SCALABLE GAUSSIAN PROCESSES FOR HIGH-DIMENSIONAL PROBLEMS

Mads Greisen Højlund, postdoctoral researcher

Department of Chemistry, Aarhus University

OUTLINE

- My way into GP research
- The curse of dimensionality
- GP regression (GPR)
- Two strategies for accelerating GPs
- CUTS-GPR

MY WAY INTO GP RESEARCH

MY BACKGROUND

- Chemistry at Aarhus University (2017–2021)
- PhD (2021–2026)
- Postdoc (2026–present)
- Quantum chemistry

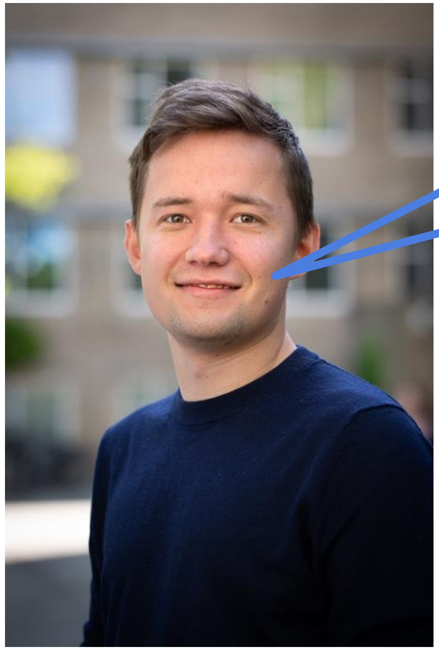
Prof. Ove Christiansen

$$i\hbar \frac{\mathrm{d}}{\mathrm{d}t}\Psi = \hat{H}\Psi$$

Wave function

Hamiltonian

NEW IDEAS



My PhD colleague
August

… … grid GPs … …
Kronecker products … …
very cool stuff … …

Me on paternity leave (2025)

Don't Get Your Kroneckers in a Twist: Gaussian Processes on High-Dimensional Incomplete Grids

Mads Greisen Højlund
Department of Chemistry
Aarhus University
madsgh@chem.au.dk

August Smart Lykke-Møller
Department of Chemistry
Aarhus University
alm@chem.au.dk

Henry Moss
School of Mathematical Sciences
Lancaster University
henry.moss@lancaster.ac.uk

Ove Christiansen
Department of Chemistry
Aarhus University
ove@chem.au.dk

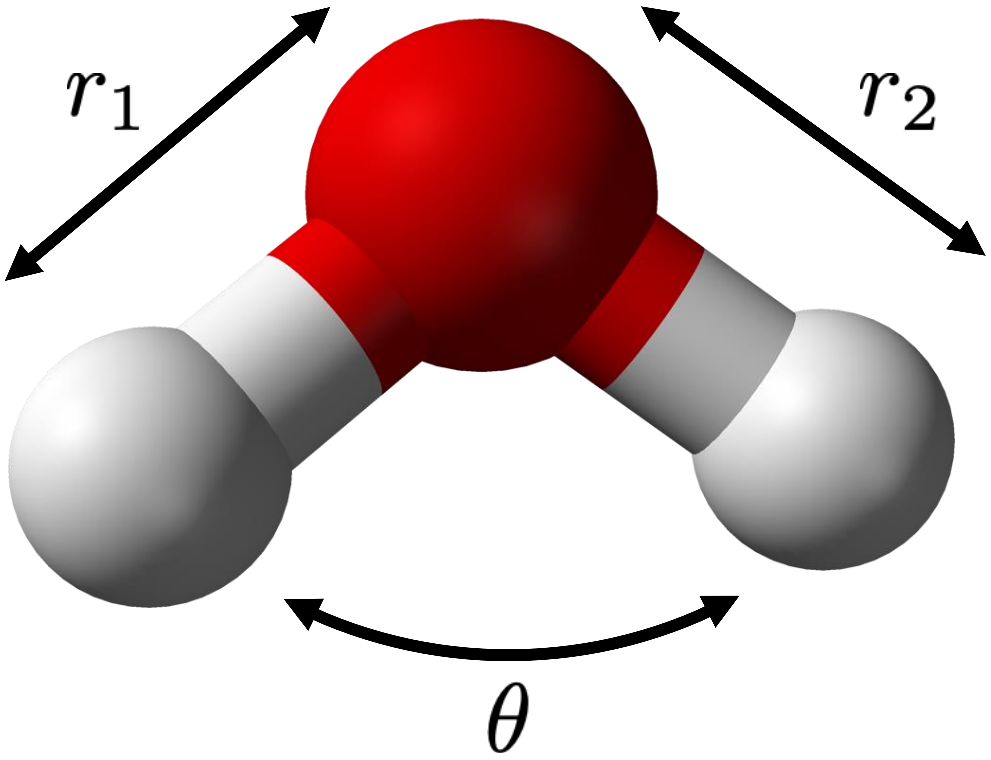
Submitted to NeurIPS 2026

New method: CUTS-GPR

Joint work with August

THE CURSE OF DIMENSIONALITY

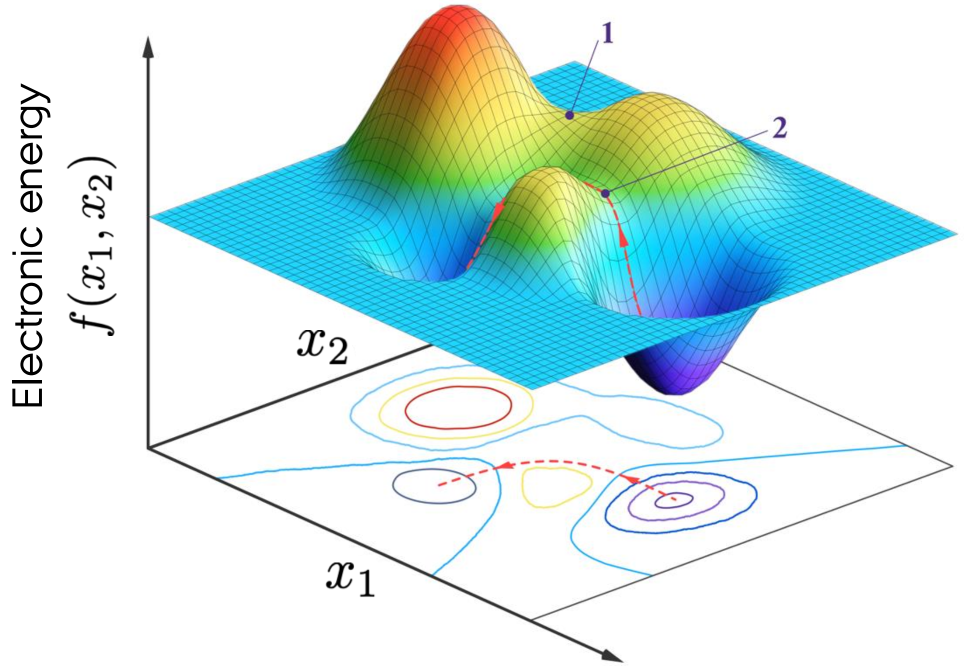
HIGH-DIM PROBLEMS ARE IMPORTANT



Example
Potential energy surface (PES)

$$f(r_1, r_2, \theta)$$

HIGH-DIM PROBLEMS ARE IMPORTANT



x_1, x_2 : Bond lengths or angles

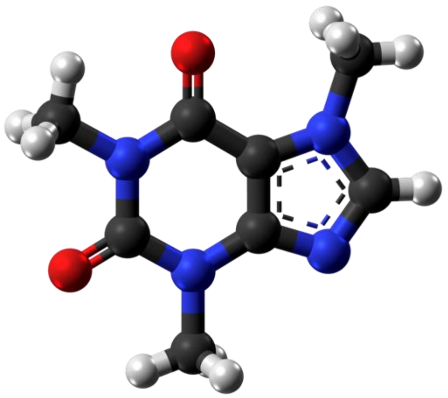
Chemistry is governed by potential energy surfaces (PESs)

Two challenges:

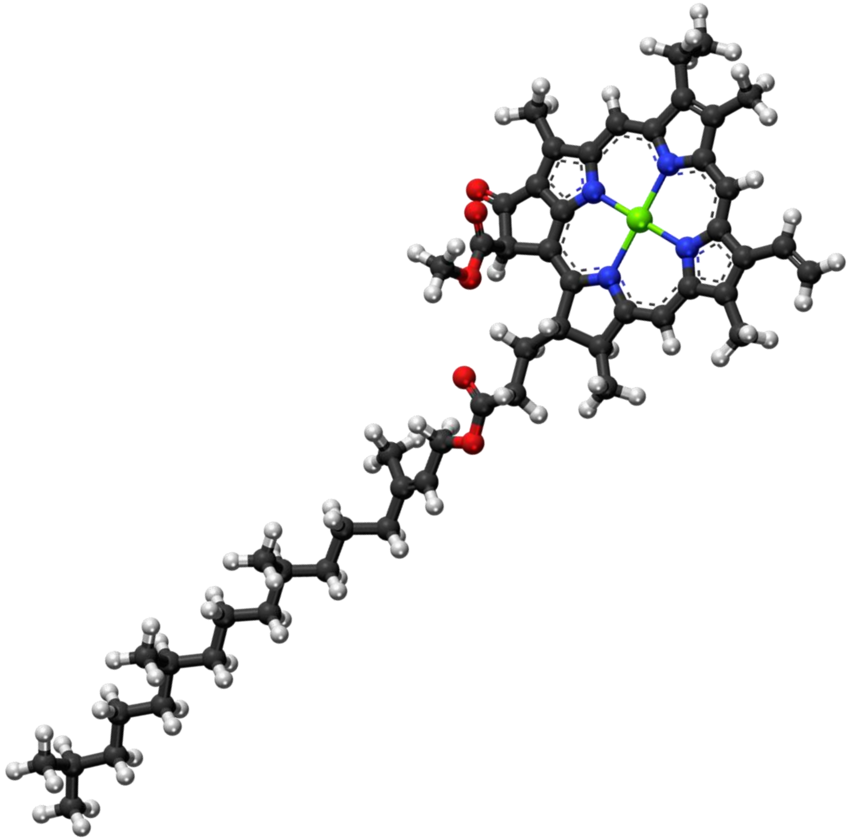
- Each point is expensive
- You need many points

$$D = 3N_{\mathrm{atom}} - 6$$

HIGH-DIM PROBLEMS ARE DIFFICULT



Caffeine
24 atoms
66 dimensions



Chlorophyll
137 atoms
405 dimensions

HIGH-DIM PROBLEMS ARE DIFFICULT

$$f : [0, 10]^D \to \mathbb{R} \longrightarrow V = 10^D$$

The volume to be explored/sampled grows exponentially!

The curse of dimensionality.

SIMPLIFYING ASSUMPTIONS

$f(x_1, \ldots, x_D)$ (not always tractable)

Class 1
Few dimensions are important
The intrinsic dimensionality is small
Low-dimensional model

Class 2 (my talk today)
All dimensions are important
The interaction order is small
Low-order model

INTERACTION ORDER DECOMPOSITION

$$f(\mathbf{x}) = f_0 + \underbrace{\sum_i f^{[i]}(x_i)}_{\text{Main effects terms}} + \underbrace{\sum_{i<j} f^{[ij]}(x_i, x_j)}_{\text{Second-order interaction terms}} + \underbrace{\sum_{i<j<k} f^{[ijk]}(x_i, x_j, x_k)}_{\text{Third-order interaction terms}} + \cdots + D\text{th order term}$$

Statistics
Analysis of variance (ANOVA)

LOW-ORDER MODELS

Example
Second-order model ($\alpha = 2$)

$$f(\mathbf{x}) \approx f_0 + \sum_i f^{[i]}(x_i) + \sum_{i<j} f^{[ij]}(x_i, x_j)$$

How to learn such a model using GPs?

LOW-ORDER MODELS

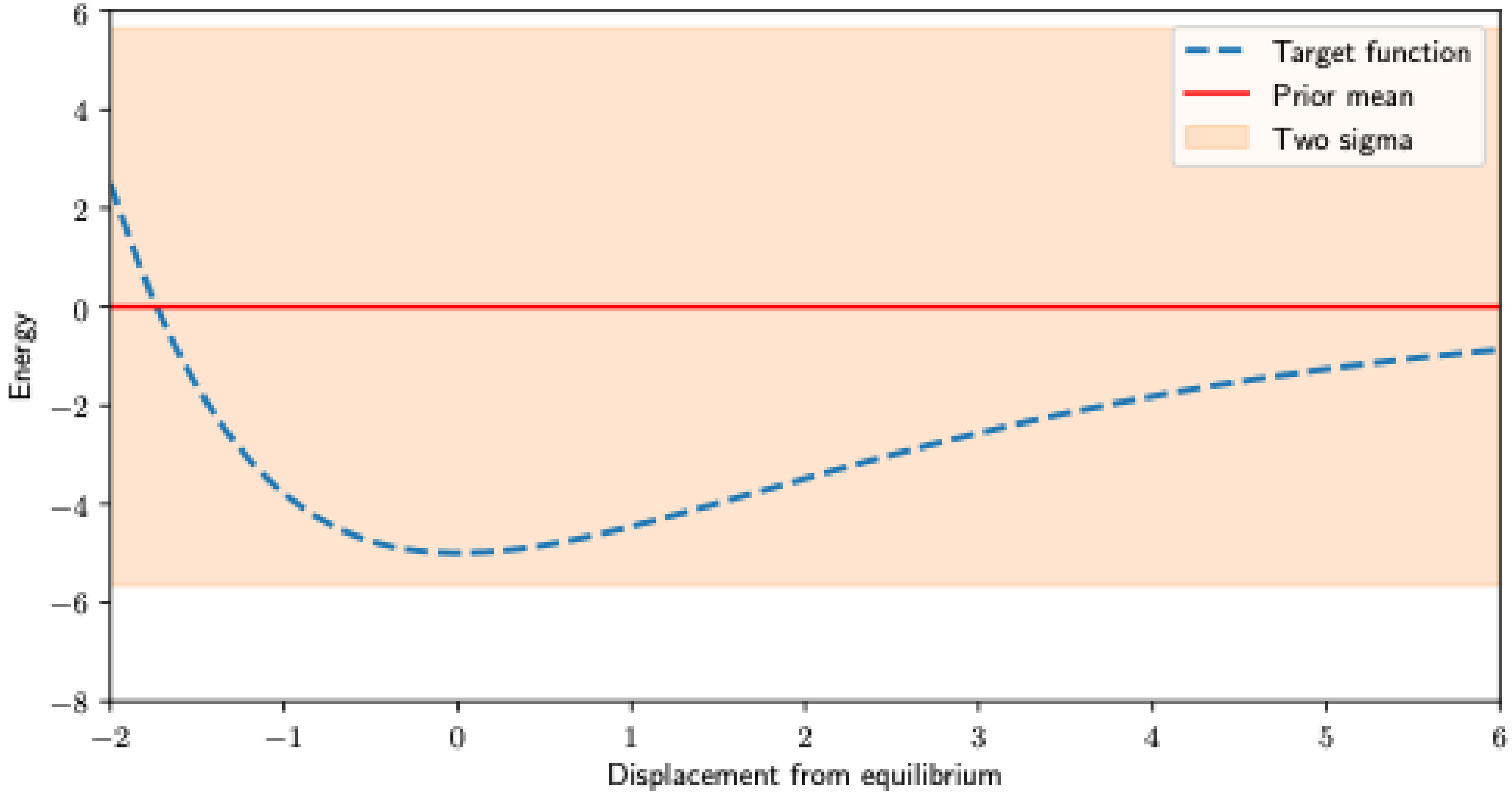
$$f(\mathbf{x}) \approx f_0 + \sum_i f^{[i]}(x_i) + \sum_{i<j} f^{[ij]}(x_i, x_j)$$

How to make a good model?
Choice of kernel
Data sampling

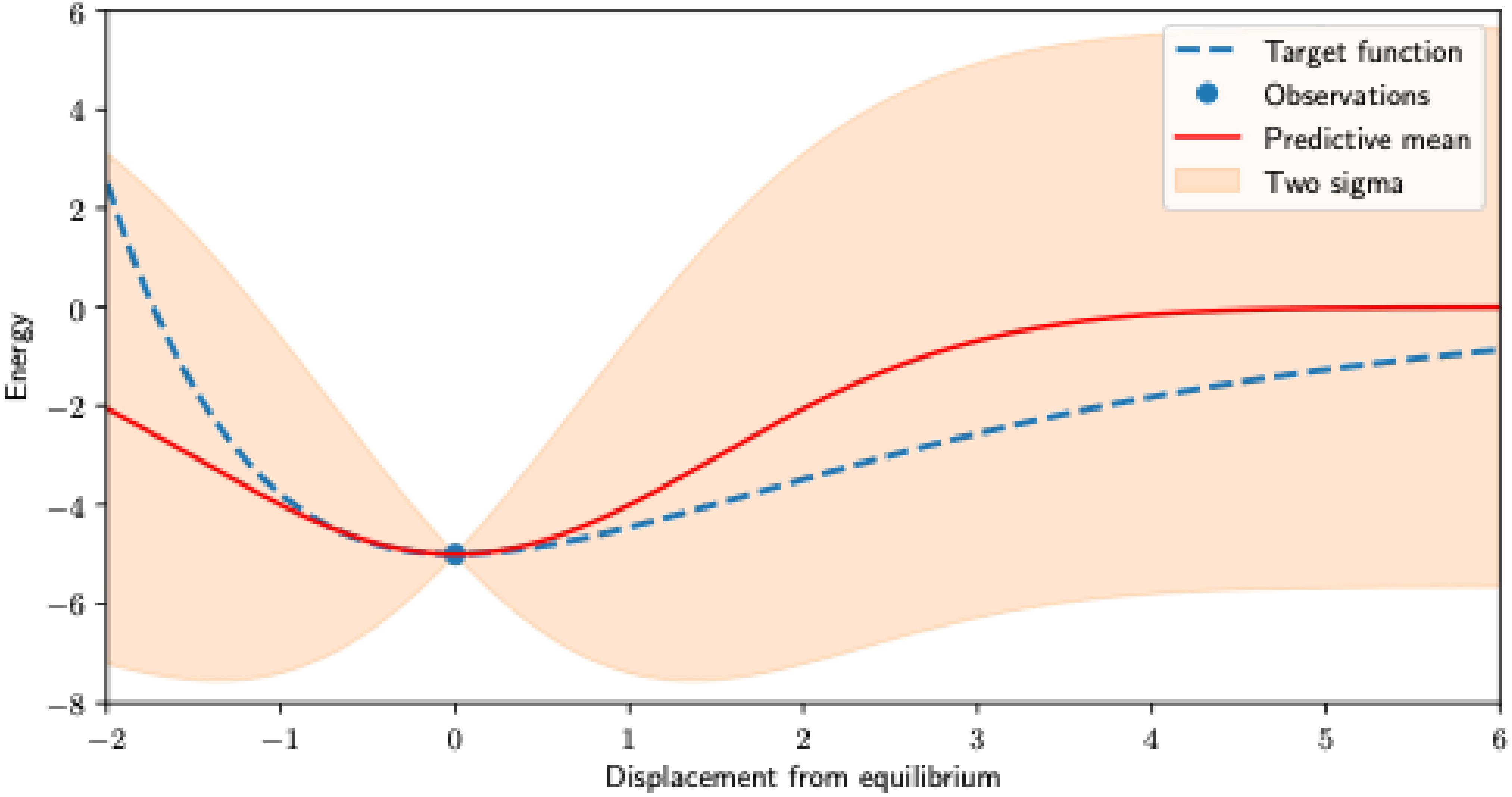
How to make an inexpensive model?
Memory usage
Computation time

GAUSSIAN PROCESS REGRESSION (GPR)

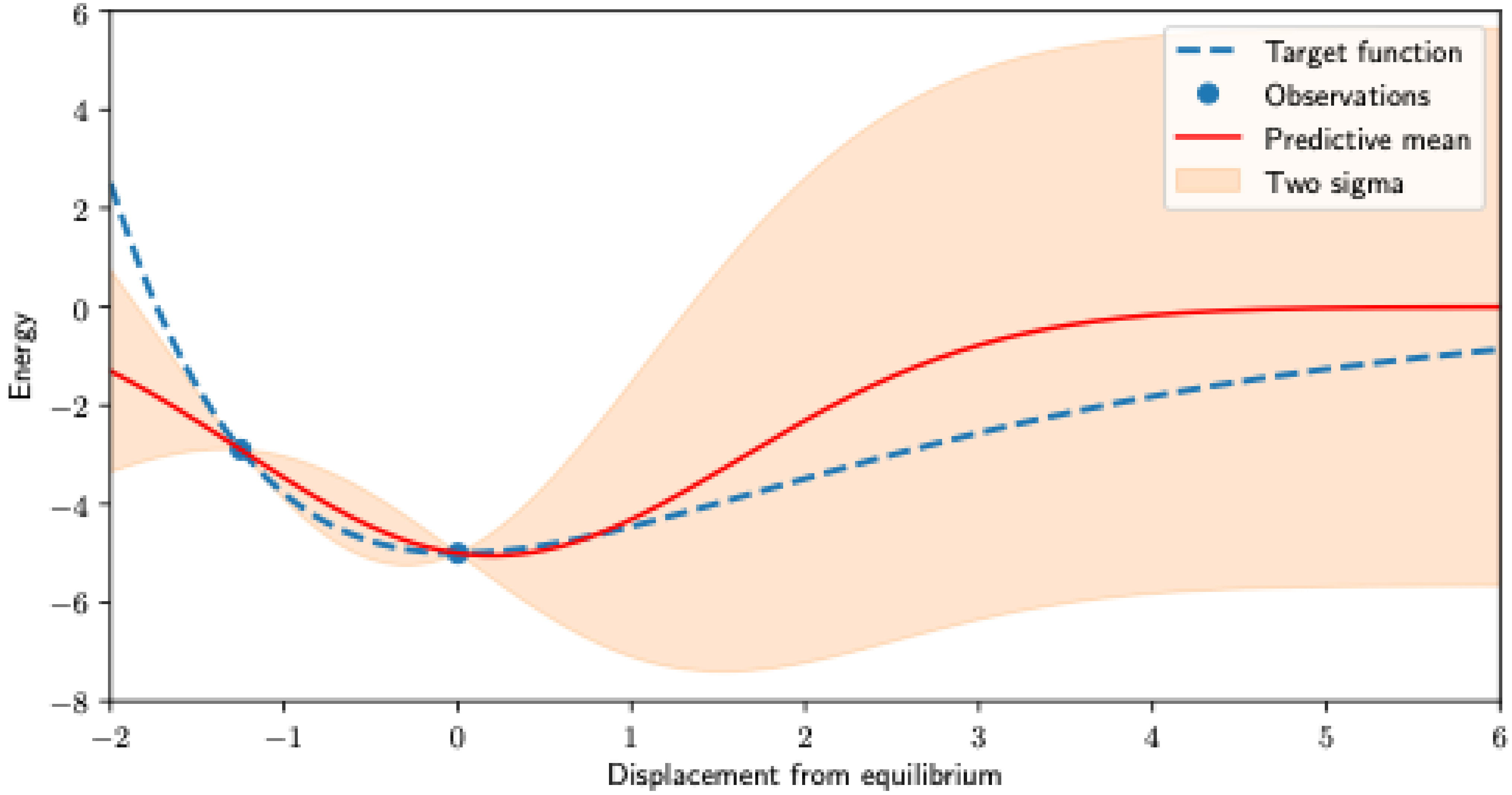
PRIOR BELIEFS: 0 OBSERVATIONS



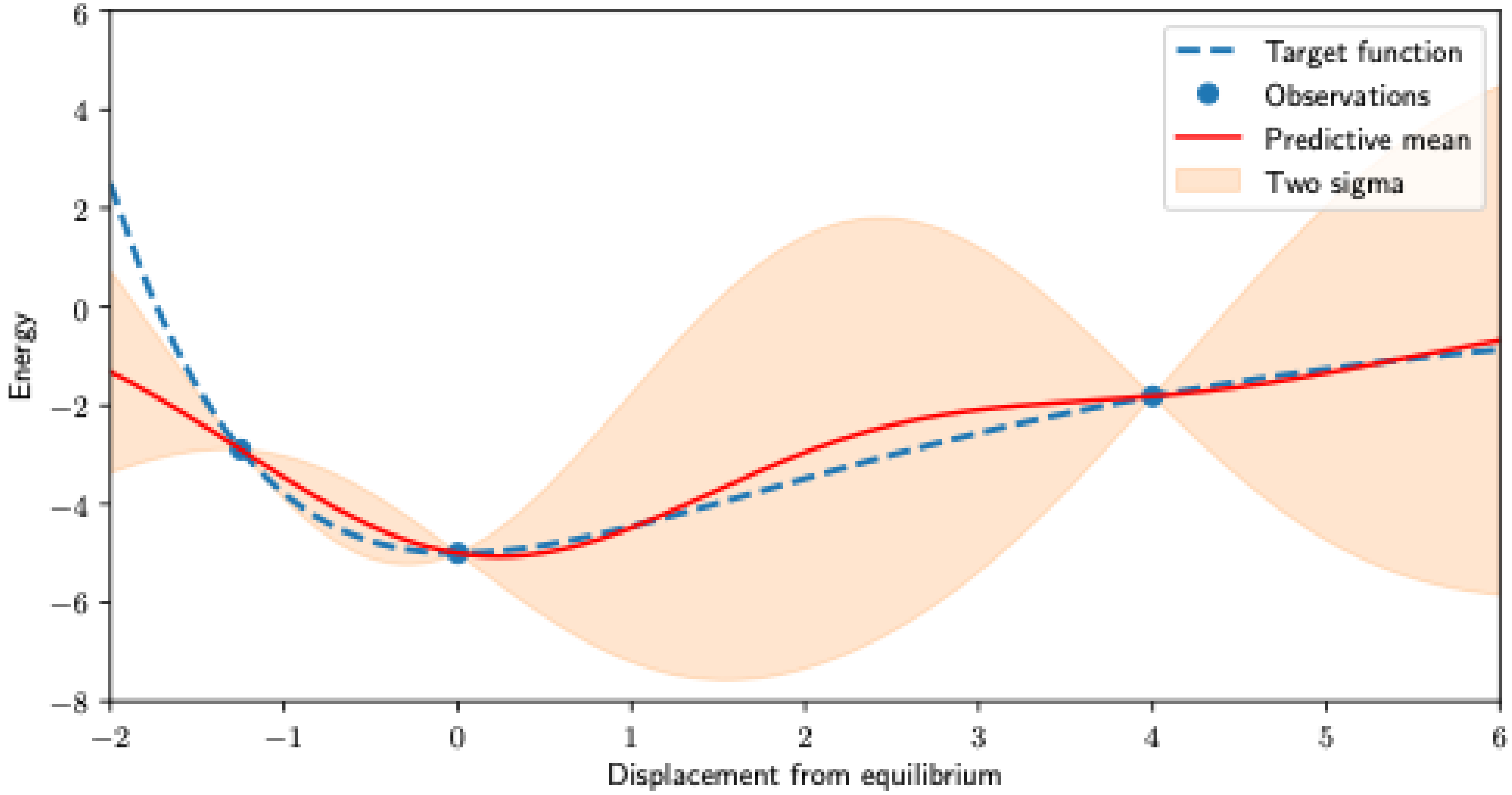
UPDATED BELIEFS: 1 OBSERVATION



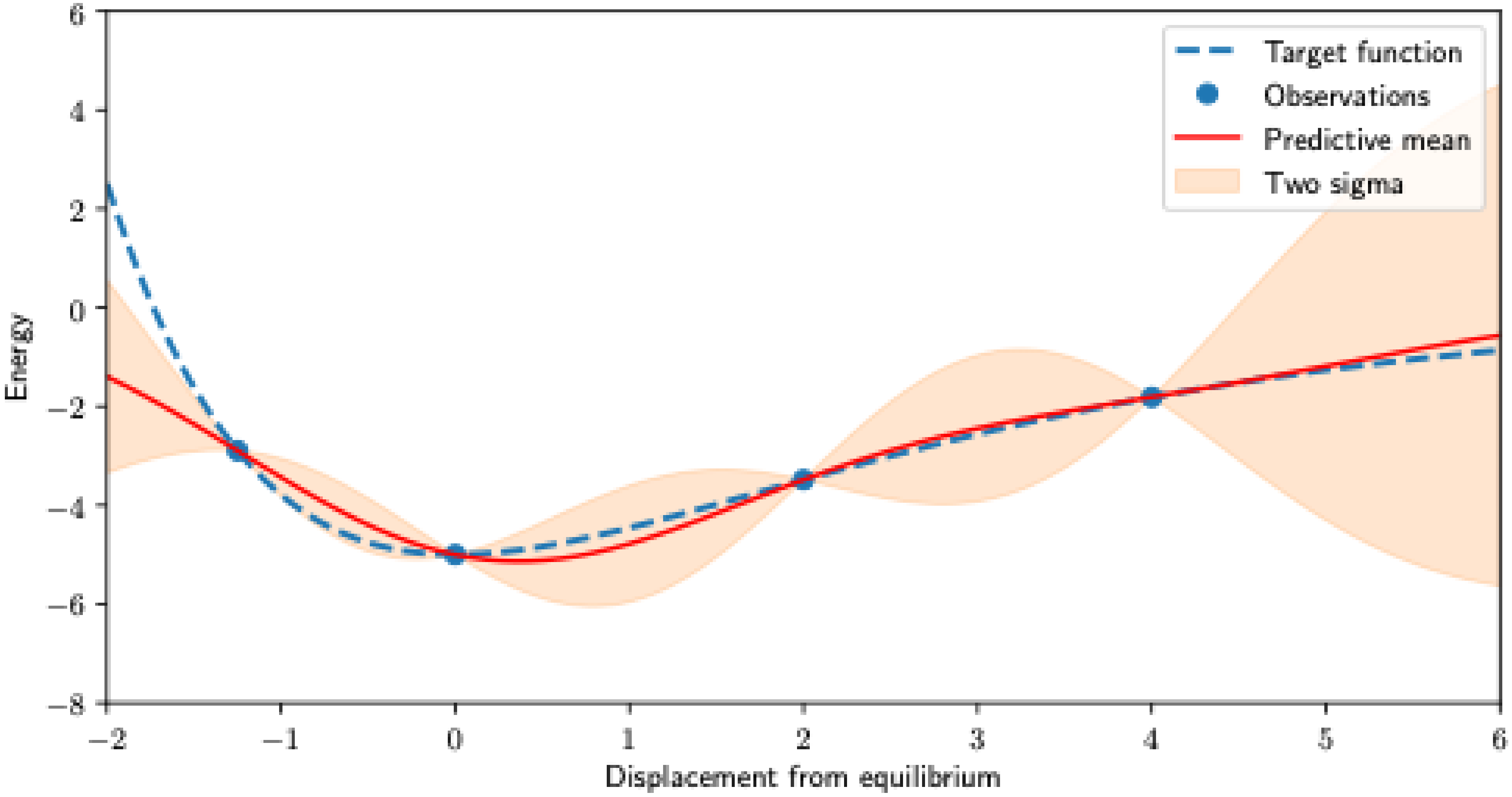
UPDATED BELIEFS: 2 OBSERVATIONS



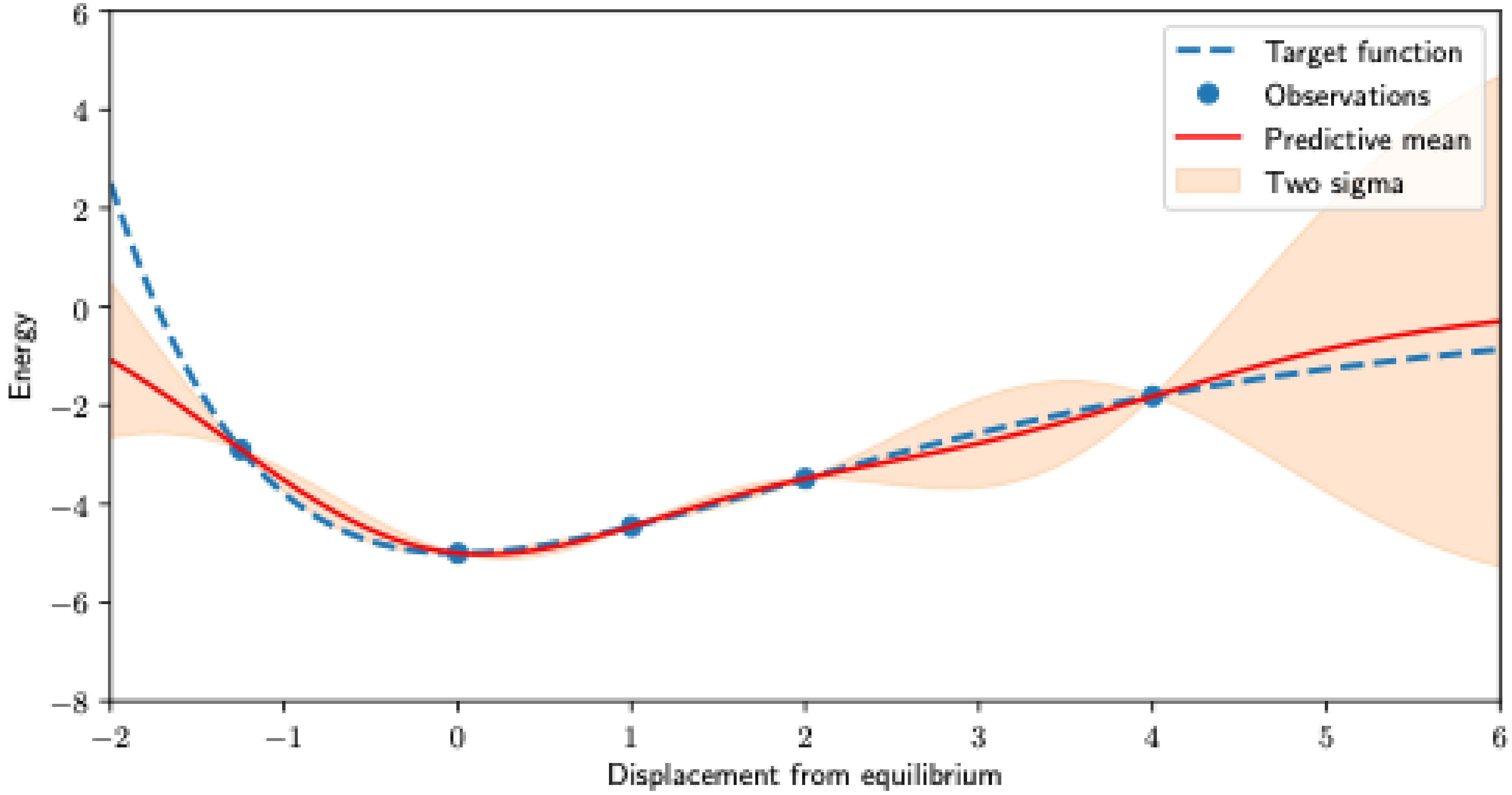
UPDATED BELIEFS: 3 OBSERVATIONS



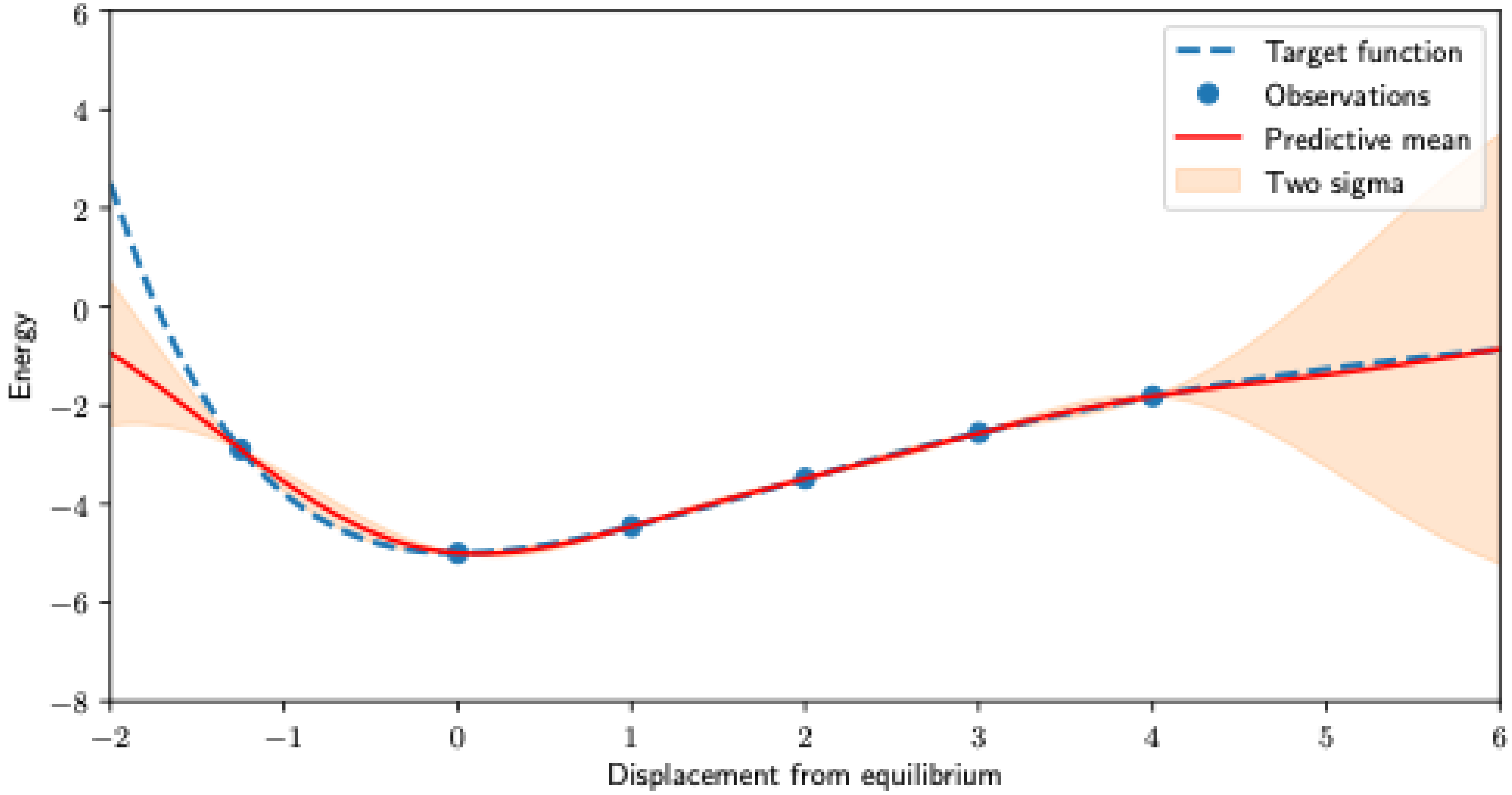
UPDATED BELIEFS: 4 OBSERVATIONS



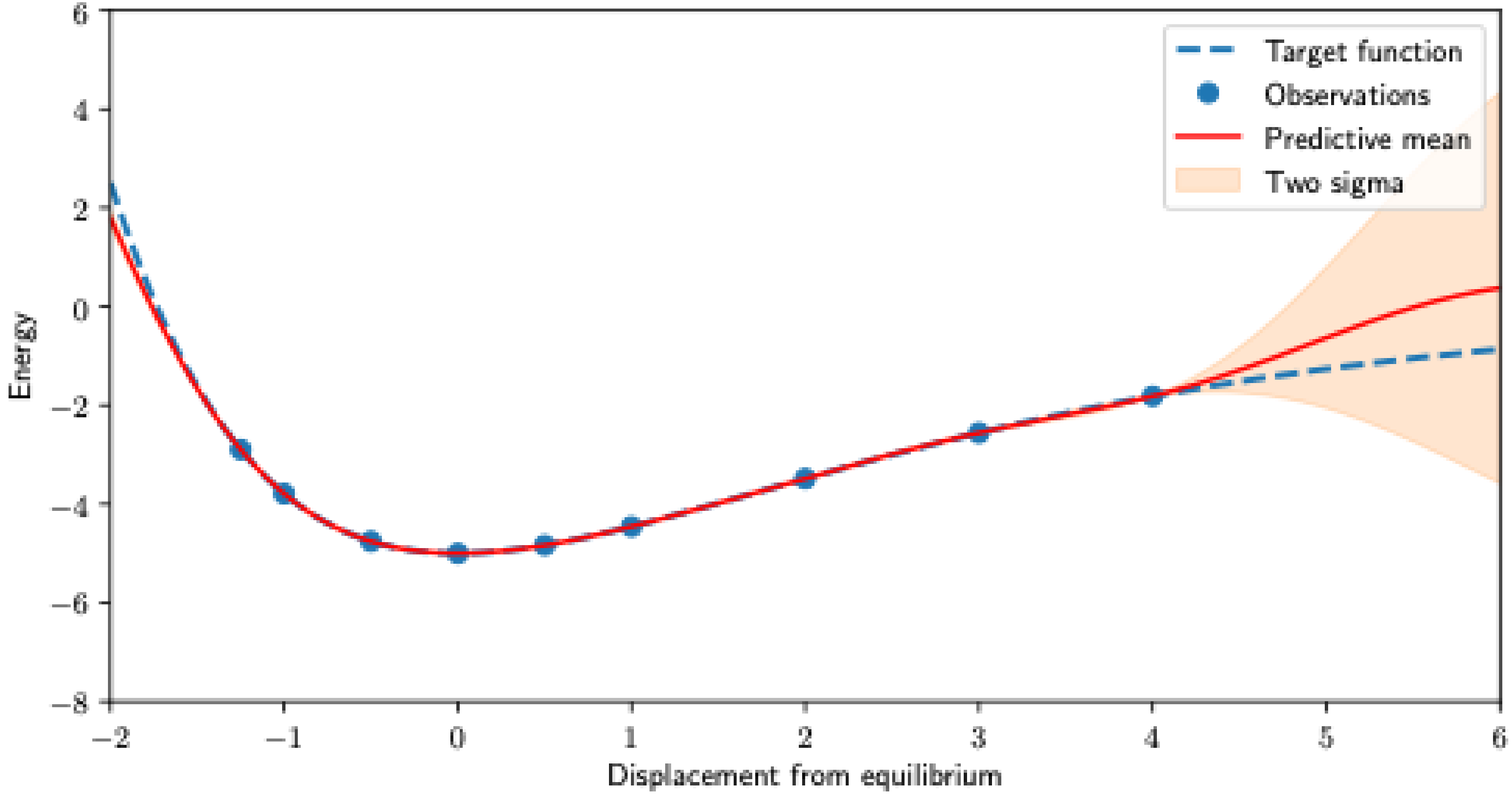
UPDATED BELIEFS: 5 OBSERVATIONS



UPDATED BELIEFS: 6 OBSERVATIONS



UPDATED BELIEFS: 9 OBSERVATIONS



DATA AND KERNEL

The inputs:

$$\mathcal{X} = \{\mathbf{x}_i\} = \{D\text{-dimensional training inputs}\}$$

$$\mathcal{X}_* = \{\mathbf{x}_i^*\} = \{D\text{-dimensional test inputs}\}$$

The kernel matrices:

$$\left.\begin{array}{ll} k(\mathcal{X},\mathcal{X}) = \mathbf{K} & (N \times N) \\ k(\mathcal{X},\mathcal{X}_*) = \mathbf{K}_* & (N \times N_*) \\ k(\mathcal{X}_*,\mathcal{X}_*) = \mathbf{K}_{**} & (N_* \times N_*) \end{array}\right\}$$

N training inputs ($\mathcal{X}$)
N_* test/prediction inputs ($\mathcal{X}_*$)

The noisy kernel matrix:

$$\mathbf{C} = \mathbf{K} + \sigma^2\mathbf{I}$$

σ^2 models observational noise and acts as a regularizer

WHAT TO COMPUTE

Predictions:

$$\boldsymbol{\mu} = \mathbf{K}_*^\mathsf{T} \underline{\mathbf{C}^{-1}\mathbf{y}} = \mathbf{K}_*^\mathsf{T} \boldsymbol{\alpha}$$

$$\boldsymbol{\Sigma} = \mathbf{K}_{**} - \mathbf{K}_*^\mathsf{T} \mathbf{C}^{-1} \mathbf{K}_*$$

Function values

Uncertainties

Depend on the kernel function, which depends on hyperparameters

Hyperparameter optimization:

$$\mathcal{L} = -\frac{1}{2}\left(\mathbf{y}^\mathsf{T}\boldsymbol{\alpha} + \underline{\log|\mathbf{C}|} + N \log 2\pi\right)$$

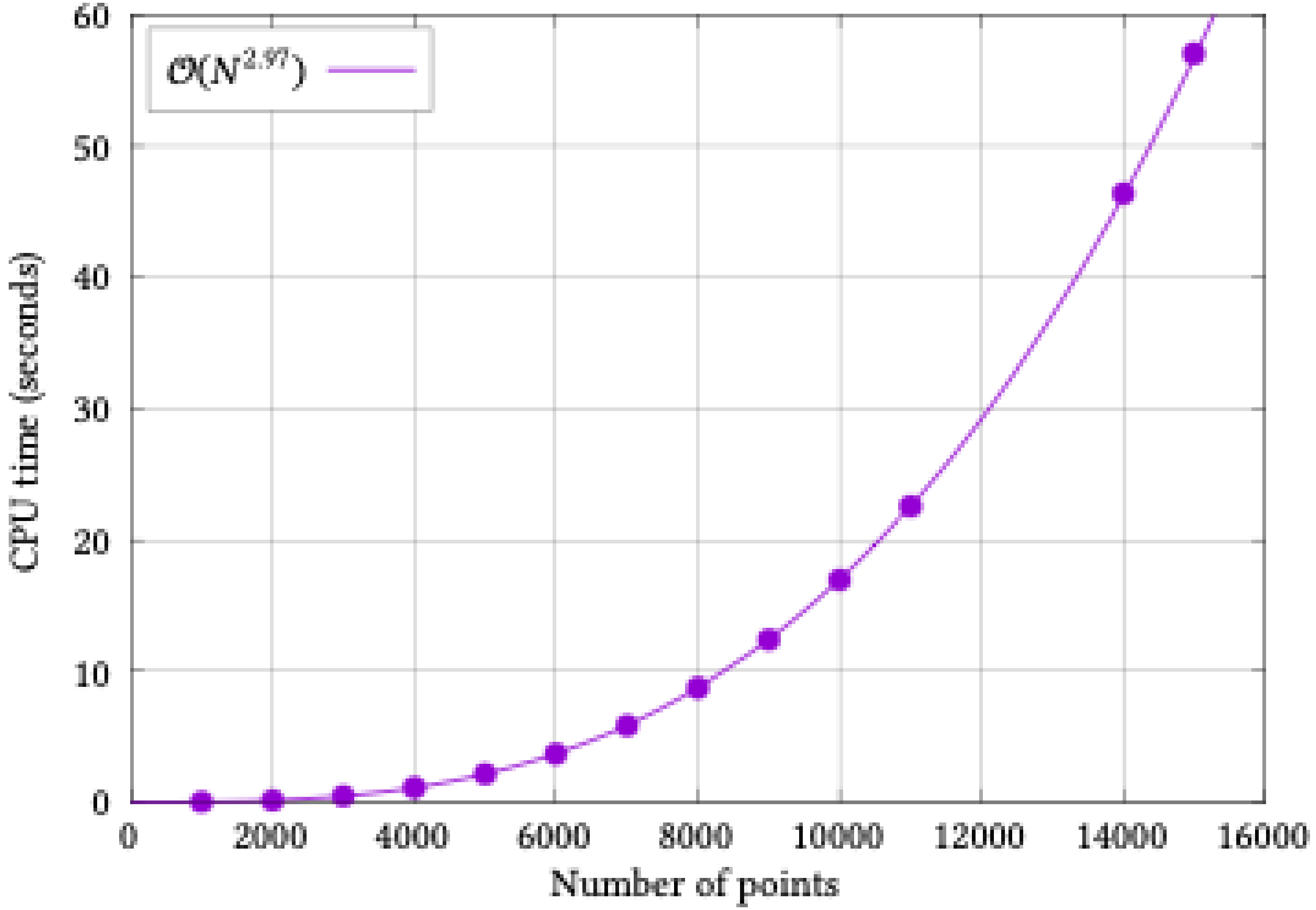
Marginal log-likelihood (MLL)

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{1}{2}\boldsymbol{\alpha}^\mathsf{T}\frac{\partial \mathbf{C}}{\partial \theta}\boldsymbol{\alpha} - \frac{1}{2}\underline{\mathrm{tr}\left(\mathbf{C}^{-1}\frac{\partial \mathbf{C}}{\partial \theta}\right)}$$

WHAT TO COMPUTE

Conventional strategy: $\mathcal{O}(N^3)$

Cholesky decomposition / benchmark



WHAT TO COMPUTE

Conventional strategy: $\mathcal{O}(N^3)$

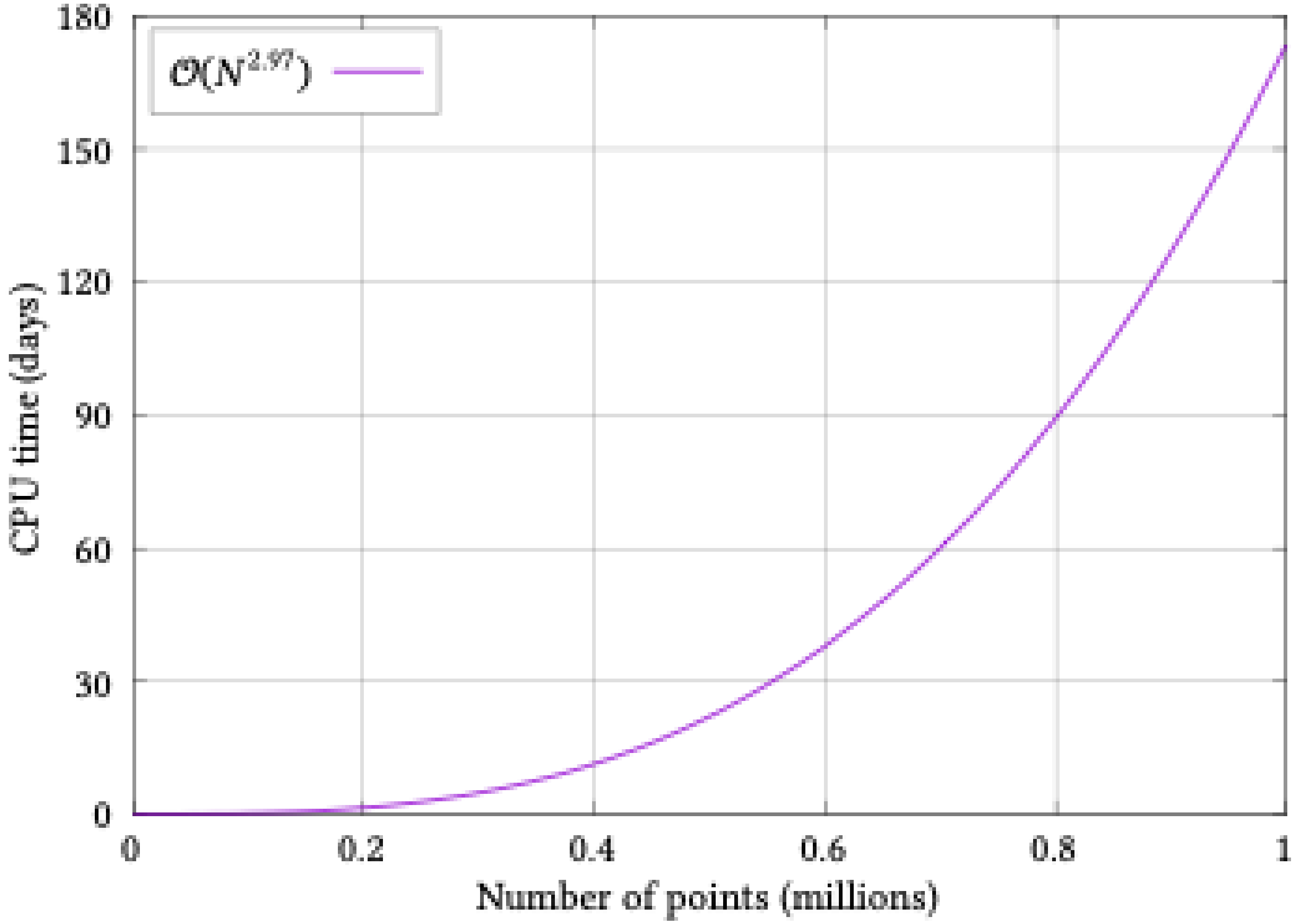
Extrapolated cost:

1 million points ≈ 173 days

1 million points ≈ 8 TB memory

How to reduce cost?

Cholesky decomposition / extrapolation from benchmark data



REDUCING COST WITH MATRIX-VECTOR PRODUCTS (MVPs)

GPs USING MVPs

$$\mathbf{C}^{-1}\mathbf{y}$$

Inverse-times-vector

↓

Conjugate gradient (CG)

$$\operatorname{tr}\left(\mathbf{C}^{-1}(\partial\mathbf{C}/\partial\theta)\right)$$

Trace terms

↓

Stochastic trace estimation (STE)

$$\log|\mathbf{C}|$$

Log-determinant

↓

Stochastic trace estimation (STE) + Lanczos

Everything can be computed using kernel MVPs!

GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration

Jacob R. Gardner*, Geoff Pleiss*,
David Bindel, Kilian Q. Weinberger, Andrew Gordon Wilson
Cornell University
{jrg365,kqw4,andrew}@cornell.edu,
{geoff,bindel}@cs.cornell.edu

CONJUGATE GRADIENT

Method for solving $\mathbf{Cx} = \mathbf{b}$ when $\mathbf{C}$ is SPD

The algorithm (basically):

- Initial guess: $\mathbf{x}_0$
- Initial residual: $\mathbf{r}_0 = ||\mathbf{C}\mathbf{x}_0 - \mathbf{b}||$
- Iteratively update guess:

$$\left.\begin{aligned} \mathbf{x}_k &= \mathbf{x}_{k-1} + \mathbf{update} \\ \mathbf{r}_k &= ||\mathbf{C}\mathbf{x}_k - \mathbf{b}|| \end{aligned}\right\} \longleftarrow$$

Each iteration
One kernel MVP + a few dot products

STOCHASTIC TRACE ESTIMATION

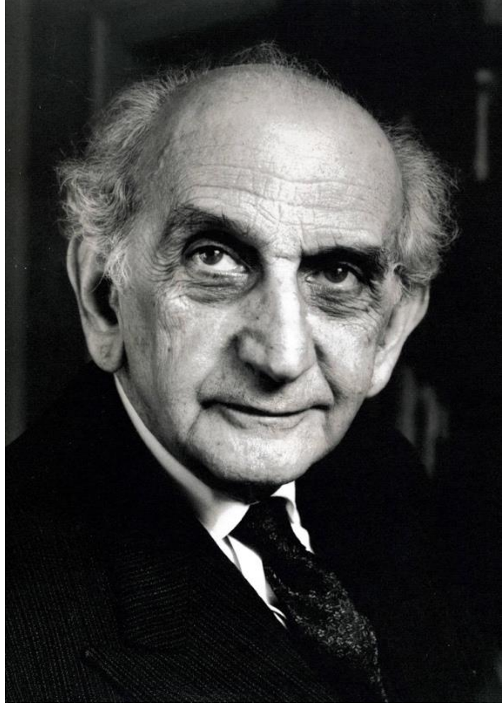
Method for estimating $\mathrm{tr}(\mathbf{A})$

The algorithm:

- Isotropic random vector: $\mathbb{E}[\mathbf{b}\mathbf{b}^\mathsf{T}] = \mathbf{I}$
- Estimate trace: $\mathrm{tr}(\mathbf{A}) \approx \frac{1}{m}\sum_{i=1}^{m} \mathbf{b}_i^\mathsf{T}\mathbf{A}\mathbf{b}_i$
- In our case: $\mathrm{tr}\left(\mathbf{C}^{-1}\frac{\partial \mathbf{C}}{\partial \theta}\right) \approx \frac{1}{m}\sum_{i=1}^{m} \mathbf{b}_i^\mathsf{T}\mathbf{C}^{-1}\frac{\partial \mathbf{C}}{\partial \theta}\mathbf{b}_i$ ⟵ Use CG for inverse-times-probe

LANCZOS

Method for approximating functions of Hermitian matrices



Cornelius Lanczos
1893–1974

$$\log|\mathbf{C}| = \mathrm{tr}(\log \mathbf{C}) \qquad \text{(rewrite)}$$

$$\mathrm{tr}(\log \mathbf{C}) \approx \frac{1}{m}\sum_{i=1}^{m} \mathbf{b}_i^\mathsf{T} \log(\mathbf{C})\mathbf{b}_i \qquad \text{(trace estimation)}$$

$$\mathbf{b}_i^\mathsf{T} \log(\mathbf{C})\mathbf{b}_i \approx ||\mathbf{b}_i||^2 \mathbf{e}_0^\mathsf{T} \log(\mathbf{T}_{i\ell})\mathbf{e}_0 \qquad \text{(Lanczos)}$$

$$\log|\mathbf{C}| \approx \frac{1}{m}\sum^{m} ||\mathbf{b}_i||^2 \mathbf{e}_0^\mathsf{T} \log(\mathbf{T}_{i\ell})\mathbf{e}_0 \qquad \text{(result)}$$

$\ell \times \ell$ tridiagonal matrix obtained by ℓ iterations of Lanczos — or modified CG.

NUMERICALLY EXACT GPs USING MVPs

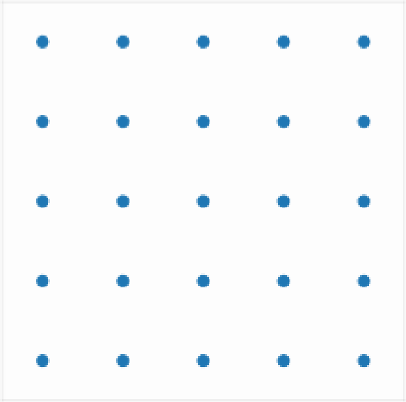
Kernel MVP → Black-box engine → Gaussian process

Should be fast and scalable!

REDUCING COST WITH STRUCTURED KERNELS

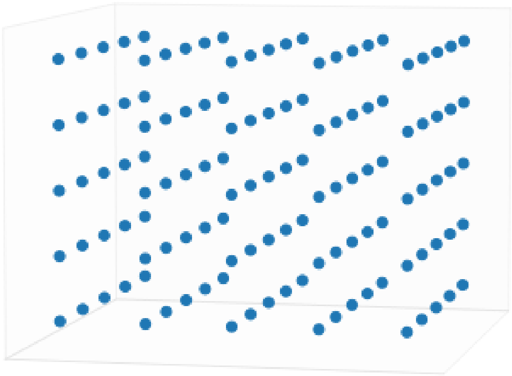
COMPLETE GRIDS

$n = 5$ grid points along each dimension



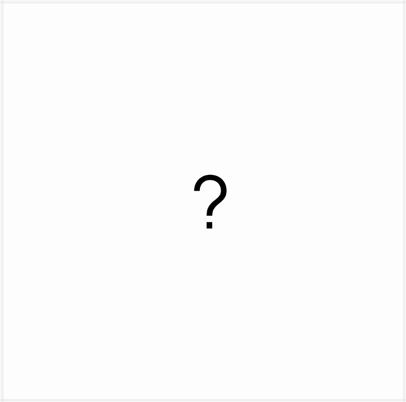
$D = 2$

$N = n^2 = 25$



$D = 3$

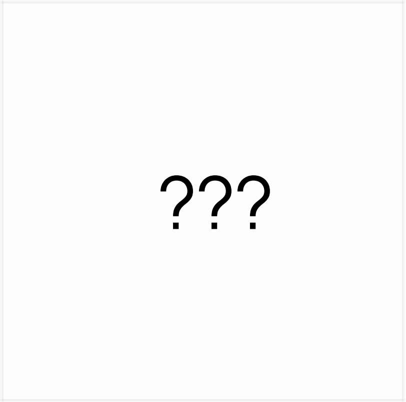
$N = n^3 = 125$



$D = 4$

$N = n^4 = 625$

. . . .



D

$N = n^D = \text{huge number}$

COMPLETE GRIDS

Separable kernel ...

$$k(\mathbf{x}, \tilde{\mathbf{x}}) = \exp\left(-\frac{\|\mathbf{x} - \tilde{\mathbf{x}}\|^2}{2\ell^2}\right) = \prod_{d=1}^{D} \exp\left(-\frac{|x^{(d)} - \tilde{x}^{(d)}|^2}{2\ell^2}\right) = \prod_{d=1}^{D} k^{(d)}$$

... plus complete grid:

$$\mathbf{K} = \underbrace{\mathbf{K}^{(1)} \otimes \mathbf{K}^{(2)} \otimes \cdots \otimes \mathbf{K}^{(D)}}$$

Kronecker product!

D factors (one for each dimension)

$\mathbf{K}^{(m)}$ is $(n \times n)$ (a small matrix)

Saatci (2011); Wilson et al. (2014); Gilboa et al. (2015); Flaxman et al. (2015)

KRONECKER PRODUCTS

$$\mathbf{A} \otimes \mathbf{B} \otimes \mathbf{C} = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \otimes \begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \end{bmatrix} \otimes \begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix}$$

Generally (3D): $(\mathbf{A} \otimes \mathbf{B} \otimes \mathbf{C})_{(pqr)(\bar{p}\bar{q}\bar{r})} = a_{p\bar{p}} b_{q\bar{q}} c_{r\bar{r}}$

$$= \begin{array}{cc} & \begin{array}{cccccccc} 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \end{array} \\ \begin{array}{c} 000 \\ 001 \\ 010 \\ 011 \\ 100 \\ 101 \\ 110 \\ 111 \end{array} & \begin{bmatrix}
a_{00}b_{00}c_{00} & a_{00}b_{00}c_{01} & a_{00}b_{01}c_{00} & a_{00}b_{01}c_{01} & a_{01}b_{00}c_{00} & a_{01}b_{00}c_{01} & a_{01}b_{01}c_{00} & a_{01}b_{01}c_{01} \\
a_{00}b_{00}c_{10} & a_{00}b_{00}c_{11} & a_{00}b_{01}c_{10} & a_{00}b_{01}c_{11} & a_{01}b_{00}c_{10} & a_{01}b_{00}c_{11} & a_{01}b_{01}c_{10} & a_{01}b_{01}c_{11} \\
a_{00}b_{10}c_{00} & a_{00}b_{10}c_{01} & a_{00}b_{11}c_{00} & a_{00}b_{11}c_{01} & a_{01}b_{10}c_{00} & a_{01}b_{10}c_{01} & a_{01}b_{11}c_{00} & a_{01}b_{11}c_{01} \\
a_{00}b_{10}c_{10} & a_{00}b_{10}c_{11} & a_{00}b_{11}c_{10} & a_{00}b_{11}c_{11} & a_{01}b_{10}c_{10} & a_{01}b_{10}c_{11} & a_{01}b_{11}c_{10} & a_{01}b_{11}c_{11} \\
a_{10}b_{00}c_{00} & a_{10}b_{00}c_{01} & a_{10}b_{01}c_{00} & a_{10}b_{01}c_{01} & a_{11}b_{00}c_{00} & a_{11}b_{00}c_{01} & a_{11}b_{01}c_{00} & a_{11}b_{01}c_{01} \\
a_{10}b_{00}c_{10} & a_{10}b_{00}c_{11} & a_{10}b_{01}c_{10} & a_{10}b_{01}c_{11} & a_{11}b_{00}c_{10} & a_{11}b_{00}c_{11} & a_{11}b_{01}c_{10} & a_{11}b_{01}c_{11} \\
a_{10}b_{10}c_{00} & a_{10}b_{10}c_{01} & a_{10}b_{11}c_{00} & a_{10}b_{11}c_{01} & a_{11}b_{10}c_{00} & a_{11}b_{10}c_{01} & a_{11}b_{11}c_{00} & a_{11}b_{11}c_{01} \\
a_{10}b_{10}c_{10} & a_{10}b_{10}c_{11} & a_{10}b_{11}c_{10} & a_{10}b_{11}c_{11} & a_{11}b_{10}c_{10} & a_{11}b_{10}c_{11} & a_{11}b_{11}c_{10} & a_{11}b_{11}c_{11}
\end{bmatrix} \end{array}$$

KRONECKER PRODUCTS

Fast matrix-vector product (MVP):

$$\mathbf{K}\mathbf{v} = \left(\mathbf{K}^{(1)} \otimes \mathbf{K}^{(2)} \otimes \cdots \otimes \mathbf{K}^{(D)}\right)\mathbf{v}$$

$$= \left(\mathbf{K}^{(1)} \otimes \mathbf{I}^{(2)} \otimes \cdots \otimes \mathbf{I}^{(D)}\right)\left(\mathbf{I}^{(1)} \otimes \mathbf{K}^{(2)} \otimes \cdots \otimes \mathbf{I}^{(D)}\right) \cdots \left(\mathbf{I}^{(1)} \otimes \mathbf{I}^{(2)} \otimes \cdots \otimes \mathbf{K}^{(D)}\right)\mathbf{v}$$

$$= \underbrace{\left(\mathbf{K}^{[1]}\mathbf{K}^{[2]} \cdots \mathbf{K}^{[D]}\right)}\mathbf{v}$$

Sequence of bracket matrices (nice!) → Complexity $\mathcal{O}(nDN)$ (near-linear scaling with N)

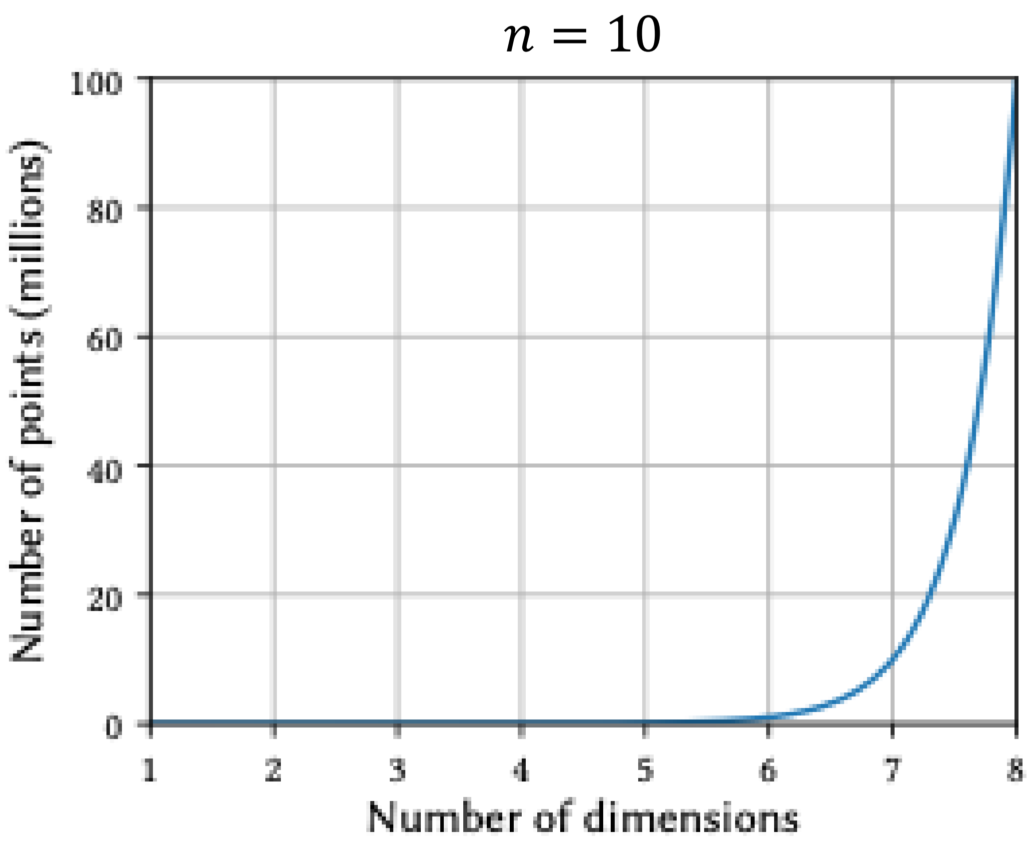
$$v'_{i_1 \ldots i_m \ldots i_D} = \sum_{j_m} K^{(m)}_{i_m j_m} v_{i_1 \ldots j_m \ldots i_D}$$

Bracket matrix multiplication *aka* one-index tensor contraction

PROBLEMS

- Near-linear scaling with N, but exponential scaling with D

- The separable kernel



How to address both at the same time?

CHOOSING THE RIGHT KERNEL

The low-order model …

$$f(x_1, \ldots, x_D) = f_0 + \sum_i f^{[i]}(x_i) + \sum_{i<j} f^{[ij]}(x_i, x_j) + \cdots + \alpha\text{th order terms}$$

… corresponds to an additive kernel:

$$k = \sigma_0^2 + \sigma_1^2 \sum_i k^{(i)} + \sigma_2^2 \underbrace{\sum_{i<j} k^{(i)} k^{(j)}}_{\text{Each term is a product of 1D kernels}} + \cdots + \alpha\text{th order terms}$$

Duvenaud et al. (2011); Durrange et al. (2012); Lu et al. (2022); Ishida et al. (2025)

ADDITIVE KERNELS ON GRIDS

We start by evaluating the kernel ...

$$k = \sigma_0^2 + \sigma_1^2 \sum_i k^{(i)} + \sigma_2^2 \sum_{i<j} k^{(i)} k^{(j)} + \cdots$$

$$\mathbf{J}^{(m)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \quad (n = 4)$$

... on a complete grid:

$$k^{(i)} \longrightarrow \mathbf{J}^{(1)} \otimes \cdots \otimes \mathbf{K}^{(i)} \otimes \cdots \otimes \mathbf{J}^{(j)} \otimes \cdots \otimes \mathbf{J}^{(D)} = \mathbf{K}^{[i]} \left(\prod_{k \neq i} \mathbf{J}^{[k]} \right) = \mathbf{K}^{[i]} \mathbf{J}^{[\neg i]}$$

$$k^{(i)} k^{(j)} \longrightarrow \mathbf{J}^{(1)} \otimes \cdots \otimes \mathbf{K}^{(i)} \otimes \cdots \otimes \mathbf{K}^{(j)} \otimes \cdots \otimes \mathbf{J}^{(D)} = \mathbf{K}^{[i]} \mathbf{K}^{[j]} \left(\prod_{k \notin \{i,j\}} \mathbf{J}^{[k]} \right) = \mathbf{K}^{[ij]} \mathbf{J}^{[\neg ij]}$$

$$\prod_{i \in \mathbf{m}} k^{(i)} \longrightarrow \quad = \left(\prod_{i \in \mathbf{m}} \mathbf{K}^{[i]} \right) \left(\prod_{k \notin \mathbf{m}} \mathbf{J}^{[k]} \right) = \mathbf{K}^{[\mathbf{m}]} \mathbf{J}^{[\neg \mathbf{m}]}$$

ADDITIVE KERNELS ON GRIDS

Summing up everything:

$$\mathbf{K} = \sum_{\mathbf{m}} \sigma^2_{|\mathbf{m}|} \mathbf{K}^{[\mathbf{m}]} \mathbf{J}^{[\neg \mathbf{m}]} = \sum_{\mathbf{m}} \sigma^2_{|\mathbf{m}|} \underbrace{\left(\prod_{m \in \mathbf{m}} \mathbf{K}^{[m]} \right)}_{\text{Few (at most } \alpha)} \underbrace{\left(\prod_{m \notin \mathbf{m}} \mathbf{J}^{[m]} \right)}_{\text{Many (at least } D - \alpha)}$$

Sum of Kronecker products
Ishida et al. (2025)

Few (at most α)

Many (at least $D - \alpha$)

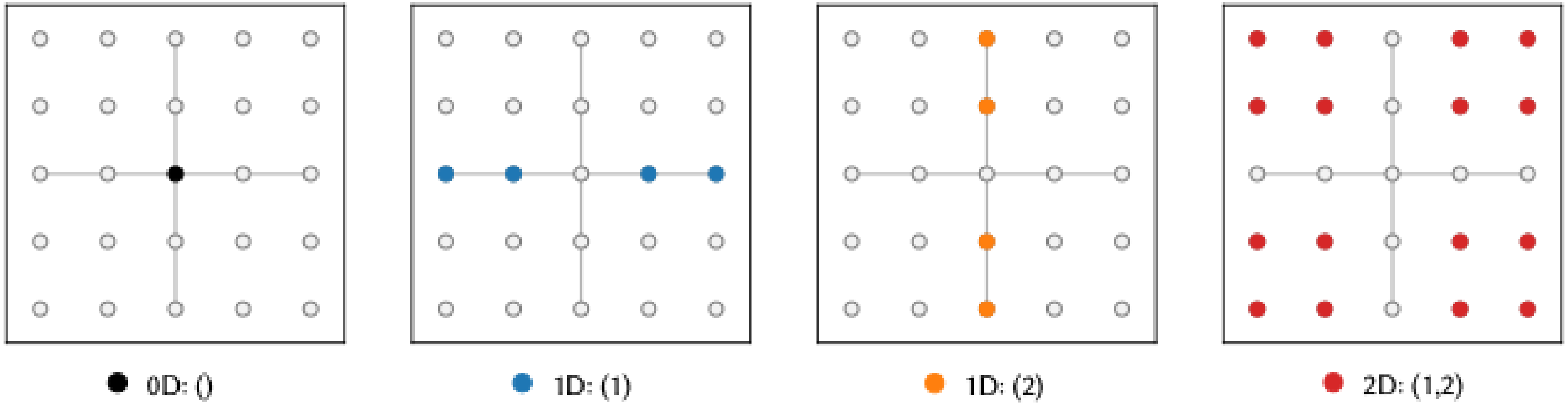
$$\mathbf{J}^{(m)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \quad (n = 4)$$

INCOMPLETE GRIDS

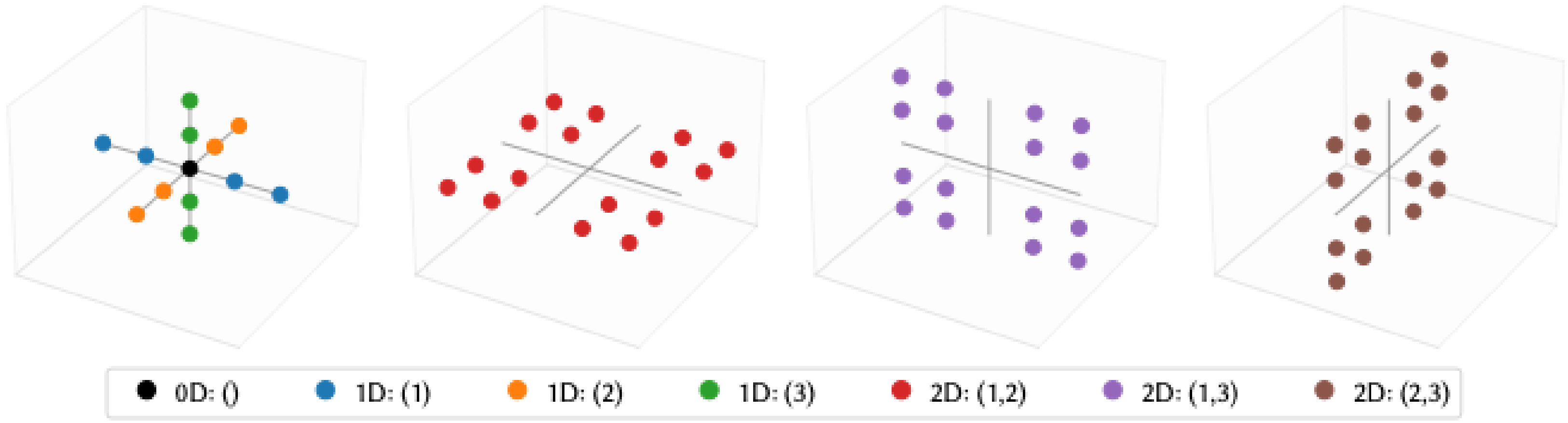
We need an data structure compatible with

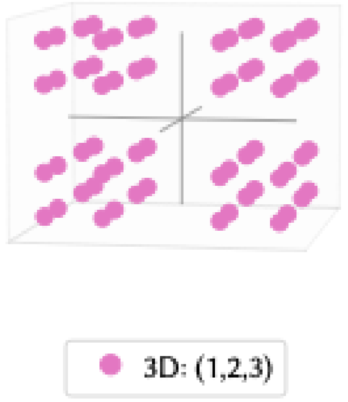
$$f(x_1,\ldots,x_D) = f_0 + \sum_i f^{[i]}(x_i) + \sum_{i<j} f^{[ij]}(x_i,x_j) + \cdots$$

Sub-functions $\leftrightarrow$ sub-grids/cuts



INCOMPLETE GRIDS





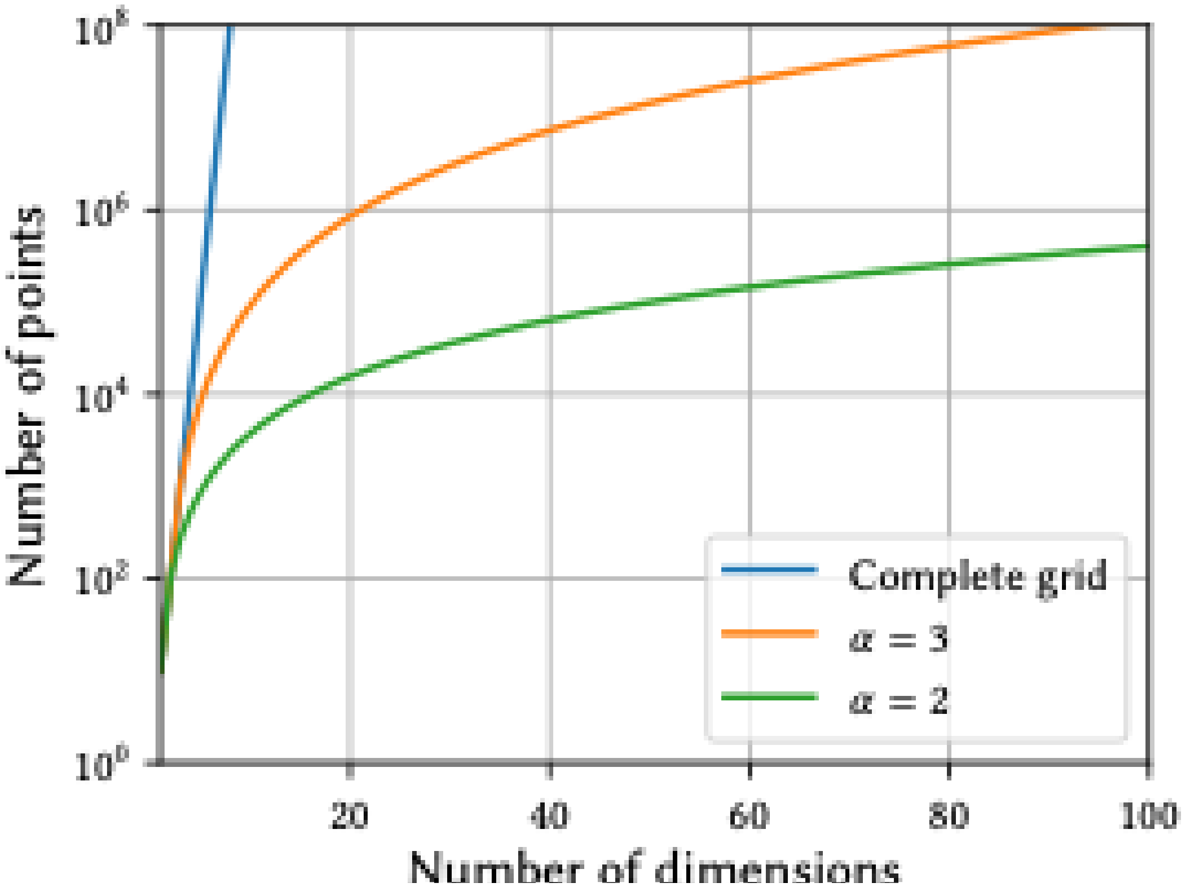
Include low-order cuts (up to order α)

Omit higher-order cuts

INCOMPLETE GRIDS

Complete grid: $N = n^D$
Exponential scaling

Incomplete grid: $\hat{N} = \mathcal{O}(n^\alpha D^\alpha / \alpha!) \sim \mathcal{O}(D^\alpha)$
Polynomial scaling



100 M

1 M

$n = 10$ grid points per dimension

KERNELS ON INCOMPLETE GRIDS

Complete grid + additive kernel:

$$\mathbf{K} = \sum_{\mathbf{m}} \sigma_{|\mathbf{m}|}^2 \mathbf{K}^{[\mathbf{m}]} \mathbf{J}^{[\neg \mathbf{m}]} = \sum_{\mathbf{m}} \sigma_{|\mathbf{m}|}^2 \left(\prod_{m \in \mathbf{m}} \mathbf{K}^{[m]} \right) \left(\prod_{m \notin \mathbf{m}} \mathbf{J}^{[m]} \right) \in \mathbb{R}^{N \times N}$$

Sum of Kronecker products

Incomplete grid + additive kernel:

$$\widehat{\mathbf{K}} = \mathbf{\Gamma}^\mathsf{T} \mathbf{K} \mathbf{\Gamma} = \sum_{\mathbf{m}} \sigma_{|\mathbf{m}|}^2 \mathbf{\Gamma}^\mathsf{T} (\mathbf{K}^{[\mathbf{m}]} \mathbf{J}^{[\neg \mathbf{m}]}) \mathbf{\Gamma} \in \mathbb{R}^{\widehat{N} \times \widehat{N}}$$

Sum of chopped Kronecker products

Chopping matrix

$\mathbf{\Gamma} \in \mathbb{R}^{N \times \widehat{N}}$

Number of terms is $\mathcal{O}(D^\alpha)$

Wodraszka and Carrington (2019); Zak and Carrington (2019)
Simmons and Carrington (2023); Holzmüller and Pflüger (2021)

CHOPPED KRONECKER PRODUCTS

$$\mathbf{A}\otimes\mathbf{B}\otimes\mathbf{C} = \begin{array}{cc} & \begin{array}{cccccccc} 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \end{array} \\ \begin{array}{c} 000 \\ 001 \\ 010 \\ 011 \\ 100 \\ 101 \\ 110 \\ 111 \end{array} & \begin{bmatrix} a_{00}b_{00}c_{00} & a_{00}b_{00}c_{01} & a_{00}b_{01}c_{00} & a_{00}b_{01}c_{01} & a_{01}b_{00}c_{00} & a_{01}b_{00}c_{01} & a_{01}b_{01}c_{00} & a_{01}b_{01}c_{01} \\ a_{00}b_{00}c_{10} & a_{00}b_{00}c_{11} & a_{00}b_{01}c_{10} & a_{00}b_{01}c_{11} & a_{01}b_{00}c_{10} & a_{01}b_{00}c_{11} & a_{01}b_{01}c_{10} & a_{01}b_{01}c_{11} \\ a_{00}b_{10}c_{00} & a_{00}b_{10}c_{01} & a_{00}b_{11}c_{00} & a_{00}b_{11}c_{01} & a_{01}b_{10}c_{00} & a_{01}b_{10}c_{01} & a_{01}b_{11}c_{00} & a_{01}b_{11}c_{01} \\ a_{00}b_{10}c_{10} & a_{00}b_{10}c_{11} & a_{00}b_{11}c_{10} & a_{00}b_{11}c_{11} & a_{01}b_{10}c_{10} & a_{01}b_{10}c_{11} & a_{01}b_{11}c_{10} & a_{01}b_{11}c_{11} \\ a_{10}b_{00}c_{00} & a_{10}b_{00}c_{01} & a_{10}b_{01}c_{00} & a_{10}b_{01}c_{01} & a_{11}b_{00}c_{00} & a_{11}b_{00}c_{01} & a_{11}b_{01}c_{00} & a_{11}b_{01}c_{01} \\ a_{10}b_{00}c_{10} & a_{10}b_{00}c_{11} & a_{10}b_{01}c_{10} & a_{10}b_{01}c_{11} & a_{11}b_{00}c_{10} & a_{11}b_{00}c_{11} & a_{11}b_{01}c_{10} & a_{11}b_{01}c_{11} \\ a_{10}b_{10}c_{00} & a_{10}b_{10}c_{01} & a_{10}b_{11}c_{00} & a_{10}b_{11}c_{01} & a_{11}b_{10}c_{00} & a_{11}b_{10}c_{01} & a_{11}b_{11}c_{00} & a_{11}b_{11}c_{01} \\ a_{10}b_{10}c_{10} & a_{10}b_{10}c_{11} & a_{10}b_{11}c_{10} & a_{10}b_{11}c_{11} & a_{11}b_{10}c_{10} & a_{11}b_{10}c_{11} & a_{11}b_{11}c_{10} & a_{11}b_{11}c_{11} \end{bmatrix} \end{array}$$

$$\mathbf{\Gamma}^{\mathsf{T}}(\mathbf{A}\otimes\mathbf{B}\otimes\mathbf{C})\mathbf{\Gamma} =$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_{00}b_{00}c_{00} & a_{00}b_{00}c_{01} & a_{00}b_{01}c_{00} & a_{00}b_{01}c_{01} & a_{01}b_{00}c_{00} & a_{01}b_{00}c_{01} & a_{01}b_{01}c_{00} & a_{01}b_{01}c_{01} \\ a_{00}b_{00}c_{10} & a_{00}b_{00}c_{11} & a_{00}b_{01}c_{10} & a_{00}b_{01}c_{11} & a_{01}b_{00}c_{10} & a_{01}b_{00}c_{11} & a_{01}b_{01}c_{10} & a_{01}b_{01}c_{11} \\ a_{00}b_{10}c_{00} & a_{00}b_{10}c_{01} & a_{00}b_{11}c_{00} & a_{00}b_{11}c_{01} & a_{01}b_{10}c_{00} & a_{01}b_{10}c_{01} & a_{01}b_{11}c_{00} & a_{01}b_{11}c_{01} \\ a_{00}b_{10}c_{10} & a_{00}b_{10}c_{11} & a_{00}b_{11}c_{10} & a_{00}b_{11}c_{11} & a_{01}b_{10}c_{10} & a_{01}b_{10}c_{11} & a_{01}b_{11}c_{10} & a_{01}b_{11}c_{11} \\ a_{10}b_{00}c_{00} & a_{10}b_{00}c_{01} & a_{10}b_{01}c_{00} & a_{10}b_{01}c_{01} & a_{11}b_{00}c_{00} & a_{11}b_{00}c_{01} & a_{11}b_{01}c_{00} & a_{11}b_{01}c_{01} \\ a_{10}b_{00}c_{10} & a_{10}b_{00}c_{11} & a_{10}b_{01}c_{10} & a_{10}b_{01}c_{11} & a_{11}b_{00}c_{10} & a_{11}b_{00}c_{11} & a_{11}b_{01}c_{10} & a_{11}b_{01}c_{11} \\ a_{10}b_{10}c_{00} & a_{10}b_{10}c_{01} & a_{10}b_{11}c_{00} & a_{10}b_{11}c_{01} & a_{11}b_{10}c_{00} & a_{11}b_{10}c_{01} & a_{11}b_{11}c_{00} & a_{11}b_{11}c_{01} \\ a_{10}b_{10}c_{10} & a_{10}b_{10}c_{11} & a_{10}b_{11}c_{10} & a_{10}b_{11}c_{11} & a_{11}b_{10}c_{10} & a_{11}b_{10}c_{11} & a_{11}b_{11}c_{10} & a_{11}b_{11}c_{11} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

CHOPPED KRONECKER PRODUCTS

$$\mathbf{\Gamma}^{\mathsf{T}}(\mathbf{A} \otimes \mathbf{B} \otimes \mathbf{C})\mathbf{\Gamma} = \begin{bmatrix} a_{00}b_{00}c_{00} & a_{00}b_{00}c_{01} & a_{00}b_{01}c_{00} & & a_{01}b_{00}c_{00} & & & \\ a_{00}b_{00}c_{10} & a_{00}b_{00}c_{11} & a_{00}b_{01}c_{10} & & a_{01}b_{00}c_{10} & & & \\ a_{00}b_{10}c_{00} & a_{00}b_{10}c_{01} & a_{00}b_{11}c_{00} & & a_{01}b_{10}c_{00} & & & \\ & & & & & & & \\ a_{10}b_{00}c_{00} & a_{10}b_{00}c_{01} & a_{10}b_{01}c_{00} & & a_{11}b_{00}c_{00} & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \end{bmatrix}$$

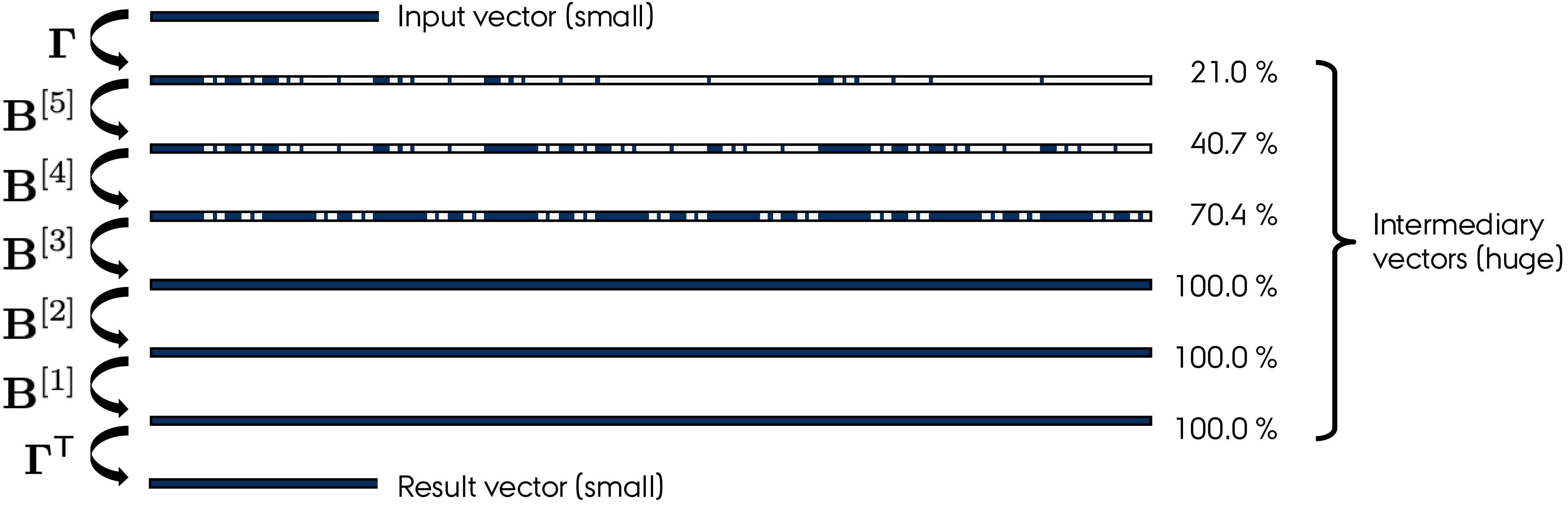
$$= \begin{bmatrix} a_{00}b_{00}c_{00} & a_{00}b_{00}c_{01} & a_{00}b_{01}c_{00} & a_{01}b_{00}c_{00} \\ a_{00}b_{00}c_{10} & a_{00}b_{00}c_{11} & a_{00}b_{01}c_{10} & a_{01}b_{00}c_{10} \\ a_{00}b_{10}c_{00} & a_{00}b_{10}c_{01} & a_{00}b_{11}c_{00} & a_{01}b_{10}c_{00} \\ a_{10}b_{00}c_{00} & a_{10}b_{00}c_{01} & a_{10}b_{01}c_{00} & a_{11}b_{00}c_{00} \\ a_{10}b_{00}c_{10} & a_{10}b_{00}c_{11} & a_{10}b_{01}c_{10} & a_{11}b_{00}c_{10} \end{bmatrix}$$

How to multiply a chopped Kronecker product with a vector?

A SIMPLE MVP

Naive approach: $\widehat{\mathbf{B}}\widehat{\mathbf{v}} = \mathbf{\Gamma}^{\mathsf{T}}\mathbf{B}\mathbf{\Gamma}\widehat{\mathbf{v}} = \mathbf{\Gamma}^{\mathsf{T}}(\mathbf{B}^{[1]}\mathbf{B}^{[2]}\mathbf{B}^{[3]}\mathbf{B}^{[4]}\mathbf{B}^{[5]})\mathbf{\Gamma}\widehat{\mathbf{v}}$

Example: $D = 5,\ n = 3,\ \alpha = 2$



A FASTER MVP

Incomplete grid + additive kernel:

$$\hat{\mathbf{K}} = \mathbf{\Gamma}^\top \mathbf{K} \mathbf{\Gamma} = \sum_{\mathbf{m}} \sigma^2_{|\mathbf{m}|} \mathbf{\Gamma}^\top (\mathbf{K}^{[\mathbf{m}]} \mathbf{J}^{[\neg \mathbf{m}]}) \mathbf{\Gamma}$$

Chopping matrix

Number of terms is $\mathcal{O}(D^\alpha)$

Sum of chopped Kronecker products

LU-based machinery: $\mathcal{O}(nD\hat{N})$ for a single chopped Kronecker product

Existing machinery gives polynomial scaling, but the total cost is still too large!

Wodraszka and Carrington (2019); Zak and Carrington (2019)
Simmons and Carrington (2023); Holzmüller and Pflüger (2021)

A VERY FAST MVP

Observation:

$$\mathbf{J}^{(m)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \mathbf{L}^{(m)}\mathbf{R}^{(m)}\mathbf{U}^{(m)}$$

A final rewrite:

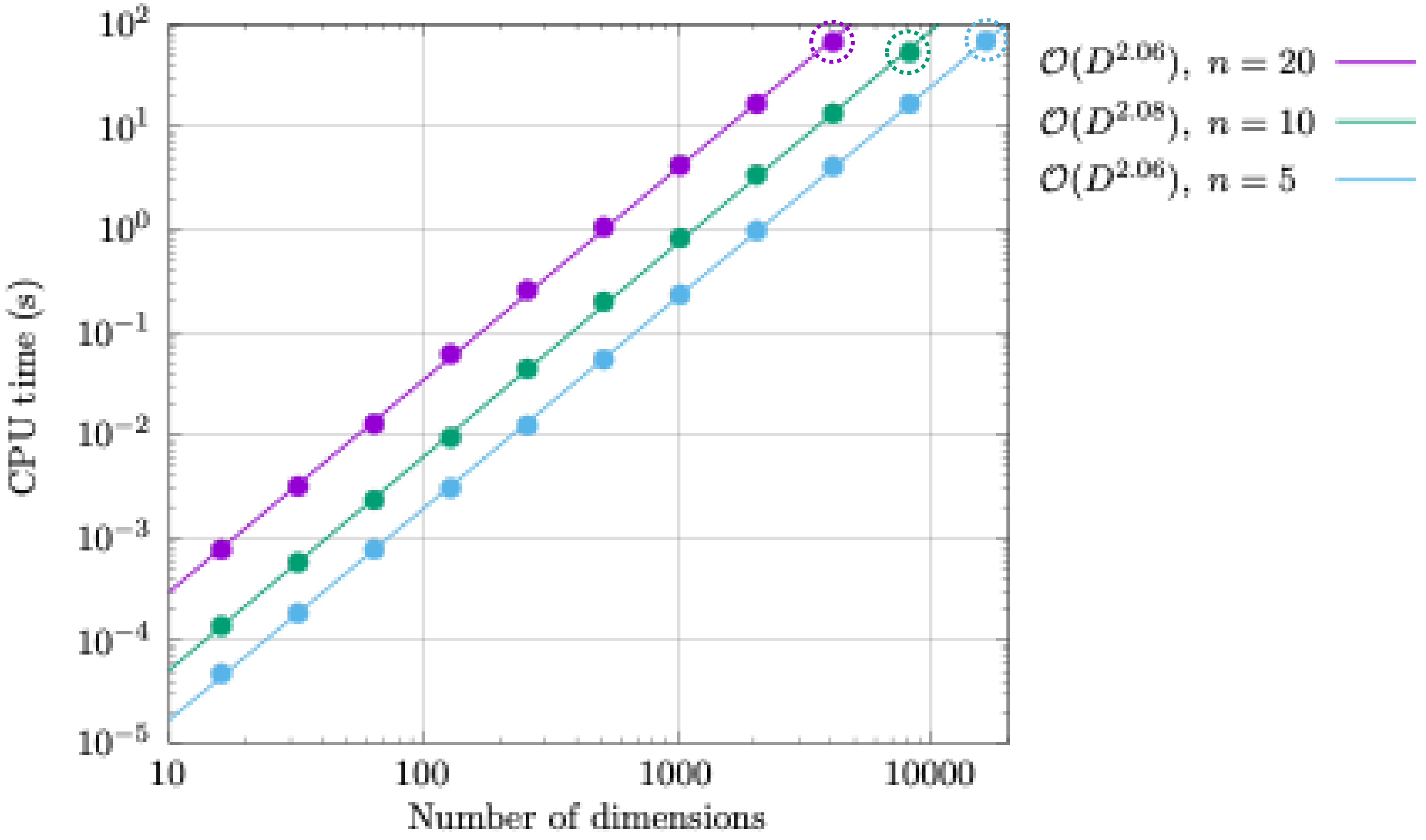
$$\underbrace{\widehat{\mathbf{K}} = \widehat{\mathbf{L}}}_{\mathcal{O}(\alpha\widehat{N})} \underbrace{\left(\sum_{\mathbf{m}} \sigma^2_{|\mathbf{m}|} \widehat{\mathbf{M}^{[\mathbf{m}]}} \widehat{\mathbf{R}^{[\neg\mathbf{m}]}}\right)}_{\mathcal{O}(n\alpha\widehat{N})} \underbrace{\widehat{\mathbf{U}}}_{\mathcal{O}(\alpha\widehat{N})}$$

Total cost $\mathcal{O}(n\alpha\widehat{N}) \sim \mathcal{O}(D^{\alpha})$!
Near-linear N-scaling
Polynomial D-scaling

BENCHMARK: D-SCALING ($\alpha = 2$)

Computational scaling: $\mathcal{O}(n\alpha\widehat{N}) \sim \mathcal{O}(D^{\alpha})$

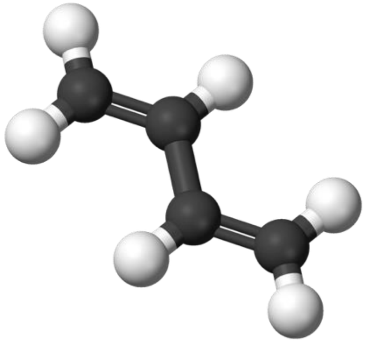
$\alpha = 2 \longrightarrow \mathcal{O}(D^2)$



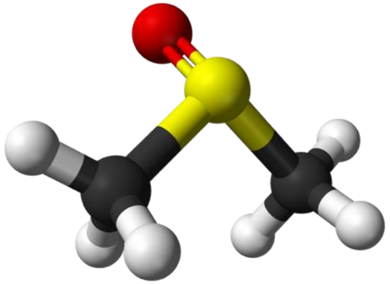
Three largest tests

n	D	$\widehat{N}$	CPU time (s)
20	4096	3.03×10^9	67.1
10	8192	2.72×10^9	53.1
5	16 384	2.15×10^9	68.7

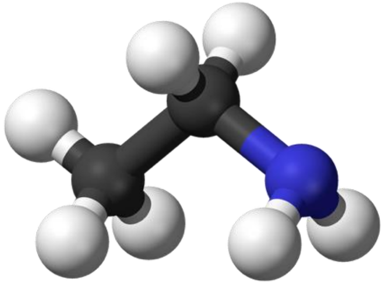
FULL CUTS-GPR CALCULATIONS



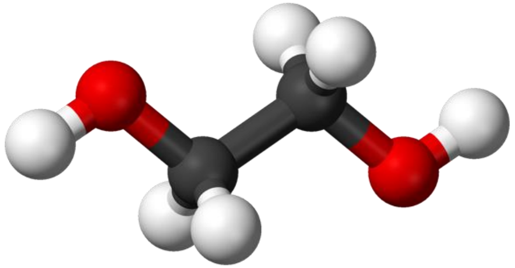
Butadiene



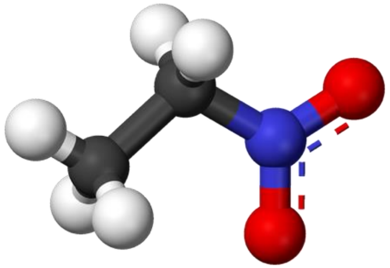
DMSO



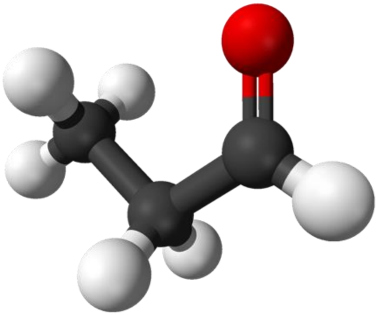
Ethylamine



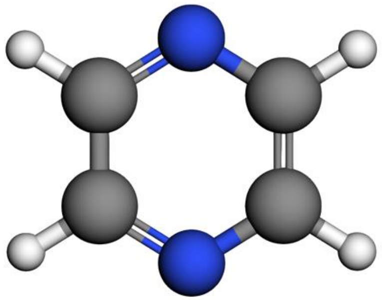
Ethylene glycol



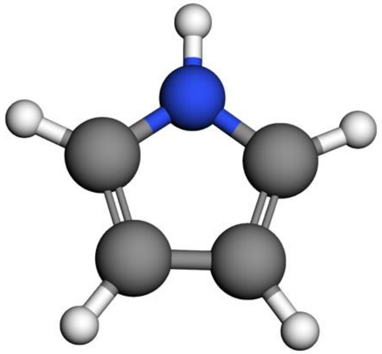
Nitroethane



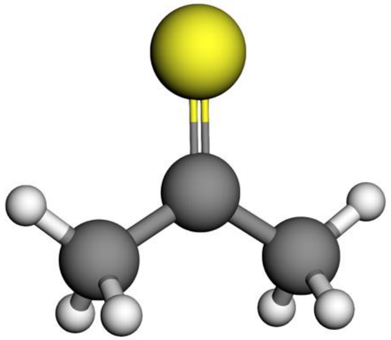
Propanal



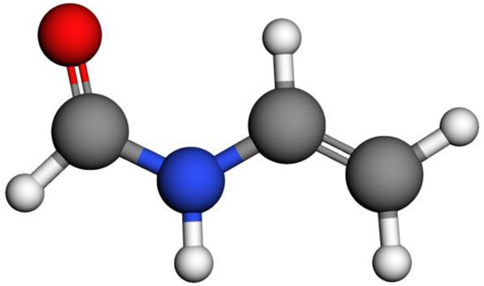
Pyrazine



Pyrrole



Thioacetone



Vinylformamide

FULL CUTS-GPR CALCULATIONS

Optimize hyperparameters and make predictions

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{1}{2}\boldsymbol{\alpha}^{\mathsf{T}} \frac{\partial \mathbf{C}}{\partial \theta} \boldsymbol{\alpha} - \frac{1}{2}\operatorname{tr}\left(\mathbf{C}^{-1} \frac{\partial \mathbf{C}}{\partial \theta}\right) \qquad \boldsymbol{\mu} = \mathbf{K}_*^{\mathsf{T}} \mathbf{C}^{-1} \mathbf{y} = \mathbf{K}_*^{\mathsf{T}} \boldsymbol{\alpha}$$

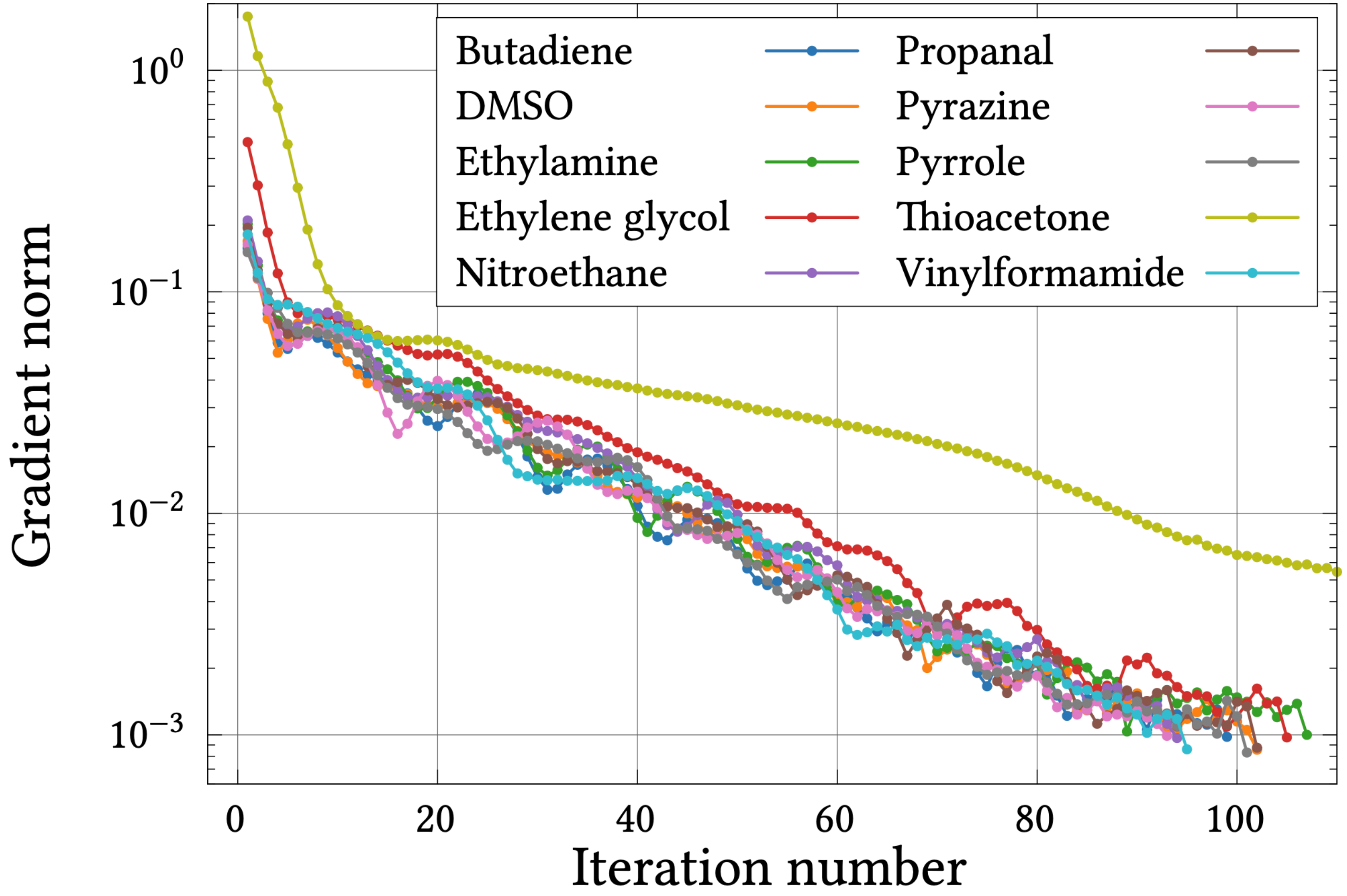
Everything can be computed using MVPs (plus a few bits and pieces)

Computational setup:

- 10 molecules with 10 atoms
- $D = 24$
- $\alpha = 3$
- Training points: $n = 7\ ,\quad \widehat{N} = 0.45\ \mathrm{M}$
- Test points: $n = 11,\quad \widehat{N}_* = 1.60\ \mathrm{M}$

(more details in the paper)

HYPERPARAMETER OPTIMIZATION

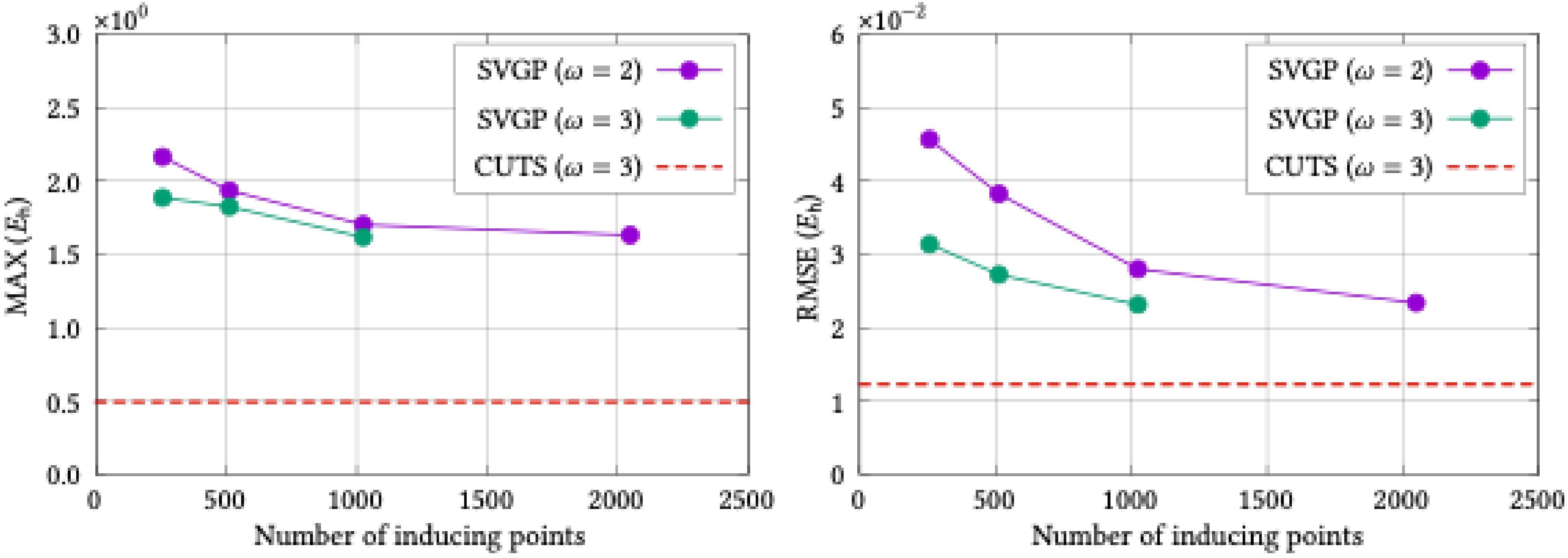


Threshold

TIMINGS (36 CORES)

	Optimization		Predictive mean	
Molecule	CPU (h)	Wall (h)	CPU (min)	Wall (min)
Butadiene	63.6	1.95	64.8	1.83
DMSO	68.6	2.15	55.3	1.60
Ethylamine	59.6	1.94	60.9	1.73
Ethylene glycol	73.7	2.25	56.4	1.65
Nitroethane	64.2	1.91	65.8	1.88
Propanal	72.2	2.16	56.3	1.60
Pyrazine	65.6	1.96	65.2	1.84
Pyrrole	67.7	2.07	64.2	1.85
Thioacetone	109.1	3.58	55.0	1.58
Vinylformamide	50.1	1.62	57.8	1.63
Mean	69.4	2.16	60.2	1.72

COMPARISON WITH SVGP



CUTS-GPR is way better and timings are comparable $\longrightarrow$ no reason to use SVGP!

SUMMARY

- High-dimensional problems and low-order models
- Two strategies for accelerating GPs
- CUTS-GPR

THANK YOU FOR LISTENING!

REFERENCES I

- Mads Greisen Højlund, August Smart Lykke-Møller, Henry Moss and Ove Christiansen, *Don't Get Your Kroneckers in a Twist: Gaussian Processes on High-Dimensional Incomplete Grids* (2026), https://arxiv.org/abs/2605.08036.
- Jacob R. Gardner, Geoff Pleiss, David Bindel, Kilian Q. Weinberger and Andrew Gordon Wilson, *GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration,* NeurIPS 2018.
- Yunus Saatchi, Scalable Inference for Structured Gaussian Process Models (2011), PhD thesis.
- Andrew Gordon Wilson, Elad Gilboa, Arye Nehorai and John P. Cunningham, *Fast Kernel Learning for Multidimensional Pattern Extrapolation,* NIPS 2014.
- Elad Gilboa, Yunus Saatci and John P. Cunningham, *Scaling Multidimensional Inference for Structured Gaussian Processes,* IEEE Trans. Pattern Anal. Mach. Intell. **37**, 424–436 (2015).
- S. Flaxman, A. G. Wilson, D. B. Neill, H. Nickisch, and A. J. Smola, *Fast Kronecker Inference in Gaussian Processes with non-Gaussian Likelihoods,* ICML 2015.
- D. K. Duvenaud, H. Nickisch and C. E. Rasmussen, *Additive Gaussian Processes,* NIPS 2011.
- N. Durrande, D. Ginsbourger, and L. Carraro, *Additive covariance kernels for high-dimensional gaussian process modeling*, Annales de la faculté des sciences de Toulouse Mathématiques **21**, 481–499 (2012).
- X. Lu, A. Boukouvalas and J. Hensman, *Additive Gaussian Processes Revisited*, ICML 2022.
- S. Ishida, F. Panero and W. Bergsma, *Hierarchical additive interaction modelling with Gaussian process prior and its efficient implementation for multidimensional grid data* (2025), https://arxiv.org/abs/2305.07073.

REFERENCES II

- R. Wodraszka and T. Carrington, *A pruned collocation-based multiconfiguration time-dependent Hartree approach using a Smolyak grid for solving the Schrödinger equation with a general potential energy surface*, J. Chem. Phys. **150**, 154108 (2019) 10.1063/1.5093317.
- E. J. Zak and T. Carrington, *Using collocation and a hierarchical basis to solve the vibrational Schrödinger equation*, J. Chem. Phys. **150**, 204108 (2019) 10.1063/1.5096169.
- J. Simmons and T. Carrington, *Computing vibrational spectra using a new collocation method with a pruned basis and more points than basis functions: Avoiding quadrature*, J. Chem. Phys. **158**, 144115 (2023) 10.1063/5.0146703.
- D. Holzmüller and D. Pflüger, *Fast Sparse Grid Operations Using the Unidirectional Principle: A Generalized and Unified Framework*, Sparse Grids and Applications – Munich 2018, edited by H.-J. Bungartz, J. Garcke, and D. Pflüger (2021), pp. 69–100, 10.1007/978-3-030-81362-8_4.