Vrije Universiteit Brussel

FACULTEIT WETENSCHAPPEN EN BIO-INGENIEURSWETENSCHAPPEN

Anomalous Couplings Project

Progress Report and documentation of Tools

Annik Olbrechts

Started 4 April 2014

Contents

Chapter 1

Pending issues

- Does this created FeynRules model still contain Effective Field Theory?
- If the kinematics doesn't change for coupling parameters larger than 1, how is it than possible to differentiate the different configurations which will be studied?
- Access to Mathematica for Width studies ... ?
- XS values which are used are the ones calculated using the MadGraph_v155 version. Should this be calculated again using the new MadGraphv5_aMC@NLO since normally this shouldn't change ...?
- Create branch for localgrid scripts! (Now already a branch exists for the documentation !)
- Understand *CorrectPhi* comment in MadWeight name
  → Maybe related with phi issues of neutrino's ...
- EventWeight calculated in analyzer should be integrated into MadWeight output (Hence multiply MadWeight weight with the EventWeight for each event!)
  Otherwise all the effort to include the JES and PU stuff has no influence at all and is not taken into account!
- Update EventNumberInformation file to only cout the information when the event has passed the eventSelection requirements. Otherwise just discard this event from the output file in order to avoid a long .txt file and long running time ...
- When creating ntuples for Data/MC comparison the EventNumberInformation output should be stored here as well. This will allow to calculate the likelihood and weight distributions for specific p_T cuts without having to run the entire base code again. So also the different python scripts which have been created should be moved to this .cc code in the future, so better not to spent too much time on improving these python scripts to avoid double work ...
  For storing the generator decay channel information the c++ enum should be used which can store multiple variables (**Check this out!**)
- Need to understand how MadWeight deals with the permutations ... Should this permutation of the light jets be done within MadWeight or should two .lhco files be created and sent separately to MadWeight. If the latter is the case, how should they be combined afterwards ?

- Be careful with used Beam Energy when using MadGraph. In the newest MadGraph directory this still seems to be set to 7TeV in stead of the required 4 GeV.
  Could this have an influence on the obtained MadWeight results when using the new MadWeight version (connected to the newest MadGraph version) ...

- Specific MadWeight question for the latest version:
  The 'refine' option allows you to realunch the computation of the weights which have a precision lower than X. ***But how can you find this precision?***

- Should a ScaleFactor correction be applied for the Branching ratio ?
  This is not the case in code of James, but should find recommendations somewhere ...

- **To Check:** Does MadWeight give a different weight when the Neutrino Mass changes?
  → Currently mass is manually set to zero ...

- Running the analyzer code with the PileUp and JES/JER influences correct and active, the number of preselected events in the Event Selection Table has changed. Need to understand if it is expected that these corrections have an influence (lowering the number) on the number of preselected events ...

- Check whether the created root file can be trusted when the number of considered b-tags is greater than 1 ... !!

Chapter 2

Used Tools and Techniques

Chapter 3

Generator level bottlenecks

The goal is to obtain as fast as possible results on generator level in order to ensure that no bottlenecks are found when running MadWeight. The advantage of only using generator level results is that one can be completely sure that the studied events are actual semi-leptonic ttbar events. Hence MadWeight should not have any problems calculating the weight for these kind of events and no CPU time will be spent on uncorrect events. This implies that any deviation from the expected results implies a bias, or even a problem, concerning the MadWeight output.

Once the results correspond to the expectations these preliminary results should be easily extended to reconstructed events. Finalizing the event selection then allows to fully trust the results obtained on reconstructed level and make sure that any deviation should be explained by the influence of the applied event selection.
These results can then be used to optimize the event selection with respect to the MadWeight output and CPU time needed.

3.1 Transfer Functions

In order to obtain results with MadWeight, the Transfer Functions which link the reconstructed energy distribution with the actual energy distributions should be calculated. In the case of generator level events this is not that important since no smearing of the energy is expected, but in order to avoid any bias from the used Transfer Functions it is adviced to use the real Transfer Functions with a smaller width.

The method to obtain the parameters describing the energy smearing is (partly) explained in the PhD Thesis of Arnaud Pin[1] which can be found in:

https : *//cp*3.*irmp.ucl.ac.be/upload/theses/phd/Pin_Arnaud.pdf*
/home/annik/Documents/Vub/PhD/ThesisSubjects/AnomalousCouplings/
*PrepareGenLevelRunning_Sep*2014/*TransferFunctions*

The method to obtain the parameters of the Transfer Functions is based on the code

[1]It should be noted that in the PhD Thesis of Arnaud, and in the current double Gaussian transfer function syntax in MadWeight, only 5 parameters are used. The narrow gaussian distribution doesn't have a normalisation parameter in front, but is normalized afterwards in the MadWeight code.

received from Petra and Lieselotte, which was used in the Master thesis of Lieselotte[2]. However it should be noted that the original code only had 4 bins due to limited statistics, and now 10 bins are used.
This code is almost identical to the ROOT class FitSlicesY() but has some minor differences. Some of these have already been changed in order to match better with the ROOT class.
One of the most important differences between the two approaches was the treatment of the underflow and overflow bin. In the received code these two bins were respectively added to the first and last bin and hence included in the fitting range. This is not the desired behavior since the size of underflow/overflow bin can be relatively large compared to the first/last bin and significantly change the value of these bins. This would then imply that the position of the first and last bin is not located at the correct position and will, especially in the case of a limited number of bins, have a significant influence on the fit result. Now these underflow and overflow bins are just discarded from the fit range and will have no influence on the final result.

Another important, but useful, difference between the two methods is the number of histograms which are saved. The received code saves for each distribution which is considered the ProjectionY distribution together with the double Gaussian fit for this bin. This can not be changed in the ROOT class which only stores the distribution of the 6 parameters of the double Gaussian fit formula.

However, even after carefully ensuring that both methods are identical the obtained results are not. Up to now it is not clear what is the reason for the discrepancy between the two results and the only way to found out is comparing the distributions and results for a significant amount of statistics.
One possibility is the used fit ranges and number of bins. If the last bins are low on statistics their distribution might not agree with a double Gaussian distribution and hence result in a failed fit. Therefore the distribution for each ProjectionY bin is now closely studied for all of the considered histograms. This can be found is the following section 3.1.2.

3.1.1 Creating the Transfer Function

In this subsection the different steps which were performed in order to built the actual Transfer Functions and implemented in MadWeight will be discussed in detail. The Transfer Functions used in this analysis are assumed to be uncorrelated, as shortly discussed in the PhD thesis of Arnaud. This assumption is necessary since otherwise it would be impossible to build them ourselves. The implication of this assumption is given in Equation 3.1 and should be checked by looking at E vs θ, E vs ϕ and θ vs ϕ histograms.

$$W(E, \theta, \phi) = W(E)W(\theta)W(\phi) \tag{3.1}$$

A first important remark which should be made is the choice for using p_T dependent Transfer Functions, in stead of the general E-dependent Transfer Functions already implemented in MadWeight. The changes which have to be done in order to correcty

[2]Additional feedback was also received from Arnaud, but he was using a completely different unbinned likelihood method. Since this only arrived after a couple of weeks waiting, it was decided to continue with the binned likelihood method used by Petra and Lieselotte.

implement the Transfer Function in MadWeight are discussed in 3.1.1.1.

The following equation, Equation 3.2, gives explicitely the functional form of both the double Gaussian fit and the p_T-dependent calorimeter formula. As can be seen from this equation, the calorimeter part should not be taken to literally since it is only the actual calorimeter p_T dependency in the case of the σ-parameters of the two Gaussian fits.

$$\begin{aligned} a_3 * e^{-\frac{(x-a_1)^2}{2*a_2^2}} + a_6 * e^{-\frac{(x-a_4)^2}{2*a_5^2}} \\ a_{x,0} + a_{x,1} * \sqrt{p_T} + a_{x,2} * p_T \quad (\text{ for x = 2 \& 5 }) \\ a_{x,0} + a_{x,1} * p_T + a_{x,2} * p_T^2 + a_{x,3} * p_T^3 + a_{x,4} * p_T^4 \quad (\text{for x = 1, 3, 4 \& 6}) \end{aligned} \tag{3.2}$$

3.1.1.1 Using p_T dependency

The Transfer Function configuration files in MadWeight are flexible enough to change the kinematic variables used for the Transfer Function calculations. Hence it seems to be more relevant to utilize the transverse momentum in stead of the energy of the considered partons and jets. Especially since the LHCO file used in the MadWeight calculations has the transverse momentum as input variable. Therefore it seems much more useful and realistic to use this parameter and not the energy. This implies that the Transfer Function configuration file should be adapted to use the *pt(p)* and *pt(pexp)* variables and not the currently used *p(0)* and *pexp(0)*.

After actually implementing the created Transfer Function files[3] and trying to use this Transfer Function one issue showed up which is currently not completel solved yet. MadWeight contains hard-coded files which only allow the use of the E, THETA and PHI variable for the Transfer Function dependencies. Therefore the so-called "block name" PT is not accepted by the *change_tf.py* file. However it seems that there is no clear physical motivation for this limited choice of kinematic variables, but in order to be completely sure additional information is needed from Olivier and Pierre.
Once this issue is resolved the following file, containing the restricted list of kinematic variables, should be adapted[4]:

bin/internal/madweight/change_tf.py

3.1.1.2 Creation of the Transfer Functions

The Transfer Functions are created from the simulated $t\bar{t}$ sample which will be used throughout the entire analysis. A very small nTuple is created from this simulated sample containing only the TLorentzVector information of both generated and reconstructed particles. From this the necessary diagrams, such as the 2D distributions of p_T, θ and η difference between the generated and reconstructed particle with respect to the generated value, are created and saved in a ROOT file:

AnomalousCouplings/TFInformation/PlotsForTransferFunctions_FromTree.root

[3]This was tested the first time on 30 November 2014.

[4]Another solution is to keep the block name equal to E but just use the transverse momentum information. However this might result in confusion when in some files the hard-coded E parameter gets written down (for example on plots).

This analyzer, called $TFFit.cc$, also performs the double Gaussian fit of these 2D histograms and afterwards the E-dependent calorimeter fit. The technicalities and specific details of these fit procedures are documented in the following class:

$$AnomalousCouplings/PersonalClasses/src/TFCreation.cc$$
$$AnomalousCouplings/PersonalClasses/interface/TFCreation.h$$

The results of the two consecutive fits performed on the Y-projections of these 2D distributions are stored in another ROOT file, together with the original 2D histograms. Also the function form of the double Gaussian fit formula using the obtained fit parameters, added in order to test the robustness of the fit results outside the fitted range, can be found in this ROOT file.

$$AnomalousCouplings/TFInformation/CreatedTFFromDistributions\_FromTree.root$$

This analyzer also has the flexibility to perform the fit on the entire range or on pre-defined ranges set by the user. For this a separate function, called *SetFitRange*, is created where for each histogram the fit range for each separate bin can be defined. This is extremely useful to optimize the doubleGaussian fit which has to cover both the peak and the tail of the distributions in order to correctly calculate the 6 fit parameters.
Another useful aspect of this analyzer is the automatic creation of the necessary .dat Transfer Function files needed for implementation in MadWeight. This is done in the *WriteTF* class and the created files are:

$$AnomalousCouplings/TFInformation/TF\_user.dat$$
$$AnomalousCouplings/TFInformation/transfer\_card\_user.dat$$

This first file contains the functional form of both the E-dependent calorimeter fit and the functional form of how these 6 parameters should be included in the double Gaussian formula. Also the width for the different kinematic variables is defined within this file. For the moment the method used in the Transfer Functions already implemented in MadWeight is followed, implying that the width of the Transfer Function is defined as the maximum of the σ-parameter of the two Gaussians considered in the double Gaussian fit. The only difference is that in stead of the generated kinematic information, which is the variable on the abscissa of the considered 2D histograms, the reconstructed one is used.

The second file contains the actual values of the different fit parameters for all the considered kinematic variables and particle types. Therefore this file is an extensive list of values to which is referred in the previous $TF\_user.dat$ file. For each particle type and kinematic variable the numbering used should be unique such that the correct values are implemented in the functional forms of the used fit formulas.

3.1.1.3 Importance of start values

Since the double Gaussian fit really needs accurate information of both the peak and the tails, detailed review of the start values for each of the 6 fit parameters significantly improves the success rate of the fitting method. However it is important to note that setting these start values clearly improves the results obtained from the fit so special care should be taken to ensure the correctness of these start values. One way is by intensively comparing the fit distributions for the different particles since in the case of light jets and

b-jets some similar behavior is expected.
After quite a while it is possible to quickly see whether the fit distribution has the expected shape and hence whether the used start values can be considered as stable. The start values used for the different 2D histograms is given in Table 3.1

2D histogram	Used start value for fit parameter					
	First (narrow) gaussian			Second (wide) gaussian		
	Mean a_1	Sigma a_2	Amplitude a_3	Mean a_4	Sigma a_5	Amplitude a_6
b-jet $\Delta\phi$	0.0002	0.022	8000	0.0002	0.06	3000
b-jet Δp_T	10	-12	20000	13	10	-5000
b-jet $\Delta\theta$	0	0.013	6000	0	0.04	2000
light jet $\Delta\phi$	0	0.022	8000	0.0004	0.002	3000
light jet Δp_T	0	8	4000	0	12	4000
light jet $\Delta\theta$	0	-0.014	6000	0	-0.05	2000
electron $\Delta\phi$	0	0.0012	1500	0	0.006	600
electron Δp_T	0	0.9	1500	0	-2	600
electron $\Delta\theta$	0	0.0013	2500	0	0.007	600
muon $\Delta\phi$	0	0.0004	800	0	0.0026	600
muon $\frac{1}{\Delta p_T}$	0	0.0003	2000	0	0.0006	500
muon $\Delta\theta$	0	0.002	500	0	0.0004	500

Table 3.1: caption ... **Need to make sure that the narrow and wide gaussian is always the same ... Otherwise are the start values not correct for the different eta-bins ...!!**
Need to look at the sigma value to decide on the narrow/wide gaussian!

3.1.1.4 Splitting in separate $|\eta|$ bins

Since the kinematic variables tend do depend on the pseudorapidity η the considered 2D histograms are created for four distinctive $|\eta|$ regions. It has been chosen to split the barrel region into three separate bins while the entire endcap region is contained within one single bin. The chosen binning is given in Table 3.2 together with the percentage of events present in each of the considered $|\eta|$ bins. This clearly shows the lower statistics available in the endcap region which results in larger difficulties of properly reconstructing the fit parameters in this region. This is shortly discussed below.

	$\lvert\eta\rvert \leq 0.375$	$0.375 < \lvert\eta\rvert \leq 0.75$	$0.75 < \lvert\eta\rvert \leq 1.45$	$1.45 < \lvert\eta\rvert \leq 2.5$
Relative # events	26.21 %	23.91 %	32.54 %	17.34 %

Table 3.2: caption ...

The analyzer mentioned above is developed in such a way that both the fit results for all events as the results for the four separate $|\eta|$ bins are stored together. Therefore all the results can always be compared in the created ROOT files and in the distinct *dat* files.

One important difference between the 2D-distributions containing all events and the 2D-distributions specific for one of the $|\eta|$ bins is the number of bins used. Since the statistics is significantly lower for the $|\eta|$ specific histograms, the predefined bin number

is lowered with 25%. This ensures a more stable tail for the distribution and still a correct reconstruction of the peak.

However for the last $|\eta|$ bin considered, the endcap part, a slightly different method is used. Since the statistics is much lower in this part of the detector the used range had to be slightly stretched in order to ensure a nice overview of the tails. This is necessary for the correctness of the double Gaussian fit. Therefore the range of the abscissa is enlarged on both sides with 20% and the used number of bins for this axis is identical to the one used for the overall 2D-distribution.

3.1.1.5 Separating narrow and wide gaussian

Remark: Should check whether this is actually necessary for the MadWeight implementation. MadWeight only seems to need the general fit formula and doesn't need to know which of the two distributions is the narrow and which is the wide one.

In order to select the narrow and wide gaussian both the amplitude and the σ-parameter should be compared, but keeping in mind that this second one is the most important variable. However it is expected that the distribution with the narrowest distribution also has the highest peak. In order to easily compare the two distributions a stacked canvas is added to the ROOT file which shows both distributions for different $p_{T,gen}$ values in one stacked canvas. The distinction between narrow and wide gaussian distribution is currently being made by the size of the σ variable. Hence the histogram with the narrowest distribution is plotted in red while the widest one is plotted in green.
Such a canvas is made for each of the considered 2D-histograms, both for the one containing all events and for the ones splitted in separate $|\eta|$ bins. They should be analyzed in detail in order to understand the correctness of the double Gaussian fit applied for the creation of the Transfer Functions. For the moment there a still a couple of 2D-histograms which don't show the expected behavior. Additional investigation of these stacked canvasses should be performed as soon as possible.

3.1.1.6 Coping with low statistics in last $|\eta|$ bin

Solution : Excluding bins! + combining bins

3.1.1.7 Extrapolation method

The implementation in MadWeight should be done in such a way that the value of the outermost bins is used for all the p_T values outside the fitted region. This is necessary since the extrapolation using the obtained fit parameters doesn't result in the desired double Gaussian behavior for these p_T values. For values outside the fitted region, an inverted double Gaussian distribution or a distribution with two distinct peaks occurs rather often. This can be seen in Figure 3.1

3.1.1.8 Actual implemenation in MadWeight

All the possible Transfer Functions which are implemented in MadWeight can be found in the $Source/MadWeight/transfer\_function/data$ directory. Any file can be added

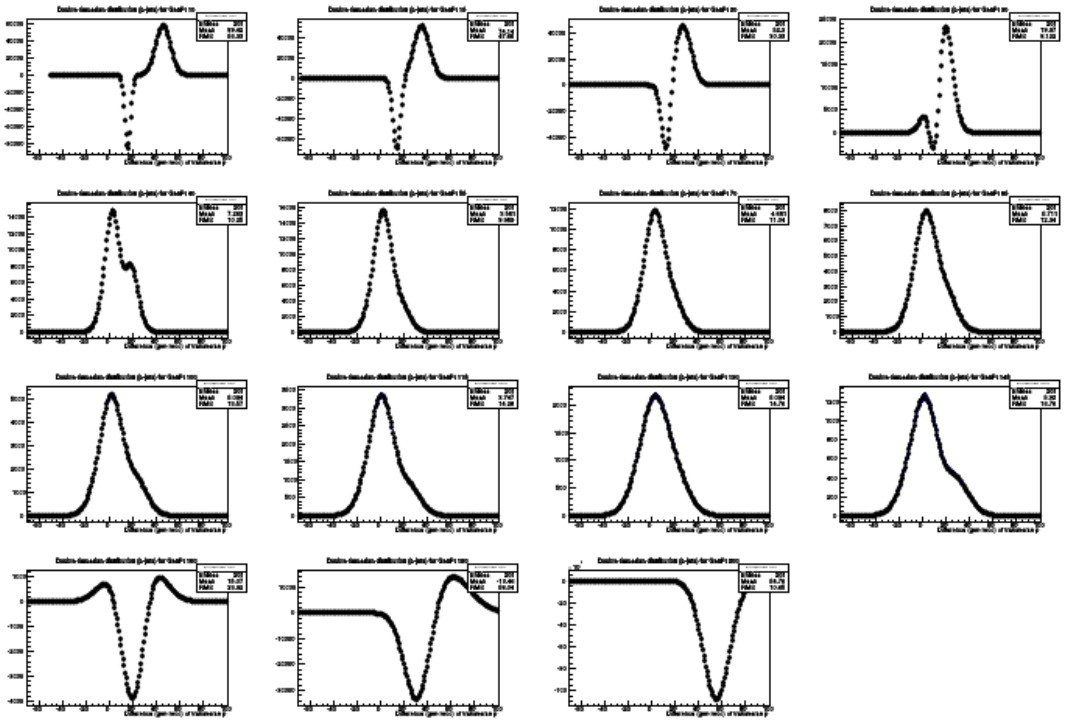

Figure 3.1: Extrapolation obtained using the fit parameters of the double Gaussian functional form. Values outside the fitted range show a distinctly different shape with respect to the ones actually fitted.

to this list and used within MadWeight.
The relevant files used for the creation of the Transfer Functions are given below. The first one is the translation of the used $TF\_user.dat$ into a MadWeight readable file which can be implemented. The second file is only relevant around line 292 where the function used for the Transfer Function creation is explained. This is the general MadWeight constructor file where all the different MadWeight functions are defined.

$$Source/MadWeight/transfer\_function/transfer\_function.f$$
$$bin/internal/madweight\_interface.py$$

The commands which have to be executed in order to build this MadWeight readable file for the Transfer Function are given here:

$$./bin/mw\_options$$
$$define\_transfer\_fct$$

3.1.2 Obtained distributions

Each of the considered histograms is a 2D histogram where the abscissa represents the transverse momentum of the generator level parton and the ordinate the difference between the generator level parton and the reconstructed matched particle. This is done for the difference in transverse momentum and in θ and ϕ angles.
All the interesting histograms can be created automatically, for each of the desired $|\eta|$ bins separately, using the following ROOT analyzer:

$$AnomalousCouplings/TFInformation/FitDistributions/SaveFitHistograms.C$$

This analyzer is able to create each of the histograms separately, both as *pdf* and *png*, and automatically saves all the histograms of one type in a large stacked canvas. This allows to quickly see the used 2D distributions for each of the particle types (b-jets, light jets, electrons and muons) and the three kinematic variables (p_T, η and ϕ). This stacked canvas is shown in Figure 3.2. Also the general behavior of each of the fitted diagrams together with the overal χ^2 distribution is created for each 2D histogram as can be seen from Figure 3.3, showing this for the p_T distribution of the b-jets. Finally the distribution of the 6 double Gaussian fit parameters together with the fit result of the E-dependent calorimeter formula is also collected in a stacked canvas, as can be seen in Figure 3.4.

The 2D distributions for the difference in transverse momentum tend to show a slightly

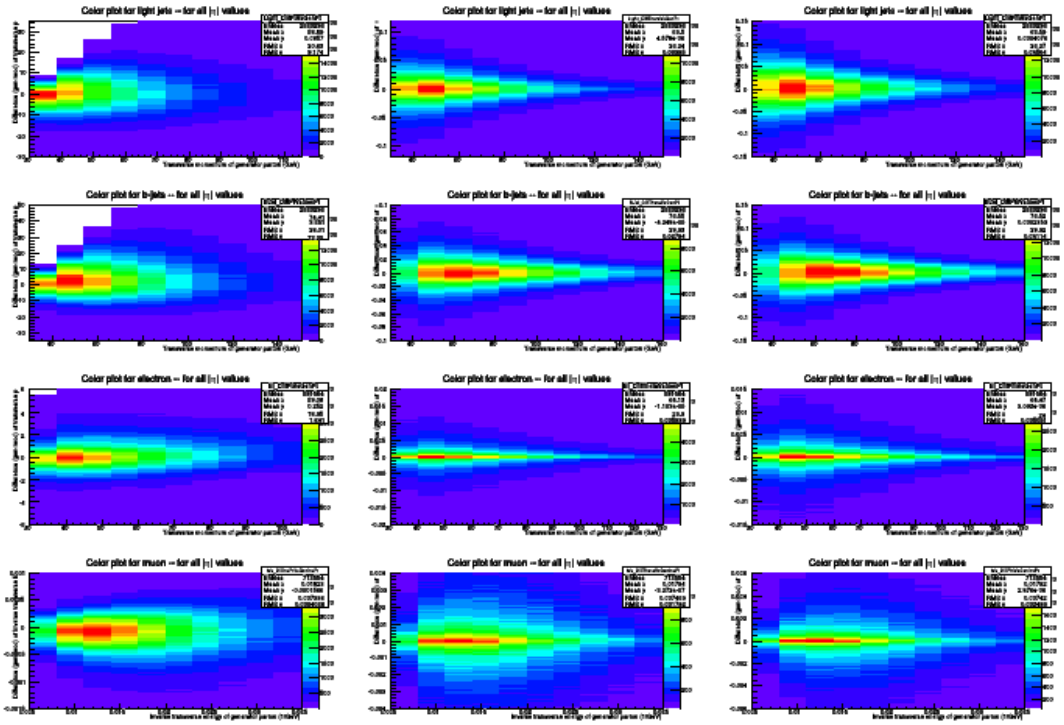

Figure 3.2: Used 2D distributions for double Gaussian fit. **IMPROVE CAPTION!**

asymmetric behavior, as can be seen from Figure 3.2. This can be explained by the influence of the event selection, which has a difference effect on the generated particle than the reconstructed particle. This because a particle surviving the p_T cut actually has a different p_T value on generated level due to **bad resolution, detector effects (???)**. This effect is almost negligible for the ϕ and θ angles **(Definitely sure that this is the case ??)**.

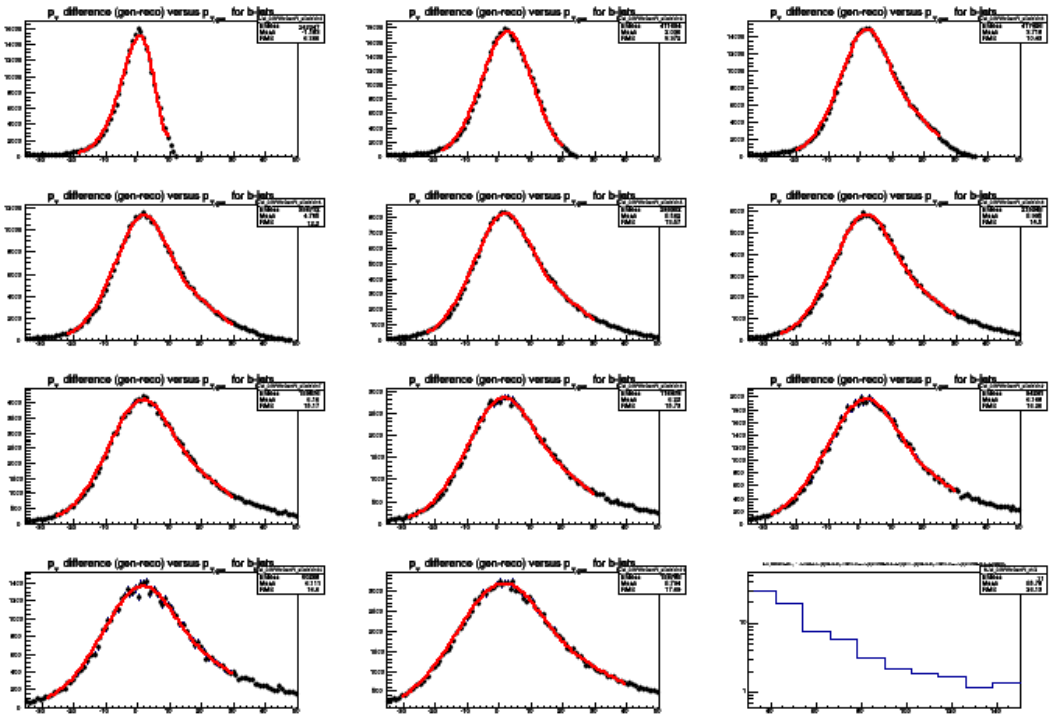

Figure 3.3: Distribution of the energy difference between the generator level parton and the corresponding light quark jet for each of the 10 considered bins and the overflow bin. All distributions were fitted with a double Gaussian function.

STILL TO REVIEW

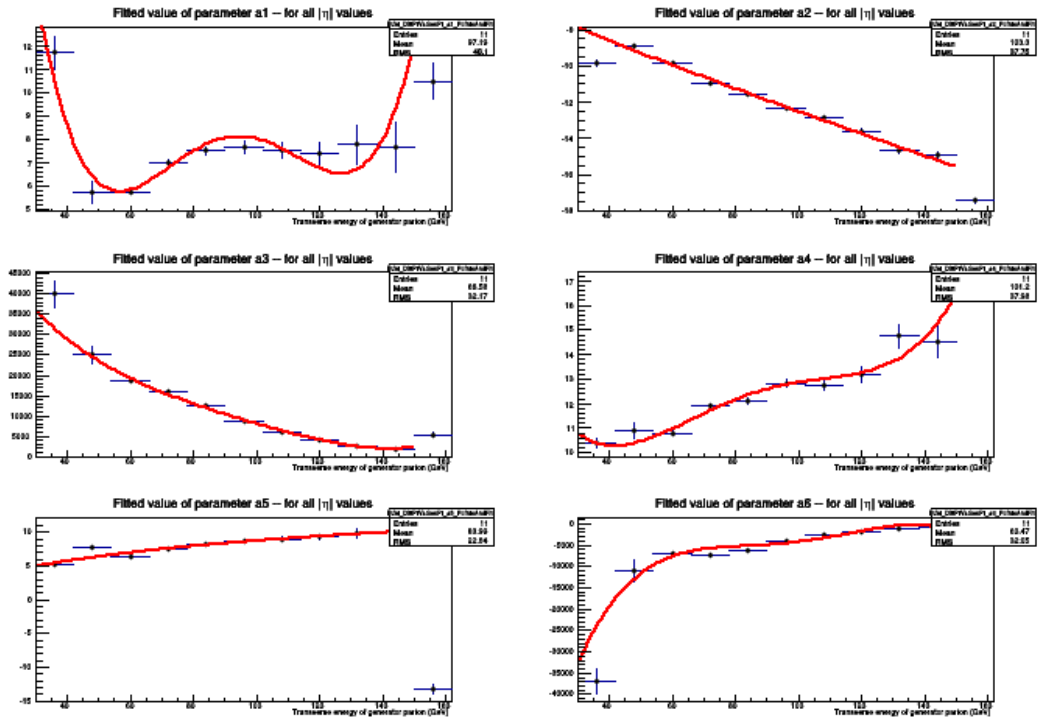

Figure 3.4: Energy dependency of the 6 parameters of the double Gaussian fit function. The result of the double Gaussian fit for each p_T bin is combined in a p_T dependent histogram and then fitted with the Calorimeter energy function as explained in the PhD Thesis of Arnaud Pin.

3.1.3 Review of Transfer Functions

3.2 Cross Section distribution for new grid

Remark: Cross section comparison

Need to check how it is possible that the XS values in the XS grid are identical for both versions (first check whether this is indeed the case), but are different for the top quark mass simulation ... Table 3.3.

3.3 First results: Wrong log(likelihood) minimum

The first obtained MadWeight results for the enlarged grid ($V_L \in [0.8, 1.2]$ and $V_R \in [-1, 1]$) using only parton-level ttbar events did not result in the expected minimum of $(V_L, V_R) = (1, 0)$. This can be seen from Figure 3.5, which shows the distribution of the log(likelihood) for each point in the considered grid.

One of the possible influences on the displaced minimim of the log(likelihood) distribu-

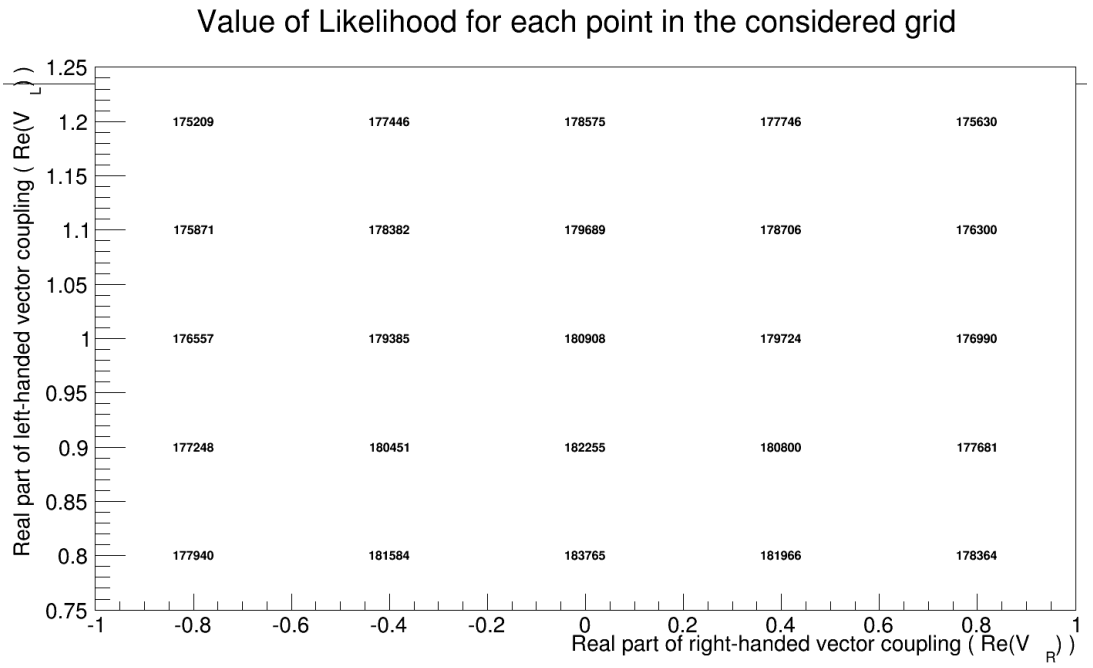

Figure 3.5: Distribution of the log(likelihood) for each point in the grid using 3200 parton-level positive semi-muonic ttbar events. The transfer function used to smear the parton-level kinematics is the single-gaussian function standard included in MadWeight.

tion could be the normalisation of the Cross Section influence. This XS normalisation ($\frac{XS}{XS^{SM}}$) should be multiplied with the likelihood value, not the log(likelihood). Hence in order to correctly take this into account the obtained log(likelihood) value for each point in the grid should be corrected using the logarithm of this XS normalisation. The distribution of the normalisation on the Cross Section can be found in Figure 3.6 together with the log(likelihood) distribution after correctly taking into account this XS normalisation.

The formula which has been used is the following (Equation 3.3):

$$P(y|a) = \frac{1}{\sigma(a) * Acc(a)} \int W(y|x,a) * Eff(x,a)|M(x,a)|^2 T(x,a) dx \tag{3.3}$$

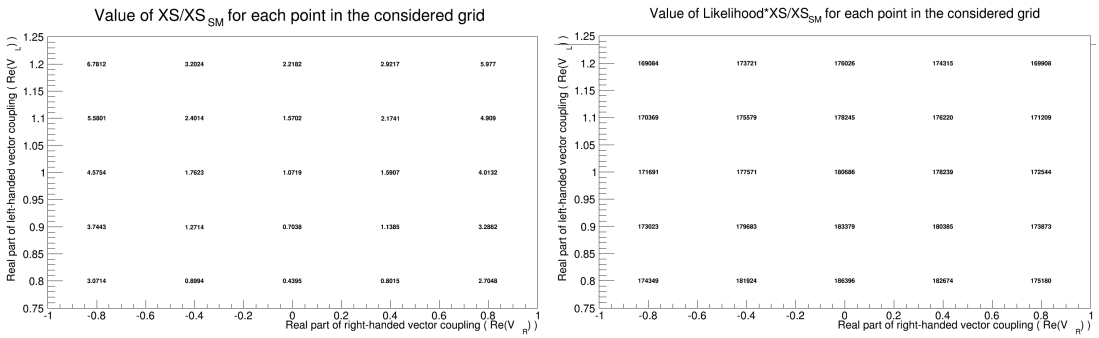

Figure 3.6: Distribution of the XS normalisation for positive semi-muonic ttbar events (left) and distribution of the log(likelihood) after taking into account this normalisation. As in the previous figure 3200 positive semi-muonic have been used to obtain this distribution and a single gaussian transfer function has been applied to smear the kinematics of these parton-level events.

$$\mathcal{L} = \prod P(y|a) \tag{3.4}$$

The normalisation which has been applied in Figure 3.6 is given in Equation 3.5:

$$\mathcal{L}_{Norm} = -ln(\sum P(y|a) * \frac{XS}{XS^{SM}}) = -ln(\mathcal{L}) - ln(\frac{XS}{XS^{SM}} * N) \tag{3.5}$$

It should be checked whether this is the correct method to take into account the normalisation of the XS. Currently it has been assumed that this XS normalisation should be applied for each weight and hence is multiplied with the number of considered events. In the case that this normalisation should just be multiplied with the overall likelihood value ($\mathcal{L}$) the sum over the number of considered events drops out of the equation implying a very small influence of the XS on the log(likelihood) distribution.

As highlighted in the general Matrix Element Method formula (Equation 3.3), the probability to measure the observed quantities y already has a normalisation factor for the cross section. This factor is defined as the channel cross section and is calculated using Equation 3.6:

$$\sigma(a) = \int_{X_i} |M(x,a)|^2 T(x,a) dx \tag{3.6}$$

Hence it should be investigated in detail whether this XS normalisation should still be applied. From the above equations could be concluded that the change in cross section is actually already incorporated in the weight obtained from MadWeight. This could make sense since MadWeight has all the necessary information to calculate the cross section for each point in the considered grid. The cross section values for each point have been calculated using MadGraph and the same model as used for the MadWeight calculations. Unfortunately it is not completely clear from the MadWeight documentation whether this is actually included in the weight or not.

3.4 Correct normalisation of Matrix Element probability

As could be seen in Equation 3.3 a term $\sigma(a)$ is included in the general Matrix Element Techniques formula. However it is not clear whether this cross section normalisation is actually performed within the MadWeight calculations or whether this normalisation should be done afterwards.
This question is closely related to the order of the current obtained weight values with MadWeight. Up to now no weight larger than 10^{-22} have been obtained, resulting in a very large log(likelihood) value.

This small value can be caused by many different reasons for which the most plausible ones are listed here:

- The normalisation of the MadWeight probability should still be done and is not performed within the Matrix Element Techniques formula.
  This can only be ruled out by contacting the Madweight experts and asking explicitely what is done in the Madweight calculations. Also a possible hint could be found inside the MadWeight python files (but this should only be done if the received answer is not perfectly clear).

- The smallness of the weight could be caused by an error inside the created FeynRules model. *Should also look for the mail where one of the MadWeight experts (Olivier/Pierre or even Celine) answered about the possible explanation for the smallness of the weight and whether this implies some wrong assumptions).*
  A possible way to exclude that the origin of this problem is the AnomalousCouplings FeynRules model is by comparing the results for the top mass fit when both the SM FeynRules model and the AnomalousCouplings model is used. If the weights are also this small when the SM model is used this smallness should be solved by an additional normalisation factor.

Update 31/10/2014: Probability function NOT normalized (according to mail Olivier)

As was expected from the smallness of the obtained MadWeight probabilities should the cross section normalisation be applied afterwards. Only in the older versions of MadWeight (based on MG) was this normalisation included automatically.

3.4.0.1 Measurement of top quark mass using Matrix Element Method

Comparing SM model with AnomalousCouplings model

As a first step the Feynmann diagrams belonging to the two different models should be compared. This information can be found in the *index.html* file in the following directories (and the files should be opened using firefox on mtop since this is the only m-machine with a working browser):

/AnomalousCouplings/MadGraph5_aMC@NLO/madgraph5/SM_ttbarSemiMuPlus
/AnomalousCouplings/MadGraph5_aMC@NLO/madgraph5/ttbarSemiMuPlus_QED2

Comparing SM cross section with MassiveLeptons cross section

In order to be sure that both models have the same Standard Model base, the cross sections for both models have been compared. This resulted in an unexpected outcome, namely that the obtained cross sections differ significantly depending on which MadGraph version is used to generate the considered events. A summary can be found in Table 3.3.

Top quark mass	MadGraph aMC@NLO		MadGraph v155	
	SM model	MassiveLeptons model	SM model	MassiveLeptons model
153	9.23 pb	9.645 pb	6.692 pb	6.984 pb
163	11.12 pb	11.63 pb	7.844 pb	8.199 pb
173	12.98 pb	13.54 pb	8.897 pb	9.281 pb
183	14.77 pb	15.4 pb	9.884 pb	10.3 pb
193	16.5 pb	17.22 pb	10.78 pb	11.25 pb

Table 3.3: Cross section values for semi-muonic (+) ttbar decay obtained using two different MadGraph versions.

From this table can be seen that there is, for both considered MadGraph versions, a small difference between the SM FeynRules model and the MassiveLeptons one. This could be caused by the different treatment of the leptons. In the SM model they are considered to be massless while in the MassiveLeptons one they are defined to have their actual mass.
A larger differrence occurs when both MadGraph versions are compared. From the answer received by Olivier it is not clear whether this difference is worrysome or could be explained by the LO theoretical uncertainties. Should also be investigated whether this difference is related to the NLO behavior of the newest MadGraph version. In case the MadGraph v155 version is not up to NLO a difference in cross section is definitely expected.

Chapter 4

Preliminary Results

Chapter 5

Understanding Results Obtained With MadWeight

Chapter 6

Analyzing created FeynRules model

Chapter 7

Event Selection

Chapter 8

Event corrections and reconstruction