Probabilistic Numerics
– Part II –
Linear Algebra and Nonlinear Optimization

Philipp Hennig

MLSS 2015
20 / 07 / 2015

MAX-PLANCK-GESELLSCHAFT

Emmy Noether Group on Probabilistic Numerics
Department of Empirical Inference
Max Planck Institute for Intelligent Systems
Tübingen, Germany

Emmy Noether-Programm
DFG Deutsche Forschungsgemeinschaft

Probabilistic Numerics

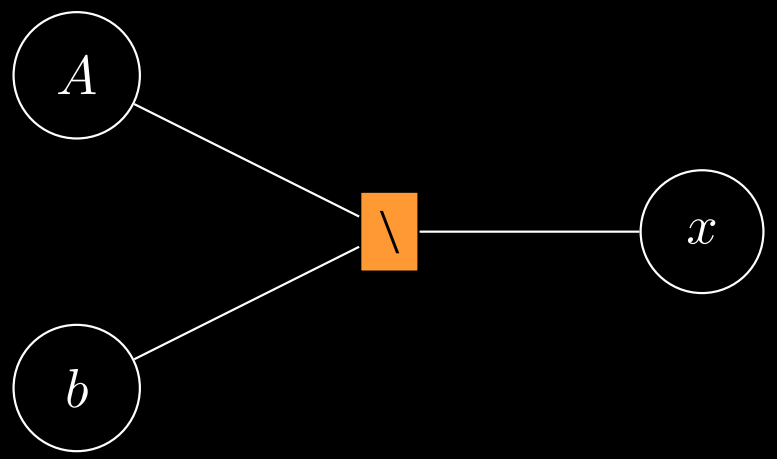
Recap from Saturday

On Saturday

- computation is inference
- classic methods for integration and solution of differential equations can be interpreted as MAP inference from Gaussian models
- customizing the implicit prior gives faster, tailored numerics
- probabilistic formulation allows propagation of uncertainty through composite computations

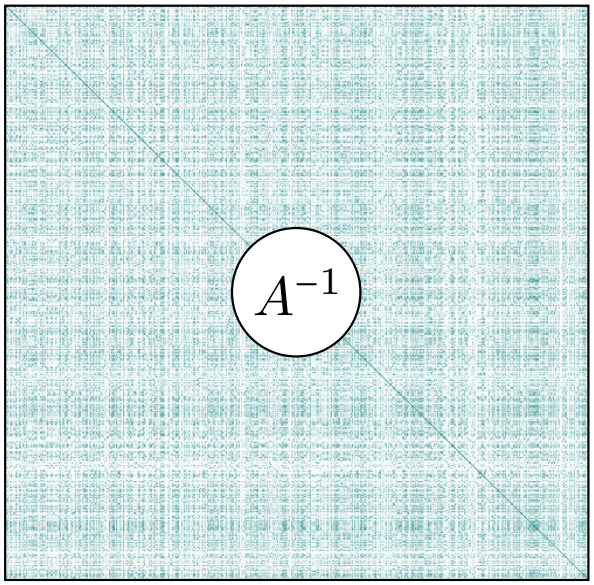
Linear Algebra

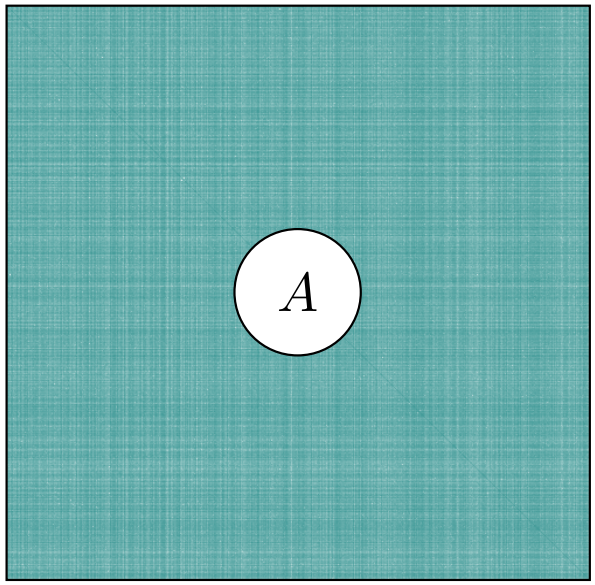
$Ax = b \qquad A \in \mathbb{R}^{N \times N}$ symmetric positive definite



Why you should care about linear algebra

least-squares: a most basic machine learning task





$$\hat{f}(x) = k_{xX}(k_{XX} + \sigma^2 I)^{-1} b = k_{xX} A^{-1} b$$

Inference on Matrix Elements

generic Gaussian priors [Hennig, SIOPT, 2015]

- prior on elements of inverse $H = A^{-1} \in \mathbb{R}^{N\times N}$ with $\Sigma \in \mathbb{R}^{N^2\times N^2}$

$$p(H) = \mathcal{N}(\overrightarrow{H}; \overrightarrow{H_0}, \Sigma) = \frac{1}{(2\pi)^{N^2/2}|\Sigma|^{1/2}} \exp\left[\left(\overrightarrow{H - H_0}\right)^\top \Sigma^{-1} \left(\overrightarrow{H - H_0}\right)\right]$$

- can collect noise-free observations $p(S, Y \mid H) = \delta(S - HY)$

$$AS = Y \qquad \Leftrightarrow \qquad S = HY \in \mathbb{R}^{N\times M}$$

- a linear projection: (using the Kronecker product)

$$\overrightarrow{S}_{km} = \sum_{ij} \delta_{ki} Y_{jm} H_{ij}. \quad \overrightarrow{S} = (I \otimes Y^\top)\overrightarrow{H} = \mathcal{C}\overrightarrow{H} \qquad \mathcal{C} \in \mathbb{R}^{NM\times N^2}$$

- posterior:

$$p(H \mid S, Y) = \mathcal{N}\left[\overrightarrow{H}; \overrightarrow{H_0} + \Sigma\mathcal{C}^\top(\mathcal{C}\Sigma\mathcal{C}^\top)^{-1}(\overrightarrow{S - \mathcal{C}H_0}), \Sigma - \Sigma\mathcal{C}^\top(\mathcal{C}\Sigma\mathcal{C}^\top)^{-1}\mathcal{C}\Sigma\right]$$

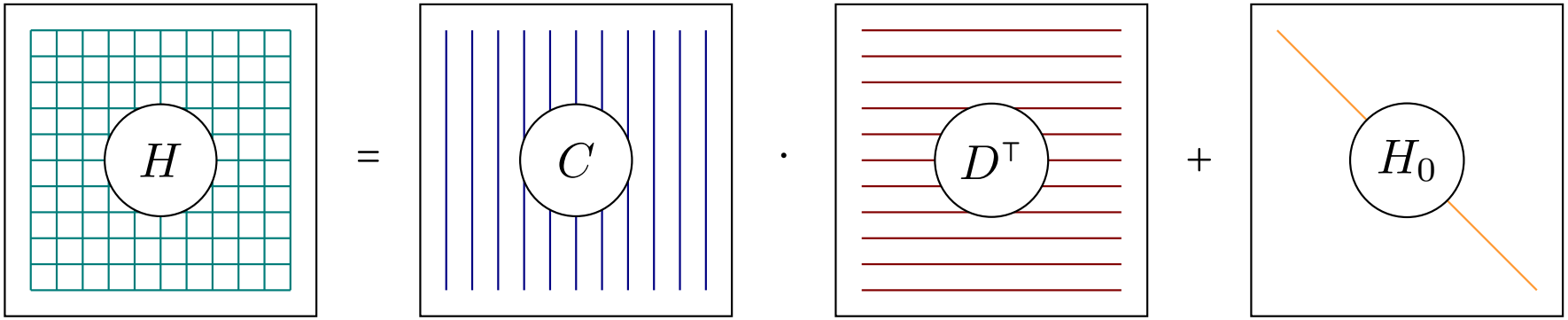
- requires $\mathcal{O}(N^3 M)$ operations! Need structure in Σ

$$p(H \mid S, Y) = \mathcal{N}\left[\overrightarrow{H}; \overrightarrow{H_0} + \Sigma\mathcal{C}^\top(\mathcal{C}\Sigma\mathcal{C}^\top)^{-1}(\overrightarrow{S - \mathcal{C}H_0}), \Sigma - \Sigma\mathcal{C}^\top(\mathcal{C}\Sigma\mathcal{C}^\top)^{-1}\mathcal{C}\Sigma\right]$$

- good probabilistic numerical methods must have both
  - low computational cost
  - meaningful prior assumptions

A factorization assumption

with support on all matrices



- $\mathrm{cov}(H_{ij}, H_{k\ell}) = V_{ik} W_{j\ell}$

$$\Rightarrow \qquad p(H) = \mathcal{N}(H; H_0, V \otimes W)$$

- if $V, W \succ 0$, this puts nonzero mass on *all* $H \in \mathbb{R}^{N \times N}$

$$\mathrm{var}(H_{ij}) = V_{ii} W_{jj}$$

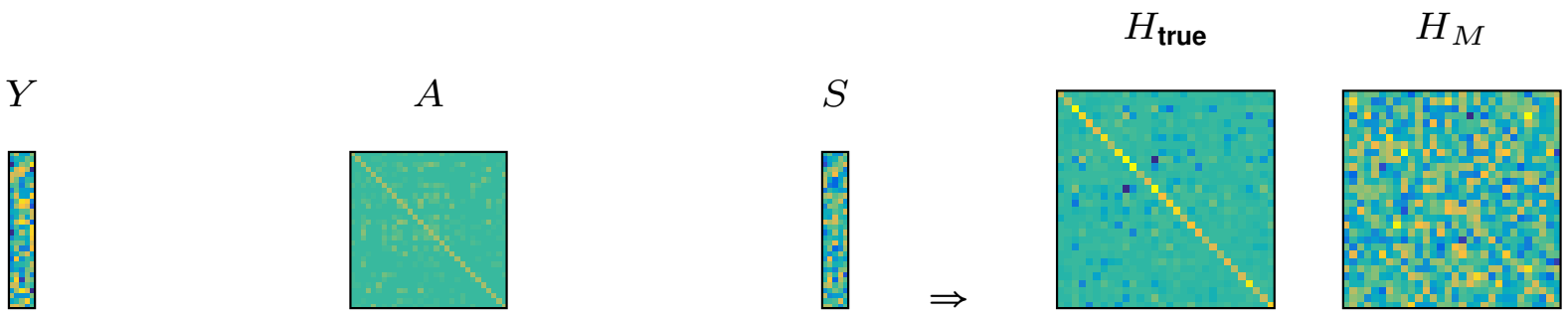
- draw n columns of C iid. from $\mathcal{N}(C_{:i}; 0, V/n)$
- draw n columns of D iid. from $\mathcal{N}(D_{:i}; 0, W/n)$

A Structured Prior

computation requires trading expressivity and cost [Hennig, SIOPT, 2015]

- prior $p(H) = \mathcal{N}(\overrightarrow{H}; \overrightarrow{H_0}, V \otimes W)$ gives

$$p(H \mid S, Y) = \mathcal{N}\Big[H; H_0 + (S - H_0 Y)(Y^\top W Y)^{-1} Y^\top W, \\ V \otimes (W - WY(Y^\top W Y)^{-1} Y^\top W)\Big]$$

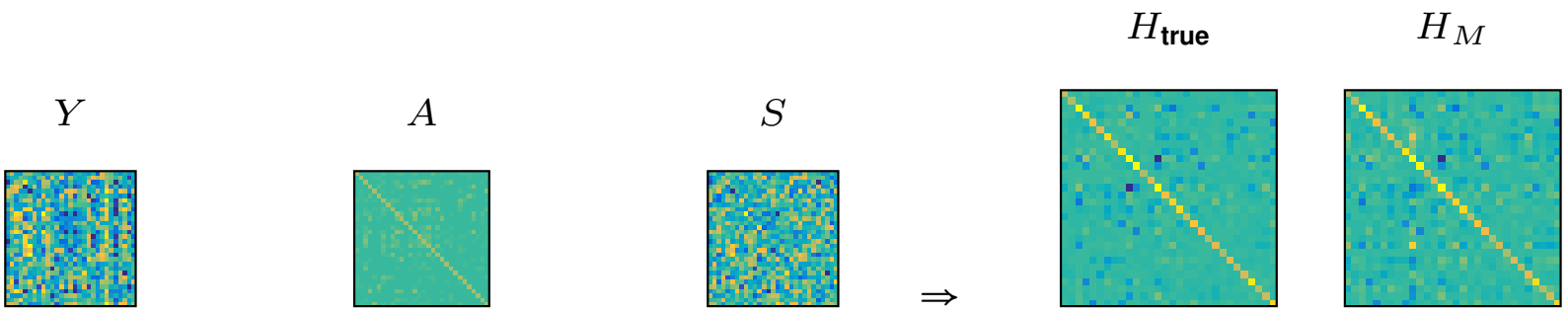


A Structured Prior

computation requires trading expressivity and cost [Hennig, SIOPT, 2015]

- prior $p(H) = \mathcal{N}(\overrightarrow{H}; \overrightarrow{H_0}, V \otimes W)$ gives

$$p(H \mid S, Y) = \mathcal{N}\Big[H; H_0 + (S - H_0 Y)(Y^\top W Y)^{-1} Y^\top W, \\ V \otimes (W - W Y (Y^\top W Y)^{-1} Y^\top W)\Big]$$

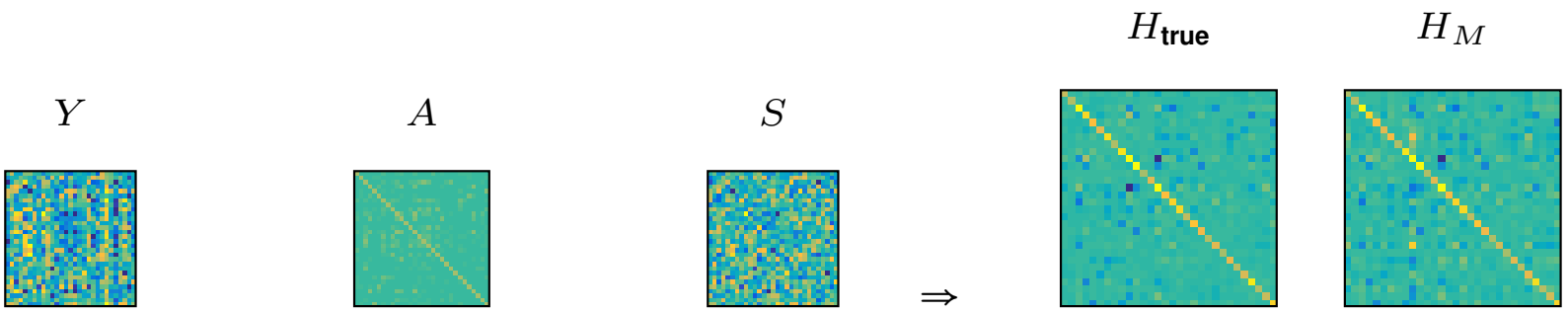


A Structured Prior

computation requires trading expressivity and cost [Hennig, SIOPT, 2015]

- prior $p(H) = \mathcal{N}(\overrightarrow{H}; \overrightarrow{H_0}, V \otimes W)$ gives

$$p(H \mid S, Y) = \mathcal{N}\Big[H; H_0 + (S - H_0 Y)(Y^\top W Y)^{-1} Y^\top W, \\ V \otimes (W - WY(Y^\top W Y)^{-1} Y^\top W)\Big]$$



- two problems:
  - still requires $\mathcal{O}(M^3)$ inversion just to compute mean
  - $\rightsquigarrow$ would like diagonal $Y^\top W Y$ (conjugate observations)
  - how to choose H_0, V, W to get well-scaled prior?
  - $\rightsquigarrow$ 'empirical Bayesian' choice to include H

A Scaled Prior

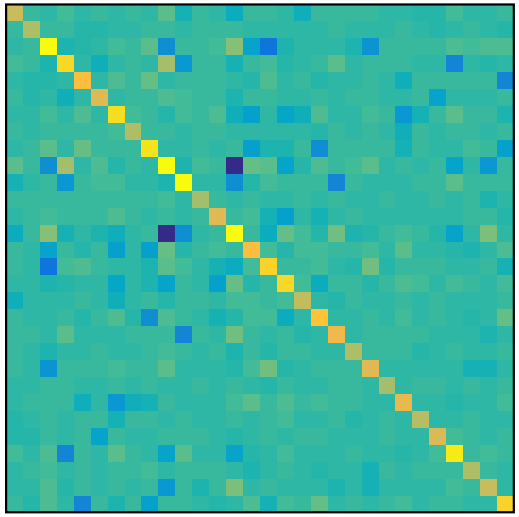
probabilistic computation needs meaningful priors [Hennig, SIOPT, 2015]

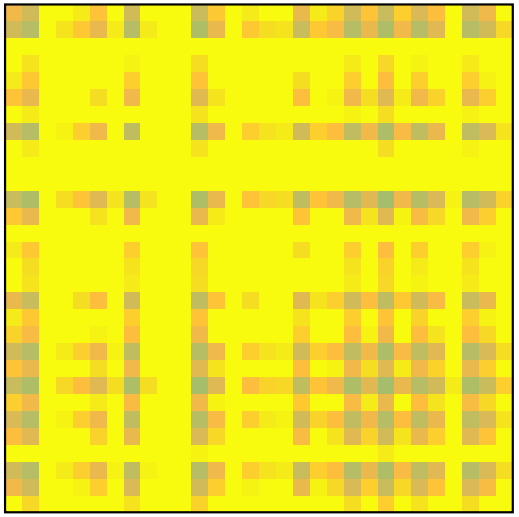
- using $H_0 = \epsilon I$ with $\epsilon \ll 1$. It would be nice to have $W = V = H$:

$$\mathrm{var}(H)_{ij} = V_{ii} W_{jj} = H_{ii} H_{jj}$$

  for symmetric positive definite matrices, $H_{ii} > 0$, $H_{ij}^2 \leq H_{ii} H_{jj}$
- if $W = V = H$,

$$p(H \mid S, Y) = \mathcal{N}\Big[H; H_0 + (S - H_0 Y)(Y^\top W Y)^{-1} Y^\top W, \\ V \otimes (W - WY(Y^\top W Y)^{-1} Y^\top W)\Big]$$





A Scaled Prior

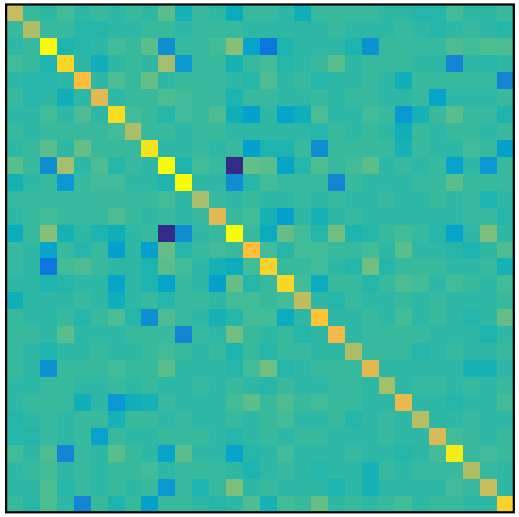
probabilistic computation needs meaningful priors [Hennig, SIOPT, 2015]

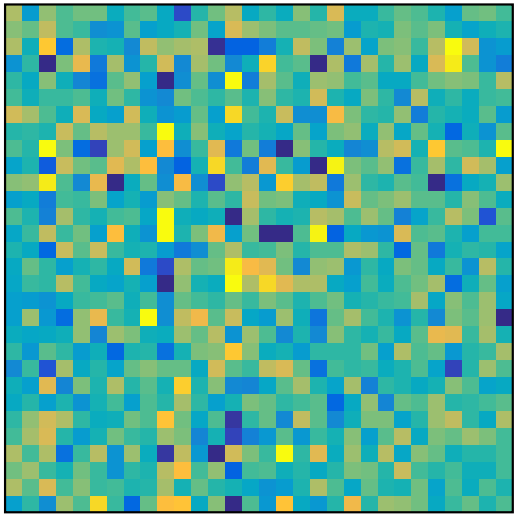
- using $H_0 = \epsilon I$ with $\epsilon \ll 1$. It would be nice to have $W = V = H$:

$$\mathrm{var}(H)_{ij} = V_{ii} W_{jj} = H_{ii} H_{jj}$$

  for symmetric positive definite matrices, $H_{ii} > 0$, $H_{ij}^2 \leq H_{ii} H_{jj}$
- if $W = V = H$,

$$p(H \mid S, Y) = \mathcal{N}\Big[H; H_0 + (S - H_0 Y)(Y^\top S)^{-1} S^\top, \\ W \otimes (W - S(Y^\top S)^{-1} S^\top)\Big]$$





A Scaled Prior

probabilistic computation needs meaningful priors [Hennig, SIOPT, 2015]

- using $H_0 = \epsilon I$ with $\epsilon \ll 1$. It would be nice to have $W = V = H$:

$$\mathrm{var}(H)_{ij} = V_{ii} W_{jj} = H_{ii} H_{jj}$$

  for symmetric positive definite matrices, $H_{ii} > 0$, $H_{ij}^2 \leq H_{ii} H_{jj}$
- if $W = V = H$,

$$p(H \mid S, Y) = \mathcal{N}\Big[H; H_0 + (S - H_0 Y)(Y^\top S)^{-1} S^\top, \\ W \otimes (W - S(Y^\top S)^{-1} S^\top)\Big]$$

- can choose conjugate directions $S^\top A S = S^\top Y = \mathrm{diag}_i\{g_i\}$ using Gram-Schmidt process. Choose orthogonal set $\{u_1, \ldots, u_N\}$

$$s_i = u_i - \sum_{j=1}^{i-1} \frac{y_j^\top u_i}{y_j^\top s_j} s_j$$

  then

$$\mathsf{E}_{\mid S,Y}[H] = H_0 + \sum_{i=1}^{M} \frac{(s_m - H_0 y_m) s_m^\top}{y_m^\top s_m}$$

Active Learning of Matrix Inverses

Gaussian Elimination [C.F. Gauss, 1809]

which set of orthogonal directions should we choose?

- e.g. $\{u_1, \ldots, u_N\} = \{e_1, \ldots, e_N\}$

$H_{\textbf{true}}$ $|S|$ $|Y|$ $p(H)$ $|A \cdot H_M|$

Active Learning of Matrix Inverses

Gaussian Elimination

[C.F. Gauss, 1809]

which set of orthogonal directions should we choose?

- e.g. $\{u_1, \ldots, u_N\} = \{e_1, \ldots, e_N\}$

$H_{\textbf{true}}$ $|S|$ $|Y|$ $p(H)$ $|A \cdot H_M|$

Active Learning of Matrix Inverses

Gaussian Elimination [C.F. Gauss, 1809]

which set of orthogonal directions should we choose?

- e.g. $\{u_1, \ldots, u_N\} = \{e_1, \ldots, e_N\}$

$H_{\textbf{true}}$ $|S|$ $|Y|$ $p(H)$ $|A \cdot H_M|$

Active Learning of Matrix Inverses

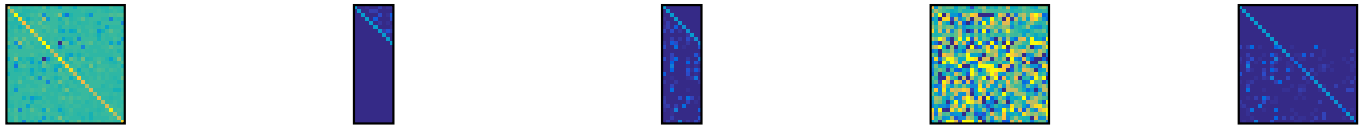
Gaussian Elimination

[C.F. Gauss, 1809]

which set of orthogonal directions should we choose?

- e.g. $\{u_1, \ldots, u_N\} = \{e_1, \ldots, e_N\}$

$H_{\textbf{true}}$ $|S|$ $|Y|$ $p(H)$ $|A \cdot H_M|$



Active Learning of Matrix Inverses

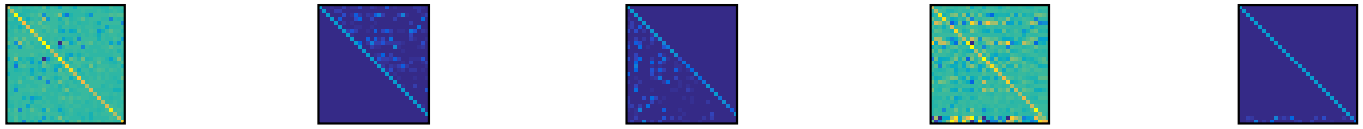
Gaussian Elimination

[C.F. Gauss, 1809]

which set of orthogonal directions should we choose?

- e.g. $\{u_1, \ldots, u_N\} = \{e_1, \ldots, e_N\}$

$H_{\textbf{true}}$ $|S|$ $|Y|$ $p(H)$ $|A \cdot H_M|$



Active Learning of Matrix Inverses

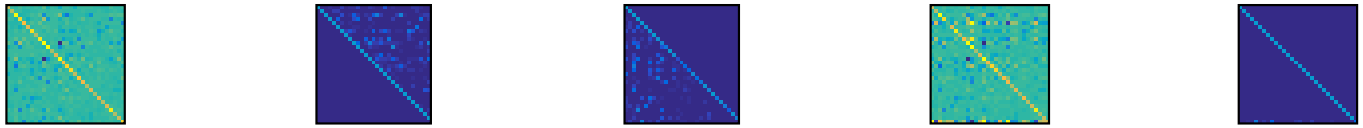
Gaussian Elimination

[C.F. Gauss, 1809]

which set of orthogonal directions should we choose?

- e.g. $\{u_1, \ldots, u_N\} = \{e_1, \ldots, e_N\}$

$H_{\textbf{true}}$ $|S|$ $|Y|$ $p(H)$ $|A \cdot H_M|$

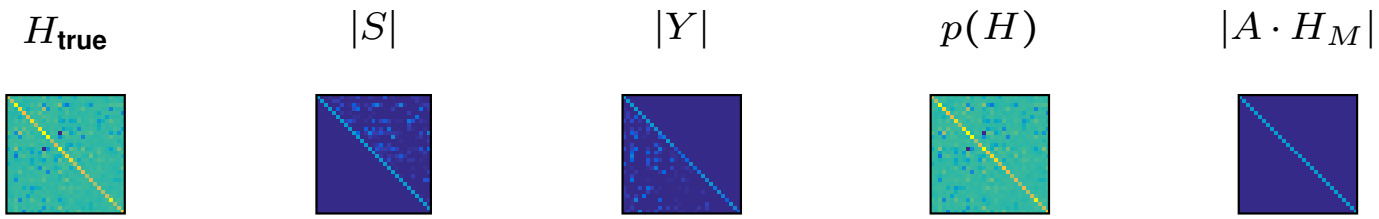


Active Learning of Matrix Inverses

Gaussian Elimination [C.F. Gauss, 1809]

which set of orthogonal directions should we choose?

- e.g. $\{u_1, \ldots, u_N\} = \{e_1, \ldots, e_N\}$



Gaussian eliminiation of A is maximum a-posteriori estimation of H under a well-scaled Gaussian prior, if the search directions are chosen from the unit vectors.

Gaussian elimination as MAP inference:

- decide to use Gaussian prior
- factorization assumption (Kronecker structure) in covariance gives simple update
- implicitly choosing "$W = H$" gives well-scaled prior
- conjugate directions for efficient bookkeeping
- construct projections from unit vectors

What about Uncertainty?

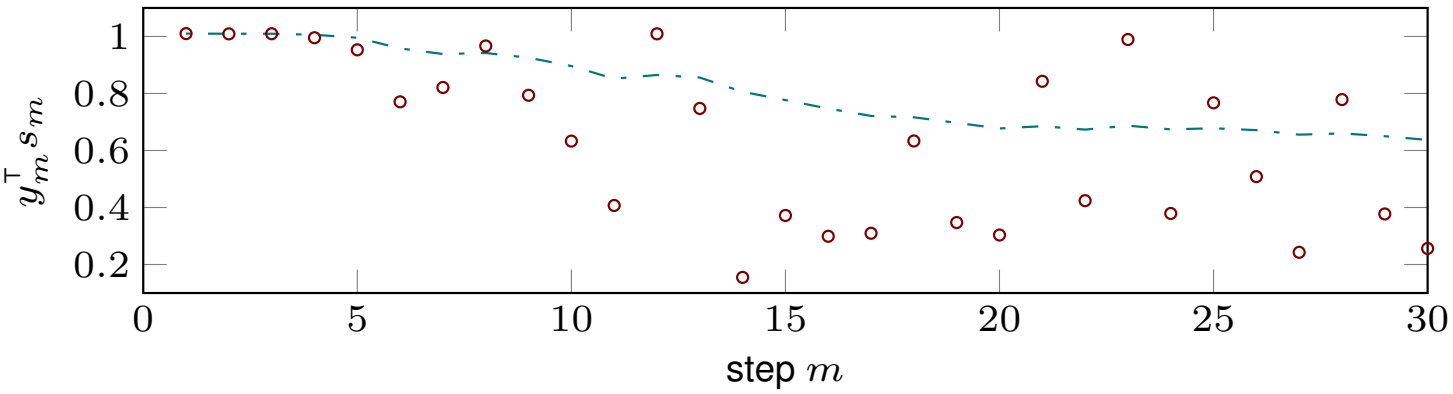
calibrating prior covariance at runtime [Hennig, SIOPT, 2015]

under "$W = H$"

$$p(H \mid S, Y) = \mathcal{N}\left[H; H_0 + (S - H_0 Y)(Y^\top S)^{-1} S^\top, W \otimes (W - S(Y^\top S)^{-1} S^\top)\right]$$

just need $WY = S$. So choose

$$W = S(Y^\top S)^{-1} S^\top + (I - Y(Y^\top Y)^{-1} Y^\top)\Omega(I - Y(Y^\top Y)^{-1} Y^\top)$$



What about Uncertainty?

calibrating prior covariance at runtime [Hennig, SIOPT, 2015]

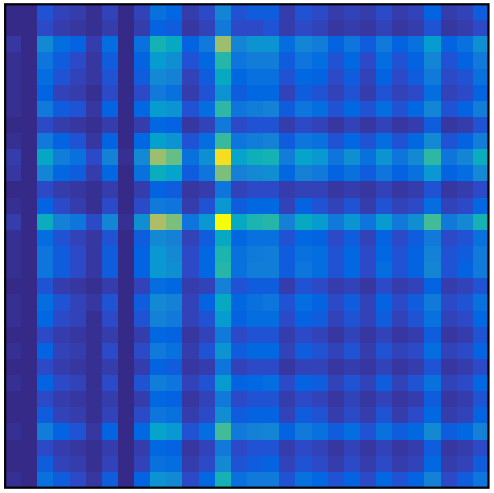
under "$W = H$"

$$p(H \mid S, Y) = \mathcal{N}\left[H; H_0 + (S - H_0 Y)(Y^\top S)^{-1} S^\top, W \otimes (W - S(Y^\top S)^{-1} S^\top)\right]$$

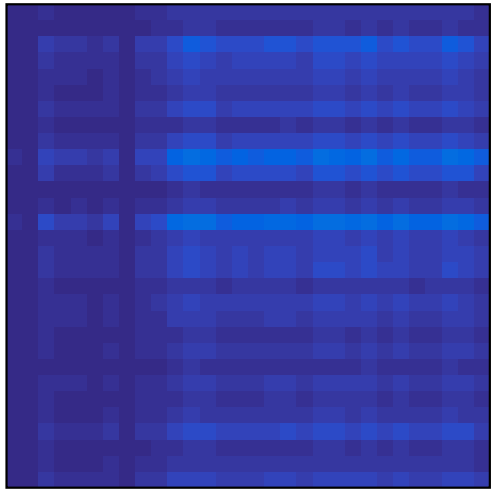
just need $WY = S$. So choose

$$W = S(Y^\top S)^{-1} S^\top + (I - Y(Y^\top Y)^{-1} Y^\top)\Omega(I - Y(Y^\top Y)^{-1} Y^\top)$$

W_M **for** $W_0 = H$



W_M **for** W_0 **estimated**



- scaled, structured prior, exploration along unit vectors gives Gaussian elimination
- empirical Bayesian estimation of covariance gives scaled posterior uncertainty, retains classic estimate, at very low cost overhead

Can we do better than Gaussian Elimination?

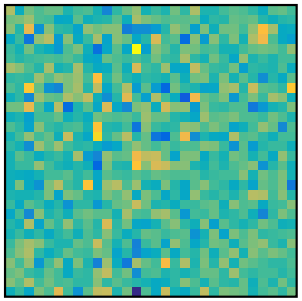
encode symmetry $H = H^\top$ [Hennig, SIOPT, 2015]

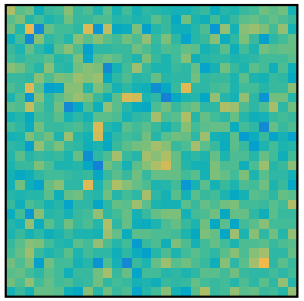
- Using $\Gamma\overrightarrow{H} = {}^1\!/\!_2(\overrightarrow{H + H^\top})$, $p(\text{symm.} \,|\, H) = \lim_{\beta \to 0} \mathcal{N}(0; \Gamma\overrightarrow{H}, \beta)$

$$p(H \,|\, \text{symm.}) = \mathcal{N}(\overrightarrow{H}; \overrightarrow{H}_0, W \otimes W)$$

$$(W \otimes W)_{ij,k\ell} = {}^1\!/\!_2(W_{ik}W_{j\ell} + W_{i\ell}W_{jk})$$

$H \sim \mathcal{N}(H_0, W \otimes W)$ $H \sim \mathcal{N}(H_0, W \otimes W)$





- $p(S, Y \,|\, H) = \delta(S - HY)$ now gives ($\Delta = S - H_0 Y$, $G = Y^\top W Y$)

$$p(H \,|\, S, Y) = \mathcal{N}\Big[H;$$
$$H_0 + \Delta G^{-1} Y^\top W + W Y G^{-1} \Delta^\top - W Y G^{-1} \Delta^\top Y G^{-1} Y^\top W,$$
$$(W - W Y G^{-1} Y^\top W) \otimes (W - W Y G^{-1} Y^\top W)\Big]$$

Active Learning for a Single Linear Problem

choose 'search directions' from gradients

$$Ax = b \qquad \Leftrightarrow \qquad x = \arg\min_{\tilde{x}} f(\tilde{x}) \qquad f(x) = \left[{}^1\!/_2 x^\top A x - x^\top b \right]$$

$$r(x) = \nabla f(x) = Ax - b$$

Algorithm 1 Solve $Ax = b$ under $p(H \mid H_0, W)$

1: $x_0 = H_0 b$, $r_0 = Ax_0 - b$, $s_0 = r_0$
2: **for** $i = 1, \ldots, M$ **do**
3: $\quad y_i = As_i$ // collect observation
4: $\quad p(H \mid S, Y) = \mathcal{N}(H; H_i, W_i \otimes W_i)$ // inference (see previous slide)
5: $\quad x_i = H_i b$ // update mean estimate for x
6: $\quad r_i = Ax_i - b$ // new gradient. $r_i \perp r_{j<i}$
7: $\quad s_i = r_i - \sum_{j<i} \frac{y_j^\top r_i}{y_j^\top s_j} s_j$ // next action (conjugate direction)
8: **end for**

Conjugate Gradients

[Hestenes & Stiefel, 1952; Hennig, SIOPT 2015]

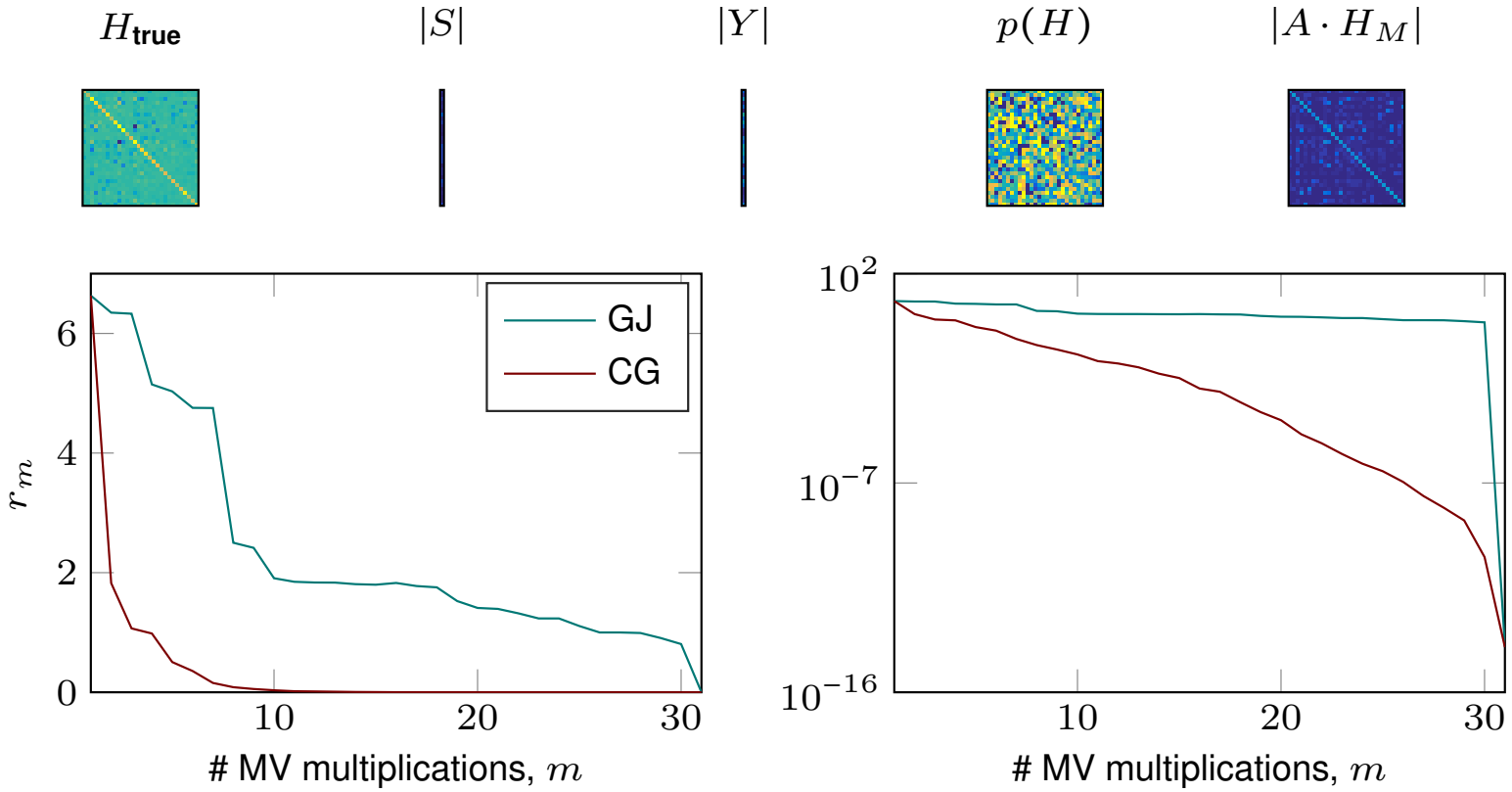
Set $H_0 = \epsilon I$, '$W = H$' as before. Some simplifications give:

Algorithm 2 Conjugate Gradients (A, b) [Hestenes & Stiefel 1952]

1: $r_0 \leftarrow -b$, $p_0 \leftarrow -r_0$, $k \leftarrow 0$
2: **for** $k = 0, \ldots, M$ **do**
3: $\quad d \leftarrow Ap_k$
4: $\quad \alpha_k \leftarrow r_k^\top r_k / p_k^\top d$
5: $\quad x_{k+1} \leftarrow x_k + \alpha_k p_k$
6: $\quad r_{k+1} \leftarrow r_k + \alpha_k d$
7: $\quad \beta_{k+1} \leftarrow r_{k+1}^\top r_{k+1} / r_k^\top r_k$
8: $\quad p_{k+1} \leftarrow -r_{k+1} + \beta_{k+1} p_k$
9: **end for**

Conjugate Gradients as Inference

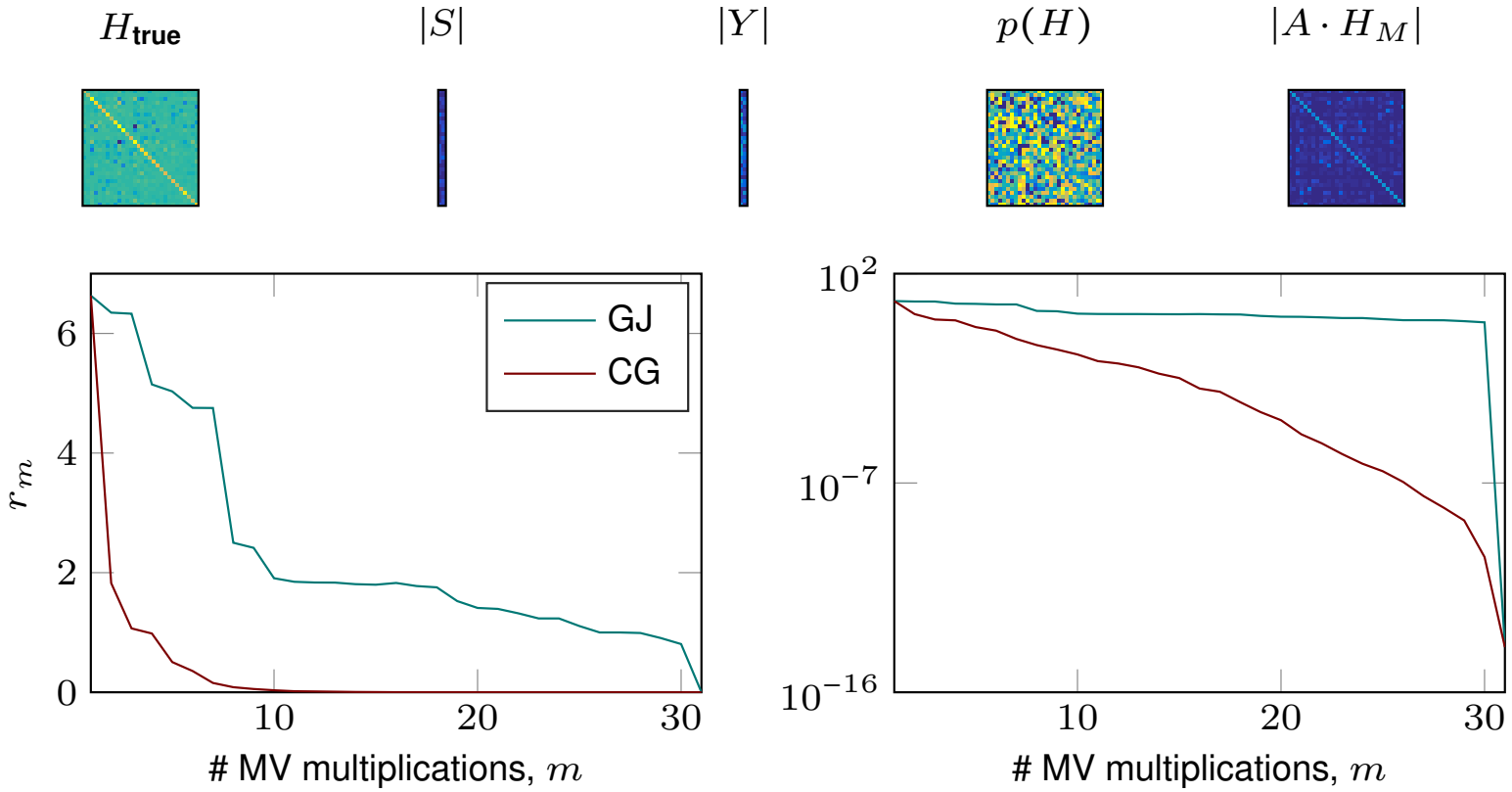
[Hestenes & Stiefel, 1952; Hennig, SIOPT 2015]



The Method of Conjugate Gradients is maximum a-posteriori inference of $x = Hb$ under a well-scaled Gaussian prior on H, if the search directions are chosen from the sequence of residuals on $r_i = Ax_i - b$.

Conjugate Gradients as Inference

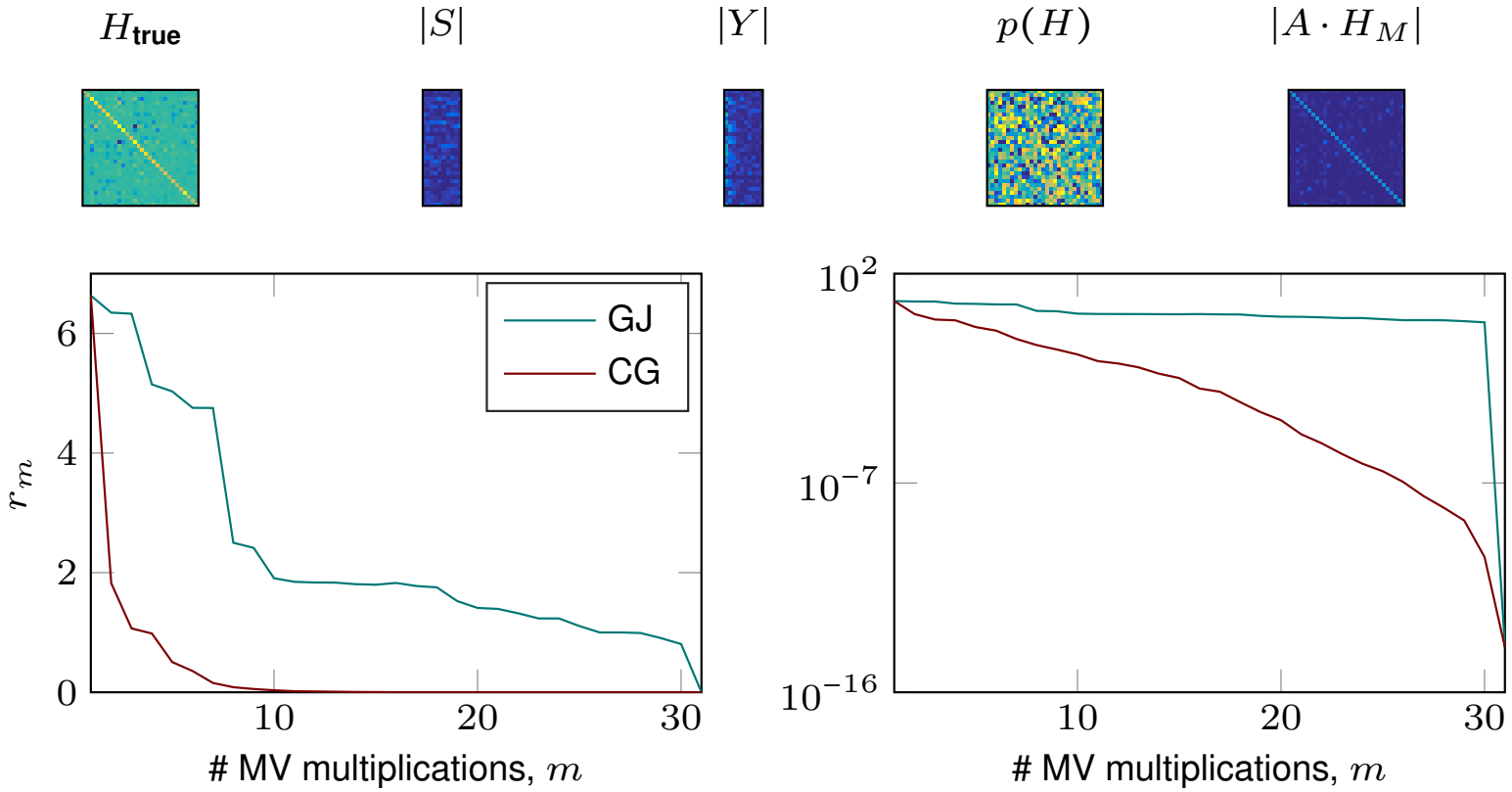
[Hestenes & Stiefel, 1952; Hennig, SIOPT 2015]



The Method of Conjugate Gradients is maximum a-posteriori inference of $x = Hb$ under a well-scaled Gaussian prior on H, if the search directions are chosen from the sequence of residuals on $r_i = Ax_i - b$.

Conjugate Gradients as Inference

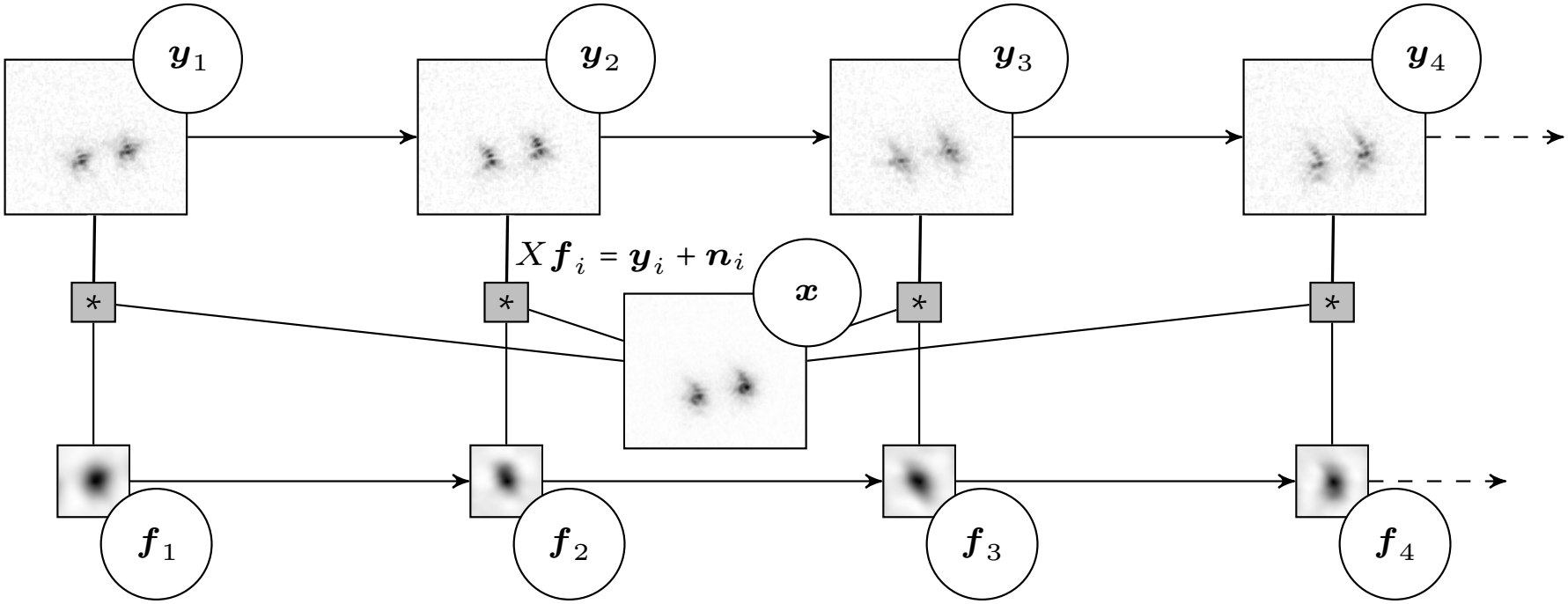
[Hestenes & Stiefel, 1952; Hennig, SIOPT 2015]



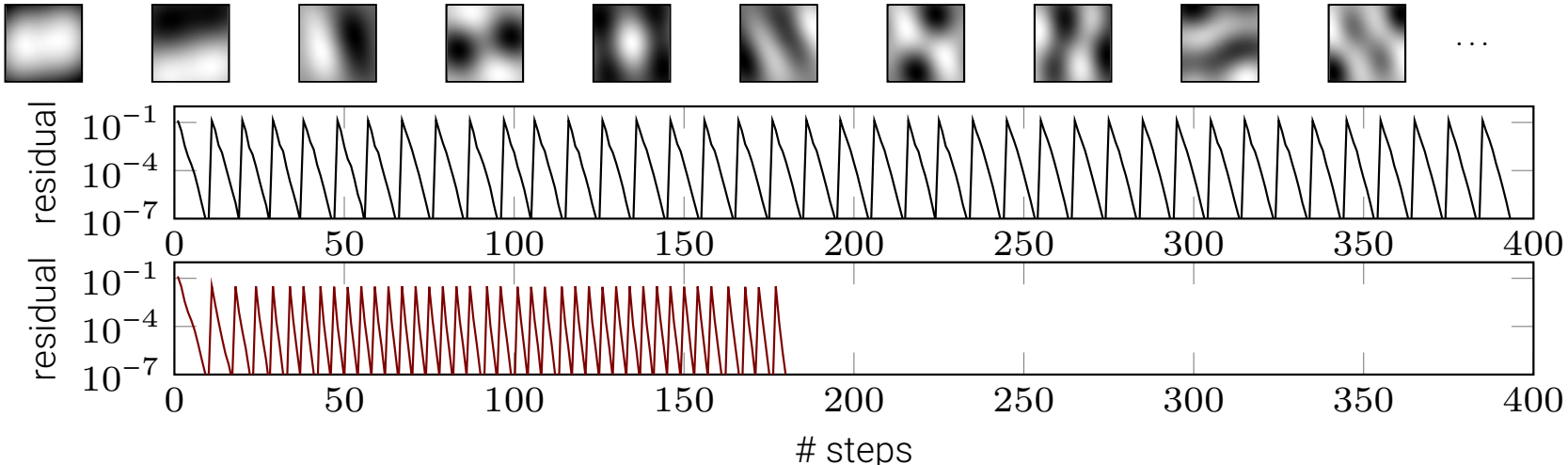
The Method of Conjugate Gradients is maximum a-posteriori inference of $x = Hb$ under a well-scaled Gaussian prior on H, if the search directions are chosen from the sequence of residuals on $r_i = Ax_i - b$.

Transfer Learning in Computation

"recycling Krylov sequences" [Parks et al., SISC 2006; Hennig, Osborne, Girolami, 2015]



eigen-vectors of inferred approximation to X^{-1}:



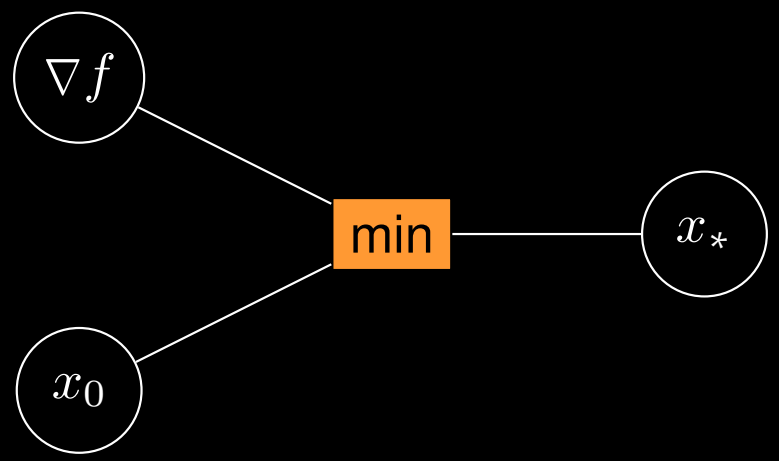
Summary: Linear Algebra

- basic algorithms have probabilistic interpretation as MAP inference from Gaussian priors on H
  - Gaussian Elimination: inference along unit vector projections
  - Conjugate-Gradients: inference along gradients of specific linear problem
- structured (factorization) assumptions required to achieve low computational cost
- calibrated uncertainty can be added at low cost, from regularity of collected numbers
- information can be shared between related computations through covariance models

Nonlinear Optimization

(just a quick aside)

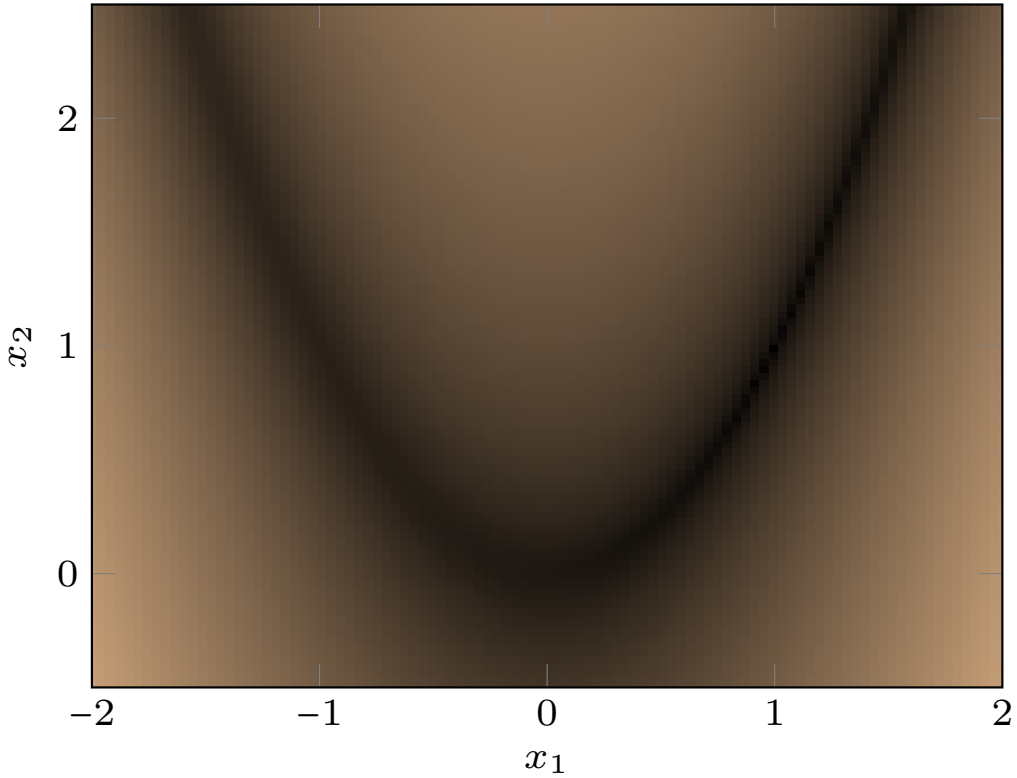
$$f : \mathbb{R}^N \to \mathbb{R} \qquad\qquad 0 \overset{!}{=} \nabla f(x_*)$$



BFGS is a filter

just a marginal remark

[Hennig & Kiefel, ICML/JMLR 2013]



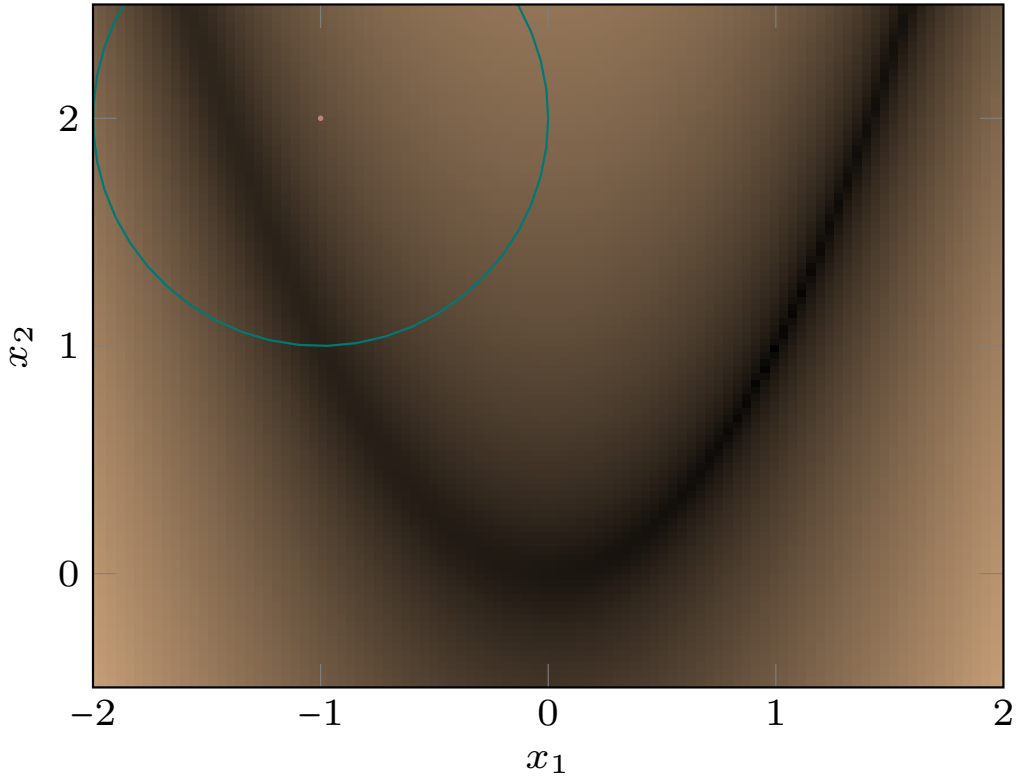
$$f(x) \approx f(x_t) + (x - x_t)^\top \nabla f(x_t) + {}^1\!/\!_2 (x - x_t)^\top A(x_t)(x - x_t)$$

$$x_{t+1} = x_t - \alpha H_M \nabla f(x_t) \approx x_t - \alpha A^{-1} \nabla f(x_t)$$

BFGS is a filter

just a marginal remark

[Hennig & Kiefel, ICML/JMLR 2013]



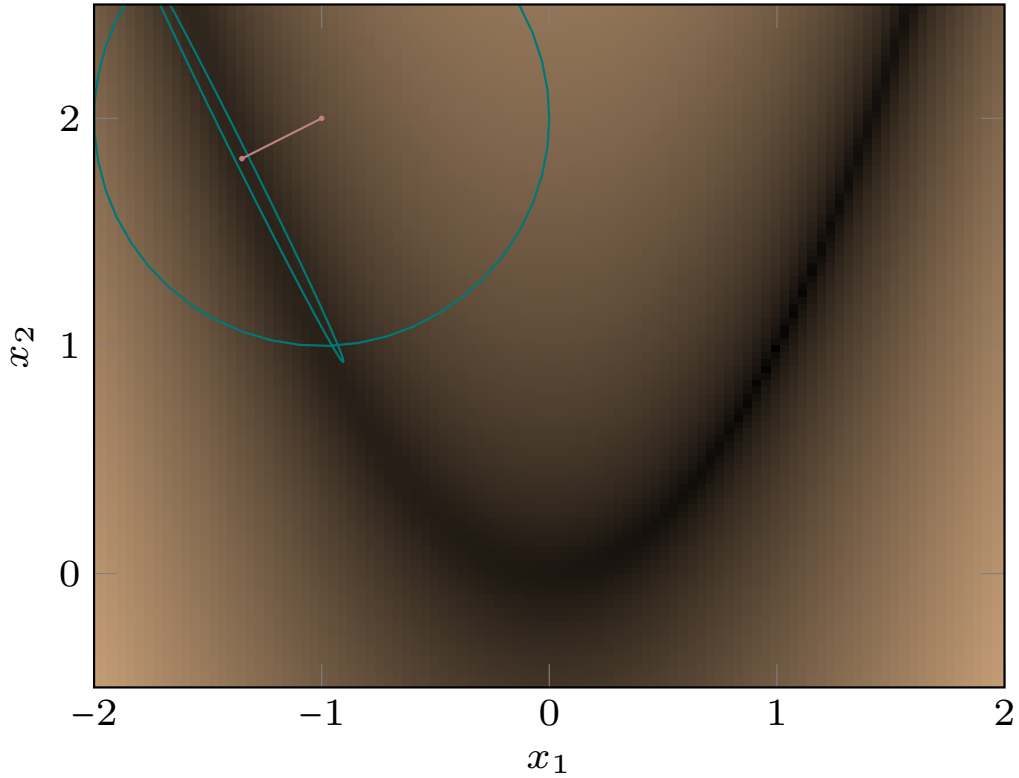
$$f(x) \approx f(x_t) + (x - x_t)^\top \nabla f(x_t) + {}^1\!/\!_2 (x - x_t)^\top A(x_t)(x - x_t)$$

$$x_{t+1} = x_t - \alpha H_M \nabla f(x_t) \approx x_t - \alpha A^{-1} \nabla f(x_t)$$

BFGS is a filter

just a marginal remark

[Hennig & Kiefel, ICML/JMLR 2013]



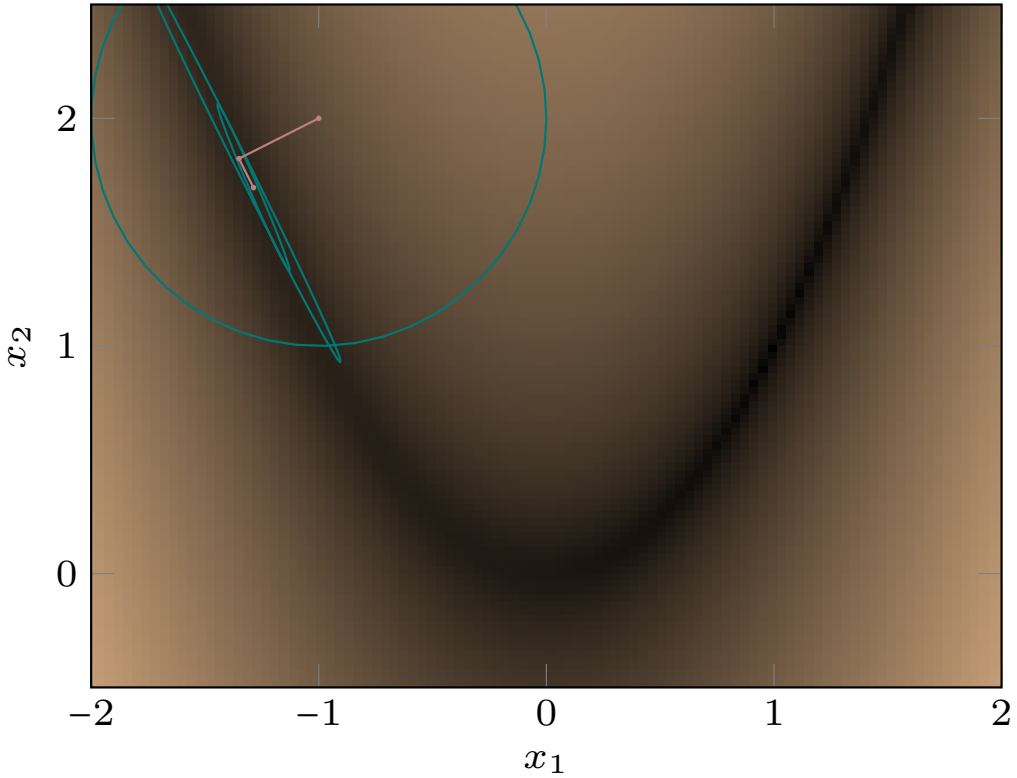
$$f(x) \approx f(x_t) + (x - x_t)^\top \nabla f(x_t) + {}^1\!/\!_2 (x - x_t)^\top A(x_t)(x - x_t)$$

$$x_{t+1} = x_t - \alpha H_M \nabla f(x_t) \approx x_t - \alpha A^{-1} \nabla f(x_t)$$

BFGS is a filter

just a marginal remark

[Hennig & Kiefel, ICML/JMLR 2013]



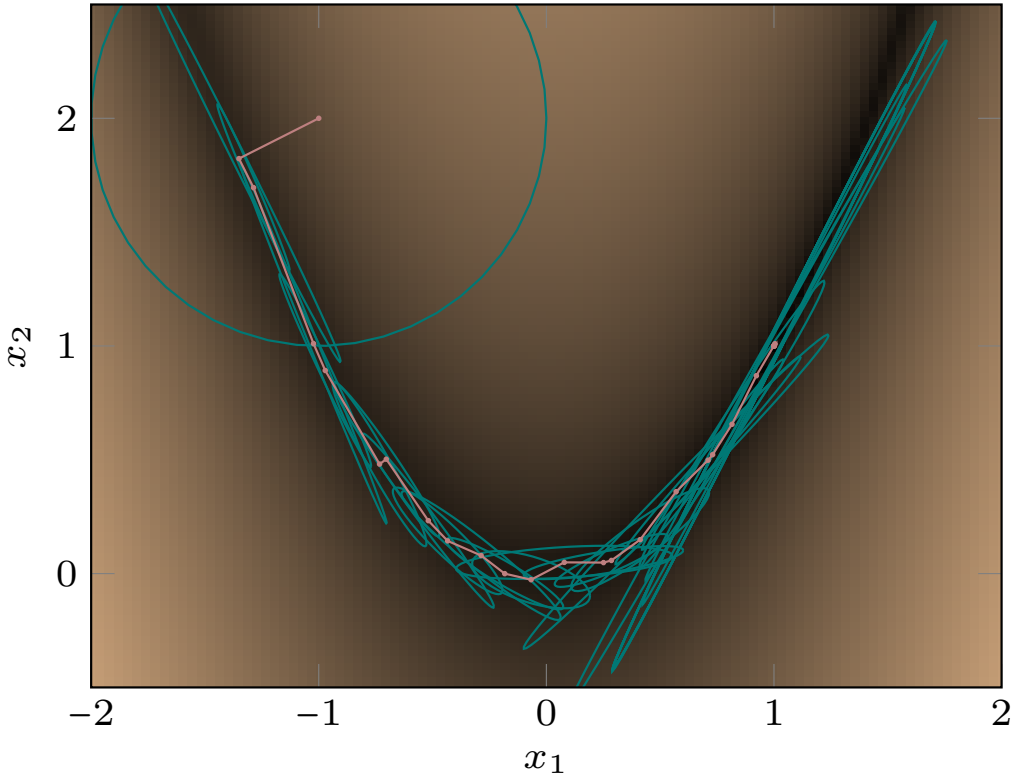
$$f(x) \approx f(x_t) + (x - x_t)^\top \nabla f(x_t) + {}^1\!/\!_2 (x - x_t)^\top A(x_t)(x - x_t)$$

$$x_{t+1} = x_t - \alpha H_M \nabla f(x_t) \approx x_t - \alpha A^{-1} \nabla f(x_t)$$

BFGS is a filter

just a marginal remark

[Hennig & Kiefel, ICML/JMLR 2013]

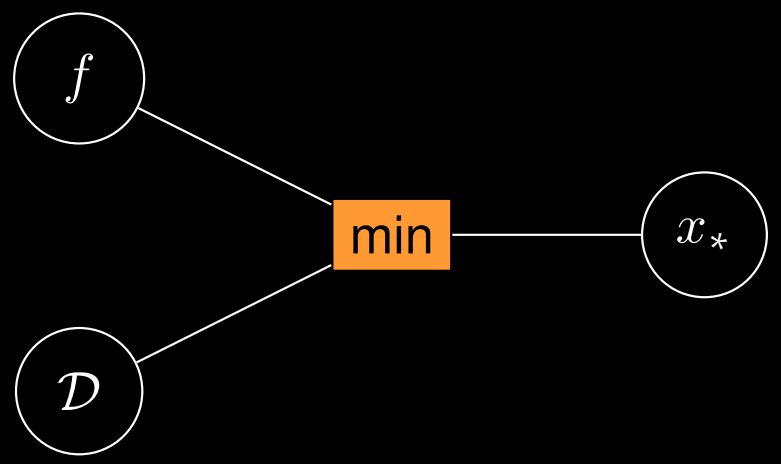


$$f(x) \approx f(x_t) + (x - x_t)^\top \nabla f(x_t) + {}^1\!/\!_2 (x - x_t)^\top A(x_t)(x - x_t)$$

$$x_{t+1} = x_t - \alpha H_M \nabla f(x_t) \approx x_t - \alpha A^{-1} \nabla f(x_t)$$

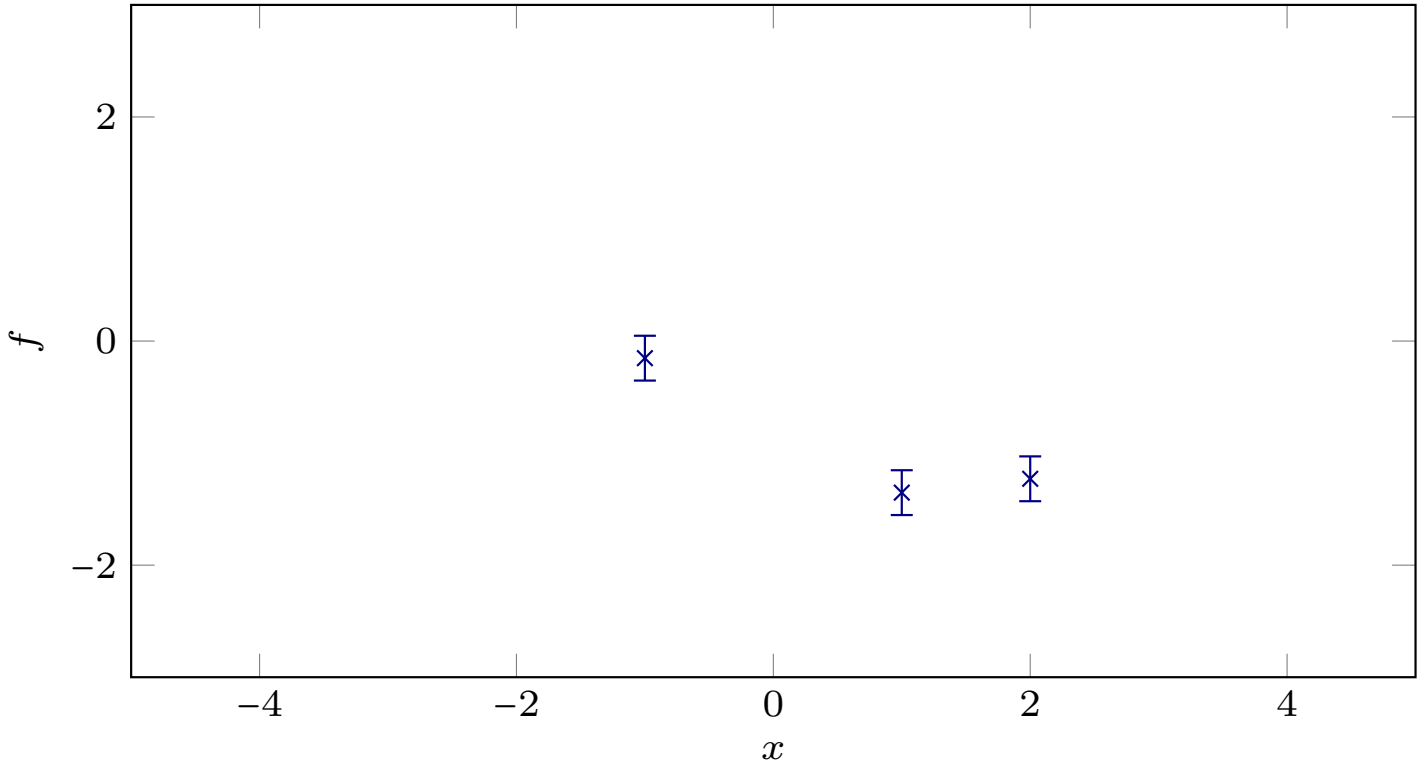
Global Optimization

$$f : \mathbb{R}^N \to \mathbb{R} \qquad\qquad 0 \overset{!}{=} \nabla f(x_*)$$



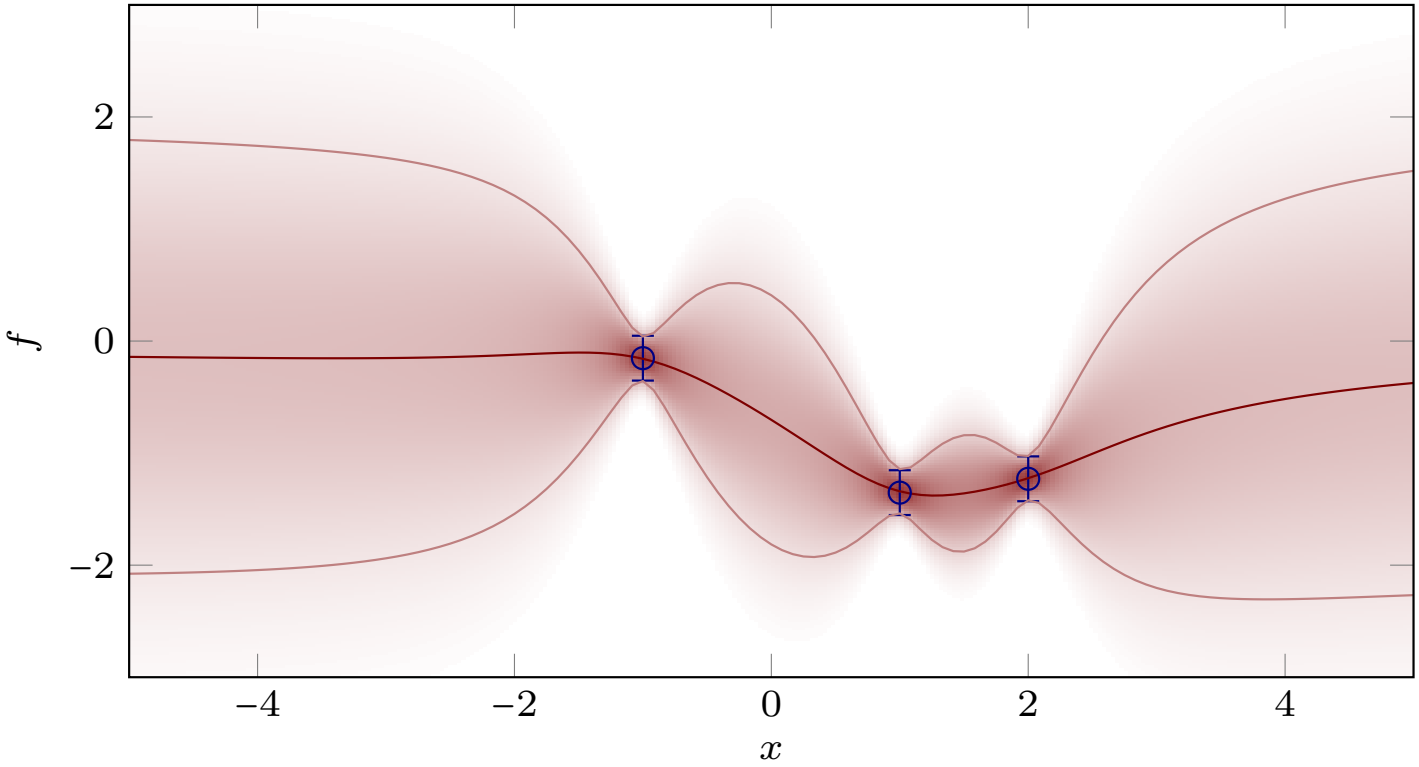
Bayesian Optimization

using a GP surrogate [Kushner, 1964; Jones, Schonlau, Welch, 1998]



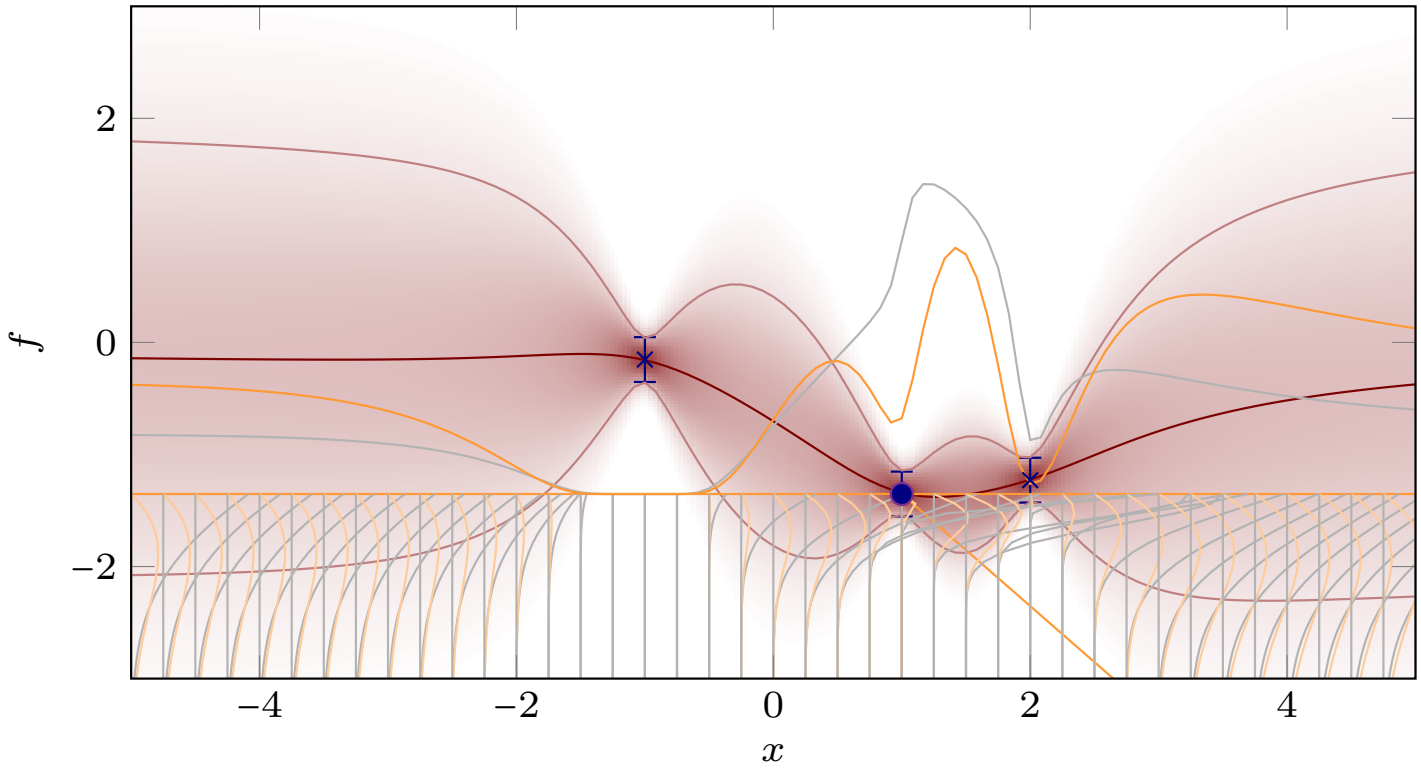
Bayesian Optimization

using a GP surrogate [Kushner, 1964; Jones, Schonlau, Welch, 1998]



Local Objectives

Expected Improvement and Probability of Improvement [Jones, Schonlau, Welch, 1998; Lizotte, 2008]

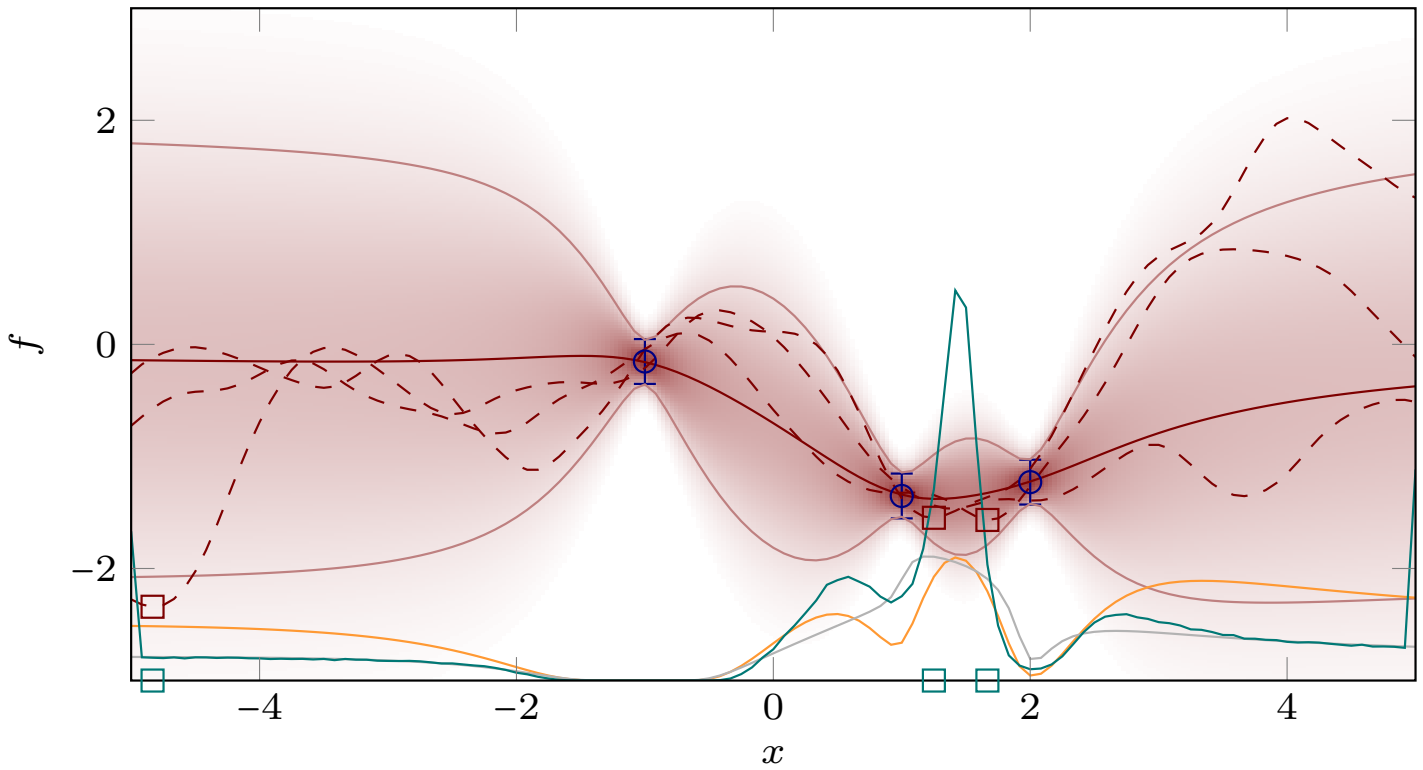


- $p(f(x) < \eta)$ Probability of Improvement [Lizotte, 2008]
- $\mathrm{E}_p[\min(0, \eta - f(x))]$ Expected Improvement [Jones et al., 1998]

Probabilistic Objectives

Entropy Search

[Hennig & Schuler, 2012]

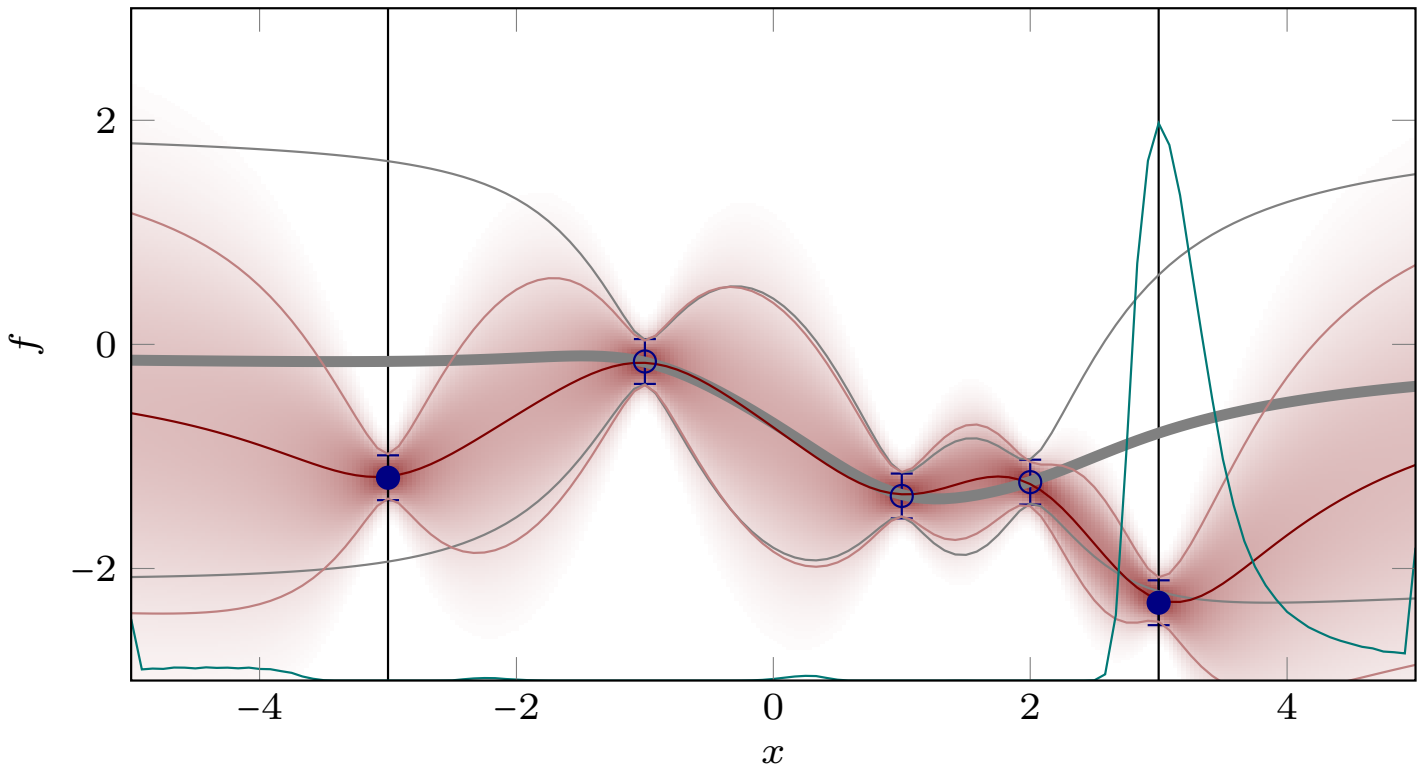


- $p(f(x) < \eta)$ — Probability of Improvement [Lizotte, 2008]
- $\mathrm{E}_p[\min(0, \eta - f(x))]$ — Expected Improvement [Jones et al., 1998]
- $p[x = \arg\min(f)]$ — [Hennig & Schuler, 2012]

Probabilistic Objectives

Entropy Search

[Hennig & Schuler, 2012]

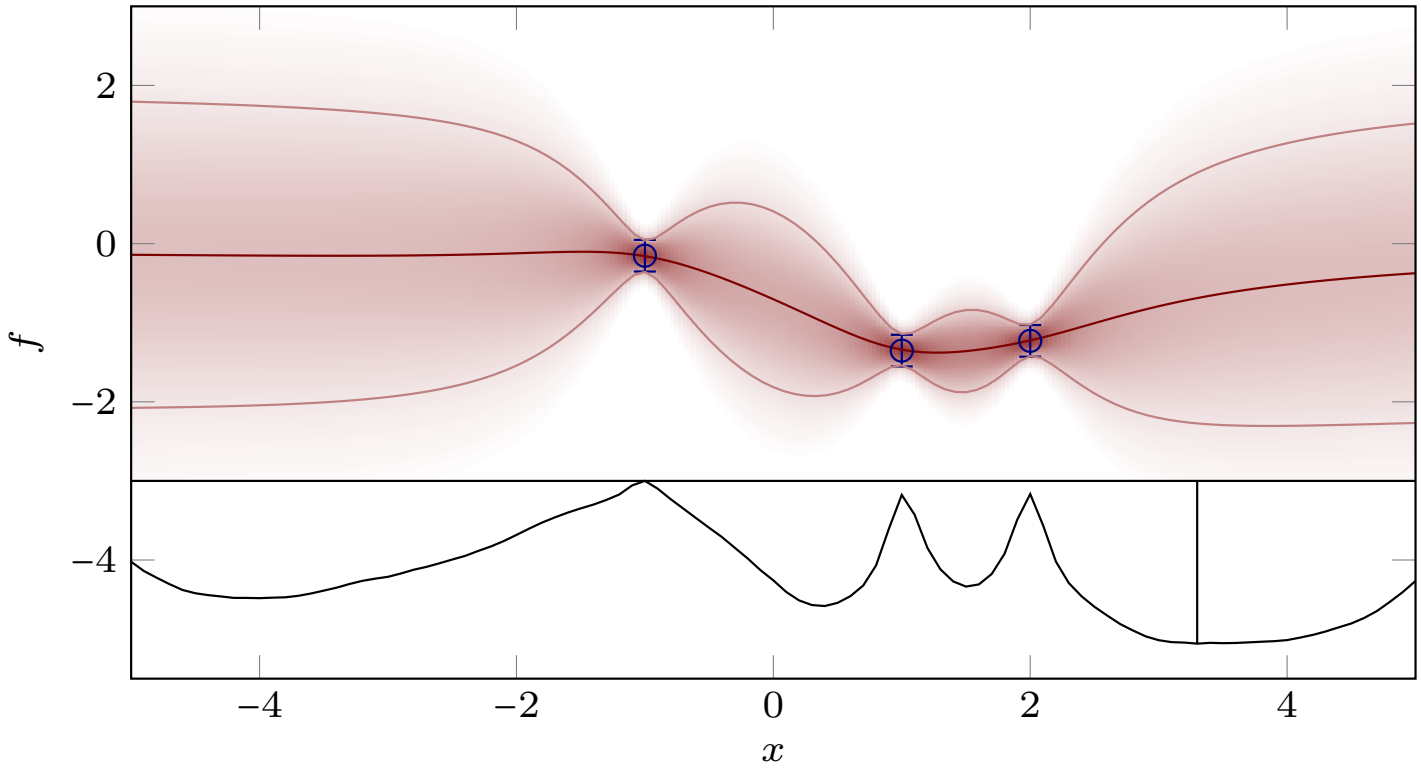


- $p(f(x) < \eta)$ — Probability of Improvement [Lizotte, 2008]
- $\mathrm{E}_p[\min(0, \eta - f(x))]$ — Expected Improvement [Jones et al., 1998]
- $p[x = \arg\min(f)]$ — [Hennig & Schuler, 2012]

Probabilistic Objectives

Entropy Search

[Hennig & Schuler, 2012]

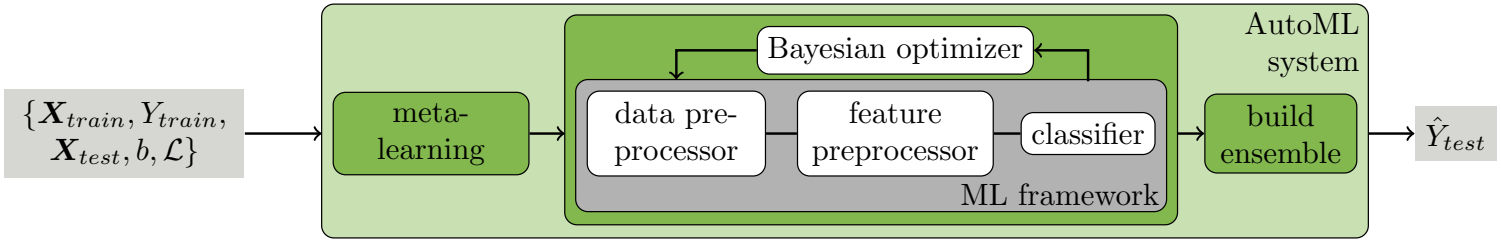


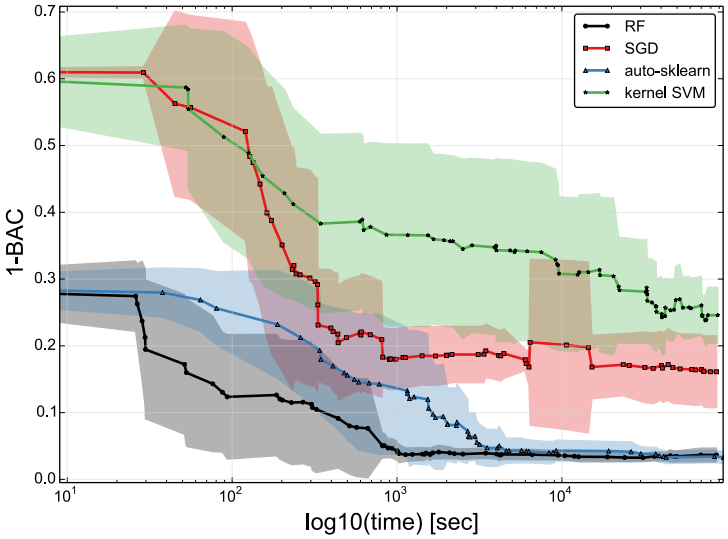
- $\mathrm{E}[\Delta \mathrm{H}[p(x = \arg\min(f))]]$
- expected information gain about location of minimum
- e.g. combine with evaluation cost to get cost-efficient exploration

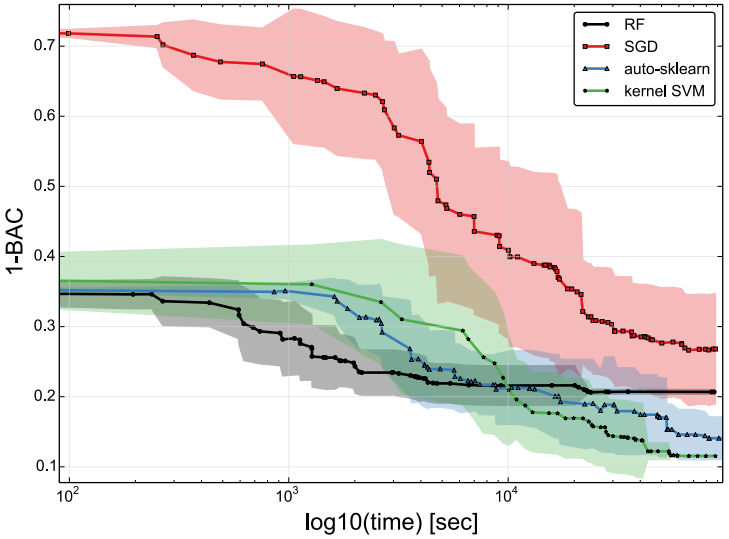
[K. Swersky, J. Snoek, R. Adams, 2013]

Automated Machine Learning

[M. Feurer, A. Klein, Katharina Eggensperger, J. Springenberg, M. Blum, F. Hutter, AutoML@ICML 2015]







Bayesian Optimization is usually sort of as a “top-level” method,
because it can be very expensive.
Numerical methods must be fast.
But Bayesian Optimization can still help in low-level computations!

Optimization with Noisy Gradients

A huge Problem in ML

- $x_{t+1} \leftarrow x_t - \alpha_t \nabla f(x_t)$

Optimization with Noisy Gradients

A huge Problem in ML

- $x_{t+1} \leftarrow x_t - \alpha_t \nabla f(x_t)$
- not invariant under even linear transformations

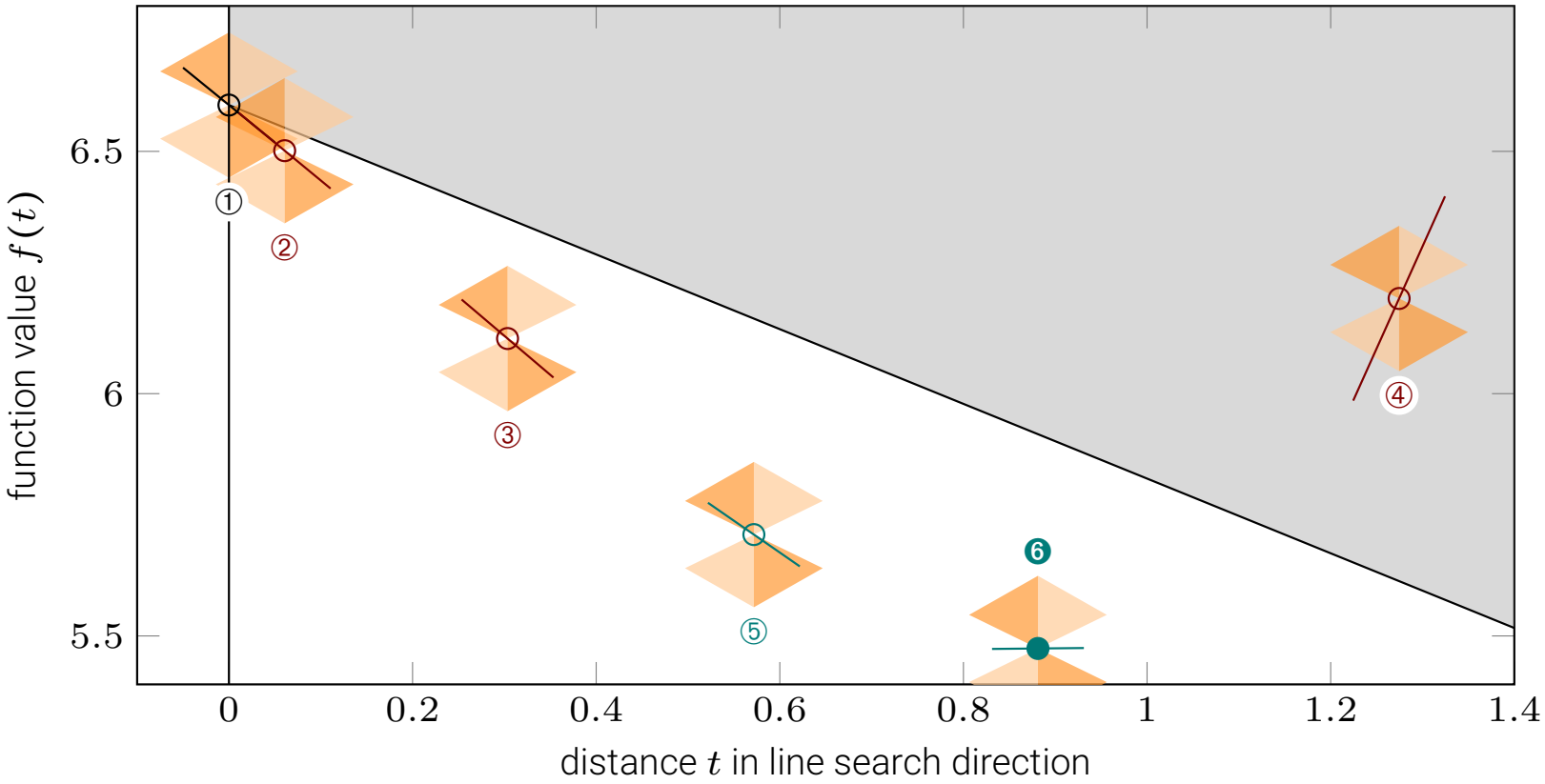
$$x \rightarrow Ax \qquad \rightsquigarrow \qquad \nabla f(x) \rightarrow A^{-1} \nabla f(x)$$

$$f(x) = 9.81 \frac{\mathrm{m}}{\mathrm{s}^2} \cdot h(x) \quad = 4\,473 \frac{\mathrm{kJ}}{\mathrm{kg}} \quad (@\ 456\mathrm{m}) \qquad \nabla f(x) = 5 \frac{\mathrm{J}}{\mathrm{kg} \cdot \mathrm{m}}$$

$$f(x) = 32.19 \frac{\mathrm{ft}}{\mathrm{s}^2} \cdot h(x) = 30.31 \frac{\mathrm{Cal}}{\mathrm{oz}} \quad (@\ 1496\mathrm{ft}) \quad \nabla f(x) = 1.03 \cdot 10^{-5} \frac{\mathrm{Cal}}{\mathrm{oz} \cdot \mathrm{ft}}$$

Line Searches

choosing meaningful step-sizes, at very low overhead [Wolfe, SIAM Review, 1969]



- Wolfe conditions: accept when

$$f(t) \leq f(0) + c_1 t f'(0) \quad \text{(W-I)} \qquad \text{and} \qquad f'(t) \geq c_2 f'(0) \quad \text{(W-II)}$$

What about Noisy Gradients?

stochastic gradient descent

- mini-batching gives noisy gradients

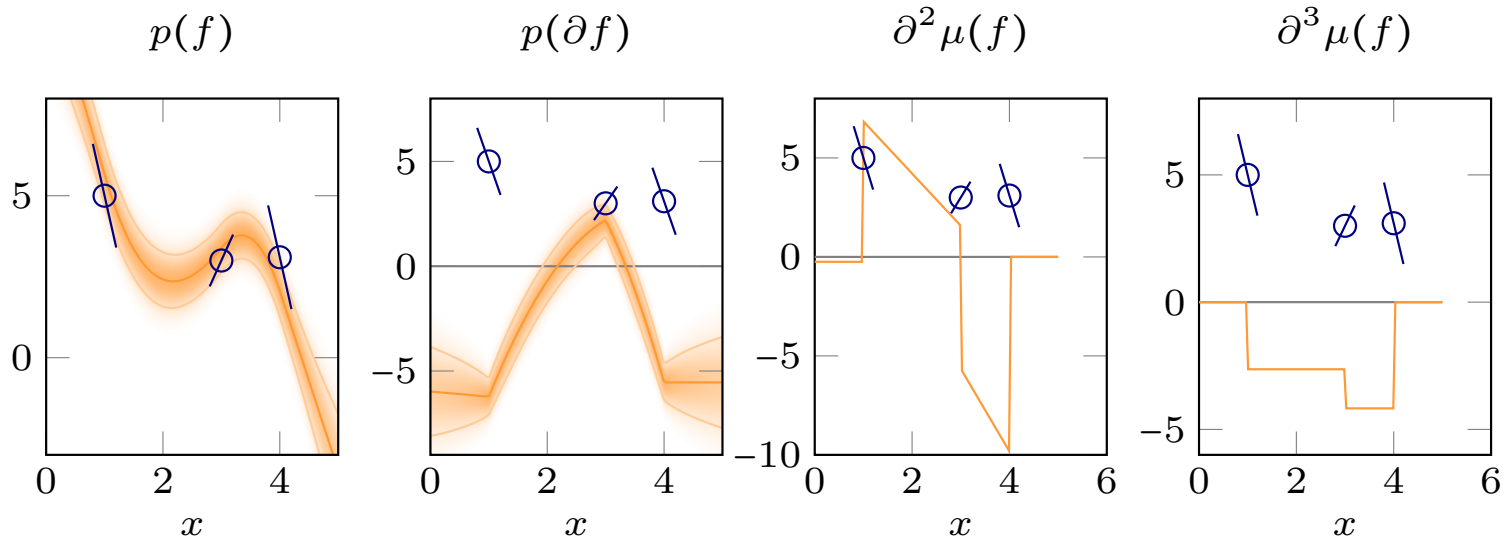
$$\mathcal{L}(x) := \frac{1}{M}\sum_{i=1}^{M}\ell(x, y_i) \approx \frac{1}{m}\sum_{j=1}^{m}\ell(x, y_j) =: \hat{\mathcal{L}}(x) \qquad m \ll M.$$

- for iid. batches, noise is approximately Gaussian

$$\hat{\mathcal{L}}(x) \approx \mathcal{L}(x) + \epsilon \qquad \epsilon \sim \mathcal{N}\left[0, \mathcal{O}\left(\frac{N-m}{m}\right)\right])$$

Building a Probabilistic Line Search

Step 1: robust surrogate [Mahsereci & Hennig, in review, arXiv 1502.02846]

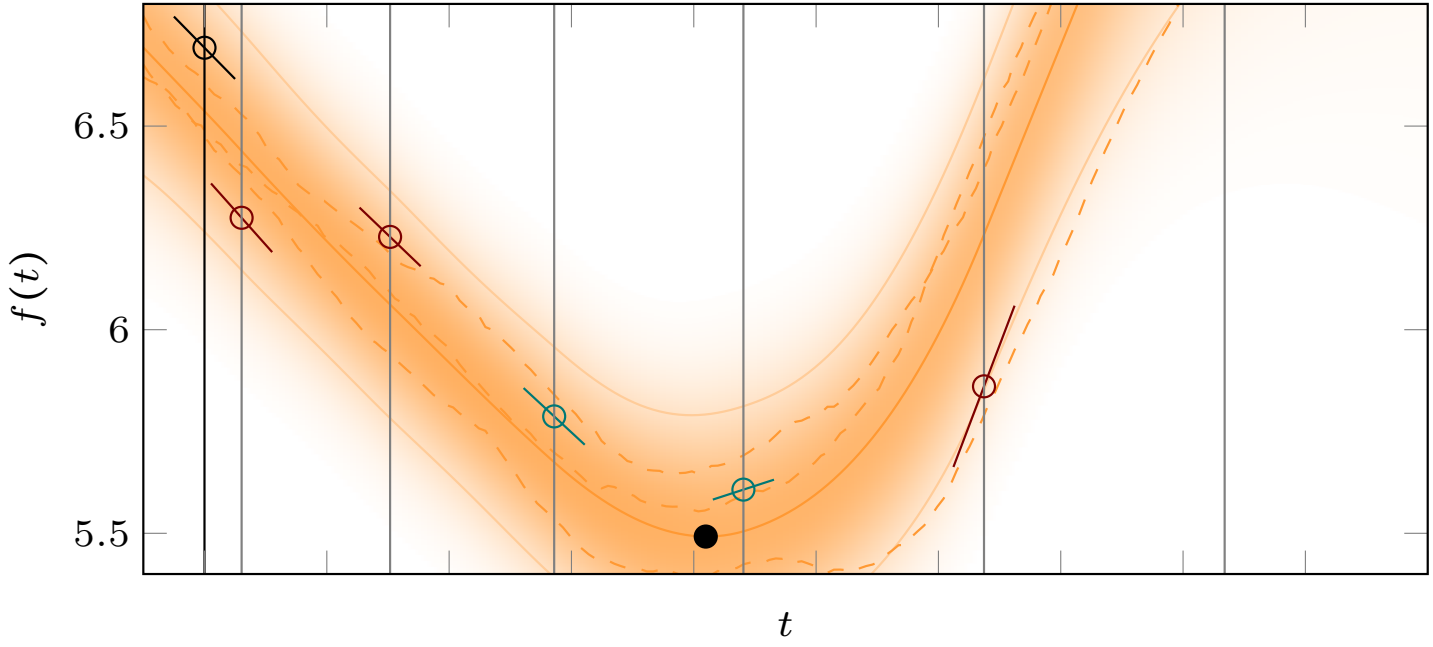


$$p(f) = \mathcal{GP}(f(t), 0; k) \qquad k(t,t') = [1/3 \min^3(t,t') + 1/2|t-t'|\min^2(t,t')]$$

- robust cubic spline posterior

Building a Probabilistic Line Search

Step 2: Bayesian Optimization for Exploration [Mahsereci & Hennig, in review, arXiv 1502.02846]



- analytically compute at most N local minima
- choose the one maximizing expected improvement

Building a Probabilistic Line Search

Step 3: Probabilistic Wolfe Termination Conditions [Mahsereci & Hennig, in review, arXiv 1502.02846]

$$f(t) \leq f(0) + c_1 t f'(0) \quad \text{(W-I)} \qquad \text{and}$$
$$f'(t) \geq c_2 f'(0) \quad \text{(W-II)}$$

$$\begin{bmatrix} a_t \\ b_t \end{bmatrix} = \begin{bmatrix} 1 & c_1 t & -1 & 0 \\ 0 & -c_2 & 0 & 1 \end{bmatrix} \begin{bmatrix} f(0) \\ f'(0) \\ f(t) \\ f'(t) \end{bmatrix} \geq 0.$$

$$p(a_t, b_t) = \mathcal{N}\left(\begin{bmatrix} a_t \\ b_t \end{bmatrix}; \begin{bmatrix} m_t^a \\ m_t^b \end{bmatrix}, \begin{bmatrix} C_t^{aa} & C_t^{ab} \\ C_t^{ba} & C_t^{bb} \end{bmatrix} \right),$$

with $\quad m_t^a = \mu(0) - \mu(t) + c_1 t \mu'(0) \quad$ and $\quad m_t^b = \mu'(t) - c_2 \mu'(0)$

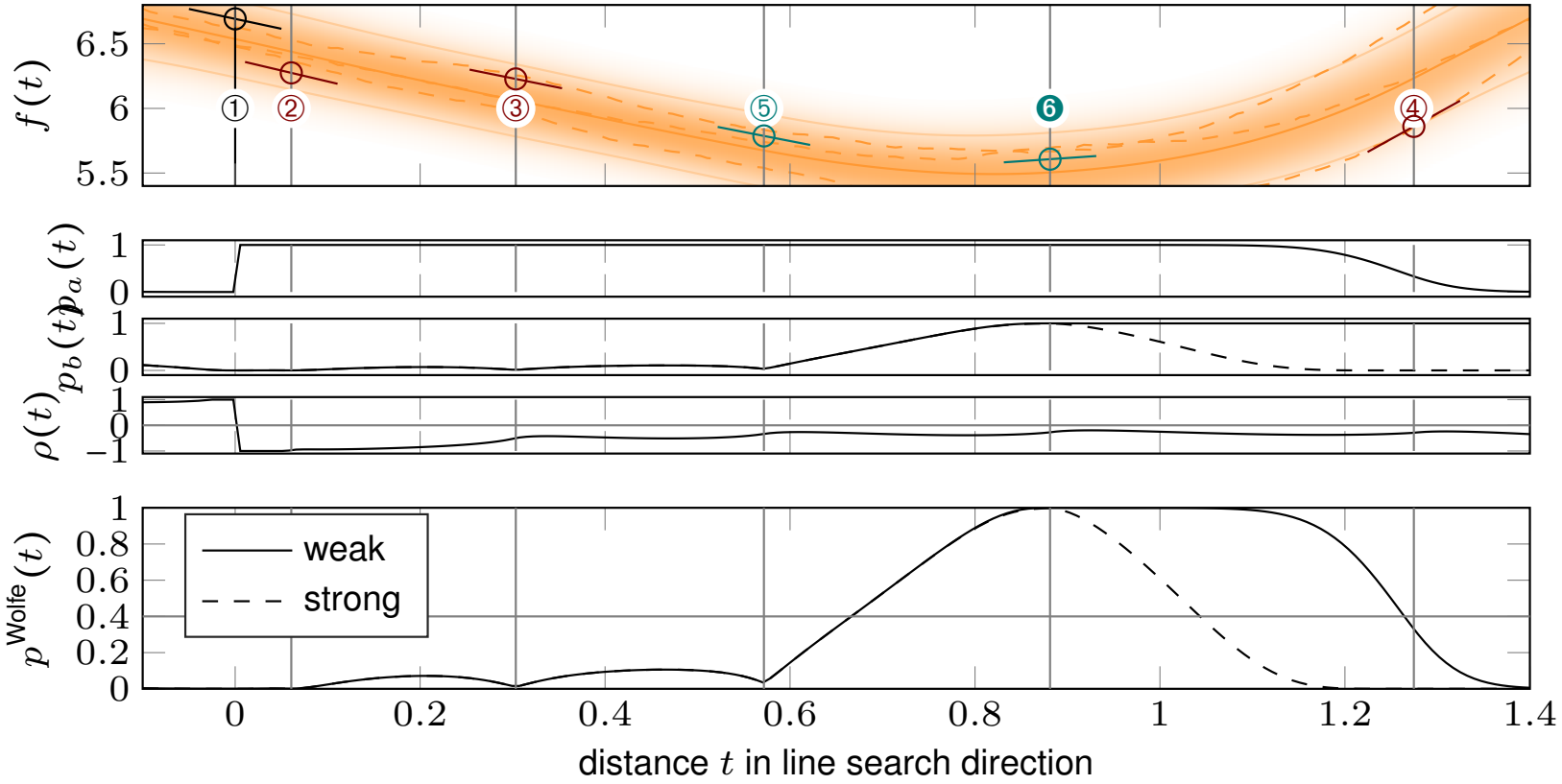
and $\quad C_t^{aa} = \tilde{k}_{00} + (c_1 t)^2 \, {}^{\partial}\tilde{k}_{00}^{\partial} + \tilde{k}_{tt} + 2[c_1 t({}^{\partial}\tilde{k}_{00}^{\partial} - {}^{\partial}\tilde{k}_{0t}) - \tilde{k}_{0t}]$

$$C_t^{bb} = c_2^2 \, {}^{\partial}\tilde{k}_{00}^{\partial} - 2c_2 \, {}^{\partial}\tilde{k}_{0t}^{\partial} + {}^{\partial}\tilde{k}_{tt}^{\partial}$$

$$C_t^{ab} = C_t^{ba} = -c_2({}^{\partial}\tilde{k}_{00}^{\partial} + c_1 t \, {}^{\partial}\tilde{k}_{00}^{\partial}) + (1 + c_2) \, {}^{\partial}\tilde{k}_{0t} + c_1 t \, {}^{\partial}\tilde{k}^{\partial}{}_{0t} - \tilde{k}_{tt}^{\partial}.$$

Probabilistic Line Searches

fast univariate Bayesian optimization [Mahsereci & Hennig, in review, arXiv 1502.02846]



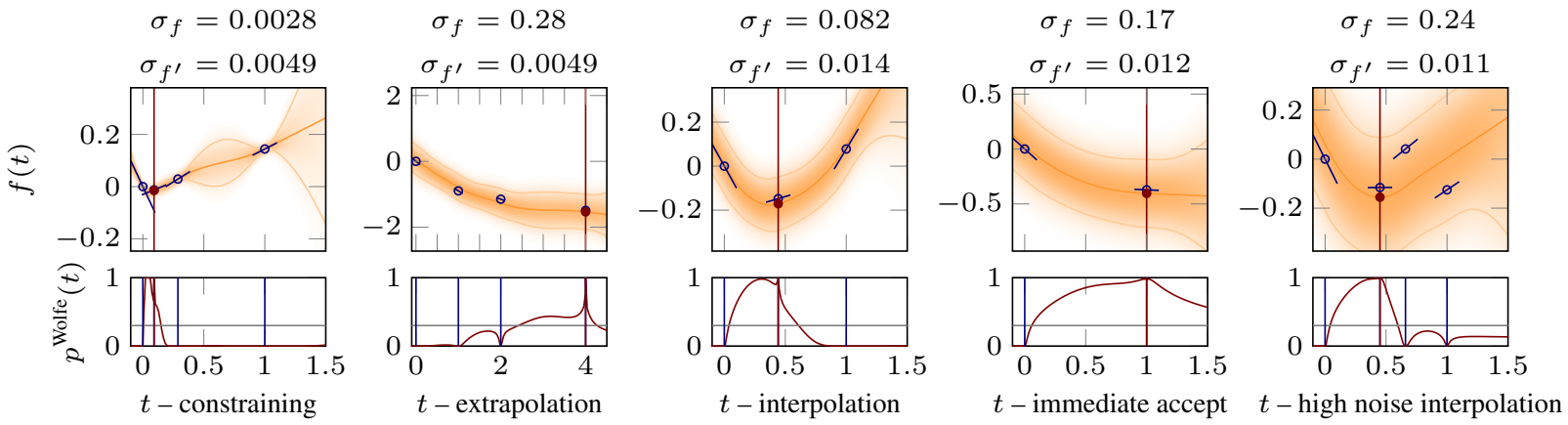
- Wolfe conditions: accept when

$$f(t) \leq f(0) + c_1 t f'(0) \quad \text{(W-I)} \qquad \text{and} \qquad f'(t) \geq c_2 f'(0) \quad \text{(W-II)}$$

- Probabilistic Wolfe conditions: accept when $p(\text{W-I} \wedge \text{W-II}) > 1 - \epsilon$

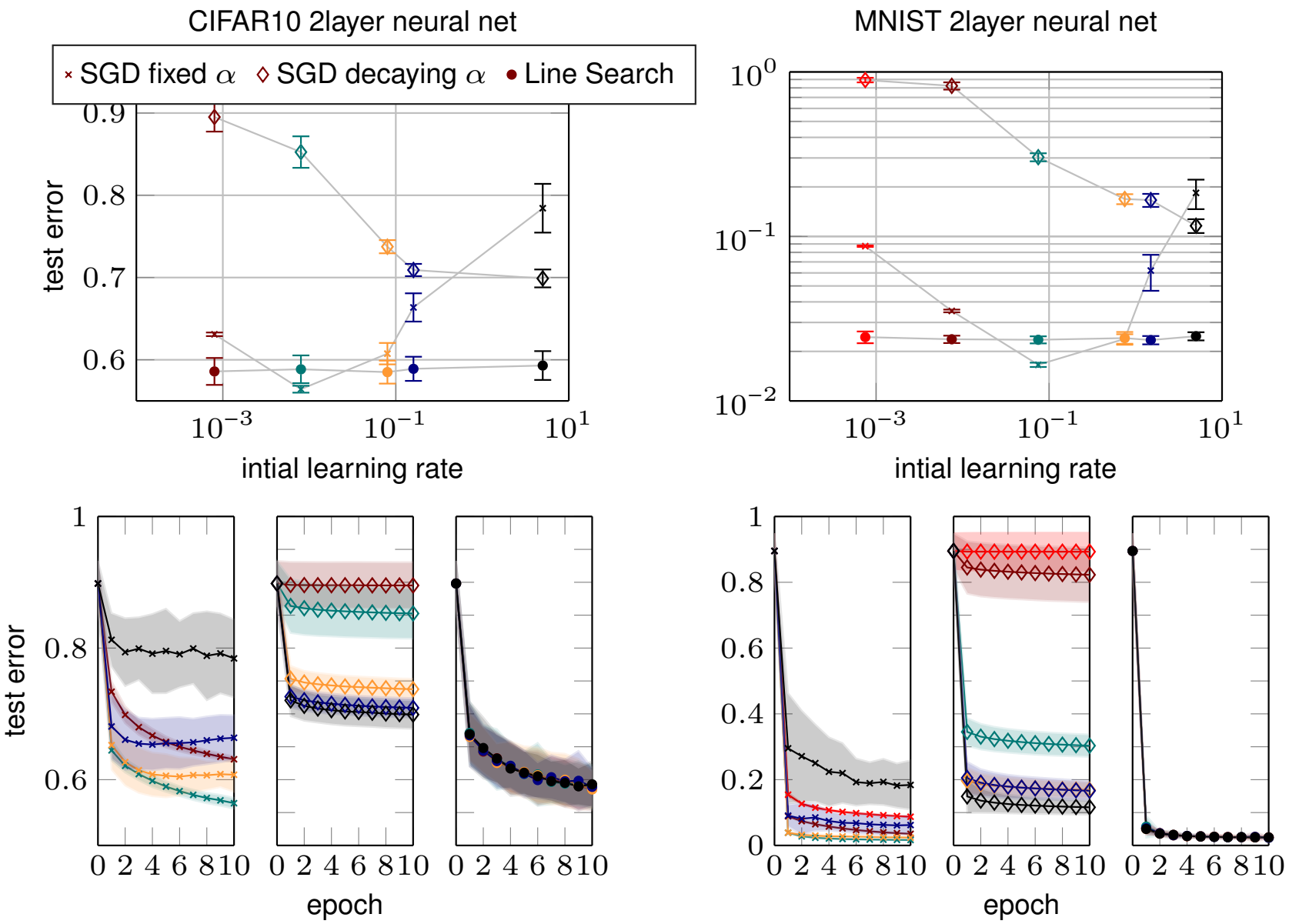
Probabilistic Line Searches in Action

some curated snapshots [Mahsereci & Hennig, in review, arXiv 1502.02846]



Forget about Learning Rates

probabilistic line searches automatically tune SGD [M. Mahsereci & P.H., in review, arXiv 1502.02846]



Probabilistic Numerics

— the big picture —

- Computation is Inference. Performing a computation means collecting information about the value of a latent quantity
- some basic algorithms are equivalent to Gaussian MAP inference
  - Gaussian Quadrature rules for Integration
  - Runge-Kutta solvers for ODEs
  - Conjugate Gradients for linear algebra
  - BFGS et al. for nonlinear optimization
- probabilistic formulations of computation offer opportunities for gains in efficiency and functionality

Do not think of numerical sub-routines as black boxes. They are active learning machines, and a primary source of efficiency gains.

Probabilistic Numerics

— applications —

- sampling for visualization
- customized numerics using structured priors to add information
- multi-task numerics using covariance models to share information
- uncertainty propagation using message passing
  - numerical methods for noisy inputs
  - identification of error / failure sources

ML has focussed on uncertainty from data;
it is time to consider uncertainty from computation.

Probabilistic Numerics

— a young community —

Uncertainty over the result of a computation at runtime is an exciting paradigm, with a wealth of applications and many, even fundamental, open questions.

Join us at
http://probabilistic-numerics.org
See you soon at a PN workshop?