Aalto University
School of Science
and Technology

# Advances in using GPs with derivative observations

Gaussian Process approximations 2017 –workshop

by Eero Siivola[1], joint work with Aki Vehtari[1], Juho Piironen[1], Javier González[2], Jarno Vanhatalo[3] and Olli-Pekka Koistinen[1]

[1] Aalto University, Finland

[2] Amazon, Cambridge, UK

[3] Univeristy of Helsinki, Finland

# Contents of this talk

- Theory behind GPs + derivatives
- GP-NEB
- Automatic monotonicity detection with GPs
- Bayesian optimization with derivative sign information

# Theory: GP + derivative observations

How to use (partial) derivatives with GPs?
We need to consider two parts:

- Covariance function
- Likelihood function
  - Posterior -> Inference method

# Covariance function

Nice property (See e.g. Papoulis [1991, ch. 10]):

$$\operatorname{cov}\left(\frac{\partial f^{(1)}}{\partial \mathbf{x}_g^{(1)}}, f^{(2)}\right) = \frac{\partial}{\partial \mathbf{x}_g^{(1)}} \operatorname{cov}\left(f^{(1)}, f^{(2)}\right) = \frac{\partial}{\partial \mathbf{x}_g^{(1)}} k\left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\right)$$

and:

$$\begin{aligned}\operatorname{cov}\left(\frac{\partial f^{(1)}}{\partial \mathbf{x}_g^{(1)}}, \frac{\partial f^{(2)}}{\partial \mathbf{x}_h^{(2)}}\right) &= \frac{\partial^2}{\partial \mathbf{x}_g^{(1)} \partial \mathbf{x}_h^{(2)}} \operatorname{cov}\left(f^{(1)}, f^{(2)}\right) \\ &= \frac{\partial^2}{\partial \mathbf{x}_g^{(1)} \partial \mathbf{x}_h^{(2)}} k\left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\right)\end{aligned}$$

Let $\mathbf{X} = \left[\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right]^T$ and $\tilde{\mathbf{X}} = \left[\tilde{\mathbf{x}}^{(1)}, \ldots, \tilde{\mathbf{x}}^{(m)}\right]^T$, be points where we observe function values and partial derivative values.

The covariance between latent function values $\mathbf{f}_{\mathbf{X}} = \left[f^{(1)}, \ldots, f^{(n)}\right]^T$ and latent function derivative values $\tilde{\mathbf{f}}'_{\tilde{\mathbf{X}}} = \left[\frac{\partial \tilde{f}^{(1)}}{\partial \tilde{\mathbf{x}}_g^{(1)}}, \ldots, \frac{\partial \tilde{f}^{(m)}}{\partial \tilde{\mathbf{x}}_g^{(m)}}\right]^T$ is:

$$\mathbf{K}_{\mathbf{X},\tilde{\mathbf{X}}} = \begin{bmatrix} \frac{\partial}{\partial \tilde{\mathbf{x}}_g^{(1)}} \mathrm{cov}(f^{(1)}, \tilde{f}^{(1)}) & \cdots & \frac{\partial}{\partial \tilde{\mathbf{x}}_g^{(m)}} \mathrm{cov}(f^{(1)}, \tilde{f}^{(m)}) \\ \vdots & \ddots & \vdots \\ \frac{\partial}{\partial \tilde{\mathbf{x}}_g^{(1)}} \mathrm{cov}(f^{(n)}, \tilde{f}^{(1)}) & \cdots & \frac{\partial}{\partial \tilde{\mathbf{x}}_g^{(m)}} \mathrm{cov}(f^{(n)}, \tilde{f}^{(m)}) \end{bmatrix} = \mathbf{K}_{\tilde{\mathbf{X}},\mathbf{X}}^T$$

And between latent function derivative values $\tilde{\mathbf{f}}_{\tilde{\mathbf{X}}}$ and $\tilde{\mathbf{f}}_{\tilde{\mathbf{X}}}$

$$\mathbf{K}_{\tilde{\mathbf{X}},\tilde{\mathbf{X}}} = \begin{bmatrix} \frac{\partial^2}{\partial \tilde{\mathbf{x}}_g^{(1)} \partial \tilde{\mathbf{x}}_g^{(1)}} \mathrm{cov}(\tilde{f}^{(1)}, \tilde{f}^{(1)}) & \cdots & \frac{\partial^2}{\partial \tilde{\mathbf{x}}_g^{(1)} \partial \tilde{\mathbf{x}}_g^{(m)}} \mathrm{cov}(\tilde{f}^{(1)}, \tilde{f}^{(m)}) \\ \vdots & \ddots & \vdots \\ \frac{\partial^2}{\partial \tilde{\mathbf{x}}_g^{(m)} \partial \tilde{\mathbf{x}}_g^{(1)}} \mathrm{cov}(\tilde{f}^{(m)}, \tilde{f}^{(1)}) & \cdots & \frac{\partial^2}{\partial \tilde{\mathbf{x}}_g^{(m)} \partial \tilde{\mathbf{x}}_g^{(m)}} \mathrm{cov}(\tilde{f}^{(m)}, \tilde{f}^{(m)}) \end{bmatrix}$$

# Likelihood function

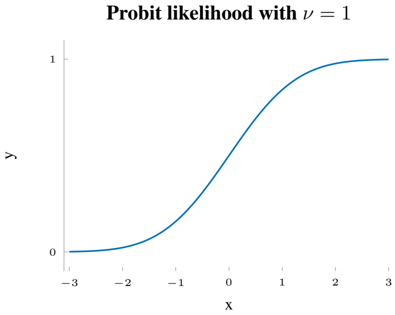
Observations are assumed independent given latent function values:

$$p(\mathbf{y}, \tilde{\mathbf{y}}' | \mathbf{f}_{\mathbf{X}}, \tilde{\mathbf{f}}'_{\tilde{\mathbf{X}}}) = \left( \prod_{i=1}^{n} p(y^{(i)} | f^{(i)}) \right) \left( \prod_{i=1}^{m} p \left( \frac{\partial \tilde{y}^{(i)}}{\partial \mathbf{x}_g^{(i)}} \middle| \frac{\partial \tilde{f}^{(i)}}{\partial \mathbf{x}_g^{(i)}} \right) \right)$$

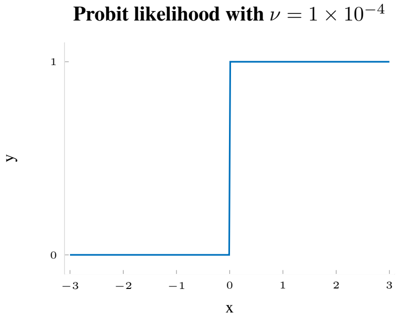
How to select the likelihood of derivatives?

- If direct derivative values can be observed: Gaussian likelihood
- If we only have hint about the direction: Probit likelihood with a tuning parameter (Riihimäki and Vehtari (2010))

$$p \left( \frac{\partial \tilde{y}^{(i)}}{\partial \mathbf{x}_g^{(i)}} \middle| \frac{\partial \tilde{f}^{(i)}}{\partial \mathbf{x}_g^{(i)}} \right) = \Phi \left( \frac{\partial \tilde{f}^{(i)}}{\partial \mathbf{x}_g^{(i)}} \frac{1}{\nu} \right), \text{ where } \left( \phi(a) = \int_{-\infty}^{a} \mathrm{N}(x | 0, 1) \mathrm{d}x \right)$$



Probit likelihood with $\nu = 1$
1
0
y
−3
−2
−1
0
1
2
3
x

Probit likelihood with $\nu = 1 \times 10^{-4}$
1
0
y
−3
−2
−1
0
1
2
3
x

# Posterior distribution

Posterior distribution of joint values:

$$p(\mathbf{f}, \tilde{\mathbf{f}}' | \mathbf{y}, \tilde{\mathbf{y}}', \mathbf{X}, \tilde{\mathbf{X}}) = \frac{p(\mathbf{f}, \tilde{\mathbf{f}}' | \mathbf{X}, \tilde{\mathbf{X}}) \left( \prod_{i=1}^{n} p(y^{(i)} | f^{(i)}) \right) \left( \prod_{i=1}^{m} p \left( \frac{\partial \tilde{y}^{(i)}}{\partial \mathbf{x}_g^{(i)}} \middle| \frac{\partial \tilde{f}^{(i)}}{\partial \mathbf{x}_g^{(i)}} \right) \right)}{Z}$$

Different parts:

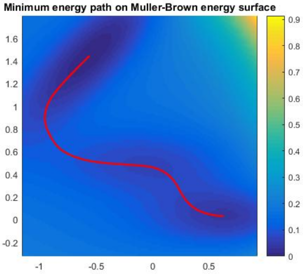
- $p(\mathbf{f}, \tilde{\mathbf{f}}' | \mathbf{X}, \tilde{\mathbf{X}})$ is Gaussian
- $p(y^{(i)} | f^{(i)})$ are Gaussian
- $p \left( \frac{\partial \tilde{y}^{(i)}}{\partial \mathbf{x}_g^{(i)}} \middle| \frac{\partial \tilde{f}^{(i)}}{\partial \mathbf{x}_g^{(i)}} \right)$ Gaussian/probit

The posterior distribution is either Gaussian or similar as in classification problems

- We might need posterior approximation methods

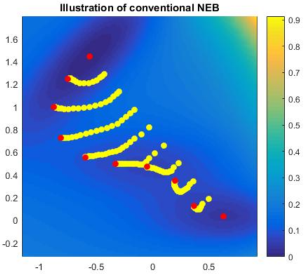
# Saddle point search using GPs + derivative observations

- The properties of the system can be described by an energy surface
- Finding a minimum energy path and the saddle point between two states is useful when determining properties of transitions



# Nudged elastic band (NEB)

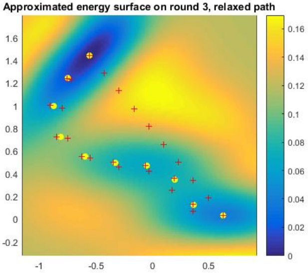
- Starting from an initial guess, the idea is to move the images downwards on the energy surface but keep them evenly spaced
- The images are moved along a force vector, which is a resultant of two components:
  - (Negative) energy gradient component perpendicular to the path
  - A spring force parallel to the path, which tends to keep the images evenly spaced



- The convergence of NEB may require hundreds or thousands of iterations
- Each iteration requires evaluation of the energy gradient for all images, which is often a time-consuming operation

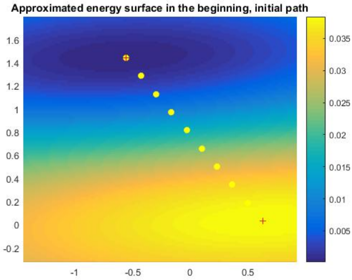
# Speedup of NEB

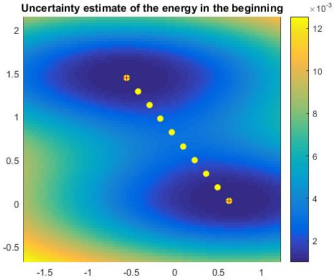
- Repeat until convergence:
  1. Evaluate the energy (and forces) at the images of the current path
  2. If path not converged, approximate the energy surface using machine learning based on the observations so far
  3. Find the predicted minimum energy path on the approximate surface and go to 1
- The details in paper by Peterson (2016)

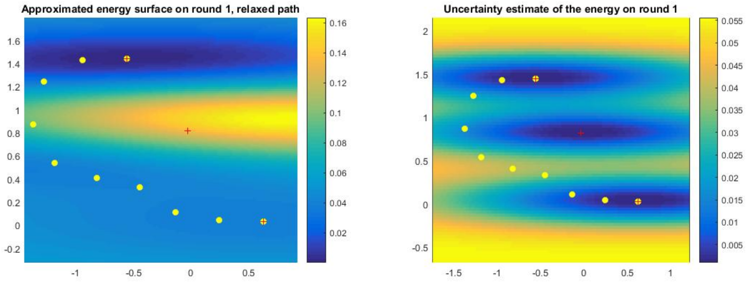


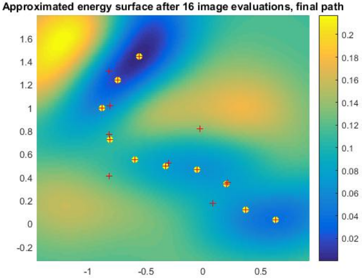
# Speedup of NEB with GP and derivatives

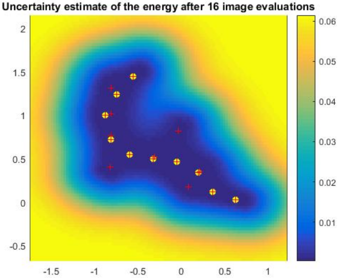
- Evaluate the energy (and forces) only at the image with the highest uncertainty
- Re-approximate the energy surface and find a new MEP guess after each image evaluation
- Convergence check:
  - If the magnitude of the force (may be accurate or approximation) is below the convergence limit for all images, we don't move the path, but evaluate more images, until the convergence limit is not met any more or all images have been evaluated
  - If we manage to evaluate all images without moving the path, we know for sure if the path is converged
- The details in paper by Koistinen, Maras, Vehtari and Jónsson (2016):


Approximated energy surface in the beginning, initial path
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-1
-0.5
0
0.5
0.035
0.03
0.025
0.02
0.015
0.01
0.005

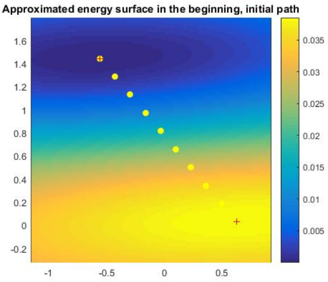

Uncertainty estimate of the energy in the beginning
$\times 10^{-3}$
2
1.5
1
0.5
0
-0.5
-1.5
-1
-0.5
0
0.5
1
12
10
8
6
4
2

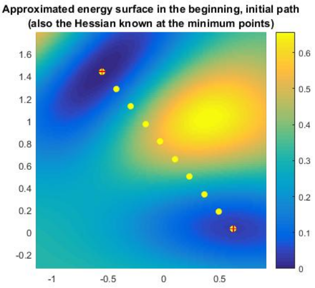

Approximated energy surface on round 1, relaxed path
Uncertainty estimate of the energy on round 1


Approximated energy surface after 16 image evaluations, final path
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
-1
-0.5
0
0.5
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02

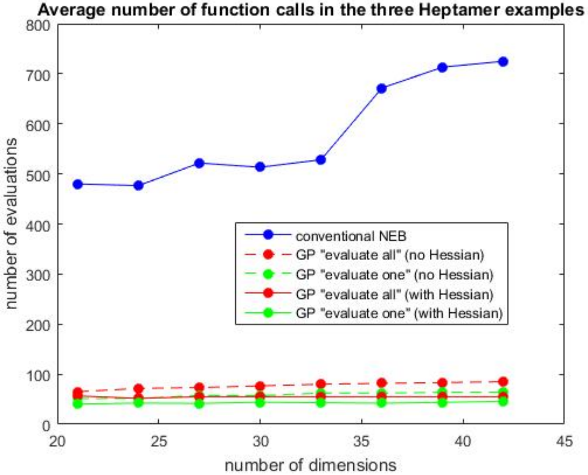

Uncertainty estimate of the energy after 16 image evaluations
2
1.5
1
0.5
0
-0.5
-1.5
-1
-0.5
0
0.5
1
0.06
0.05
0.04
0.03
0.02
0.01

- When evaluating the transition rates, the Hessian of the minimum points needs to be evaluated at some phase
- This information can be used to improve the GP approximations, especially in the beginning, when there is little information



# Comparison of methods in heptamer case study



# Automatic monotonicity detection

- Derivative sign information can be used to find monotonic input output directions
- The basic idea:
  - Add derivative sign observations to the GP model
  - See if the additions affect to the probability of the data
    - the dimension is monotonic if not
- The details in paper by Siivola, Piironen and Vehtari (2016)

# Theoretical background

Energy comparison:

$$E(\mathbf{y}, \tilde{\mathbf{y}}' | \mathbf{X}, \tilde{\mathbf{X}}_m) = -\log p(\mathbf{y}, \tilde{\mathbf{y}}' | \mathbf{X}, \tilde{\mathbf{X}}_m)$$
$$= -\log \left( p(\mathbf{y}|\mathbf{X}) \overbrace{p(\tilde{\mathbf{y}}' | \mathbf{y}, \mathbf{X}, \tilde{\mathbf{X}}_m)}^{\approx 1} \right) \approx E(\mathbf{y}|\mathbf{X}).$$

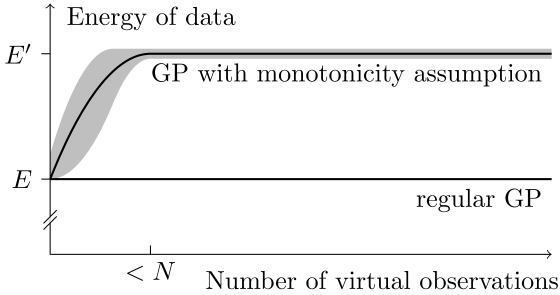



Figure: Change in energy in reality as a function of virtual derivative sign observations

# Using automatic monotonicity detection in modelling

- Monotonic dimensions can be detected from the data and used in modelling
- The method makes the modelling results especially on the borders.

# Experiment

- Six different functions of varying monotonicity
- Different amount of noise added to training samples (signal to noise ratio (SNR) between 0 and 1)
- Measure the log predictive posterior density of samples from a hold out set that resemble 20 % of the bordermost samples in the training data:

$$\text{lppd} = \sum_{i=1}^{L} \log \int p(y_i|f) p_{\text{post}}(f|x_i) df$$

- Do this for three different models for 200 times:
  - Use fixed monotonicity
  - Use monotonicity if the it does not change the energy (adaptive monotonicity)
  - Use model without derivative observations

# Results

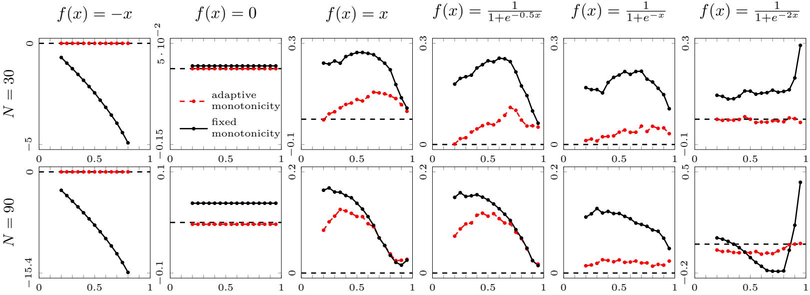



Figure: ΔLPPD of baseline and named method on y axis, SNR on x axis

# Multidimensional experiment

Diabetes data[1]:

- Target value: a measure of diabetes progression one year after baseline
- 10 dimensions
- Detect monotonic dimensions and use them if needed

[1] diabetes data, available at:
`http://web.stanford.edu/~hastie/Papers/LARS/diabetes.data`

# Results

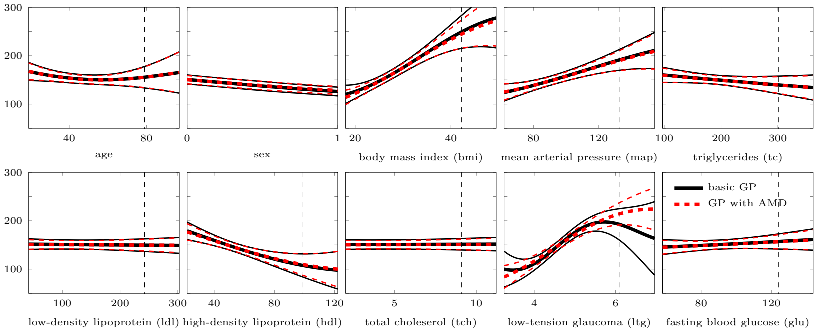



Figure: Target value as a function of single predictive values while others are kept at the median of dataset. Regular black lines correspond to regular GP mean and 90 % posterior central interval. Red dashed lines correspond to AMD GPs mean and standard deviation when body mass index and low-tension glaucoma are detected as increasing. Black dashed line corresponds to the largest value of covariate.

# Bayesian optimization with virtual derivative sign observations

Bayesian optimization (BO):

- A global optimization strategy designed to find the minimum of expensive black-box functions:
    1. Fit GP to the available dataset $\mathbf{X}$, $\mathbf{y}$
    2. Evaluate the function at a new location based on some acquisition function
    3. If stopping criterion is not met, go to 1
- Usually the search space is selceted so that the minimum is not on the border
- An over-exploration of the edges is a typical problem

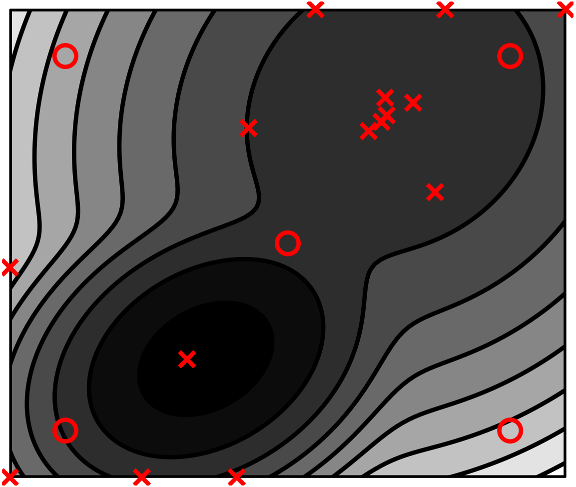


Figure: Over exploration of the edges visualized with LCB as an acquisition function. Circles are initial samples and crosses are acquisitions.

# Fixing over exploration with derivative sign observations

- By adding fake derivative observations to the borders, the over-exploration problem can be solved:
  1. Fit GP to the available dataset $\mathbf{X}$, $\mathbf{y}$
  2. Find a new location based on some acquisition function
  3. If the new location is at the border:
     - add a derivative sign observation to the border
  4. Else:
     - add the new location.
  5. If stopping criterion is not met, go to 1
- The details in paper by Siivola, Vehtari, Vanhatalo and Gonzalez (2017)

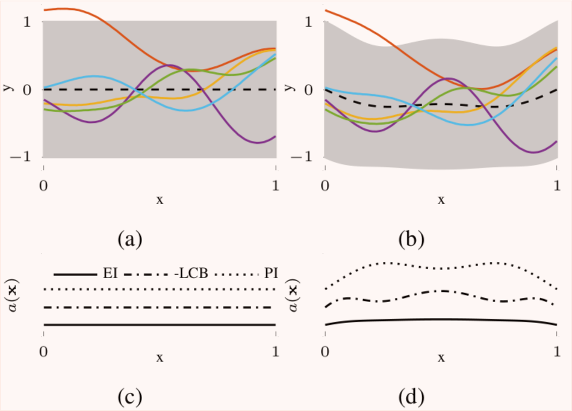



Figure: GP prior and acquisition functions for one dimensional space. a) and c), without fake derivative sign observations. b) and d) with derivative sign observations.

# Experiments

Metrics for comparing performances of two BO algorithm:

- *Percentual minimum difference* (PMD): PMD is designed to compare the absolute performances of the algorithms and intuitively it measures the difference of the best values of both algorithms.
- *Percentual hit difference* (PHD): PHD is created for comparing the speeds of the algorithms and intuitively it measures difference of how fast both algorithms are able to find good enough values.
- *Percentual border hit difference* (PBHD): Assuming that the minimum is not near the border, BHD tells the scaled difference of unnecessary samples taken near the borders.

- *Average evaluation distance difference* (AED): Intuitively, AED measures the overall performance of the algorithm before finding the minimum.
- *Virtual derivative observations per dimension* (VDO): Intuitively, larger VDOs are worse, since they increase the computational burden of the algorithm as GP's scale as $\mathcal{O}\left((n+q)^3\right)$.

The interpretation for the magnitude of PMD, PHD and PBHD are that negative values tell that the proposed method is better, the values are always scaled between $-1$ and 1 and the further away the value is from 0, the bigger the difference between the two methods is.

# Experiment 1

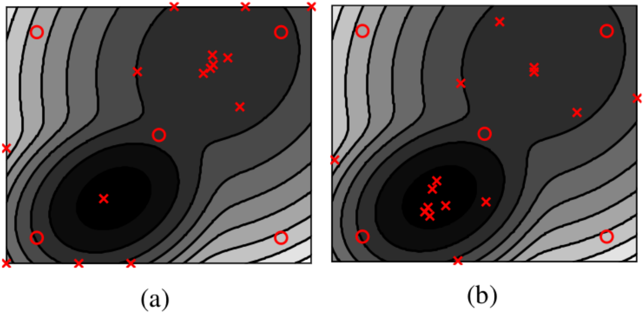
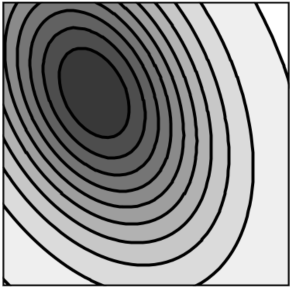
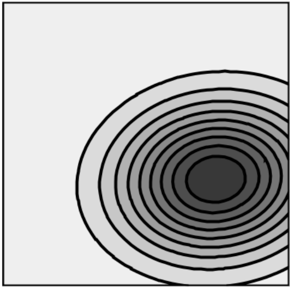



Figure 2: 15 acquisitions with (a) standard BO (b) BO with virtual derivative observations. In both figures the five red circles are the points used to initialise the GP and the 15 red crosses are the acquisitions. In both algorithms, the used acquisition function is LCB. In both figures, darker colors represent lower function values.

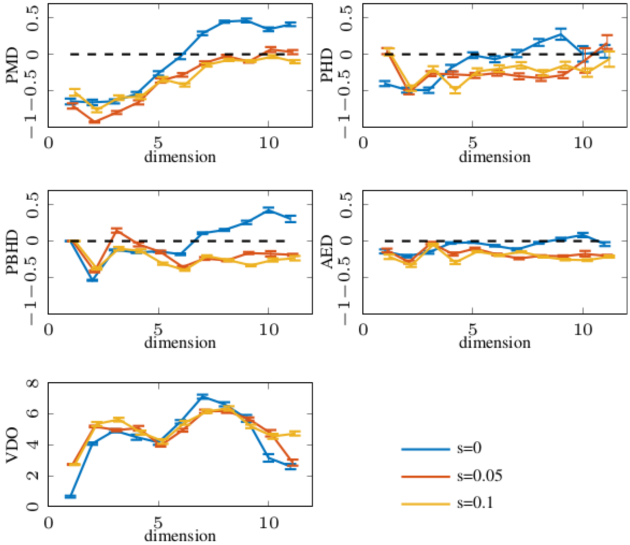
# Experiment 2

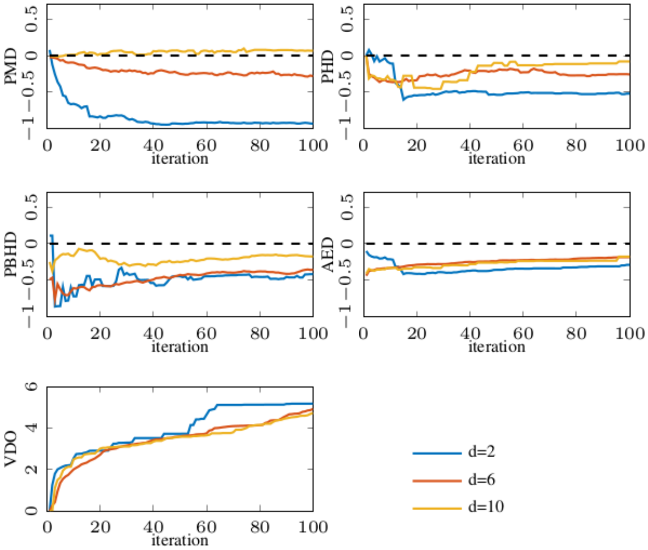
- 100 d-dimensional multivariate normal distribution functions as $d = 1, ..., 11$
- Different amount of noise added to the functions
- BO and BO with derivatives ran for 100 acquisitions





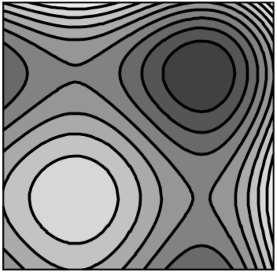
# Results

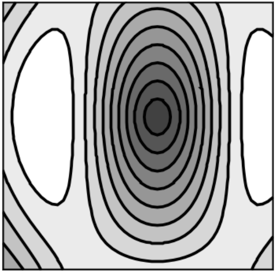



PMD
PHD
PBHD
AED
VDO
iteration
0
20
40
60
80
100
0.5
0
−0.5
−1
2
4
6
d=2
d=6
d=10

# Experiment 3

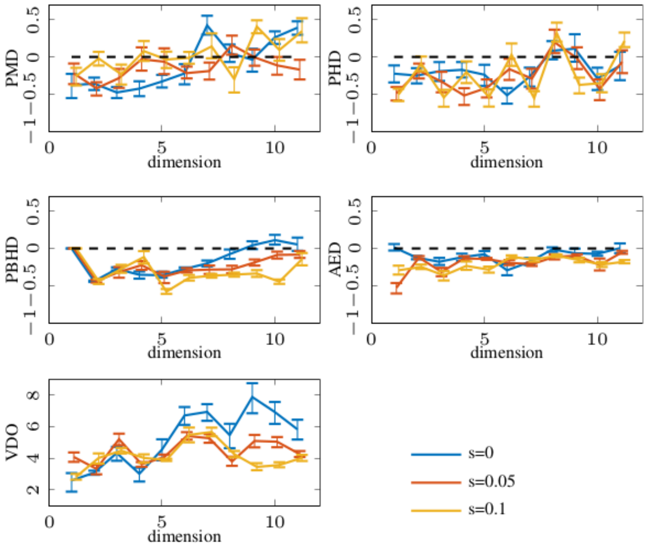
- Same as in experiment 2, but for sigopt function dataset[2]
- 113 functions from 1 to 11 dimensions

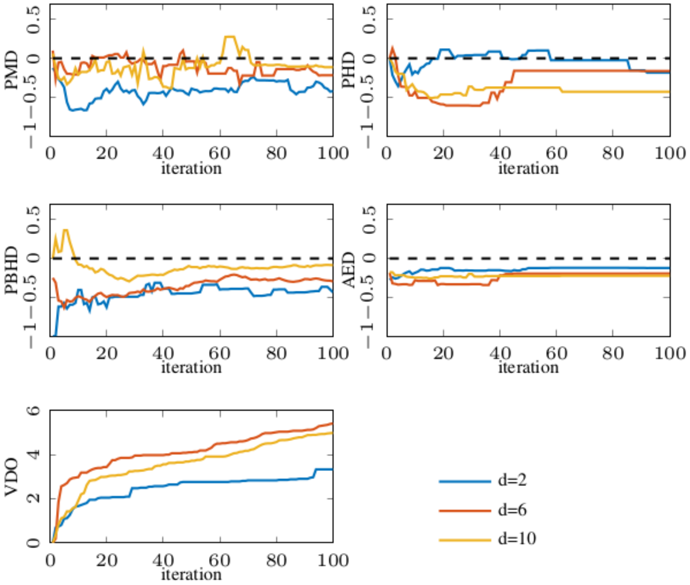




[2] Dataset available at: `https://github.com/sigopt/sigopt-examples`

# Results




PMD
PHD
PBHD
AED
VDO
iteration
d=2
d=6
d=10

# Summary

Derivatives can be used with GPs in many new ways:

- To improve accuracy of GPs in simulation of energy surfaces
- To automatically find monotonic dimensions from data
- To fix border over-exploration problem of BOs

# Questions?

- email: eero.siivola@aalto.fi

# References


- Papoulis, A. (1991). Probability, Random Variables, and Stochastic Processes. McGraw-Hill, New York. Third Edition.
- Riihimäki, J. and Vehtari, A. (2010) "Gaussian processes with monotonicity information." In proceedings of AISTATS 2010. vol. 9, pp. 645-652.
- Peterson (2016) "Acceleration of saddle-point searches with machine learning". In J. Chem. Phys., 145, p. 074106
- Koistinen, O.-P., Maras, E., Vehtari, A. and Jónsson, H. (2016) "Minimum energy path calculations with Gaussian process regression". Nanosystems: Physics, Chemistry, Mathematics, 2016, 7 (6), p. 925-935
- Siivola, E., Piironen, J., and Vehtari, A. (2016) "Automatic monotonicity detection for Gaussian Processes" arXiv: `https://arxiv.org/abs/1610.05440`
- Siivola, E., Vehtari, A., Vanhatalo, J., and González, J. (2017) "Bayesian optimization with virtual derivative sign observations" arXiv: `https://arxiv.org/abs/1704.00963`