# Contents

# Chapter 1

# Evaluating Cross Sections with Gaussian Processes

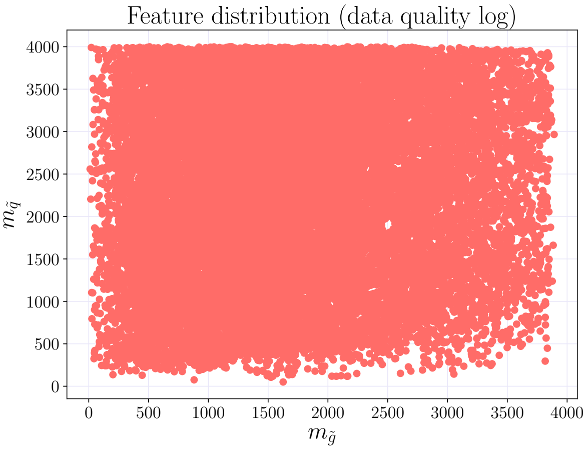
## 1.1 Data Generation

How was the data generated?

- NLL-FAST
- SOFTSUSY

### Feature Distributions

Lin and log distributions.



**Data Quality**

## 1.2 Dataset Transformations

As previously mentioned, the partonic cross sections can be written in terms of scaling functions $f$

$$\hat{\sigma}_{ij} = \frac{\alpha_s^2(Q^2)}{m^2}\left\{f_{ij}^B(\eta, r) + 4\pi\alpha_s(Q^2)\left[f_{ij}^{V+S}(\eta, r, r_t) + f_{ij}^H(\eta, r) + \bar{f}_{ij}(\eta, r)\log\left(\frac{Q^2}{m^2}\right)\right]\right\},$$

where

$$\eta = \frac{s}{m^2} - 1, \qquad r = \frac{m_{\tilde{g}}^2}{m_{\tilde{q}}^2}, \qquad r_t = \frac{m_t^2}{m^2}.$$

The scaling functions are the different contributions, as explained in Chapter 2, and $m = (\sqrt{p_1^2} + \sqrt{p_2^2})/2$ is the average mass of the particles produced.

The energy near the threshold is the base for an important part of the contributions to the cross section [1]. In this region the scaling functions can be expanded in $\beta$, the low velocity of produced particles, leading to the following expressions [1]

$$\begin{aligned}
f_{qq}^B &= \frac{8\pi\beta m_{\tilde{q}}^2 m_{\tilde{g}}^2}{27(m_{\tilde{q}}^2 + m_{\tilde{g}}^2)^2}, & f_{q'q}^B &= \frac{8\pi\beta m_{\tilde{q}}^2 m_{\tilde{g}}^2}{9(m_{\tilde{q}}^2 + m_{\tilde{g}}^2)^2} \\
f_{qq}^{V+S} &= f_{qq}^B \frac{1}{24\beta} & f_{q'q}^{V+S} &= f_{q'q}^B \frac{1}{24\beta} \\
f_{qq}^H &= f_{qq}^B\left[\frac{2}{3\pi^2}\log^2(8\beta^2) - \frac{7}{2\pi^2}\log(8\beta^2)\right] & f_{q'q}^H &= f_{q'q}^B\left[\frac{2}{3\pi^2}\log^2(8\beta^2) - \frac{19}{6\pi^2}\log(8\beta^2)\right] \\
\bar{f}_{qq} &= -f_{qq}^B \frac{2}{3\pi^2}\log(8\beta^2) & \bar{f}_{q'q} &= -f_{q'q}^B \frac{2}{3\pi^2}\log(8\beta^2).
\end{aligned} \tag{1.1}$$

Seeing as the main part of the contributions originate from this energy region, it may be possible to remove some of the complexity of the function using the expressions in Eq. 1.1. Possible transformations are

$$\sigma \rightarrow \sigma_{m_{\tilde{g}}} = \frac{\sigma}{m_{\tilde{g}}^2}, \tag{1.2}$$

$$\sigma \rightarrow \sigma_{m_{\tilde{q}}^2} = \frac{\sigma}{m_{\tilde{q}}^2}. \tag{1.3}$$

Plots of $\sigma_{m_{\tilde{g}}}$ and $\sigma_{m_{\tilde{q}}}$ for the production of $\tilde{d}_L\tilde{d}_L$ as a function of $m_{\tilde{g}}$ and $m_{\tilde{q}}$ are found in Fig. (1.1).

**Figure 1.1:** The cross sections $\sigma$, $\sigma_{m_{\tilde{g}}}$ and $\sigma_{\tilde{q}}$ as a function of the gluino mass $m_{\tilde{q}}$ and squark mass $m_{\tilde{q}}$ for the production of $\tilde{d}_L\tilde{d}_L$. Here, $m_{\tilde{q}} = m_{\tilde{d}_L}$. The cross sections have less spread when some of the mass dependency is removed, which may make learning easier.

## 1.3 The Benchmark

GP with

- 2000 training points + 20 000 test points
- Kernel: RBF
- Process: $\tilde{d}_L\tilde{d}_L$ and $\tilde{d}_L\tilde{u}_L$.
- Features: Physical masses $m_{\tilde{d}_L}$ and $m_{\tilde{g}}$ (and $m_{\tilde{u}_L}$).
- Known error.
- Outliers removed (mask for sigma==0).

**Figure 1.2:** The relative deviance $\varepsilon$ as a function of the logarithm of the normalized cross section for $\tilde{d}_L\tilde{d}_L$. 2000 training points and 20 000 test points were used on a regular GP, with the RBF kernel for $\boldsymbol{\ell} = [1000, 1000]$ and $\alpha = 7.544 \cdot 10^{-7}$. Features are the physical masses $m_{\tilde{d}_L}$ and $m_{\tilde{g}}$.

## 1.4 Outliers

Include and remove the outliers set to zero by prospino.

**Figure 1.3:** The relative deviance $\varepsilon$ as a function of the logarithm of the normalized cross section for $\tilde{d}_L\tilde{d}_L$, with and without outlier points with $\sigma_{\tilde{d}_L\tilde{d}_L} = 0$. Otherwise, the settings are the BM settings. The prediction becomes very chaotic when outliers are included, also in the region of large cross sections.

**Figure 1.4:** True values for the logarithm of $\sigma_{\tilde{d}_L\tilde{d}_L}$ as a function of $m_{\tilde{d}_L}$, with the BM settings and outliers included. Outliers are circled in purple, they are initially 0, but set to $\epsilon = 10^{-32}$ to avoid NaN in the calculation.

**Figure 1.5:** GP predicted values for the logarithm of $\sigma_{\tilde{d}_L\tilde{d}_L}$ as a function of $m_{\tilde{d}_L}$, with the BM settings and outliers included. The outliers have not been predicted, but the rest of the function values are smeared out around that area.

**Figure 1.6:** GP predicted values for the logarithm of $\sigma_{\tilde{d}_L\tilde{d}_L}$ as a function of $m_{\tilde{d}_L}$, with the BM settings and outliers *not* included. The function values are less smeared out at all orders of magnitude.

## 1.5 Cuts on Cross Sections

Including lower powers of the cross section give worse predicitons.

**Figure 1.7:** The relative deviance $\varepsilon$ as a function of the logarithm of the normalized cross section for $\tilde{d}_L\tilde{d}_L$, with and without $\sigma_{\tilde{d}_L\tilde{d}_L} < 10^{-16}$. Otherwise the settings are the BM settings. The prediction becomes better at high cross sections when $\sigma < 10^{-16}$ are removed, and this area is well under the 1 event limit.

## 1.6 Noise Term

### Proof for Relative Errors

The error in the observations comes from the numerical error (?) in the Prospino calculation. The relative error has a standard deviation of $\varepsilon = 0.002$ multiplied by the cross section. The question is *whether we can use this information* when doing the GP fit. Denote the cross section provided by Prospino as $Y_i$ and the real cross section as $y_i^{true}$, we then have

$$Y_i = y_i^{true} + \epsilon_i = y_i^{true}(1 + \varepsilon_i). \tag{1.4}$$

However, we do not calculate the prediction of the cross section, but the *logarithm* of the cross section. The distributions are then

$$Y_i = \mathcal{N}(y_i^{true}, \varepsilon y_i^{true}), \tag{1.5}$$

where the only random variable is $\varepsilon$. Changing variables to $\log_{10}$ gives

$$X_i = \log_{10} Y_i \to Y_i = 10^{X_i} \tag{1.6}$$

$$P_{X_i}(X_i) = P_{Y_i}(Y_i)\Big|\frac{\partial Y_i}{\partial X_i}\Big| \tag{1.7}$$

$$= P_{Y_i}(y_i)10^{X^i}\log 10 \tag{1.8}$$

$$= \mathcal{N}(10^{x_i^{true}}, \varepsilon 10^{x_i^{true}}) \cdot 10^{X_i} \cdot \log 10. \tag{1.9}$$

This means that the relevant distribution is in fact

$$X_i = \log_{10} Y_i = \log_{10} y_i^{true} + \log_{10}(1 + \mathcal{N}(0,\varepsilon)),$$

where we can make the expansion

$$\log_{10}(1 + \mathcal{N}(0,\varepsilon)) \simeq \frac{\mathcal{N}(0,\varepsilon)}{\log 10} - \frac{\mathcal{N}(0,\varepsilon)^2}{\log 100} + ... \tag{1.10}$$

Since the leading order term is clearly the dominant term, the logarithm of the cross section may be written as

$$X_i \simeq \log_{10} y_i^{true} + \frac{1}{\log 10}\mathcal{N}(0,\varepsilon) \tag{1.11}$$

Since the distribution has a standard deviation $\varepsilon = 2 \cdot 10^{-3}$, the Gaussian noise covariance should be

$$\Big(\frac{\varepsilon}{\log 10}\Big)^2 = \frac{(2 \cdot 10^{-3})^2}{(\log 10)^2} = \frac{4 \cdot 10^{-6}}{5.301} \simeq 7.544 \cdot 10^{-7}.$$

Compare WhiteKernel, setting $\alpha$ by hand, and not including noise.

**Figure 1.8:** The relative deviance $\varepsilon$ as a function of the logarithm of the normalized cross section for $\tilde{d}_L\tilde{d}_L$, with error terms $\alpha = 1\cdot 10^{-3}, 7.544\cdot 10^{-7}, 1\cdot 10^{-9}$. Otherwise the settings are the BM settings. The smaller and larger errors, $10^{-9}$ and $10^{-3}$, somewhat improve the prediction in the high and low cross sections, respectively, but $7.544 \cdot 10^{-7}$ gives the best overall prediction.

## 1.7 Features

### 1.7.1 Lagrangian Masses

Compare with physical masses, show that we either need way too many parameters, or the fit is bad.

### 1.7.2 Mean Mass

Because of how Prospino calculates NLO terms.

Compare with and without the mean mass.

**Figure 1.9:** The relative deviance $\varepsilon$ as a function of the logarithm of the normalized cross section for $\tilde{d}_L\tilde{d}_L$, with features $(m_{\tilde{d}_L}, m_{\tilde{g}})$ and $(m_{\tilde{d}_L}, m_{\tilde{g}}, \bar{m})$. Otherwise the settings are the BM settings. The fit is worse when another feature is added, except for the highest cross sections, where it is significantly better.

Since the BDT has shown to be very dependent on $\bar{m}$, and the prediction is better at higher cross sections, we wish to continue trying, this time with another kernel.

## 1.8 Kernel

Compare Matern and RBF

**Figure 1.10:** The relative deviance $\varepsilon$ as a function of the logarithm of the normalized cross section for $\tilde{d}_L\tilde{d}_L$, with kernels RBF for the features $(m_{\tilde{d}_L}, m_{\tilde{g}})$ and Matérn for the features $(m_{\tilde{d}_L}, m_{\tilde{g}}, \bar{m})$. Otherwise the settings are the BM settings. For the three features $m_{\tilde{d}_L}$, $m_{\tilde{g}}, \bar{m}$, the Matern kernel gives a better fit.

### 1.8.1 Hyperparameters

Plot of different $\nu$-values?

### 1.8.2 Target Transformation

For the Matérn kernel with features $(m_{\tilde{d}_L}, m_{\tilde{g}}, \bar{m})$ the altered cross section $\sigma_{m_{\tilde{g}}}$ is tested. Could be because the scaling functions depend on the quantity $r = m_{\tilde{g}}^2/m_{\tilde{q}}^2$.

The functions as a function of $m_{\tilde{g}}$ and $m_{\tilde{d}_L}$ become more similar at high cross sections, so the kernel is more accurate. The band is wider but with less spread for the cross section as a function of the squark mass.

**Figure 1.11:** The relative deviance $\varepsilon$ as a function of the logarithm of the normalized cross section for $\tilde{d}_L\tilde{d}_L$, with kernel Matérn for the features $(m_{\tilde{d}_L}, m_{\tilde{g}}, \bar{m})$. One model is trained on $\sigma_{\tilde{d}_L\tilde{d}_L}$ and the other on $\hat{\sigma}_{\tilde{d}_L\tilde{d}_L} = \sigma_{\tilde{d}_L\tilde{d}_L}/m_{\tilde{g}}$. Otherwise the settings are the BM settings.

For 2000 training points, BM:

- C RBF + WK
- $\alpha = 10^{-10}$
- $(m_{\tilde{g}}, m_{\tilde{q}})$
- No limits
- Removed outliers

Kernels dLdL $\sigma_{m_{\tilde{g}}}$ are:

- BM : `54.6**2 * RBF(length_scale=[5.47e+03, 2.19e+03]) + WhiteKernel(noise_level=`
- Outliers : `98.5**2 * RBF(length_scale=[5.74e+03, 215]) + WhiteKernel(noise_level=0`
- Cut at $10^{-16}$ : `22.7**2 * RBF(length_scale=[1.17e+03, 998]) + WhiteKernel(noise_leve`
- mean : `33.1**2 * RBF(length_scale=[1.19e+03, 200, 846]) + WhiteKernel(noise_leve`
- matern : `21.8**2 * Matern(length_scale=[2.02e+03, 3.29e+03], nu=1) + WhiteKernel(`

Kernels dLuL $\sigma_{m_{\tilde{g}}}$ are:

- BM : `49.9**2 * RBF(length_scale=[5.87e+03, 4e+03, 2.22e+03]) + WK(noise_level=0.`
- Outliers : `80.2**2 * RBF(length_scale=[4.42e+03, 3.1e+03, 240]) + WK(noise_level=0`
- Cut at $10^{-16}$ : `100**2 * RBF(length_scale=[1.54e+03, 2.9e+03, 2.15e+03]) + WK(noise_`
- mean : `61.7**2 * RBF(length_scale=[1.34e+03, 3.59e+04, 251, 748]) + WhiteKernel(`
- matern : `20.6**2 * Matern(length_scale=[1.93e+03, 1.93e+03, 9.22e+03], nu=1) + Wh`

## 1.9 Distributed Gaussian Processes

Time plots.

Matrix with number of experts and training points per expert, with mean relative deviance.

| Training points | 2000 p/ expert | GP |
|---:|:---:|:---:|
| 2000 | 00:03:32 | 00:03:32 |
| 4000 | 00:04:39 | 00:16:33 |
| 6000 | 00:04:10 | |
| 8000 | 00:05:29 | |
| 10 000 | 00:05:46 | |
| 12 000 | 00:06:31 | |
| 14 000 | 00:05:46 | |

### 1.9.1 Adding Experts

**Figure 1.12:** The $R^2$-factor defined in Eq. () for an increasing number of experts, where each expert has 1000 (blue) and 5000 (red) training points. Cross section for $\tilde{d}_L\tilde{d}_L$. A value of 1 is a perfect prediction, and 0 is a bad prediction.

### 1.9.2 Cross validation for DGP

There is no `scikit-learn` function for DGP, so we've implemented a method for calculating the learning curve of a DGP when experts are added. The program uses $k$-fold cross validation with loss function $R^2$.

# Bibliography

[1] Wim Beenakker, R Höpker, Michael Spira, and PM Zerwas. Squark and gluino production at hadron colliders. *Nuclear Physics B*, 492(1-2):51–103, 1997.