Rev. Bayes to the rescue: fast BSM predictions from Gaussian process regression

Anders Kvellestad, University of Oslo

4th annual N-PACT meeting, Kristiansand, 5 August 2019

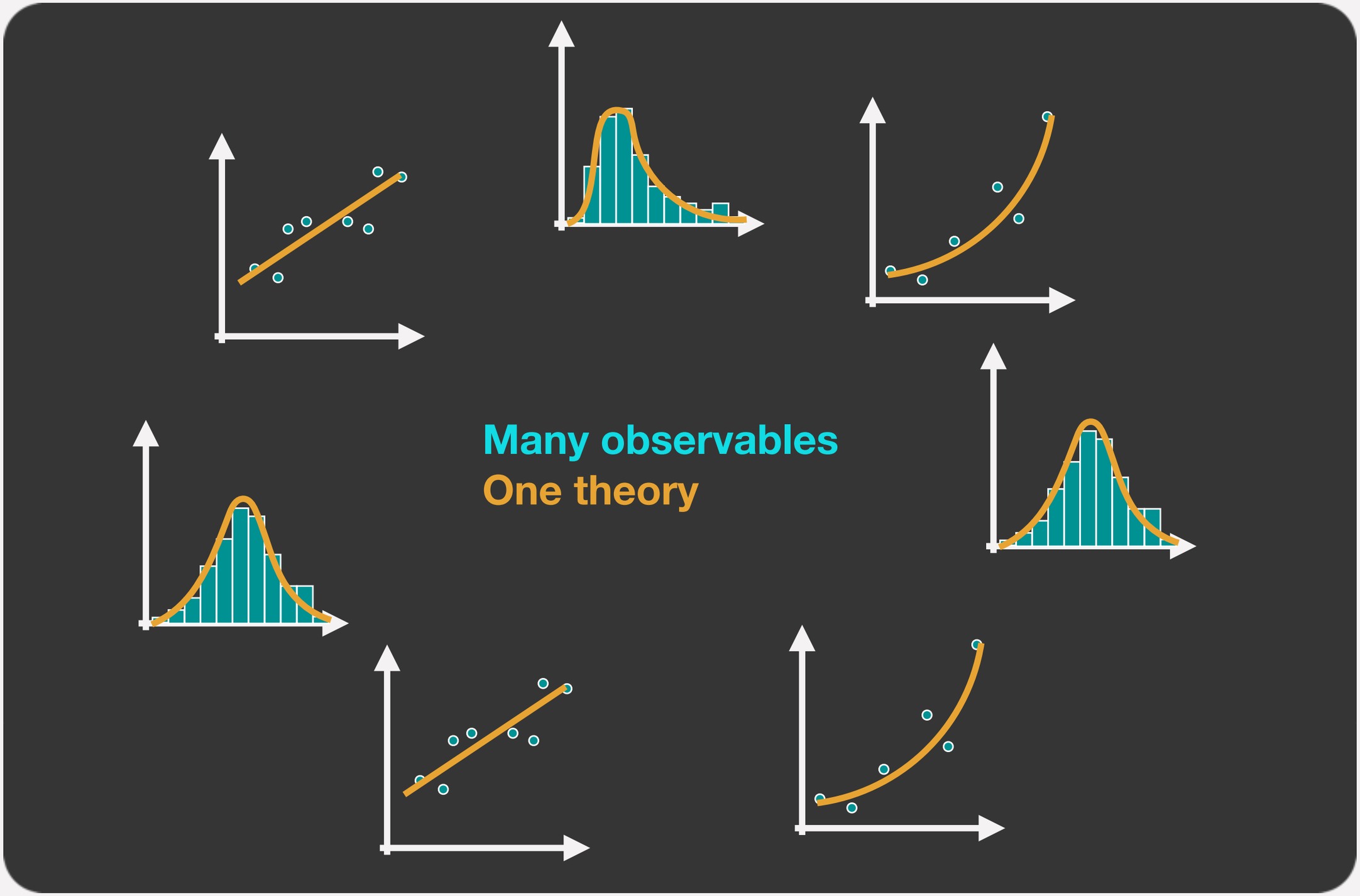
UiO : University of Oslo

Outline

1. The challenge
2. The ideal solution
3. The pragmatic solution
4. The results

1. The challenge

Global fits

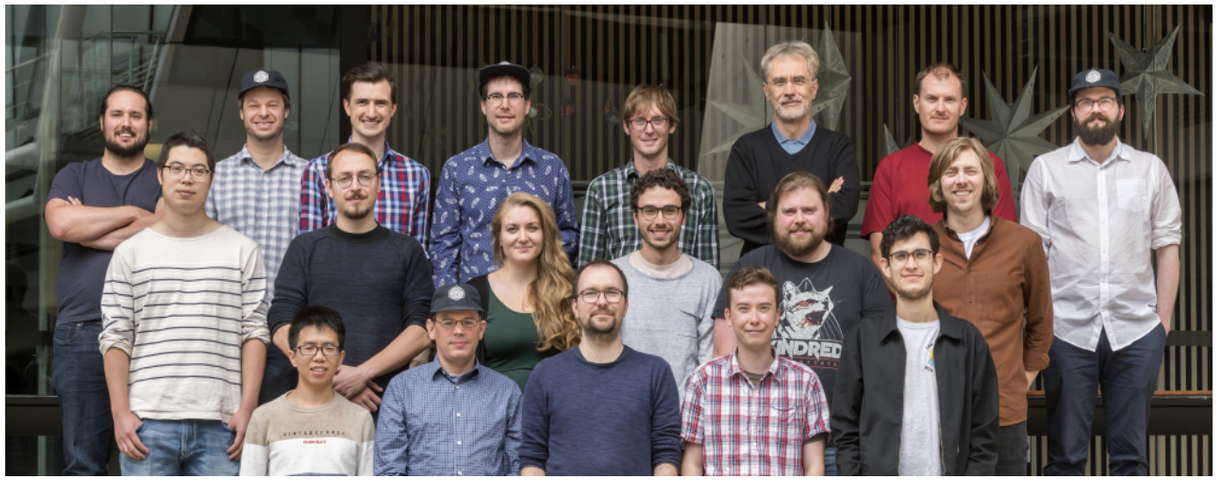


GAMBIT: The Global And Modular BSM Inference Tool

gambit.hepforge.org EPJC **77** (2017) 784 arXiv:1705.07908

- Extensive model database – not just SUSY
- Extensive observable/data libraries
- Many statistical and scanning options (Bayesian & frequentist)
- *Fast* LHC likelihood calculator
- Massively parallel
- Fully open-source

- Fast definition of new datasets and theories
- Plug and play scanning, physics and likelihood packages



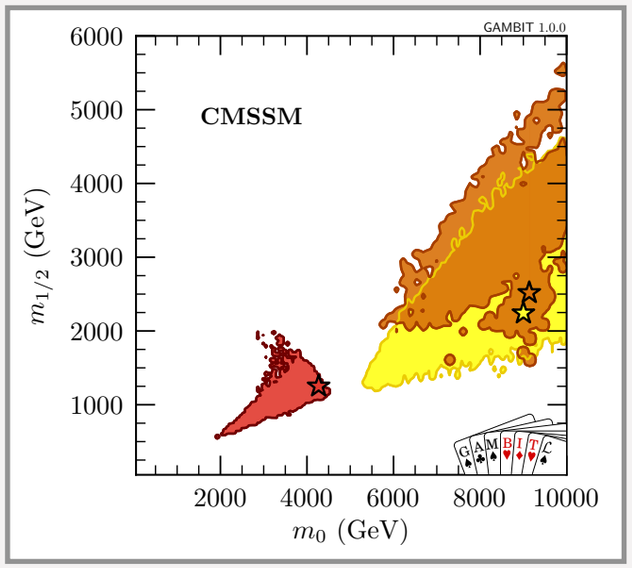
Members of:
ATLAS, Belle-II, CLiC, CMS, CTA, *Fermi*-LAT, DARWIN, IceCube, LHCb, SHiP, XENON

Authors of:
DarkSUSY, DDCalc, Diver, FlexibleSUSY, gamlike, GM2Calc, IsaTools, nulike, PolyChord, Rivet, SoftSUSY, SuperISO, SUSY-AI, WIMPSim

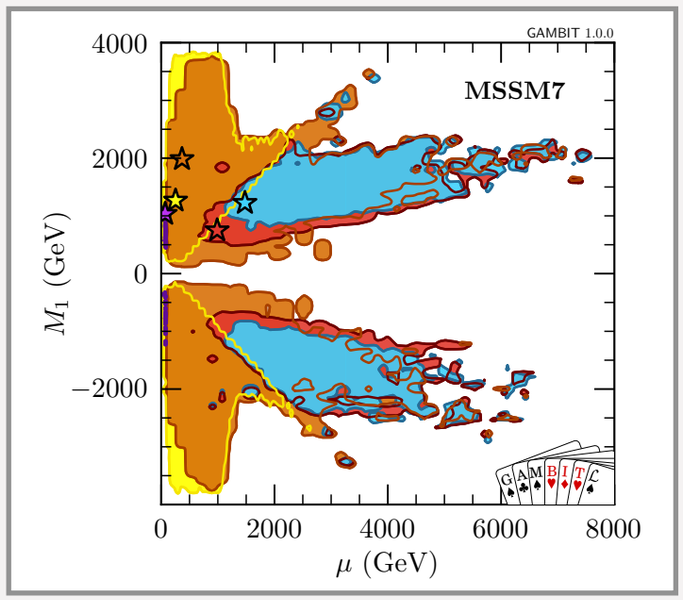
Recent collaborators:
P Athron, C Balázs, A Beniwal, S Bloor, T Bringmann, A Buckley, J Eliel Camargo-Molina, C Chang, M Chrzaszcz, J Conrad, J Cornell, M Danninger, J Edsjö, B Farmer, A Fowlie, T Gonzalo, P Grace, W Handley, J Harz, S Hoof, F Kahlhoefer, N Avis Kozar, A Kvellestad, P Jackson, R Jardine, A Ladhu, N Mahmoudi, G Martinez, M Prim, F Rajec, A Raklev, J Renk, C Rogan, R Ruiz, I Sáez Casares, N Serra, A Scaffidi, P Scott, P Stöcker, W Su, J Van den Abeele, A Vincent, C Weniger, M White, Y Zhang

40+ participants in 11 experiments and 14 major theory codes

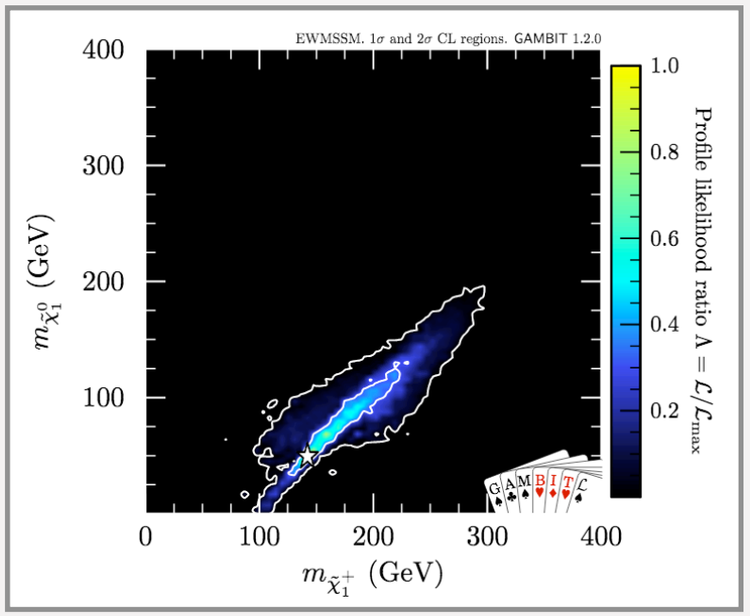
Some GAMBIT physics results



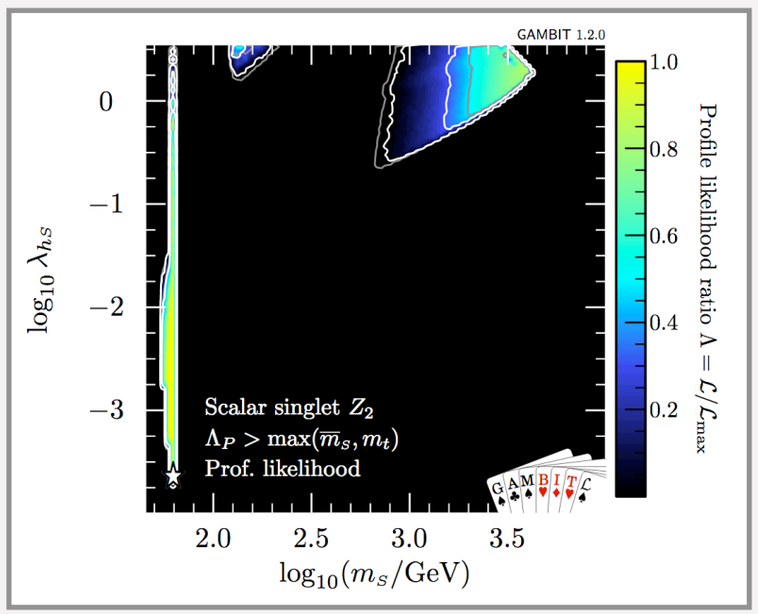
CMSSM, NUHM1, NUHM2:
1705.07935



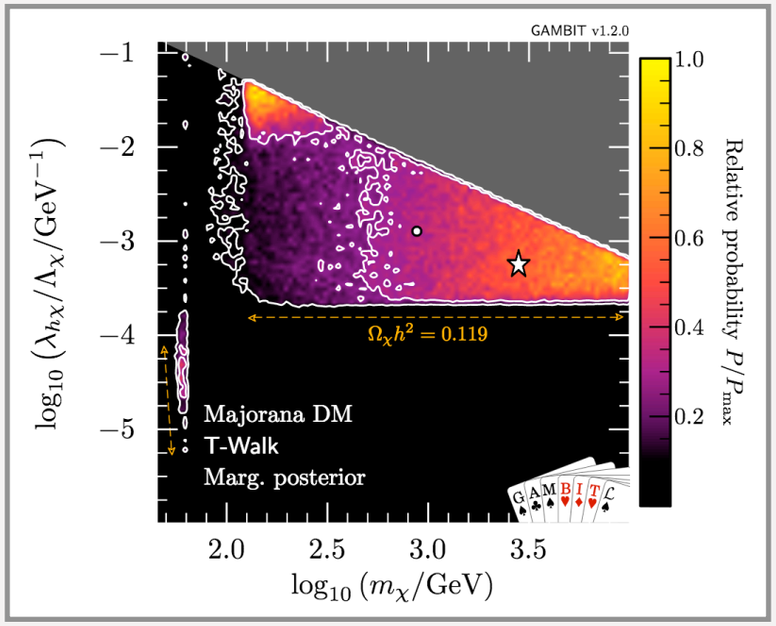
MSSM7: 1705.07917



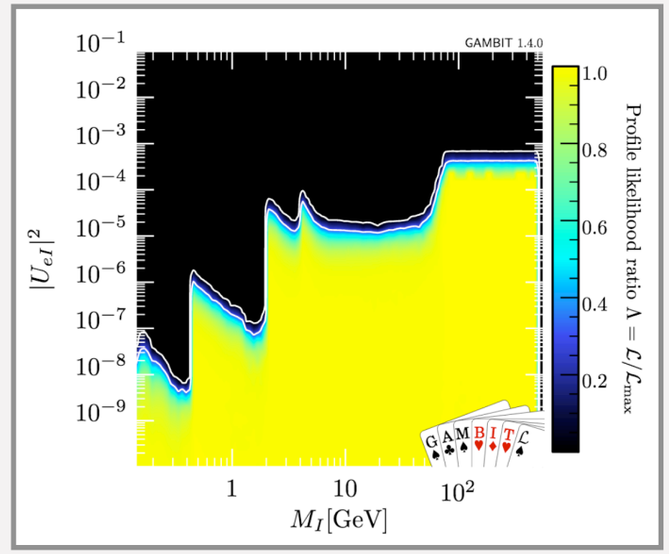
EW-MSSM: 1809.02097



Scalar Higgs portal DM (Z2 & Z3):
1705.07931, 1806.11281



Vector and fermion Higgs portal DM:
1808.10465



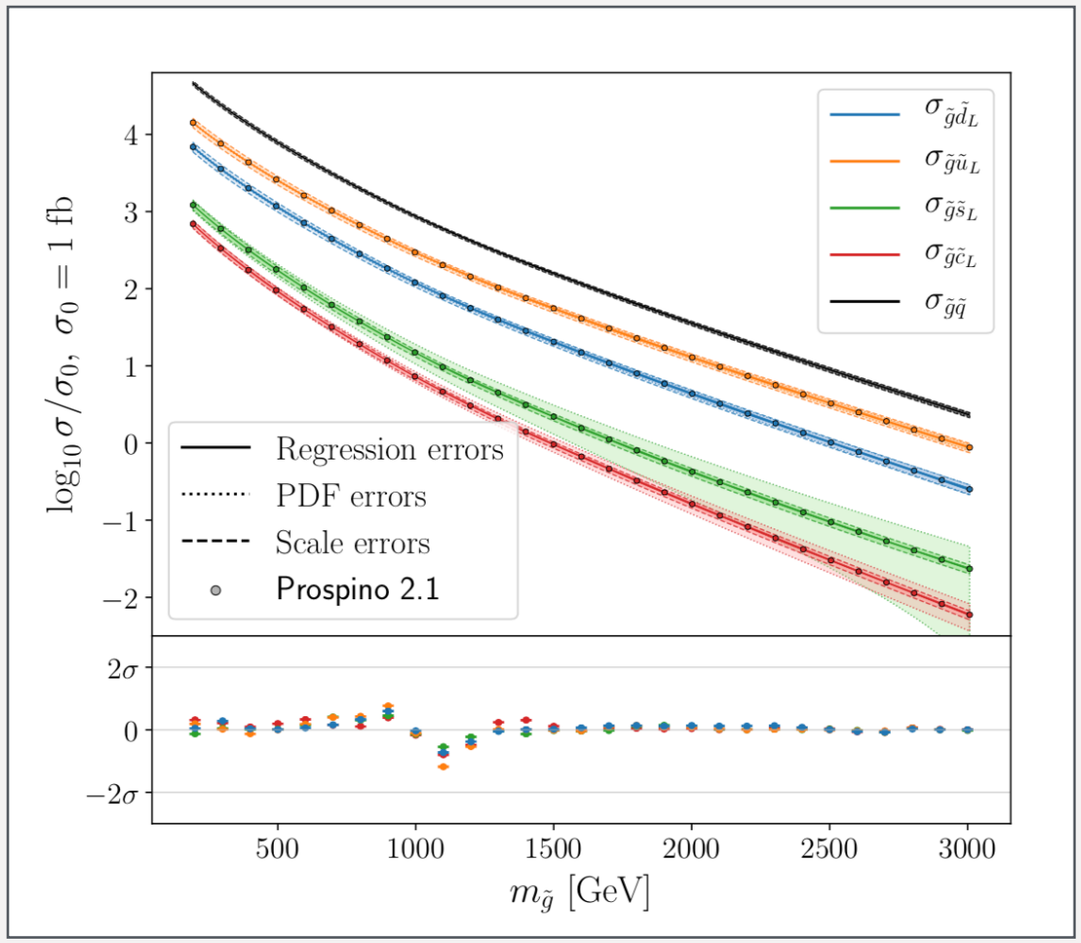
Right-handed neutrinos:
1908.02302

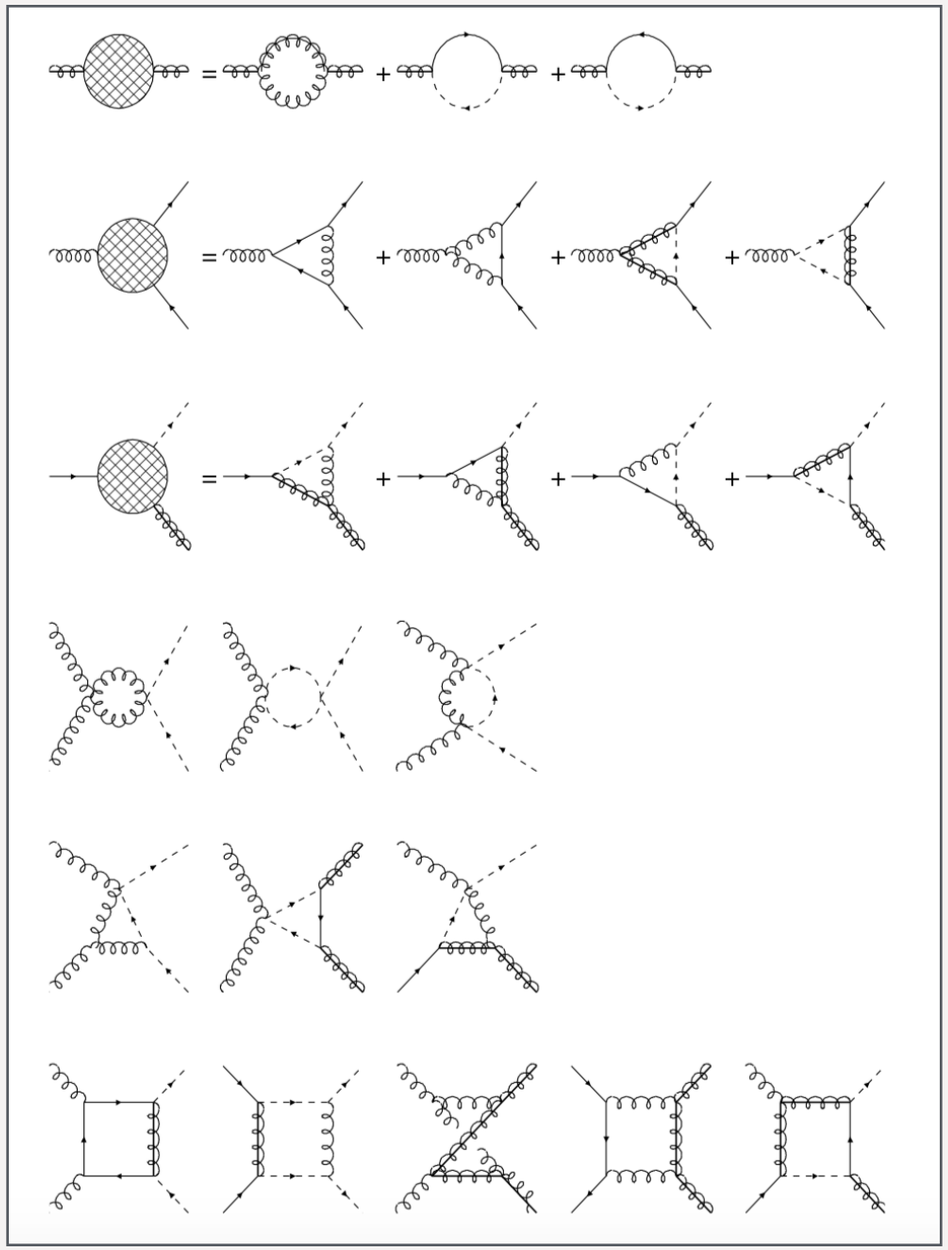
- A typical BSM global fit requires **O(10M) parameter samples**

- Need **fast**, yet **sufficiently accurate,** theory predictions…

One of the most CPU-expensive theory predictions:

Higher-order BSM production cross-sections for LHC predictions

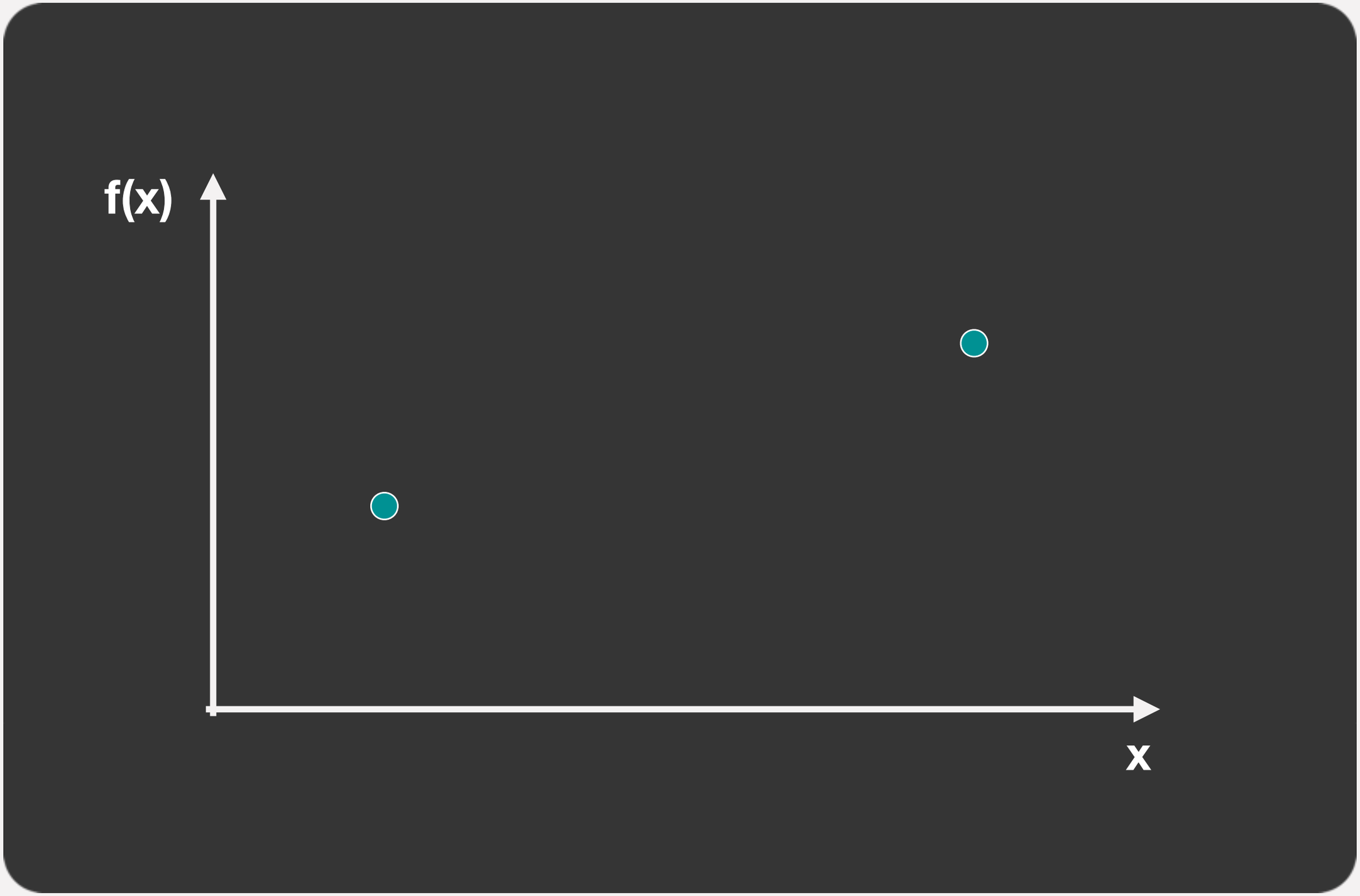


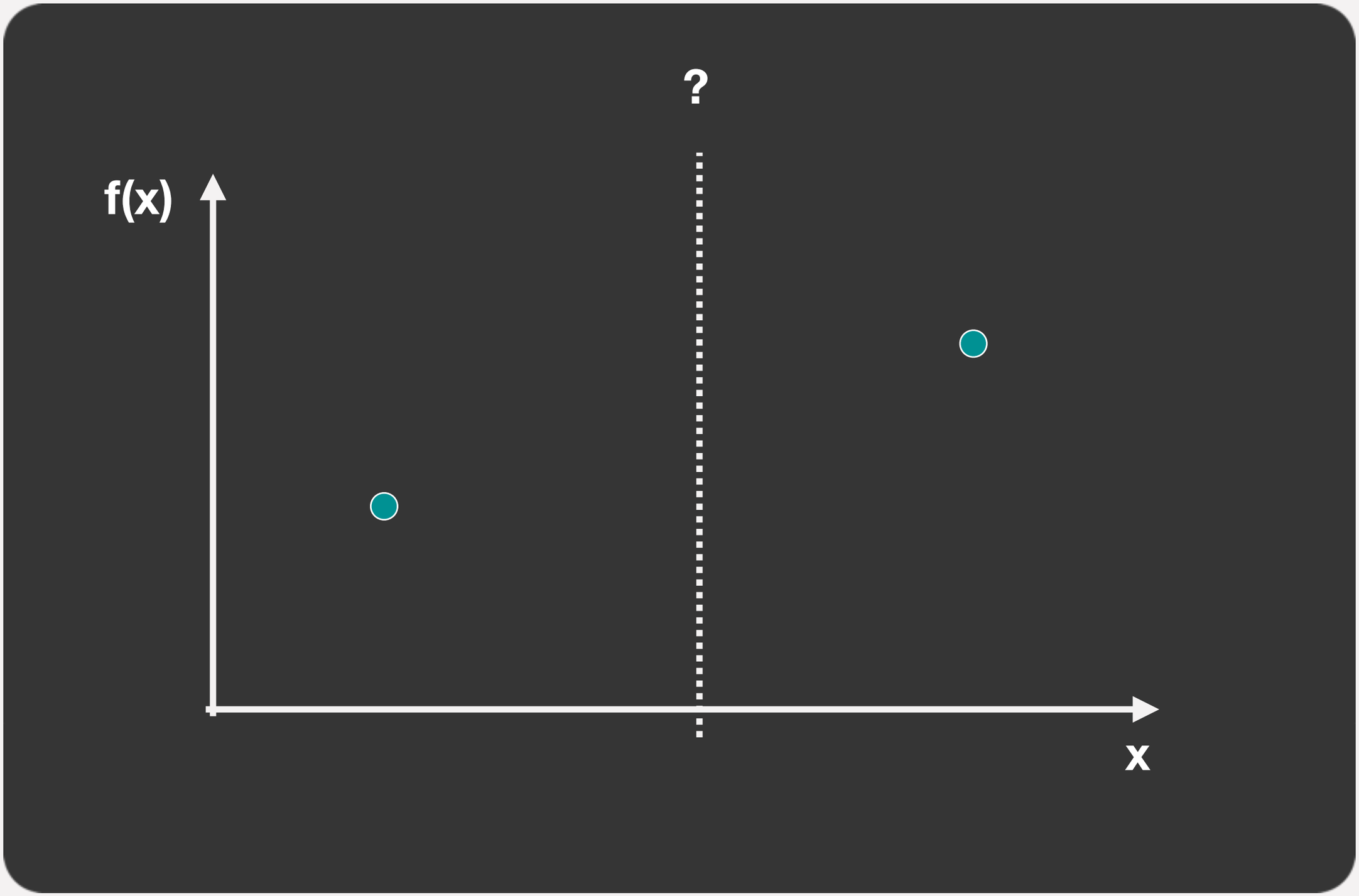


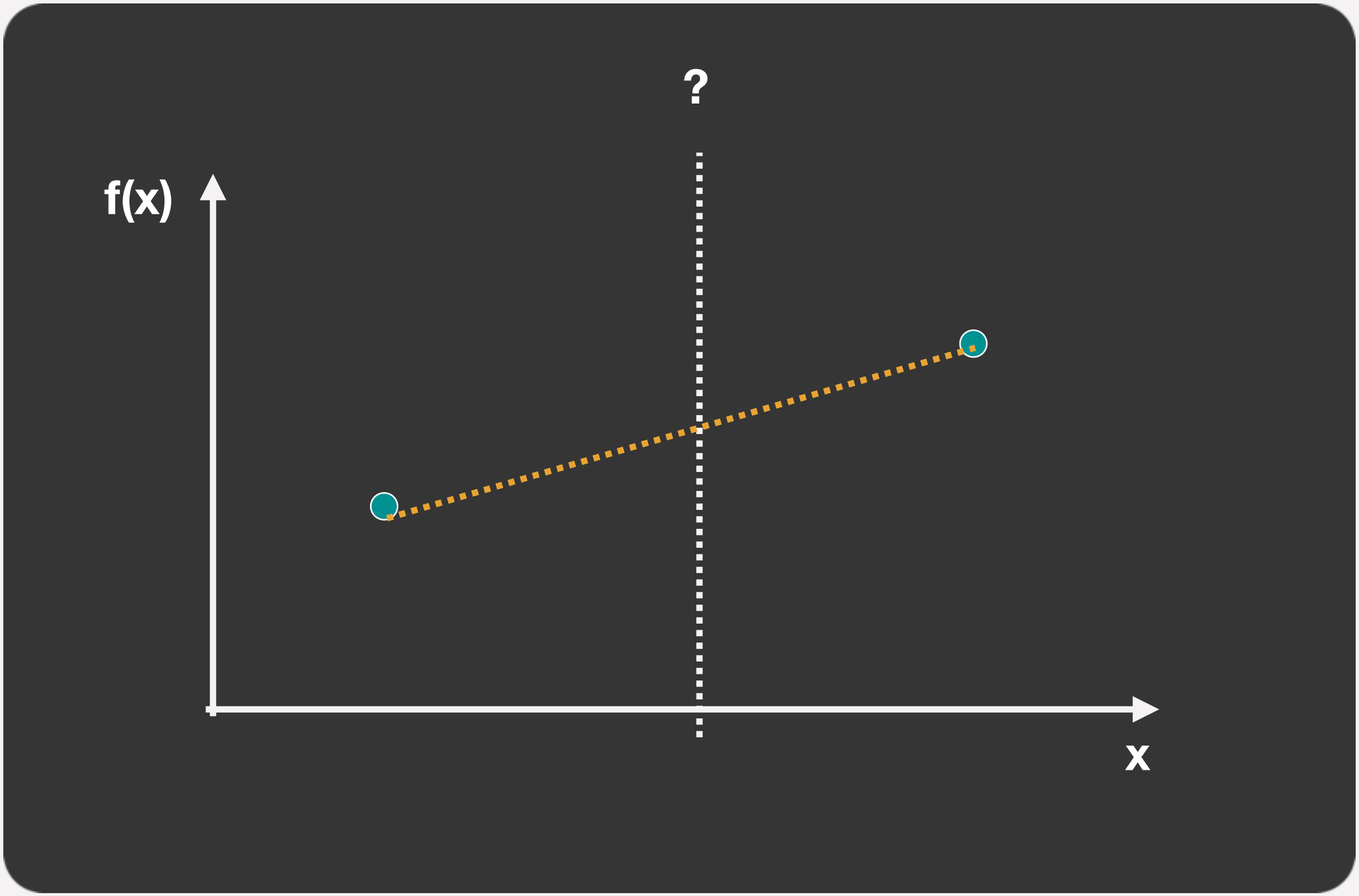
Evaluation time for a single parameter point:
~minutes/hours…

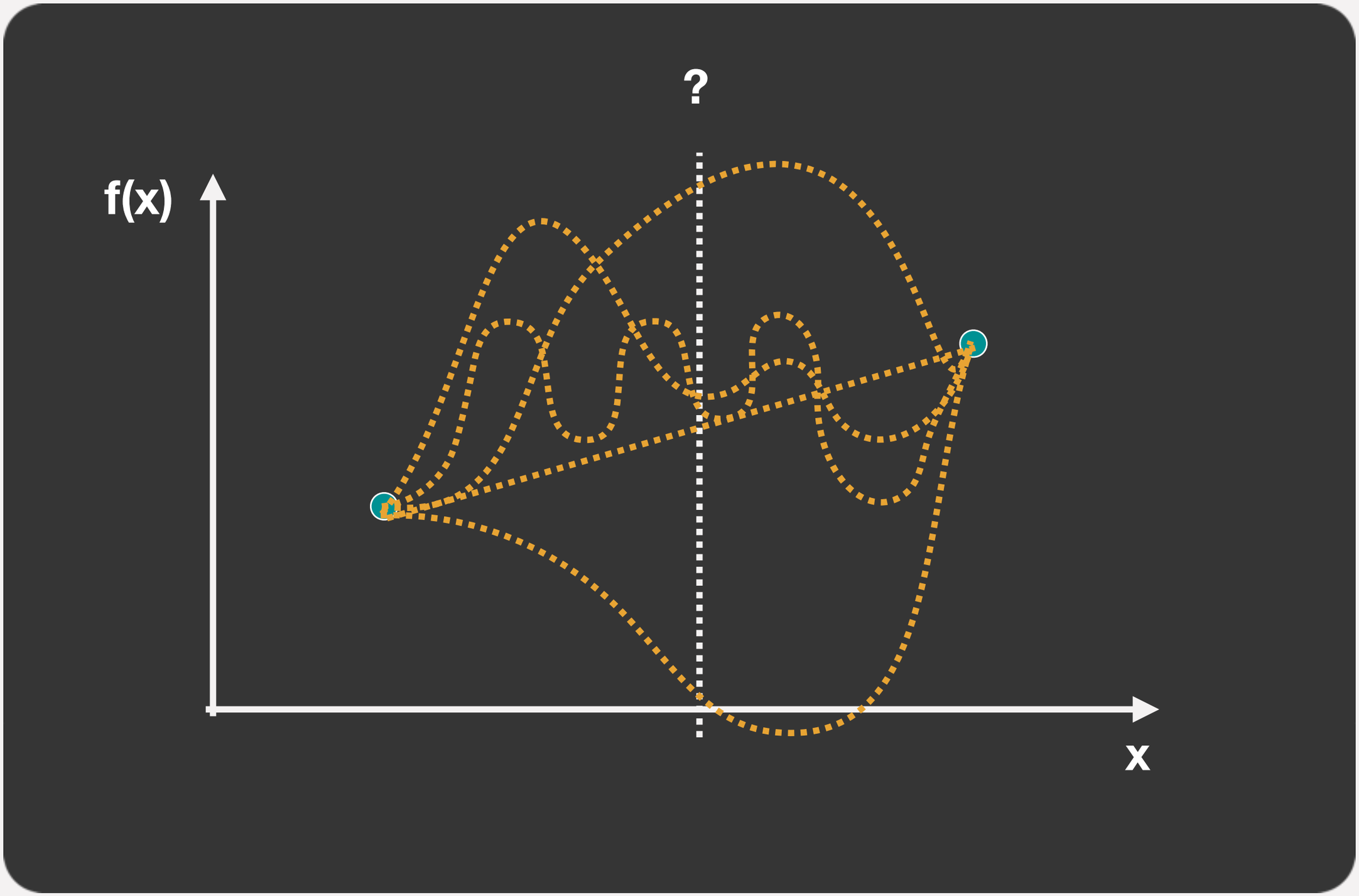
Let’s see if we can speed things up with some smart regression…

2. The ideal solution

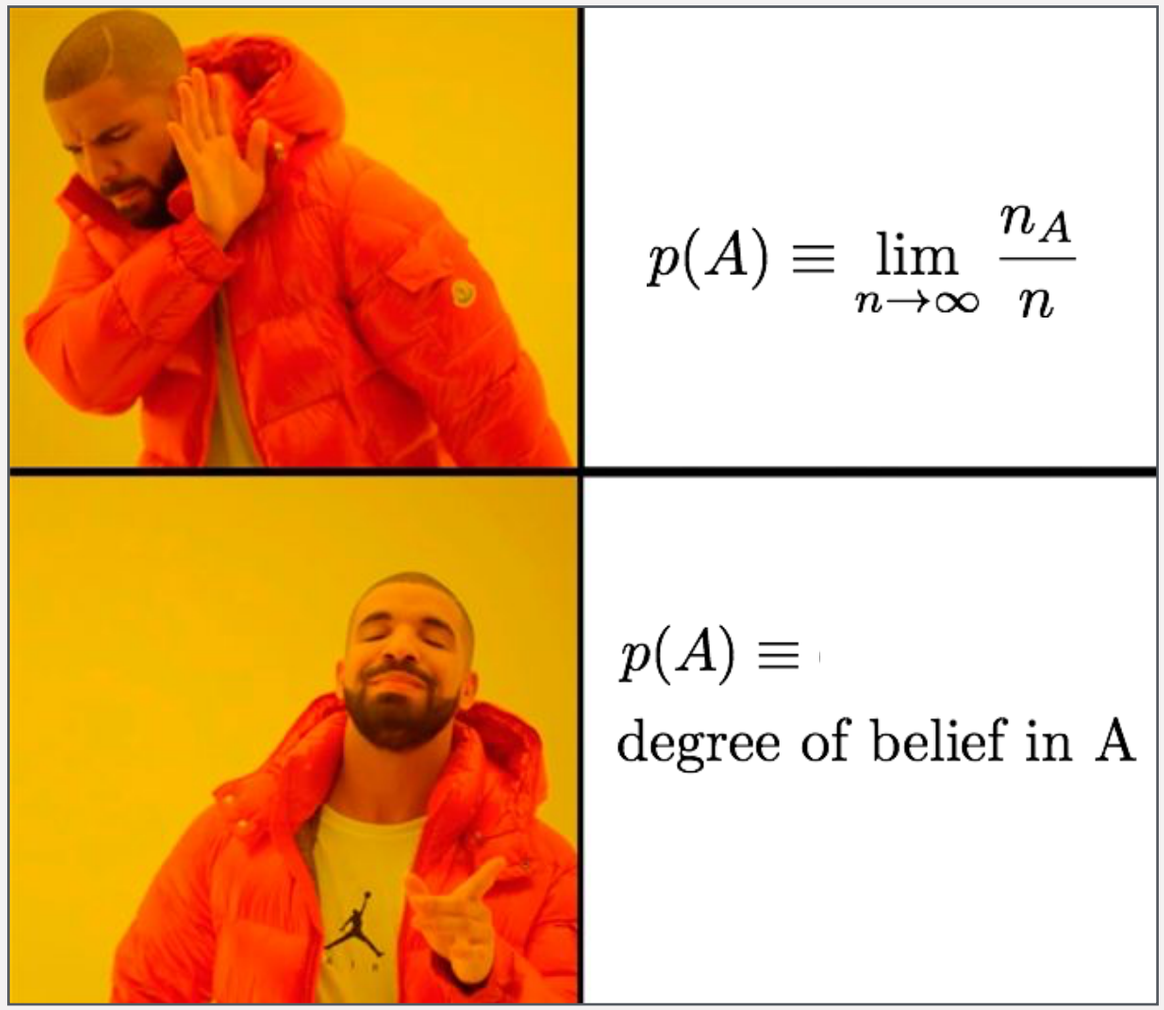

f(x)
x


?
f(x)
x

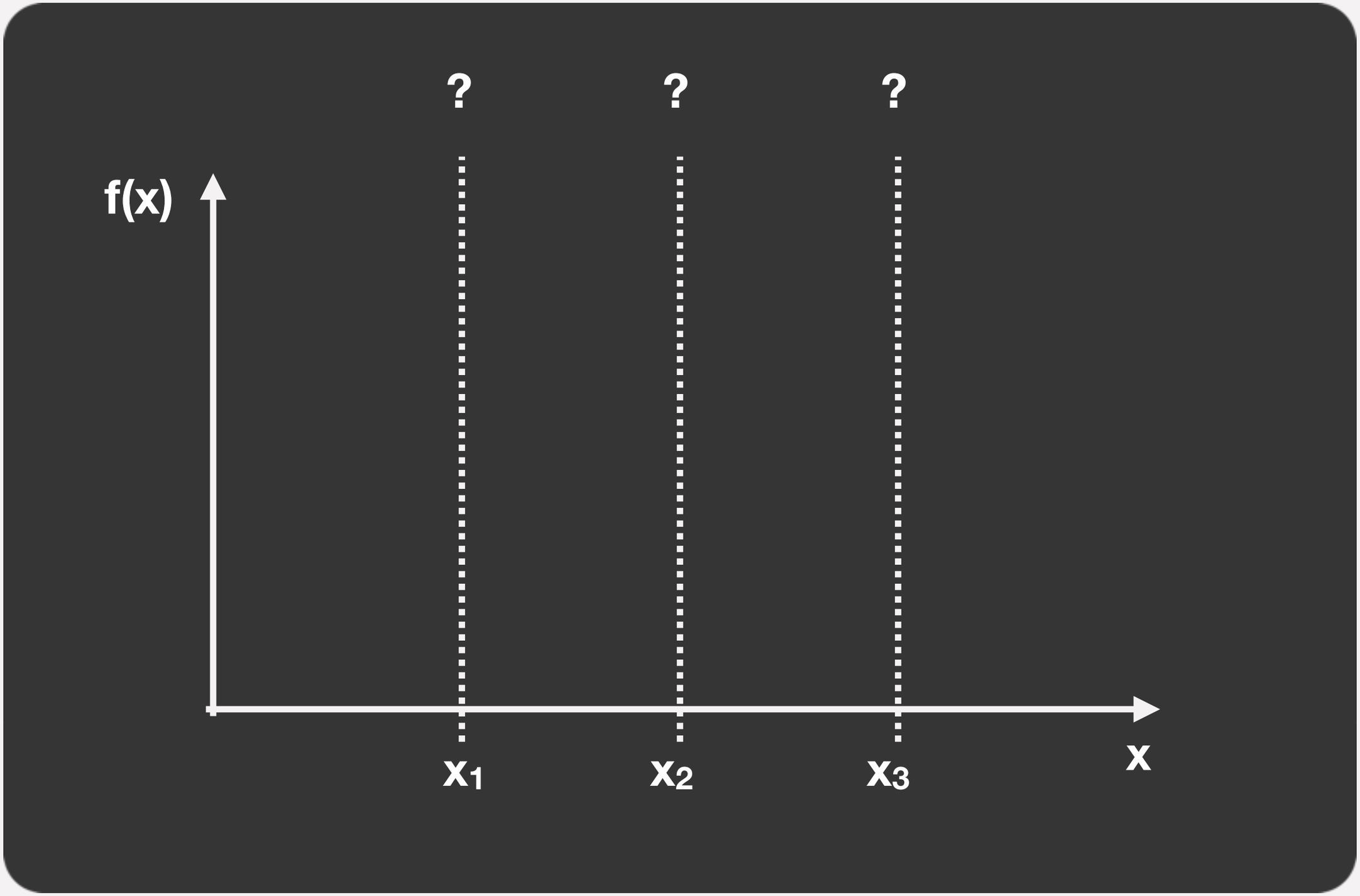

?
f(x)
x


?
f(x)
x

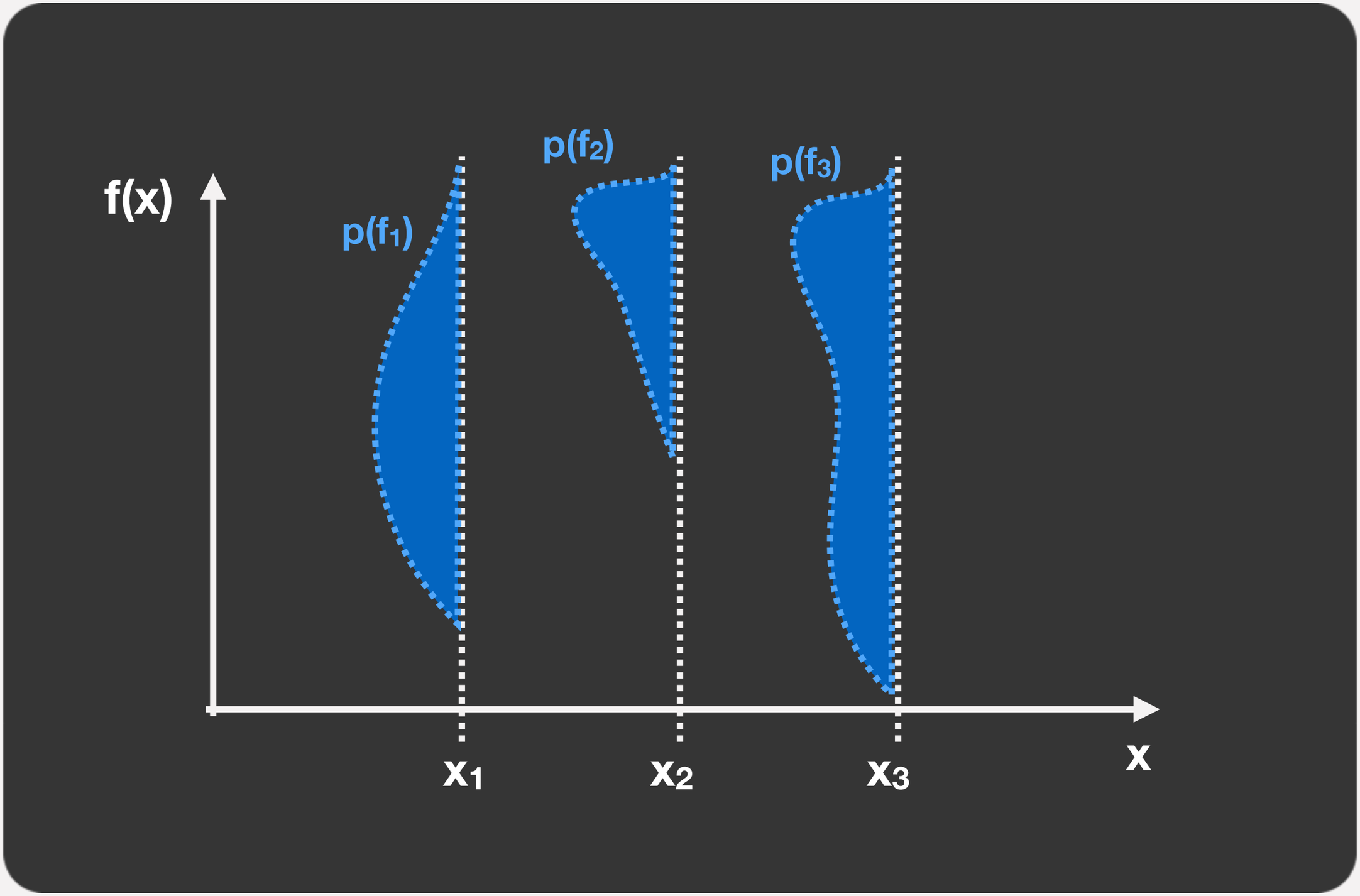
- Need to make an educated guess for the unknown **f** at any given **x**
- Being sensible Bayesians, we know we should **quantify our beliefs about f** using **probabilities**



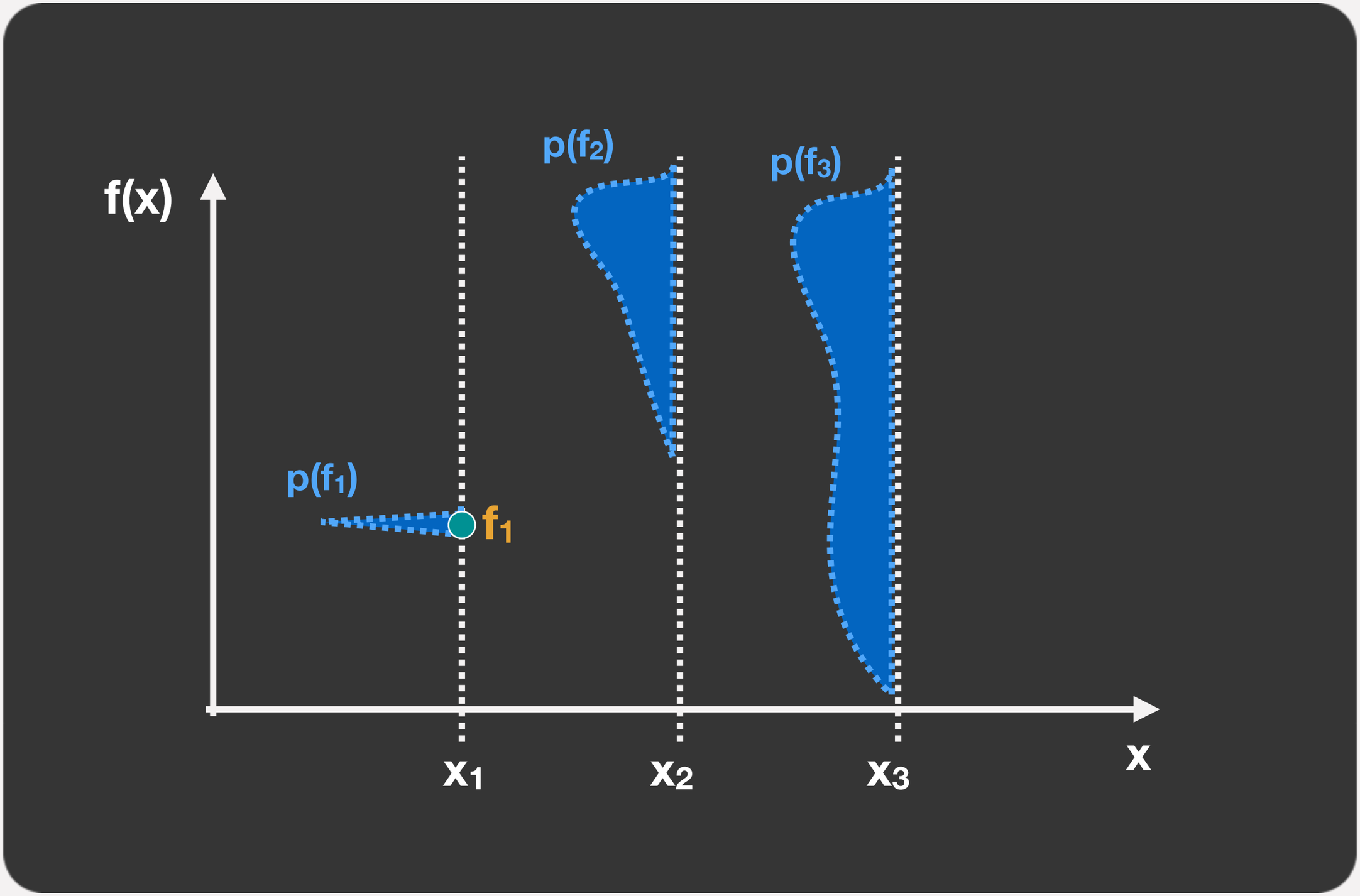
Consider three unknown function values: $\mathbf{f_1}$, $\mathbf{f_2}$, $\mathbf{f_3}$



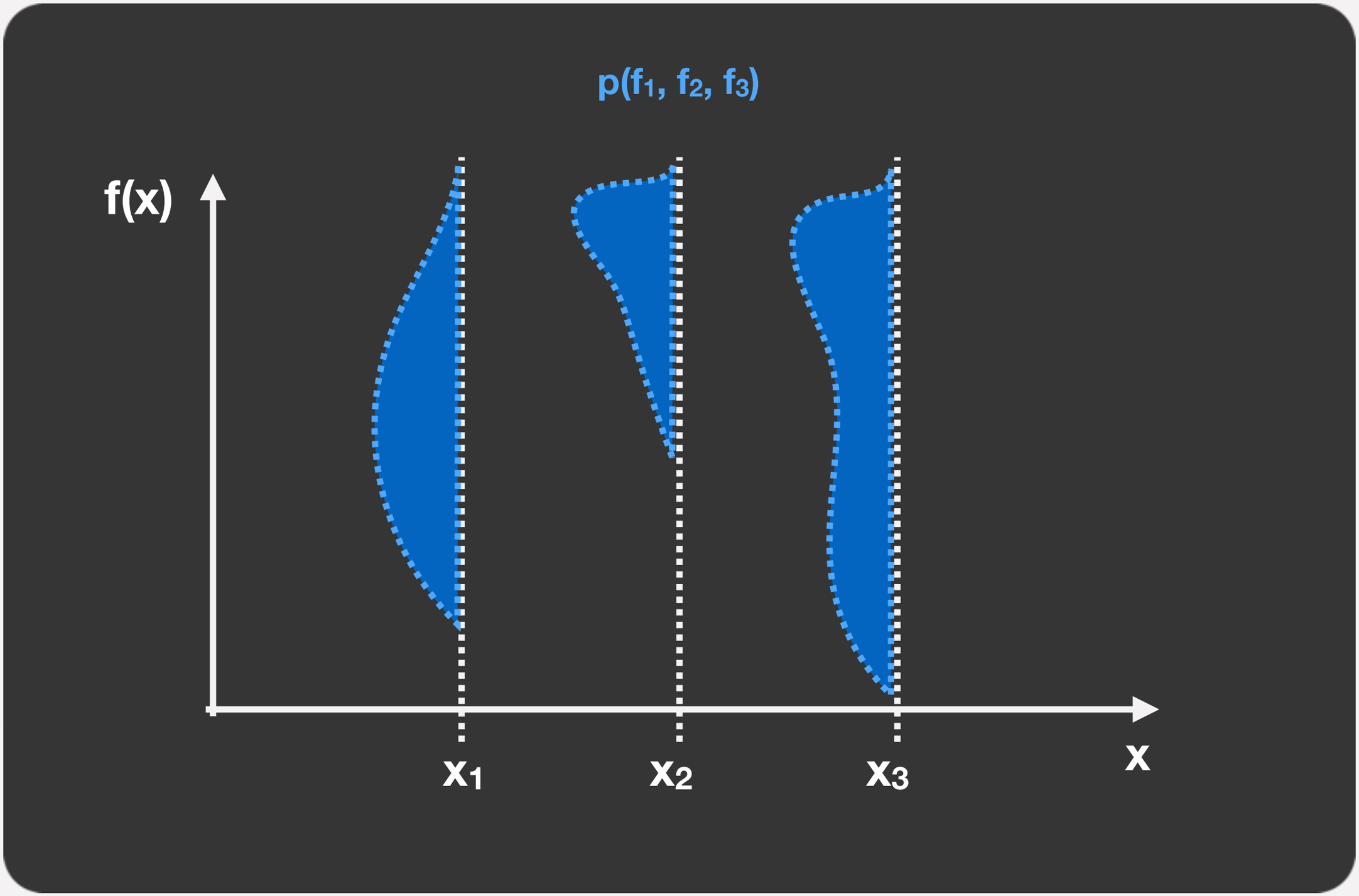
Could formulate three independent prior beliefs…



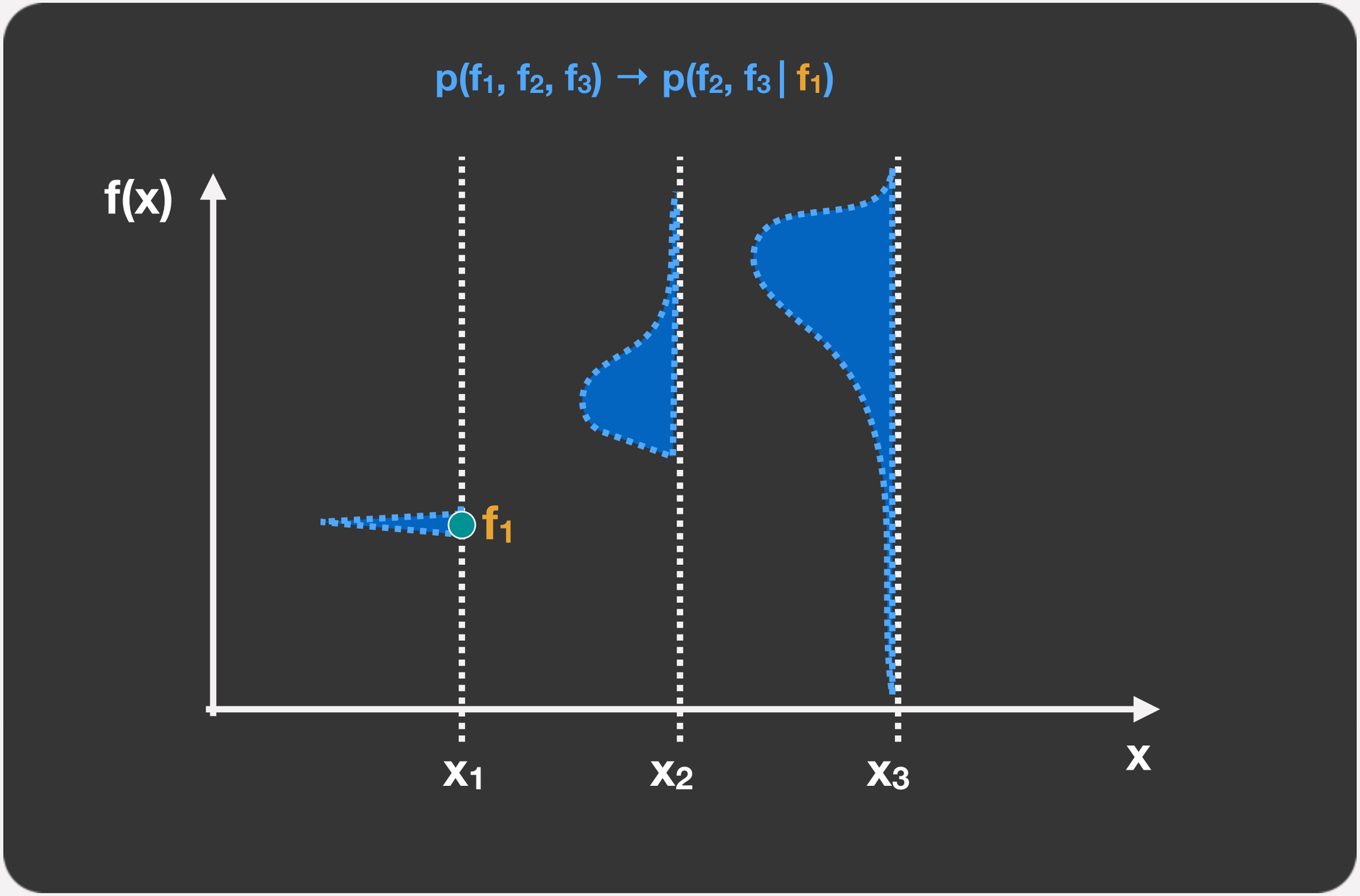
…but then learning what $\mathbf{f_1}$ is won't tell us anything about $\mathbf{f_2}$ or $\mathbf{f_3}$



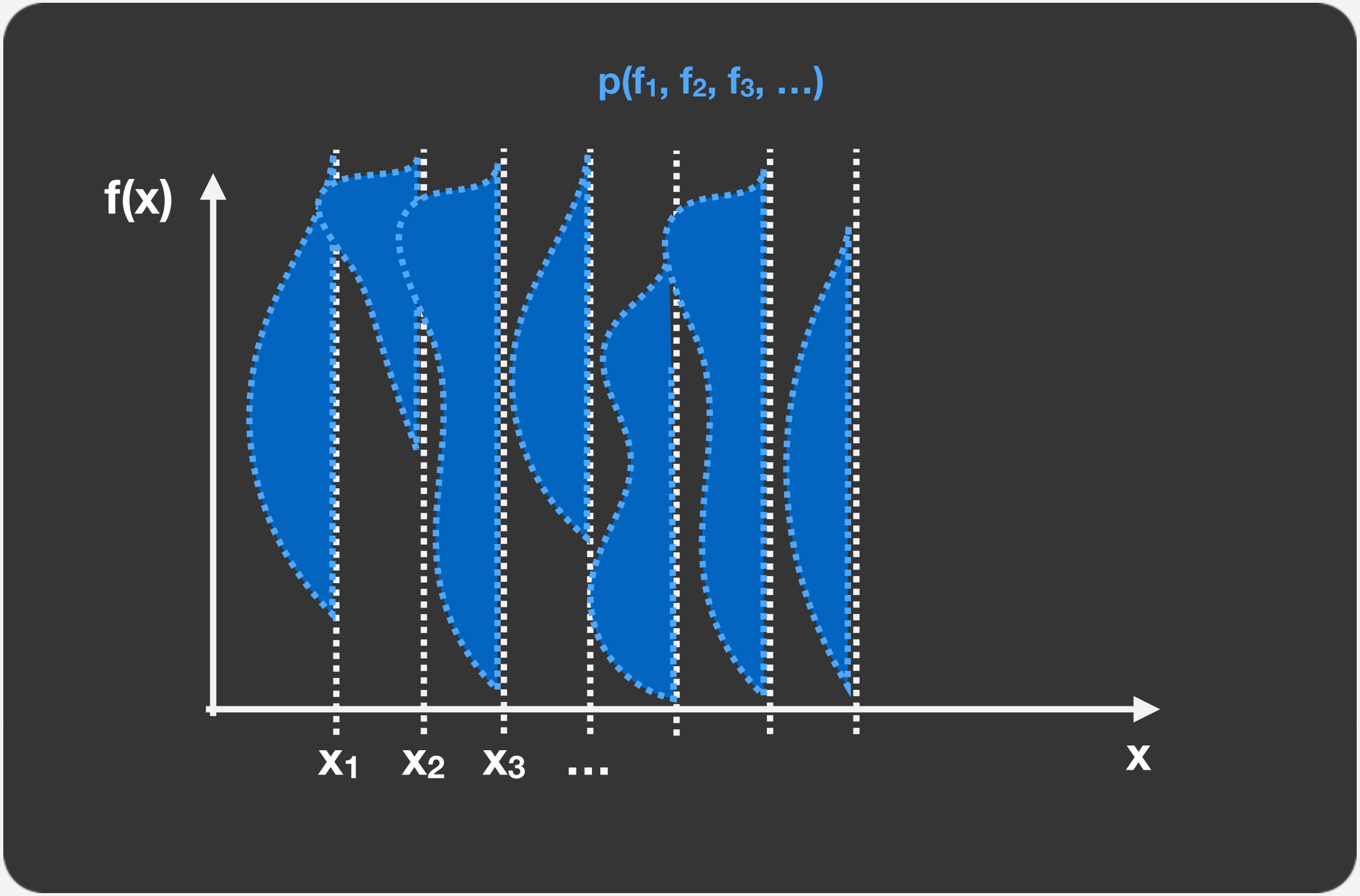
Need a **joint prior** → includes our belief about how $\mathbf{f_1}$, $\mathbf{f_2}$ and $\mathbf{f_3}$ are **correlated**



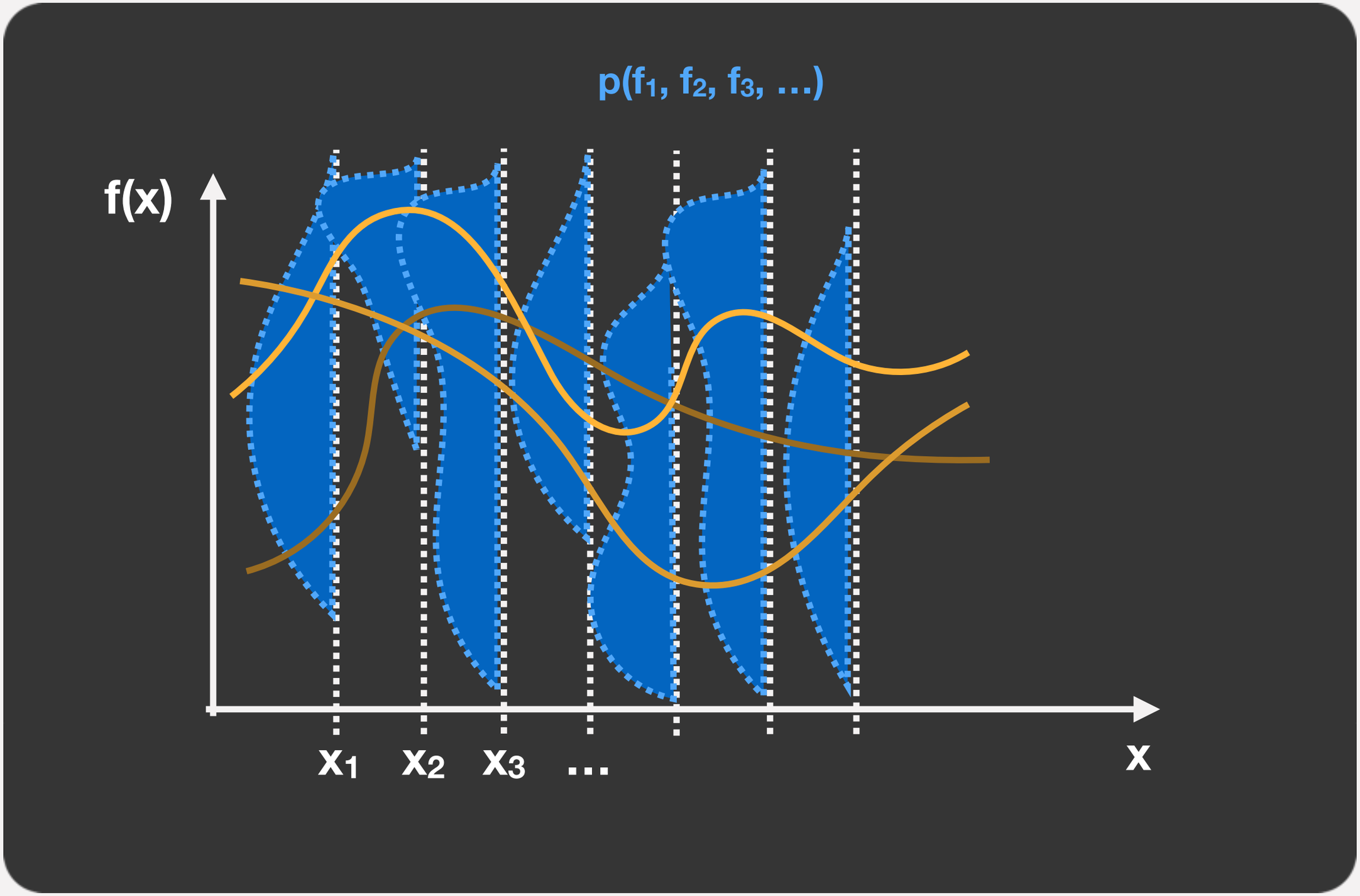
Now learning $\mathbf{f_1}$ tells us something about probable values for $\mathbf{f_2}$ and $\mathbf{f_3}$



Limit of **Δx → 0**: joint prior **p(f_1, f_2, f_3, ...)** → a **prior on function space**



Limit of **Δx → 0**: joint prior **p(f_1, f_2, f_3, ...)** → a **prior on function space**



The ideal solution:

- Incorporate all our prior knowledge about the problem in a joint prior **$p(f_1, f_2, f_3, \ldots)$** of **any form we choose**
- Perform the **full, expensive calculation of f(x)** for **some of the x-values**
- For any other x-value **x'**: obtain our best guess for the corresponding **f'** in the form of the posterior **$p(f' \mid f_1, f_2, \ldots)$**

But doing this with arbitrary prior pdfs is not practically feasible…

3. The pragmatic solution

Gaussian processes

- **Gaussian process:** an infinite set of variables such that for any finite collection of these variables, the **joint pdf is a multivariate Gaussian distribution**
- Defined by **a mean** and **a covariance matrix**
- The multivariate Gaussian has some very important properties:
  - **marginalising** (intergrating) out some variables **gives another Gaussian**
  - **conditioning** on some variables **gives another Gaussian**

What this means for regression

- Start from joint Gaussian prior
- Generally intractable pdf calculations now reduce to **analytical expressions** for a **new mean** and **a new covariance matrix**
- *In particular:* simple, closed-form expression to get the **posterior from the prior**, **$p(f_1, f_2, \ldots, f') \rightarrow p(f' \mid f_1, f_2, \ldots)$**

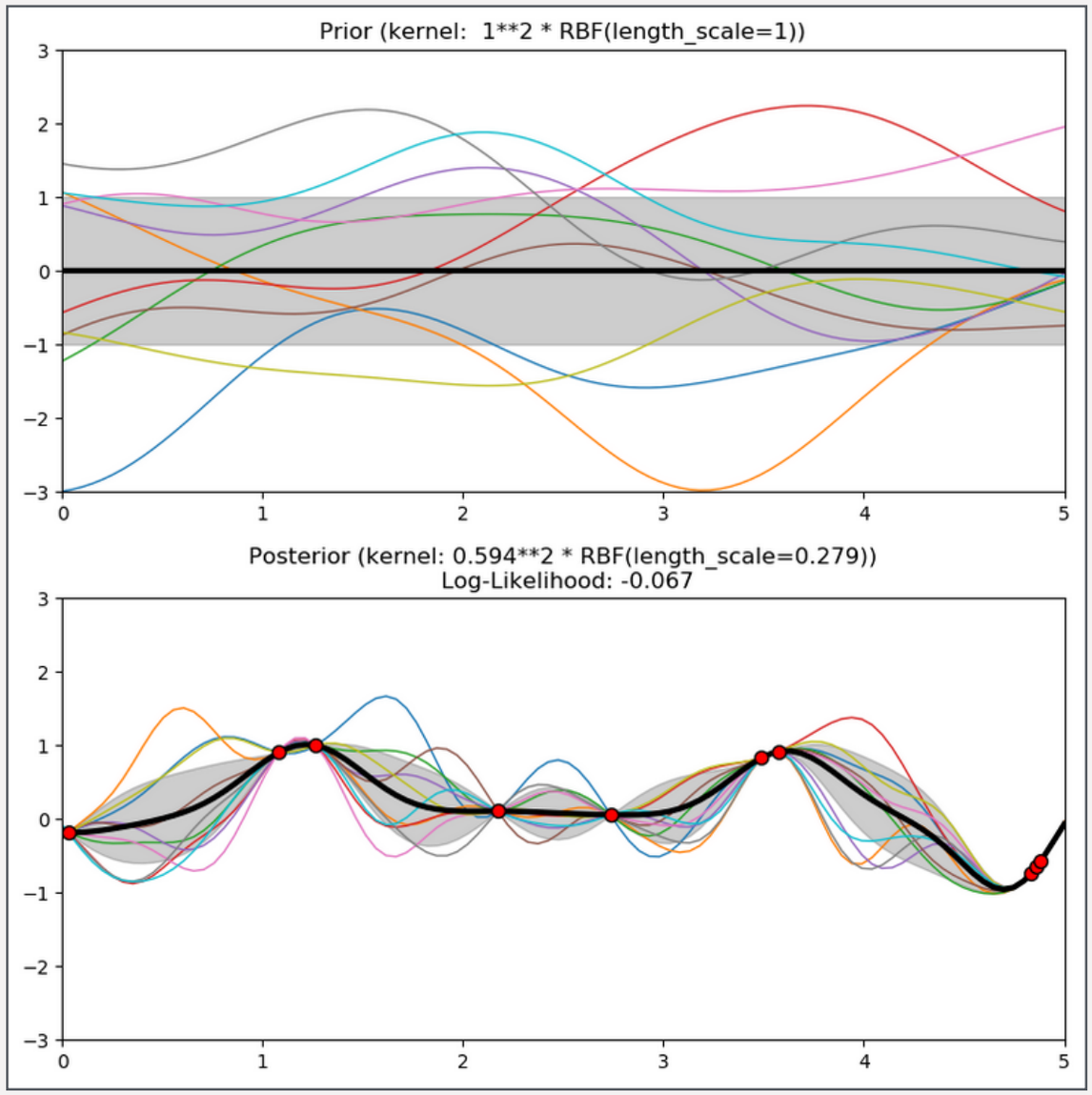
Gaussian processes regression

- Use probabilistic inference to **learn a function from data** in an **interpretable**, yet **non-parametric** framework

- *A major advantage*: the probabilistic framework **automatically provides a regression uncertainty**

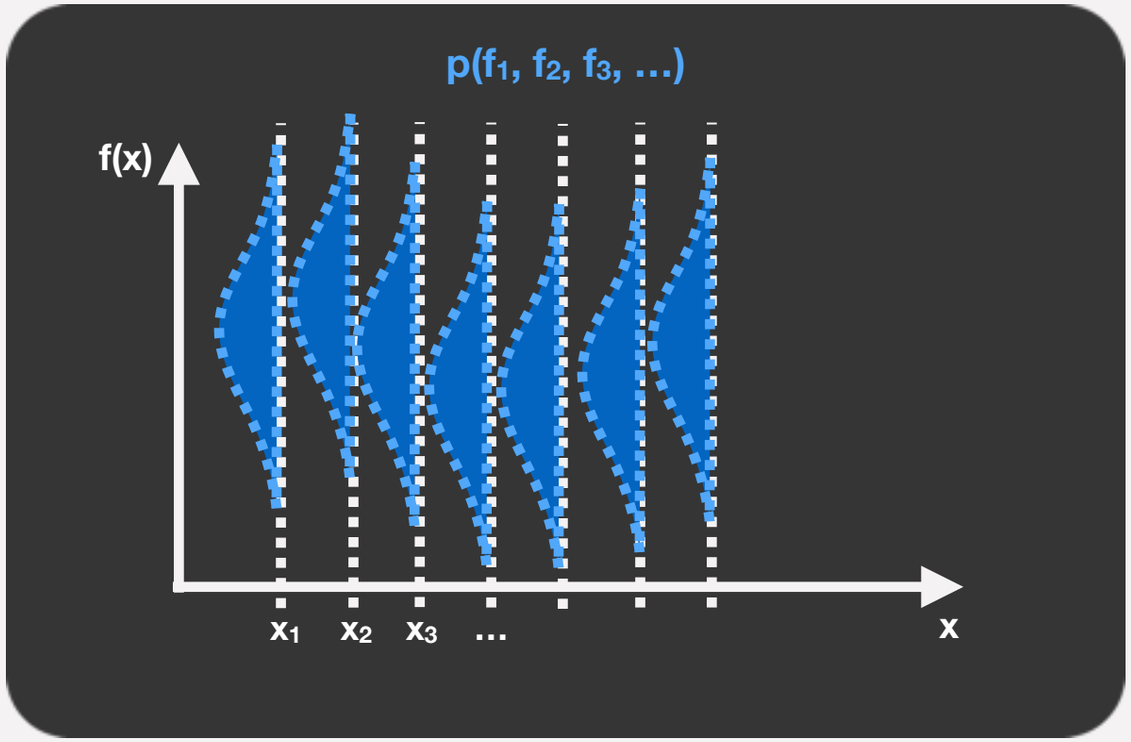
- *Main drawback*: given N data points, must invert the N x N covariance matrix
  → **can't use too large datasets**
  (but there are ways to alleviate the problem)

- Standard GP reference: Rasmussen & Williams, *Gaussian Processes for Machine Learning*



Choosing our Gaussian prior

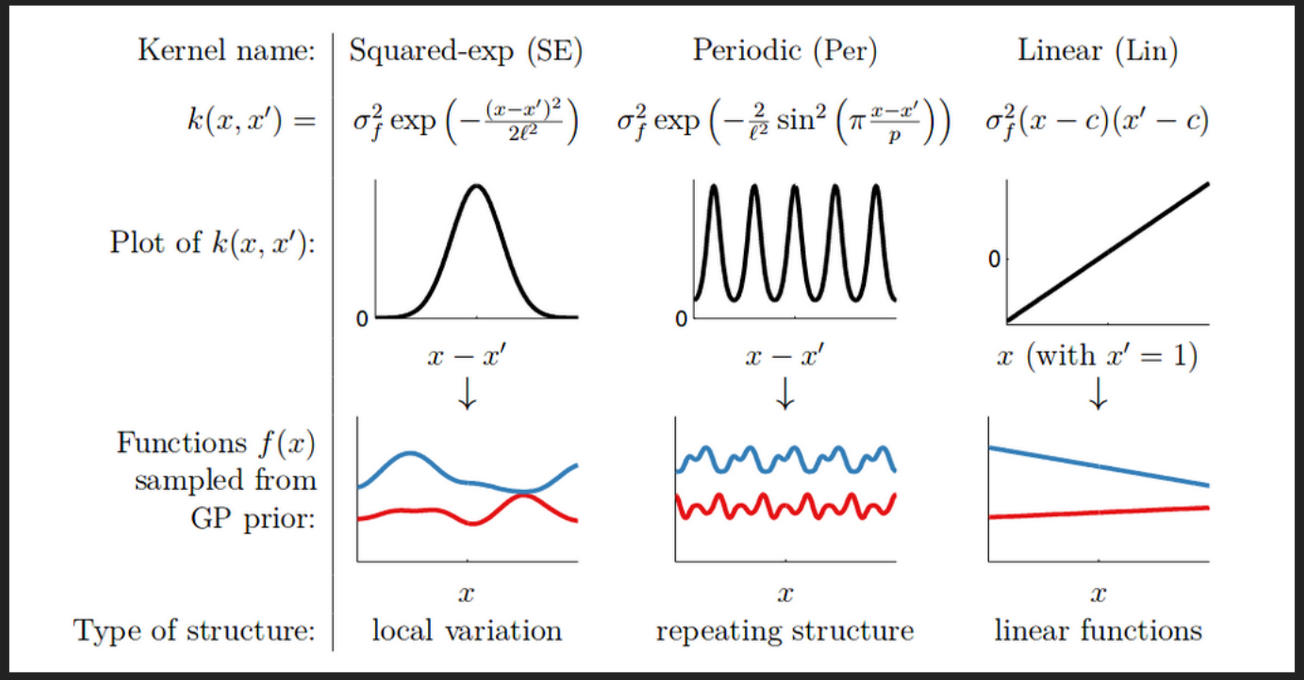
- Need to specify:
  - the means $\mathbf{E[f_i]}$
  - the covariance $\mathbf{cov(f_i,f_j)}$ for all pairs $\mathbf{(f_i,f_j)}$
- Do this indirectly by choosing
  - a mean function $\mathbf{m(x)}$, such that $\mathbf{E[f_i] = m(x_i)}$
  - a covariance (kernel) function $\mathbf{k(x,x')}$ such that $\mathbf{cov(f_i,f_j) = k(x_i,x_j)}$
- The art of GP regression:
  - Choose/design $\mathbf{k(x,x')}$ to match what you expect about the unknown $\mathbf{f(x)}$



MY KINGDOM FOR A GOOD KERNEL ...

The choice of kernel allows for great **flexibility**. But once chosen, it fixes the type of functions likely under the GP prior and determines the kind of structure captured by the model, e.g., periodicity and differentiability.

[D. Duvenaud, PhD thesis]

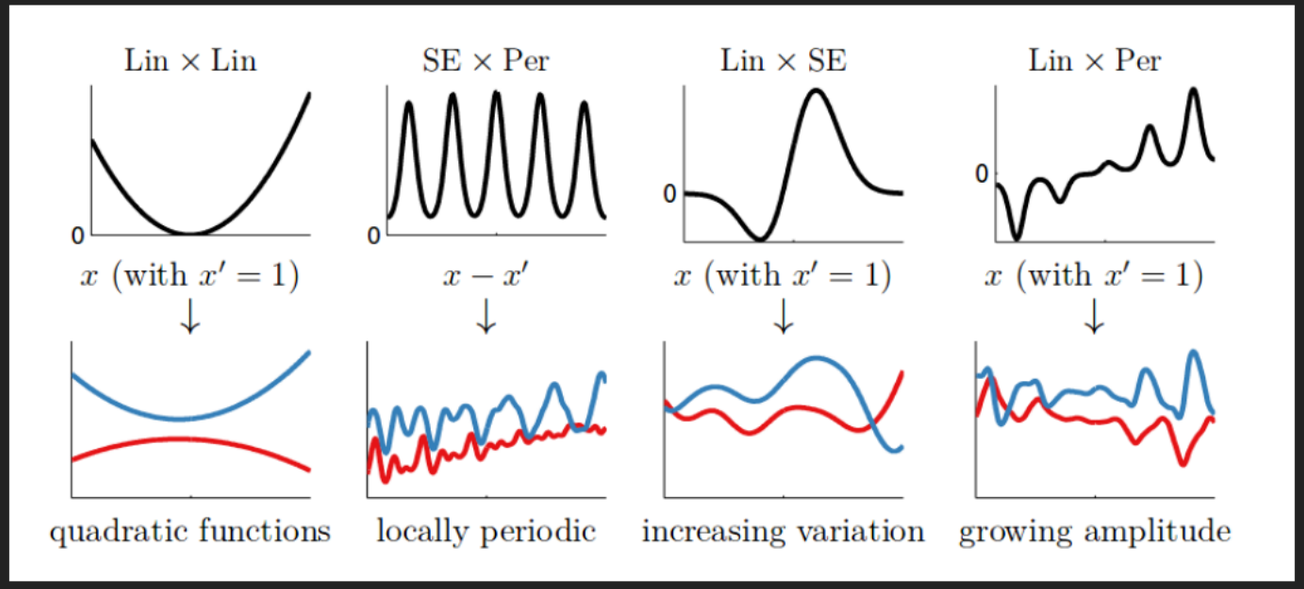


[Slide from Jeriek Van den Abeele]

MY KINGDOM FOR A GOOD KERNEL ...

The choice of kernel allows for great **flexibility**. But once chosen, it fixes the type of functions likely under the GP prior and determines the kind of structure captured by the model, e.g., periodicity and differentiability.

[D. Duvenaud, PhD thesis]



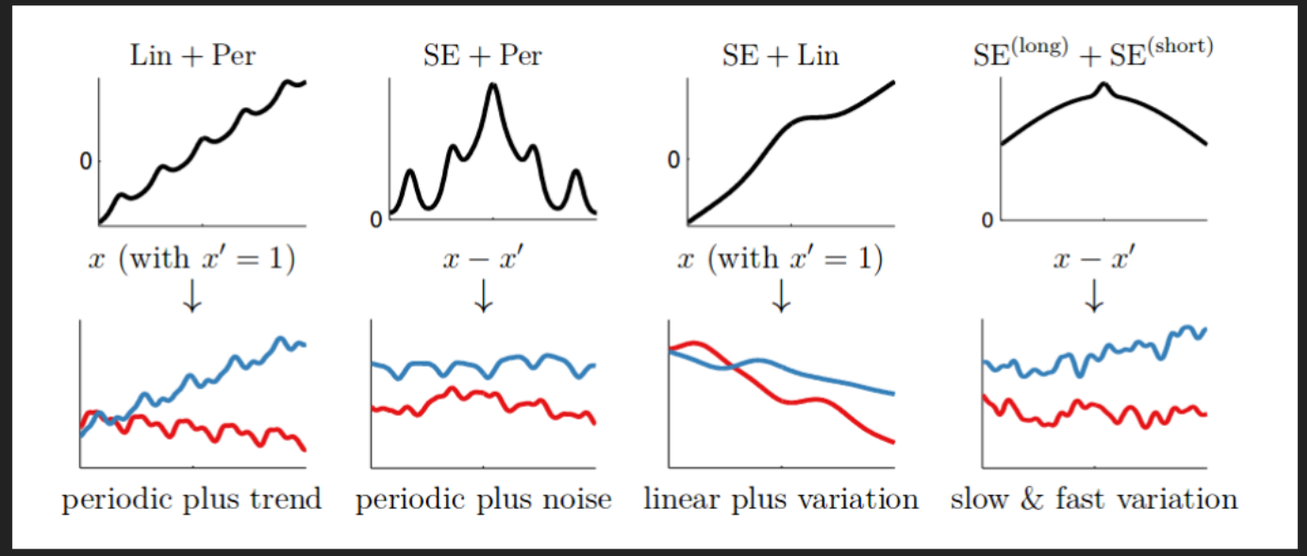
14

[Slide from Jeriek Van den Abeele]

MY KINGDOM FOR A GOOD KERNEL ...

The choice of kernel allows for great **flexibility**. But once chosen, it fixes the type of functions likely under the GP prior and determines the kind of structure captured by the model, e.g., periodicity and differentiability.

[D. Duvenaud, PhD thesis]



15

[Slide from Jeriek Van den Abeele]

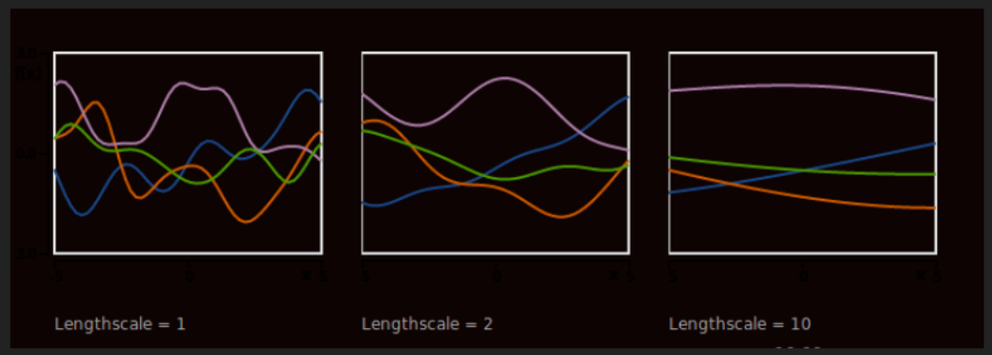
MY KINGDOM FOR A GOOD KERNEL ...

For our multi-dimensional case of cross-section regression, we get good results by multiplying **Matérn** ($\nu = 3/2$) kernels over the different mass dimensions:

$$k_s(\vec{x}, \vec{x}') = \prod_{d=1}^{D} \sigma_d^2 \left(1 + \sqrt{3}\frac{|x'_d - x_d|}{l_d}\right) \exp\left(-\sqrt{3}\frac{|x'_d - x_d|}{l_d}\right)$$

This is an **anisotropic, stationary** kernel. It allows for functions that are less smooth than with the standard squared-exponential kernel.

The different lengthscale parameters l_d lead to **automatic relevance determination** for each feature: short-range correlations for important features over which the latent function varies strongly.



[Deisenroth+, unpublished]

16

[Slide from Jeriek Van den Abeele]

The covariance function hyperparameters

- Typically don't choose a fully specified **k(x, x')**, but rather some parameterised function **k(x, x' ; θ)**

- **To be fully consistent (Bayesian):** introduce prior **p(θ)** and marginalise over **θ** to obtain the posterior

$$p(f'|f_1, f_2, \ldots) \propto \int p(f'|f_1, f_2, \ldots, \theta) p(f_1, f_2, \ldots|\theta) p(\theta)\, d\theta$$

  …but this is computationally very expensive

- **Common approach:** Fix hyperparameters by **maximising the log-likelihood**

$$\log p(f_1, f_2, \ldots|\theta)$$

  Conceptually: adjust your prior until the belief it assigned to the known function values («training points») is maximised.

4. The result

xsec: the cross-section evaluation code

- A new **Python tool**
- **Pre-trained GPs** that evaluate BSM production cross-sections at **NLO**
- First version: BSM = SUSY
- All **strong SUSY** cross-sections in the **MSSM-24** for LHC @ 13 TeV
- Provides **pdf, scale** and α_s uncertainties, in addition to the subdominant GP regression uncertainty
- pip installable, source code on github

README.md

xsec: the cross-section evaluation code

arxiv 2006.16273 | release v1.0.2 | pypi v1.0.2 | build passing | python 2.7 | 3 | code style black | license GPL-3.0

`xsec` is a tool for fast evaluation of cross-sections, taking advantage of the power and flexibility of Gaussian process regression.

For a detailed description of the methodology, validation and instructions for usage, see https://arxiv.org/abs/2006.16273.

Installation

Requirements

`xsec` is compatible with both Python 2.7 and 3. The following external Python libraries are required to run `xsec`:

- setuptools 20.7.0 or later
- numpy 1.16 or later
- scipy 1.0.0 or later
- joblib 0.12.2 or later
- dill 0.3.2 or later
- pyslha 3.2.3 or later

`pip` installation

`xsec` can be installed from PyPI using

```
pip install xsec
```

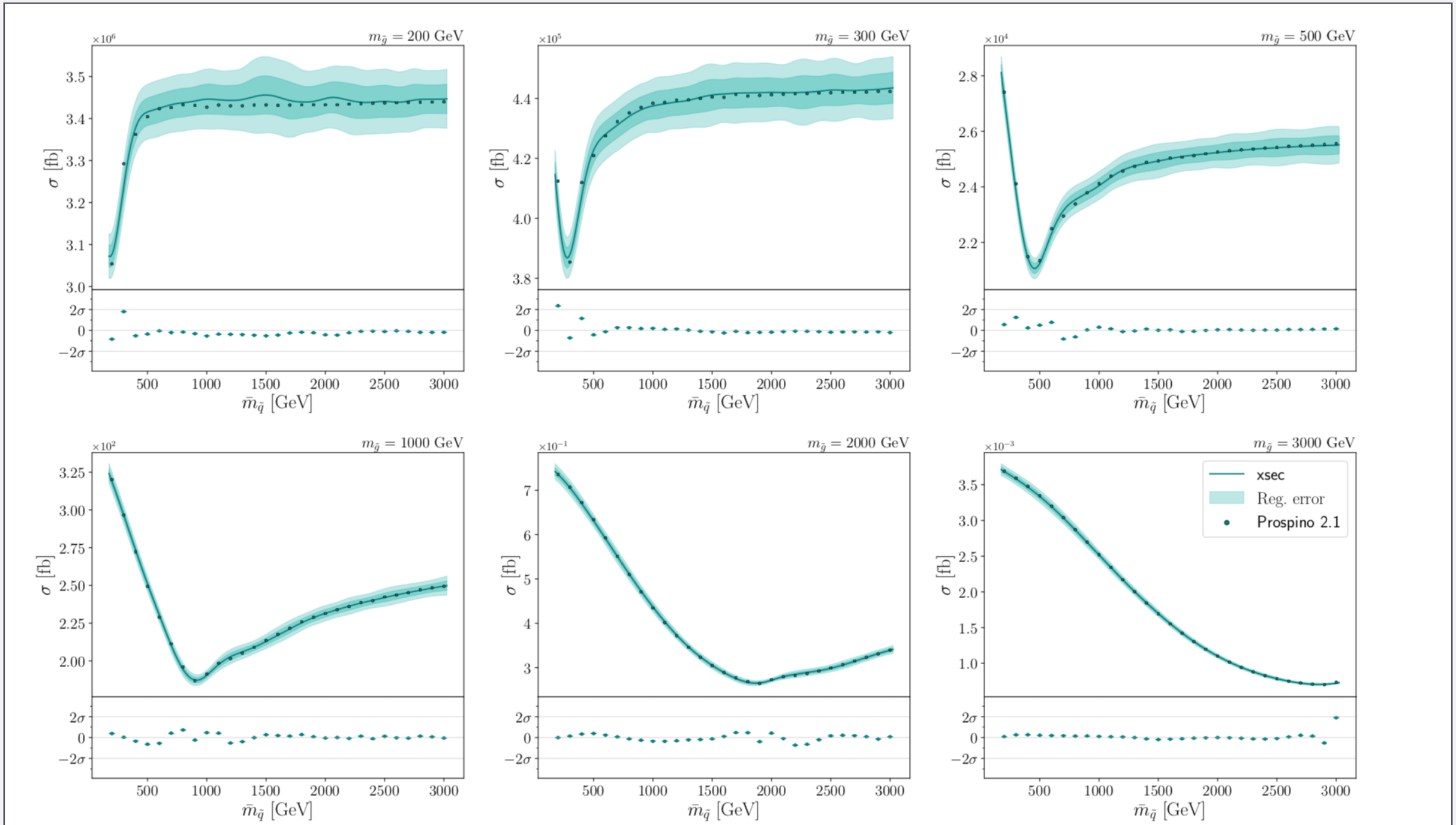
Optionally with the `--user` flag if your IT department is mean.

Alternatively, you can clone the repo from GitHub and install from there:

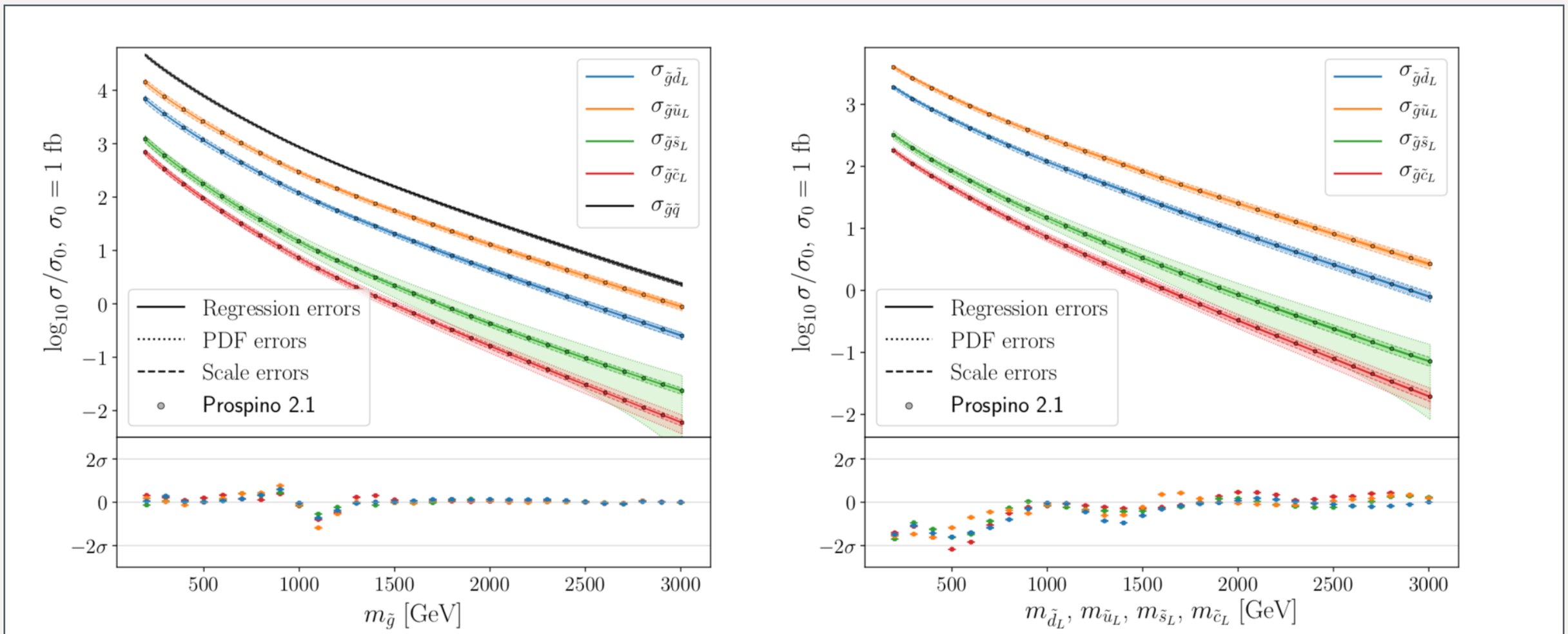
```
git clone https://github.com/jeriek/xsec.git
pip install ./xsec
```

A. Buckley, AK, A. Raklev, P Scott, J.V. Sparre, J. Van den Abeele, I. A. Vazquez-Holm
[arXiv:2006.16273]
github.com/jeriek/xsec

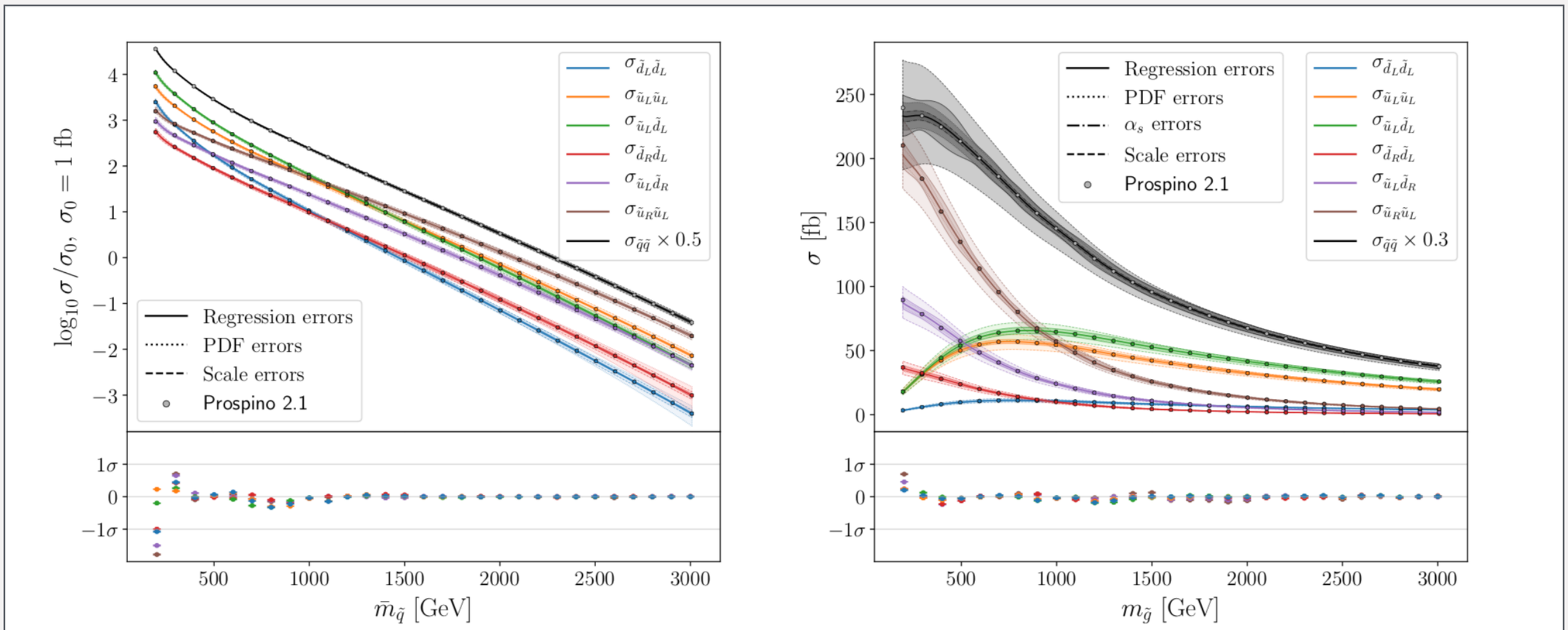
gluino—gluino



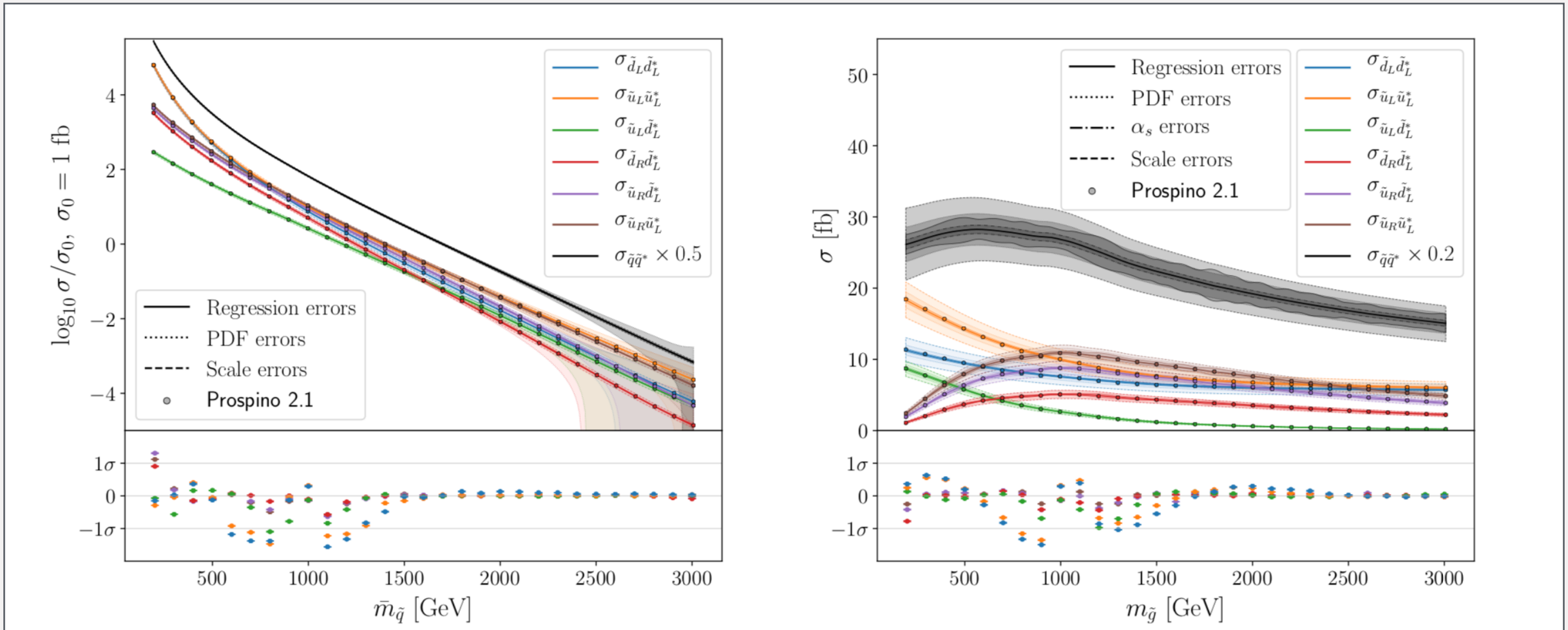
gluino—squark



squark—squark



squark—antisquark



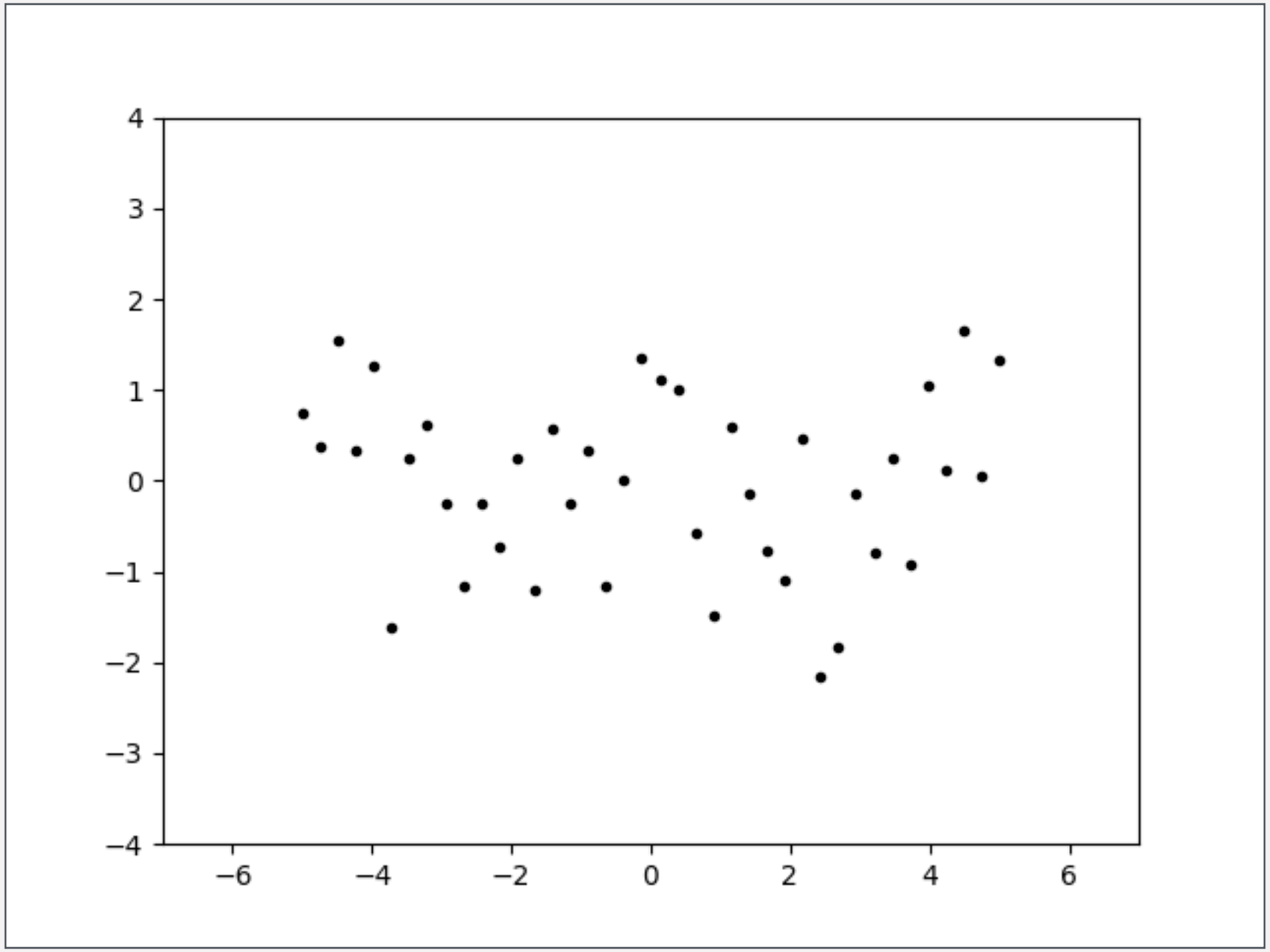
Summary

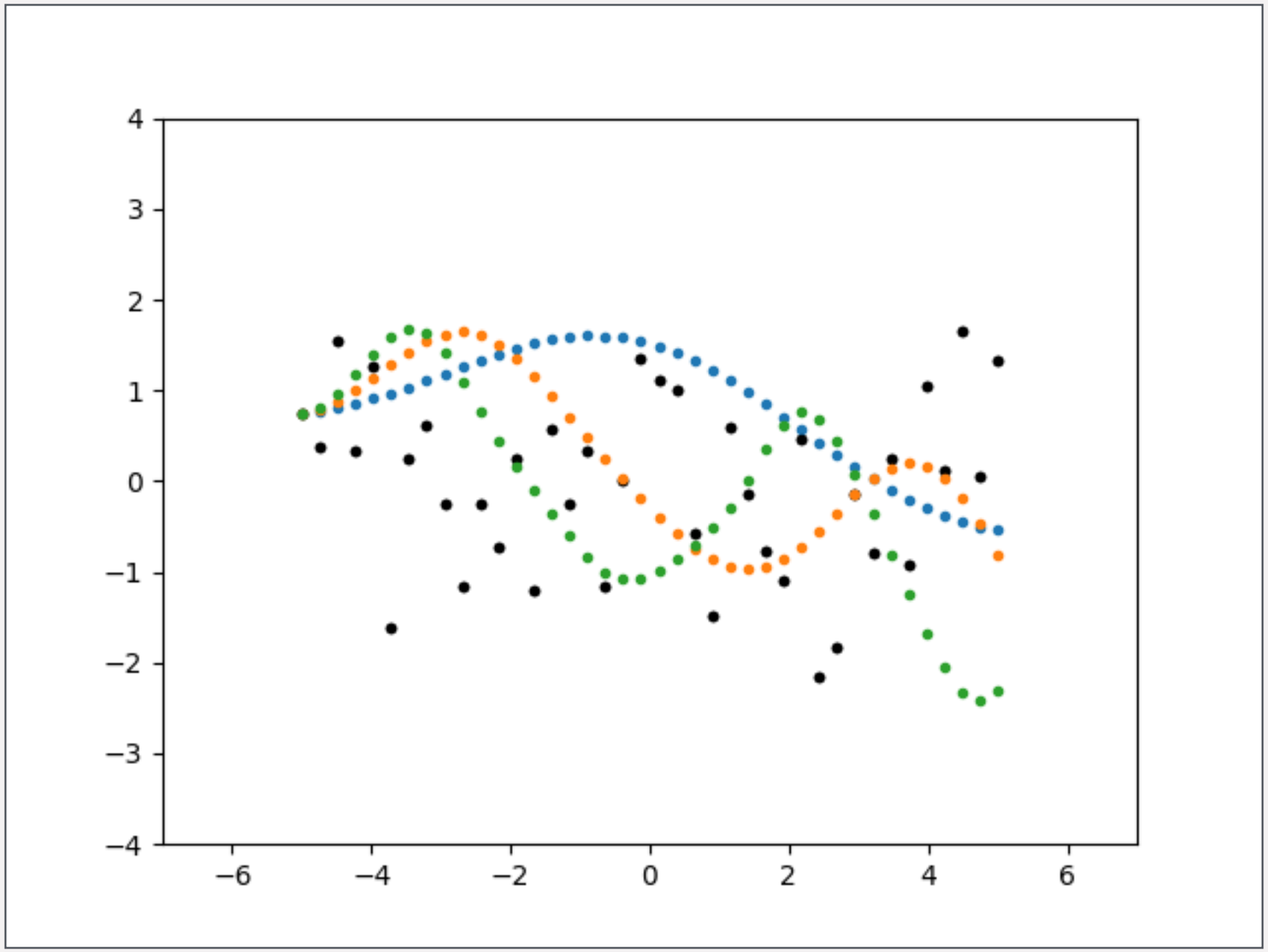
- **Theory space is huge** — need **fast** ways to evaluate theory predictions
- One approach: **ML-based regression techniques**
- **Gaussian process regression**:
  - is a **Bayesian** approach, so it makes sense :)
  - naturally **incorporates prior knowledge** :)
  - automatically **provides regression uncertainty** :)
  - gets **complicated for large datasets** :(
- **xsec 1.0 is released** — give it a try if you need some SUSY cross-sections

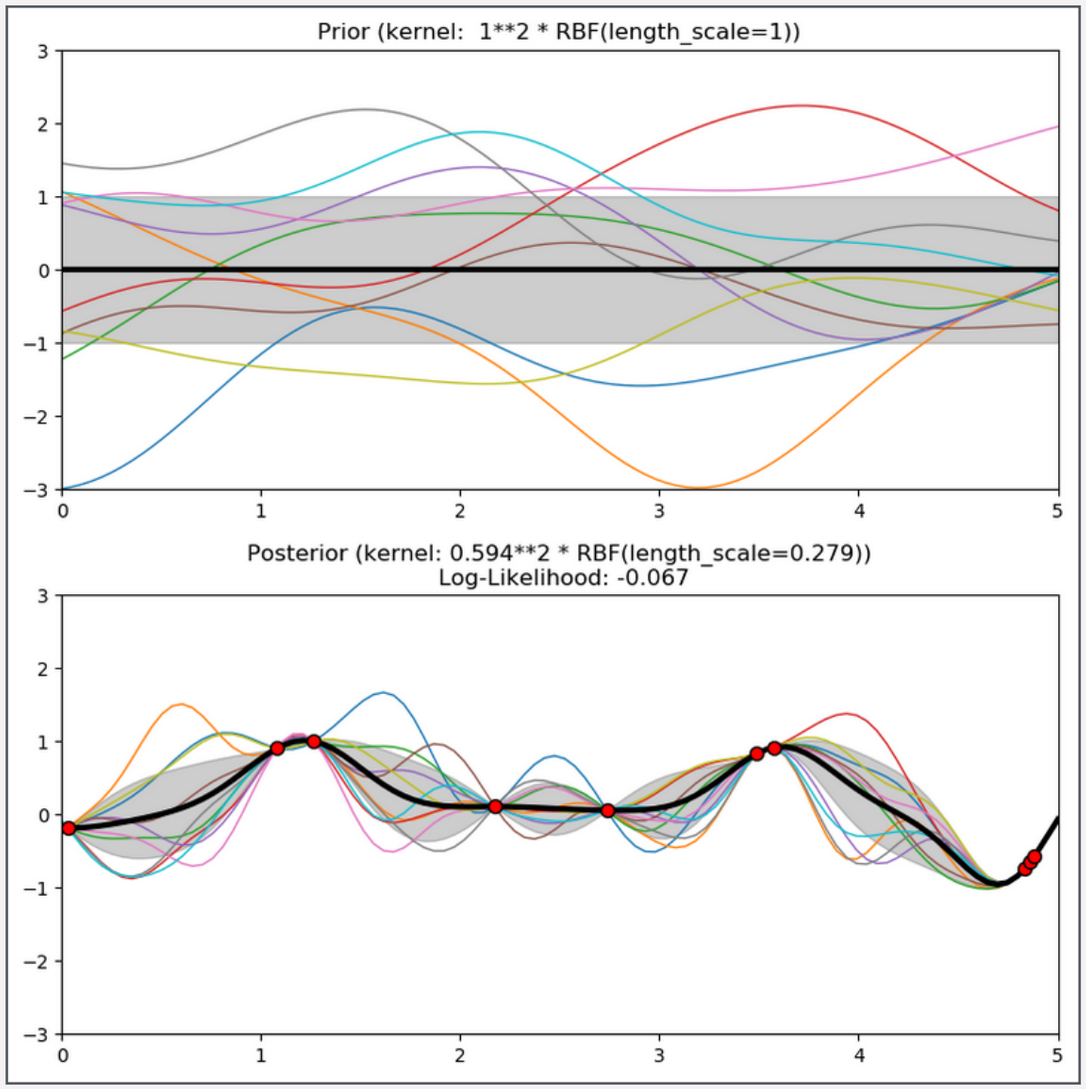
- **Theory space is huge** — need **fast** ways to evaluate theory predictions
- One approach: **ML-based regression techniques**
- **Gaussian process regression**:
  - is a **Bayesian** approach, so it makes sense :)
  - naturally **incorporates prior knowledge** :)
  - automatically **provides regression uncertainty** :)
  - gets **complicated for large datasets** :(
- **xsec 1.0 is released** — give it a try if you need some SUSY cross-sections

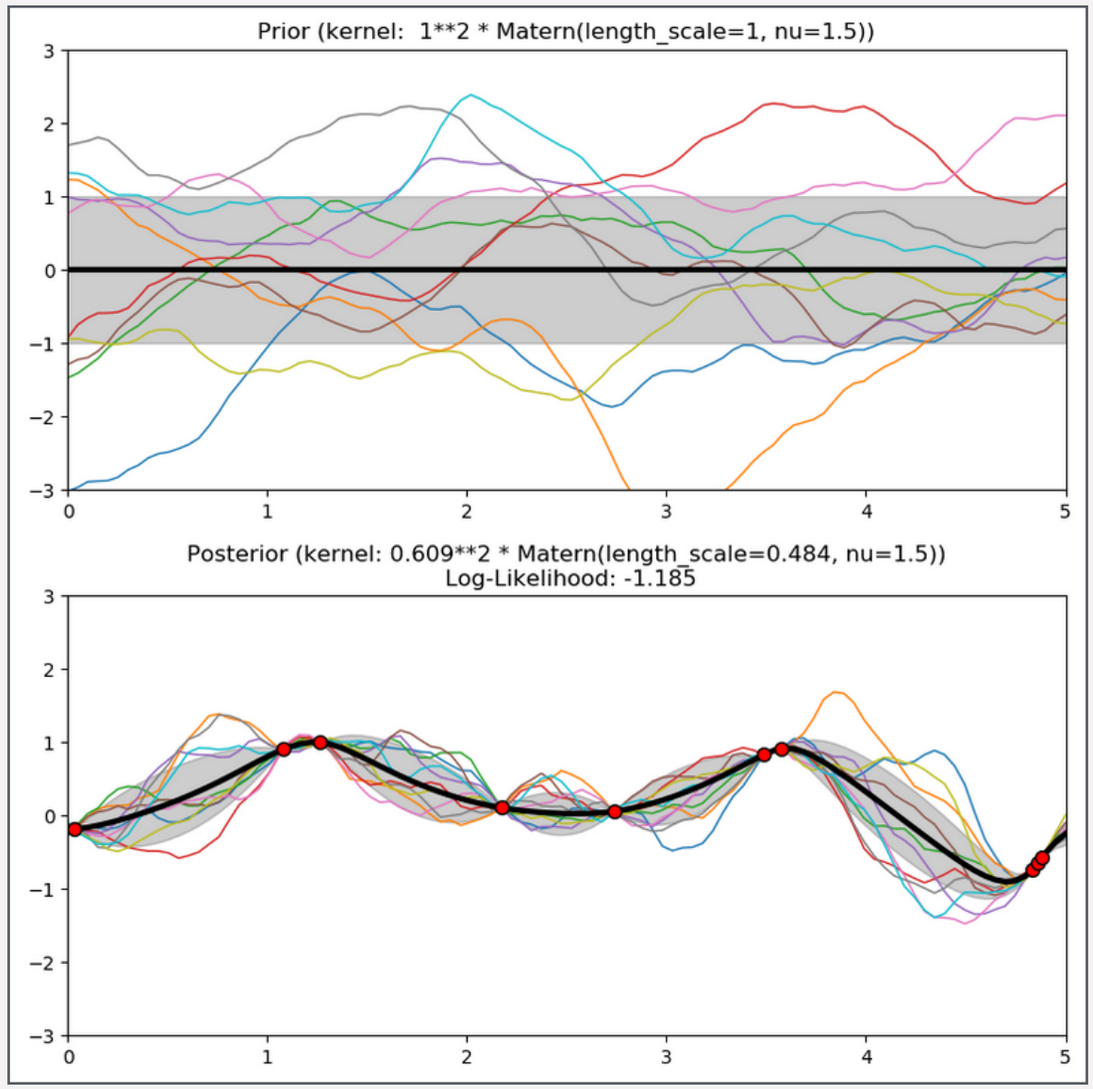
Thank you!

Bonus content


4
3
2
1
0
−1
−2
−3
−4
−6
−4
−2
0
2
4
6


4
3
2
1
0
−1
−2
−3
−4
−6
−4
−2
0
2
4
6


Prior (kernel: 1**2 * RBF(length_scale=1))
Posterior (kernel: 0.594**2 * RBF(length_scale=0.279))
Log-Likelihood: -0.067

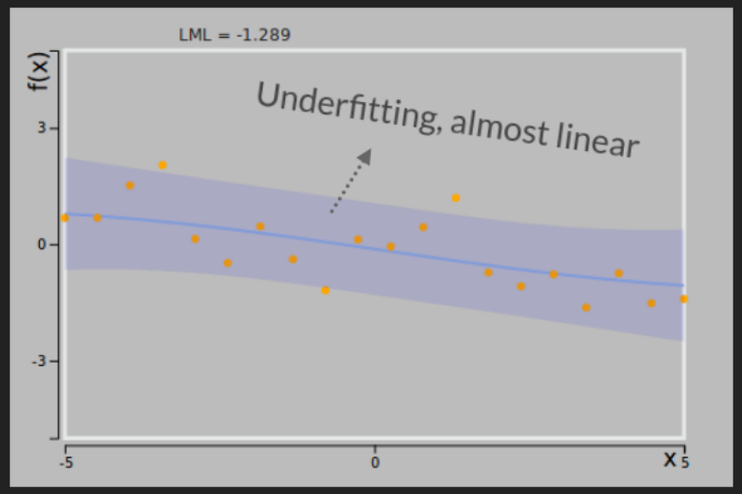

Prior (kernel: 1**2 * Matern(length_scale=1, nu=1.5))
Posterior (kernel: 0.609**2 * Matern(length_scale=0.484, nu=1.5))
Log-Likelihood: -1.185

... WITH GOOD HYPERPARAMETERS

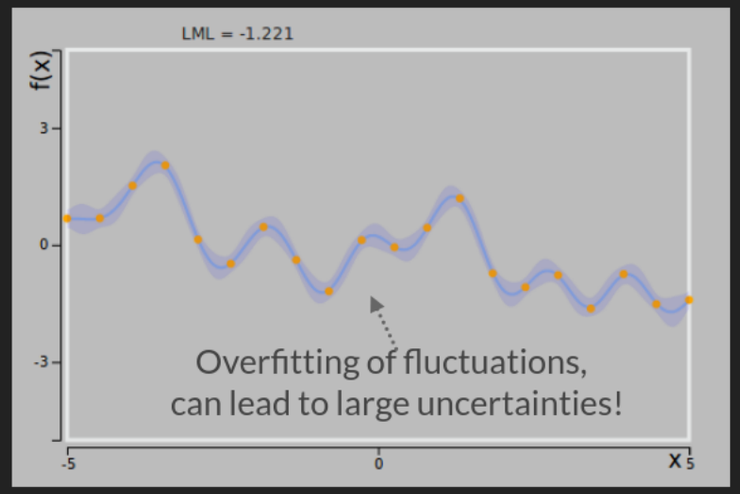
Typically, kernel hyperparameters are estimated by maximising the (log) marginal likelihood $p(\vec{y} \mid \vec{X}, \vec{\theta})$, aka the **empirical Bayes** method.

Alternative: MCMC integration over a range of $\vec{\theta}$.

Gradient-based optimisation is widely used, but can get stuck in local optima and plateaus. Multiple initialisations can help, or global optimisation methods like differential evolution.



Long lengthscale, large noise



Short lengthscale, small noise

[Deisenroth+, unpublished]

17

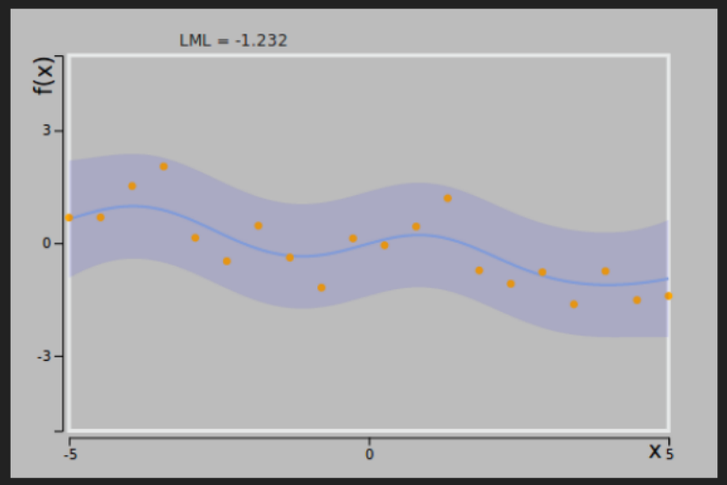
[Slide from Jeriek Van den Abeele]

... WITH GOOD HYPERPARAMETERS

Typically, kernel hyperparameters are estimated by maximising the (log) marginal likelihood $p(\vec{y} \mid \vec{X}, \vec{\theta})$, aka the **empirical Bayes** method.

Alternative: MCMC integration over a range of $\vec{\theta}$.

Gradient-based optimisation is widely used, but can get stuck in local optima and plateaus. Multiple initialisations can help, or global optimisation methods like differential evolution.



[Deisenroth+, unpublished]

Global optimum: somewhere in between

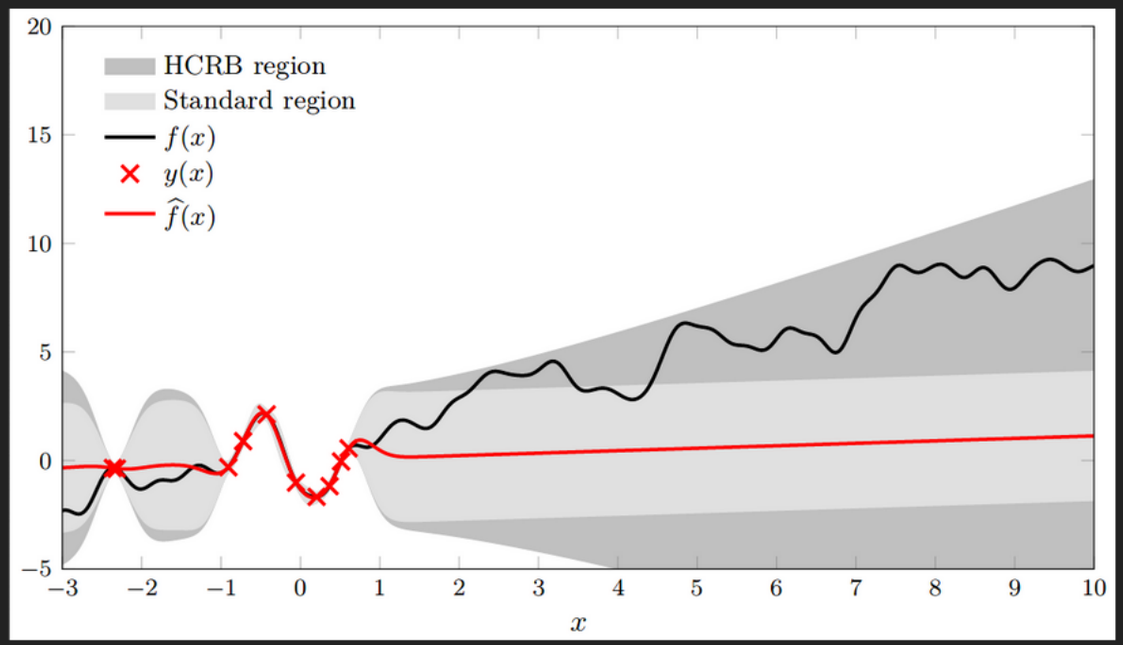
18

[Slide from Jeriek Van den Abeele]

... WITH GOOD HYPERPARAMETERS

The standard approach systematically underestimates prediction errors.

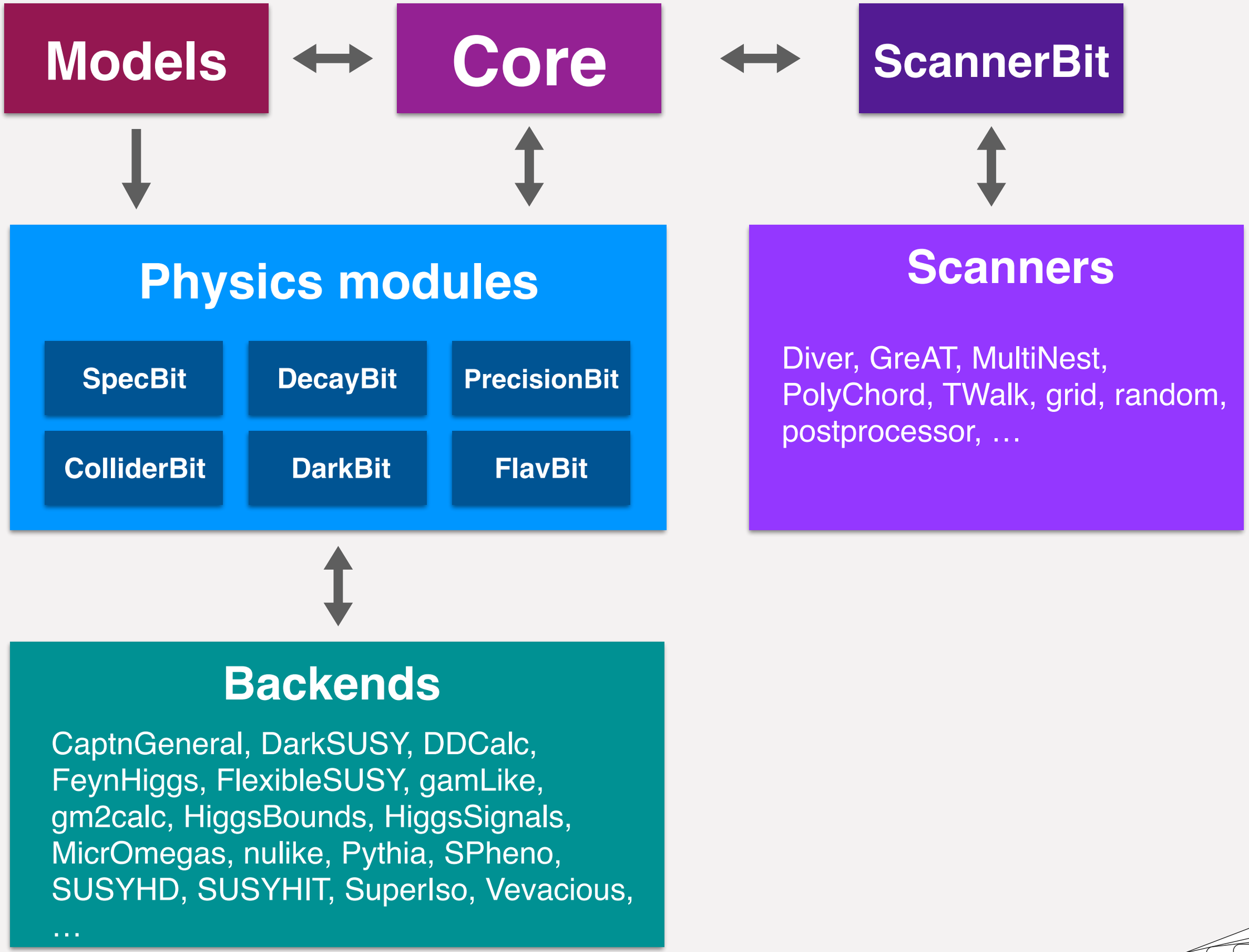
After accounting for the additional uncertainty from **learning the hyper-parameters**, the prediction error increases when far from training points.



[Wågberg+, 1606.03865]

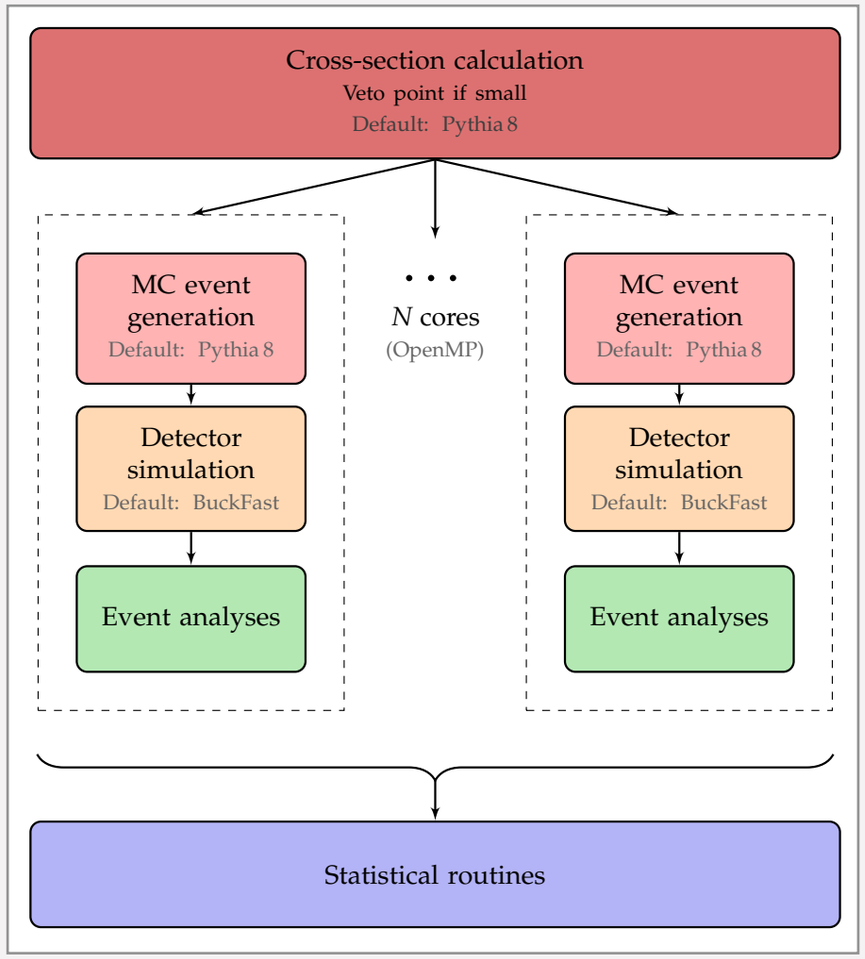
19

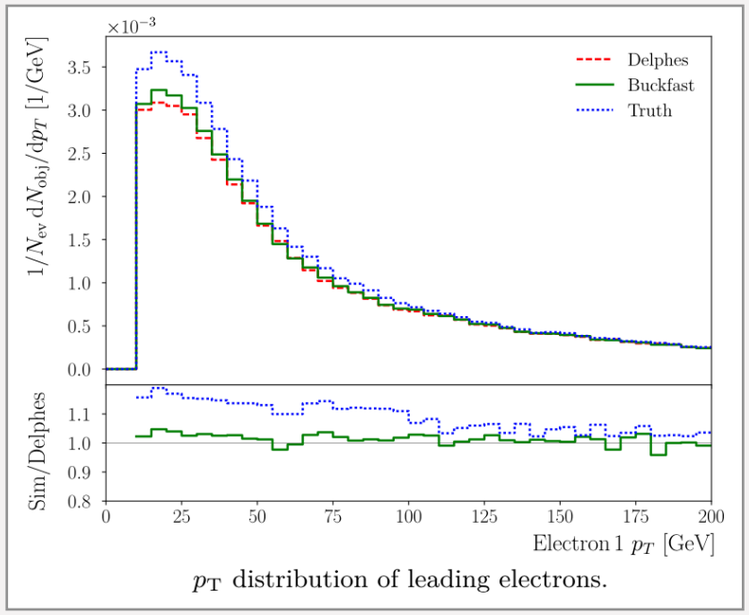
[Slide from Jeriek Van den Abeele]


Models
Core
ScannerBit
Physics modules
SpecBit
DecayBit
PrecisionBit
ColliderBit
DarkBit
FlavBit
Scanners
Diver, GreAT, MultiNest, PolyChord, TWalk, grid, random, postprocessor, …
Backends
CaptnGeneral, DarkSUSY, DDCalc, FeynHiggs, FlexibleSUSY, gamLike, gm2calc, HiggsBounds, HiggsSignals, MicrOmegas, nulike, Pythia, SPheno, SUSYHD, SUSYHIT, SuperIso, Vevacious, …

ColliderBit

- **LHC particle searches:** Full Poisson likelihood from fast MC simulation of LHC searches
  - Parallellized MC event generation and analysis loop inside ColliderBit
  - Event generation with Pythia 8
  - Fast detector simulator: BuckFast (4-vector smearing)
- **Focus on speed,** as required for use in global fits
- **Extensive list ATLAS/CMS searches** and more being added…
- **LEP limits (SUSY):** Calculate $\sigma \times BR$ and check against published limits





[arXiv:1705.07919]

The basic steps of a BSM global fit

- Choose your **BSM model and parameterisation**

- Construct the **combined likelihood function** including observables from collider physics, dark matter, flavor physics, +++

$$\mathcal{L} = \mathcal{L}_{\text{collider}} \mathcal{L}_{\text{DM}} \mathcal{L}_{\text{flavor}} \mathcal{L}_{\text{EWPO}} \cdots$$

- Use **sophisticated scanning techniques** to explore the likelihood function across the parameter space of the theory

- Test **parameter regions** in a statistically sensible way — not just single points *(parameter estimation)*

- Test **different theories the same way** *(model comparison)*

Comparing BSM theories to data

How to compare the predictions of a theory against *many* experimental results?