Gaussian Processes for Supersymmetry

by

Ingrid Holm

Thesis

for the degree of

Master of Science

Faculty of Mathematics and Natural Sciences
University of Oslo

May 2018

Abstract

This is an abstract text.

To someone

This is a dedication to my cat.

Acknowledgements

I acknowledge my acknowledgements.

Contents

Chapter 1

Introduction

- why nlo?
    - why gp?
    - why dgp?

Chapter 2

Physics Background

This chapter is about supersymmetry. Familiarity with quantum field theory, the Standard Model of particle physics and some group theory is assumed. The Higgs mechanism and the hierarchy problem are reviewed, before supersymmetry is outlined. Finally, the Minimal Supersymmetric Standard Model is introduced, with its corresponding field content.

2.1 The Standard Model

The Standard Model of particle physics has successfully explained almost all experimental results and predicted several phenomena in particle physics. One of the main attributes of this model is that particles with different values of the *spin* quantum number behave differently. Particles with half-integer and integer spin values are called *fermions* and *bosons*, respectively. Fermions are particles such as quarks and leptons, which interact through the exchange of bosons. The Standard Model bosons are the *photon* (electromagnetic interaction), the *gluon* (strong interaction that holds atoms together), the W and Z bosons (the weak interaction) and the famously elusive *Higgs boson*. The equations of motion and allowed interactions can all be derived from the *Lagrangian* of the Standard Model. The Lagrangian, of which the time integral is the action S, is invariant to transformations under the Lorentz group — or a change of reference frame in the language of special relativity.

The Higgs Mechanism

The Standard Model is a gauge theory based on the symmetry group $SU(3)_C \times SU(2)_L \times U(1)_Y$. The $SU(3)$ group is the symmetry group for strong interactions, or quantum chromodynamics, and $SU(2)_L \times U(1)_Y$ is the electroweak symmetry group. In order for the particles to obtain masses the electroweak symmetry must be spontaneously broken down to $U(1)_{em}$. The symmetry is broken when

the Higgs field obtains a non-zero *vacuum expectation value* (vev) — meaning that it has some field value when the governing potential is at its minimum. The Higgs field Φ is a self-interacting complex $SU(2)_L$ doublet whose Lagrangian is given by

$$\mathcal{L}_\Phi = \partial_\mu \Phi^\dagger \partial^\mu \Phi + V(\Phi), \tag{2.1}$$

where the first term is the kinetic term, and the scalar potential describing the Higgs $V(\Phi)$ is the famous Mexican hat potential

$$V(\Phi) = \mu^2 \Phi^\dagger \Phi + \lambda (\Phi^\dagger \Phi)^2. \tag{2.2}$$

For $\mu^2 < 0$ and $\lambda > 0$ this potential aquires a non-trivial minimum given by

$$|\Phi_0| = \sqrt{\frac{-\mu^2}{2\lambda}} \equiv \frac{v}{\sqrt{2}}, \tag{2.3}$$

where v is the vacuum expectation value. A special parameterization of the Higgs $SU(2)_L$ doublet, $\Phi^T(x) = \frac{1}{\sqrt{2}}(0, v + h(x))$, leads to the Lagrangian developing mass terms for fermions and the gauge bosons Z and $W^\pm$. The mass terms are proportional to v, *e.g.*

$$M_W = \frac{1}{2} v g,$$

where M_W is the mass of the W and g is the $SU(2)_L$-coupling. The Higgs' own mass provides one of the strongest arguments for introducing supersymmetry, namely the *hierarchy problem*, which is discussed below.

Another argument for the introduction of supersymmetry is gauge coupling unification. Gauge coupling unification is the assumption that the Standard Model symmetry group is a unified gauge group, *e.g.* $SU(5)$ or $SO(10)$, broken down to $SU(3)_C \times SU(2)_L \times U(1)_Y$ at some high energy scale. However, this is not discussed in this thesis.

2.2 The Hierarchy Problem

The Higgs boson was discovered at the LHC in 2012, and its mass measured at $m_H \sim 125$ GeV [**?**]. The expression for the Higgs mass in the Standard Model includes loop corrections, which provide a large discrepancy between theory and experiment. The Higgs mass receives fermionic or scalar loop-contributions to its mass such as those shown in Fig. (2.1). The expression for the mass can thus be written in terms of the bare parameter m_{H0} and the corrections Δm_H

$$m_H^2 = m_{H0}^2 + \Delta m_H^2.$$

Loop diagrams contain divergences, because of integrals over all possible momenta for the virtual particles in the loops. A way to get rid of these infinities is to regularize the expressions. Regularization is a neat trick that introduces a *cut-off scale*, which sets an upper limit on the momentum that is integrated over. A common choice for the cut-off scale Λ is the Planck scale, as this is where new physics is needed to explain gravity in the Standard Model. The Planck scale is of the order of $\Lambda_{UV} \sim 10^{18}$ GeV. After regularization, the mass correction terms are

$$\Delta m_H^2 = -\frac{|\lambda_f|^2}{8\pi^2}\Lambda_{UV}^2 + \frac{\lambda_s}{16\pi^2}\Lambda_{UV}^2 + ..., \tag{2.4}$$

where λ_s is the coupling of the Higgs to the scalar, and λ_f is the Higgs coupling to the fermion. The problem now becomes apparent: the correction to the mass is proportional to the Planck scale, placing it at the order of 10^{18} Gev, yet the mass has been experimentally measured around 125 GeV. There must be some colossal cancellation of terms with a tremendous tuning of the SM parameters in λ_s and λ_f^2. Tuning of parameters is undesirable — the model should be as natural as possible.

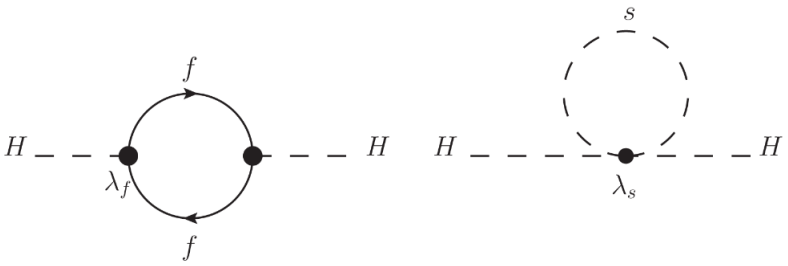


Figure 2.1: Fermion and scalar one-loop corrections to the Higgs mass. Figure from [1].

Supersymmetry provides an elegant solution to the hierarchy problem. In simple terms, supersymmetry introduces a fermionic superpartner for each boson, and vice versa. These are called sparticles, and have a prefix -s for the partners of fermions, such as *squarks* and *leptons*, and the suffix -ino for partners of bosons, such as the *photino* and *Higgsino*. In unbroken supersymmetry these particles have identical mass, and their couplings to the Higgs are the same $\lambda_s = |\lambda_f|^2$. In addition, there are twice as many scalars as fermions, which gives a perfect cancellation of these enormous corrections. Unbroken supersymmetry therefore solves the hierarchy problem. The case of broken supersymmetry is revisited in a later section.

2.3 Supersymmetry

Supersymmetry is an extension of symmetries. Relativistic field theories are invariant under boosts, rotations and translations in spacetime. These are called Poincaré transformations, and are given by

$$x^\mu \to x'^\mu = \Lambda^\mu{}_\nu x^\nu + a^\mu, \tag{2.5}$$

where $\Lambda^\mu{}_\nu$ is a Lorentz transformation and a^μ is a translation. The assumption behind supersymmetry is that Nature obeys a non-trivial extension of the related Poincaré algebra, namely the *superalgebra*.

The elements of the superalgebra and their representations can be described using *superspace*. Coordinates in superspace are given by $z^\pi = (x^\mu, \theta^A, \bar{\theta}_{\dot{A}})$, where x^μ are the well-known Minkowski coordinates, and $\theta^A, \bar{\theta}_{\dot{A}}$ are four Grassmann numbers [1]. Any Super-Poincaré transformation can be written in the following way

$$L(a, \alpha) = \exp[-ia^\mu P_\mu + i\alpha^A Q_A + i\bar{\alpha}^{\dot{A}} \bar{Q}_{\dot{A}}]. \tag{2.6}$$

In addition, since P_μ commutes with the generators Q one can always boost between reference frames, so in general a supersymmetry transformation is taken to mean

$$\delta_S = \alpha^A Q_A + \bar{\alpha}_{\dot{A}} \bar{Q}^{\bar{A}}. \tag{2.7}$$

The super-Poincaré algebra is given by the following commutation and anti-commutation relations [2]

$$\{Q_A, Q_B\} = \{\bar{Q}_A, \bar{Q}_B\} = 0, \tag{2.8}$$
$$\{Q_A, \bar{Q}_{\dot{A}}\} = 2(\sigma^\mu)_{A\dot{A}} P_\mu, \tag{2.9}$$
$$[Q_A, P^\mu] = [\bar{Q}_A, P^\mu] = 0, \tag{2.10}$$
$$[Q_A, M^{\mu\nu}] = \frac{1}{2}(\sigma^{\mu\nu})^B_A Q_B, \tag{2.11}$$
$$[\bar{Q}_{\dot{A}}, M^{\mu\nu}] = \frac{1}{2}(\bar{\sigma}^{\mu\nu})^{\dot{b}}_{\dot{a}} Q^\dagger_{\dot{b}}, \tag{2.12}$$

where $Q_A, A = 1, 2, 3, 4$ are the superalgebra generators, P^μ are the generators of translation, and $M^{\mu\nu}$ are the generators of the Lorentz group.

The supersymmetry generators turn fermions into bosons and vice versa. More specifically, these operators have the following commutation relations with the rotation generator J^3

$$[Q_A, J^3] = \frac{1}{2}(\sigma^3)^B_A Q_B, \tag{2.13}$$

[1] Grassmann numbers are numbers that anti-commute.

which for the Q_1 generator becomes

$$[Q_1, J^3] = \frac{1}{2}Q_1. \tag{2.14}$$

Using this operator on a state in an irreducible representation of the Poincaré algebra with mass m and spin j_3 gives

$$J^3 Q_1 \left|m, j_3\right\rangle = (j_3 - \frac{1}{2})Q_1 \left|m, j_3\right\rangle, \tag{2.15}$$

thus lowering the spin of the state by 1/2. Similarly, Q_2 would increase the spin. They do not, however, change the mass. This can be seen from Eq. (2.10)

$$P^\mu P_\mu Q_A \left|m, j_3\right\rangle = Q_A P^\mu P_\mu \left|m, j_3\right\rangle = m^2 Q_A \left|m, j_3\right\rangle. \tag{2.16}$$

States that transform into each other via Q_A and $\bar{Q}_{\dot{A}}$ are called *superpartners*. In unbroken supersymmetry, therefore, the partnering fermions and bosons have the same mass. If this were the case, supersymmetric particles would already have been discovered, so supersymmetry must be a broken symmetry.

2.3.1 Superfields

A supersymmetric Lagrangian will need a derivative that is invariant under supersymmetry transformations. The general covariant derivatives are defined as

$$D_A \equiv \partial_A + i(\sigma^\mu \bar{\theta})_A \partial_\mu, \tag{2.17}$$

$$\bar{D}^{\dot{A}} \equiv -\partial^{\dot{A}} - i(\sigma^\mu \theta)^{\dot{A}} \partial_\mu. \tag{2.18}$$

The covariant derivatives work on the *superfields* Φ, which are functions on superspace $\Phi(x, \theta, \bar{\theta})$. These are affected by the covariant derivatives in the following way

$$\bar{D}_{\dot{A}} \Phi(x, \theta, \bar{\theta}) = 0 \qquad \text{(left-handed scalar superfield)}, \tag{2.19}$$

$$D^A \Phi^\dagger(x, \theta, \bar{\theta}) = 0 \qquad \text{(right-handed scalar superfield)} \tag{2.20}$$

$$\tag{2.21}$$

The fields Φ are required to be Lorentz scalars or pseudoscalars, which restricts the properties of their component fields. It can be shown that the left- and right-handed scalar fields can be written in terms of their component fields as [1]

$$\begin{aligned}\Phi(x, \theta, \bar{\theta}) =& A(x) + i(\theta\sigma^\mu\bar{\theta})\partial_\mu A(x) - \frac{1}{4}\theta\theta\bar{\theta}\bar{\theta}\Box A(x) + \sqrt{2}\theta\psi(x) \\ &- \frac{i}{\sqrt{2}}\theta\theta\partial_\mu\psi(x)\sigma^\mu\bar{\theta} + \theta\theta F(x),\end{aligned} \tag{2.22}$$

$$\begin{aligned}\Phi^\dagger(x, \theta, \bar{\theta}) =& A^*(x) - i(\theta\sigma^\mu\bar{\theta})\partial_\mu A^*(x) - \frac{1}{4}\theta\theta\bar{\theta}\bar{\theta}\Box A^*(x) + \sqrt{2}\bar{\theta}\bar{\psi}(x) \\ &- \frac{i}{\sqrt{2}}\bar{\theta}\bar{\theta}\theta\sigma^\mu\partial_\mu\bar{\psi}(x) + \bar{\theta}\bar{\theta}F^*(x),\end{aligned} \tag{2.23}$$

where $A(x)$ and $F(x)$ are complex scalars and $\psi_A(x)$ and $\bar{\psi}^{\dot{A}}(x)$ are left-handed and right-handed Weyl spinors, respectively.

A vector field is defined by the constraint

$$\Phi(x,\theta,\bar{\theta}) = \Phi^\dagger(x,\theta,\bar{\theta}). \tag{2.24}$$

From Eq. (2.24) the structure of a general vector field should be [1]

$$\begin{aligned}\Phi(x,\theta,\bar{\theta}) =& f(x) + \theta\varphi(x) + \bar{\theta}\bar{\varphi}(x) + \theta\theta m(x) + \bar{\theta}\bar{\theta}m^*(x) \\ &+ \theta\sigma^\mu\bar{\theta}V_\mu(x) + \theta\theta\bar{\theta}\bar{\lambda}(x) + \bar{\theta}\bar{\theta}\theta\lambda(x) + \theta\theta\bar{\theta}\bar{\theta}d(x),\end{aligned} \tag{2.25}$$

where $f(x)$, $d(x)$ are real scalar fields, $\varphi_A(x)$, $\lambda_A(x)$ are Weyl spinors, $m(x)$ is a complex scalar field and $V_\mu(x)$ is a real Lorentz four-vector. An example of a vector field is the product $V = \Phi^\dagger\Phi$. In the $j = \frac{1}{2}$ representation of the superalgebra, this field does not correspond to the promised number of degrees of freedom. This problem is fixed by the super-gauge.

2.3.2 Supersymmetric Lagrangian

Symmetry transformations of the Lagrangian should leave the action

$$S \equiv \int d^4x\mathcal{L}, \tag{2.26}$$

invariant. This is automatically fulfilled if the Lagrangian only changes by a total derivative. It can be shown that the highest order component fields in θ and $\bar{\theta}$ of the scalar and vector superfields have this property. To ensure that the action is invariant under supersymmetry transformations, the Lagrangian is redefined such that

$$S = \int d^4x \int d^4\theta\mathcal{L}, \tag{2.27}$$

where there is now an integral over Grassmann numbers and $d^4\theta = d^2\theta d^2\bar{\theta}$.

Restrictions on the supersymmetric Lagrangian, such as invariance under supersymmetric transformations and renormalizability, mean that the most general Lagrangian as a function of the scalar superfields Φ_i is

$$\mathcal{L} = \Phi_i^\dagger\Phi_i + \bar{\theta}\bar{\theta}W[\Phi] + \theta\theta W[\Phi^\dagger], \tag{2.28}$$

where $\Phi_i^\dagger\Phi_i$ is the kinetic term, and $W[\Phi]$ is the *superpotential*

$$W[\Phi] = g_i\Phi_i + m_{ij}\Phi_i\Phi_j + \lambda_{ijk}\Phi_i\Phi_j\Phi_k, \tag{2.29}$$

where m_{ij} and λ_{ijk} are symmetric. So to specify a supersymmetric Lagrangian all that is needed is to specify the superpotential.

Supergauge

A natural further step is to require that the Lagrangian be gauge invariant. A supergauge transformation (global or local) on left-handed scalar superfields Φ_i is defined as [1]

$$\Phi \to \Phi' = e^{-i\Lambda_a T^a q_i}\Phi, \tag{2.30}$$

where q is the $U(1)$ charge of the superfield Φ, Λ_a are the parameters of the transformation, and T^a are the generators of the gauge group. For a left-handed superfield Φ_i the Λ^a must also be left-handed superfields, and correspondingly a right-handed superfield $\Phi^\dagger$ must have right-handed superfields $\Lambda^{\dagger a}$.

For the Lagrangian to be gauge invariant the potential W must be invariant. From the requirement that $W[\Phi] = W[\Phi']$, the following restrictions on the superpotential follow

$$g_i = 0 \text{ if } g_i U_{ir} \neq g_r, \tag{2.31}$$

$$m_{ij} = 0 \text{ if } m_{ij} U_{ir} U_{js} \neq m_{rs}, \tag{2.32}$$

$$\lambda_{ijk} = 0 \text{ if } \lambda_{ijk} U_{ir} U_{js} U_{kt} \neq \lambda_{rst}, \tag{2.33}$$

where the indices on U are matrix indices.

The kinetic term must also be invariant under gauge transformations. For this term to be invariant, a gauge compensating vector superfield V with the appropriate gauge transformation is introduced. The kinetic term can then be written as $\Phi^\dagger e^{qV^a T_a}\Phi$, and the kinetic term transforms as

$$\Phi^\dagger e^{qV^a T_a}\Phi \to \Phi'^\dagger e^{qV'^a T_a}\Phi' = \Phi^\dagger e^{iq\Lambda^{a\dagger} T_a} e^{qV'^a T_a} e^{-iq\Lambda^a T_a}\Phi, \tag{2.34}$$

meaning that the vector superfield V^a is required to transform as

$$e^{qV'^a T_a} = e^{-iq\Lambda^{a\dagger} T_a} e^{qV^a T_a} e^{iq\Lambda^a T_a}. \tag{2.35}$$

Supersymmetric Field Strength

The supersymmetric Lagrangian also requires field strengths, analogous to the electromagnetic field strength $F_{\mu\nu}$. The supersymmetric field strengths are

$$W_A \equiv -\frac{1}{4}\bar{D}\bar{D}e^{-V} D_A e^V, \tag{2.36}$$

$$\bar{W}_{\dot{A}} \equiv -\frac{1}{4} D D e^{-V} \bar{D}_{\dot{A}} e^V, \tag{2.37}$$

where $V = V^a T_a$. W_A ($\bar{W}_{\dot{A}}$) is a left-handed (right-handed) superfield, and it can be shown that the trace $\mathrm{Tr}[W_A W^A]$ is supergauge invariant [1]. The Lagrangian for a supersymmetric theory with (possibly) non-Abelian gauge groups is then

$$\mathcal{L} = \Phi^\dagger e^V \Phi + \delta^2(\bar{\theta}) W[\Phi] + \delta^2(\theta) W[\Phi^\dagger] + \frac{1}{2T(R)}\delta^2(\bar{\theta})\mathrm{Tr}[W_A W^A], \tag{2.38}$$

where $T(R)$ is the Dynkin index for normalization, $\delta^2(\bar{\theta}) = \bar{\theta}\bar{\theta}$ and $\delta^2(\theta) = \theta\theta$. The Dynkin index of the representation R in terms of matrices T_a is given by $\mathrm{Tr}[T_a, T_b] = T(R)\delta_{ab}$.

2.3.3 Soft Supersymmetry Breaking

As previously mentioned, in unbroken supersymmetry particles and their corresponding sparticles would have the same mass. Since sparticles have not yet been observed, supersymmetry must be a broken symmetry. In this section soft supersymmetry breaking is considered as a way of providing mass to particles, without comprimising the solution to the hierarchy problem.

In the Standard Model particles obtain mass through spontaneous symmetry breaking of the electroweak symmetry, as described in Section 1.1.1. In supersymmetry this mechanism does not work, because it would be required that the sum of scalar particles squared be equal to the sum of fermion masses squared. Since the consequence of this would be that not all scalar partners could be heavier than the known particles, this cannot be the case [1].

In stead, supersymmetry can be broken through *soft breaking*. This entails adding terms to the Lagrangian that break supersymmetry, while preserving the cancellations of divergences that fixes the hierarchy problem. These are called *soft terms*, and there are several restrictions on them. Soft terms should have mass dimension one or higher, so as to avoid divergences from loop contributions to scalar masses. The soft terms can be written in terms of their component fields

$$\begin{aligned}\mathcal{L}_{soft} = &-\frac{1}{2}\lambda^A\lambda_A - \left(\frac{1}{6}a_{ijk}A_iA_jA_k + \frac{1}{2}b_{ij}A_iA_j + t_iA_i + \frac{1}{2}c_{ijk}A_i^*A_jA_k + \text{ c.c.}\right)\\ &- m_{ij}^2A_i^*A_j,\end{aligned} \tag{2.39}$$

where λ_A are Weyl spinor fields and A_i are scalar fields. The soft breaking terms give masses to both the scalar and fermionic superpartners of the SM particles.

Restrictions on the new parameters are necessary to avoid reintroducing the hierarchy problem. If the breaking terms are soft, the correcting mass terms are at most

$$\Delta m_h^2 = -\frac{\lambda_s}{16\pi^2}m_s^2\ln\frac{\Lambda_{UV}}{m_s^2} + ...,$$

at leading order in the breaking scale Λ_{UV}, where m_s is the soft breaking scale. In this scheme m_s is restricted to $m_s \sim \mathcal{O}(1\text{ TeV})$.

2.4 The Minimal Supersymmetric Standard Model

The Minimal Supersymmetric Standard Model (MSSM) is 'minimal' in the sense that it requires the least amount of new fields introduced in order to have all the SM fields and supersymmetry. The MSSM is based on the minimal extension of the Poincaré algebra. In this section the field content of the MSSM and the introduction of R-parity is discussed.

2.4.1 Field Content

Each left-handed scalar superfield has a left-handed Weyl spinor and a complex scalar. After using the equations of motion, these have two fermionic and two bosonic degrees of freedom remaining each. In combination with a right-handed Weyl spinor, one can construct a Dirac fermion. The right-handed Weyl spinor can be aquired from a *different* scalar superfield. There are now four fermionic degrees of freedom, from which two Dirac fermions can be constructed. These constitute a fermionic particle-antiparticle pair and four scalars, namely two particle-antiparticle pairs.

For leptons the scalar superfields are

$$L_i = \begin{pmatrix} \nu_i \\ l_i \end{pmatrix} \quad \text{and} \quad \bar{E}_i, \tag{2.40}$$

where L_i are $SU(2)_L$ doublets, the superfields l_i and $\bar{E}_i$ for charged leptons, and ν_i for (left-handed) neutrinos, and $i = 1, 2, 3$ is the generation index. These fields and their Hermitian conjugates are used to construct the Standard Model leptons and their superpartners, the sleptons. Note that there is no right-handed $\bar{N}_i$. This is a convention, as MSSM is older than the discovery of massive neutrinos. Similarly, for up-type and down-type quarks the superfields are

$$Q_i = \begin{pmatrix} u_i \\ d_i \end{pmatrix}, \ \bar{U}_i \quad \text{and} \quad \bar{D}_i. \tag{2.41}$$

These fields and their Hermitian cojugates are used to construct quarks and squarks. Color indices of the quarks are omitted for simplicity.

Vector superfields are needed to construct the gauge bosons. After applying the equations of motion these contain a massless vector boson with two scalar degrees of freedom, and one Weyl spinor of each handedness, with two fermionic degrees of freedom. Consider the definition in the previous section, where $V = T_a V^a$. This implies that one superfield V^a is needed per generator of the algebra T_a. These superfields are called C^a, W^a and B^0. The fermions constructed from the corresponding Weyl spinors have the symbols $\tilde{g}$ (gluino), $\tilde{W}^0$ (wino) and

Supermultiplet	Scalars	Fermions	Vectors	$SU(3)_c$	$SU(2)_L$	$U(1)_Y$
Q_i	$(\tilde{u}_{iL}, \tilde{d}_{iL})$	(u_{iL}, d_{iL})		3	2	$\frac{1}{6}$
$\bar{u}_i$	$\tilde{u}^*_{iR}$	$u^\dagger_{iR}$		$\bar{3}$	1	$-\frac{2}{3}$
$\bar{d}_i$	$\tilde{d}^*_{iR}$	$d^\dagger_{iR}$		$\bar{3}$	1	$\frac{1}{3}$
L_i	$(\tilde{\nu}_{iL}, \tilde{e}_{iL})$	(ν_{iL}, e_{iL})		1	2	$-\frac{1}{2}$
$\bar{e}_i$	$\tilde{e}^*_{iR}$	$e^\dagger_{iR}$		1	1	1
H_u	(H^+_u, H^0_u)	$(\tilde{H}^+_u, \tilde{H}^0_u)$		1	2	$\frac{1}{2}$
H_d	(H^0_d, H^-_d)	$(\tilde{H}^0_d, \tilde{H}^-_d)$		1	2	$-\frac{1}{2}$
g		$\tilde{g}$	g	8	1	0
W		$\tilde{W}^{1,2,3}$	$W^{1,2,3}$	1	3	0
B		$\tilde{B}$	B	1	1	0

Table 2.1: Gauge and chiral supermultiplets in the Minimal Supersymmetric Standard Model with SM gauge group representations. The index $i = 1, 2, 3$ runs over the three generations of quarks and lepton. Table from [2].

$\tilde{B}^0$ (bino). Tilde above the symbol indicates that these are the supersymmetric partners of the known SM particles, or the *sparticles*.

Finally, superfields are needed for the Higgs. The superfield version of the SM Higgs $SU(2)_L$ doublet would mix left- and right-handed superfields, and so cannot appear in the superpotential. The minimal allowed Higgs content are two $SU(2)_L$ Higgs doublets H_u and H_d, indexed according to the quarks they give mass to. The doublets are

$$H_u = \begin{pmatrix} H^+_u \\ H^0_u \end{pmatrix}, \qquad H_d = \begin{pmatrix} H^0_d \\ H^-_d \end{pmatrix}. \tag{2.42}$$

These left-handed superfields contain in total four Weyl spinors and eight bosonic degrees of freedom. Three degrees of freedom are used to give masses to the $W^\pm$ and Z^0 bosons through the Higgs mechanism. The remaining five are manifest through the mass eigenstates h^0, H^0, A^0 and $H^\pm$. The Weyl spinors combine into the *higgsinos*. The entire field content of the MSSM can be found in Table (2.1).

The Lagrangian for the MSSM may now be constructed, consisting of kinetic terms $\mathcal{L}_{\text{kin}}$, supersymmetric field strength terms $\mathcal{L}_V$, the superpotential terms $\mathcal{L}_W$ and the soft breaking terms $\mathcal{L}_{\text{soft}}$,

$$\mathcal{L}_{\text{MSSM}} = \mathcal{L}_{\text{kin}} + \mathcal{L}_V + \mathcal{L}_W + \mathcal{L}_{\text{soft}}. \tag{2.43}$$

The kinetic terms are constructed from the fields introduced above

$$\begin{aligned}\mathcal{L}_{\text{kin}} =& L_i^\dagger e^{\frac{1}{2}g\sigma W-\frac{1}{2}g'B}L_i + Q_i^\dagger e^{\frac{1}{2}g_s\lambda C+\frac{1}{2}g\sigma W+\frac{1}{3}\cdot\frac{1}{2}g'B}Q_i \\ &+ \bar{U}_i^\dagger e^{\frac{1}{2}g_s\lambda C-\frac{4}{3}\cdot\frac{1}{2}g'B}\bar{U}_i + \bar{D}_i^\dagger e^{\frac{1}{2}g_s\lambda C-\frac{2}{3}\cdot\frac{1}{2}g'B}\bar{D}_i \\ &+ \bar{E}_i^\dagger e^{2\frac{1}{2}g'B}\bar{E}_i + H_u^\dagger e^{\frac{1}{2}g\sigma W+\frac{1}{2}g'B}H_u + H_d^\dagger e^{\frac{1}{2}g\sigma W-\frac{1}{2}g'B}H_d, \end{aligned} \tag{2.44}$$

where g, g' and B are the couplings of the $U(1)_Y$, $SU(2)_L$ and the $SU(3)_C$.

The supersymmetric field strength contributions with pure gauge terms are

$$\mathcal{L}_V = \frac{1}{2}\text{Tr}\{W^A W_A\}\bar{\theta}\bar{\theta} + \frac{1}{2}\text{Tr}\{C^A C_A\}\bar{\theta}\bar{\theta} + \frac{1}{4}B^A B_A\bar{\theta}\bar{\theta} + \text{h.c.}, \tag{2.45}$$

with the field strengths W_A, C_A and B_A given by

$$W_A = -\frac{1}{4}\bar{D}\bar{D}e^{-W}D_A e W, \qquad W = \frac{1}{2}g\sigma^a W^a, \tag{2.46}$$

$$C_A = -\frac{1}{4}\bar{D}\bar{D}e^{-C}D_A e^C, \qquad C = \frac{1}{2}g_s\lambda^a C^a, \tag{2.47}$$

$$B_A = -\frac{1}{4}\bar{D}\bar{D}D_A B, \qquad B = \frac{1}{2}g'B^0. \tag{2.48}$$

The gauge invariant terms in the superpotential are

$$\begin{aligned} W =& \mu H_u H_d + \mu' L_i H_u + y^e_{ij}L_i H_d E_j + y^u_{ij}Q_i H_u \bar{U}_j + y^d_{ij}Q_i H_d \bar{D}_j \\ &+ \lambda_{ijk}L_i L_j \bar{E}_k + \lambda'_{ijk}L_i Q_j \bar{D}_k + \lambda''_{ijk}\bar{U}_i\bar{D}_j\bar{D}_k, \end{aligned} \tag{2.49}$$

where $H_u H_d$ is shorthand for $H_u^T i\sigma_2 H_d$ — and similarly for the other doublet pairs — which is a construction invariant under $SU(2)_L$. μ is the Lagrangian mass parameter and μ' is some other mass parameter in the superpotential.

2.4.2 R-parity

The most general supersymmetric Lagrangian with the fields in Sec. 1.4.1 results in couplings that violate lepton and baryon numbers, such as $\mu' L_i H_u$ and $\lambda_{ijk}\bar{U}_i\bar{D}_j\bar{D}_k$. However, these violations are under strict restrictions from experiment, such as the search for proton decay $p \to e^+\pi^0$. Therefore, a new, multiplicative conserved quantity is introduced, namely R-parity

$$P_R \equiv (-1)^{3(B-L)+2s}, \tag{2.50}$$

where s is spin, B is baryon number and L is lepton number. This quantity is $+1$ for SM particles, and -1 for the sparticles. If R-parity is to be conserved sparticles must therefore always be produced and annihilated in pairs. A further consequence is that there must exist a stable, *lightest supersymmetric particle* (LSP), to which all other supersymmetric particles decay eventually. For this particle to have gone undetected it should have zero eletric and color charge. These properties make the LSP a good candidate for dark matter [3].

2.4.3 Soft Breaking terms

The allowed soft breaking terms that conserve R-parity and gauge invariance are, in component fields, as follows

$$\begin{aligned}\mathcal{L}_{\text{soft}} = &-\frac{1}{2}M_1\tilde{B}\tilde{B} + M_2\tilde{W}^a\tilde{W}^a + M_3\tilde{g}^a\tilde{g}^a + c.c. \\ &- a^u_{ij}\tilde{Q}_i H_u \tilde{u}^*_{jR} - a^d_{ij}\tilde{Q}_i H_d \tilde{d}^*_{iR} - a^e_{ij}\tilde{L}_i H_d \tilde{e}^*_{jR} + c.c. \\ &- (m^2_u)_{ij}\tilde{u}^*_{iR}\tilde{u}_{jR} - (m^2_d)_{ij}\tilde{d}^*_{iR}\tilde{d}_{jR} - (m^2_e)_{ij}\tilde{e}^*_{iR}\tilde{e}_{jR} \\ &- (m^2_Q)_{ij}\tilde{Q}^\dagger_i\tilde{Q}_j - (m^2_L)_{ij}\tilde{L}^\dagger_i\tilde{L}_j \\ &- m^2_{H_u}H^*_u H_u - m^2_{H_d}H^*_d H_d - (bH_u H_d + c.c.),\end{aligned} \tag{2.51}$$

where the M_i are potentially complex valued, introucing six new parameters; the a_{ij} are potentially complex values, introducing 54 new parameters, b is potentially complex values, introducing two new parameters; the m^2_{ij} are complex valued and hermitian, introducing 47 new parameters. After removing excessive degrees of freedom, the MSSM Lagrangian has introduced a total of 105 new parameters, where 104 come from the soft terms and μ comes from the superpotential.

2.4.4 Radiative Electroweak Symmetry Breaking

As discussed in Sec. 1.1, the SM particles obtain their mass when the Higgs has a field value at the minimum of its governing potential. In supersymmetry, the scalar potential for the Higgs component fields is

$$\begin{aligned}V(H_u, H_d) =&|\mu|^2(|H^0_u|^2 + |H^+_u|^2 + |H^0_d|^2 + |H^-_d|^2) \\ &+ \frac{1}{8}(g^2 + g'^2)(|H^0_u|^2 + |H^+_u|^2 - |H^0_d|^2 - |H^-_d|^2)^2 \\ &+ \frac{1}{2}g^2|H^+_u H^{0*}_d + H^0_u H^{-*}_d|^2 \\ &+ m^2_{H_u}(|H^0_u|^2 + |H^+_u|^2) + m^2_{H_d}(|H^0_d|^2 + |H^-_d|^2) \\ &+ [b(H^+_u H^-_d - H^0_u H^0_d) + \text{c.c.}]\end{aligned} \tag{2.52}$$

Using gauge freedom, this potential can be simplified to

$$\begin{aligned}V(H^0_u, H^0_d) =&(|\mu|^2 + m^2_{H_u})|H^0_u|^2 + (|\mu|^2 + m^2_{H_d})|H^2_d| \\ &+ \frac{1}{8}(g^2 + g'^2)(|H^0_u|^2 - |H^0_d|^2)^2 - (bH^0_u H^0_d + \text{c.c.})\end{aligned} \tag{2.53}$$

Analogous to the SM, $SU(2)_L \times U(1)_Y$ should be broken down to $U(1)_{em}$ in order to give masses to gauge bosons and SM fermions. It can be shown that this potential has a minimum for finite field values, that this minimum has a remaining $U(1)_{em}$ symmetry, and that the potential is bounded from below. For

the potential to have a negative mass term and be bounded from below, the following is required

$$b^2 > (|\mu|^2 + m_{H_u}^2)(|\mu|^2 + m_{H_d}^2) \tag{2.54}$$

and

$$2b < 2|\mu|^2 + m_{H_u}^2 + m_{H_d}^2. \tag{2.55}$$

If it is assumed that $m_{H_u} = m_{H_d}$ at some high scale, the requirements Eq. (2.54) and Eq. (2.54) cannot be simultaneously satisfied at that scale. However, to one-loop the Renormalization Group Equation (RGE) [2] for $m_{H_u}^2$ and $m_{H_d}^2$ are

$$16\pi^2 \beta_{m_{H_u}^2} \equiv 16\pi^2 \frac{dm_{H_u}^2}{dt} = 6|y_t|^2(m_{H_u}^2 + m_{Q_3}^2 + m_{u3}^2) + ... \tag{2.56}$$

$$16\pi^2 \beta_{m_{H_d}^2} \equiv 16\pi^2 \frac{dm_{H_d}^2}{dt} = 6|y_b|^2(m_{H_d}^2 + m_{Q_3}^2 + m_{d3}^2) + ..., \tag{2.57}$$

where y_t and y_b are the top and bottom quark Yukawa couplings, respectively, and $m_{Q_3} = m_{33}^Q$, $m_{u3} = m_{33}^u$ and $m_{d3} = m_{33}^d$. Since $y_t \gg y_b$, $m_{H_u}^2$ runs much faster than $m_{H_d}^2$ as they approach the electroweak scale. This is called the *radiative electroweak symmetry breaking.*

The vector boson masses are known from experiment, and provide constraints on Higgs vevs $v_u = \langle H_u^0 \rangle$ and $v_d = \langle H_d^0 \rangle$

$$v_u^2 + v_d^2 \equiv v^2 = \frac{2m_Z^2}{g^2 + g'^2} \approx (174 \text{ GeV})^2. \tag{2.58}$$

The vevs therefore provide a single free parameter, which can be expressed as

$$\tan\beta \equiv \frac{v_u}{v_d}. \tag{2.59}$$

The parameters b and $|\mu|$ can be eliminated as free parameters of the model, but the sign of mu, sgnμ, cannot.

2.4.5 Sparticles

Gluinos

The gluino is the superpartner of the gluon, which is the boson responsible for the strong interaction. At tree level the gluino does not mix with anything in the MSSM and the mass is the soft term M_3, but with loop contributions such as those in Fig. (2.2) the mass runs quickly with energy μ. The gluino mass with

[2] The RGE describes how parameters change as a function of energy scale.

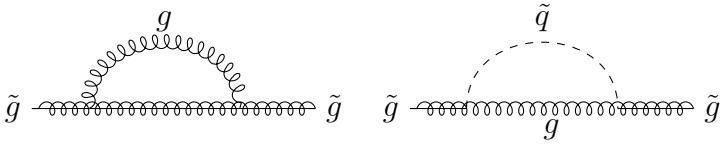


Figure 2.2: One loop contributions to the gluino mass.

one loop contributions in the $\bar{DR}$ scheme is

$$m_{\tilde{g}} = M_3(\mu)\left[1 + \frac{\alpha_s}{4\pi}\left(15 + 6\ln\frac{\mu}{M_3} + \sum_{\text{all } \tilde{q}} A_{\tilde{q}}\right)\right], \tag{2.60}$$

where the squark contributions are

$$A_{\tilde{q}} = \int_0^1 dx\ x\ln\left(x\frac{m_{\tilde{q}}^2}{M_3^2} + (1-x)\frac{m_q^2}{M_3^2}x(1-x) - i\epsilon\right). \tag{2.61}$$

Squarks

In supersymmetry every fermion f gets two supersymmetric partners f_L and f_R, which are scalar partners of the corresponding left-handed and right-handed fermion. So there are two squarks $\tilde{q}_L$, $\tilde{q}_R$ per quark q, and in the MSSM several terms contribute to their masses. For the first two generations, which are the ones relevant to this thesis, the main contributions come from the soft terms and the scalar potential. The contributions from soft terms assume that the soft masses are close to diagonal, and provide contributions $-m_Q^2\tilde{Q}_i^\dagger\tilde{Q}_i$ and $-m_q^2\tilde{q}_{iR}^*\tilde{q}_{iR}$ for an $SU(2)_L$ doublet $\tilde{Q}_i$ and singlet $\tilde{q}_i$ with index i, respectively.

The scalar potential contributes with hyperfine terms, that come from the d-terms $\frac{1}{2}\sum_2 g_a^2(A^*T^aA)^2$. This becomes of the form (sfermion)^2(Higgs)^2 when one of the scalar fields A is a Higgs field. These become mass terms when the Higgs develope vacuum expectation values v

$$\Delta_Q = (T_{3F}g^2 - Y_F g'^2)(v_d^2 - v_u^2) = (T_{3F} - Q_F\sin^2\theta_W)\cos 2\beta\ m_Z^2, \tag{2.62}$$

where the isospin T_3, hypercharge Y, and electric charge Q are the charges of the left-handed supermultiplet to which the squark belongs. These terms are the same for other sfermions.

The mass terms of the squarks are then *e.g.*

$$m_{\tilde{u}_L} = m_{Q_1}^2 + \Delta\tilde{u}_L, \tag{2.63}$$
$$m_{\tilde{d}_L} = m_{Q_1}^2 + \Delta\tilde{d}_L, \tag{2.64}$$
$$m_{\tilde{u}_R} = m_{u_1}^2 + \Delta\tilde{u}_R, \tag{2.65}$$
$$\tag{2.66}$$

with the mass splittings between the same generations

$$m^2_{\tilde{d}_L} - m^2_{\tilde{u}_L} = -\frac{1}{2}g^2(v_d - v_u) = -\cos 2\beta \; m^2_W. \tag{2.67}$$

Neutralinos and Charginos

Because the electroweak symmetry is broken, the gauge fields are now free to mix. The only requirement is that fields of the same $U(1)_{em}$ charge mix. This gives fields like the photino and zino, which are supersymmetric partners to the photon and Z boson. These are mixes of the neutral $\tilde{B}^0$ and $\tilde{W}^0$. However, the gauge fields are also free to mix with the Higgsinos, the fermions in the Higgs superfields, giving particles known as *neutralinos*. There are four neutralinos,

$$\tilde{\chi}^0_i = N_{i1}\tilde{B}^0 + N_{i2}\tilde{W}^0 + N_{i3}\tilde{H}^0_d + N_{i4}\tilde{H}^0_u, \tag{2.68}$$

where N_{ij} indicates how much of each component field is mixed in the neutralino. There are also charged particles, known as *charginos*, which are similar to the neutralinos but mixes $\tilde{W}^+$, $\tilde{H}^+_u$, $\tilde{W}^-$ and $\tilde{H}^-_d$.

2.4.6 MSSM-24

As mentioned in Sec. 1.4.3 the MSSM introduces 105 new parameters, where 104 come from soft terms and 1 comes from the scalar superpotential. However, experimental results can put restrictions on some parameters at high energy scales. One of the models with a restricted number of parameters is MSSM-24, where the number of parameters is reduced to 24. This is the model used to generate data in this project.

Off-diagonal terms in the slepton and squark mass matrices $(m^2_f)_{ij}$ could induce flavour-changing processes, such as $\mu \to e\gamma$. Since these processes have not yet been observed experimentally the squark and lepton mass matrices are assumed to be diagonal,

$$(m^2_f)_{ij} = \text{diag}(m^2_{\tilde{f}1}, m^2_{\tilde{f}2}, m^2_{\tilde{f}3}), \;\; f = u, d, e, Q, L. \tag{2.69}$$

Another restriction comes from CP-violation. To avoid inducing large CP-violating phases, the gaugino masses and three-scalar couplings are assumed to be real

$$\text{Im}(M_1) = \text{Im}(M_2) = \text{Im}(M_3) = \text{Im}(A^u_0) = \text{Im}(A^d_0) = \text{Im}(A^e_0) = 0. \tag{2.70}$$

Finally, as there is a one-to-one correspondence between the three-scalar terms the SM y^f_{ij} and in the MSSM a^f_{ij}, these are taken to be related through proportionality constants

$$a^u_{ij} = A^u_0 y^u_{ij}, \;\; a^d_{ij} = A^d_0 y^d_{ij}, \;\; a^e_{ij} = A^e_0 y^e_{ij}. \tag{2.71}$$

The parameters of the MSSM-24 are then the following

$M_1, M_2, M_3,$	Gaugino mass parameters,
A_0^u, A_0^d, A_0^e	Trinilear couplings,
$\tan\beta, m_{H_u}^2, m_{H_d}^2, \text{sgn } \mu$	Higgs parameters,
$m_{\tilde{Q}_1}^2, m_{\tilde{Q}_2}^2, m_{\tilde{Q}_3}^2$	Squark mass parameters,
$m_{\tilde{u}_1}^2, m_{\tilde{u}_2}^2, m_{\tilde{u}_3}^2$	
$m_{\tilde{d}_1}^2, m_{\tilde{d}_2}^2, m_{\tilde{d}_3}^2,$	
$m_{\tilde{L}_1}^2, m_{\tilde{L}_2}^2, m_{\tilde{L}_3}^2,$	Slepton mass parameters
$m_{\tilde{e}_1}^2, m_{\tilde{e}_2}^2, m_{\tilde{e}_3}^2.$	

Chapter 3

Supersymmetry at Hadron Colliders

The Large Hadron Collider (LHC) at CERN is one of the largest and most important particle physics experiments in the world. In this chapter, some of the advantages and challenges of using hadron colliders are discussed, along with some techniques for moving from theory to observable signals. A short description of supersymmetric phenomenology follows, along with current bounds on some supersymmetric particles. Finally, the squark production cross section is calculated to leading order, and next-to-leading order terms are investigated.

3.1 Hadron Colliders

Colliding hadrons makes it possible to reach very high center-of-mass energies, as they allow for the use of circular accelerators. While most linear colliders collide leptons, such as e^-e^+, they are not advantageous for circular accelerators because of *synchrotron radiation*. Synchrotron radiation is the radiation of energy from a particle being accelerated. The power radiated by a relativistic charged particle forced to move in circular motion with radius R is given by the Schwinger's formula [4]

$$P_e = \frac{2}{3}\frac{e^2 c}{R^2}\left(\frac{E}{mc^2}\right)^4, \tag{3.1}$$

where E is the particle energy, m is the mass and c is the speed of light in vacuum. For light particles, such as leptons, a lot of energy is therefore wasted in circular accelerators. Because the proton mass is much larger than the electron mass, this effect is relatively small, allowing the LHC to operate at energies currently as high as 13 TeV. In this project the data is generated at 8 TeV. The circular form means the accelerating structures can be reused as many times

as one desires, thus putting 'no limits' on the energies obtained. There are, of course, limits. At around energies of around $5 - 7$ TeV per particle, synchrotron radiation becomes an important effect also for protons. The photons emitted from synchrotron radiation hit the walls of the vacuum chamber walls, where they can interact with electrons and cause *electron clouds*. Electron clouds occur when electrons are ejected from the walls of the vacuum chamber and accelerated towards a passing beam bunch. When the electrons reach the center of the chamber, however, the beam bunch has passed, and the now-energetic electrons hit the other side of the chamber, producing more free electrons. This becomes a 'cloud' of electrons that can in turn affect the particle beam.

A challenge more spesific to hadron collisions is the distribution of momentum. Hadrons are made up of valence and sea quarks, which make the kinematics of collisions very difficult to calculate. Valence quarks are the quarks used to classify a hadron, such as *uud* for the proton and *udd* for the neutron. In addition to these, hadrons contain a sea of virtual quarks and gluons. The quarks and gluons, the so-called *partons*, distribute the hadron momentum somewhat randomly amongst themselves. Since the distribution of energy and momentum is unknown, so is the momenta of the ingoing parton. To that end the transverse momentum and energy, not to mention the *missing* transverse energy, are important when analysing data from hadron colliders. There is some experimental knowledge of the distribution of longitudinal momentum, however, contained in *parton distribution functions*.

3.1.1 Parton Distribution Functions

Partonic cross sections are calculated for colliding partons, *e.g.* two quarks $q_1 q_2$. The cross section is a function of the center-of-mass energy, s, which can be written as a fraction of the center of mass energy of the colliding protons

$$s = S x_1 x_2, \tag{3.2}$$

where S is the center-of-mass energy of the colliding protons, and x_i is the momentum fraction of the quark q_i. The fractions of momenta are then integrated over, using *parton distribution functions* $f(x_i)$, which are specified for the different partons in different hadrons. For example, the fraction x_u for an up-flavour quark in a proton would be much larger than that for a top-flavour quark. Integrating over parton distribution functions yields the total cross section

$$\sigma_{q_1 q_2} = \int f(x_1) f(x_2) \hat{\sigma}_{q_1 q_2}(s) dx_1 dx_2, \tag{3.3}$$

where $\hat{\sigma}_{q_1 q_2}$ is the partonic cross section. In this project the CTEQ6 parton distribution functions from the LHAPDF Fortran library [5] are used.

3.1.2 Luminosity

Another important concept is *luminosity*. The instantaneous luminosity $\mathcal{L}$ is a measure on how many collisions happen at a collider per unit time. The integrated luminosity is the luminosity integrated with respect to time, and is here given in inverse femto-barn $\text{fb}^{-1} = 10^{43}\ \text{m}^{-2}$. The luminosity of the data can be used to set limits on the size of cross sections, by considering the following relation for a process where a particle A is produced

$$n_A = \mathcal{L}\sigma_A, \tag{3.4}$$

where $\mathcal{L}$ is the integrated luminosity, σ_A is the cross section for the production of A, and n_A is the number of produced particles. Setting $n = 1$ for a single produced particle, and using the integrated luminosity for the 8 TeV dataset considered in this project $\mathcal{L} = 20.3\ \text{fb}^{-1}$, the lower limit on cross sections is

$$\sigma = \frac{1}{20.3\ \text{fb}^{-1}} \approx 0.05\ \text{fb}. \tag{3.5}$$

Therefore, cross sections below $\sigma \sim \mathcal{O}(10^{-3}\ \text{fb}^{-1})$, which correspond to 0.02 produced particles, will be considered less important in this project.

3.2 Phenomenology

Phenomenology deals with the application of theory to high energy experiments, such as the Large Hadron Collider described in Sec. 1.1. This section deals with searches for supersymmetry at hadron colliders, using missing transverse energy and quark jets. Some current bounds on sparticles are discussed as well.

3.2.1 Searches For Supersymmetry

Supersymmetry at hadron colliders will likely be in the form of QCD processes, as the colliding particles are quarks and gluons. The traces left in detectors by such processes will be closely related to the conservation of R-parity, in that all sparticles are produced in pairs and eventually decay to the LSP. If the LSP is indeed only weakly interacting, this makes it very difficult to detect. An indirect way of detecting it is by looking for *missing transverse energy* $\not{E}_T$. The energy is considered solely in the transverse plane, as the longitudinal momentum is difficult to predict in the hadronic case (see the discussion of hadron colliders in Sec. 1.1). To look for $\not{E}_T$ the *effective mass* is defined as

$$M_{eff} = \sum p_T^{jet} + \not{E}_T, \tag{3.6}$$

and used to search for deviations from SM expectations. Here p_T^{jet} is the transverse momentum of *jets*. Jets are collimated bunches of final-state partons and

hadrons that appear in hard interactions [1]. They can be thought of as energetic partons that undergo showering and the hadronization. Jets are common traces in hadron collision detectors. Because of asymptotic freedom[2] partons do not remain unbound for long and form the jet hadrons. In supersymmetric processes from hadron collisions the production of jets should also be common, as the LSP is flavour neutral and the color charge of gluinos and squarks must end up in SM particles with color charge. The hadronic jets produced from supersymmetric processes can be complicated, such as the squark-squark production shown in Fig. 3.1. The very large background from SM processes provides another difficulty in searching for supersymmetry.

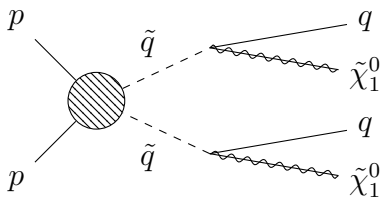


Figure 3.1: Possible signature of a supersymmetric QCD process, with two quark jets and large missing transverse energy in the final state.

Allowing for R-parity violation opens up for more final-state possibilities. The LSP can then decay, and sparticles can be produced one at a time. It is possible to have *massive metastable charged particles* (MMCPs), which are typical for scenarios with a gravitino LSP [1].

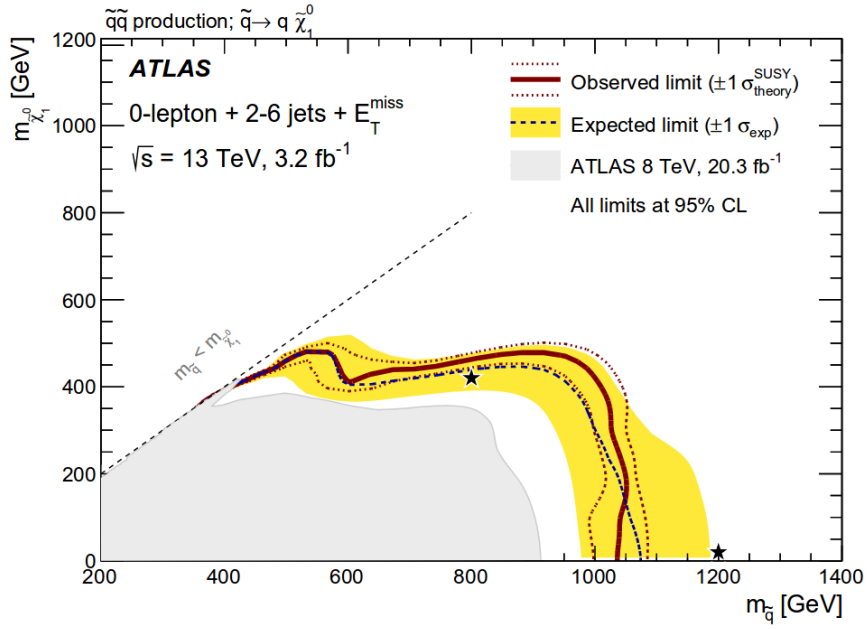
3.2.2 Current Bounds on Sparticles

Searches and limits are available from the data recorded in 2015 by the ATLAS experiment in $\sqrt{s} = 13$ TeV proton-proton collisions at the LHC, with 2.3 fb^{-1} of analyzed data. The analysis searched for jets and missing transverse energy. Simplified models were assumed, with R-parity conservation and the lighest neutralino as the lightest supersymmetric particle. At 95% confidence level, exclusion limits of the gluino and squark masses are set at 1.51 TeV and 1.03 TeV, respectively [6], assuming a *massless lightest neutralino*. A plot of exclusion limits is shown in Fig. 3.2. Note that for a massive lightest neutralino the bounds on the gluino and squark masses are significantly reduced, down to 400 GeV for the squark mass and around 650 GeV for the gluino mass. Values below these mean that the lightest neutralino is no longer the lightest particle $m_{\tilde{q}} < m_{\tilde{\chi}_1^0}$, $m_{\tilde{g}} < m_{\tilde{\chi}_1^0}$, and sparticles must decay to another particle. In this project squark

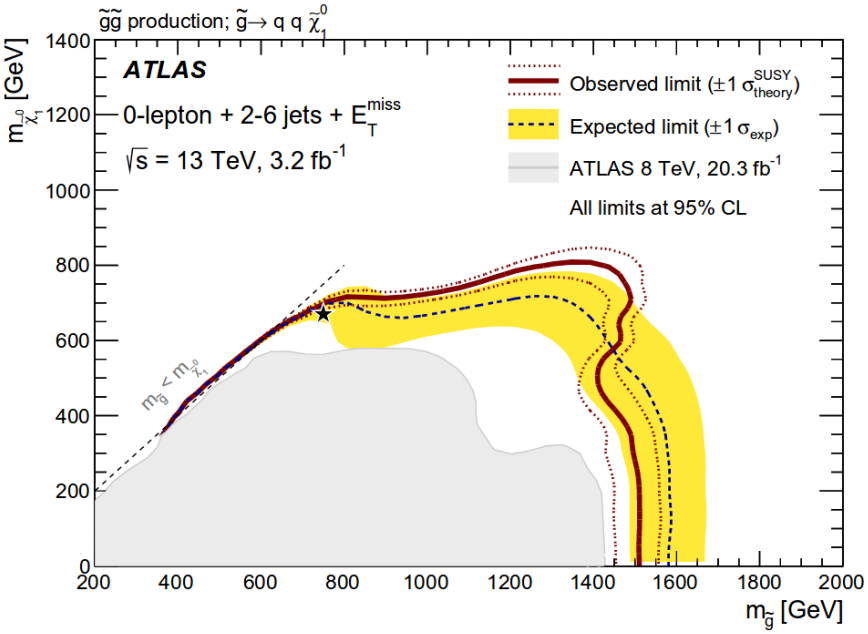
[1] Hard interactions are where the ingoing particles are very energetic

[2] The strong coupling constant becomes larger as the distance between partons increases, meaning that partons are *confined* in hadrons.

and gluino masses of $0 - 4000$ GeV are investigated, as the neutralino is not assumed to be massless.



(a) Squark mass



(b) Gluino mass

Figure 3.2: Exclusion limits from the ATLAS experiment in $\sqrt{s} = 13$ TeV proton-proton collisions. The analysis assumes conservation of R-parity and a lightest neutralino LSP. The exclusion limits also assume a massless lightest neutralino. The dashed lines indicate the limit where $m_{\tilde{q}/\tilde{g}} < m_{\tilde{\chi}_1^0}$, so the squarks and gluinos cannot decay to the lightest neutralino above this limit. Figures from [6].

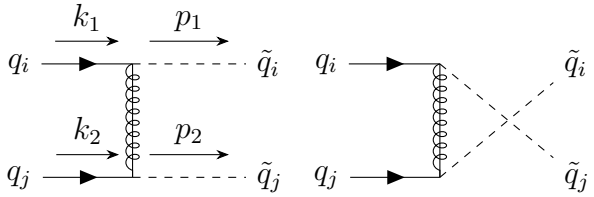


Figure 3.3: Feynman diagrams for squark pair production in quark quark collisions, both t and u diagram. Note that the u-channel (right diagram) is only possible for $i = j$.

3.3 Squark-Squark Cross Section

In this project the relevant QCD process will be the production of squark pairs in quark-quark collisions,

$$q_i q_j \to \tilde{q}_i \tilde{q}_j, \tag{3.7}$$

where i, j are the 4 light quark flavours u, d, s, c. Feynman diagrams for tree-level contributions are found in Fig. 3.3. For equal flavour quarks the t- and u-channel contribute, while only the t-channel contributes for different flavours.

3.3.1 Leading Order Cross Section

In this section the partonic cross section for squark pair-production is calculated to leading order. The exchanged gluino momentum is denoted p in the calculations, and defined as $p = k_2 - p_2$ for the t-channel and $p = k_2 - p_1$ for the u-channel. The following set of kinematical invariants are used

$$\begin{aligned}
s &= (k_1 + k_2)^2 = 2k_1 \cdot k_2, & t_1 &= (k_2 - p_2)^2 - m_{\tilde{q}}^2, \quad t_g = (k_2 - p_2)^2 - m_{\tilde{g}}^2, \\
t &= (k_2 - p_2)^2 = m_{\tilde{q}}^2 - 2(k_2 \cdot p_2), & u_1 &= (k_1 - p_2)^2 - m_{\tilde{q}}^2, \quad u_g = (k_1 - p_2)^2 - m_{\tilde{g}}^2, \\
u &= (k_1 - p_2)^2 = m_{\tilde{q}}^2 - 2(k_1 \cdot p_2),
\end{aligned}$$

where the Mandelstam variables are related by $t+u+s = p_1^2+p_2^2$. The expressions for the different chiralitites are considered later, until then the chiral projection operators in the matrix element are denoted as P and P'. The Feynman gauge is used, and the $n_f = 5$ light flavour quarks are treated as massless.

Pure t-channel

The matrix element for the pure t-channel becomes (reading direction is from q_j to q_i)

$$\begin{aligned} i\mathcal{M}_t &= \bar{v}(k_1)\Big(-i\sqrt{2}gP(t_a)^{ij}\Big) \times \delta^{ab}\frac{i}{\not{p} - m_{\tilde{g}}} \times \Big(-i\sqrt{2}gP'(t_b)^{lk}\Big) \times u(k_2) \\ &= -(t_a)^{ij}(t^a)^{lk} \times \frac{i2g^2}{t_g^2} \times \bar{v}(k_1)P(\not{p} + m_{\tilde{g}})P'u(k_2), \end{aligned}$$

where the color factor has been factored out. The matrix element squared is then

$$|\mathcal{M}_t|^2 = (t_a)^{ij}(t^a)^{lk}(t_b)_{ij}(t^b)_{lk} \times \frac{4g^4}{t_g^2}\big(\bar{v}(k_1)P(\not{p} + m_{\tilde{g}})P'u(k_2)\big)\big(\bar{u}(k_2)P(\not{p} + m_{\tilde{g}})P'v(k_1)\big).$$

To sum over colors, the following relation is used [7]

$$\sum_a (t^a)_{ij}(t_a)_{lk} = \frac{1}{2}(\delta_{ik}\delta_{lj} - \frac{1}{N}\delta_{ij}\delta_{lk}), \tag{3.8}$$

which gives for the color factor

$$(t^a)^{ij}(t_a)^{kl}(t^b)_{ij}(t_b)_{kl} = \frac{1}{4}\big(\delta_{ik}\delta_{lj} - \frac{1}{N}\delta_{ij}\delta_{lk}\big)\big(\delta^{ik}\delta^{lj} - \frac{1}{N}\delta^{ij}\delta^{lk}\big) = \frac{1}{4}(N^2-1) = \frac{1}{2}NC_F,$$

where $C_F = (N^2-1)/(2N)$. Averaging over spin

$$\sum |\mathcal{M}_t|^2 = \frac{1}{2}NC_F \times \frac{4g^4}{t_g^2}\mathrm{tr}\big[\not{k}_1 P(\not{p} + m_{\tilde{g}})P'\not{k}_2 P(\not{p} + m_{\tilde{g}})P'\big].$$

As previously mentioned, the quark masses are set to zero. The final state squarks can have equal or different chiralities, both of which contribute to the total cross section.

Different chiralities $P = P_{R/L}$, $P' = P_{L/R}$

In the case of different chiralities the trace becomes

$$\begin{aligned} \mathrm{tr}\big[\not{k}_1 P_{R/L}(\not{p} + m_{\tilde{g}})P_{L/R}\not{k}_2 P_{R/L}(\not{p} + m_{\tilde{g}})P_{L/R}\big] &= 2\big(2(p\cdot k_2)(k_1\cdot p) - p^2(k_1\cdot k_2)\big) \\ &= (t - m_{\tilde{q}}^2)s + (t - m_{\tilde{q}}^2)(u - m_{\tilde{q}}^2) - ts \\ &= t_1 u_1 - m_{\tilde{q}}^2 s. \end{aligned}$$

For same flavour ingoing quarks qq, there is one possibility for different chiralities $\tilde{q}_R\tilde{q}_L$. For different flavours $q'q$ there are two possibilities; $\tilde{q}_R\tilde{q}'_L$ and $\tilde{q}_L\tilde{q}'_R$.

Equal chiralities $P = P_{R/L}$, $P' = P_{R/L}$

In the case of equal chiralities the trace becomes

$$\mathrm{tr}\big[\not{k}_1 P_{R/L}(\not{p} + m_{\tilde{g}})P_{R/L}\not{k}_2 P_{R/L}(\not{p} + m_{\tilde{g}})P_{R/L}\big] = 2m_{\tilde{g}}^2(k_1 \cdot k_2) = m_{\tilde{g}}^2 s,$$

where $P_{R/L}P_{R/L} = P_{R/L}$, $(\gamma^5)^2 = 1$ and $\mathrm{tr}[\gamma^5\gamma^\mu\gamma^\nu] = 0$. For equal flavour quarks qq there are now two possibilities for equal chiralities: $\tilde{q}_R\tilde{q}_R$ and $\tilde{q}_L\tilde{q}_L$. For different flavour quarks qq' there are two possibilities for equal chiralities: $\tilde{q}_L\tilde{q}'_L$ and $\tilde{q}_R\tilde{q}'_R$.

Note that the contribution from $\tilde{q}_R\tilde{q}_R$ is identical to $\tilde{q}_L\tilde{q}_L$ in QCD processes. If no electroweak corrections are included in the higher order term, the NLO cross section should also be identical as a function of $m_{\tilde{q}}$, because only the electroweak interaction couples to right- and left-handed states differently.

Sum over Chiralities

For incoming quarks $q_i q_j$ the sum over chiralities yields

$$\sum |\mathcal{M}_t|^2 = NC_F \times \frac{4g^4}{t_g^2}\Big[\delta_{ij}\big(\frac{1}{2}(t_1u_1 - sm_{\tilde{q}}^2) + sm_{\tilde{g}}^2\big) + (1-\delta_{ij})\big(t_1u_1 - s(m_{\tilde{q}}^2 - m_{\tilde{g}}^2)\big)\Big].$$

Pure u-channel

The expression for the u-channel diagram is identical to the t-channel, but for the exchange $t_g^2 \to u_g^2$. The u-channel only contributes in the case where $i = j$, so the matrix element is

$$\sum |\mathcal{M}_u|^2 = \delta_{ij} NC_F \frac{4g^4}{u_g^2}\Big[\frac{1}{2}(t_1u_1 - sm_{\tilde{q}}^2) + sm_{\tilde{g}}^2\Big].$$

Cross term

The ut cross term has the matrix element $\mathcal{M}_{tu} + \mathcal{M}_{ut}$, where $\mathcal{M}_{tu}$ is given by

$$i\mathcal{M}_{tu} = (t^a)^{ij}(t_a)^{kl}(t^b)_{ik}(t_b)_{jl}\frac{-i2g^2}{t_g}\big(\bar{v}(k_1)P(\not{p} + m_{\tilde{g}})P'u(k_2)\big)\frac{i2g^2}{u_g}\big(\bar{u}(k_2)P(\not{p} + m_{\tilde{g}})P'v(k_1)\big)$$

The color factor is

$$\sum_{a,b}(t^a)^{ij}(t_a)^{kl}(t^b)_{ik}(t_b)_{jl} = -\frac{1}{2}C_F.$$

Average over spins

$$\sum \mathcal{M}_{tu} = -\frac{1}{2} C_F \times \mathrm{tr}\left[\frac{4g^4}{u_g t_g}\left(\not{k}_1 P(\not{k}_2 - \not{p}_2 + m_{\tilde{g}})P'\not{k}_2 P(\not{k}_2 - \not{p}_1 + m_{\tilde{g}})P'\right)\right].$$

Summing over equal and different chiralities gives

$$\sum \mathcal{M}_{tu} = -\frac{1}{2} C_F \frac{4g^4}{u_g t_g}\left(-t_1 u_1 + m_{\tilde{q}}^2 s + m_{\tilde{g}}^2 s\right)$$

The other term – $\mathcal{M}_{ut}$ – is similar, but yields a slightly different combination

$$\sum i\mathcal{M}_{ut} = -\frac{1}{2} C_F \frac{4g^4}{u_g t_g}\left(u_1 t_1 - m_{\tilde{q}}^2 s + m_{\tilde{g}}^2 s\right).$$

Adding the cross terms then gives

$$\sum |\mathcal{M}_{ut+tu}| = -\frac{1}{2} C_F \delta_{ij} \frac{4g^4}{u_g t_g}\left(2m_{\tilde{g}}^2 s\right).$$

Matrix Elements

The matrix elements can be divided into contributions from $\tilde{q}_{iR}\tilde{q}_{iR}$, $\tilde{q}_{iR}\tilde{q}_{iL}$, $\tilde{q}_{iR}\tilde{q}_{jR}$ and $\tilde{q}_{iR}\tilde{q}_{jL}$, and equivalently for $R \leftrightarrow L$. These are

$$\tilde{q}_{iR}\tilde{q}_{iR}, \tilde{q}_{iL}\tilde{q}_{iL}: \quad \sum |\mathcal{M}|_{iRiR} = 4g^4 s m_{\tilde{g}}^2\left[\frac{1}{2} N C_F\left(\frac{1}{t_g^2} + \frac{1}{u_g^2}\right) - C_F \frac{1}{t_g u_g}\right], \tag{3.9}$$

$$\tilde{q}_{iR}\tilde{q}_{iL}: \quad \sum |\mathcal{M}|_{iRiL} = 4g^4 (u_1 t_1 - s m_{\tilde{q}}^2)\left[\frac{1}{2} N C_F\left(\frac{1}{t_g^2} + \frac{1}{u_g^2}\right)\right], \tag{3.10}$$

$$\tilde{q}_{iR}\tilde{q}_{jR}, \tilde{q}_{iL}\tilde{q}_{jL}: \quad \sum |\mathcal{M}|_{iRjR} = 4g^4 s m_{\tilde{g}}^2\left[\frac{1}{2} N C_F \frac{1}{t_g^2}\right], \tag{3.11}$$

$$\tilde{q}_{iR}\tilde{q}_{jL}, \tilde{q}_{iL}\tilde{q}_{jR}: \quad \sum |\mathcal{M}|_{iRjL} = 4g^4 (t_1 u_1 - s m_{\tilde{q}}^2)\left[\frac{1}{2} N C_F \frac{1}{t_g^2}\right], \tag{3.12}$$

$$\tag{3.13}$$

Summing over the terms from different chirality combinations gives for the total sum over matrix element squared

$$\begin{aligned}\sum |\mathcal{M}|^2 = \delta_{ij}\Bigg[&2g^4 N C_F (u_1 t_1 - s m_{\tilde{q}}^2)\left(\frac{1}{t_g^2} + \frac{1}{u_g^2}\right) \\ &+ 4g^4 s m_{\tilde{g}}^2\left(N C_F\left(\frac{1}{t_g^2} + \frac{1}{u_g^2}\right) - 2C_F \frac{1}{u_g t_g}\right)\Bigg] \\ &+ (1 - \delta_{ij})\left[4g^4 N C_F \frac{u_1 t_1 - s(m_{\tilde{g}}^2 - m_{\tilde{g}}^2)}{t_g^2}\right].\end{aligned} \tag{3.14}$$

Partonic Cross Section

The $SU(3)$ color factors are given by $N = 3$, and therefore $C_F = 4/3$. To find the total, lowest order partonic cross section an n-dimensional phase space integral is performed, and spin and color averaging are taken into account [8]. The lowest order double-differential distributions are then given by

$$\begin{aligned} s^2\frac{d^2\sigma^B}{dtdu} =& K_{ij}\frac{\pi S_\varepsilon}{\Gamma(1-\varepsilon)}\left[\frac{(t-p_2^2)(u-p_2^2)-p_2^2 s}{\mu^2 s}\right]^{-\varepsilon}\Theta([t-p_2^2][u-p_2^2]-p_2^2 s) \\ &\times\Theta(s-4m^2)\delta(s+t+u-p_1^2-p_2^2)\sum|\mathcal{M}_B|^2, \end{aligned} \tag{3.15}$$

where the averaging over initial state colours and spins is given by the factor K_{ij}, which for squark and antisquark prodution is [8]

$$K_{qq} = K_{\bar{q}q} = \frac{1}{4N^2}. \tag{3.16}$$

Integration over the remaining parameters gives for squark production $q_i q_j \to \tilde{q}_i\tilde{q}_j$ [8]

$$\begin{aligned} \hat{\sigma}^B =& \frac{\pi\alpha_s^2}{s}\left[\beta_{\tilde{q}}\Big(-\frac{4}{9}-\frac{4m_-^4}{9(m_{\tilde{g}}^2 s+m_-^4)}\Big)+\Big(-\frac{4}{9}-\frac{8m_-^2}{9s}\Big)L_1\right] \\ &+\delta_{ij}\frac{\pi\alpha_s^2}{s}\left[\frac{8m_{\tilde{g}}^2}{27(s+2m_-^2)}L_1\right], \end{aligned} \tag{3.17}$$

where

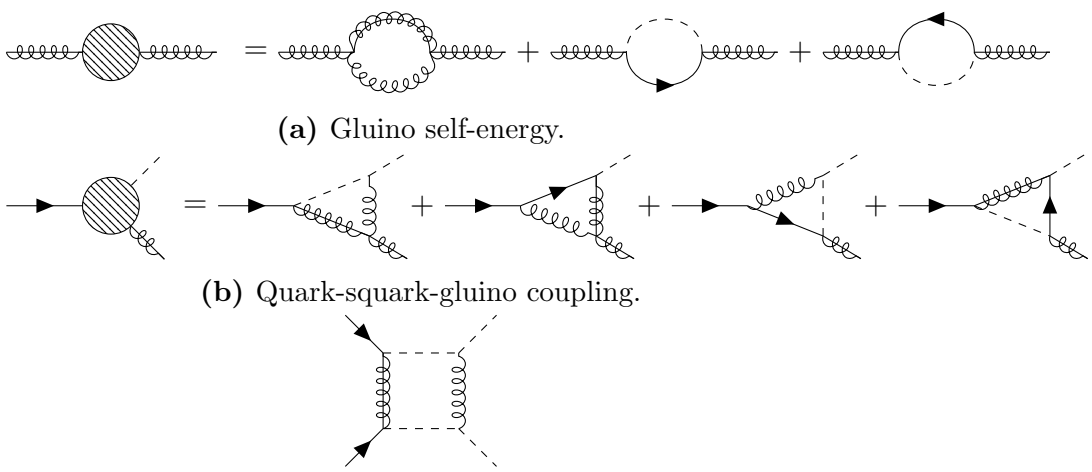
$$L_1 = \ln\Big(\frac{s+2m_-^2-s\beta_{\tilde{q}}}{s+2m_-^2+s\beta_{\tilde{q}}}\Big), \quad \beta_{\tilde{q}} = \sqrt{1-\frac{4m_{\tilde{q}}^2}{s}}, \quad m_-^2 = m_{\tilde{g}}^2 - m_{\tilde{q}}^2, \quad \alpha_s = \frac{g_s^2}{4\pi}.$$

The cross section for antisquark pair production is similar.

3.4 Next-to-leading Order Corrections

The next-to-leading order (NLO) terms contain virtual corrections to the Born level, or tree-level, Feynman diagrams. By allowing more complicated Feynman diagrams, virtual particles can appear in the process. These never make it to the final state and become real particles, but are instead absorbed by the process. Since they are never real, or on-shell, they can have any momentum and mass. Thus, it is necessary to integrate over all possible momenta, which leads to divergences and infinities.

There are three main categories of divergences, namely ultraviolet, infrared and collinear. Ultraviolet divergences appear when the integral is over an energy



(a) Gluino self-energy.

(b) Quark-squark-gluino coupling.

(c) Box diagram.

Figure 3.4: A selected set of Feynman diagrams for the virtual corrections to $q_i q_j \to \tilde{q}_i \tilde{q}_j$ [8].

without upper bounds. When the opposite problem occurs, that low momentum gives infinities, this is called an infrared divergence. Collinear divergences are also called mass singularities, and stem from exactly that — masses going to zero, where infinities can appear from splitting at zero angle.

In order to include the corrections and still get physical cross sections, it is necessary to integrate out the divergences using dimensional regularization and renormalise parameters. Renormalising a parameter means giving it a scale dependence, and baking the infinities into this expression. Physically, this can be interpreted as giving the coupling constant a length scale (energy scale^{-1}) dependence. For example, the electromagnetic coupling becomes stronger the closer to a charged particle you get. These calulations are often very complicated and tedious. Some of the diagrams for virtual corrections to $q_i q_j \to \tilde{q}_i \tilde{q}_j$ are shown in Fig. 3.4.

There are several reasons for calculating these cumbersome NLO terms. Firstly, the leading order terms are highly dependent on the *a priori* unknown renormalization scale, leading to a large uncertainty in theoretical predictions (up to a factor of two). Adding higher order terms reduces this scale dependency. An example is shown in Fig. 3.5, where the LO and NLO cross sections for $qq \to \tilde{q}\tilde{q}$ are plotted as a function of the renormalization/factorization scale [8]. The NLO terms are clearly less dependent on the scale. Secondly, the NLO contributions are expected to be large and positive, thereby raising the cross sections significantly and allowing for stronger bounds on the gluino and squark masses [8].

The calculations from [8] assume degenerate squark masses $m_{\tilde{q}}$, set the 5 lightest quark masses to zero as they are much lighter than the squarks, and the top quark mass to $m_t = 175$ GeV. The two free parameters left are then

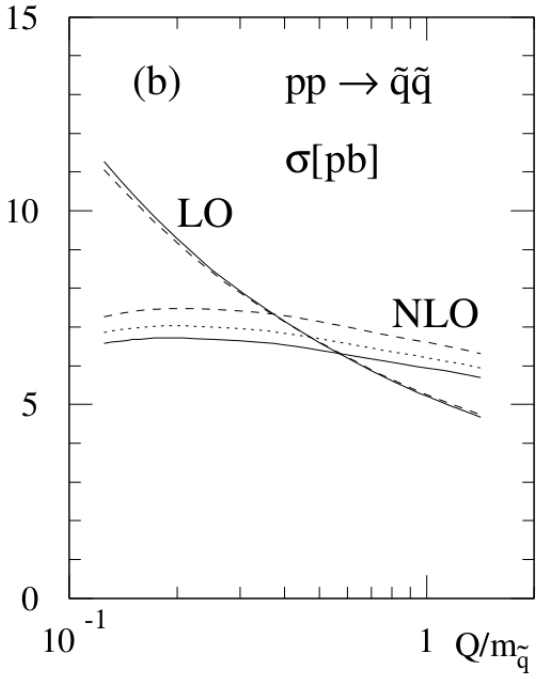


Figure 3.5: The dependence on the renormalization/factorization scale Q for the LO and NLO cross sections for squark-squark production at the LHC ($\sqrt{s}$ = 14 TeV). Parton densities are GRV94 (solid), CTEQ3 (dashed) and MRS(A') (dotted). Mass parameters are $m_{\tilde{q}} = 600$ GeV, $m_{\tilde{g}} = 500$ GeV and $m_t = 175$ GeV. Figure from [8].

$m_{\tilde{g}}$ and $m_{\tilde{q}}$. This indicates that these might be good features for the learning. The renormalization scheme used is the $\bar{M}S$ scheme. Masses and couplings are renormalized, and the resulting parameters can be found in [8].

Next-to-leading Order Partonic Cross-Section

As previously discussed, calculations for hadron colliders require first finding the hadronic cross sections. In order to analyze these, scaling functions are introduced [8] giving the LO+NLO result as

$$\hat{\sigma}_{ij} = \frac{\alpha_s^2(Q^2)}{m^2}\Big\{ f_{ij}^B(\eta, r) + 4\pi\alpha_s(Q^2)\Bigg[f_{ij}^{V+S}(\eta, r, r_t) + f_{ij}^H(\eta, r) + \bar{f}_{ij}(\eta, r)\log\left(\frac{Q^2}{m^2}\right)\Bigg]\Big\}, \tag{3.18}$$

where Q^2 is the renormalization scale, often set to $Q^2 = m^2$, and $m = (\sqrt{p_1^2} + \sqrt{p_2^2})/2$ is the average mass of the produced particles. The scaling functions f are as follows: the Born term f^B from Eq. (3.17), the sum of virtual and soft-gluon corrections f^{V+S}, the hard gluon corrections f^H, and the scale-dependent contributions $\bar{f}$. The partonic cross section depends on the parameters

$$\eta = \frac{s}{4m^2} - 1, \qquad r = \frac{m_{\tilde{g}}^2}{m_{\tilde{q}}^2}, \qquad r_t = \frac{m_t^2}{m^2}. \tag{3.19}$$

The energy near the threshold is the base for an important part of the contributions to the cross section [8]. In this region the scaling functions can be expanded in the low velocity of produced particles β, leading to the following expressions [8]

$$\begin{aligned}
f_{qq}^B &= \frac{8\pi\beta m_{\tilde{q}}^2 m_{\tilde{g}}^2}{27(m_{\tilde{q}}^2 + m_{\tilde{g}}^2)^2}, & f_{q'q}^B &= \frac{8\pi\beta m_{\tilde{q}}^2 m_{\tilde{g}}^2}{9(m_{\tilde{q}}^2 + m_{\tilde{g}}^2)^2} \\
f_{qq}^{V+S} &= f_{qq}^B \frac{1}{24\beta} & f_{q'q}^{V+S} &= f_{q'q}^B \frac{1}{24\beta} \\
f_{qq}^H &= f_{qq}^B\Big[\frac{2}{3\pi^2}\log^2(8\beta^2) - \frac{7}{2\pi^2}\log(8\beta^2)\Big] & f_{q'q}^H &= f_{q'q}^B\Big[\frac{2}{3\pi^2}\log^2(8\beta^2) - \frac{19}{6\pi^2}\log(8\beta^2)\Big] \\
\bar{f}_{qq} &= -f_{qq}^B \frac{2}{3\pi^2}\log(8\beta^2) & \bar{f}_{q'q} &= -f_{q'q}^B \frac{2}{3\pi^2}\log(8\beta^2).
\end{aligned} \tag{3.20}$$

The main contributions to the partonic cross section come from this region.

Hadronic Cross-Section

As per the above discussion, the partonic cross-sections must be integrated over in order to obtain the total hadronic cross section. A convolution integral over

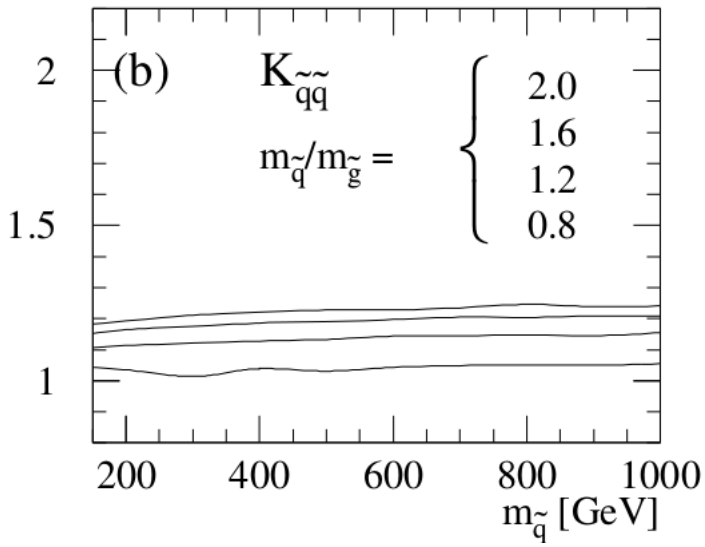


Figure 3.6: The K-factors for the LHC ($\sqrt{s} = 14$ TeV). Parton densities are GRV94, with scale $Q = m$, and the top squark mass is $m_t = 175$ GeV. Figure from [8].

the parton distribution functions yields the expression

$$\sigma(S, Q^2) = \sum_{i,j=q,q,\bar{q}} \int_{\tau}^{1} dx_1 \int_{\tau/x_1}^{1} dx_2 f_i^{h_1}(x_1, Q^2) f_j^{h_2}(x_2, Q^2) \hat{\sigma}_{ij}(x_1 x_2 S, Q^2)\Big|_{\tau=4m^2/S}. \tag{3.21}$$

The integrals are calculated numerically using the VEGAS integration routine [9] in [8]. The uncertainty due to different parameterizations of the parton densities for the LHC calculations of the NLO terms amount to $\lesssim 13\%$ at the central scale.

K-Factor

To quantify the change in the cross section found by adding NLO terms, the *K-factor* is introduced. The K-factor is the ratio between the cross sections

$$K = \sigma_{NLO}/\sigma_{LO}. \tag{3.22}$$

The K-factor for squark production for varying mass ratios $m_{\tilde{q}}/m_{\tilde{g}} = 2.0$, 1.6, 1.2, 0.8 at the LHC is shown in Fig. 3.6. As seen from the figure, the K-factor is larger than one and quite stable as a function of the squark mass.

3.4.1 State-of-the-art Tools

There are two numerical tools for the calculation of NLO SUSY cross sections, namely Prospino [10] and NLL-fast [11].

Prospino

`Prospino 2.1` [10] is a numerical tool that calculates supersymmetric cross sections to next-to-leading order for non-degenerate squark masses. It uses the K-factor to calculate NLO cross sections, by first computing LO cross sections for the 5 or 4 lightest squark flavors. The LO and NLO rates are then calculated for a mean value of the squark mass, and the corresponding K-factor is calculated. The LO cross sections for different squarks are then multiplied by the K-factor, giving an approximation to the NLO terms for non-degenerate masses. The calculations are outlined in the section above, and are described in more detail in [10].

`Prospino 2.1` is quite time-consuming, however. Evaluating the strong process cross sections for just a single CMSSM benchmark point takes about 15 minutes of CPU time on a modern processor [12].

In addition, as was discovered during the project, `Prospino 2.1` has a weakness in the heavy squark mass space. Consider the cross section for the production of $\tilde{c}_L\tilde{c}_L$. When all squark masses are heavy, but $m_{\tilde{c}_L}$ is slightly lighter than the others, the K-factor can be set to zero if the average mass is above threshold ($\tau > s$). This means that $LO \neq 0$ and $NLO = 0$, which gives problematic outlier points that cause trouble during learning.

NLL-fast

`NLL-fast 2.1` [8, 13, 14, 15, 16] computes the hadronic cross sections of gluino and squark pair production including NLO supersymmetric QCD corrections, and the resummation of soft gluon emission at next-to-leading-logarithmic (NLL) accuracy. It also provides an error estimation, based on the errors from renormalization scale dependence and the parton distribution functions. The calculation is done by reading in tables of LO and NLO+NLL results, the scale uncertainty and the PDF and α_s error, and uses fast interpolation to calculate cross sections for any squark and gluino mass within the range $[200, 2500]$ GeV for $\sqrt{s} = 8$ TeV. This does mean, however, that `NLL-fast 2.1` is limited in that it only calculates cross sections for mass-degenerate squarks. Since the true total cross section for squark pair production consists of 36 different cross sections for combinations of the 4 lightest quarks, the estimate from `NLL-fast 2.1` is very simplified and unfit for *e.g.* MSSM-24, as shown in a later section.

Accuracy

Including the NLO+NLL cross sections has reduced the theoretical error to 10% for a wide range of processes and masses, see the discussion in [12]. There are other uncertainties, however, such as the ones from the PDFs and α_s. These must also be included in the total error estimate. PDFs are based on experimental data,

and so the uncertainty increases with the sparticle masses because the PDFs are most poorly constrained at large scales and at large partons x.

To illustrate, consider both the gluino and (degenerate) squark mass to be 1.5 TeV, which is at the edge of what the LHC is able to produce at 8 TeV, as mentioned in the above section. `NLL-fast 2.1` gives errors of $(+24.3\%, -22.2\%)$ for the PDFs and $(+8.3\%, -7.3\%)$ for α_s, when using the MSTW2008 NLO PDF set [17]. In this case the total error will not be much lower than 25%. However, with new data from the LHC errors from PDFs and α_s will be reduced over time.

Thus, the relative regression errors obtained in this thesis should not exceed 10%, in order to keep the regression errors subdominent.

Chapter 4

Gaussian Processes

In this chapter Gaussian process regression is introduced. First, some concepts and terminology in Bayesian statistics are reviewed. The following section introduces the mathematical framework needed for Gaussian processes, and selected covariance functions are discussed. Concepts in Bayesian model selection are used as a basis to quantify and improve the quality of predictions. Finally, distributed Gaussian processes are introduced as a way of scaling Gaussian processes to larger datasets.

4.1 Introduction to Bayesian Statistics

There are two general philosophies in statistics, namely *Bayesian* and *frequentist* statistics. To understand where they differ, consider a statement statisticians from both branches would probably consider to be true

> *Statisticians use probability to describe uncertainty.*

The difference between Bayesian and frequentist statistics is at the definition of the *uncertain*. Since uncertainty is described by probability this understanding must also vary, and one distinguishes between *objective* and *subjective* probability. Consider an example in which a statistician throws a die. Before throwing, he is uncertain about the outcome of the toss. This uncertainty related to the outcome is *objective*: no one can know if he will throw a 1 or a 4. On the other hand, he might also be uncertain about the underlying probability distribution of the toss. Is the die loaded? Is one of the edges sharper than the others? This uncertainty is *subjective*, as it may vary depending on how much information is available about the system, and how that information is used. One of the main critisisms of subjective probability posed by frequentists is that the final probability depends on who you ask.

4.1.1 Bayes' Theorem

To further illustrate the difference between frequentist and Bayesian statistics *Bayes' theorem* [**?**] is introduced. Bayes' theorem can be derived from the familiar rules of probability

$$P(X|I) + P(\bar{X}|I) = 1, \tag{4.1}$$

$$P(X, Y|I) = P(X|Y, I) \times P(Y|I), \tag{4.2}$$

commonly known as the *sum rule* and *product rule*, respectively. $P(X|I)$ is the probability of outcome X given the information I, and $P(X|Y, I)$ is the probability of outcome X given the information I *and* outcome Y. The bar over $\bar{X}$ means that the outcome X does *not* happen. The sum rule states that the total probability of the outcomes X and $\bar{X}$ is equal to 1. This is rather intuitive, considering that an event either takes place or not. The second rule, the product rule, states that the probability of both outcomes X and Y is equal to the probability of Y times the probability of X given that Y has ocurred. These expressions combine to give Bayes' theorem, first formulated by the reverend Thomas Bayes in 1763,

$$P(X|Y, I) = \frac{P(Y|X, I) \times P(X|I)}{P(Y|I)}. \tag{4.3}$$

This theorem states that the probability of X given Y equals the probability of Y given X times the probability of X, divided by the probability of Y. Surprisingly, there is nothing Bayesian about Bayes' theorem. It is merely a reformulation of the rules of logical consistent reasoning by Richard Cox in 1946 [18]. Laplace was the one to make Bayes' theorem Bayesian in the modern statistical sense, when he used the theorem to perform inference about probability distributions [**?**]. The resulting expression is the *posterior probability distribution*

$$P(\Theta|X, I) = \frac{P(X|\Theta, I)P(\Theta|I)}{P(X|I)}, \tag{4.4}$$

where Θ are the probability distribution parameters, X are the possible outcomes, $P(X|\Theta, I)$ and $P(\Theta|I)$ are the *likelihood* and *prior*, respectively, and $P(X|I)$ is a normalization constant called the *marginal likelihood* or evidence. The marginal likelihood is independent of the parameters

$$P(X|I) = \int P(X|\Theta, I)P(\Theta|I)d\Theta. \tag{4.5}$$

In other words, Eq. (4.4) states the probability of the parameters Θ given the knowledge of outcomes X.

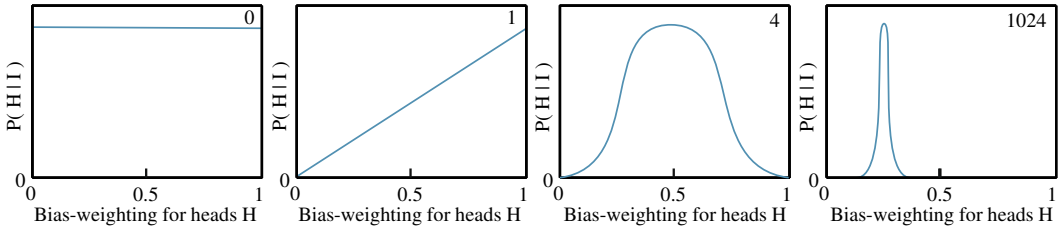


Figure 4.1: The posterior probability distribution of the bias-weighting of a coin for the uniform prior, $P(H|I)$. The first panel from the left is before the coin is tossed, the second panel is after 1 toss, the third is after 4 tosses, and the fourth is after 1024 tosses. The posterior converges towards a narrow peak at 0.25, so the coin is biased.

A crucial parting of Bayesian statistics from frequentist statistics is at the introduction of the *prior*, which expresses a probability distribution on the *parameters* of the probability distribution before data. The prior and likelihood are discussed in the next section, while the marginal likelihood is revisited in Sec. 1.4.

4.1.2 Priors and Likelihood

The likelihood $P(X|\Theta, I)$ is simply the probability of the observations X given the parameters of the probability distribution Θ, and is revisited in Sec. 4.4.1. Conversely, the prior expresses a prior belief or assumption of the parameters, and has to be determined beforehand. As mentioned previously, the measure $P(\Theta|X, I)$ from Eq. (4.4) is called the posterior distribution. This can be thought of as the prior belief, modified by how well this belief fits the data,

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{marginal likelihood}}.$$

Consider an example. The statistician from before now sets about tossing a coin. Before tossing he assumes the probability of heads is uniformly distributed, and so adopts a flat, or uniform, prior probability distribution. The uniform distribution is illustrated in the first panel in Fig. 4.1. After one toss he gets heads, and the posterior changes to a function with high probability for heads, and low for tails, illustrated in the second panel. After four tosses, of which two gave heads and two gave tails, the posterior in the third panel shows an equal probabilty for heads and tails, with a wide distribution centered at 0.5. After several tosses the distribution converges to a narrow peak around 0.25, illustrated in the fourth panel. This indicates an unfair coin that is biased towards tails.

4.1.3 Best Estimate and Reliability

Given a posterior distribution $P(X|\mathcal{D}, I)$ over some random variable X where the prior, $P(X|I)$, has been modified by some data $\mathcal{D}$, it is important to decide how well the posterior fits the data. As will be shown, the posterior can be approximated by a Gaussian distribution, defined by the *best estimate* and *reliability* of the posterior. The best estimate X_0 is the outcome with the highest probability. In other words, it is the maximum of the posterior distribution

$$\frac{dP}{dX}\Big|_{X_0} = 0, \qquad \frac{d^2P}{dX^2}\Big|_{X_0} < 0, \tag{4.6}$$

where P is the posterior $P(X|\mathcal{D}, I)$. The second derivative must be negative to ensure that X_0 is, in fact, a maximum.

Once a best estimate is found, it is important to know how reliable it is. Reliability, or uncertainty, is connected to the width of the distribution. The width of the distribution tells how much the random variables X are smeared out around the mean value X_0. A narrow distribution has low uncertainty, while a wide distribution has large uncertainty. As an example, the third panel in Fig. 4.1 shows a distribution with a mean value of 0.5 with large uncertainty, while the fourth panel shows a distribution with mean 0.25 with small uncertainty.

The width is found by taking the logarithm [1] and Taylor expanding the posterior distribution $P(X|\mathcal{D}, I)$

$$L = L(X_0) + \frac{1}{2}\frac{d^2L}{dX^2}\Big|_{X_0}(X - X_0)^2 + ..., \qquad L = \log_e\Big[P(X|\mathcal{D}, I)\Big] \tag{4.7}$$

The first term, $L(X_0)$, is just a constant. From Eq. (4.6) the condition of the best estimate is that $dL/dX|_{X_0} = 0$. The dominant term in determining the width is therefore the quadratic term in Eq. (4.7).

The Gaussian Distribution

Taking the exponential of Eq. (4.7) and ignoring higher order terms, the posterior can then be approximated as

$$P(X|\mathcal{D}, I) \approx A \exp\left[\frac{1}{2}\frac{d^2L}{dX^2}\Big|_{X_0}(X - X_0)^2\right], \tag{4.8}$$

where $A = \exp\big[L(X_0)\big]$ is a constant.

Equation (4.8) is now in the shape of a *Gaussian distribution*, given by

$$P(X|\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(X - \mu)^2}{2\sigma^2}\right], \tag{4.9}$$

[1] L is a monotonic function of P, so the maximum of L is at the maximum of P.

where μ and σ^2 are the two parameters of the Gaussian distribution mentioned above. An example of a Gaussian distribution is shown in Fig. 4.2. μ is the *mean* value of X, which will be written as $m(X)$, and σ^2 is shorthand for the *variance* of the distribution around the mean $m(X)$, which will be written as $\mathbb{V}(X)$. The variance is discussed further below. The mean and variance for the approximated Gaussian distribution of $P(X|\mathcal{D}, I)$ are then given by

$$m(X) = X_0,\ \mathbb{V}(X) = \Big(- \frac{d^2 L}{dX^2} \Big)^{-1/2}. \tag{4.10}$$

The Gaussian distribution is also referred to as the *normal distribution*, and a Gaussian distribution with mean $m(X) = \mu$ and variance $\mathbb{V}(X) = \sigma^2$ is therefore written as $\mathcal{N}(\mu, \sigma^2)$. A random variable X drawn from this distribution is denoted by a tilde, $X \sim \mathcal{N}(\mu, \sigma^2)$. The Gaussian distribution is symmetric with respect to the maximum at the mean μ, and has a full width at half maximum (FWHM) at around 2.35σ, where $\sigma = \sqrt{\sigma^2}$ is the standard deviation.

In Gaussian process regression of a function $f(X)$ the Gaussian distribution is central, as the basic idea is to predict a Gaussian distribution over function values $f(X)$ for each X. Gaussian processes are discussed in Sec. 4.3, so for now it will suffice to sum up that the quality of a posterior distribution, $P(X|\mathcal{D}, I)$, can be summed up in two measures: the best estimate and the reliability. These can be seen as the mean and variance of a Gaussian distribution $\mathcal{N}(m(X), \mathbb{V}(X))$,

$$m(X) \ : \ \text{Mean of } X, \tag{4.11}$$
$$\mathbb{V}(X) \ : \ \text{Variance of } X. \tag{4.12}$$

Variance

Now that some basics of Gaussian processes have been covered, a more formal definition of the variance is in order. The variance $\mathbb{V}(X)$ is defined as the expectation value of the square of deviations from the mean. The expectation value of a random variable X with a probability distribution $P(X|I)$ is here written as $\mathbb{E}[X]$, and defined as

$$\mathbb{E}[X] = \int X P(X|I) dX \ : \ \text{Expectation value of } X. \tag{4.13}$$

For the posterior probability distribution $P(X|\mathcal{D}, I)$ the variance of the random variable X is then given by [18]

$$\mathbb{V}(X) = \mathbb{E}\big[(X - X_0)^2\big] = \int \int (X - X_0)^2 P(X|\mathcal{D}, I) dX. \tag{4.14}$$

The variance in X is often denoted $\sigma_X^2 = \mathbb{V}(X)$, and its square root is the *standard deviation* $\sqrt{\sigma_X^2} = \sigma_X$.

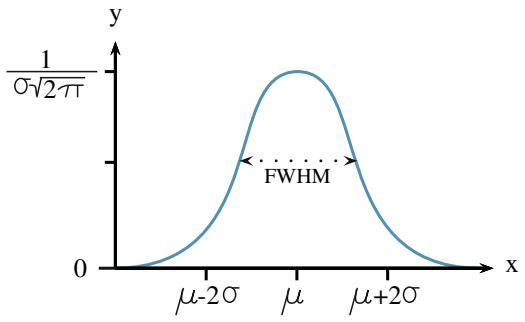


Figure 4.2: A Gaussian probability distribution. The maximum is at the mean value μ, with a full width at half maximum (FWHM) at around 2.35σ. Figure from [18].

4.1.4 Covariance

In distributions over several random variables, $P(X_i|\mathcal{D}, I)$, varying one variable can affect the variance of another variable. This is called *covariance*. For these distributions the equations are not as simple to solve as in Eq. (4.7). In the case of several random variables X_i, a set of *simultaneous equations* must be solved to get the best estimate

$$\frac{dP}{dX_i}\Big|_{X_{0j}} = 0, \qquad \frac{d^2P}{dX_i^2}\Big|_{X_{0j}} < 0 \tag{4.15}$$

To simplify expressions consider the problem in two dimensions, so that $\{X_i\} = (X, Y)$. Analogously to Eq. (4.7), the Taylor expansion of $L = \log_e\Big[P(X, Y|I)\Big]$ is found

$$\begin{aligned} L =& L(X_0, Y_0) + \frac{1}{2}\Big[\frac{d^2L}{dX^2}\Big|_{X_0,Y_0}(X - X_0)^2 \\ &+ \frac{d^2L}{dY^2}\Big|_{X_0,Y_0}(Y - Y_0)^2 + 2\frac{d^2L}{dX\,dY}\Big|_{X_0,Y_0}(X - X_0)(Y - Y_0)\Big] + \ldots \end{aligned} \tag{4.16}$$

There are now four partial derivatives, reduced to three using the rules for mixed partial derivatives $\frac{\partial^2}{\partial X \partial Y} = \frac{\partial^2}{\partial Y \partial X}$. Writing the quadratic terms of Eq. (4.16) in matrix form gives

$$Q = \begin{pmatrix} X - X_0 & Y - Y_0 \end{pmatrix} \begin{pmatrix} A & C \\ C & B \end{pmatrix} \begin{pmatrix} X - X_0 \\ Y - Y_0 \end{pmatrix}, \tag{4.17}$$

where the matrix elements are

$$A = \frac{\partial^2 L}{\partial X^2}\Big|_{X_0,Y_0}, \qquad B = \frac{\partial^2 L}{\partial Y^2}\Big|_{X_0,Y_0}, \qquad C = \frac{\partial^2 L}{\partial X \partial Y}\Big|_{X_0,Y_0}. \tag{4.18}$$

The expression for the variance of X for the distribution $P(X,Y|I)$ is very similar to Eq. (4.14), except for an additional integral over the random variable Y

$$\mathbb{V}(X) = \sigma_X^2 = \mathbb{E}\big[(X - X_0)^2\big] = \int\int (X - X_0)^2 P(X,Y|\mathcal{D},I) dX dY. \tag{4.19}$$

A similar expression, σ_Y^2, can be found for Y by substituting X and Y.

The simultaneous deviations of the random variables X and Y is the aforementioned covariance, often written as σ_{XY}^2. In two dimensions the covariance is given by

$$\sigma_{XY}^2 = \mathbb{E}\big[(X - X_0)(Y - Y_0)\big] = \int\int (X - X_0)(Y - Y_0) P(X,Y|\mathcal{D},I) dX dY. \tag{4.20}$$

The covariance indicates how an over- or underestimation of one random variable affects another. If, for example, an overestimation of X leads to an overestimation of Y, the covariance is positive. An example of positive covariance is shown in the third panel of Fig. 4.3. If the overestimation of X has little or no effect on the estimation of Y, the covariance is negligible or zero $|\sigma_{XY}| \ll \sqrt{\sigma_X^2 \sigma_Y^2}$, as seen in the first panel of Fig. 4.3. The second panel shows negaitve covariance.

Covariance Matrix

The variances and covariances are the elements of the *covariance matrix*. For N random variables $X_1, ..., X_N$ the covariance matrix is an $N \times N$-matrix. The covariance matrix for X and Y is here denoted $\mathrm{cov}(X,Y)$, and it can be shown that [18]

$$\mathrm{cov}(X,Y) = \begin{pmatrix} \sigma_X^2 & \sigma_{XY}^2 \\ \sigma_{XY}^2 & \sigma_Y^2 \end{pmatrix} = -\begin{pmatrix} A & C \\ C & B \end{pmatrix}^{-1}. \tag{4.21}$$

To sum up, the posterior probability distribution over a random variable X is denoted $P(X|\mathcal{D},I)$, and its best estimate and the associated reliability can be approximated as a Gaussian distribution $\mathcal{N}(m(X), \mathbb{V}(X))$. The Gaussian distribution is defined by the mean value $m(X)$ and the variance $\mathbb{V}(X)$. For distributions over several random variables X_i one can also find the covariance $\sigma_{X_i X_j}^2$, and all variances and covariances are contained in the covariance matrix $\mathrm{cov}(X_i)$. In the next section *covariance functions* are introduced, which are used to calculate the elements of the covariance matrix.

4.2 Covariance Functions

As mentioned in Sec. 4.1.4, the elements of a covariance matrix can be determined by *covariance functions*, or *kernels*. A function that maps two arguments $\mathbf{x} \in$

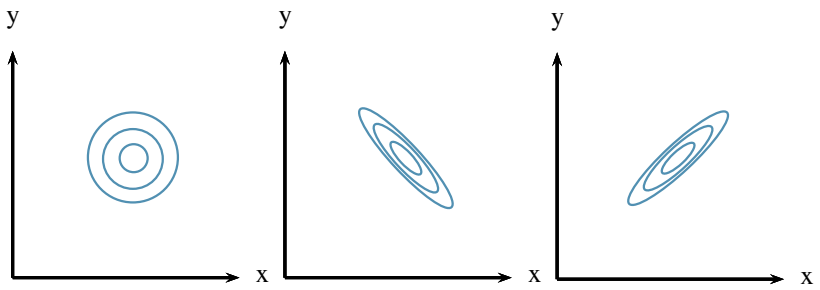


Figure 4.3: A schematic illustration of covariance and correlation. (a) The contours of a posterior pdf with zero covariance, where the inferred values of X and Y are uncorrelated. (b) The corresponding plot when the covariance is large and negative; (c) The case of positive correlation.

$\mathcal{X}$, $\mathbf{x}' \in \mathcal{X}$ into $\mathbb{R}$, where $\mathcal{X}$ is the input space, is generally called a kernel k. Covariance functions are symmetric kernels, meaning that $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$. In this project both kernel and covariance function are taken to mean covariance function.

In Gaussian processes, which are discussed in Sec. 4.3., the covariance between two function values $f(\mathbf{x})$ and $f(\mathbf{x}')$, where $\mathbf{x}$ and $\mathbf{x}'$ are input vectors, is given by the covariance of the input vectors. Further, the covariance of the input vectors is given by a kernel,

$$\text{cov}(f(\mathbf{x}), f(\mathbf{x}')) = k(\mathbf{x}, \mathbf{x}'). \tag{4.22}$$

The matrix containing all the covariance elements is, as mentioned, called the *covariance matrix*, or the *Gram matrix* K, whose elements are given by

$$K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j). \tag{4.23}$$

Note that a covariance matrix calculated using a covariance function k is denoted by a capital K.

A kernel function that only depends on the difference between two points, $\mathbf{x} - \mathbf{x}'$, is called *stationary*. This means that the function is invariant to translations in input space. If, in addition, it only depends on the length $r = |\mathbf{x} - \mathbf{x}'|$, the function is *isotropic* [2]. Isotropic functions are commonly referred to as *radial basis functions* (RBFs), as they are only functions of r.

The covariance function can also depend on the dot product, $\mathbf{x} \cdot \mathbf{x}'$, and is then called a *dot product* covariance function. The most important covariance functions for this project are the squared exponential covariance function and the Matérn class of covariance functions.

[2]Invariant to rigid rotations in input space.

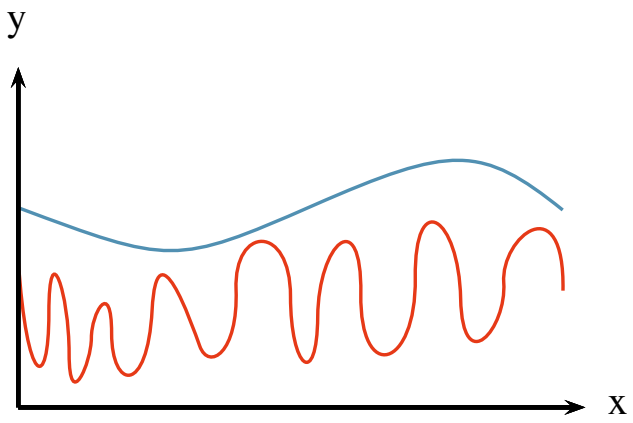


Figure 4.4: The effect of varying the length scale ℓ. A long length scale (blue) gives a smooth, slowly varying function, while a short length scale (red) gives a more staccato, quickly varying function.

4.2.1 The Squared Exponential Covariance Function

The *squared exponential covariance function* (SE) has the form

$$k_{SE}(r) = \exp\Big(-\frac{r^2}{2\ell^2}\Big), \tag{4.24}$$

where ℓ is the *characteristic length scale*. Equation (4.24) is sometimes also called the *radial basis function* (RBF). The length scale, ℓ, determines the smoothness of the function. It can be loosely interpreted as how far you need to move (along a particular axis) in input space for the function values to become uncorrelated. For a large length scale one should expect a very slowly varying function, while a shorter length scale means a more rapidly varying function, see the illustration in Fig. 4.4. The SE is infinitely differentiable and therefore very smooth. Stein [19] argues that such strong smoothness assumptions are unrealistic for most physical problems, and recommends another class of kernels, namely the *Matérn class of covariance functions*.

4.2.2 The Matérn Class of Covariance Functions

The *Matérn class of covariance functions* is given by

$$k_{Matérn}(r) = \frac{2^{1-\nu}}{\Gamma(\nu)}\Big(\frac{\sqrt{2\nu}r}{\ell}\Big)^{\nu} K_\nu\Big(\frac{\sqrt{2\nu}r}{\ell}\Big), \tag{4.25}$$

where $\nu, \ell > 0$, and K_ν is a modified Bessel function [20]. The hyperparameter ν controls the smoothness of the function. For $\nu \to \infty$ this becomes the SE

kernel, and for $\nu = 1/2$ is becomes the very rough absolute exponential kernel $k(r) = \exp(-r/\ell)$. In the case of half integer ν, $\nu = p + \frac{1}{2}$ for $p \in \mathbb{N}$, the covariance function is simply the product of an exponential and a polynomial

$$k_{\nu=p+\frac{1}{2}} = \exp\Big(-\frac{\sqrt{2\nu}r}{\ell}\Big)\frac{\Gamma(p+1)}{\Gamma(2p+1)}\sum_{i=0}^{p}\frac{(p+i)!}{i!(p-i)!}\Big(\frac{\sqrt{8\nu}r}{\ell}\Big)^{p-i}. \tag{4.26}$$

In machine learning the two most common cases are for $\nu = 3/2$ and $\nu = 5/2$

$$k_{\nu=3/2}(r) = \Big(1 + \frac{\sqrt{3}r}{\ell}\Big)\exp\Big(-\frac{\sqrt{3}r}{\ell}\Big), \tag{4.27}$$

$$k_{\nu=5/2}(r) = \Big(1 + \frac{\sqrt{5}r}{\ell} + \frac{5r^2}{3\ell^2}\Big)\exp\Big(-\frac{\sqrt{5}r}{\ell}\Big). \tag{4.28}$$

4.2.3 Noise

The covariance function can also contain information about noise in the data. In the case where the noise, ε, follows a Gaussian distribution $\varepsilon \sim \mathcal{N}(0, \sigma_n^2)$, the noise is represented by adding the variance, σ_n^2, to the diagonal of the covariance matrix

$$k(\mathbf{x}_i, \mathbf{x}_j)_{noise} = \sigma_n^2\delta_{ij}, \tag{4.29}$$

where δ_{ij} is the Kronecker-delta. The model where the noise is assumed to follow a Gaussian distribution is called the *Gaussian noise model*, and is discussed in Sec. 4.3.

4.2.4 Hyperparameters

The SE kernel in Eq. (4.24) and the Matérn kernel in Eq. (4.25) are both isotropic, *i.e.* functions only of the distance between input points, r. Isotropic kernels can, however, be generalized to an anisotropic form by setting

$$r^2(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T M(\mathbf{x} - \mathbf{x}'), \tag{4.30}$$

where M is some positive definite matrix. M can take multiple forms, such as

$$M_1 = \ell^{-2}\mathbb{I}, \qquad M_2 = \mathrm{diag}(\boldsymbol{\ell})^{-2}, \tag{4.31}$$

where ℓ^2 is a scalar, and $\boldsymbol{\ell}$ is a vector of the same dimension as the input vectors $\mathbf{x}$, $\mathbf{x}'$. Choosing $\boldsymbol{\ell}$ to be a vector in stead of a scalar is in many cases useful, especially if the input vector $\mathbf{x}$ contains values of different scales.

M is now part of the *hyperparameters* of the kernel. Each kernel has a vector of hyperparameters, denoted $\boldsymbol{\theta}$. An example is the following generalisation of the SE kernel,

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2\exp\big(-\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)^T M(\mathbf{x}_i - \mathbf{x}_j)\big) + \sigma_n^2\delta_{ij}, \tag{4.32}$$

where σ_f is a scaling factor and σ_n^2 is the Gaussian noise parameter, discussed in Sec. 4.2.3. The hyperparameters of this kernel are $\boldsymbol{\theta} = (\{M\}, \sigma_f^2, \sigma_n^2)^T$, where $\{M\}$ denotes the parameters in the symmetric matrix M.

Other Covariance Functions

There are several other types of covariance functions that are not discussed here. In addition, kernels can be multiplied and summed to form new kernels, making the space of possible kernels infinite. For further details see chapter 4 in [21].

4.3 Gaussian Process Regression

Gaussian processes (GP) is a supervised machine learning method, designed to solve regression and probabilistic classification problems. Only regression is considered here. This section begins by introducing Gaussian processes and important notation, first in a general sense, and then in the function space view. Then distributions over functions are considered, followed by a short discussion on how functions can be drawn from these distributions. Finally, a quick overview of the noise-free model and the Gaussian noise nodel, and their covariance matrices, are given.

Gaussian Processes

Consider a set of points $\mathcal{D} = \{\mathbf{x}_i, y_i\}$, where y is some (possibly noisy) function of $\mathbf{x}$, $y(\mathbf{x}) = f(\mathbf{x}) + \varepsilon$. Here, ε is the noise and $f(\mathbf{x})$ is the true value of the function. These points are illustrated by the black dots in Fig. 4.5. In machine learning $\mathcal{D}$ is the *training data*, as it is used to train the model. It consists of *features*, which are the components of the input vectors $\mathbf{x}_i$, and *targets*, which are the function values y_i. The set of points is discrete, so there is some $\mathbf{x}^*$ for which the target y^* is unknown. The test point $\mathbf{x}^*$ is marked on the x-axis in Fig. 4.5.

Gaussian processes (GP) predict a Gaussian distribution *over function values* at this point $\mathbf{x}^*$. The distribution for a single test point $\mathbf{x}^*$ has a corresponding mean $m(\mathbf{x}^*)$ and variance $\mathbb{V}(\mathbf{x}^*)$. Note that the mean $m(\mathbf{x}^*)$ *is not the mean of the input vector* $\mathbf{x}^*$, but rather *the mean of function values* $f(\mathbf{x})$ *evaluated at* $\mathbf{x}^*$. The GP prediction for the target value $y^* = f(\mathbf{x}^*)$ is the mean $m(\mathbf{x}^*)$. Similarly, the variance, $\mathbb{V}(\mathbf{x}^*)$, is in fact the variance in function values $f(\mathbf{x}^*)$, or the width of the Gaussian distribution (red line) in the y-direction in Fig. 4.5. The mean value y^* is drawn as a blue cross in Fig. 4.5. As will be shown, the predicted target y^* is a linear combination of the known targets y_i, where the weights are controlled by the covariances between training points $\mathbf{x}_i$ and the test point $\mathbf{x}^*$.

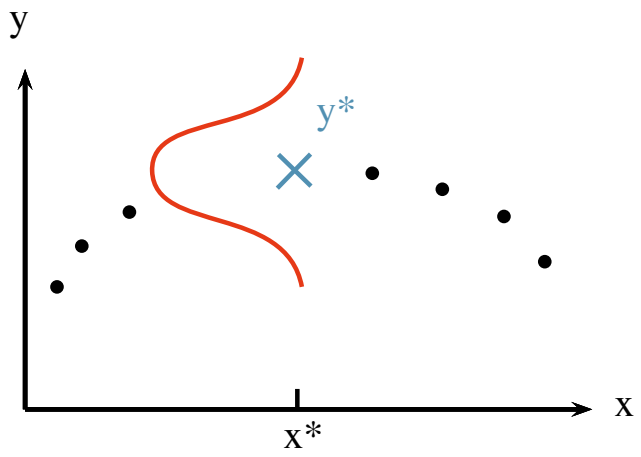


Figure 4.5: An illustration of a GP prediction of the target value y^* (blue cross), given the known set of points $\{x_i, y_i\}$ (black dots). The prediction is a Gaussian distribution in y with mean y^* and variance σ^2. The Gaussian distribution is drawn in red with y on the vertical axis, with uncertainty in the y-direction.

Some Notation

As Gaussian process regression is notoriously confusing, it is helpful to begin with some notation.

As discussed in Sec. 4.3, the Gaussian distribution with a mean μ and variance σ^2 is denoted $\mathcal{N}(\mu, \sigma^2)$. The distribution can be over a single random variable, *e.g.* f, or a finite set of random variables, f_i. A Gaussian distribution over several random variables is called a *multivariate Gaussian distribution*. The n random variables f_i, $i = 1, ..., n$ drawn from a multivariate Gaussian distribution make up the n-dimensional vector $\mathbf{f}$. The mean values, μ_i, are then contained in the n-dimensional *mean vector* $\bar{\mathbf{f}}$. The variance, σ^2, is replaced by an $n \times n$-dimensional covariance matrix, $\mathrm{cov}(\mathbf{f}, \mathbf{f})$. The multivariate Gaussian distribution over $\mathbf{f}$ is written as

$$\mathbf{f} \sim \mathcal{N}\big(\bar{\mathbf{f}},\ \mathrm{cov}(\mathbf{f}, \mathbf{f})\big). \tag{4.33}$$

For n points $\{\mathbf{x}_i, y_i\}$, where $\mathbf{x}_i$ is an m-dimensional feature vector and y_i is the target, the features comprise the $n \times m$-matrix X. The targets make up the corresponding n-dimensional vector $\mathbf{y}$. A central assumption in Gaussian processes is that the covariance between targets y_i, y_j is given by the covariance of their features $\mathbf{x}_i$, $\mathbf{x}_j$

$$\mathrm{cov}(y_i, y_j) = k(\mathbf{x}_i, \mathbf{x}_j), \tag{4.34}$$

where $k(\mathbf{x}_i, \mathbf{x}_j)$ is a covariance function, as described in Sec. 4.2. In Gaussian processes the distribution over target values $\mathbf{f}$ will therefore be written

$$\mathbf{f} \sim \mathcal{N}\big(\, \bar{\mathbf{f}}, K(X, X) \,\big), \tag{4.35}$$

where $K(X, X)$ is the covariance matrix containing the covariances of the features contained in X, calculated using the covariance function $k(\mathbf{x}, \mathbf{x}')$.

Finally, a Gaussian process will be denoted $\mathcal{GP}$. It may be difficult to distinguish between a Gaussian *distribution*, $\mathcal{N}$, and a Gaussian *process*, $\mathcal{GP}$. The difference can be thought of as the difference between a finite collection of function values $f(\mathbf{x}_i)$, and the continuous function, $f(\mathbf{x})$. The former can be viewed as a vector $\mathbf{f}$, and can be drawn from a distribution such as the one in Eq. (4.35), $\mathbf{f} \sim \mathcal{N}(\bar{\mathbf{f}}, K(X, X))$. The latter is a *function*, drawn from a distribution *over functions*, where the mean $m(\mathbf{x})$ and covariances $k(\mathbf{x}, \mathbf{x}')$ are functions as well. A function $f(\mathbf{x})$ drawn from a Gaussian process is written as

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')). \tag{4.36}$$

Function Space View

Gaussian processes provide distributions over functions $\mathcal{GP}$. It is therefore useful to consider the problem in the function space view introduced in [21]. For a function $f(\mathbf{x})$ the Gaussian process mean $m(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$ are defined as

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})], \tag{4.37}$$

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))]. \tag{4.38}$$

In other words, the mean $m(\mathbf{x})$ is the expected value of the function $f(\mathbf{x})$ at $\mathbf{x}$, and the covariance function is the expected value of the simultaneous deviations of $f(\mathbf{x})$ from the mean $m(\mathbf{x})$ at $\mathbf{x}$, and $f(\mathbf{x}')$ from the mean $m(\mathbf{x}')$ at $\mathbf{x}'$. As mentioned, the mean and covariance are now *functions* of the input vector $\mathbf{x}$. This means that for every input vector $\mathbf{x}$, there is a Gaussian distribution over function values with a mean $m(\mathbf{x})$ and covariance given by $k(\mathbf{x}, \mathbf{x}')$. This is a generalization of the single test point in Fig. 4.5, where every point $\mathbf{x}^*$ gets a similar distribution.

The collection of Gaussian distributions over functions that are now a function of the input vector $\mathbf{x}$, are the Gaussian processes, $\mathcal{GP}$. In the same way that a random variable is drawn from a distribution, random functions can be drawn from the $\mathcal{GP}$,

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')). \tag{4.39}$$

How functions are drawn from distributions is discussed in more detail in a later section.

The covariance between two points $f(\mathbf{x})$, $f(\mathbf{x}')$ is determined by the covariance function $k(\mathbf{x}, \mathbf{x}')$, as discussed in Sec. 4.2. Since the distribution $\mathcal{GP}$ is over function values, and so the covariance function calculates the covariance between the *function values*, $f(\mathbf{x}_i)$ and $f(\mathbf{x}_j)$, and *not* the input vectors, $\mathbf{x}_i$ and $\mathbf{x}_j$. Rather, the covariance between two function values is a function of the input vectors. Consider again the illustration in Fig. 4.4. The variation in function values $f(\mathbf{x}_i)$ and $f(\mathbf{x}_j)$ depends on the distance between the points $\mathbf{x}_i$ and $\mathbf{x}_j$, and the characteristic length scale of the process. If the length scale is large, the two input vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ can be far away, and still have similar function values $f(\mathbf{x}_i)$ and $f(\mathbf{x}_j)$. For short length scales, however, nearby points can have very different function values, because the function varies rapidly.

Distributions over Functions

As metioned above, functions can be drawn from the Gaussian process with mean $m(\mathbf{x})$ and covariance $k(\mathbf{x}, \mathbf{x}')$, $\mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$. All functions $f(\mathbf{x})$ that are drawn from this distribution must have a covariance between function values $f(\mathbf{x})$, $f(\mathbf{x}')$ determined by the covariance function $k(\mathbf{x}, \mathbf{x}')$. If for example the covariance function has a large length scale, quickly varying functions are not allowed. The mean of the distribution, $m(\mathbf{x})$, is *not* the mean value of each function $f(\mathbf{x})$ drawn from $\mathcal{GP}$, but rather the mean function value you would get in $\mathbf{x}$ if you drew enough functions from $\mathcal{GP}$. Drawing functions from a distribution in this way will be referred to as *drawing samples*.

Gaussian processes are Bayesian in that there is a prior and a posterior distribution over functions, where the posterior is obtained by conditioning the prior on the training data. Consider the $n^* \times m$-matrix of test points X^*, containing n^* test points $\mathbf{x}_i^*$, with unknown function values $f(\mathbf{x}^*)$. Using the kernel on the matrix X^* gives the covariance matrix, as discussed in Sec. 4.2. The covariance matrix $K(X^*, X^*)$ now contains the covariance of all test points $\mathbf{x}_i^*$, calculated using the covariance function $k(\mathbf{x}_i^*, \mathbf{x}_j^*)$. Combined with an initial mean of zero [3] one obtains the *prior* distribution of predicted target values $\mathbf{f}^*$

$$\mathbf{f}^* \sim \mathcal{N}(\mathbf{0}, K(X^*, X^*)). \tag{4.40}$$

This distribution contains the prior assumptions about the function values $f(\mathbf{x})$, in that the smoothness of the function and the correlation between function values are encoded in the covariance matrix. This is the prior probability distribution discussed in Sec. 4.1.2, that will be modified by the data to provide the posterior probability distribution. The choice of kernel is therefore one of the most important steps in learning with GPs.

[3] The mean does not have to be zero, it could for example be the mean of the training data.

Noise-Free Model

Now consider the addition of training data to the prior distribution in Eq. 4.40, in the form of a noise-free set of n training points $\{\mathbf{x}_i, y_i\}$, so that $y = f(\mathbf{x})$. The input vectors $\mathbf{x}_i$ form the $n \times m$-matrix X, where the rows are the input vectors. The training targets y_i form the corresponding n-dimensional vector $\mathbf{f}$. The test points are still contained in the $n^* \times m$ matrix X^*. The goal is to predict an n^*-dimensional vector $\mathbf{f}^*$ containing the predictions of the function values at the points $\mathbf{x}_i^*$, which is conditioned on the known function values $\mathbf{f}$.

The joint distribution of training outputs, $\mathbf{f}$, and test outputs, $\mathbf{f}^*$, according to the prior in Eq. 4.40 is

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X^*) \\ K(X,X^*) & K(X^*,X^*) \end{bmatrix}\right), \tag{4.41}$$

where, as before, $K(X_i, X_j')$ is the covariance matrix between the sets of points $\{\mathbf{x}_i\}$ and $\{\mathbf{x}_j'\}$ calculated using the covariance function $k(\mathbf{x}_i, \mathbf{x}_j')$. By conditioning the distribution of $\mathbf{f}^*$ on the observations $\mathbf{f}$, the posterior distribution over $\mathbf{f}^*$ is obtained[4] [21]

$$\mathbf{f}^* \big| X^*, X, \mathbf{f} \sim \mathcal{N}(K(X^*,X)K(X,X)^{-1}\mathbf{f}, \tag{4.42}$$

$$K(X^*,X^*) - K(X^*,X)K(X,X)^{-1}K(X,X^*)), \tag{4.43}$$

where $\mathbf{f}^*|X^*, X, \mathbf{f}$ means the target values $\mathbf{f}^*$ for the features X^*, given the known features X and targets $\mathbf{f}$. The prior in Eq. 4.40 has now been modified by the data X, $\mathbf{f}$ to give the posterior in Eq. 4.42, leaving two distributions from which function values can be drawn.

Drawing Samples

To generate samples $\mathbf{f} \sim \mathcal{N}(\mathbf{m}, K)$ with mean $\mathbf{m}$ and covariance matrix K using a scalar Gaussian generator[5], one proceeds as follows: first the Cholesky decomposition — also known as the matrix square root — L of the covariance matrix is found using $K = LL^T$. L is a lower triangular matrix. A vector $\mathbf{u}$ is then generated by multiple calls to the scalar Gaussian generator $\mathbf{u} \sim \mathcal{N}(0, I)$. Then $\mathbf{f} = \mathbf{m} + L\mathbf{u}$ has the desired distribution with mean $\mathbf{m}$ and covariance $L\mathbb{E}[\mathbf{u}\mathbf{u}^T]L^T = LL^T = K$ [21].

Using the method described above, random functions have been drawn from the prior and posterior in Eq. 4.40 and Eq. 4.42, respectively, and shown in Fig. 4.6. As discussed, the samples drawn from the prior have mean equal to

[4] For more details, see Appendix A.2 in [21].

[5] A scalar Gaussian generator generates random numbers from a Gaussian distributions, and can be found in most programming enviroments, such as the `random` environment in `Python`.

zero $m(\mathbf{x}) = \mathbf{0}$, and constant covariance, $K(X^*, X^*)$, meaning that if you drew enough functions the mean of all function values at every $\mathbf{x}$ would be zero. The prior is shown in the upper panel of Fig. 4.6, where the mean of the distribution is represented by the thick, black line, and the covariance is the light blue band around the mean. In the posterior distribution the mean values and covariances have been modified by the training data, represented by red dots in the lower panel of Fig. 4.6. In a point where there is training data the uncertainty is zero[6], and so all samples drawn from the posterior distribution must pass through this point. Far away from training points the covariance is large. The mean has also been modified to pass through the training points, as seen by the thick black line in the lower panel.

Gaussian Noise Model

Noise-free observations are rare. In most cases targets will contain some noise $y = f(\mathbf{x}) + \varepsilon$, where the noise ε is assumed to follow a Gaussian distribution $\varepsilon \sim \mathcal{N}(0, \sigma_n^2)$. This is the *Gaussian noise model*. As discussed in Sec. 4.2 the covariance of a function with Gaussian noise can be expressed as

$$\text{cov}(y_i, y_j) = k(\mathbf{x}_i, \mathbf{x}_j) + \sigma_n^2 \delta_{ij}. \tag{4.44}$$

Training targets are now contained in the n-dimensional vector $\mathbf{y}$, while training features are contained in the $n \times m$-matrix X, test features in the $n^* \times m$-matrix X^* and predicted targets in the n^*-dimensional vector $\mathbf{f}^*$, as before. With the addition of the noise the prior distribution becomes

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \sigma_n^2 \mathbb{I} & K(X,X^*) \\ K(X,X^*) & K(X^*,X^*) \end{bmatrix}\right). \tag{4.45}$$

The conditioned distribution is then

$$\mathbf{f}^* \big| X^*, X, \mathbf{f} \sim \mathcal{N}(\bar{\mathbf{f}}^*, \text{cov}(\mathbf{f}^*)) \tag{4.46}$$

where

$$m(\mathbf{f}^*) = K(X^*, X)[K(X,X) + \sigma_n^2 \mathbb{I}]^{-1} \mathbf{y}, \tag{4.47}$$

$$\text{cov}(\mathbf{f}^*) = K(X^*, X^*) - K(X^*, X)[K(X,X) + \sigma_n^2 \mathbb{I}]^{-1} K(X, X^*). \tag{4.48}$$

Equations (4.47)-(4.48) are the key predictive equations for Gaussian process regression. The calculation requires the inverting the $n \times n$-matrix $[K(X,X) + \sigma_n^2 \mathbb{I}]$, which for large n becomes computationally unviable. This is discussed further in Sec. **??**, where Distributed Gaussian processes are proposed as a way of scaling Gaussian processes to larger training sets.

[6] Assuming there is no noise in the data.

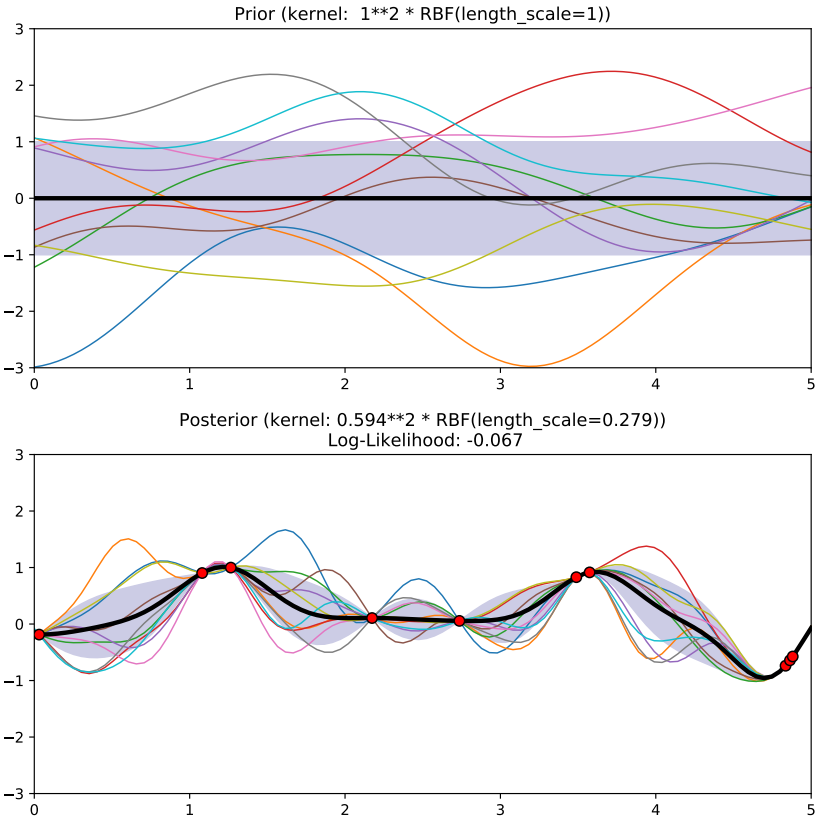


Figure 4.6: Drawing functions from the prior (top) and posterior (bottom) distributions. The thick, black line represents the mean of the distribution, while the shaded, blue area is the variance. The multiple colored lines are functions drawn randomly from the prior and posterior distributions, whose correlation are dictated by the covariance function. The prior has mean 0 and covariance given by the squared exponential function. The posterior has been modified by training points (red dots), giving rise to zero uncertainty at the points where training data exists, and an altered mean value for the distribution. The kernel of the prior distribution has hyperparameters $\sigma_f = 1$ and $\ell = 1$, while for the posterior they are $\sigma_f = 0.594$ and $\ell = 0.279$. Figure generated using scikit-learn.

For the sake of tidying up the expression define the matrix $K \equiv K(X, X)$ and the matrix $K^* \equiv K(X, X^*)$. In the case of a single test point $\mathbf{x}^*$ the matrix K^* is written as a vector $\mathbf{k}(X, \mathbf{x}^*) = \mathbf{k}^*$ to denote the covariances between the n training points and the test point $\mathbf{x}^*$. Using this compact notation the GP prediction of $f^* = f(\mathbf{x}^*)$ for a single test point $\mathbf{x}^*$ is

$$m(f^*) = \mathbf{k}^{*T}(K + \sigma_n^2 \mathbb{I})^{-1}\mathbf{y}, \tag{4.49}$$

$$\mathbb{V}[f^*] = k(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}^{*T}(K + \sigma_n^2 \mathbb{I})^{-1}\mathbf{k}^*. \tag{4.50}$$

Note that, as mentioned in the beginning of Sec. 4.3, the predicted mean value $\bar{f}^*$ can be viewed as a linear combination of y_i of the form $\alpha\mathbf{y}$, where $\alpha = \mathbf{k}^{*T}(K + \sigma_n^2 \mathbb{I})^{-1}$. α then only contains the covariance between features.

Eqs. (4.49)-(4.50) form the basis for GP prediction in `scikit-learn` [22]. The algorithm for Gaussian process regression on a single test point $\mathbf{x}^*$, with training data X, $\mathbf{y}$ is outlined in Algorithm 1. The algorithm uses the Cholesky decomposition, L, of the covariance matrix to find the weights $\boldsymbol{\alpha}$ used to calculate the predictive mean f^*. The variance $\mathbb{V}[\mathbf{x}^*]$ is calculated using L and the covariance of the test point with the training points, $\mathbf{k}^* = k(X, \mathbf{x}^*)$.

Data: X (inputs), $\mathbf{y}$ (targets), k (covariance function/kernel), σ_n^2 (noise level), $\mathbf{x}_*$ (test input).
L = Cholesky decomposition $(K + \sigma_n^2 I)$;
$\boldsymbol{\alpha} = (L^T)^{-1}(L^{-1}\mathbf{y})$;
$\bar{f}_* = \mathbf{k}_*^T\boldsymbol{\alpha}$;
$\mathbf{v} = L^{-1}\mathbf{k}_*$;
$\mathbb{V}[f_*] = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{v}^T\mathbf{v}$;
$\log p(\mathbf{y}|X) = -\frac{1}{2}\mathbf{y}^T\boldsymbol{\alpha} - \sum_i \log L_{ii} - \frac{n}{2}\log 2\pi$;
Result: f_* (mean), $\mathbb{V}[f_*]$ (variance), $\log p(\mathbf{y}|X)$ (log marginal likelihood).

Algorithm 1: Algorithm 2.1 from [21].

4.4 Model Selection

Choosing the right kernel and hyperparameters, introduced in Sec. 4.2, is an important part of Gaussian process regression. Finding the kernel and corresponding hyperparameters that best fit the data is often called *model selection*. Model selection is also referred to as *training* in machine learning. In this section Bayesian model selection for the marginal likelihood is quickly overviewed, and the log marginal likelihood and cross validation are introduced as tools to optimize Gaussian process estimators.

4.4.1 Log Marginal Likelihood

The *marginal likelihood* can be used to find the optimal hyperparameters of covariance functions. Gaussian process regression models with Gaussian noise have the wonderful trait of analytically tractable integrals for the marginal likelihood, $P(\mathbf{y}|X, \boldsymbol{\theta})$. Note that the marginal likelihood defined here looks different than in Eq. (4.4) — $\boldsymbol{\theta}$ are the hyperparameters of the covariance functions introduced in Sec. 4.2.4, $\mathbf{y}$ are the training outputs and X are the training inputs. The parameters in Eq. (4.4), denoted Θ, are now the true — or *latent* — function values, $\mathbf{f}$ where $\mathbf{y} = \mathbf{f} + \varepsilon$. The marginal likelihood is the integral of the likelihood times the prior over the latent function values $\mathbf{f}$, given by

$$P(\mathbf{y}|X, \boldsymbol{\theta}) = \int P(\mathbf{y}|\mathbf{f}, X, \boldsymbol{\theta}) P(\mathbf{f}|X, \boldsymbol{\theta}) d\mathbf{f}, \tag{4.51}$$

where $P(\mathbf{f}|X, \boldsymbol{\theta})$ is the prior and $P(\mathbf{y}|\mathbf{f}, X, \boldsymbol{\theta})$ is the likelihood. Under the Gaussian process model the prior is a Gaussian, $\mathbf{y} \sim \mathcal{N}(\mathbf{0}, K + \sigma_n^2 \mathbb{I})$, so the logarithm of the marginal likelihood in Eq. (4.51) gives for the *log marginal likelihood* (LML) [21]

$$\log P(\mathbf{y}|X, \boldsymbol{\theta}) = -\frac{1}{2}\mathbf{y}^T (K + \sigma_n^2 \mathbb{I})^{-1} \mathbf{y} - \frac{1}{2} \log \left| K + \sigma_n^2 \mathbb{I} \right| - \frac{n}{2} \log 2\pi, \tag{4.52}$$

where n is the number of training points. Each term in Eq. (4.52) has an interpretation: $-\frac{1}{2}\mathbf{y}^T (K + \sigma_n^2 \mathbb{I})^{-1} \mathbf{y}$ is the only term involving the data, and is therefore the data-fit; $-\frac{1}{2} \log \left| K + \sigma_n^2 \mathbb{I} \right|$ is the complexity penalty depending only on the covariance function and the inputs; and $-\frac{n}{2} \log 2\pi$ is a normalization term.

As mentioned, the goal is to use the marginal likelihood to determine the optimal hyperparameters of the covariance function, $\boldsymbol{\theta}$. Note that the marginal likelihood in Eq. (4.51) is the probability of the training outputs, $\mathbf{y}$, given the training inputs X and the hyperparameters $\boldsymbol{\theta}$. Maximizing the marginal likelihood, or the LML, for the hyperparameters will therefore give the best estimate for $\boldsymbol{\theta}$, as per the discussion in Sec. 4.1.3. Maximizing the LML requires finding the partial derivatives

$$\frac{\partial}{\partial \theta_j} \log p(\mathbf{y}|X, \boldsymbol{\theta}) = \frac{1}{2}\mathbf{y}^T K^{-1} \frac{\partial K}{\partial \theta_j} K^{-1} \mathbf{y} - \frac{1}{2} \mathrm{tr}(K^{-1} \frac{\partial K}{\partial \theta_j}). \tag{4.53}$$

Using partial derivatives, or the gradients, to find the optimal hyperparameters is called a *gradient based optimizer*. Computing the inverse of a matrix, K^{-1}, is computationally complex, and for n training points goes as $\mathcal{O}(n^3)$. Once this is done, however, finding the partial derivatives only requires complexity $\mathcal{O}(n^2)$, and so gradient based optimizers are advantageous.

The LML can have several local optima, as seen in the contour plot of the log marginal likelihood in Fig. 4.7. These correspond to different interpretations of

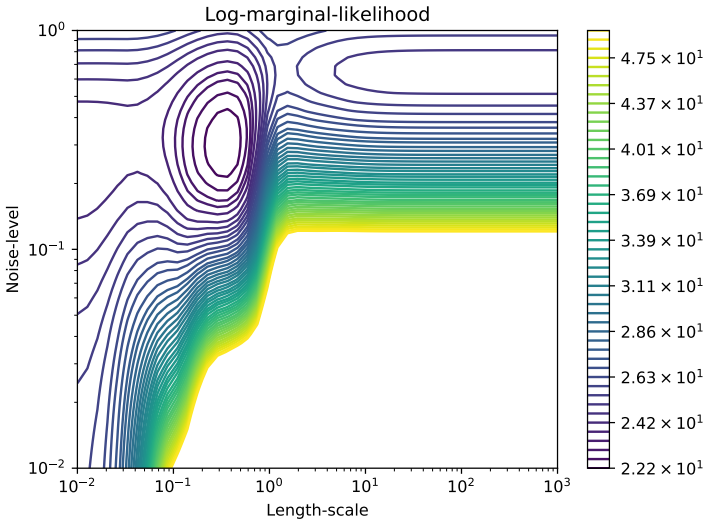


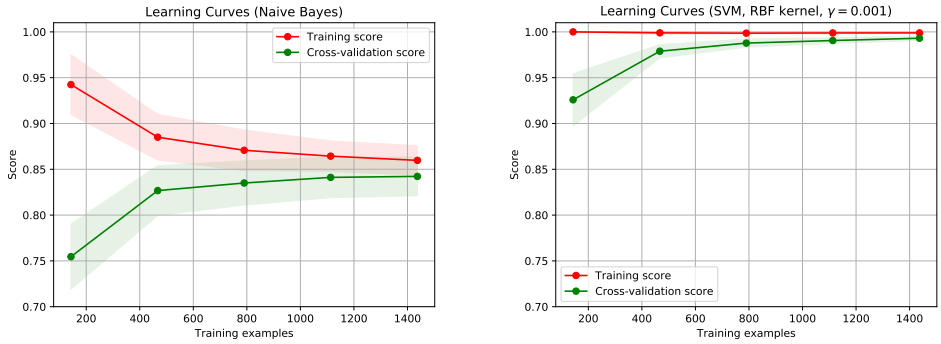
Figure 4.7: A contour plot of the log marginal likelihood with two local optima for a Gaussian process with kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \exp(-(\mathbf{x}_i - \mathbf{x}_j)^2/\ell^2) + \sigma_n^2 \delta_{ij}$. The rightmost optima favours a short length scale and low noise, with $\sigma_f = 0.64$, $\ell = 0.365$ and $\sigma_n^2 = 0.29$, while the leftmost favors a high noise level and therefore several large length scales, with $\sigma_f = 0.00316$, $\ell = 109$ and $\sigma_n^2 = 0.6$. The optimum to the right has LML -21.8 and the optimum to the right has LML -23.87. Plots were generated using scikit-learn.

the data. The leftmost optimum in Fig. 4.7, for example, favors a small length scale and smaller noise level. This means that little of the data is considered to be noise. The rightmost optimum has a higher noise level, and allows for a range of length scales, as most of the data is considered to be noise. Features with very large length scales are considered superfluous, as the function value depends little on them. To avoid ending up in a local optima, it can be wise to restart the optimizer a few times during learning.

4.4.2 Cross Validation

Cross validation is a means of monitoring the performance of a model. In k-fold validation this is done by dividing the data into k subsets and using $k-1$ folds to train the model, and a single fold to validate it. This is repeated k times, one for each possible validation set. Cross-validation requires a scoring function, such as the R^2 score. The R^2-score is given by

$$R^2 = 1 - \frac{\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{N-1}(y_i - \bar{y})^2}, \tag{4.54}$$



(a) Underfitting. **(b)** Good fit.

Figure 4.8: Learning curves for two different estimators. The training scores are shown as red lines, with uncertainty bands in light red. The validation scores are shown as green lines, with uncertainty bands in light green. The estimator in **(a)** is underfitting, as both the training and validation tend to a value less than one. The estimator in **(b)** is a good fit, and could benefit from more data.

where $\hat{y}_i$ is the predicted value of the ith sample, y_i is the true value and $\bar{y} = \frac{1}{N}\sum_{i=0}^{N-1} y_i$ for N samples. This is the score used for cross validation in this project.

Cross-validation can be used to plot *learning curves*, which are used to find out whether the estimator benefits from adding more data. The learning curve plots the *training score* and *validation score*. The training score is the mean R^2-score, $m(R^2)$, for the k predictions of the training points, *i.e.* where the $k-1$ folds used for training are also used as test points. The validation score is the mean R^2 score, $m(R^2)$, for the k predictions of validation points, *i.e.* the one fold that is not a part of the training data. The variances of the R^2-scores, $\mathbb{V}(R^2)$, are plotted as error bands. The test- and validation scores are used to find out if the model is *overfitting* or *underfitting*. *Overfitting* means that the model is a perfect fit to the training data, but predicts poorly for test data because it is not general. *Underfitting* occurs when the model is not able to capture the underlying structure of the data. Ideally, the training score should be 1, and the validation score should approach 1 with the addition of data.

Examples of learning curves are shown in Fig. 4.8 as a function of training points, for two machine learning techniques called Naive Bayes and Support Vector Machine on an example problem. In **(a)** both the training score and cross-validation score tend to a value below 1, which indicates underfitting. This model will not benefit from more data. The example in **(b)** shows a training score of approximately 1, and a cross validation score that converges towards 1. This model could benefit from more data.

4.4.3 Relative Deviance

In this project predictions are compared using the *relative deviance*. For true values y_i and values predicted by the estimator $\hat{y}_i$ the relative deviance is given by

$$\varepsilon_i = \frac{y_i - \hat{y}_i}{y_i}. \tag{4.55}$$

In this project the target values have a very wide span, ranging from about 10^{-30} fb to 10^9 fb. The data is therefore divided into decades, meaning one set contains $\sigma \in [10^i, 10^{i+1}]$, where σ is the cross section. Then a distribution over the relative deviances within each decade is found, with a mean value, $m(\varepsilon_i)$, and standard deviation, $\sigma_{std}(\varepsilon_i)$. These are plotted as a function of i, and denoted

$$m(\varepsilon_i) = m\Big(\frac{y_i - \hat{y}_i}{y_i}\Big), \tag{4.56}$$

$$\sigma^2(\varepsilon_i) = \mathbb{V}\Big(\frac{y_i - \hat{y}_i}{y_i}\Big), \tag{4.57}$$

$$\sigma_{std}(\varepsilon_i) = \sqrt{\sigma^2(\varepsilon)}. \tag{4.58}$$

Chapter 5

Evaluating Cross Sections with Gaussian Processes

This chapter is dedicated to evaluating cross sections for squark pair production with Gaussian processes. A Benchmark Gaussian process estimator for the processes $\tilde{d}_L\tilde{d}_L$ and $\tilde{d}_L\tilde{u}_L$ is considered, and compared to estimators where alterations are made in the data set, in the kernel and in the features. Then the effect of scaling Gaussian processes to larger datasets using Distributed Gaussian processes is investigated.

5.1 Data Generation

In this section the generation of MSSM-24 training and test data is discussed, following closely the discussion in [23].

Sampling of Data

An MPI parallelized script generates a sample point in parameter space by drawing random values from the distributions in Tab. 5.1. The table contains log and flat priors, discussed below, and a combination of these is used to properly cover parameter space. When a parameter point has been sampled, it is run through the program `softpoint.x` which calculates its SUSY spectrum using the `Softsusy 3.6.2`-package [24]. The spectrum is then written to a `slha`-file that is given as input to `Prospino 2.1`, which subsequently calculates the LO and NLO cross sections according to the method outlined in Section 3.4.1. The relevant masses and NLO cross sections are harvested to `.dat`-files, which are used by the Gaussian processes.

Parameter	Log prior range	Flat prior range
M_1	[0,100,4000]	[0,4000]
M_2	[0,100,4000]	[0,4000]
M_3	[0,100,4000]	[0,4000]
A_t	[-4000, -100, 100, 4000]	[-4000, 4000]
A_b	[-4000, -100, 100, 4000]	[-4000, 4000]
A_τ	[-4000, -100, 100, 4000]	[-4000, 4000]
μ	[-4000, -100, 100, 4000]	[-4000, 4000]
m_A^{pole}	[0,100,4000]	[0,4000]
$\tan\beta$	[2, 60]	[2, 60]
m_{L_1}	[0, 100, 4000]	[0, 4000]
m_{L_2}	[0, 100, 4000]	[0, 4000]
m_{L_3}	[0, 100, 4000]	[0, 4000]
m_{e_1}	[0, 100, 4000]	[0, 4000]
m_{e_2}	[0, 100, 4000]	[0, 4000]
m_{e_3}	[0, 100, 4000]	[0, 4000]
m_{Q_1}	[0, 100, 4000]	[0, 4000]
m_{Q_2}	[0, 100, 4000]	[0, 4000]
m_{Q_3}	[0, 100, 4000]	[0, 4000]
m_{u_1}	[0, 100, 4000]	[0, 4000]
m_{u_2}	[0, 100, 4000]	[0, 4000]
m_{u_3}	[0, 100, 4000]	[0, 4000]
m_{d_1}	[0, 100, 4000]	[0, 4000]
m_{d_2}	[0, 100, 4000]	[0, 4000]
m_{d_3}	[0, 100, 4000]	[0, 4000]

Table 5.1: Table showing the sampling intervals used for the parameters when sampling the MSSM-24 model, where the soft breaking scale is set to $Q = 1$ TeV. The log priors have three and four limit values, which are of the form [start_flat, start_log, end_log] and [start_log, start_flat, start_log, end_log]. All values in GeV except $\tan\beta$ which is unitless. Table from [23].

Priors

To get a reasonable distribution in parameter space it is necessary to use an objective prior distribution of parameters. This means that priors are assigned according to a set of principles of how information should be encoded in a probability distribution. More details on objective priors can be found in [2].

The first of these principles is the *transformation group invariance*, which states that the probability distribution should be invariant under any transformation that is considered irrelevant to the problem. In other words, the probability distribution function, P, should satisfy

$$P(x|I)dx = P(x+a|I)d(x+a) \qquad \Rightarrow \qquad P(x|I) = P(x+a|I), \tag{5.1}$$

where a is a translation in input space. This is often referred to as a uniform or *flat prior*. Invariance under scale transformations, which are transformations that introduce a scale to the problem $m \to m' = cm$, requires

$$P(m|I)dm = P(cm|I)cdm, \tag{5.2}$$

which is satisfied if $P(m|I) \propto 1/m$. This corresponds to $P(\log m|I)$ being flat, and this prior is therefore called the *log prior*.

The flat prior covers the edges of parameter space well, while a log prior covers the innermost points[1]. Therefore, a combination of the log and flat priors is used in order to properly cover parameter space. To avoid divergence of the log prior close to zero this region is covered by a flat prior. The limits on the priors are `[start, flat_start, flat_end, end]` for priors that include negative values, and `[flat_start, start, end]` for priors with only positive values. An illustration of a prior with positive values is shown in Fig. 5.1, where a flat distribution is used close to $x = 0$ to avoid divergences.

The weak scale MSSM model used in this project, MSSM-24, requires a soft breaking scale Q. This scale is set to $Q = 1$ TeV. It is worth noting that the parameter space for the cross sections is significantly reduced from that of the MSSM-24. The cross sections depend on the values $m_{\tilde{g}}$, $m_{\tilde{q}}$, $\tilde{g}_s$, $\hat{g}_s$ and s. Since only first and second generation squarks are considered, this contributes with 8 masses [2] which combined with the 4 other parameters reduces the parameter space to 12 dimensions. The parameter space is therefore better sampled than it would appear from the 24 parameters in MSSM-24.

Data Quality

Data quality plots are used to ensure that the data is properly distributed in parameter space. In Fig. 5.2 distributions for $m_{\tilde{g}}$, $m_{\tilde{d}_L}$ and $m_{\tilde{u}_L}$ are shown, as

[1] The points close to 0.

[2] 4 quarks with a pair of lefthanded and righthanded squarks each.

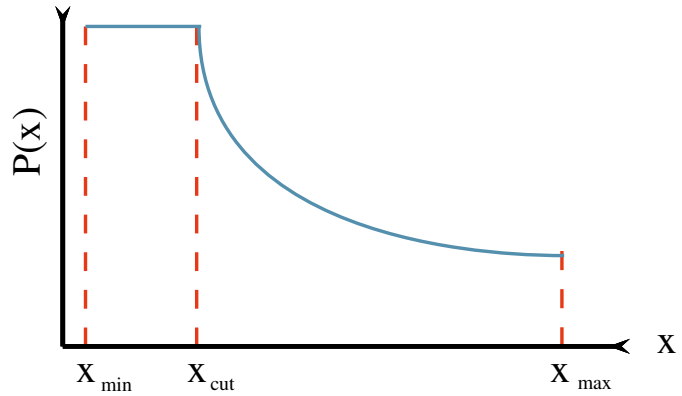


Figure 5.1: Illustration of the log prior distribution $\pi(x)$. Around $x = 0$ the prior would blow up, so at a limit x_{cut} a flat prior is used.

the benchmark processes are the production of $\tilde{u}_L\tilde{d}_L$ and $\tilde{d}_L\tilde{d}_L$. The scatter plots use 40 000 points. All parts of the parameter space seem to be covered, although the density of points is higher for small masses. This could affect predictions with few training points.

The panel in Fig. 5.2b shows a scatter plot of $m_{\tilde{d}_L}$ versus $m_{\tilde{u}_L}$. The plot is almost linear, which comes from the mass splittings of the MSSM. Same-generation left-handed squarks come in $SU(2)$-doublets, and their masses are predominantly determined by *one* mass parameter, namely Q_i for generation i. Right-handed squarks, on the other hand, get their masses from different parameters $m^2_{\tilde{d}_i}$ and $m^2_{\tilde{u}_i}$, and so are independent of each other. The mass splitting between same-generation left-handed squarks, *e.g.* $\tilde{d}_L$ and $\tilde{u}_L$, was given in Sec. 1.4.5,

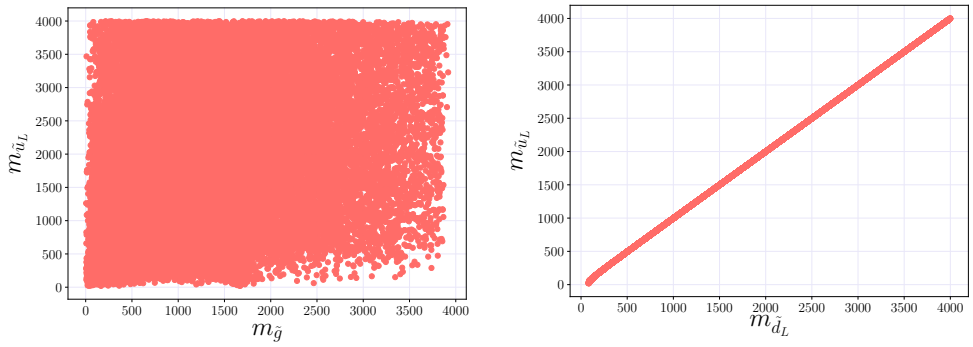
$$m^2_{\tilde{d}_L} - m^2_{\tilde{u}_L} = -\cos 2\beta \; m^2_W,$$

where $-\cos 2\beta \; m^2_W$ is relatively small ($\leq$ 15 GeV). It is therefore a possibility that training on processes with same-generation left-handed squarks, $\tilde{u}_L\tilde{d}_L$, $\tilde{s}_L\tilde{c}_L$, will effectively be training on a single squark mass, such as $m_{\tilde{d}_L}$, in stead of two, such as $m_{\tilde{d}_L}, m_{\tilde{u}_L}$.

Noise

The data contains some noise that originates in the `Prospino 2.1` calculation. In a parameter point chosen at random the relative error ϵ_i has a variance of $\sigma^2_\epsilon = 4.0 \cdot 10^{-6}$. This error fluctuates little between parameter points, so it is considered a good approximation for the order of magnitude of errors in all points. The goal is now to incorporate this information in the Gaussian process. For that purpose the following relation is considered,

$$Y_i = y_i^{true} + \Delta y_i^{true} = y_i^{true}(1 + \epsilon_i), \tag{5.3}$$



(a) $m_{\tilde{g}}$-$m_{\tilde{u}_L}$ **(b)** $m_{\tilde{d}_L}$-$m_{\tilde{u}_L}$

(c) $m_{\tilde{g}}$-$m_{\tilde{d}_L}$

Figure 5.2: Data quality plots of the distribution of mass parameters $m_{\tilde{d}_L}$, $m_{\tilde{u}_L}$ and $m_{\tilde{g}}$ for 40 000 points. In **(c)** different orders of magnitude of the cross sections for $\tilde{d}_L\tilde{d}_L$ are shown in different colors, indicating that there are few points in the very high cross sections $\sigma \propto 10^4 - 10^6$, and that lower cross sections are more spread across the gluino mass spectrum.

where Y_i are cross sections provided by Prospino, y_i^{true} are true cross sections and Δy_i^{true} is the error in cross sections. The relative error from the calculation follows a Gaussian distribution $\epsilon_i \sim \mathcal{N}(0, \sigma_\epsilon^2)$.

The target values will in fact be the logarithm of the cross sections,

$$\begin{aligned} X_i &= \log_{10} Y_i \\ &= \log 10(y_i^{true}) + \log_{10}(1 + \epsilon_i). \end{aligned} \tag{5.4}$$

For small ϵ_i he second term in Eq. (5.4) can be expanded in a Taylor series

$$\log_{10}(1 + \epsilon_i) = \frac{\epsilon_i}{\log 10} - \frac{\epsilon_i^2}{\log 100} + \mathcal{O}(\epsilon_i^3). \tag{5.5}$$

The first term in Eq. (5.5) is dominant, and for small enough ϵ_i the following approximation can be made

$$\log_{10}(1 + \epsilon_i) \approx \frac{1}{\log 10}\ \epsilon_i. \tag{5.6}$$

The targets in Eq. (5.4) can now be written as

$$X_i \approx \log_{10}(y_i^{true}) + \frac{1}{\log 10}\ \epsilon_i = \log_{10}(y_i^{true}) + \varepsilon_i \tag{5.7}$$

where the total error, ε_i, now follows a Gaussian distribution $\mathcal{N}(0, \sigma_n^2)$. For a random variable X multiplied by a number c the expectation value and variance are given by

$$\mathbb{E}[cX] = x \cdot \mathbb{E}[X], \tag{5.8}$$

$$\mathbb{V}[cX] = c^2 \cdot \mathbb{V}[X], \tag{5.9}$$

so the variance σ_n^2 is given by

$$\sigma_n^2 = \frac{\sigma_\epsilon^2}{(\log 10)^2}. \tag{5.10}$$

Equation (5.7) is now on the form of the Gaussian noise model discussed in Sec. 4.3. The distribution of the relative error ϵ_i has variance $\sigma_\epsilon^2 = 4.0 \cdot 10^{-6}$, so the variance of the Gaussian distributed noise should be

$$\sigma_n^2 = \frac{4.0 \cdot 10^{-6}}{(\log 10)^2} \simeq 7.544 \cdot 10^{-7}.$$

The noise term predicted by the Gaussian process estimator, $\varepsilon_{WK} \sim \mathcal{N}(0, \sigma_n^2)$, should therefore have a variance of the order $\sigma_n^2 \sim \mathcal{O}(10^{-7} - 10^{-6})$.

5.2 Dataset Transformations

The plot in Fig. 5.2c indicates that cross sections, in particular of the order $\mathcal{O}(1\text{ fb})$ and lower, are very spread as a function of $m_{\tilde{g}}$. The spread is also evident in the upper left panel of Fig. 5.3, where the cross section for the production of $\tilde{d}_L\tilde{d}_L$, σ, is plotted as a function of the gluino mass. The upper left panel shows σ as a function of the squark mass $m_{\tilde{q}}$, which is also quite spread. This section will be devoted to reducing the spread caused by the gluino mass dependency.

Scaling Functions

As discussed in Sec. 2.4, the partonic cross sections can be written in terms of scaling functions f, see Eq. (3.18). The scaling functions are the different contributions to the cross section, as explained in Sec. 2.4. The total cross section only differs from the partonic cross section by an integral over parton distribution functions, so the mass dependencies in Eq. (3.18) are relevant for the total cross sections as well.

The main contributions to the cross section come from the scaling functions at the threshold energy given in Eq. (3.20), which makes it possible to remove some of the complexity of the function. Note that all terms are proportional to f^B_{qq} ($f^B_{q'q}$), which is again proportional to $m^2_{\tilde{q}}m^2_{\tilde{g}}$. The partonic cross section is proportional to $\sigma \propto 1/m^2$, and $m^2 = m^2_{\tilde{q}}$ for squark pair production, so this dependency is automatically cancelled. However, the following transformation can be made

$$\sigma \to \sigma_{m_{\tilde{g}}} = \frac{\sigma}{m^2_{\tilde{g}}}, \tag{5.11}$$

reducing the gluino mass dependency. The lower panels in Fig. 5.3 show $\sigma_{m_{\tilde{g}}}$ as a function of $m_{\tilde{g}}$ and $m_{\tilde{q}}$. The spread as a function of the squark mass is much smaller, and for high cross sections the shape of $\sigma_{m_{\tilde{g}}}$ as a function of squark and gluino mass are very similar, which may contribute to finding a kernel that fits well in both dimensions. The target value in this project is therefore the logarithm of $\sigma_{m_{\tilde{g}}}$.

In the threshold region the partonic version of $\sigma_{m_{\tilde{g}}}$ is given by

$$\hat{\sigma}_{ij,m_{\tilde{g}}} = \frac{8\pi\beta\alpha_s^2(Q^2)}{27(m^2_{\tilde{q}}+m^2_{\tilde{g}})^2}\Bigg\{1 + 4\pi\alpha_s(Q^2)\Bigg[\frac{1}{24\beta} + \frac{2}{3\pi^2}\log^2(8\beta^2) - \frac{7}{2\pi^2}\log(8\beta^2) - \frac{2}{3\pi^2}\log(8\beta^2)\log\left(\frac{Q^2}{m^2}\right)\Bigg]\Bigg\}. \tag{5.12}$$

Another possibility is to further reduce the mass dependecy, by defining σ_{fac} as

$$\sigma_{fac} = \sigma\frac{(m^2_{\tilde{g}}+m^2_{\tilde{q}})^2}{m^2_{\tilde{g}}}. \tag{5.13}$$

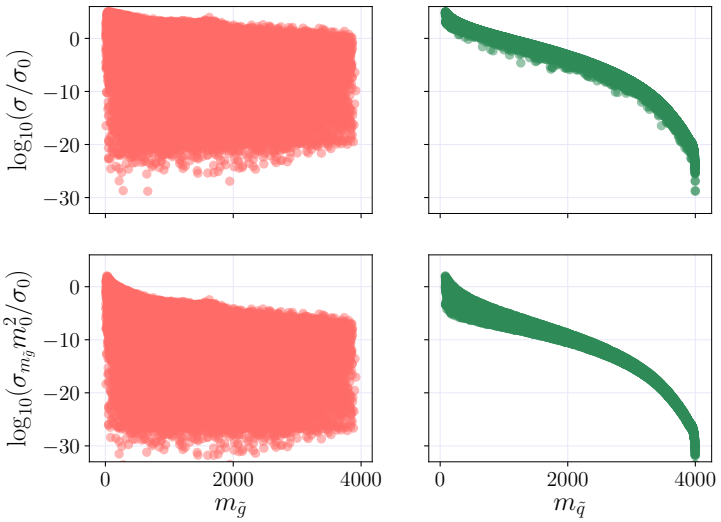


Figure 5.3: The cross section, σ, and the modified cross section, $\sigma_{m_{\tilde{g}}}$, as functions of the gluino mass $m_{\tilde{g}}$ and squark mass $m_{\tilde{q}} = m_{\tilde{d}_L}$ for the production of $\tilde{d}_L\tilde{d}_L$. The functions have less spread when some of the mass dependency is removed.

This transformation provides an even smoother function, but has been left as further work with GPs.

5.3 Learning the Gaussian Process

In this section a Gaussian process estimator is learned with benchmark settings, and a selected set of modifications to the benchmark are introduced. The quality of estimators are quantified in plots of the relative deviance distributions defined in Sec. 4.4.3.

5.3.1 The Benchmark

The benchmark processes are the pair production of $\tilde{d}_L\tilde{d}_L$ and $\tilde{d}_L\tilde{u}_L$. The benchmark settings (BM) are

- A GP estimator with 2000 training points
- 20 000 test points
- Features $m_{\tilde{g}}, m_{\tilde{d}_L}$ $(m_{\tilde{g}}, m_{\tilde{d}_L}, m_{\tilde{u}_L})$ for $\tilde{d}_L\tilde{d}_L$ $(\tilde{d}_L\tilde{u}_L)$ production

	C	$\ell_{m_{\tilde{g}}}$	$\ell_{m_{\tilde{d}_L}}$	$\ell_{\bar{m}}$	α
BM	2981	5470	2190		0.47
No outliers	9702	5740	215		0.00372
$\sigma > 10^{-16}$ fb	515	1170	998		0.0036
$\bar{m}$	1095	1190	200	846	0.0000119
Matern	1000	30200	8600		0.462

Table 5.2: Optimal kernel parameters for different settings for $\tilde{d}_L\tilde{d}_L$.

- The exponential squared kernel (ES) with a white noise term,

$$k(\mathbf{x}_i, \mathbf{x}_j) \;=\; \sigma_f^2 \exp\big(-\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)^T M(\mathbf{x} - \mathbf{x}')\big) + \alpha\delta_{ij}, \tag{5.14}$$

  where $M = \text{diag}(\boldsymbol{\ell})^{-2}$, and α is the variance of the distribution of white noise $\varepsilon_{WK} \sim \mathcal{N}(0, \alpha)$.

The kernel is implemented in `scikit-learn` in the following way

```
kernel_BM = C(constant_value=10,
    constant_value_bounds=(1e-3, 1e4)) * RBF(length_scale =
    np.array([1000, 1000]), length_scale_bounds=(1, 1e6)) +
    WhiteKernel(noise_level=1, noise_level_bounds=(2e-10,1e2))
```

where the features are `(m_gluino, m_dL)` for $\tilde{d}_L\tilde{d}_L$ and `(m_gluino, m_dL, m_uL)` for $\tilde{d}_L\tilde{u}_L$. Note that the length scale of the RBF is given a a vector of the same dimension as the feature vector $\mathbf{x}, \boldsymbol{\ell} \in \mathbb{R}^D$ [3]. This is in case the different features have different characteristic length scales, as they appear to have from the lower panels in Fig. 5.3. The target values are $\log_{10}\sigma_{m_{\tilde{g}}}$, where $\sigma_{m_{\tilde{g}}}$ was defined in Sec. 5.2.

$m(\varepsilon_i)$ and std(ϵ_i), as defined in Eq. (4.56) - (4.57), of the decades are calculated and plotted for the BM settings in Fig. 5.4 for $\tilde{d}_L\tilde{d}_L$. For $\tilde{d}_L\tilde{u}_L$ the results are very similar, and therefore not shown here. The optimal kernel parameters found by the GP are found in Tab. 5.2 - 5.3, while computation times on a laptop are found in Tab. 5.4. The predicted noise level variances, α, are very high for both processes, at $\alpha = 0.47$ for $\tilde{d}_L\tilde{d}_L$ and $\alpha = 0.593$ for $\tilde{d}_L\tilde{u}_L$, which is far from the value $\alpha \sim 7.544 \cdot 10^{-7}$ predicted in Sec. 5.1. In addition, the prediction is very unstable, with the estimator predicting values that are too small for low cross sections.

[3] This example is for $\tilde{d}_L\tilde{d}_L$ production, for $\tilde{d}_L\tilde{u}_L$ production $D = 3$ as $m_{\tilde{u}_L}$ is inluded as well.

	C	$\ell_{m_{\tilde{g}}}$	$\ell_{m_{\tilde{d}_L}}$	$\ell_{m_{\tilde{u}_L}}$	$\ell_{\bar{m}}$	α
BM	2490	5870	4000	2220		0.593
No outliers	6400	4420	3100	240		0.00348
$\sigma > 10^{-16}$ fb	10000	1540	2900	2150		0.0031
$\bar{m}$	3806	1340	3590	251	748	0.0000119
Matern	1000	31100	1000000	8260		0.585

Table 5.3: Optimal kernel parameters for different settings for $\tilde{d}_L\tilde{u}_L$.

	Time $\tilde{d}_L\tilde{d}_L$	Time $\tilde{d}_L\tilde{u}_L$
BM	00:07:48	00:08:40
Outliers	00:08:42	00:11:20
Cut	00:07:24	00:11:07
Features	00:11:20	00:16:37
Kernel	00:07:28	00:13:05

Table 5.4: Computation times for GP with 2000 training points and 20 000 test points on a laptop with 4 cores.

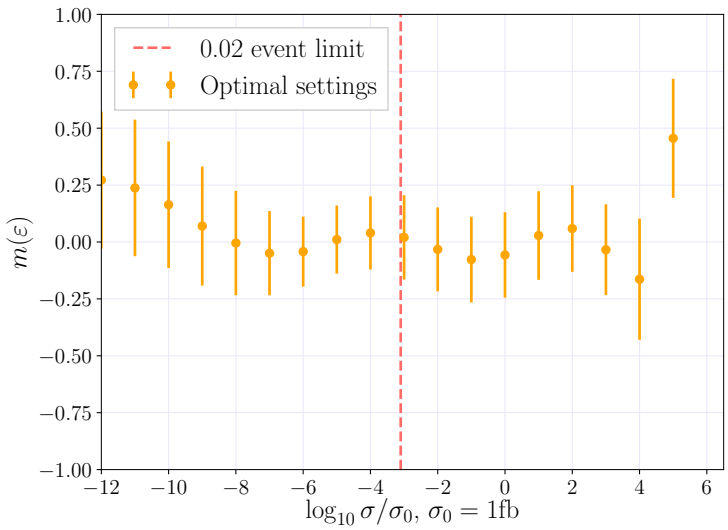


Figure 5.4: The mean and standard deviation of the relative deviance distributions, $m(\varepsilon_i)$ and $\sigma_{std}(\varepsilon_i)$, as a function of $i = \log_{10}\sigma/\sigma_0$, for the process $\tilde{d}_L\tilde{d}_L$ with benchmark settings. 2000 training points and 20 000 test points were used on a regular GP, with the final kernel parameters $C = 2981$, $\ell_{m_{\tilde{g}}} = 5470$, $\ell_{m_{\tilde{d}_L}} = 2190$ and $\alpha = 0.47$. Features are the physical masses $m_{\tilde{g}}$ and $m_{\tilde{d}_L}$.

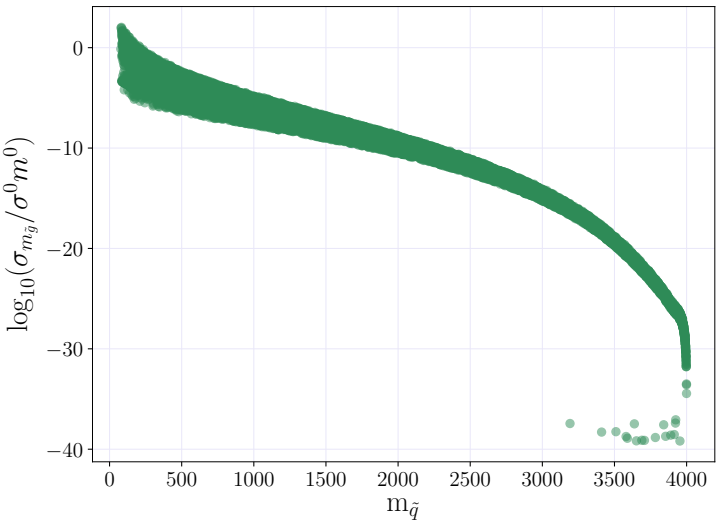


Figure 5.5: Plot of $\log(\sigma_{m_{\tilde{g}}})$ for $\tilde{d}_L\tilde{d}_L$ as a function of $m_{\tilde{d}_L}$, where outliers are included. The outliers are originally $\sigma = 0$ fb, but are set to $\sigma = 10^{-32}$ fb so as to avoid infinities.

5.3.2 Outliers

Calculations in `Prospino 2.1` set some NLO terms to zero as discussed in Sec. 3.4.1. These points become outliers in the dataset. The outliers can be seen as a cluster of points well below the other cross sections for large masses in Fig. 5.5. These points have zero cross sections, which are set to 10^{-32} fb in the calculation to avoid divergences.

A GP estimator with the BM settings is trained on a dataset where outliers have been removed. Removing outliers makes the prediction much better for small cross sections, but also stabilizes the prediction for larger values. This can be seen in Fig. 5.6a where BM estimators with and without outliers are compared. The predicted noise levels are significantly reduced with the removal of outliers for both processes, from $\alpha = 0.47$ to $\alpha = 0.00372$ for $\tilde{d}_L\tilde{d}_L$ and from $\alpha = 0.593$ to $\alpha = 0.00348$ for $\tilde{d}_L\tilde{u}_L$. This implies that including the outliers leads the GP to underfit, which means that much of the signal is considered noise.

5.3.3 Cuts on Cross Sections

Smooth functions are easier to fit with Gaussian processes, and the target values are very steep functions of large squark masses. This can be seen from the righthand panels in Fig. 5.3. In addition, small target values comprise the regions

[4]Cross sections with lower values than this predict less than 0.02 events, and are therefore less interesting in this project.

with the most spread as a function of the gluino mass. Since the limit for 0.02 events is at $\sigma = 10^{-3}$ fb [4], a lower cut is set at $\sigma_{cut} = 10^{-16}$ fb. The cut excludes all cross sections lower than σ_{cut} from both training and testing.

The resulting distributions of the relative deviance for $\tilde{d}_L\tilde{d}_L$ are shown in Fig. 5.6b, and the optimal kernel parameters are found in Tab. 5.2 - 5.3. Noise levels are further reduced from the case where outliers are removed, with the variance going from $\alpha = 0.00372$ to $\alpha = 0.00336$ for $\tilde{d}_L\tilde{d}_L$ and from $\alpha = 0.00348$ to $\alpha = 0.0031$ for $\tilde{d}_L\tilde{u}_L$. The estimated noise level seems to be approaching the expected theoretical value, indicating that the exclusion of low cross sections improves the prediction.

The relative deviance of the predictions of the largest targets is significantly improved from the cases of the BM with and without outliers. Training and testing only on high values renders a very stable prediction for all (included) orders of magnitude.

5.3.4 Features

`Prospino 2.1` calculates the NLO cross section for the mean of the squark masses, $\bar{m}$, and uses this to find the K-factor. The K-factor is then multplied with the LO cross sections for non-degenerate squark masses to find the individual NLO terms, as discussed in Sec. 3.4.1. Adding the mean squark mass as a feature could therefore improve the prediction. The features used in this section are

```
features_dLdL = (m_gluino, m_dL, m_mean)
features_dLuL = (m_gluino, m_dL, m_uL, m_mean)
```

The optimal kernel values are found in Tab. 5.2 - 5.3. Adding the mean mass as a feature reduces the estimated noise level considerably, and estimates the same level, $\alpha = 1.19 \cdot 10^{-5}$, for $\tilde{d}_L\tilde{d}_L$ and $\tilde{d}_L\tilde{u}_L$. This is an indication that the prediction is improved, considering the two processes should have approximately the same level of noise, as discussed in Sec. 5.1.

Further, the estimator predicts a shorter length scale for $\bar{m}$ than for $m_{\tilde{g}}$. This implies that there is a higher dependence on the mean mass than the gluino mass, since GPs attribute large length scales to irrelevant features. The resulting distributions for $\tilde{d}_L\tilde{d}_L$ are shown in Fig. 5.6c and compared to the BM. With some exceptions at $\log_{10}\sigma/\sigma_0 \in [2, 4]$, where the variances are very large, adding $\bar{m}$ as a feature gives a mean of almost zero and very small variances of the relative deviances for cross sections over the 0.02 event limit.

5.3.5 Kernel

The exponential squared kernel is very smooth, while the Matérn kernel has a hyperparameter ν to control its smoothness. It is sometimes argued that this makes Matérn a better kernel for physical processes, as mentioned in Sec. 4.2.2. In `scikit-learn` the hyperparameter ν is not optimized, so it must be determined beforehand. In this section the Matérn kernel with $\nu = \frac{3}{2}$ is compared to the BM RBF kernel, and the resulting error distributions for $\tilde{d}_L\tilde{d}_L$ are found in Fig. 5.4d.

The hyperparameter ν is set to $\frac{3}{2}$ as this is one of the values for which covariances are quick to calculate. For values not in $\nu = [\frac{1}{2}, \frac{3}{2}, \frac{5}{2}, \infty]$ `scikit-learn` evaluates Bessel functions explicitly, which increases the computational cost by up to a factor 10.

The predictions are somewhat more stable for high cross sections than with the RBF kernel. For low cross sections the error distributions have larger variances, but as this is currently not the region of interest the Matérn kernel is considered a better fit than the RBF. As can be seen from the optimal kernel values in Tab. 5.2 - 5.3, the Matérn kernel predicts a slightly lower noise level than the RBF. The noise variances go from $\alpha = 0.47$ with RBF to $\alpha = 0.462$ with Matérn for $\tilde{d}_L\tilde{d}_L$, and from $\alpha = 0.593$ with RBF to $\alpha = 0.585$ with Matérn for $\tilde{d}_L\tilde{u}_L$.

5.3.6 Cummulative Settings

A combination of the improved settings from the foregoing sections is used in this section. To sum up, the cummulative settings from the preceding sections are as follows

- A GP estimator with 2000 training points
- 20 000 test points
- Features $[m_{\tilde{g}}, m_{\tilde{d}_L}, \bar{m}]$ ($[m_{\tilde{g}}, m_{\tilde{d}_L}, m_{\tilde{u}_L}, \bar{m}]$) for $\tilde{d}_L\tilde{d}_L$ ($\tilde{d}_L\tilde{u}_L$) production
- The Matérn kernel with $\nu = 1.5$ and a white noise term

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \exp\left(1 + \sqrt{3}\big[(\mathbf{x}_i - \mathbf{x}_j)^T M(\mathbf{x}_i - \mathbf{x}_j)\big]^{1/2}\right) \tag{5.15}$$

$$\times \exp\left(\sqrt{3}\big[(\mathbf{x}_i - \mathbf{x}_j)^T M(\mathbf{x}_i - \mathbf{x}_j)\big]^{1/2}\right) + \sigma_n^2\delta_{ij}, \tag{5.16}$$

  where $M = \mathrm{diag}(\boldsymbol{\ell})^{-2}$.

- A cut on the cross sections $\sigma > \sigma_{cut} = 10^{-16}$ fb[5]

[5]Effenctively removing the outliers $\sigma = 0 < 10^{-16}$ fb.

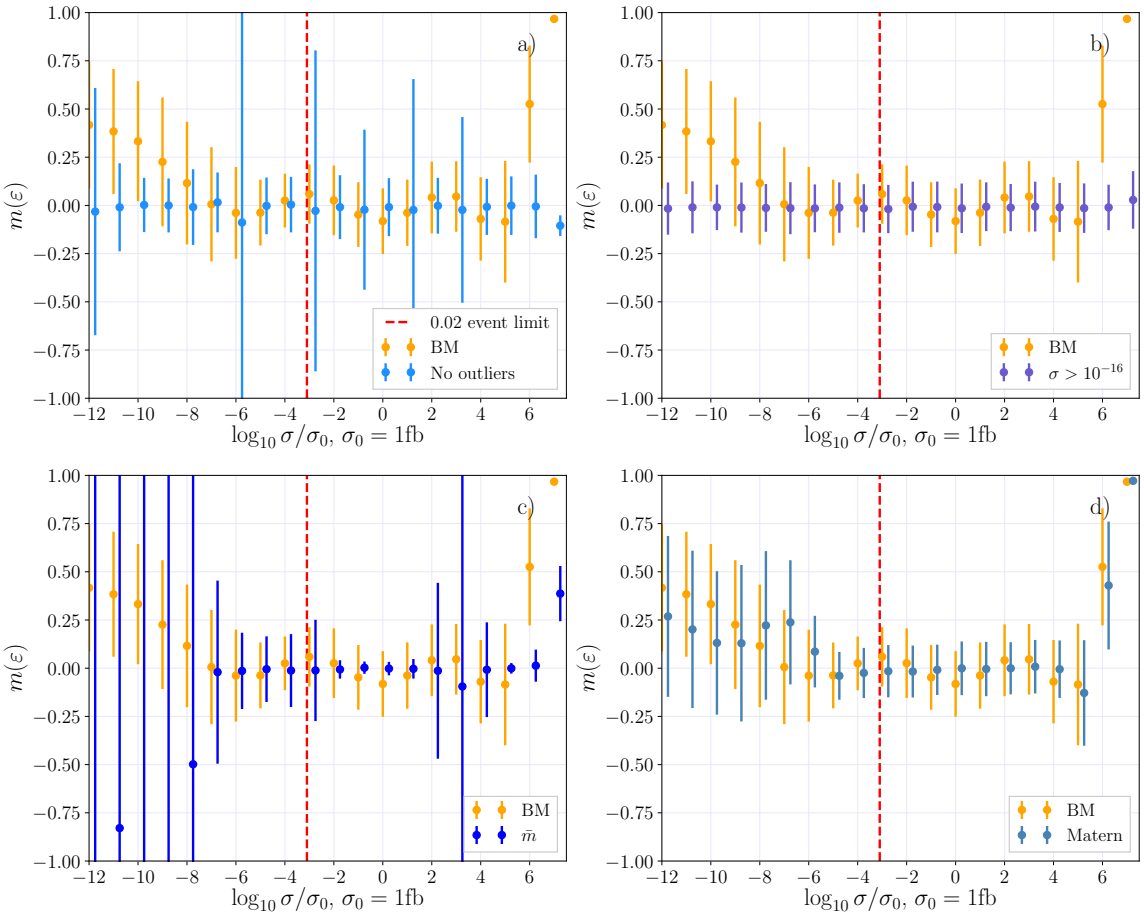


Figure 5.6: The mean and standard deviation of the relative deviance distributions, $m(\varepsilon_i)$ and $\sigma_{std}(\varepsilon_i)$, as a function of $i = \log_{10}\sigma/\sigma_0$, for the process $\tilde{d}_L\tilde{d}_L$ with **a)** benchmark settings (orange) and removed outliers (blue); **b)** benchmark settings (orange) and a lower cut on cross sections (blue); **c)** benchmark settings (orange) and the added feature $\bar{m}$ (blue); **d)** the benchmark settings (orange) and the Matérn kernel (blue). Benchmark settings are abbreviated BM.

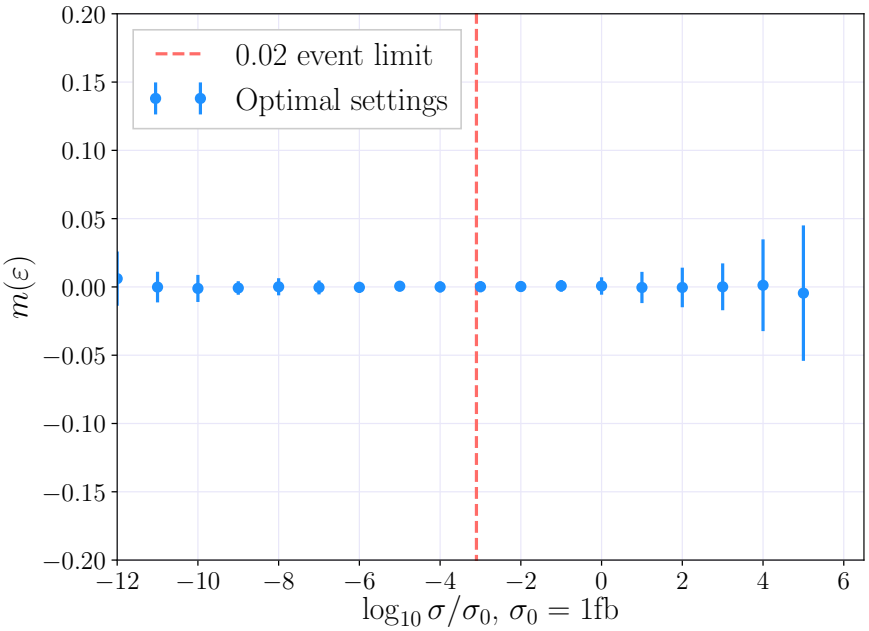


Figure 5.7: The distribution of the relative deviance ε as a function of the logarithm of the normalized cross section for $\tilde{d}_L\tilde{d}_L$ with the optimal settings; a single GP with 2000 training points that uses the Matérn kernel with $\nu = 1.5$. Outliers are removed and a lower cut at $\sigma = 10^{-16}$ fb is introduced. The features are $m_{\tilde{g}}$, $m_{\tilde{d}_L}$ and $\bar{m}$, and the target is $\sigma_{m_{\tilde{g}}}$. All error distibutions have σ_{std} smaller than 5%.

The resulting relative deviance distributions are found in Fig. 5.7. With the new settings the prediction is very good, with all standard deviations $\sigma_{std}(\varepsilon_i)$ less than 5%. The computation with the cummulative settings for 2000 training points takes 00:09:30 for $\tilde{d}_L\tilde{d}_L$ and 00:10:28 for $\tilde{d}_L\tilde{u}_L$ on a regular laptop with 4 cores.

Noise

The noise level is known to some degree, as discussed in Sec. 5.1, so a brute-force approach of explicitly adding noise to the covariance is here tested with the cummulative settings. In `scikit-learn` an option to letting the `WhiteKernel` estimate the level of noise is to specify the noise by passing it as `alpha` to the Gaussian process regressor function

```
gp = GaussianProcessRegressor(kernel=kernel, alpha=7.544e-7)
```

For the cummulative settings the `WhiteKernel` predicts values close to α_{fix},

	C	$\ell_{m_{\tilde{g}}}$	$\ell_{m_{\tilde{d}_L}}$	$\ell_{m_{\tilde{u}_L}}$	$\ell_{\bar{m}}$	α
$\tilde{d}_L\tilde{d}_L$	750	30500	17400		74800	$1 \cdot 10^{-5}$
$\tilde{d}_L\tilde{d}_L$	1000	28300	18300		69900	$7.544 \cdot 10^{-7}$ (fixed)
$\tilde{d}_L\tilde{u}_L$	1000	30200	79600	18500	89000	$5.63 \cdot 10^{-6}$
$\tilde{d}_L\tilde{u}_L$	1000	29100	66200	18300	76000	$7.544 \cdot 10^{-7}$ (fixed)

Table 5.5: Kernel parameters for the optimal settings with the noise level estimated by the GP, and given as a constant $\alpha = 7.544 \cdot 10^{-7}$.

with $\alpha = 10^{-5}$ for $\tilde{d}_L\tilde{d}_L$ and $5.63 \cdot 10^{-6}$ for $\tilde{d}_L\tilde{u}_L$. The remaining kernel parameters therefore hardly change when α is fixed, as seen in Tab. 5.5. The prediction changes very little with the addition of α_{fix}. A marginal improvement can be seen in the standard deviations of relative deviances when $\alpha = \alpha_{fix}$ in Fig. 5.8, but the mean values are almost identical.

For calculations with few training points the computation time is not affected in any great way, but for larger datasets the removal of `WhiteKernel` from the total kernel may reduce the computation time, as it leaves one less hyperparameter to optimize. For the remainder of the project the `WhiteKernel` is used.

5.4 Distributed Gaussian Processes

A significant disadvantage of Gaussian processes is that they scale poorly with the size of the data set, as discussed in Sec. 4.3. For n training points and n^* test points, training and predicting scale as $\mathcal{O}(n^3)$ and $\mathcal{O}(n^{*2})$, respectively, giving GP training data sets a practical limit of $\mathcal{O}(10^4)$.

In [25] a method of scaling GPs to large data sets is proposed, in the form of a robust Bayesian Committee Machine (rBCM). This method is based on the product-of-experts and Bayesian Committee Machine, and has the advantage of providing an uncertainty for the prediction.

5.4.1 Product-of-Experts

Product-of-expert (PoE) models are a way of parallelising large computations. They combine several independent computations on subsets of the total data, called 'experts'. In the case of distributed Gaussian processes each expert performs GP on a subset of the training data, and the predictions on a common test set are combined.

Consider the training data set $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$, which is partitioned into M subsets $\mathcal{D}^k = \{X^k, \mathbf{y}^k\}$, $k = 1, ..., M$. The feature vectors are D-dimensional, $\mathbf{x} \in \mathbb{R}^D$. Each GP expert does learning on its training data set $\mathcal{D}^k$, then predictions

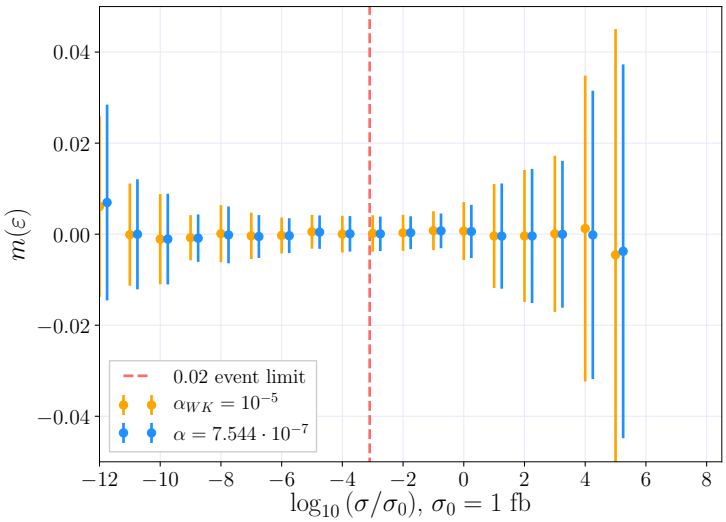
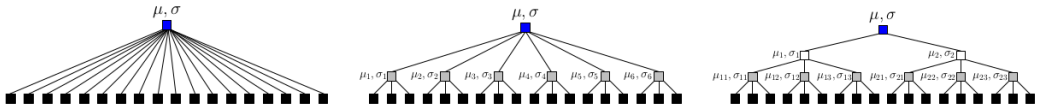


Figure 5.8: The distribution of the relative deviance ε as a function of the logarithm of the normalized cross section for $\tilde{d}_L\tilde{d}_L$ with the optimal settings. In one case the noise level α_{WK} is estimated by the GP (orange), and in the other it is given as a fixed parameter α (blue).



(a) 1-layer model. (b) 2-layer model. (c) 3-layer model.

Figure 5.9: Computational graphs of hierarchical product-of-expert models. Main computations are at the leaf nodes (GP experts in black). All other nodes recombine computations from their children nodes. The blue node at the top represents the final prediction. Figure from [25].

are combined at the parent node. This node could also be considered an expert for a PoE with several layers, see Fig. 5.9.

5.4.2 Bayesian Committee Machine

The Bayesian Comittee Machine [26] is a type of PoE where a new weighting scheme is introduced. Consider the training data set from the previous section partitioned into M sets, $\mathcal{D}^k = \{X^k, \mathbf{y}^k\}$, $k = 1, ..., M$. For simplicity a single test point, $\mathbf{x}^*$, is considered, where f^* is the unknown target value.

M estimators are trained separately on each training data set. Now let $\mathbf{D}^i = \{\mathcal{D}^1, ..., \mathcal{D}^i\}$ denote the data sets with indices smaller than or equal to i for $i = 1, ..., M$. For the ith estimator the posterior probability distribution for the

test point $\mathbf{x}^*$ is $P(f^*|\mathcal{D}^i)$. Then in general

$$P(f^*|\mathbf{D}^{i-1}, \mathcal{D}^i) \propto P(f^*)P(\mathbf{D}^{i-1}|f^*)P(\mathcal{D}^i|\mathbf{D}^{i-1}, f^*), \tag{5.17}$$

meaning that the posterior distribution for the datasets $\{\mathcal{D}^1, ..., \mathcal{D}^i\}$ is proportional to the posterior for the dataset $\mathcal{D}^i$ given the datasets $\{\mathcal{D}^1, ..., \mathcal{D}^{i-1}\}$, times the posterior of the datasets $\{\mathcal{D}^1, ..., \mathcal{D}^{i-1}\}$, multiplied by the prior $P(f^*)$. Note that this is simply a generalization of the product rule in Eq. (4.2).

Assuming the datasets $\mathbf{D}^{i-1}$ and $\mathcal{D}^i$ are independent, or at least have a very small correlation, given f^*, $\mathbf{D}^{i-1} \perp\!\!\!\perp \mathcal{D}^i \mid f^*$, the following approximation is made

$$P(\mathcal{D}^i|\mathbf{D}^{i-1}, f^*) \approx P(\mathcal{D}^i|f^*). \tag{5.18}$$

Using this approximation and applying Bayes' theorem gives

$$P(f^*|\mathbf{D}^{i-1}, \mathcal{D}^i) \approx \text{constant } \times \frac{P(f^*|\mathbf{D}^{i-1})P(f^*|\mathcal{D}^i)}{P(f^*)}. \tag{5.19}$$

The approximate posterior distribution using the entire dataset, $\mathcal{D}$, is then

$$\hat{P}(f^*|\mathcal{D}) = \text{constant } \times \frac{\prod_{i=1}^{M} P(f^*|\mathcal{D}^i)}{P(f^*)^{M-1}}, \tag{5.20}$$

where the hat over $\hat{P}$ means that the distribution is approximate. The posterior distributions of each expert are simply multiplied, and divided by the prior $M-1$ times.

5.4.3 Robust Bayesian Comittee Machine

The Bayesian Comittee Machine (BCM) can be modified to a *robust Bayesian Comittee Machine* (rBCM), which is the scheme that will be used here for distributed Gaussian processes (DGP). The M experts are assumed to be independent [25] as in Eq. (5.18), effectively block-diagonalizing the covariance matrix. The marginal likelihood from Eq. 4.52 now factorizes into the product of M individual terms because of the independence assumption. For the training data of one data set $\mathcal{D}^k = \{X^k, \mathbf{y}^k\}$ the log marginal likelihood is given by Eq. (4.52)

$$\log P(\mathbf{y}^k|X^k, \boldsymbol{\theta}) = -\frac{1}{2}\mathbf{y}^{k\,T}(K_\psi^k + \sigma_\varepsilon^2\mathbb{I})^{-1}\mathbf{y}^k - \frac{1}{2}\log|K_\psi^k + \sigma_\varepsilon^2\mathbb{I}|, \tag{5.21}$$

where K_ψ^k is the covariance matrix of the training features X^k, with the kernel hyperparameters given by $\boldsymbol{\theta} = \{\boldsymbol{\psi}, \sigma_\varepsilon^2\}$. Computing the LML now entails inverting a $n_k \times n_k$ matrix, $(\mathbf{K}_\psi^{(k)} + \sigma_\varepsilon^2\mathbf{I})$, which requires time $\mathcal{O}(n_k^3)$ and memory consumption $\mathcal{O}(n_k^2 + n_k D)$ for $\mathbf{x} \in \mathbb{R}^D$. For $n_k \ll N$, this reduces the computation time and memory use considerably, and allows for parallel computing.

The rBCM predicts a function value f^* at a corresponding test input $\mathbf{x}^*$ according to the predictive distribution

$$P(f^*|\mathbf{x}^*, \mathcal{D}) = \frac{\prod_{k=1}^{M} P_k^{\beta_k}(f^*|\mathbf{x}^*, \mathcal{D}^{(k)})}{P^{-1+\sum_k \beta_k}(f^*|\mathbf{x}^*)}, \tag{5.22}$$

where the parameters β_k control the importance of the individual experts, but also the how strong the influence of the prior is. In the article, these are chosen according to the predictive power of each expert at $\mathbf{x}^*$. More specifically, β_k is the change in differential entropy between the prior $p(f^*|\mathbf{x}^*)$ and the posterior $p(f^*|\mathbf{x}^*, \mathcal{D}^{(k)})$, which can be calculated as

$$\beta_k = \frac{1}{2}(\log \sigma_{**}^2 - \log \sigma_k^2(\mathbf{x}^*)), \tag{5.23}$$

where $\sigma_{**}^2 = k(\mathbf{x}^*, \mathbf{x}^*)$ is the prior variance of the test point, and $\sigma_k^2(\mathbf{x}^*)$ is the predictive variance of the kth expert given by Eq. 4.50.

The combined predictive mean and variance are denoted μ_*^{rbcm} and σ_*^{rbcm}, and given by

$$\mu_*^{rbcm} = (\sigma_*^{rbcm})^2 \sum_k \beta_k \sigma_k^{-2}(\mathbf{x}_*)\mu_k(\mathbf{x}_*), \tag{5.24}$$

$$(\sigma_*^{rbcm})^{-2} = \sum_{k=1}^{M} \beta_k \sigma_k^{-2}(\mathbf{x}_*) + \big(1 - \sum_{k=1}^{M} \beta_k\big)\sigma_{**}^{-2}. \tag{5.25}$$

Implementation

There is no function for an rBCM in `scikit-learn` library, so the combined predictive mean and variance in Eq. 5.24 - 5.25 were implemented in `Python`. The `scikit-learn` library's existing framework for regular Gaussian processes were used for the individual experts. The algorithm was parallelised, so that each expert can learn and predict in parallel, before being combined to the final prediction. Pseudocode for the implementation is found in Algorithm 2, which takes the training data X, $\mathbf{y}$, the initial kernel k, the noise level variance σ_n^2, the number of experts $N_{experts}$ and the test features $\mathbf{x}^*$ as input, and computes the combined predictive mean and variance, μ_*^{rbcm}, $\sigma_*^{rbcm^2}$.

For parallelisation the `scikit-learn` function `Parallel` from `joblib` was used, which runs `Python` functions as pipeline jobs. It uses the `Python` function `multiprocessing` as a backend. An example of usage with 3 parallel jobs is as follows

```
>>> from joblib import Parallel, delayed
>>> from math import sqrt
>>> Parallel(n_jobs=3)(delayed(sqrt)(i**2) for i in range(10))
[0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0]
```

where `delayed` is a simple trick to be able to create a tuple with a function-call syntax.

Data: $N_{experts}$ (number of experts), X (inputs), $\mathbf{y}$ (targets), k (initial kernel), σ_n^2 (noise level), $\mathbf{x}^*$ (test input)
Split training data into N subsets: $X_k, \mathbf{y}_k$;
for *each expert* **do**
  Fit GP to training data $X_k, \mathbf{y}_k$;
  Predict μ_*, σ_*^2 for $\mathbf{x}^*$ using GP ;
  $\sigma_{**}^2 = k(x^*, x^*)$;
end
for *each expert* **do**
  $\beta = \frac{1}{2}(\log(\sigma_{**}^2) - \log(\sigma_*^2))$;
  $(\sigma_*^{rbcm})^{-2} += \beta\sigma^{-2} + \left(\frac{1}{n_{experts}} - \beta\right)\sigma_{**}^{-2}$
end
for *each expert* **do**
  $\mu_*^{rbcm} += (\sigma_*^{rbcm})^2 \beta \sigma_*^{-2} \mu_*$
end
Result: Approximative distribution of $f_* = f(\mathbf{x}_*)$ with mean μ_*^{rbcm} and variance $(\sigma_*^{rbcm})^2$.

Algorithm 2: Pseudocode for distributed Gaussian processes on a single test point $\mathbf{x}_*$. For the fit and prediction of each GP expert Algorithm (1) is used.

5.4.4 Evaluating Cross Sections with Distributed Gaussian Processes

In this section the distributed Gaussian processes described in the previous section are applied to estimators with the benchmark settings from Sec. 5.3.1 to scale the problem to larger training sets. The prediction for a single expert with 8000 training points is compared to the combined prediction of 4 experts with 2000 training points each, both in terms of computation time and quality of prediction. Larger training sets for experts could be used, such as 4 experts with 8000 training points each, but comparing to a single expert with the same amount of data would be unfeasible as it would require an expert with $4 \times 8000 = 32000$ training points.

Adding Experts

The addition of experts with 2000 training points improves the prediction from Sec. 5.3.1 significantly, with very little increase in computational cost. In Fig. 5.10

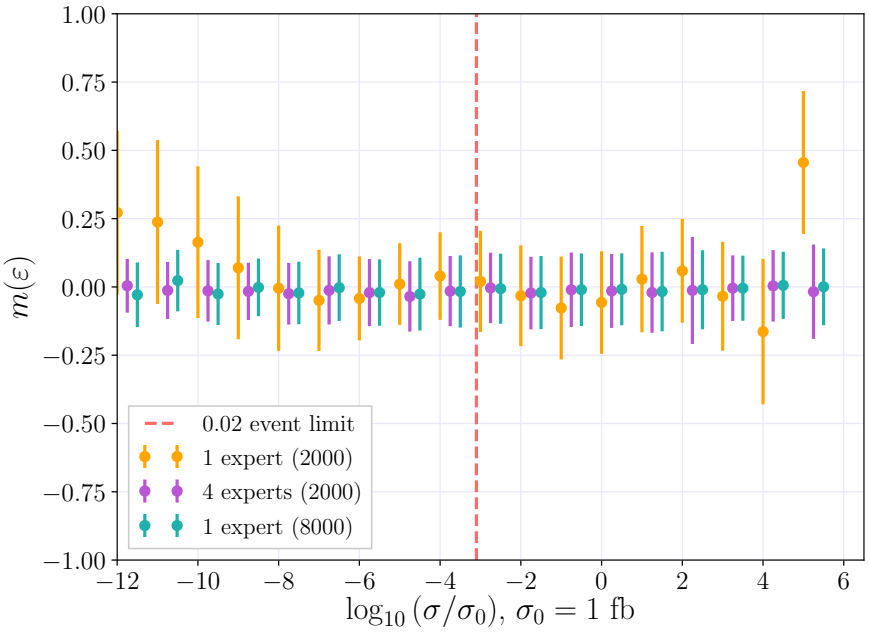


Figure 5.10: The error distributions of GP with the BM settings, using 1 (orange) and 4 (experts).

Number of experts	Points per expert	Time
1	2000	00:03:32
4	2000	00:05:46
1	8000	01:35:21

Table 5.6: Table of computation times for GP fit and prediction on Abel.

a comparison of error distributions between one expert and four experts with 2000 training points each, and one expert with 8000 training points is shown. For comparison the settings are the benchmark settings, and outliers are included in the dataset. The improvement and stability of the prediction with the addition of data is large. From the distributions of relative deviation distributions there is little difference between using four experts with 2000 training points each and using a single expert with 8000 training points. The difference in computation times, however, is very large. Four experts with 2000 points take a little under six minutes to compute, while the single expert with 8000 points takes over an hour and a half. Computation times are shown in Tab. 5.6.

5.4.5 Cross Validation

There is no `scikit-learn` function for distributed Gaussian processes, so an algorithm for k-fold cross validation as a function of experts was implemented. The algorithm uses the `scikit-learn` function `KFold` to find training and test indices for k splits of the data, and is found in the sub-library `model_selection`. For k folds it is implemented in the following way

```
from sklearn.model_selection import KFold
kf = KFold(n_splits=k, random_state=42)
```

The scoring function used in the CV is the R^2-score introduced in Sec. 4.4.2, and pseudocode for the algorithm is found in Algorithm 3.

Data: $N_{experts}$ (max number of experts), n (training points per expert), X (inputs), $\mathbf{y}$ (targets), k (number of folds for cross validation)
number of experts $\mathbf{n} = [1, ..., N_{experts}]$;
for *each number of experts* i **do**
  Training size $= n \cdot (i + 1)$;
  Total size = training size $\cdot \frac{k}{k-1}$;
  Split training data into subsets;
  Use KFold to create k-fold cross validation instance kf;
  for *training indices, test indices in* kf **do**
    Fit GP to $k - 1$ folds of training data;
    Use 1 fold as test data;
    Predict values $\hat{y}_{train}$ for training data;
    Predict values $\hat{y}_{test}$ for test data;
    $R^2_{\text{train}}(\hat{y}, y) = 1 - \frac{\sum_{i=0}^{\text{Training size}-1}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{\text{Training size}-1}(y_i - \bar{y})^2}$;
    $R^2_{\text{test}}(\hat{y}, y) = 1 - \frac{\sum_{i=0}^{n-1}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{n-1}(y_i - \bar{y})^2}$;
  end
  Find mean m and std σ_{std} of R^2_{test}-values and R^2_{train}-values
end
Result: $\mathbf{n}, \mathbf{m}(R^2_{\text{train}})$, $\boldsymbol{\sigma}_{std}(R^2_{\text{train}})$, $\mathbf{m}(R^2_{\text{test}})$, $\boldsymbol{\sigma}_{std}(R^2_{\text{test}})$.

Algorithm 3: Pseudocode for k-fold cross validation of distributed Gaussian processes, which calculates the R^2-scores for training and test data as a fucntion of the number of experts, to be used for *e.g.* learning curves. In the R^2-score calculation, y_i are true values, $\hat{y}_i$ are GP predicted values, and $\bar{y}$ is the mean of all y_i.

Appendices

Appendix A

Benchmark for Distributed Gaussian Processes

The benchmark function for parallelised distributed Gaussian processes is

$$f(x_1, x_2) = 4x_1x_2,$$

where the vectors $\mathbf{x} = (x_1, x_2)$ were drawn from a random normal distribution using the `numpy` function `random.randn`. Gaussian processes implemented by `scikit-learn` in the function `GaussianProcessRegressor` were compared to distributed Gaussian processes with 4 experts. 2000 training points and 1000 test points were used, and the resulting times for the GP and DGP were

$$\text{Gaussian processes time: } 154.12 \text{ s} \tag{A.1}$$

$$\text{Distributed Gaussian processes time: } 5.61 \text{ s} \tag{A.2}$$

Histograms of the relative deviances for Gaussian processes (GP) and Distributed Gaussian processes (DGP) are found in Fig. (A.1).

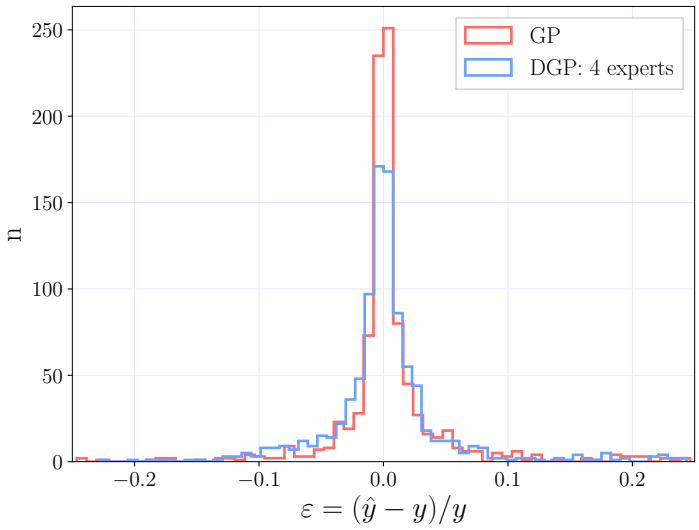


Figure A.1: Histogram of the relative deviance between true value y and predicted value $\hat{y}$ for Gaussian process regression (GP) and Distributed gaussian process regression (DGP) for the function $f(x_1, x_2) = 4x_1x_2$.

Bibliography

[1] P. Batzing and A. Raklev, *Lecture notes for fys5190/fys9190*, .

[2] A. Kvellestad, *Chasing susy through parameter space*, .

[3] S. Weinberg, *The Quantum Theory of Fields*, vol. 1. Cambridge University Press, 1995, 10.1017/CBO9781139644167.

[4] A. Balerna and S. Mobilio, *Introduction to synchrotron radiation*, in *Synchrotron radiation*, pp. 3–28. Springer, 2015.

[5] P. M. Nadolsky, H.-L. Lai, Q.-H. Cao, J. Huston, J. Pumplin, D. Stump et al., *Implications of cteq global analysis for collider observables*, *Phys. Rev. D* **78** (Jul, 2008) 013004.

[6] M. Aaboud, G. Aad, B. Abbott, J. Abdallah, O. Abdinov, B. Abeloos et al., *Search for squarks and gluinos in final states with jets and missing transverse momentum at \sqrt {s} s= 13{\mathrm {TeV}} tevwith the atlas detector*, *The European Physical Journal C* **76** (2016) 392.

[7] R. K. Ellis, W. J. Stirling and B. R. Webber, *QCD and collider physics*. Cambridge university press, 2003.

[8] W. Beenakker, R. Höpker, M. Spira and P. Zerwas, *Squark and gluino production at hadron colliders*, *Nuclear Physics B* **492** (1997) 51–103.

[9] G. P. Lepage, *A new algorithm for adaptive multidimensional integration*, *Journal of Computational Physics* **27** (1978) 192 – 203.

[10] W. Beenakker, R. Höpker and M. Spira, *Prospino: a program for the production of supersymmetric particles in next-to-leading order qcd*, *arXiv preprint hep-ph/9611232* (1996) .

[11] W. Beenakker, C. Borschensky, M. Krämer, A. Kulesza, E. Laenen, S. Marzani et al., *Nlo+ nll squark and gluino production cross sections with threshold-improved parton distributions*, *The European Physical Journal C* **76** (2016) 53.

[12] C. Balázs, A. Buckley, L. A. Dal, B. Farmer, P. Jackson, A. Krislock et al., *Colliderbit: a gambit module for the calculation of high-energy collider observables and likelihoods*, *The European Physical Journal C* **77** (2017) 795.

[13] A. Kulesza and L. Motyka, *Threshold resummation for squark-antisquark and gluino-pair production at the lhc*, *Physical review letters* **102** (2009) 111802.

[14] A. Kulesza and L. Motyka, *Soft gluon resummation for the production of gluino-gluino and squark-antisquark pairs at the lhc*, *Physical Review D* **80** (2009) 095004.

[15] W. Beenakker, S. Brensing, M. Krämer, A. Kulesza, E. Laenen and I. Niessen, *Soft-gluon resummation for squark and gluino hadroproduction*, *Journal of High Energy Physics* **2009** (2009) 041.

[16] W. Beenakker, S. Brensing, M. Krämer, A. Kulesza, E. Laenen, L. Motyka et al., *Squark and gluino hadroproduction*, *International Journal of Modern Physics A* **26** (2011) 2637–2664.

[17] A. D. Martin, W. J. Stirling, R. S. Thorne and G. Watt, *Parton distributions for the LHC*, *Eur. Phys. J.* **C63** (2009) 189–285, [0901.0002].

[18] D. Sivia and J. Skilling, *Data analysis: a Bayesian tutorial*. OUP Oxford, 2006.

[19] M. Stein, "Interpolation of spatial data: some theory for kriging. 1999."

[20] M. Abramowitz and I. A. Stegun, *Handbook of mathematical functions: with formulas, graphs, and mathematical tables*, vol. 55. Courier Corporation, 1964.

[21] C. E. Rasmussen and C. K. Williams, *Gaussian processes for machine learning*, vol. 1. MIT press Cambridge, 2006.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel et al., *Scikit-learn: Machine learning in Python*, *Journal of Machine Learning Research* **12** (2011) 2825–2830.

[23] J. V. Sparre, *Fast evaluation of supersymmetric cross sections*, .

[24] B. Allanach, *Softsusy: A program for calculating supersymmetric spectra*, *Computer Physics Communications* **143** (2002) 305 – 331.

[25] M. P. Deisenroth and J. W. Ng, *Distributed gaussian processes*, *arXiv preprint arXiv:1502.02843* (2015) .

[26] V. Tresp, *A bayesian committee machine*, *Neural computation* **12** (2000) 2719–2741.