University of Glasgow

Gaussian Processes of Sparse Datasets

Ryan Ferguson

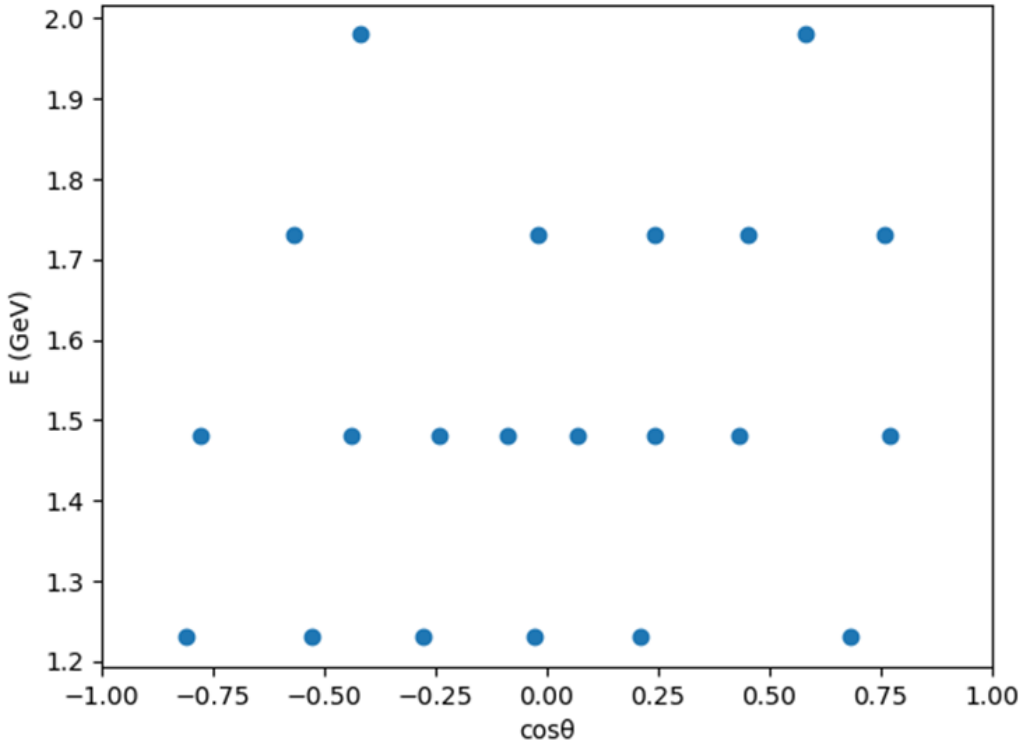
Supervisors: Dave Ireland and Bryan McKinnon

r.ferguson.3@research.gla.ac.uk

Why are we doing this?

Why?

Information from hadron data is limited by incomplete and potentially inconsistent datasets.



Current issues?

- Current hadron analyses can use data from 40+ years ago, which are unlikely to be taken again.
- These coupled channel analyses require data from different experiments which can disagree, leading to arbitrary weighting in χ^2 minimisation for theoretical modelling.
- This produces inconsistencies in the fitting of resonances which are still a matter of debate today, e.g. $\Lambda(1405)$.

Specifics of this GP model

Kernel Choice

The Radial Basis Function kernel can be used:

$$\kappa(\underline{a}, \underline{b}) = \exp\left[\sum_{i=0}^{p-1} \frac{-d(a_i, b_i)^2}{2{l_i}^2}\right]$$

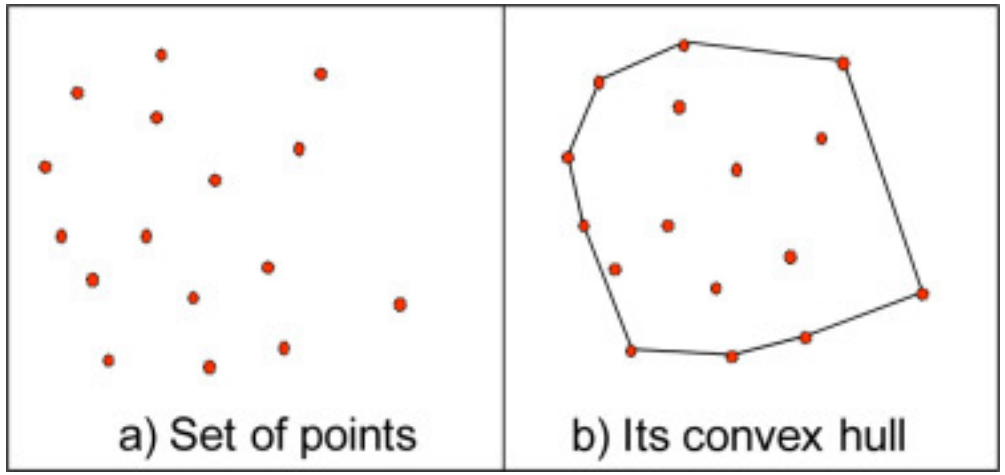
Where:

- $\underline{a}, \underline{b}$ are some vectors of length p (e.g. have p parameters)
- $d(\cdot,\cdot)$ is the Euclidean distance.
- $\underline{l}$ is a hyperparameter called the length scale. For this kernel, it is a measure of how smooth the function is.

The RBF kernel gives smooth, continuous posterior distribution which is appropriate for the cases presented here.

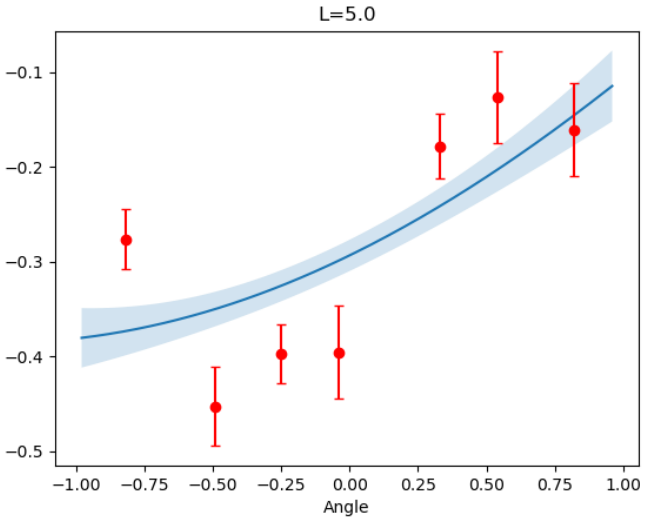
Convex Hull

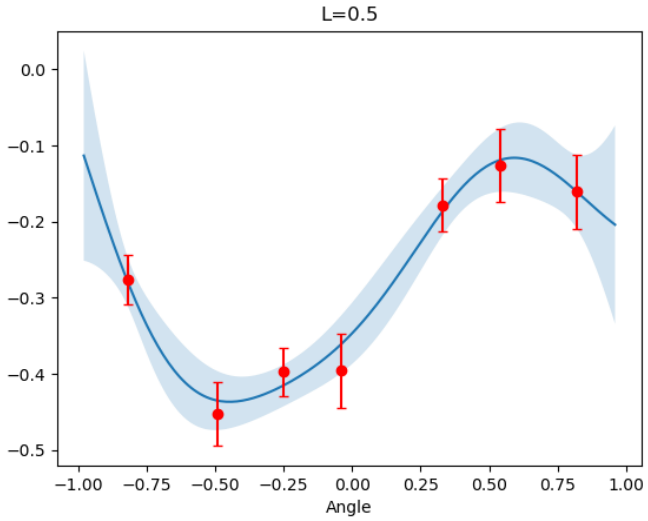
- It was found in testing that the GP performs well at interpolating but not at extrapolating.
- As such a set of discrete points of the convex hull[1] of the known datapoints is the space that the GP gives a prediction for (with resolution in each dimension chosen by the user).

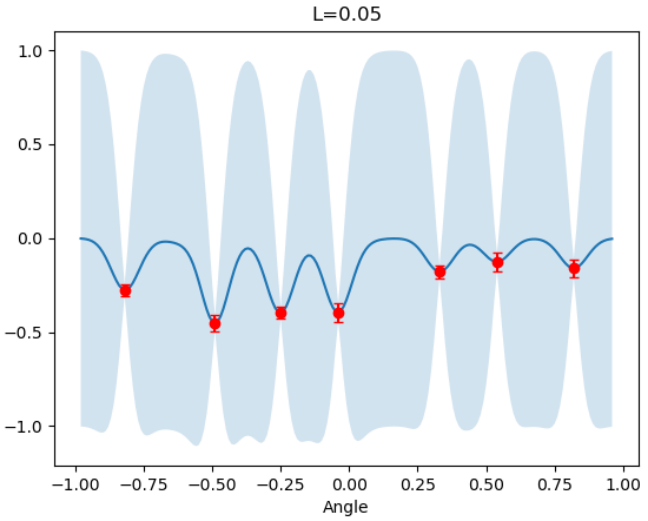


Hyperparameter Choice

The choice of length scale dictates the smoothness of our posterior and how good the overall fit is





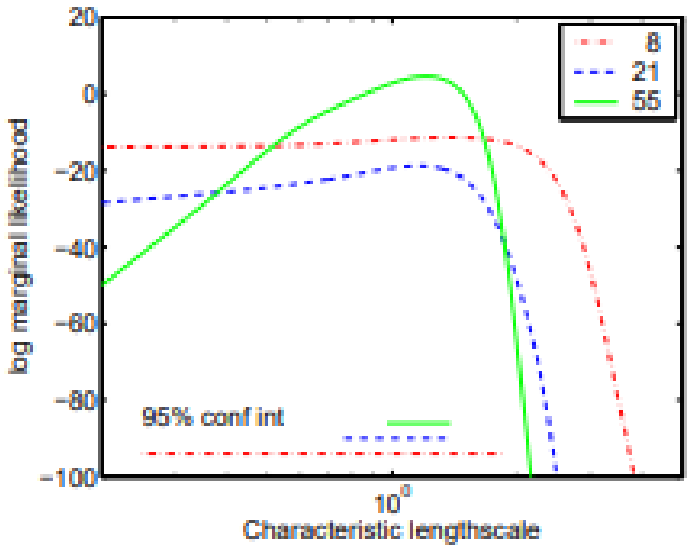


Standard Approach for Hyperparameter Optimisation

The normal approach is to use the marginal likelihood function:

$$\log p\left(y\middle|X,\vec{l}\right) = -\frac{1}{2}y^T {K_y}^{-1} y - \frac{1}{2}\log\left|K_y\right| - \frac{n}{2}\log 2\pi$$

For low number of datapoints this is not well defined[6]:

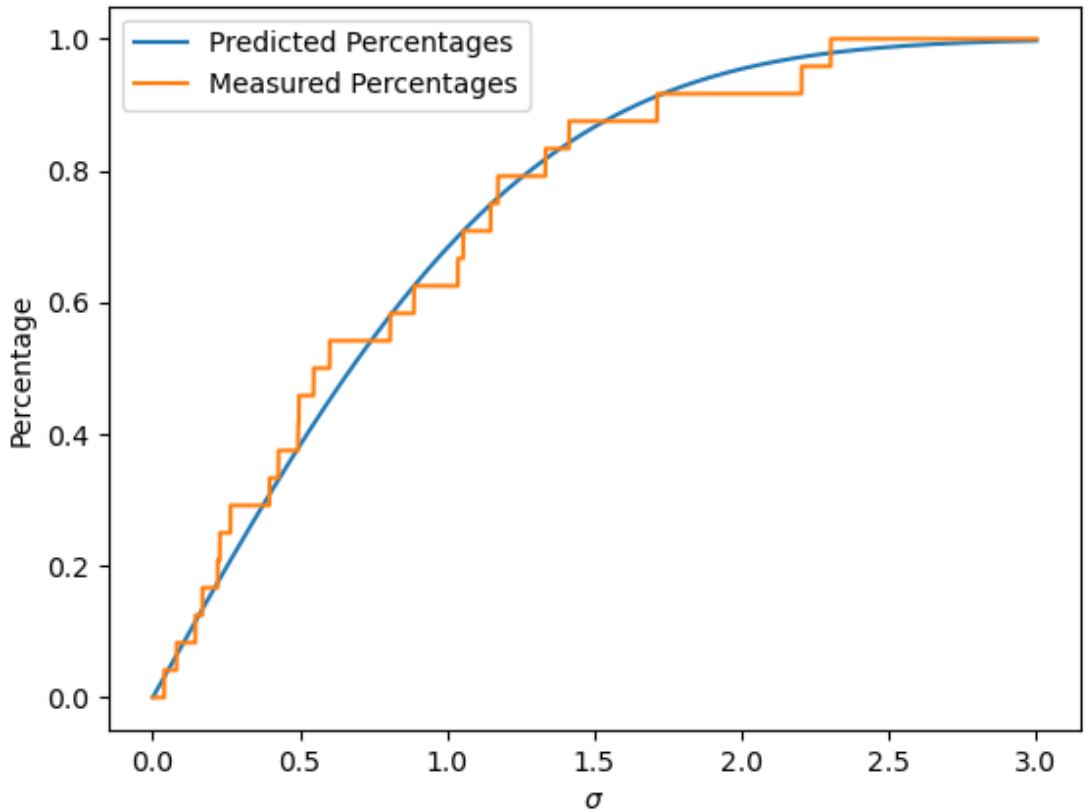


Hyperparameter Search

- The hyperparameters are measured in the same quantities as our kinematic dimensions (in this case energy and scattering angle ($\cos\theta$).

- We know from standard statistics the percentage of datapoints that we expect with different multiples of the standard deviation (e.g. 50% in 0.67σ, 68.3% in 1σ, etc.).
- For a given set of hyperparameters, the percentage of points within different multiples of the GP error (up to 3σ) from the GP mean can also be measured.
- The Wasserstein distance can be used as a measure of the similarity between the 2 CDFs.

Expected vs Measured Percents



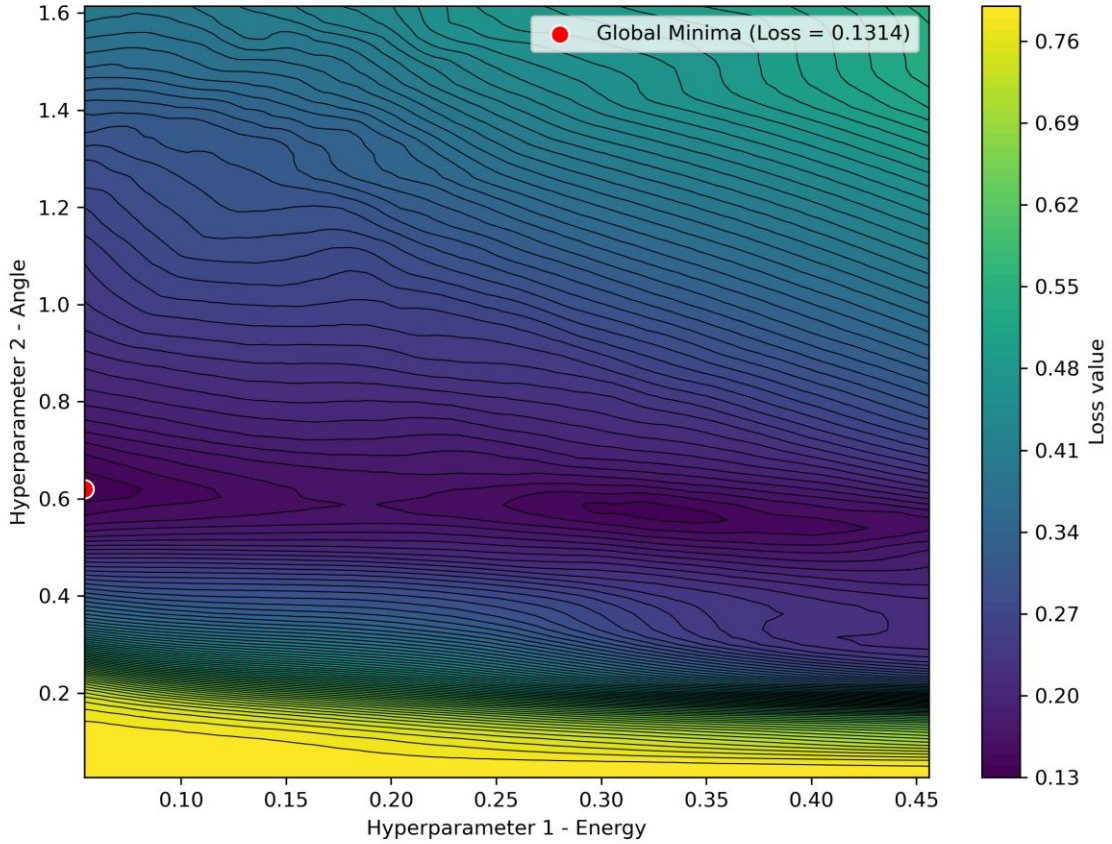
Bounds for Hyperparameter Search

- Upper – Full range (per kinematic dimension)
  - We know that the GP fails at extrapolation so want to keep within interpolating range
- Lower – Minimum distance between (sorted) adjacent datapoints.
  - Take for instance the energy hyperparameter
  - We know that we cannot infer anything smaller than the smallest distance between adjacent energy levels, (think resolution)

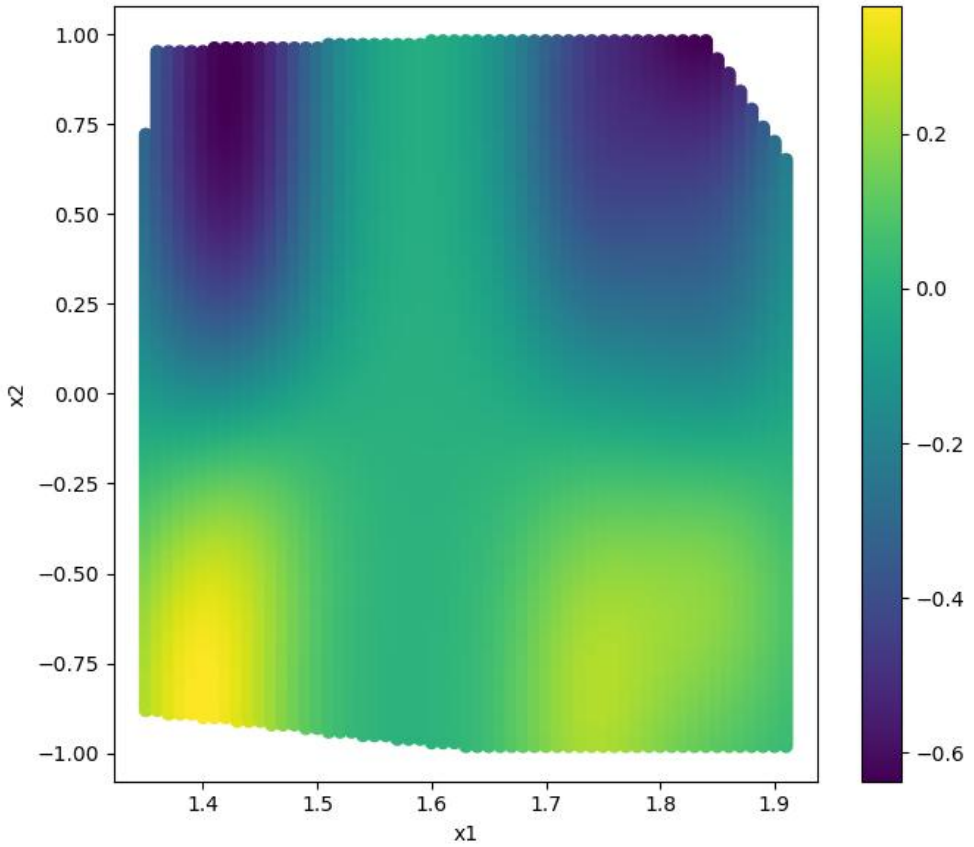
Hyperparameters on boundary

- In certain cases, the returned optimal hyperparameters sit on or close to the boundary.
- Often these still give good physical results but sometimes can return GP fits showing unphysical behaviour.
- The Wasserstein distance (or the Kolmogorov-Smirnov statistic) alone cannot be used as a suitable loss function

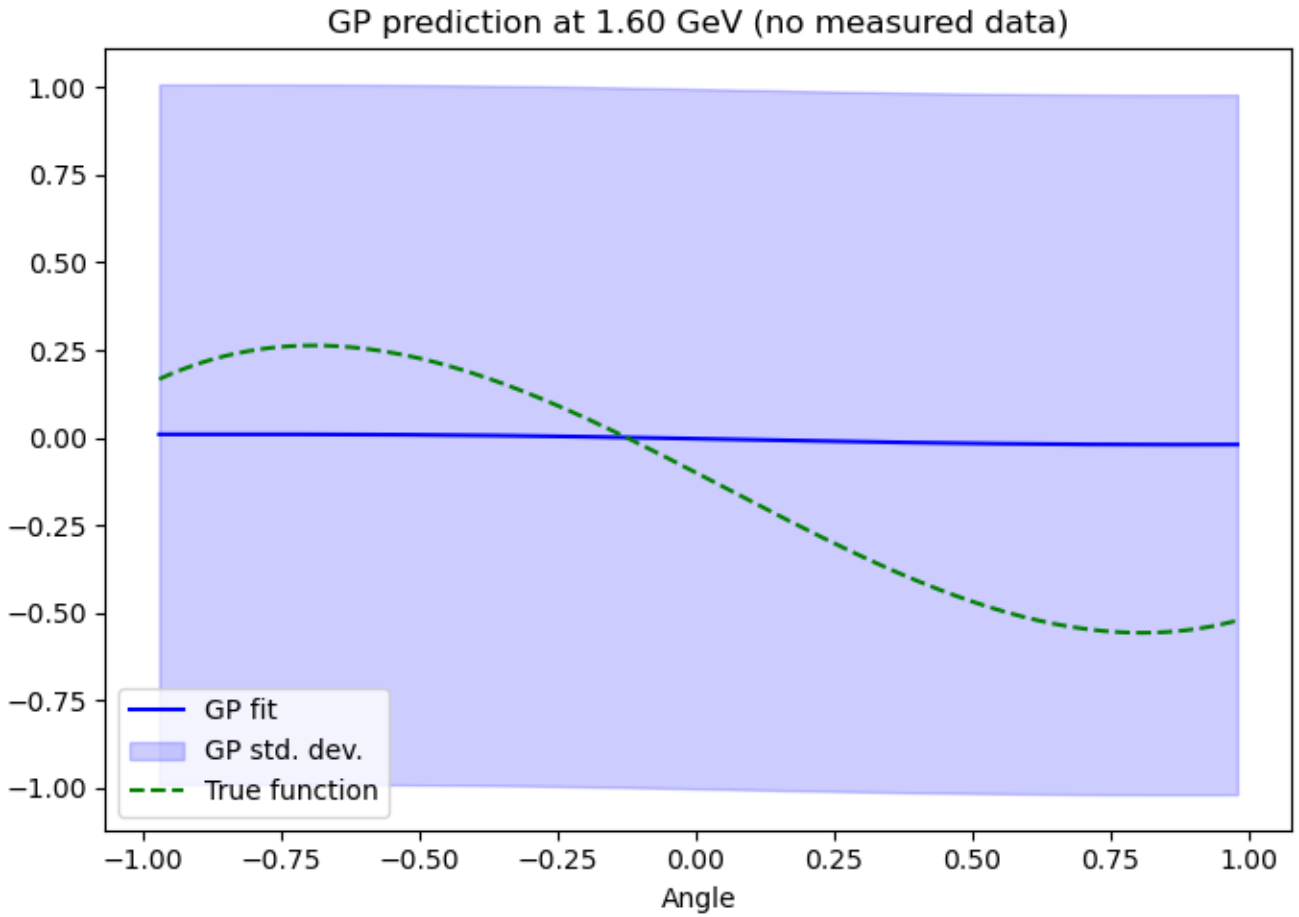
Loss surface – optimum close to boundary



Results not totally physical



Projection at E=1.6 (no measured data)



Boundary Penalty Term

- For each kinematic dim, if the prospective length scale is on any boundary assign it 1, decreasing linearly until the centre is assigned 0.
- Weight this to 1% in the final loss.
- This ensures that if there are local minima (with a similar loss to the global minima) but far from the boundary then this will be the returned optimum.
- By weighting it to only 1% it ensures that if the only viable hyperparameters lie on the boundary then these will still be the returned set.

How do we know it works?

Pseudodata

We can test the GP using some suitable pseudodata. Thus, define a 2D surface of the form, modelled on polarisation observables:

$$y_{func} = f(E_\gamma, \cos\theta) = \sum_{l=0}^{n} c_l * g_l(E_\gamma) * P_l(\cos\theta)$$

With

- $c_l \, \epsilon \, [-1,1]$ is some weight
- $g_l(E_\gamma) \sim \mathcal{N}(\mu_l, \sigma^2{}_l)$
- $P_l(\cos\theta)$ is an ordinary Legendre polynomial

In our case n=3 so we have 12 parameters.

Note also that $|y_{func}| \leq 1$.

2 Tests

A 2D known surface is generated, some points are selected and given appropriate noise and error bars. This is pseudodata which can be used to test the GP is performing as intended.

We can perform 2 tests on this:

- Number of points in different confidence intervals
- Unbiased Pull of Fitted coefficients

Points within confidence intervals

- Calculate pull: $pull = \frac{y_{func} - y_{fit}}{e_{fit}}$
- $|pull| \leq 1 \Longrightarrow y_{func} \in \left[y_{fit} - e_{fit}\,, y_{fit} + e_{fit}\right]$, i.e., the predicted point is within its uncertainty of the actual point.

- From this the total percentage of points within different confidence intervals can be calculated by scaling e_{fit} as required and repeat.

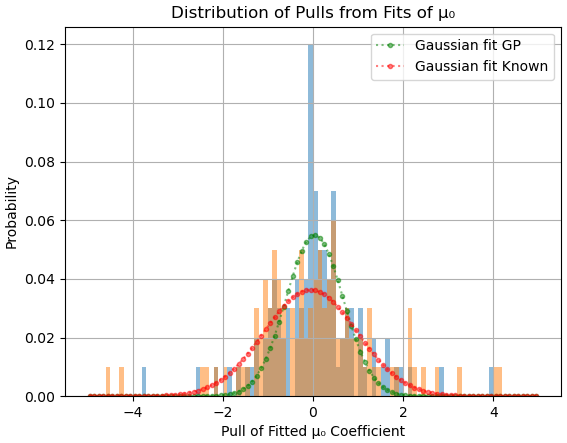
- Assuming *n* known datapoints, select *n* random GP datapoints and compare.

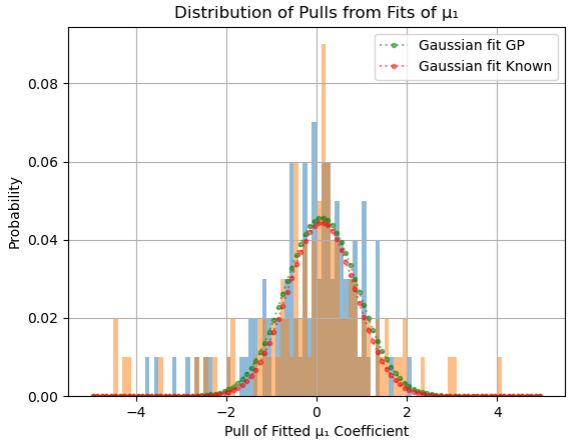
Points within confidence intervals

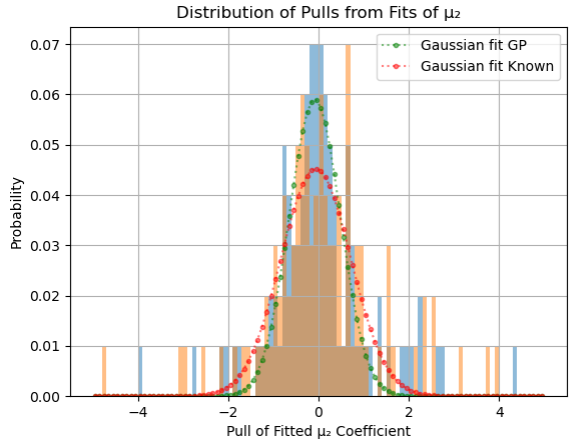
Confidence interval	Expected percentage of points within confidence interval (%)	Mean percentage of GP points within confidence interval (%)
0.67σ	50	62.6
1σ	68.3	77.0
1.96σ	95	96.4

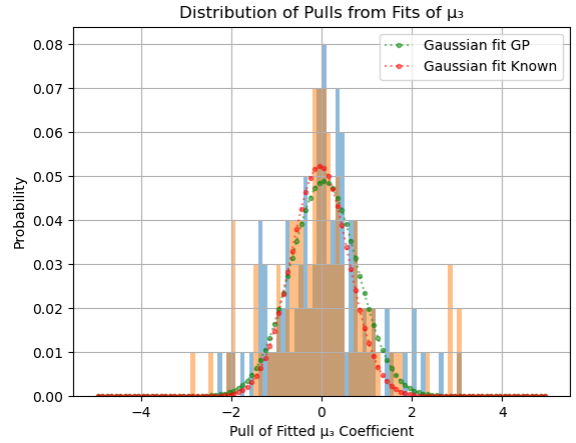
Fitting Parameters

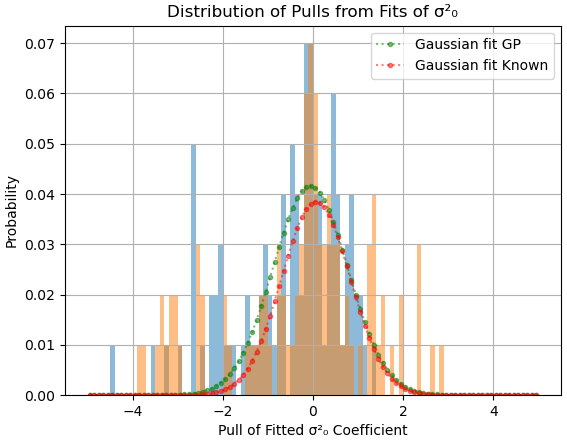
- The functional form of the 2D surface can be fitted using the known datapoints as well as the *n* random datapoints (to check the GP performs well in all kinematic regions).
- The pull of the 12 fitted coefficients compared to the actual coefficients can be calculated for both.

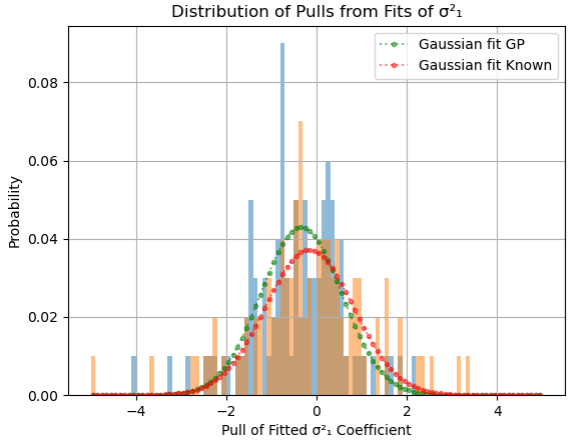

Distribution of Pulls from Fits of μ₀
Gaussian fit GP
Gaussian fit Known
Probability
Pull of Fitted μ₀ Coefficient

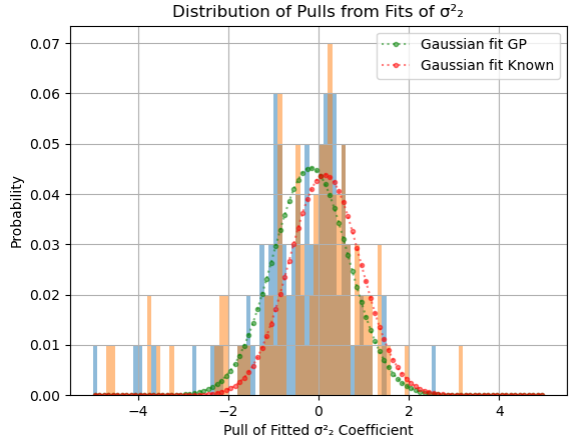

Distribution of Pulls from Fits of μ₁
Gaussian fit GP
Gaussian fit Known
Probability
Pull of Fitted μ₁ Coefficient

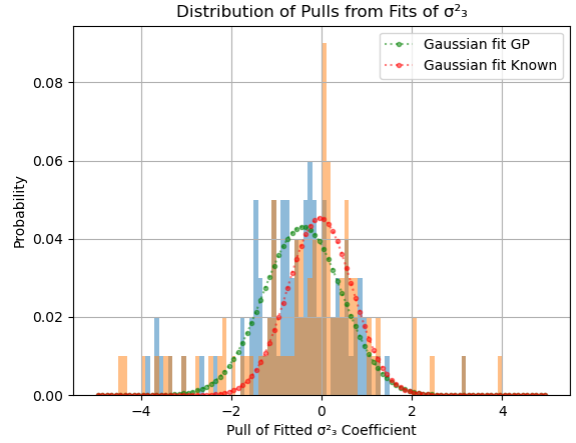

Distribution of Pulls from Fits of μ₂
Gaussian fit GP
Gaussian fit Known
Probability
Pull of Fitted μ₂ Coefficient

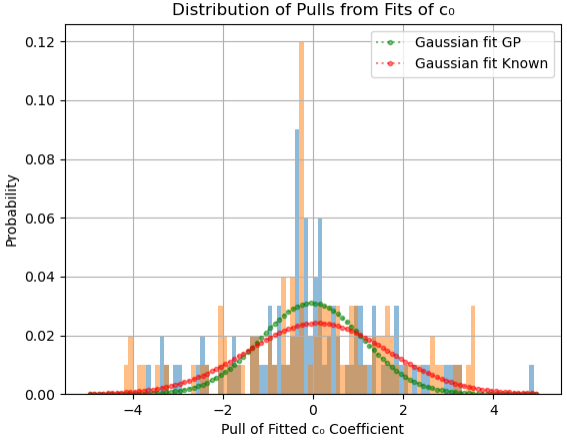

Distribution of Pulls from Fits of μ₃
Gaussian fit GP
Gaussian fit Known
Probability
Pull of Fitted μ₃ Coefficient

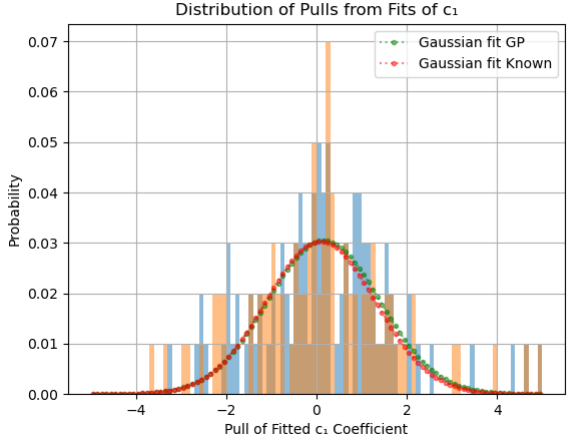

Distribution of Pulls from Fits of σ²₀
Gaussian fit GP
Gaussian fit Known
Probability
Pull of Fitted σ²₀ Coefficient

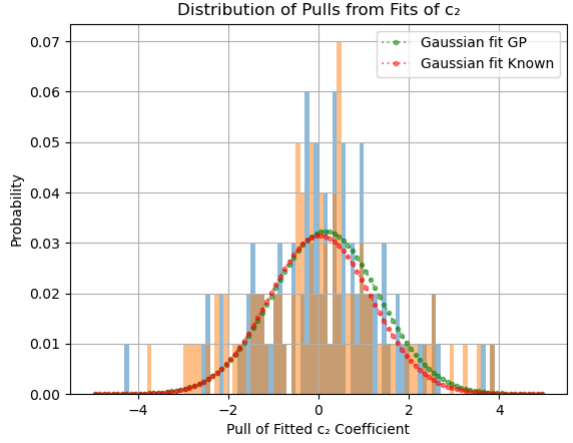

Distribution of Pulls from Fits of σ²₁
Gaussian fit GP
Gaussian fit Known
Probability
Pull of Fitted σ²₁ Coefficient

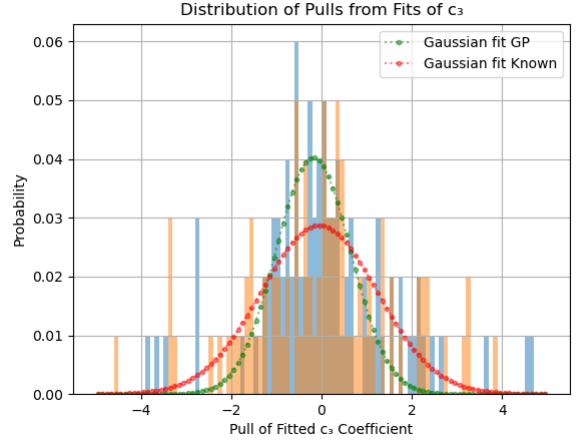

Distribution of Pulls from Fits of σ²₂
Gaussian fit GP
Gaussian fit Known
Probability
Pull of Fitted σ²₂ Coefficient


Distribution of Pulls from Fits of σ²₃
Gaussian fit GP
Gaussian fit Known
Probability
Pull of Fitted σ²₃ Coefficient


Distribution of Pulls from Fits of c₀
Gaussian fit GP
Gaussian fit Known
Probability
Pull of Fitted c₀ Coefficient


Distribution of Pulls from Fits of c₁
Gaussian fit GP
Gaussian fit Known
Probability
Pull of Fitted c₁ Coefficient

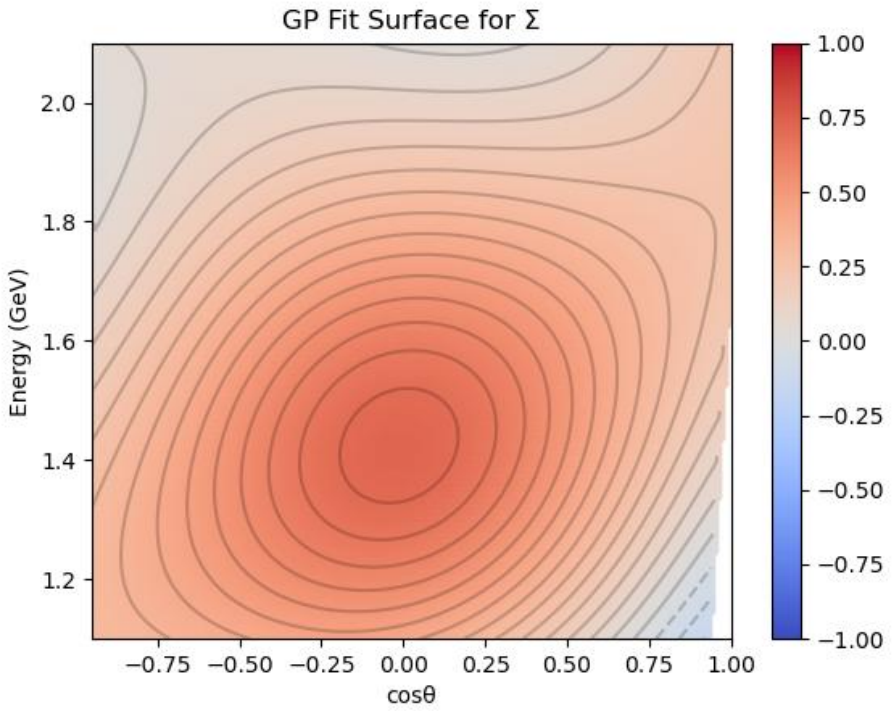

Distribution of Pulls from Fits of c₂
Gaussian fit GP
Gaussian fit Known
Probability
Pull of Fitted c₂ Coefficient


Distribution of Pulls from Fits of c₃
Gaussian fit GP
Gaussian fit Known
Probability
Pull of Fitted c₃ Coefficient

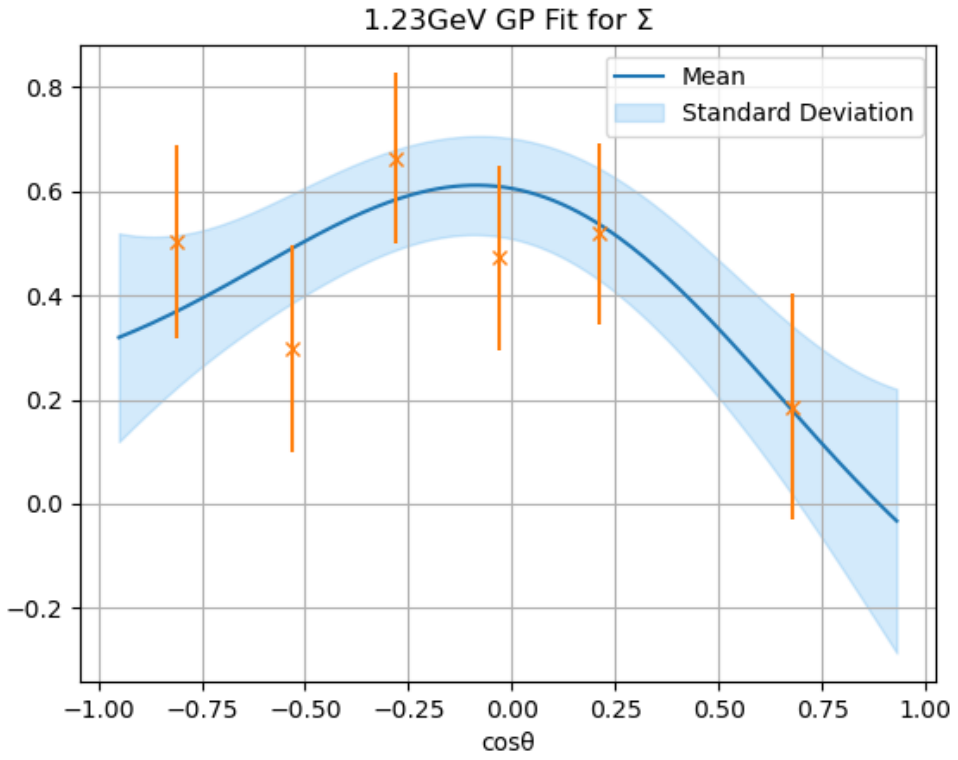
What does real data look like?

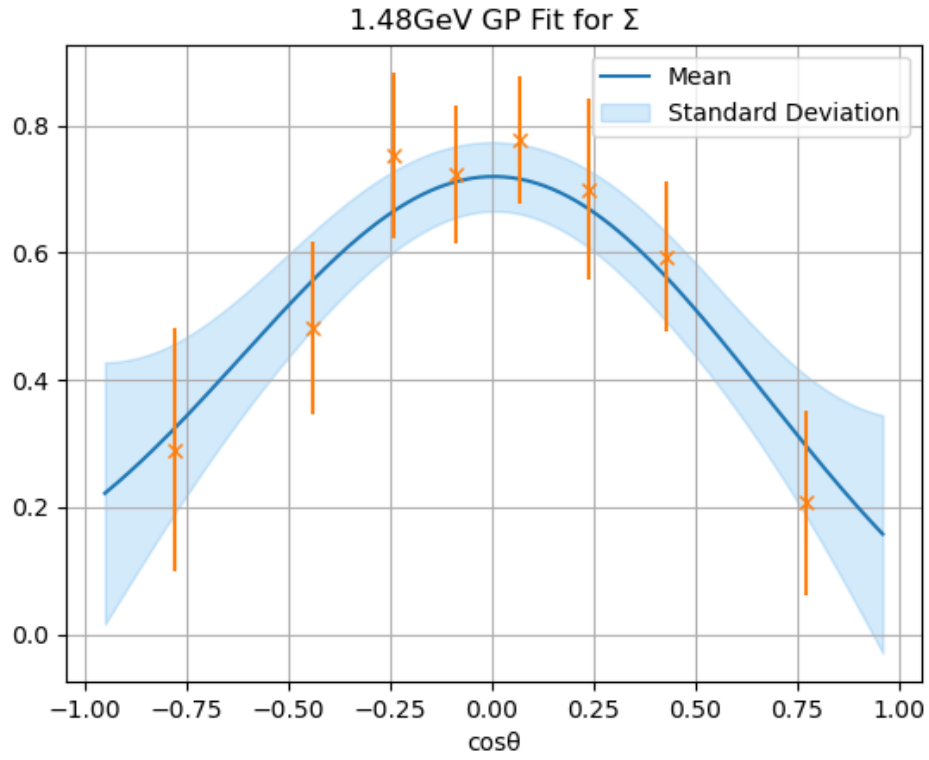
Data from CLAS

The GP has been used on data recently published by the CLAS collaboration at Jefferson Lab, specifically 5 polarisation observables (Σ, P, T, O_x and O_z) of the $K^0\Sigma^+$ reaction.[2] Example plots for Σ:

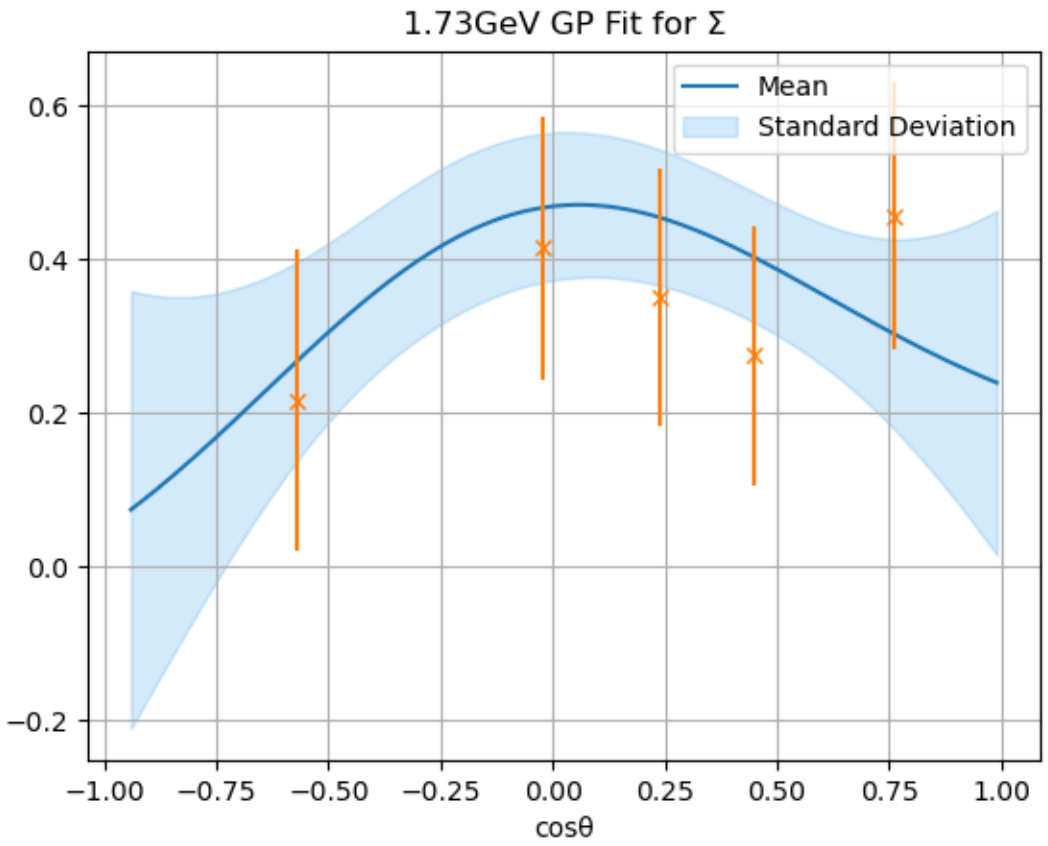


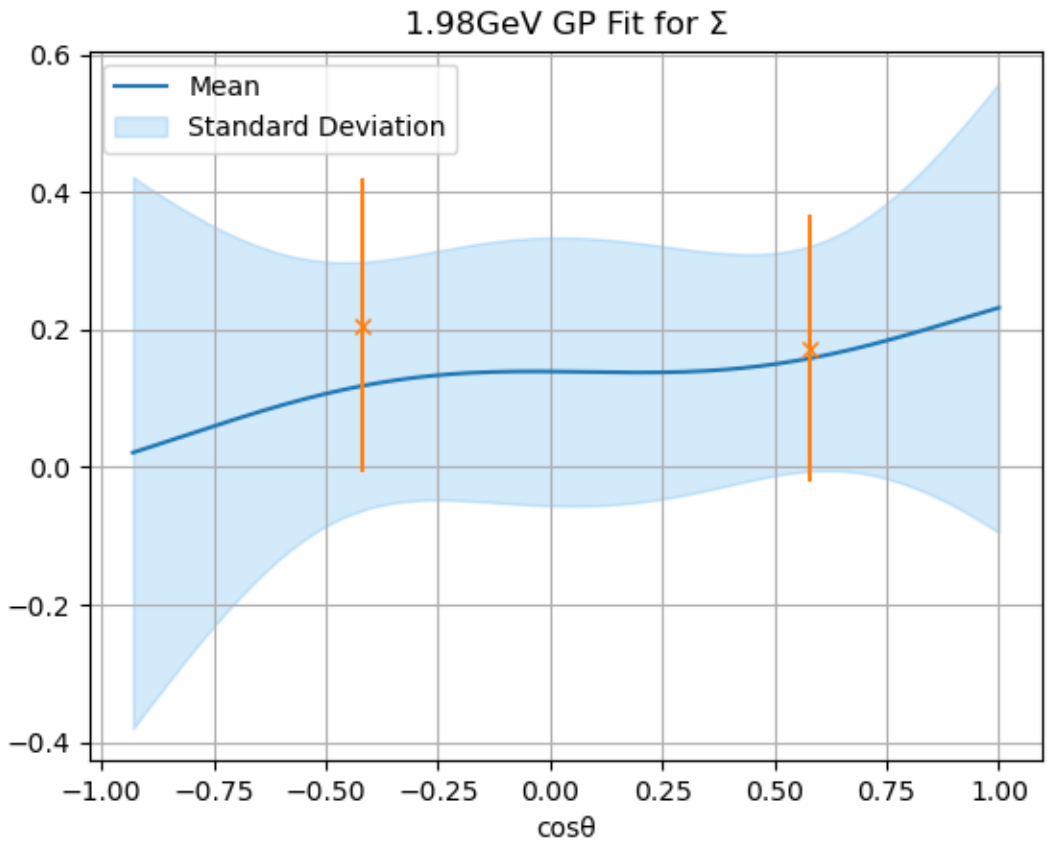
GP 1D Projections for Σ





GP 1D Projections for Σ

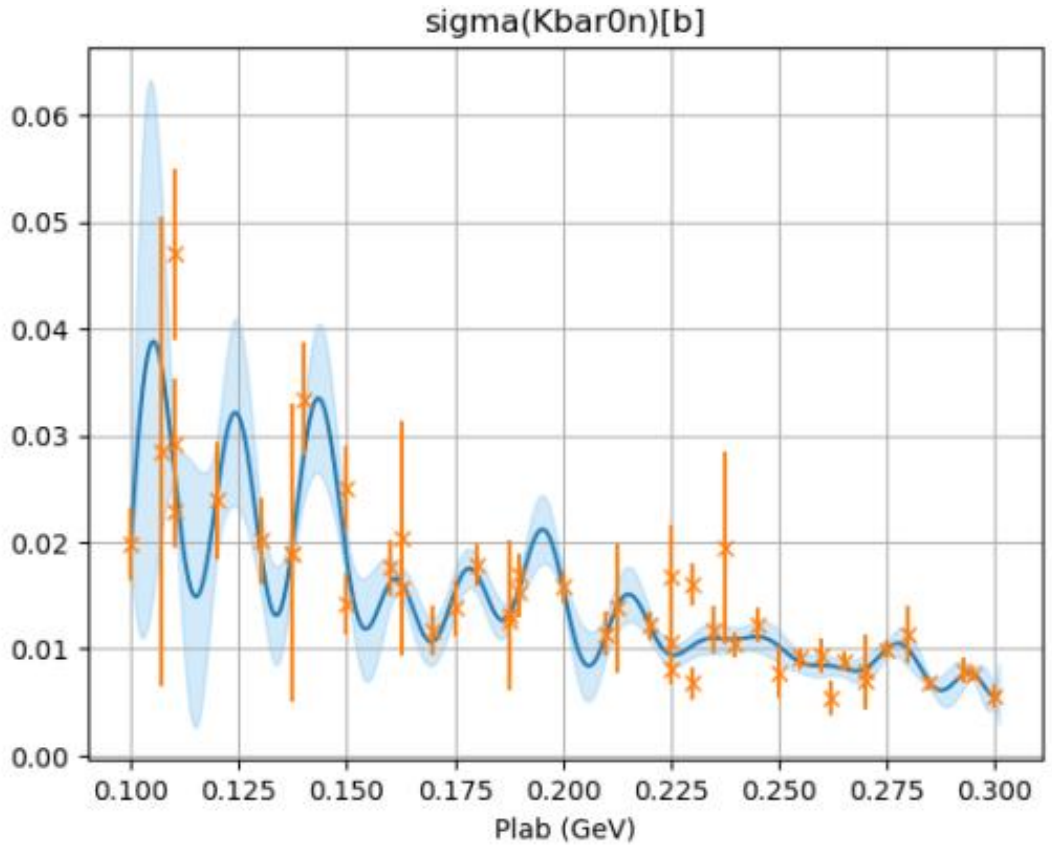




Inconsistencies across Datasets

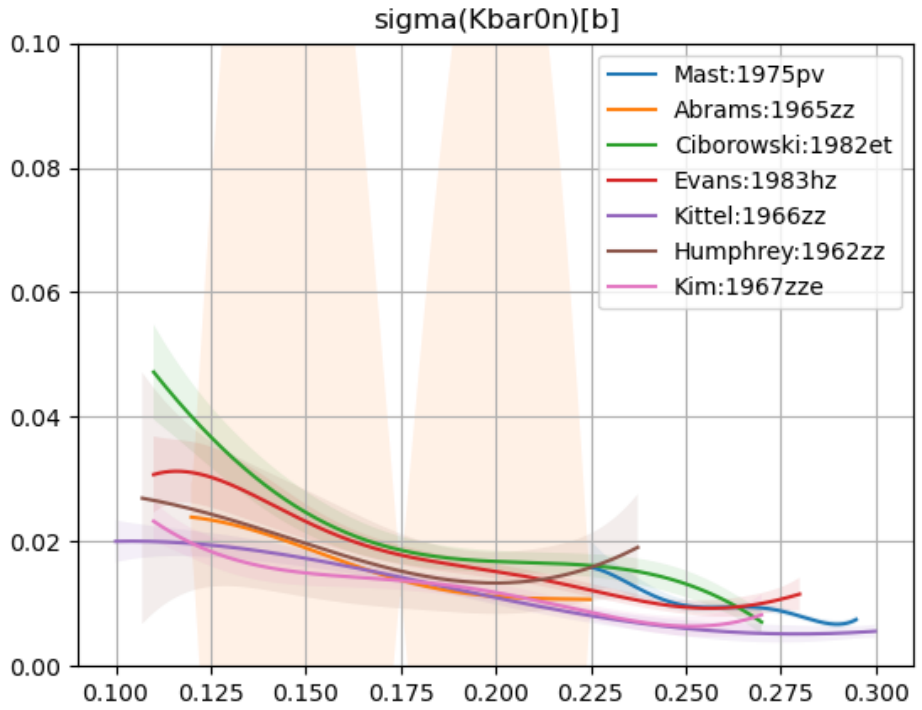
Simple 1D case - KbarN Cross Section

Try to run a GP fit on the world data for $K^- p \to \bar{K}^0 n$



KbarN

Instead running a GP on each individual experiment produces the following:

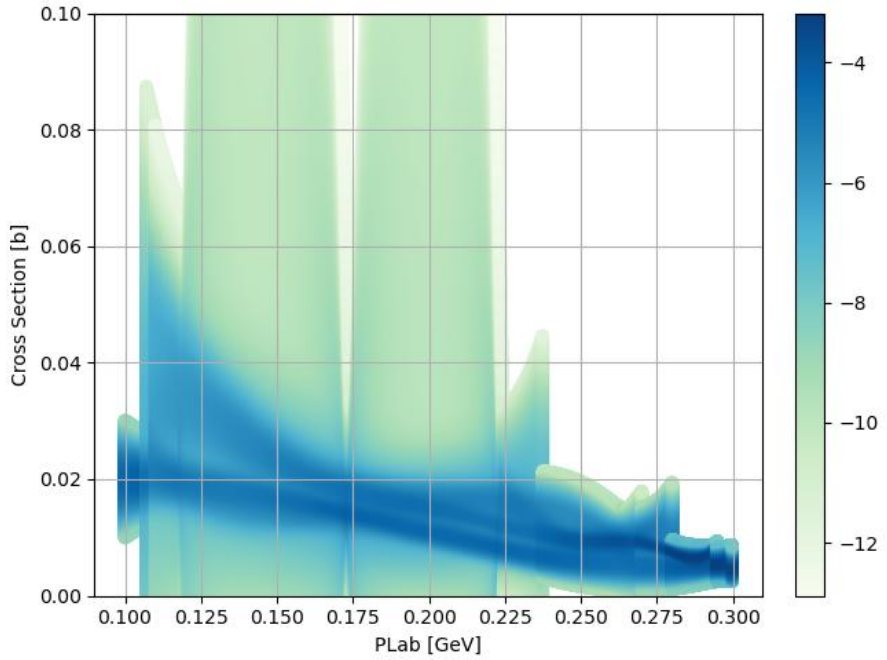


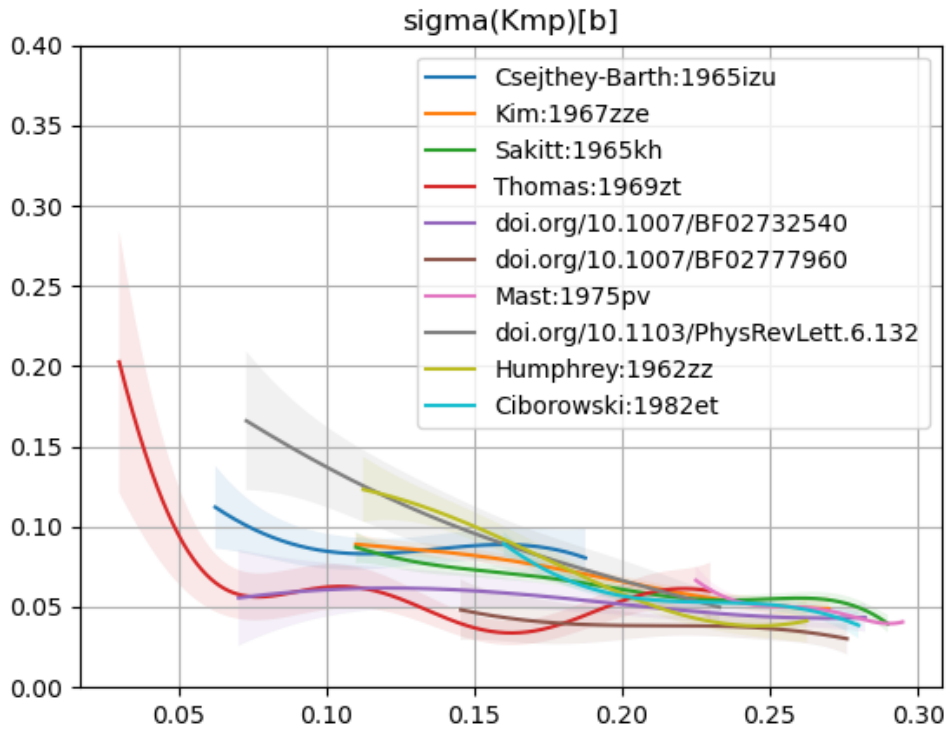
Probability surface

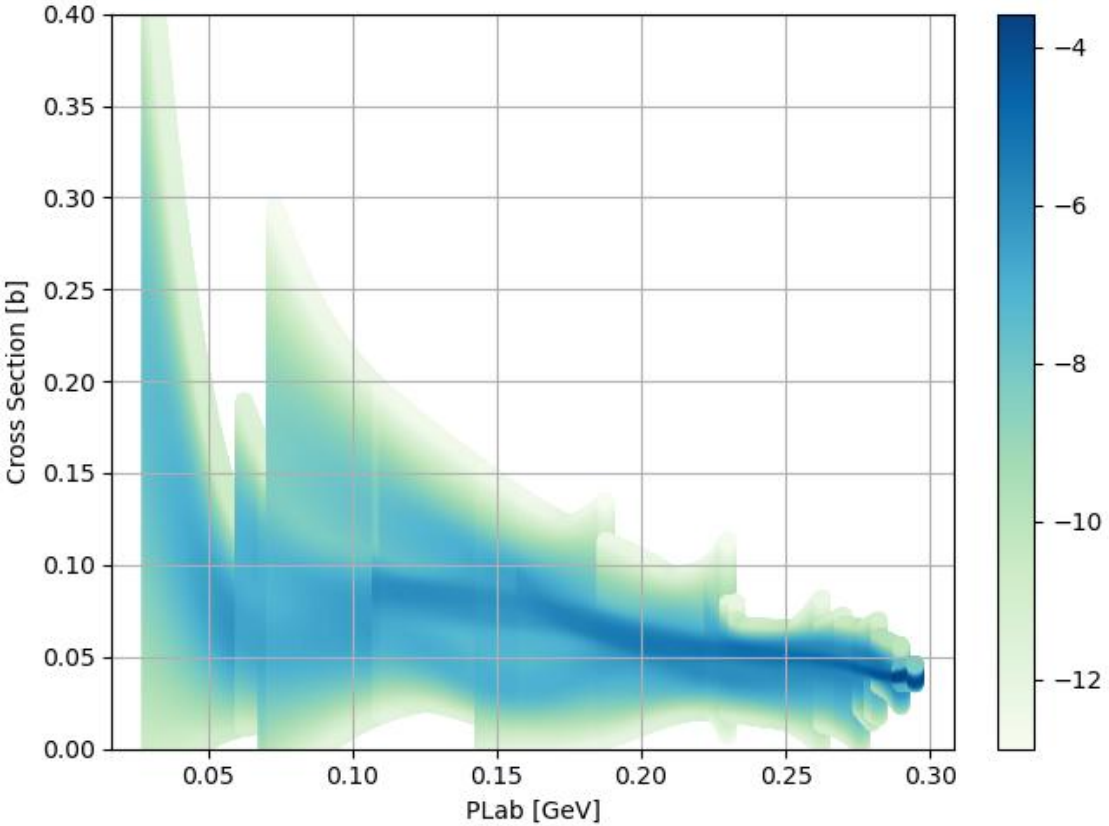
- Taking a 1D projection in energy, each result (from the different GPs) can be assumed to form a normalised gaussian.
- This means that results with larger error bars have a lower amplitude and thus contribute less to the 1D projection.

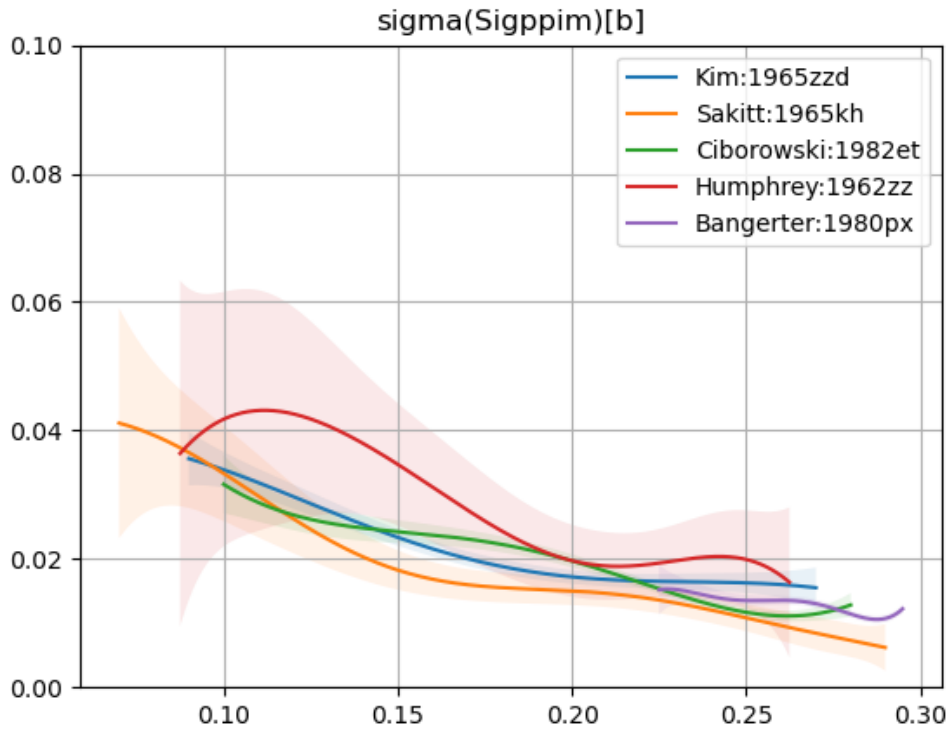
Probability Surface

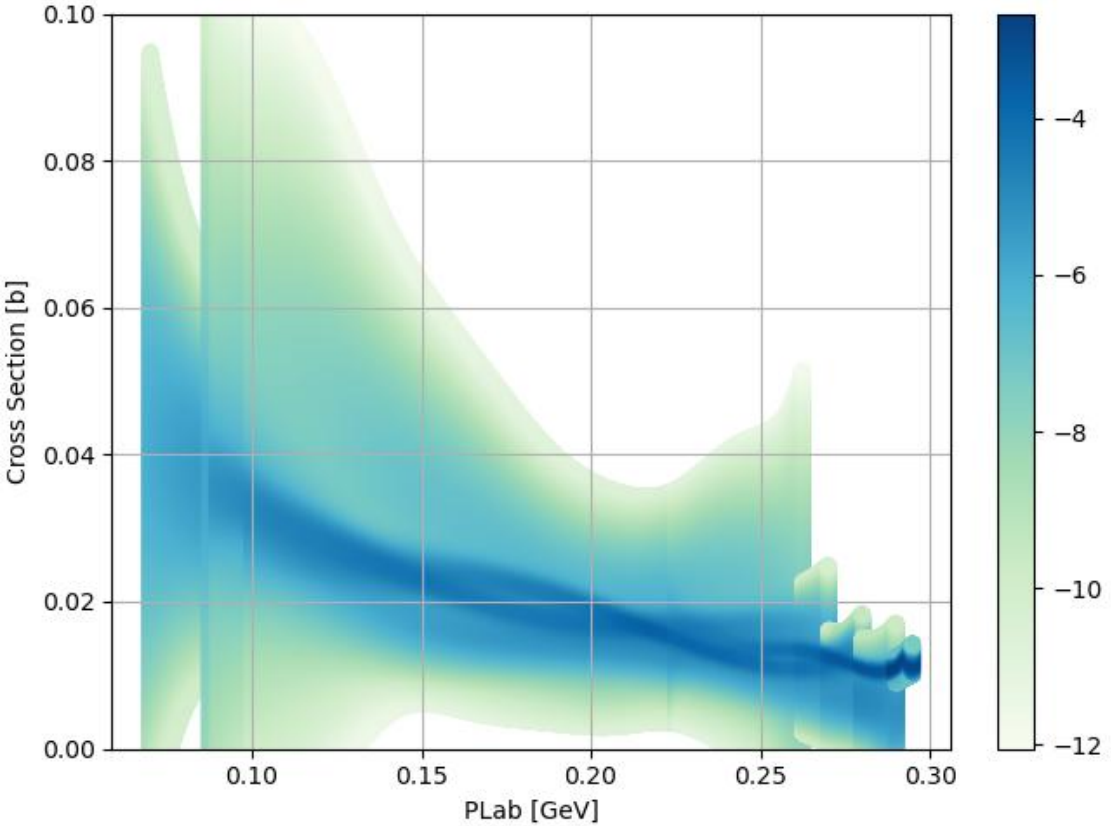
By normalising each slice, the (log) probability of a cross-section value at a given energy can be found:




sigma(Kmp)[b]
Csejthey-Barth:1965izu
Kim:1967zze
Sakitt:1965kh
Thomas:1969zt
doi.org/10.1007/BF02732540
doi.org/10.1007/BF02777960
Mast:1975pv
doi.org/10.1103/PhysRevLett.6.132
Humphrey:1962zz
Ciborowski:1982et
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0.05
0.10
0.15
0.20
0.25
0.30


Cross Section [b]
PLab [GeV]
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0.05
0.10
0.15
0.20
0.25
0.30
−4
−6
−8
−10
−12


sigma(Sigppim)[b]
Kim:1965zzd
Sakitt:1965kh
Ciborowski:1982et
Humphrey:1962zz
Bangerter:1980px
0.10
0.08
0.06
0.04
0.02
0.00
0.10
0.15
0.20
0.25
0.30


Cross Section [b]
PLab [GeV]
0.10
0.08
0.06
0.04
0.02
0.00
0.10
0.15
0.20
0.25
0.30
−4
−6
−8
−10
−12

Conclusion

- A Gaussian Process is an extremely useful machine learning tool to expand existing, limited datasets.
- The methodology presented here has been demonstrated to work on pseudodata, modelled on real-life sparse datasets.
- Inconsistencies between experiments can be measured and a probability surface found.

- Code to run GP fits and probability surface calculations can be found on the GitHub here, https://github.com/rferguson22/Gaussian-Process

Thanks for listening

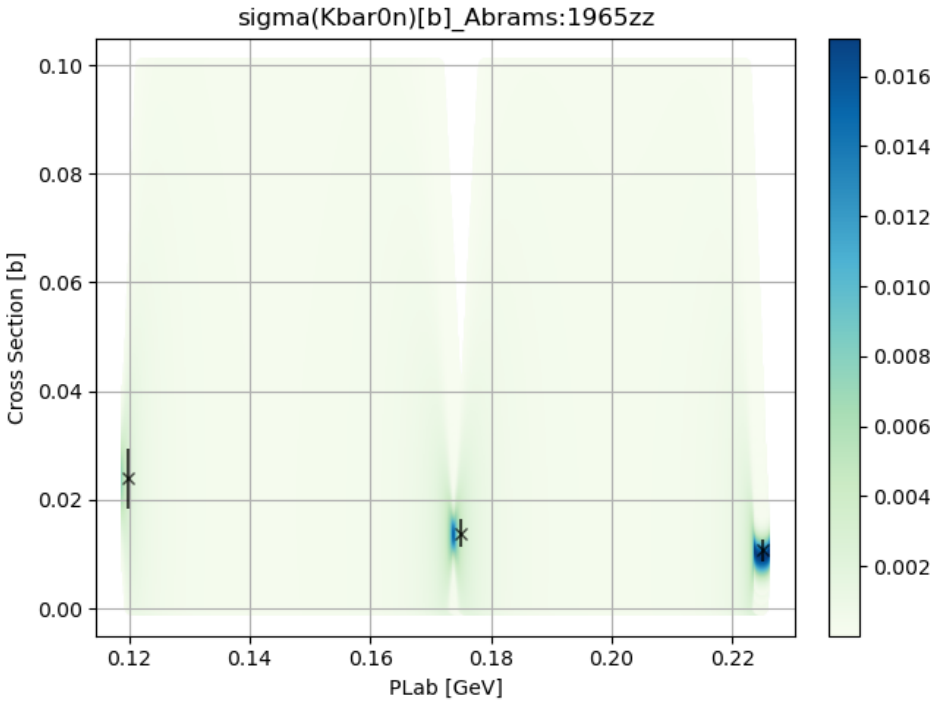
References

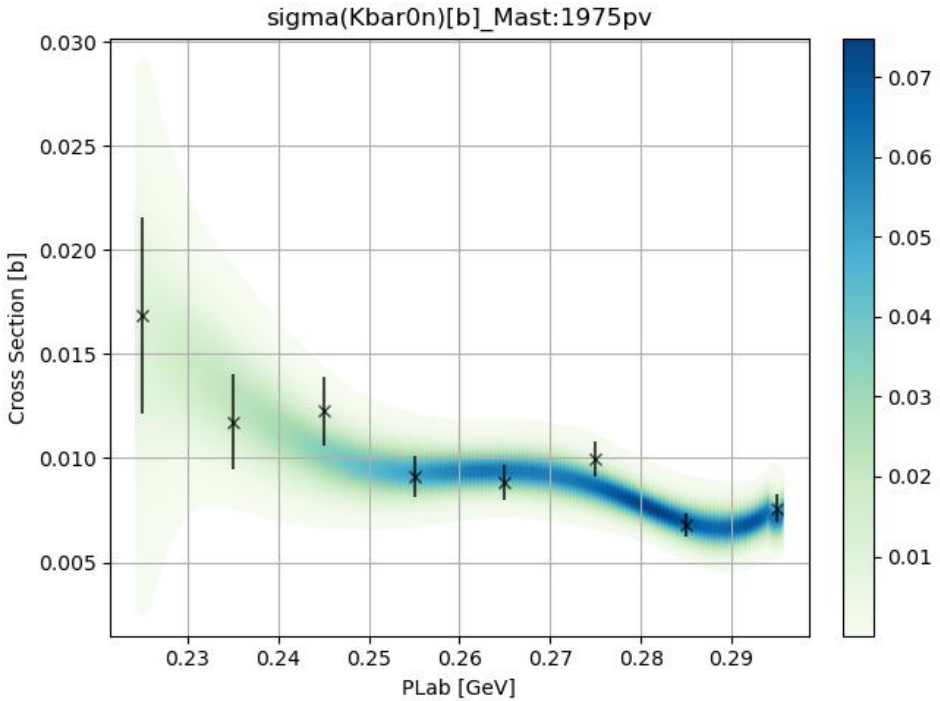
1. R. Laurini, *Geographic Knowledge Infrastructure*, 2017, www.sciencedirect.com/topics/earth-and-planetary-sciences/convex-hull [accessed May 28th 2024]
2. L. Clark et al, *Photoproduction of the Σ+ hyperon using linearly polarized photons with CLAS,* 2024, https://arxiv.org/abs/2404.19404 [accessed May 30th 2024]
3. F. Rieger, *HIGH ENERGY ASTROPHYSICS - Lecture 9,* 2024, www.mpi-hd.mpg.de/personalhomes/frieger/HEA9.pdf [accessed May 30th 2024]
4. R. Fiore et al, *Kinematically complete analysis of the CLAS data on the proton structure Function F2 in a Regge-Dual model,* 2006, https://www.researchgate.net/publication/46776238_Kinematically_complete_analysis_of_the_CLAS_data_on_the_proton_structure_Function_F2_in_a_Regge-Dual_model?_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6Il9kaXJlY3QiLCJwYWdlIjoiX2RpcmVjdCJ9fQ [accessed May 30th 2024]
5. M. Krasser, *Gaussian processes*. 2018. URL: krasserm.github.io/2018/03/19/gaussian-processes/ [accessed May 26th 2024]
6. C. E. Rasmussen and C.K. I. Williams, *Gaussian Processes for Machine Learning*, MIT Press, 2006, ISBN-13 978-0-262-18253-9.

Back-up Slides

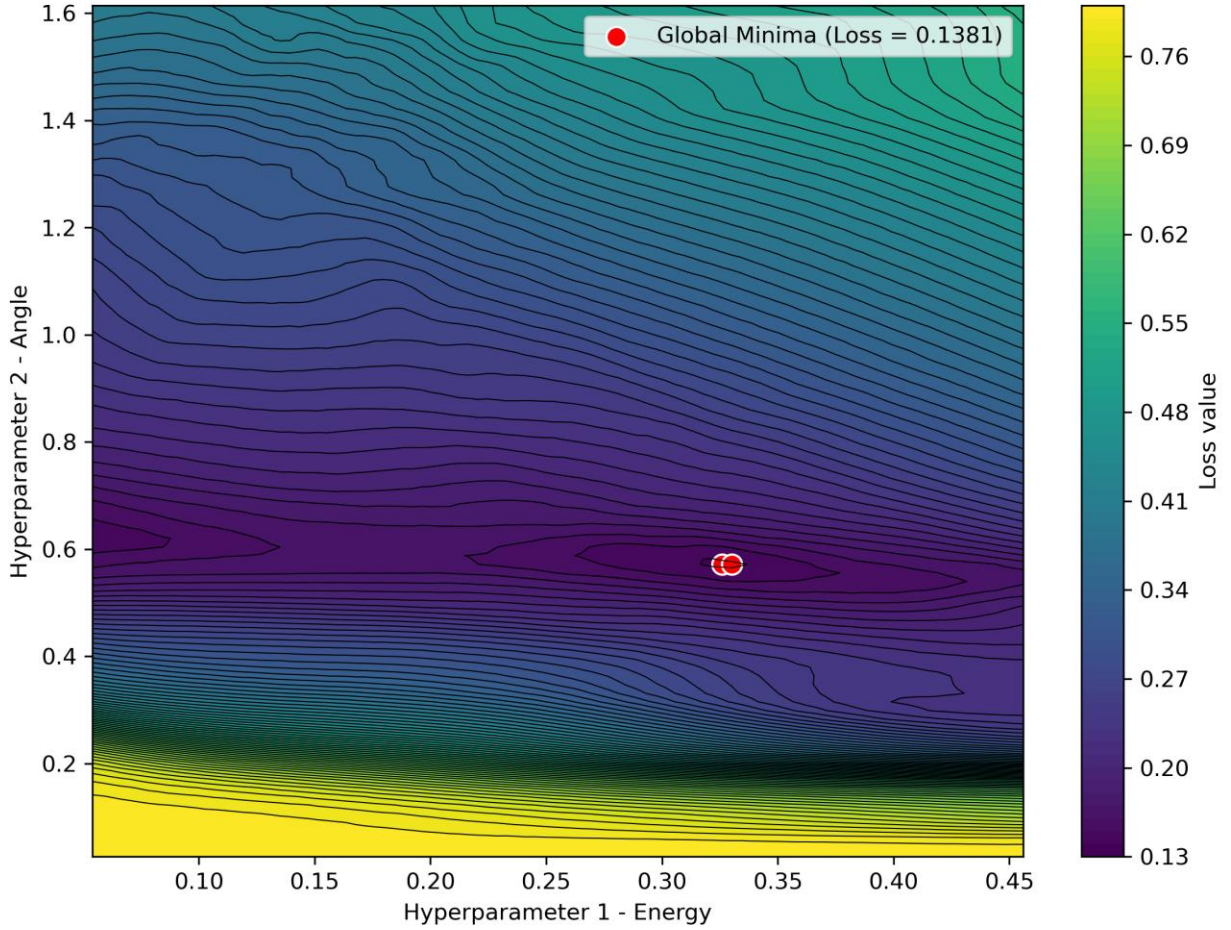
Coefficient	Mean of pull distribution from known datapoints fit	Variance of pull distribution from known datapoints fit	Mean of pull distribution from GP datapoints fit	Variance of pull distribution from GP datapoints fit
c_0	0.04	0.91	0.06	0.92
μ_0	-0.04	0.82	-0.05	0.84
${\sigma_0}^2$	0.0	0.77	-0.01	0.79
c_1	0.04	0.89	0.04	0.91
μ_1	-0.03	0.74	-0.02	0.73
${\sigma_1}^2$	-0.1	0.77	-0.09	0.78
c_2	-0.06	1.01	-0.06	1.05
μ_2	-0.05	0.73	-0.05	0.75
${\sigma_2}^2$	-0.17	0.82	-0.17	0.83
c_3	-0.06	0.95	-0.07	0.96
μ_3	-0.02	0.73	-0.04	0.74
${\sigma_3}^2$	-0.07	0.73	-0.07	0.76

Kbar0n

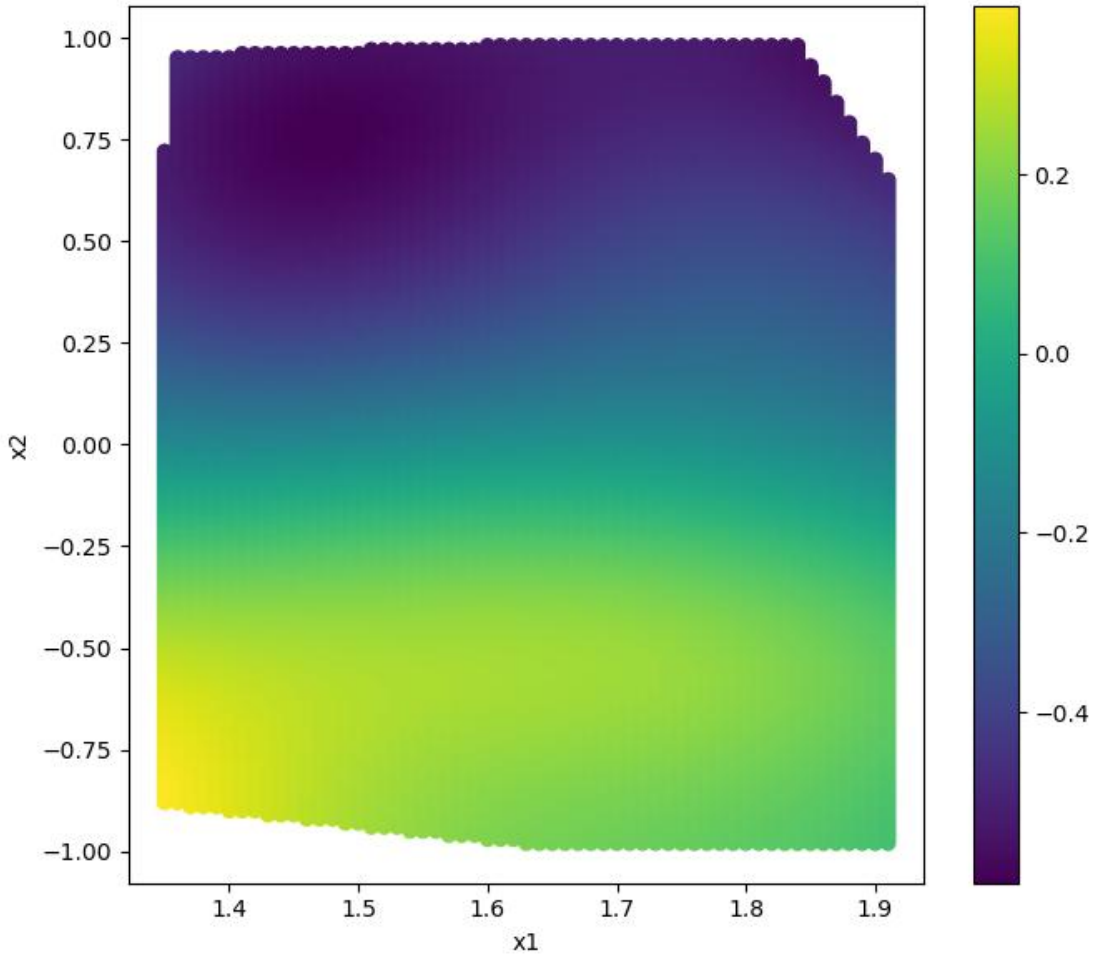

sigma(Kbar0n)[b]_Abrams:1965zz
Cross Section [b]
PLab [GeV]
0.10
0.08
0.06
0.04
0.02
0.00
0.12
0.14
0.16
0.18
0.20
0.22
0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002

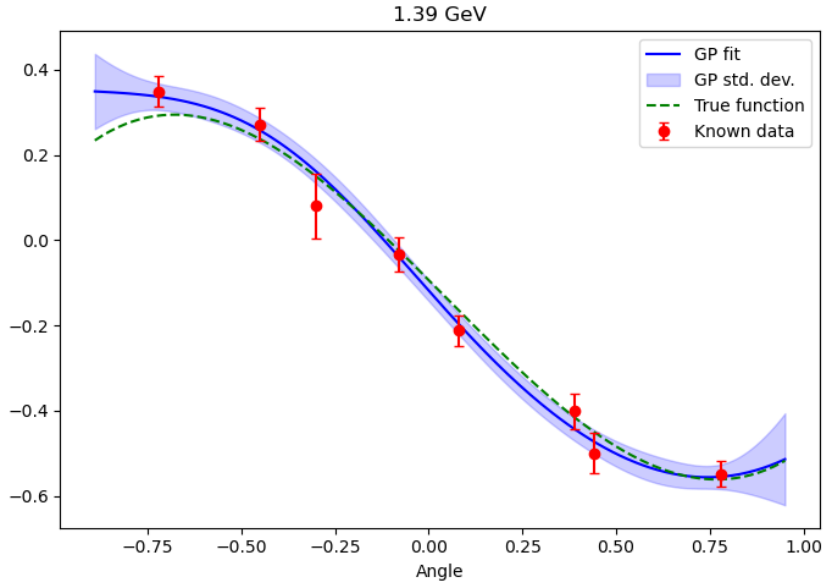

sigma(Kbar0n)[b]_Mast:1975pv
Cross Section [b]
PLab [GeV]
0.030
0.025
0.020
0.015
0.010
0.005
0.23
0.24
0.25
0.26
0.27
0.28
0.29
0.07
0.06
0.05
0.04
0.03
0.02
0.01

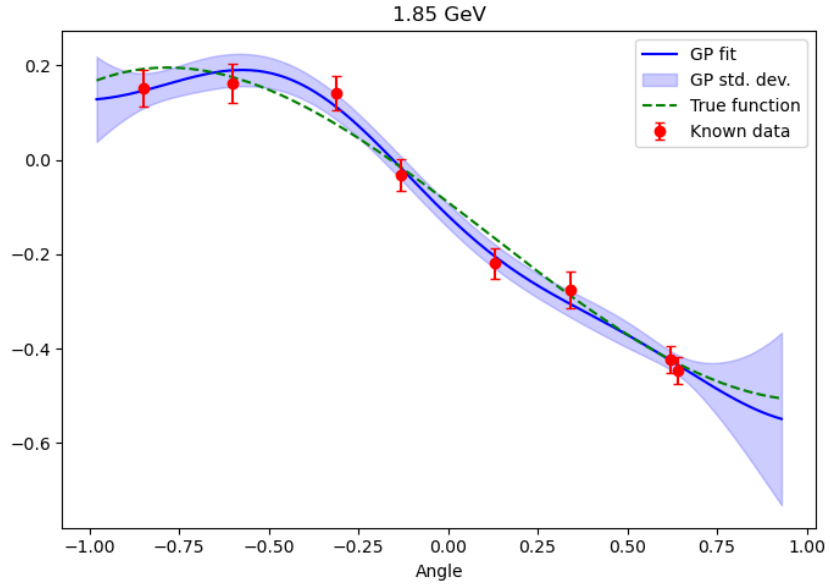
New minima far from boundary

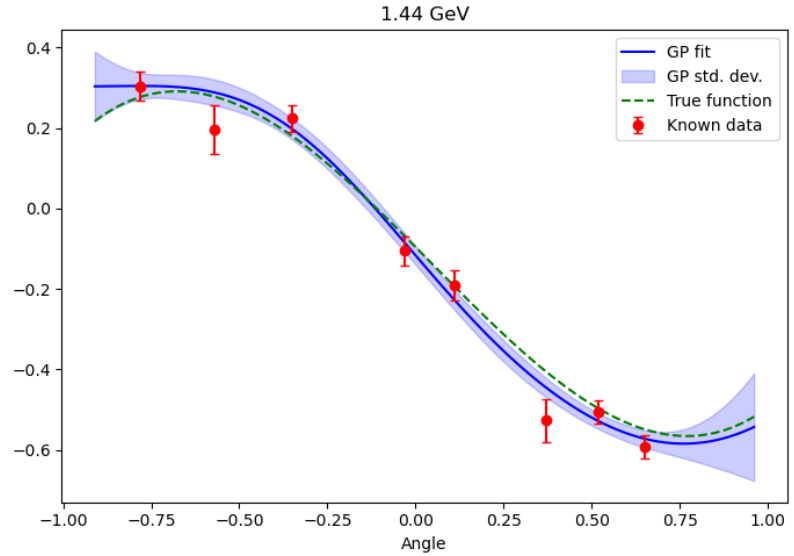


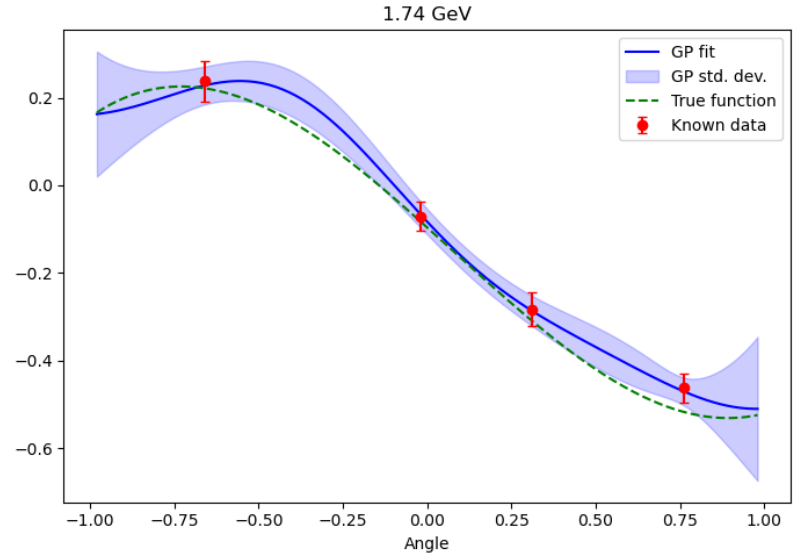
Gives reasonable looking surface



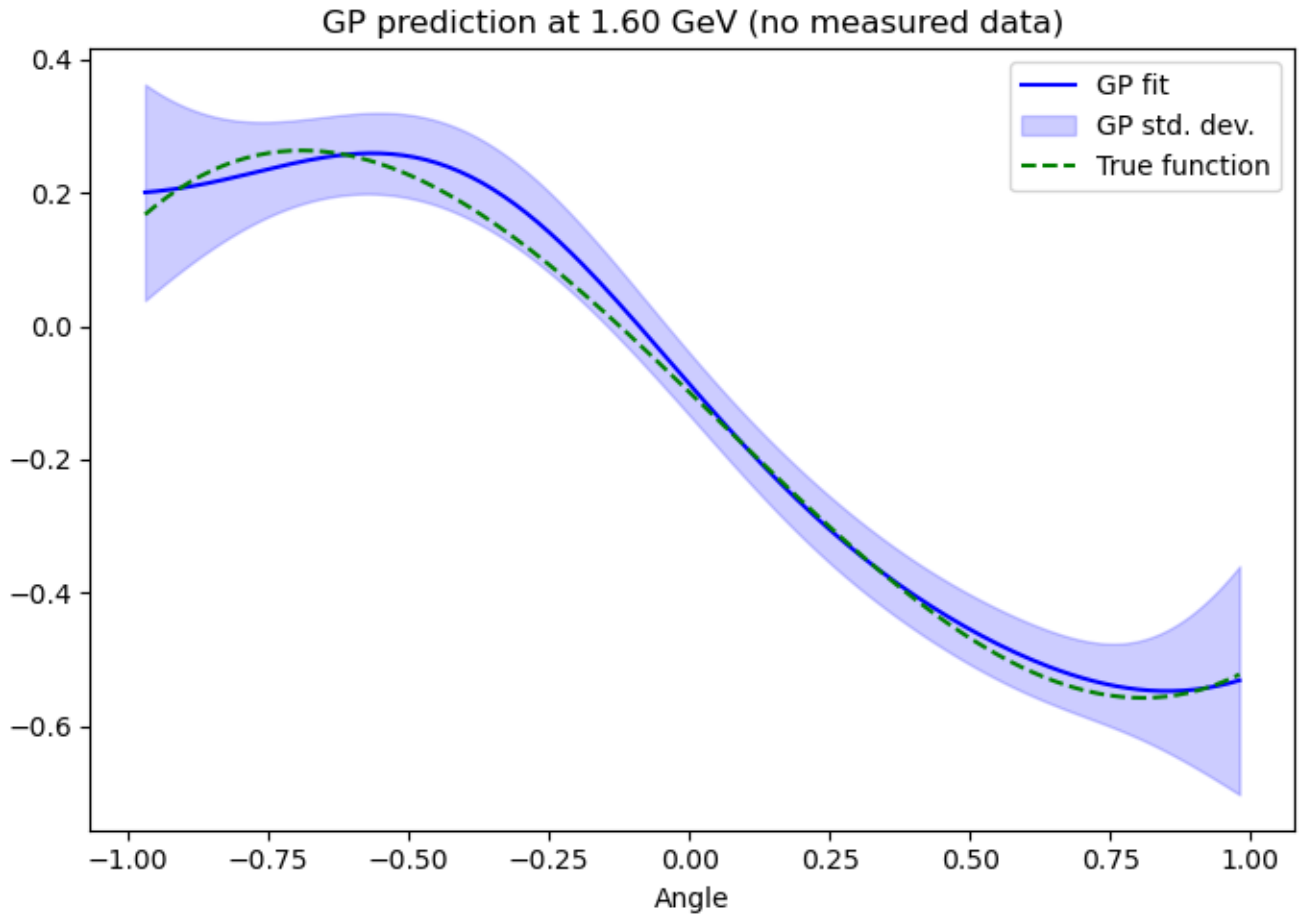

1.39 GeV
GP fit
GP std. dev.
True function
Known data
Angle


1.85 GeV
GP fit
GP std. dev.
True function
Known data
Angle

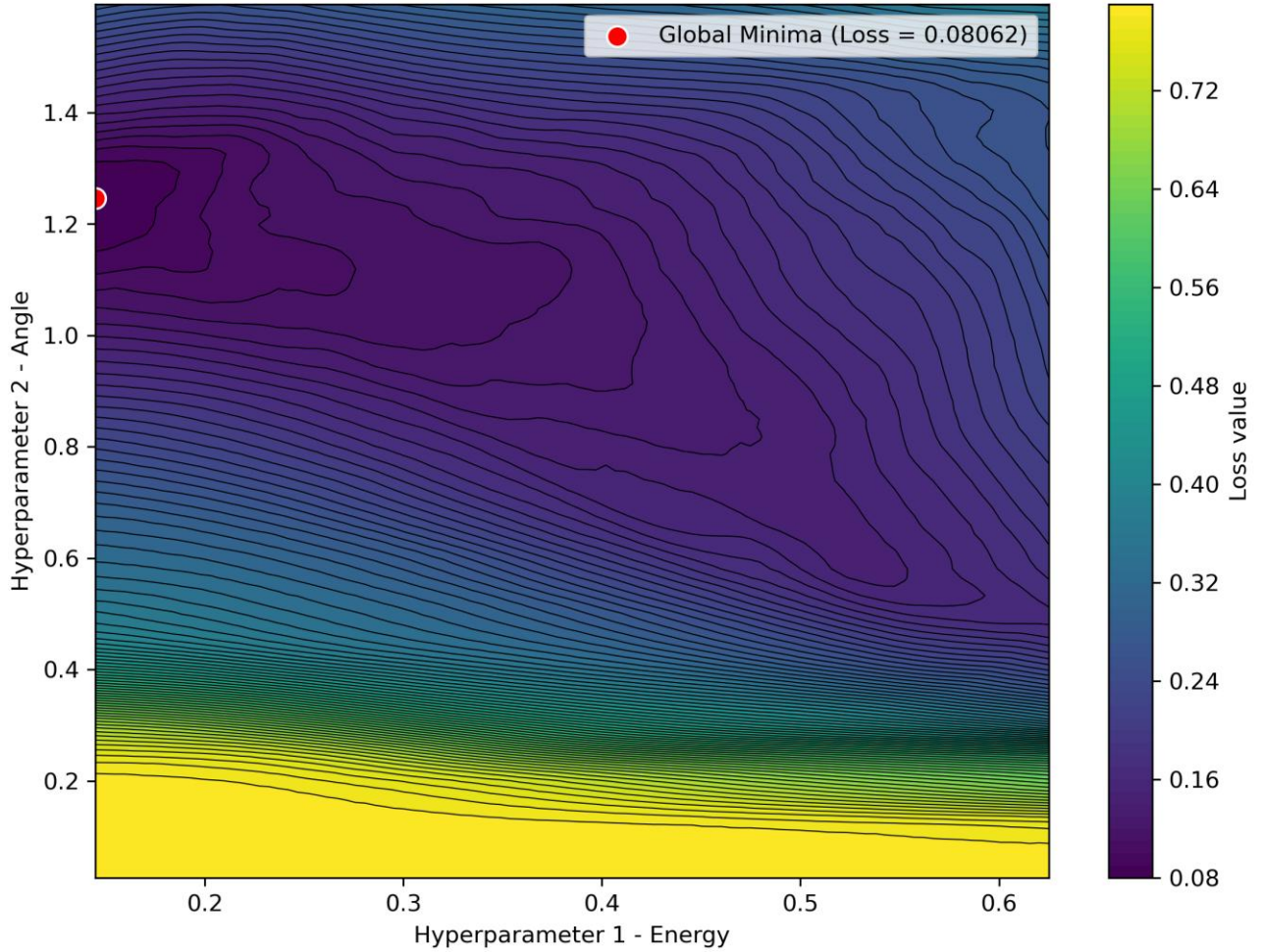

1.44 GeV
GP fit
GP std. dev.
True function
Known data
Angle

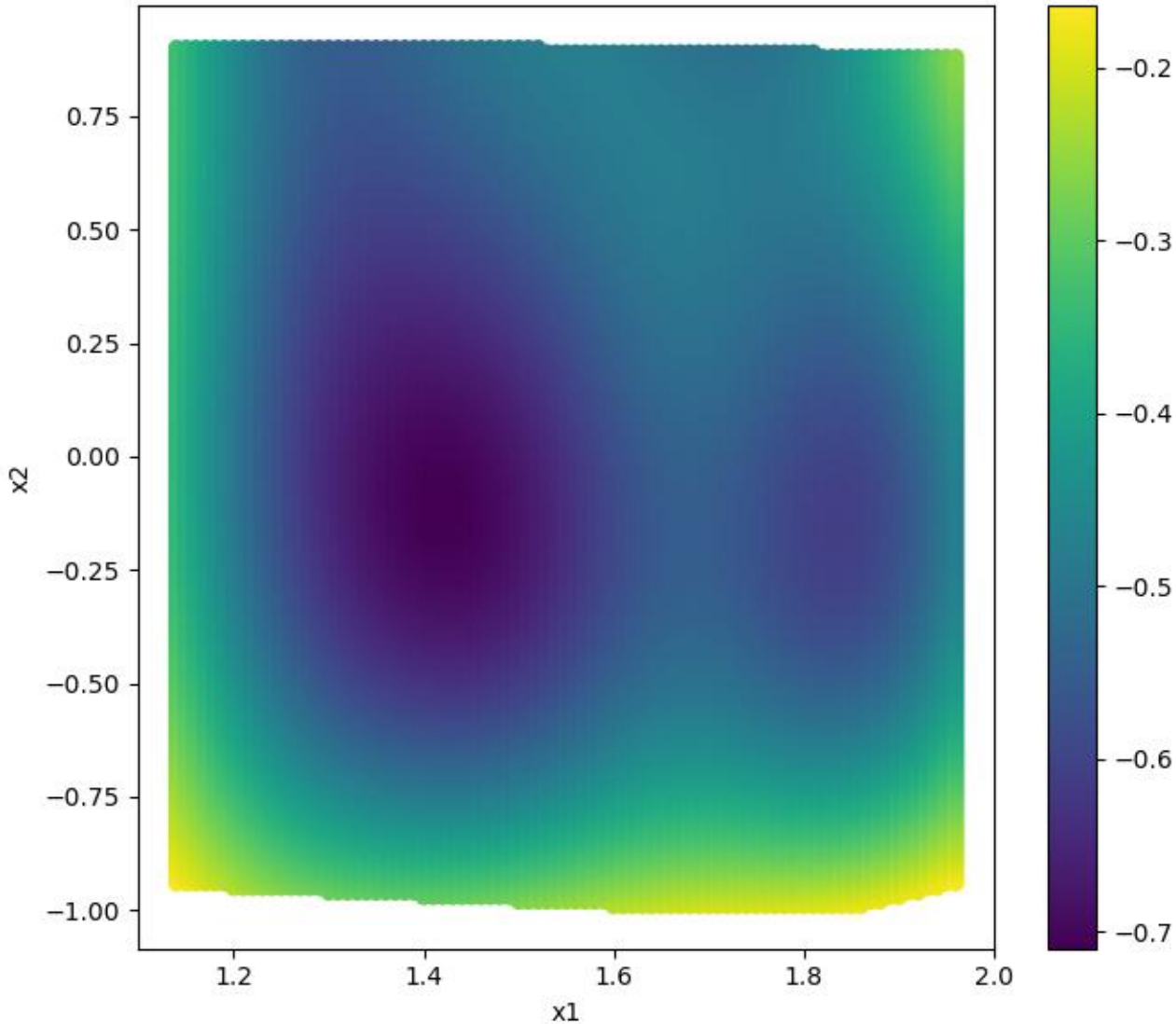

1.74 GeV
GP fit
GP std. dev.
True function
Known data
Angle

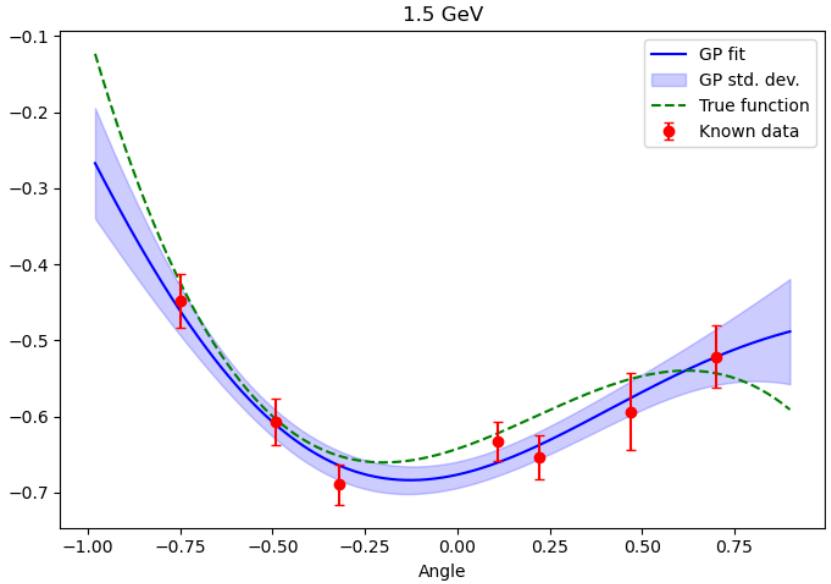
Check at E=1.6 (no measured data)

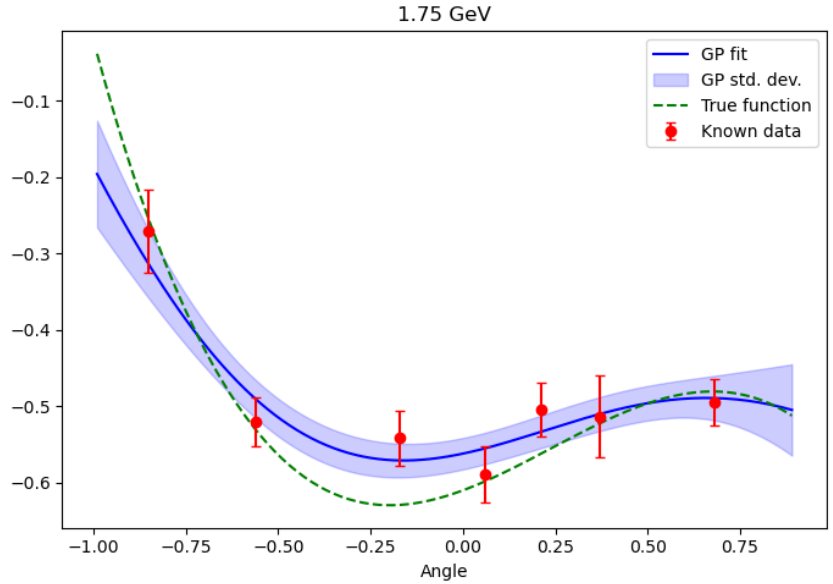


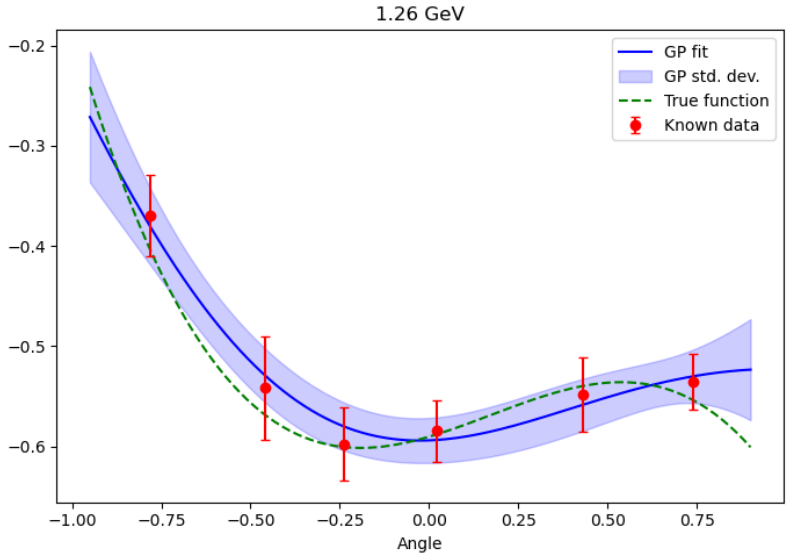
But what about other points still on the boundary?

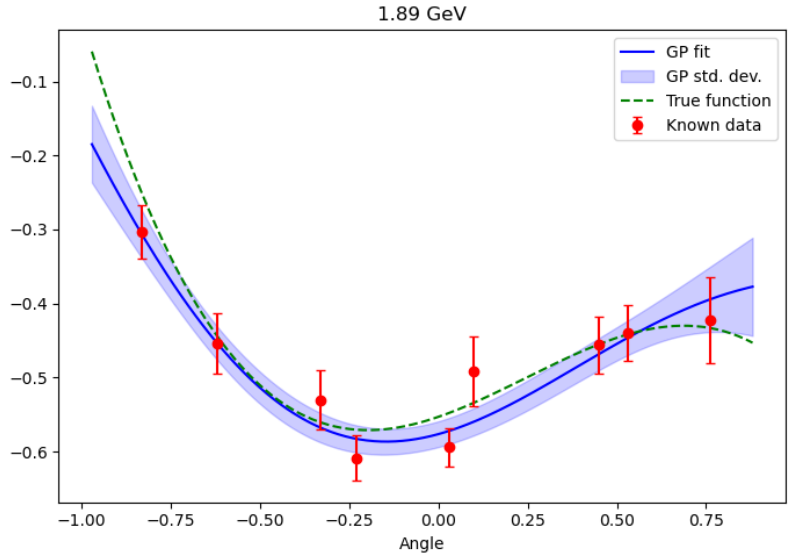



0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
x2
1.2
1.4
1.6
1.8
2.0
x1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7

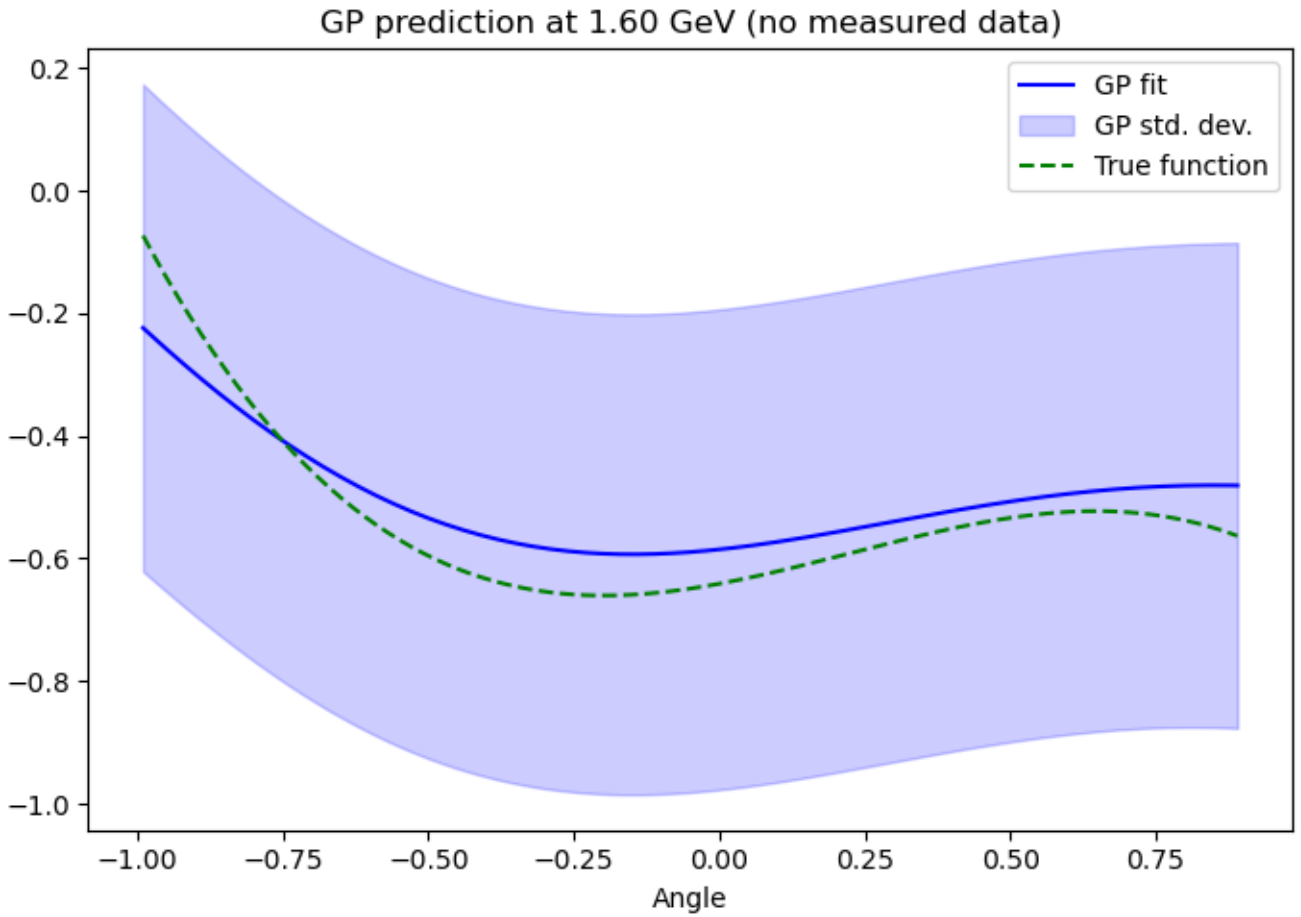

1.5 GeV
GP fit
GP std. dev.
True function
Known data
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
Angle


1.75 GeV
GP fit
GP std. dev.
True function
Known data
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
Angle


1.26 GeV
GP fit
GP std. dev.
True function
Known data
−0.2
−0.3
−0.4
−0.5
−0.6
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
Angle


1.89 GeV
GP fit
GP std. dev.
True function
Known data
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
Angle

Check at E=1.6 (no measured data)



Generating Pseudodata I

A generated asymmetry datapoint is based on the effective number of counts measured. This can be expressed as

$$A = \frac{N_+ - N_-}{N_+ + N_-}$$

where N_+, N_- are used to describe the 2 different states which are used to estimate the effective count. These take into account beam polarisation, recoils, target dilution and other such factors. These random variables are generated from "true" values:

$$N \sim \text{Pois}(n_\pm)$$

where $n_\pm = \frac{1}{2} n_e [1 \pm f(w, \cos\theta)]$. Here n_e is defined as the effective number of events and is in the range [200,1000] which is estimated based on real data.

Generating Pseudodata II

By using standard propagation of errors, the error on A is given by:

$$\delta A = \frac{2}{(N_+ + N_-)^2}\sqrt{N_+ N_- (N_+ + N_-)}$$

How does it work?

Mathematical Process

Assume that we have *n* known datapoints of the form $(\vec{x}_i, y_i)$ with known errors e_i used to define the expression form $\vec{y} = f(X)$. Here $\vec{x}_i$ is a vector of the kinematic variables (e.g. energy, scattering angle, etc.) and X is a matrix whose rows are $\vec{x}_a, \vec{x}_b$.

e.g. if we have 3 points a,b,c that have some energy E and scattering angle cosθ, then *X* is

$$X = \begin{bmatrix} \vec{x}_a \\ \vec{x}_b \\ \vec{x}_c \end{bmatrix} = \begin{bmatrix} E_a & \cos\theta_a \\ E_b & \cos\theta_b \\ E_c & \cos\theta_c \end{bmatrix}$$

i.e. *X* will always be a 2D matrix, regardless of the number of input dimensions

Mathematical Process

Assume that $\vec{y}$ is drawn from a Multivariate Gaussian of the form $p(\vec{y}|X) \sim \mathcal{N}(\vec{0}, K)$ (the zero mean assumption simplifies some maths later but doesn't impact the prediction), where $K = \kappa(X, X, \vec{l}) + \vec{e}^2 I_n$ is the *n* x *n* covariance matrix.

Here κ is some kernel function that is used to measure the covariance and $\vec{l}$ are hyperparameters (more on this later). Here $K_{ab} = \kappa(\vec{x}_a, \vec{x}_b) + \delta_{ab} {e_a}^2$, where $\vec{x}_a, \vec{x}_b$ are rows of the matrix X.

Mathematical Process

Assume that there are *m* known datapoints of the form outlined previously, with known $\overrightarrow{x_{*_i}}$ with unknown scalars y_{*_i}, which are correlated to the *n* known datapoints.

A matrix X_* can then be generated whose rows are the vectors $\overrightarrow{x_*}$.

As $\overrightarrow{y_*}$ is correlated to $\vec{y}$, they are drawn from the same multivariate Gaussian:

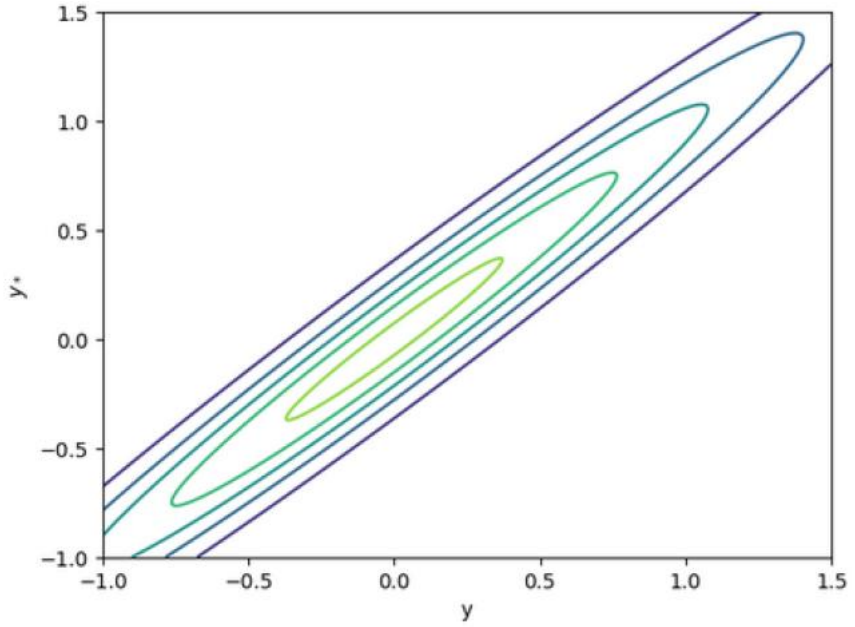
$$\begin{bmatrix} \vec{y} \\ \overrightarrow{y_*} \end{bmatrix} \sim \mathcal{N}\left(\underline{0}, \begin{bmatrix} K & K_* \\ {K_*}^T & K_{**} \end{bmatrix}\right)$$

where $K_* = \kappa(X, X_*), K_{**} = \kappa(X_*, X_*)$.

Essentially, our data can be thought of as a single sample drawn from this multivariate Gaussian.

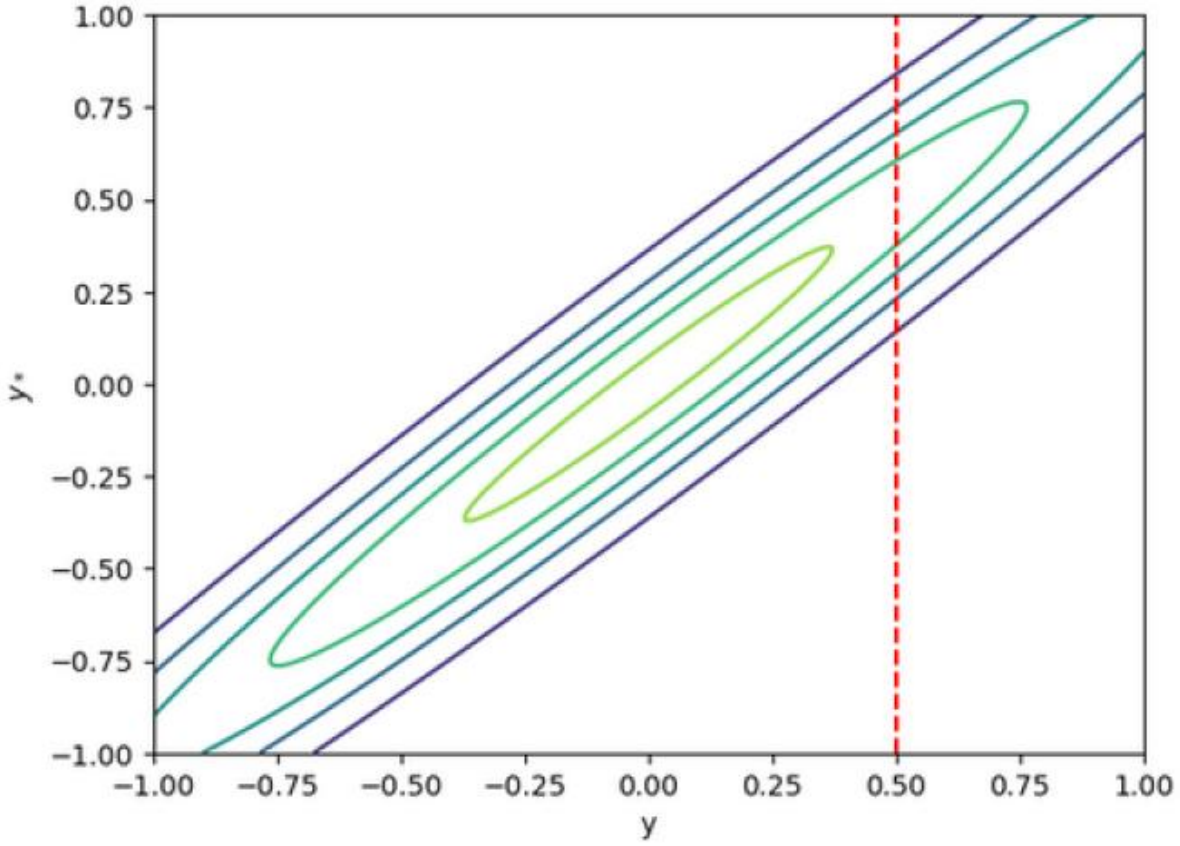
Example Case

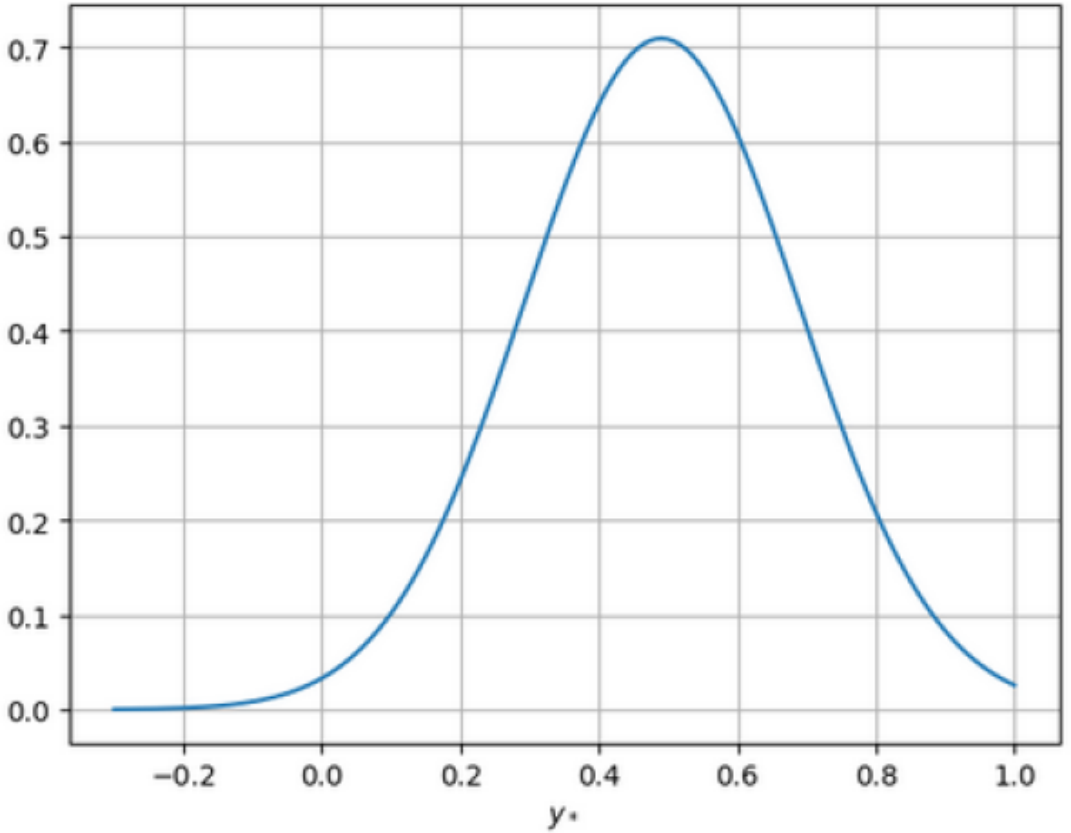
- Assume we have a single point (1,0.5) and want to get a prediction for $x_* = 0.9$.
- Plugging the values of $x = 1$, $x_* = 0.9$ into K,K_s, K_{ss} and plotting the multivariate Gaussian gives the graph below
- Note this plot is generated with the specific values $x = 1$, $x_* = 0.9$, $l = 1$, changing any of these values will result in a different multivariate Gaussian



Example Case

We know that $y = 0.5$ so can do a 1D projection to get the value for y_*





Mathematical Process III

By using the conditional of a multivariate Gaussian, a prediction for $\overrightarrow{y_*}$ can be obtained:

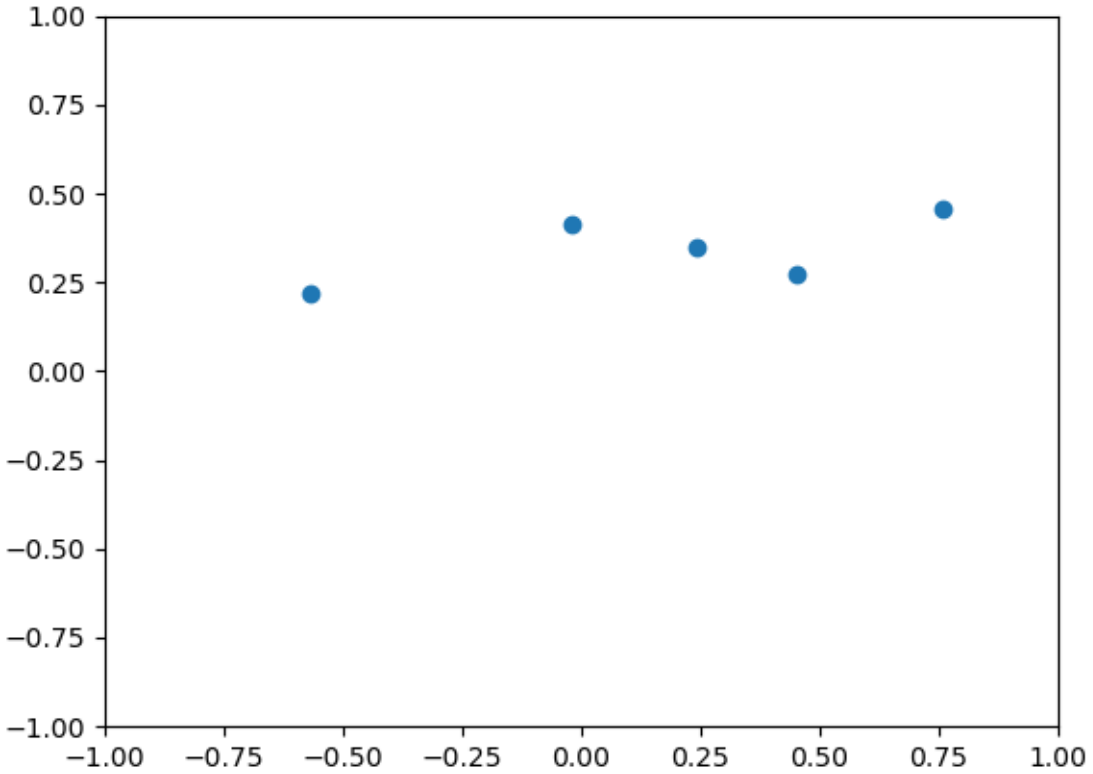
$$p(\overrightarrow{y_*}|X_*, X, \vec{y}\,) \sim N(\overrightarrow{\mu_*}, \Sigma_*) \text{ where}$$

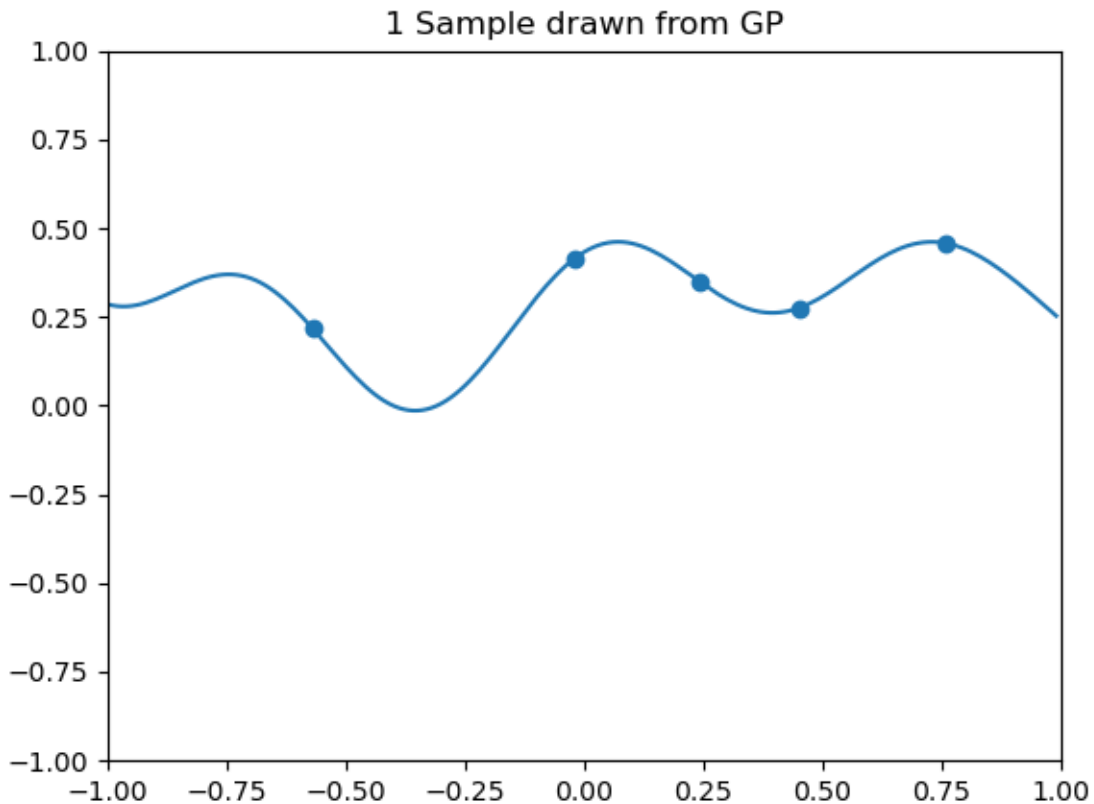
$$\overrightarrow{\mu_*} = {K_*}^T K^{-1} \vec{y}$$

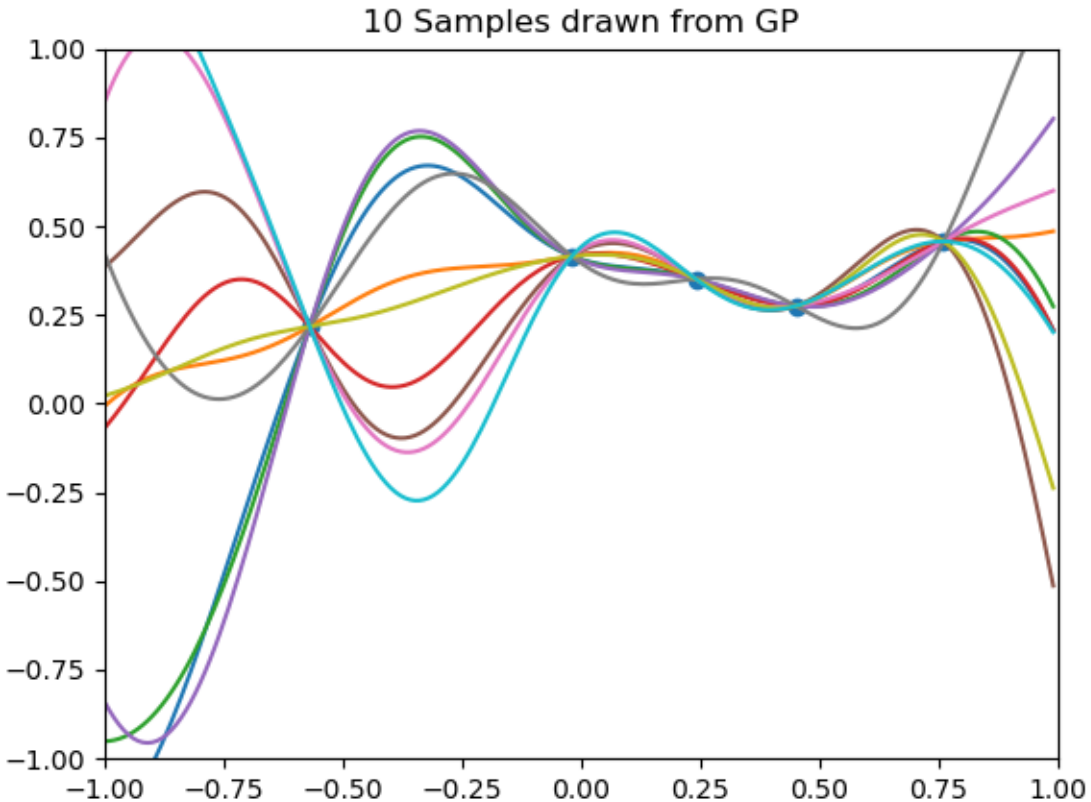
$$\Sigma_* = K_{**} - {K_*}^T K^{-1} K_*$$

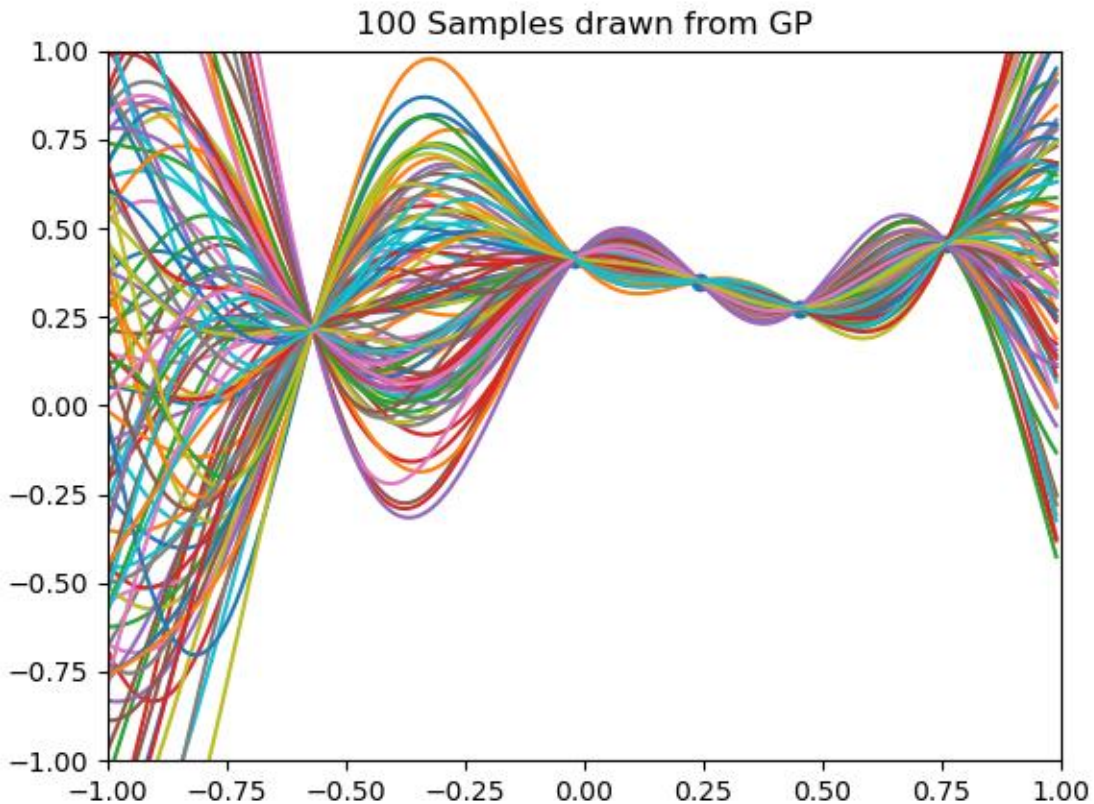
Thus, the GP now has a prediction for the mean and covariance matrix, and thus the standard deviation, of $\overrightarrow{y_*}$. [5]

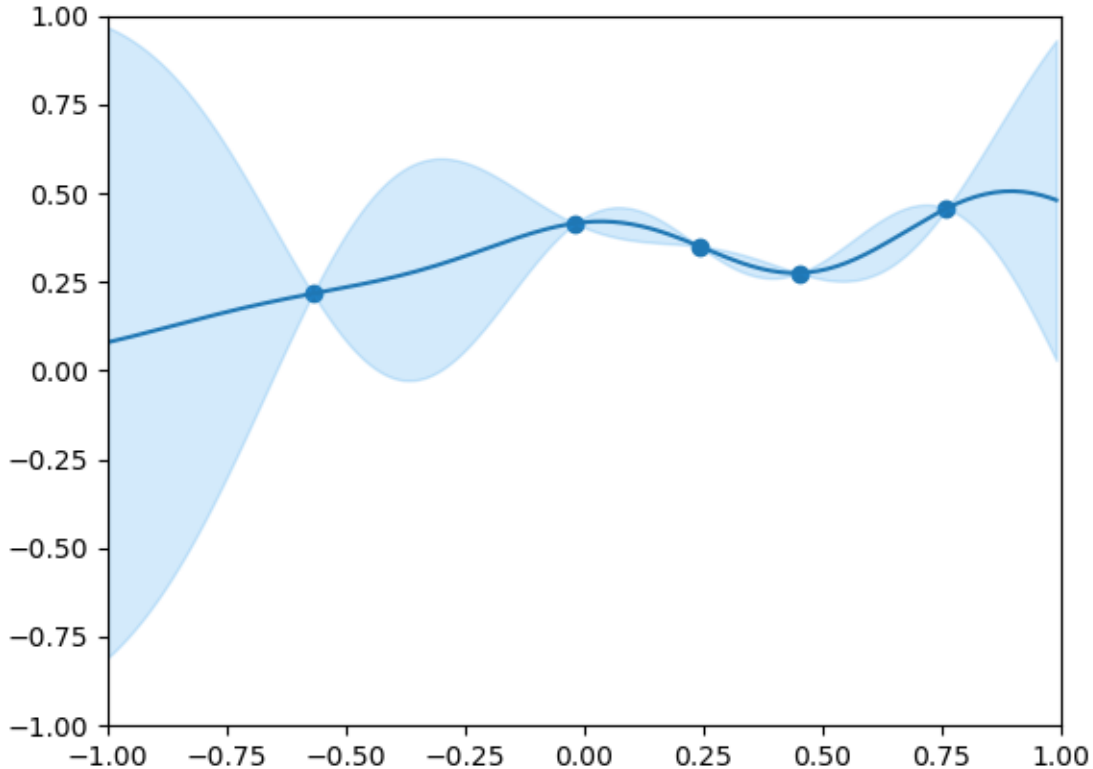
Example


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


1 Sample drawn from GP
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


10 Samples drawn from GP
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


100 Samples drawn from GP
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00