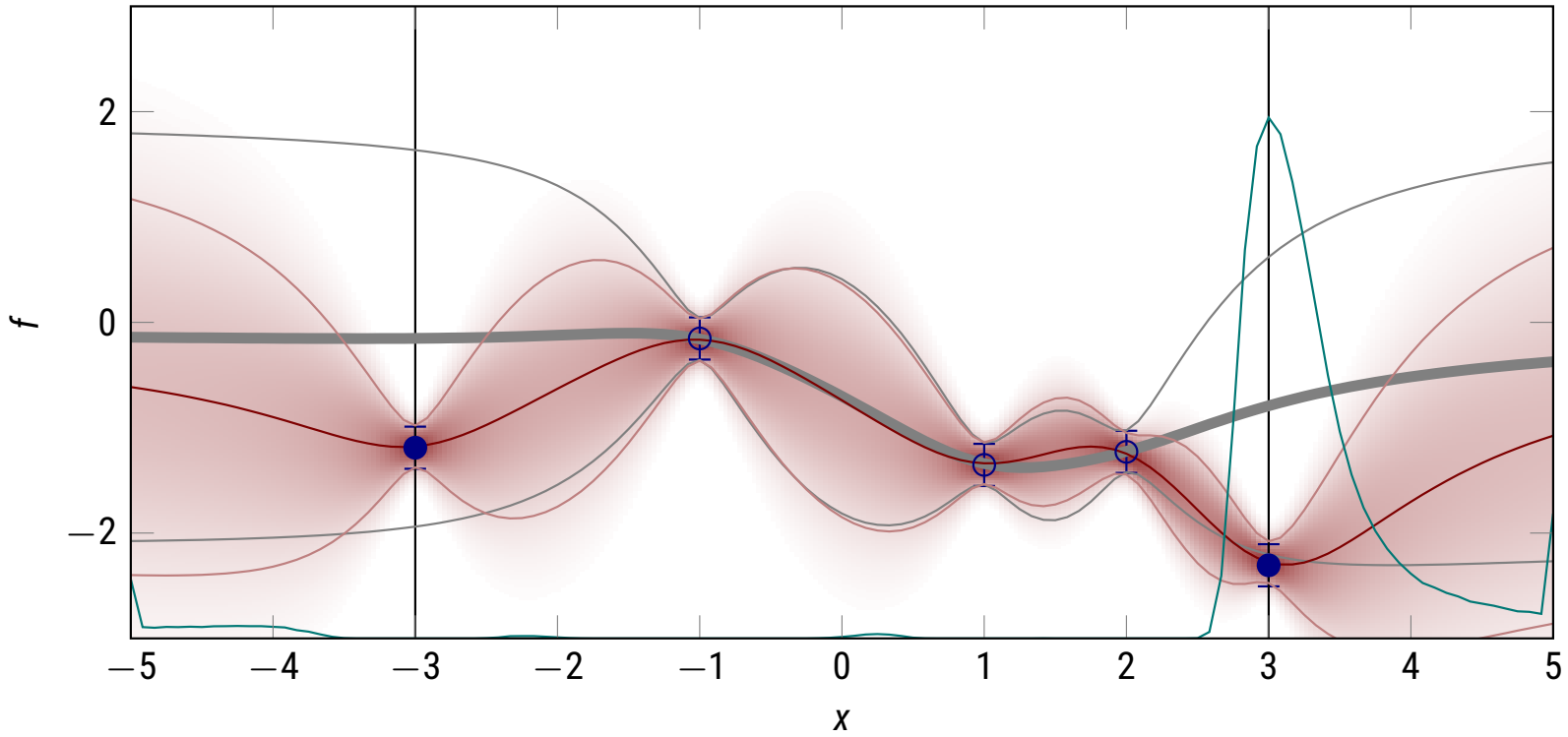
Hennig & Schuler, 2012



- the expected change in the distribution $p_{\min}(x)$ of the minimum from one (and two) evaluation(s)

Beyond Regret: Entropy Search

Hennig & Schuler, 2012

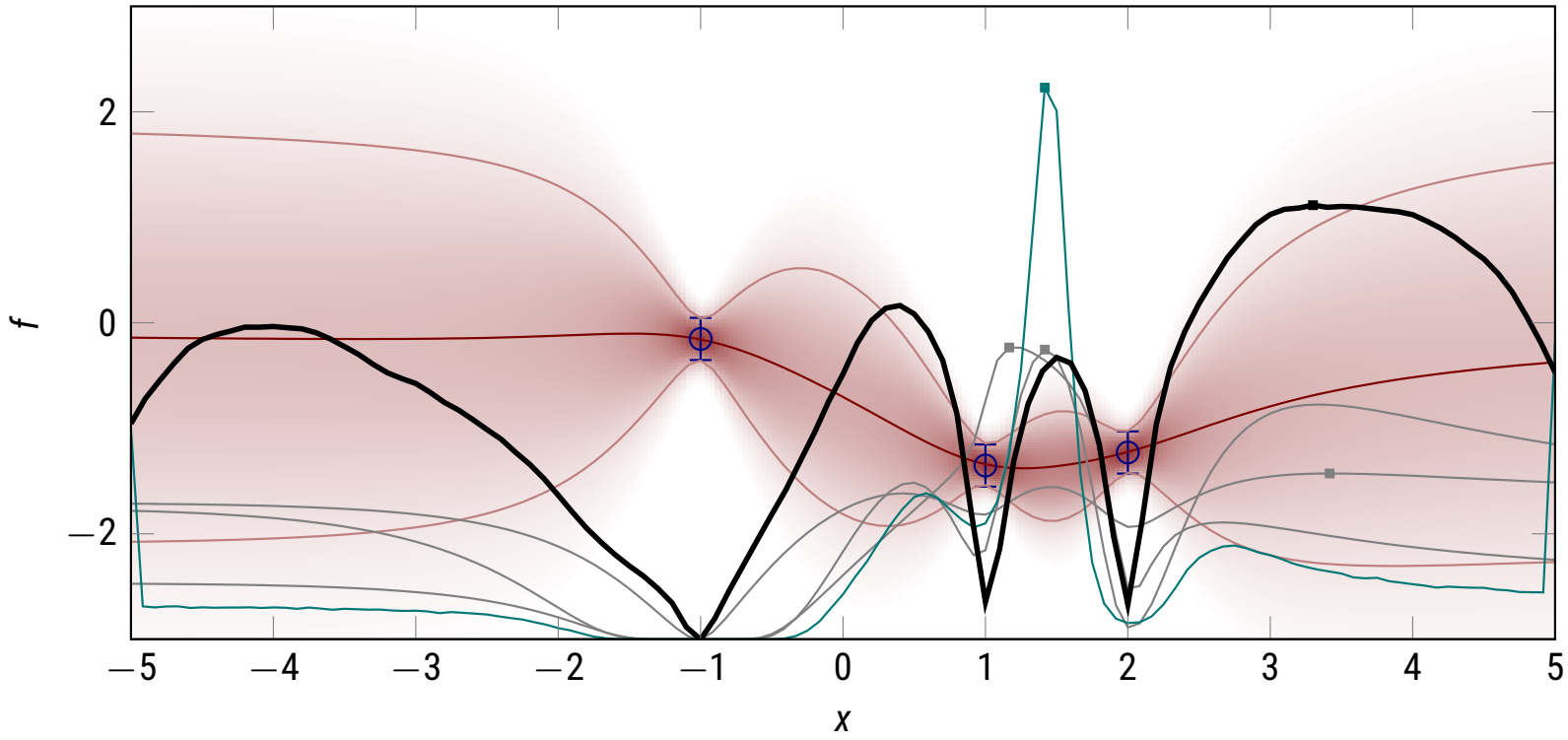


- the expected change in the distribution $p_{\min}(x)$ of the minimum from one (and two) evaluation(s)

Entropy Search utility

expected information gain about location of minimum

Hennig & Schuler, 2012



- expected change in Entropy of $p_{\min}(x)$ from one evaluation at location x_i

Entropy Search

aiming to shape $p_{\min}$

Hennig & Schuler, JMLR 2012, Villemonteix et al., 2009]

- choose utility $\mathcal{L}[p_{\min}]$
- evaluation strategy: maximize expected gain in utility

$$X_{\text{opt}} = \arg\max_X \int \mathcal{L}[p_{\min|(Y,X)}] p(Y \mid X)\, dY$$

- e.g. relative entropy (collect information)

$$\mathcal{L}[p_{\min}] = -\int p_{\min}(x) \frac{\log p(x)}{\log u(x)}\, dx = \left\langle \frac{\log p(x)}{\log u(x)} \right\rangle_{p_{\min}}$$

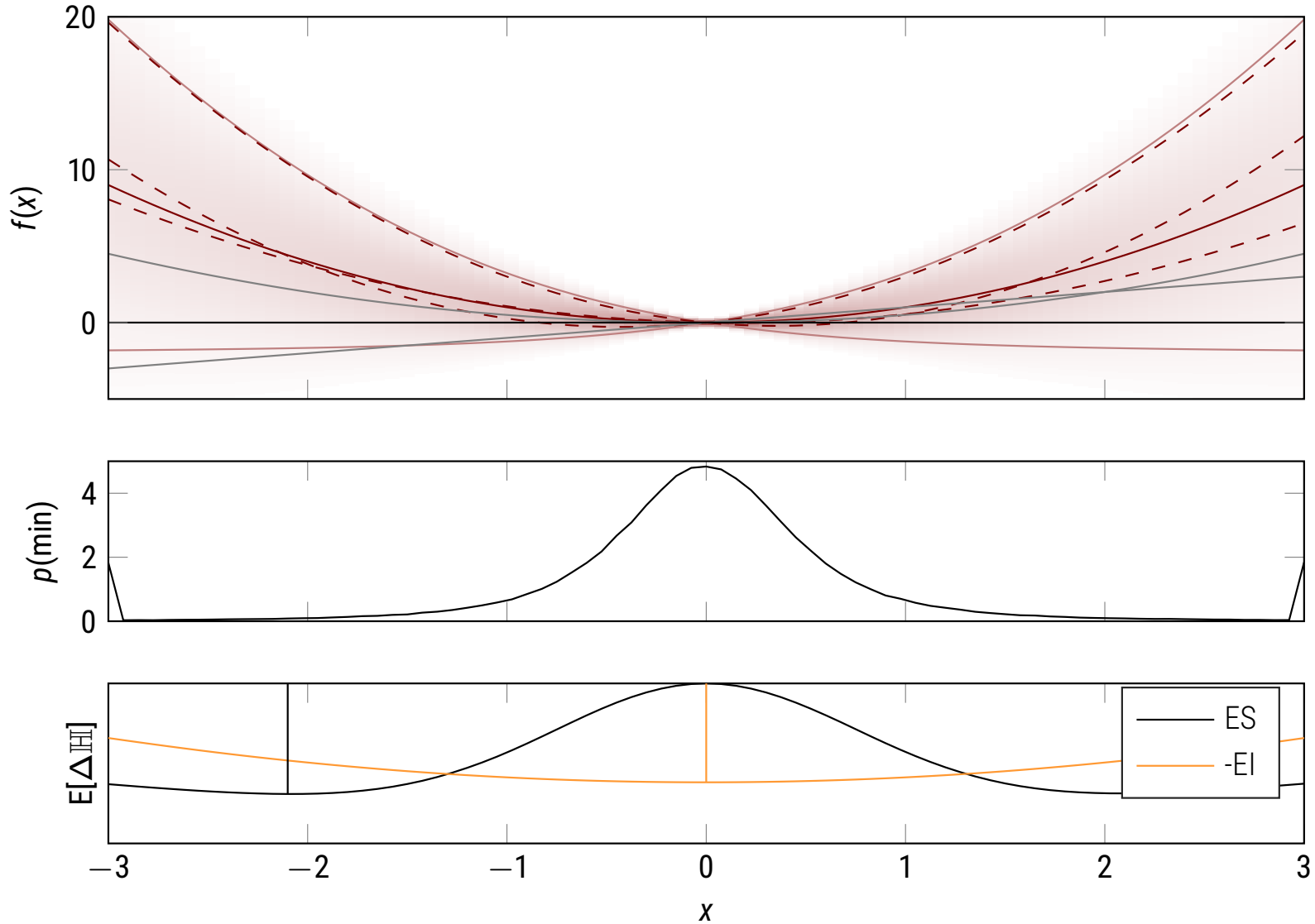
- e.g. square distance (localize minimium)

$$\mathcal{L}[p_{\min}] = \langle x x^\intercal \rangle_{p_{\min}} - \langle x \rangle_{p_{\min}} \langle x^\intercal \rangle_{p_{\min}}$$

collecting small numbers, or learning about the minimum?

informative evaluations need not have low value

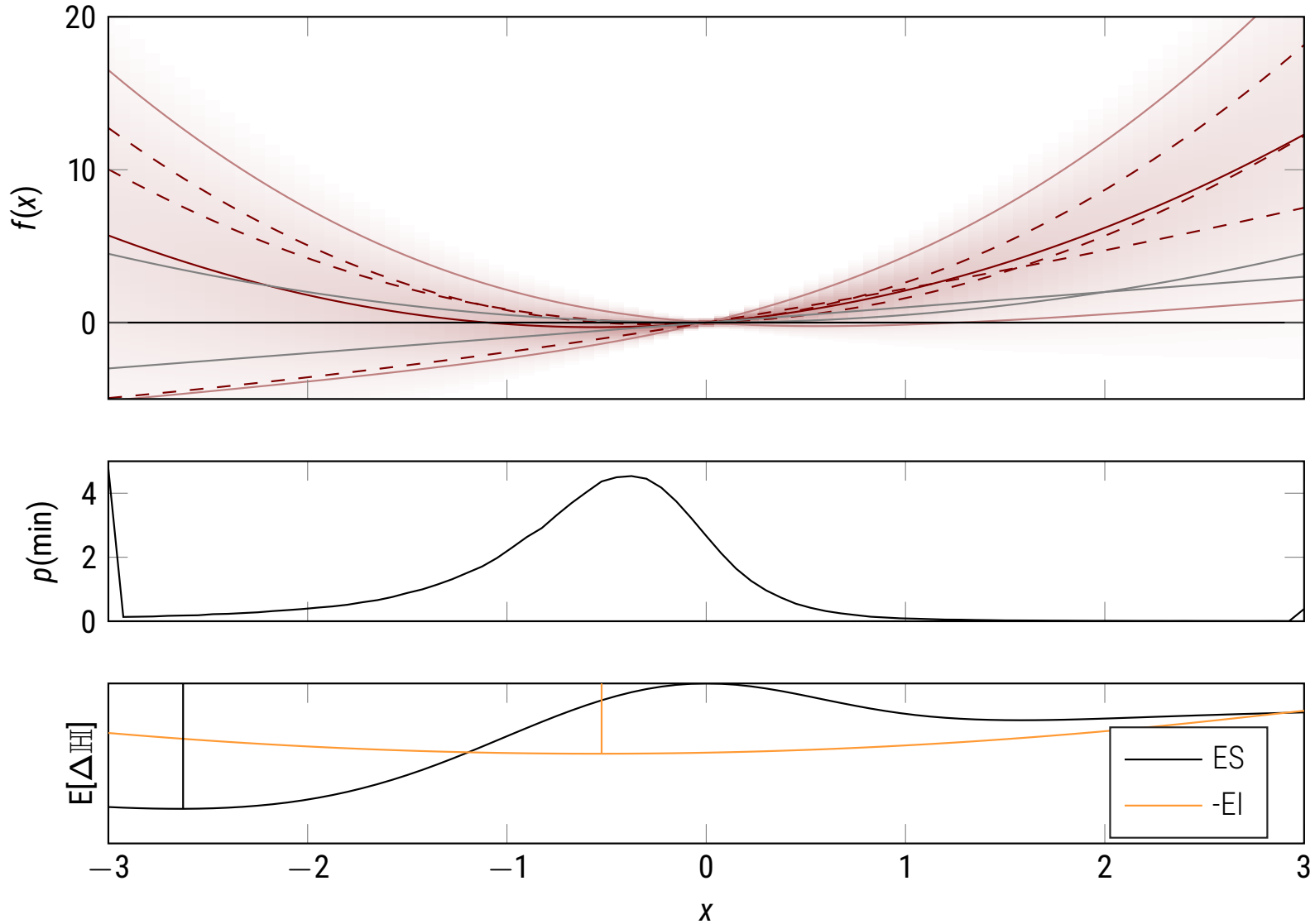
Hennig & Schuler, JMLR 2012]



collecting small numbers, or learning about the minimum?

informative evaluations need not have low value

Hennig & Schuler, JMLR 2012]



ES poses a much more challenging computation

Entropy search explicitly reasons about information

- $p_{\min}(x)$ is nonparametric over x

$$p_{\min}(x) = p[x = \arg\min f(x)] = \int_f p(f) \prod_{\tilde{x} \neq x} \mathbb{I}[f(\tilde{x}) > f(x)]\, df$$

- even on finite domain,

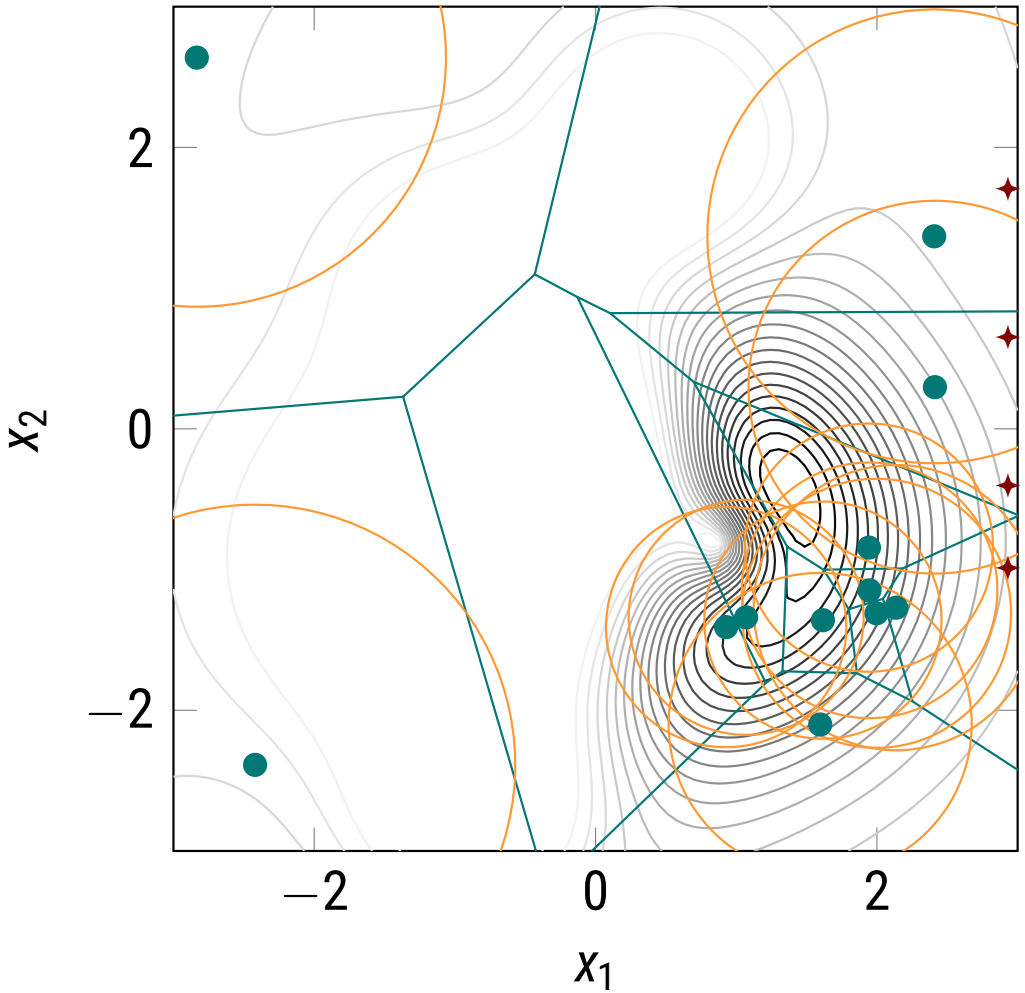
$$p_{\min}(x_i) = \int_f \prod_{j \neq i} \mathbb{I}[(f(x_i) < f(x_j)]\, df$$

is non-analyic if $p(f) = \mathcal{GP}$

Non-uniform grids address curse of dimensionality

drawn from standard utility

Hennig & Schuler, JMLR 2012



draw N representer points $\{x_i\}$ from unnormalised measure $\tilde{u}(x)$

draws represent cells of expected width $N\tilde{u}(x_i)/Z_u$

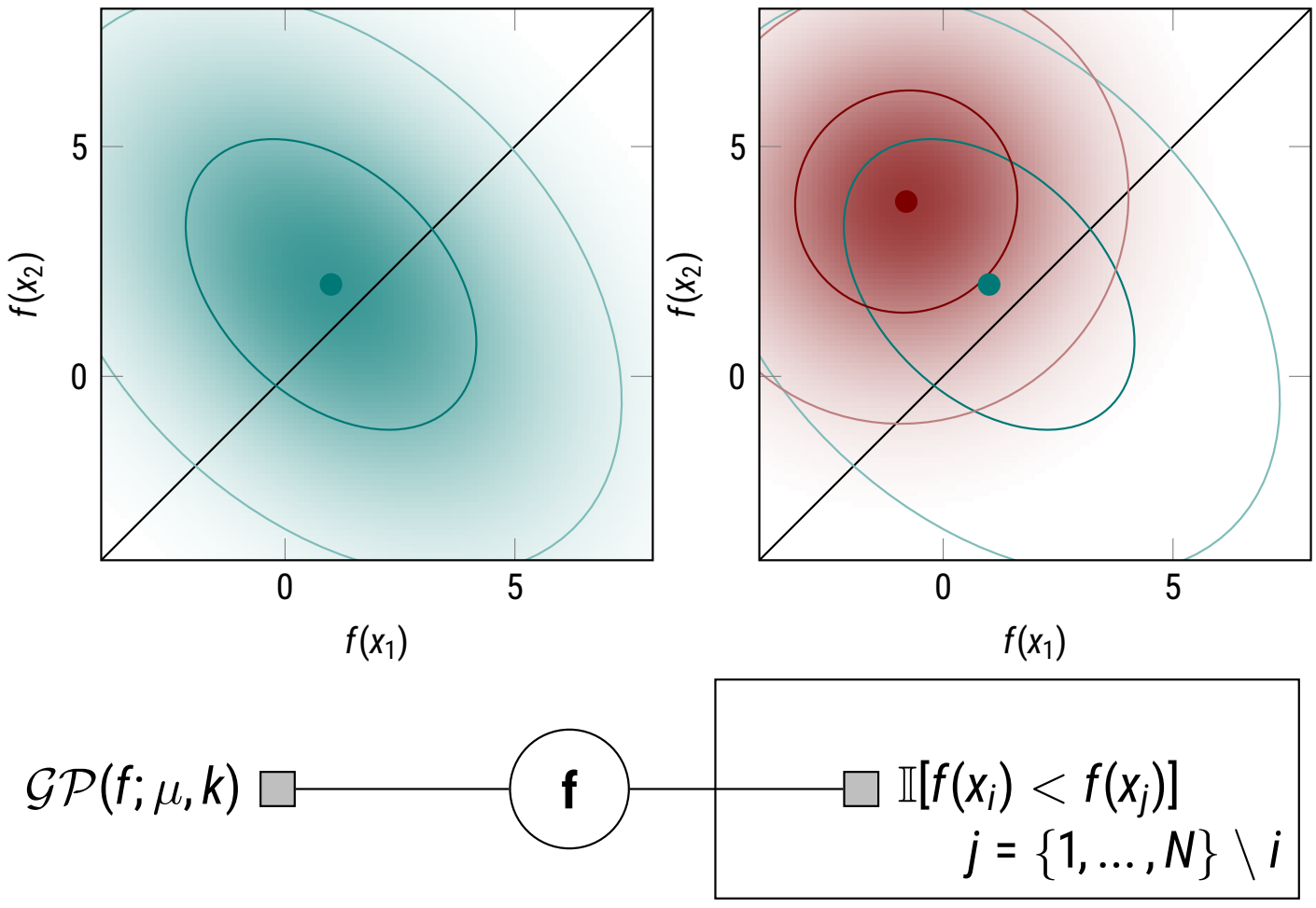
compute $\hat{p}_{\min}(x_i)$ over $\{x_i\}$

correct for cell width

$$p_{\min}(x) \approx \frac{\hat{p}_{\min}(x_i) N \tilde{u}(x_i)}{Z_u}$$

EP provides approximate inference on fixed grid

at cost $\mathcal{O}(N^4)$ [for detailed study, see Cunningham, Lacoste-Julien & Hennig, arXiv `abs/1111.6832`]



- returns $q(x \mid \mu, \Sigma) \approx p_{\min}(x \mid \mu, \Sigma)$, also $\frac{\partial q}{\partial \mu}$, $\frac{\partial q}{\partial \Sigma}$

The Algorithm

Entropy Search

procedure EntropySearch($k, l = p(y \mid f(x)), u, H, (\mathbf{x}, \mathbf{y})$)
  for $h = 1, \ldots, H$ **do**
    $\tilde{\mathbf{x}} \sim u(\mathbf{x}, \mathbf{y})$ // discretize using measure u
    $[\boldsymbol{\mu}, \boldsymbol{\Sigma}, \Delta\boldsymbol{\mu}_x, \Delta\boldsymbol{\Sigma}_x] \leftarrow \mathrm{GP}(k, l, \mathbf{x}, \mathbf{y})$ // infer function, innovation
    $[\hat{q}_{\min}(\tilde{x}), \frac{\partial \hat{q}_{\min}}{\partial \mu}, \frac{\partial^2 \hat{q}_{\min}}{\partial \mu \partial \mu}, \frac{\partial \hat{q}_{\min}}{\partial \Sigma}] \leftarrow \mathrm{EP}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ // approximate $\hat{p}_{\min}$
    $x' \leftarrow \arg\min \langle \mathcal{L} \rangle_x$ // optimize information gain (over x, not $\tilde{x}$)
    $y' \leftarrow \mathrm{Evaluate}(f(x'))$ // take measurement
    $(\mathbf{x}, \mathbf{y}) \leftarrow (\mathbf{x}, \mathbf{y}) \cup (x', y')$
  end for
  return $q_{\min}$ // At horizon, return belief for final decision
end procedure

Some Example Applications & Functionality

Is it worth the computational overhead?

- pair with cost function, to trade off cost vs. informativity
  - K. Swersky, J. Snoek, R. Adams, NIPS 2013
- same idea can also be used to
  - adaptively switch between **simulation** and **experiment** in robotics
    - Marco Valle et al., IROS 2017
  - adapt experimental parameters during search for ML models
    - Klein et al., AISTATS 2017
- use mixture model to model unknown constraints
  - J. M. Hernández-Lobato et al., ICML 2015

Bayesian Optimization has rapidly become an industrially relevant domain.

Probabilistic inference has multiple roles to play in optimization

- **Analytical insight:** some classic optimizers (notably quasi-Newton) are MAP estimates under Gaussian prior.
- **Adaptivity and Robustness:** parameter adaptation, explicit modelling of stochastic noise
- **Bayesian Optimization:** de-novo construction of optimizers for expensive functions

These slides can be found at
`http://tinyurl.com/Dobbiaco-Hennig-5`

The State of the Academic Debate

What, exactly, should PN be and mean? Two evolving views on PN

CS/ML? building practical algorithms motivated and phrased as acting agents that use **probabilities** to encode, track and manage uncertainty.

- the resulting point estimates should have **good classic properties** (convergence order, stability, etc.), but have to be consistent with the probabilistic interpretation. Classic methods are the foundation to build on. Priors have to be motivated, analysed, constructed
- computational cost, practical useability are paramount considerations. The quality of the uncertainty is subordinate. Gaussian / Dirichlet / exponential family measures, and approximate observation models, to yield tractable algorithms

Stats? analysing numerical computation from a **Bayesian** statistical perspective

- the entire computational pipeline has to be an exact application of Bayes' rule. Doing so is intractable, so use approximate computations **of** Bayes rule, which may still be costly. Practical considerations are subordinate to Bayesian calibration.
- Given the cost, classic properties are pointless aims. Priors are subjective, and can not be questioned. Classic methods may be used as black boxes, not as inference machines of their own right.

compute with approximate Bayesian inference $\leftrightarrow$ approximate Bayesian inference with computations

Some Questions for the Hike

in case you want to philosophize

- What is a **random number**? Do random numbers exist? Should the have a place in numerical computation? Why?
- Consider an algorithm that uses computational resources to build a "posterior" probability distribution. Given increasingly more computations, should the posterior become more **concentrated**, or more **structured**?