Title

Subtitle

# Abstract

TOC

# Introduction

## Reweighting

## Applications

### Sherpa

### FeynRules

# SherpaWeight

## Developing SherpaWeight

## Using SherpaME

## Using SherpaWeight

# Effective Field Theory

## Implementing a UFO model for EFT in FeynRules

## Validating EFT model using WZ production

# Reweighting

## Comparison of reweighted EFT to SM for WZ-production

## Notes on reweighting using ROOT

### Scaling luminosity

### Reweighting histograms

# Optimal Observables

# Chosen observables for parton-level WZ-production

In the following sections we will measure the fit characteristics of reweighting EFT to SM data. To do so we need to select some observables to perform fits with. In addition to a handful of phase-space observables, we will also examine optimal observables. We also need to select for each observable, the range and bin sizes, in as objective a way as possible, as these will affect the fit results to some degree.

We will continue to use the SM and EFT data samples used previously. Both samples contain 1M events, which we will scale to a luminosity of 10 fb–1. Using only the SM data sample to determine data range and binning, the chosen observables and their characteristics are shown in Table 6.1.

We have chosen 3 phase-space observables, PT(Z) – the transverse momentum of Z, M(WZ) – the invariant mass of the WZ-pair, and y(Z) – the rapidity of Z. We have also added the first- and second-order optimal observables for the 3 EFT model parameters cWWW, cW, and cB to our table of observables.

Since fitting to the mean of the optimal observables should give the least statistical error, we have included in our table of observables, profile histograms of the mean of each optimal observable plotted against the invariant mass of the hard-process. Note the Mandelstam variable s used here corresponds to the hard-process and not the proton beams.

Finally, as each optimal observable is a function of the invariant mass, we have divided out that dependency for each event, and included profile histograms of the mean across events of each optimal observable divided by their invariant mass dependency, again plotted against the invariant mass. These profiles should be nearly constant, or asymptotic to a constant at high energy.

Table 6.1: Observables for parton-level WZ-production, characteristics and selected range.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Observable** | | **Units y-axis** | **Units x-axis** | **Mean a (x-axis)** | **Std. Dev. a (x-axis)** | **IQR a (x-axis)** | **Selected Range (x-axis)** | **Under/ overflow Count b** |
| **Phase-space observables** | | | | | | | | |
| 1 | PT(Z) |  | GeV/c | 54.6 | 42.5 | 41.7 | 0 to 750 | 0 / 8.2 |
| 2 | M(WZ) = |  | GeV/c2 | 325 | 167 | 138 | 0 to 3000 | 0 / 6.3 |
| 3 | y(Z) |  |  | 1.24×10−4 | 2.02 | 3.21 | −5 to +5 | 0 / 0 |
| **Optimal observables** | | | | | | | | |
| 4 |  |  | GeV2 | −1340 | 5660 | 3210 | (−6 to 1) ×104 | 6.7 / 0.2 |
| 5 |  |  | GeV4 | 8.13×108 | 4.90×1010 | 8.35×107 | (0 to 14) ×1011 | 0 / 9.5 |
| 6 |  |  | GeV2 | −5540 | 26700 | 7280 | (−12 to 1) ×105 | 7.6 / 0 |
| 7 |  |  | GeV4 | 3.93×108 | 1.86×1010 | 8.08×107 | (0 to 53) ×1010 | 0 / 9.8 |
| 8 |  |  | GeV2 | 184 | 574 | 418 | (−1 to 5) ×103 | 0.2 / 8.2 |
| 9 |  |  | GeV4 | 6.78×105 | 2.76×106 | 4.00×105 | (0 to 13) ×107 | 0 / 9.1 |
| **Mean optimal observables vs. invariant mass** | | | | | | | | |
| 10 | vs | GeV2 | GeV | Same as M(WZ) | | | | |
| 11 | vs | GeV4 | GeV |
| 12 | vs | GeV2 | GeV |
| 13 | vs | GeV4 | GeV |
| 14 | vs | GeV2 | GeV |
| 15 | vs | GeV4 | GeV |
| **Mean optimal observables divided by powers of invariant mass vs. invariant mass** | | | | | | | | |
| 16 | vs |  | GeV | Same as M(WZ) | | | | |
| 17 | vs |  | GeV |
| 18 | vs |  | GeV |
| 19 | vs |  | GeV |
| 20 | vs |  | GeV |
| 21 | vs |  | GeV |

1. For all events including those outside the selected range.
2. SM data sample at 10 fb–1.

## Data range

Most of our observables have long tails approaching a sparse distribution of single events separated by areas with no events. Once binned, these sparse areas will result in bins with low statistics and empty bins. When fitting, all bins are treated equally and if we have a relatively high number of low statistic bins, these can dominate the fit results. With some fit objectives, such as minimizing χ2, empty data bins are skipped, reducing the degrees of freedom for the fit.

One can solve such problems, by using different bin sizes across the data range to ensure that no bins have low statistics. This is discussed in the next section. For simplicity though, we have chosen to use equal-width bins for our histograms, thus sparse data tails remain an issue, and the data range must be carefully chosen.

To set the data range in a somewhat objective manner, we have chosen ranges that exclude less than 10 events on either end, after scaling to the test luminosity of 10 fb–1. Thus the underflow and overflow bins will have less than 10 events. Why use 10 events as the criteria? Simply because, a bin with 10 events and thus an error of has more than a 3 sigma probability of not having a true value of zero. Bins with less than 10 events have a small but significant probability of being zero.

In addition, we have set the range based solely upon the SM data sample, that is the data sample that we are fitting to and not the EFT data sample that we are reweighting. A common range and bin size must be used to perform binned fits. Normally one bins one’s data (the SM sample) according to its characteristics, and then bins the model (the reweighted EFT sample) equivalently.

## Binning for binned fits

All choices in binning affect fit results to some degree, and unfortunately it is not easy to gauge these effects. We desire that fit results remain fairly stable as a function of binning near our binning choice.

We have several choices when binning our data. The **first choice** is whether a single bin-width should be used across the entire range, equivalent to dividing the data range equally into a specified number of bins, or whether different bin-widths should be used through the data range.

The choice of a single bin-width is obviously the simplest. However, as mentioned in the previous section, our observables often have areas of sparse events at one or both ends of their data range. These areas will contain bins that are empty or with low statistics. One cannot reweight an empty bin, and bins with low statistics have larger relative errors. When fitting, all bins are treated equally and if we have a relatively high number of low statistic bins, these can dominate the fit results. Also with some fit objectives, such as minimizing χ2, empty data bins are skipped, reducing the degrees of freedom for the fit.

One can solve such problems, by using different bin sizes across the data range to ensure that no bins have low statistics. Thus there will be more bins in areas of high density and fewer in areas of low density. One can divide each bin’s height by its bin width, to create a density distribution, but this is not necessary for fitting, and will cannot be used with a likelihood fit objective that compares event counts. However, gauging the binning effects of a specific choice of variable bin-widths on a fit result is not an easy task.

For this project, we have chosen for simplicity to use equal-width bins for our histograms, despite the potential benefits of variable bin-widths. This means that we must be aware of the sparse regions in our data, and take care to handle empty and low-statistic bins appropriately. This is discussed further in the sections describing the fits performed.

One potential compromise between variable and fixed bin-widths, is to use fixed bin-widths, and then to fill in empty bins with events from adjacent non-empty bins, spreading the events evenly across the empty areas. This has the same affect of creating wider pseudo-bins locally where the density is lower, without actually changing the binning scheme. This obviously can only be done on model histograms and not data histograms, as data should never be modified. This technique is used where appropriate, and discussed further in sections describing the fits performed.

One can also obviously avoid low-statistic and empty bins by using more events, or fewer number of bins.

### Optimal number of bins

The **second choice**, when binning our data using fixed bin-widths, is the number of bins to use across the data range. There are widely different suggestions on how to determine the optimal number of bins for a given data set [REF]. For example, there is the very simple square root choice, used by Microsoft Excel:

|  |  |  |
| --- | --- | --- |
|  |  | (6.1) |

where *K* is the number of bins, and *N* the number of data points. For 1M data points, this gives *k* = 1000.

There is *Rice’s rule*, for approximately normal distributions:

|  |  |  |
| --- | --- | --- |
|  |  | (6.2) |

which gives k = 200 for 1M data points.  
There is *Scott’s normal reference rule* for normal distributions:

|  |  |  |
| --- | --- | --- |
|  |  | (6.3) |

where *W* is the optimal width and is the standard deviation of the data sample.  
For non-normal distributions, there is the *Freedman-Diaconis rule* [REF]:

|  |  |  |
| --- | --- | --- |
|  |  | (6.4) |

where IQR is the interquartile range of the data sample.  
The above rules are straight-forward to calculate, and the results for Scott’s and Freedman-Diaconis rule are included in table [REF].

Somewhat more complicated are functions that when minimized give the optimal bin-width, or in our case the optimal number of bins. Here are two such functions, based upon minimizing the mean integrated square error (MISE). That is, they try to minimize the mean error between a binned histogram and the true continuous distribution it estimates. Since the true distribution is not known, they estimate the true distribution from the data itself.

The first method [REF], minimizes the MISE cost function given by:

|  |  |  |
| --- | --- | --- |
|  |  | (6.5) |

Where is the mean events per bin, is the variance in the events per bin, are the number of events in bin , and is the total number of bins.

The second method, know as the cross-validation method [REF], minimizes:

|  |  |  |
| --- | --- | --- |
|  |  | (6.6) |

During testing, it was found that both functions effectively generate the same minimization curve, differing only in scale and offset, but having the same shape and minimal point. The scale of the cross-validation function was fairly independent of the observable, whereas the scale of the MISE cost function was highly dependent upon the observable chosen, mostly due to the higher power of the term in the denominator. Therefore, we chose to only minimize the cross-validation function.

The results of Scott’s and the *Freedman-Diaconis rule* along with the results of minimizing the cross-validation function are shown in Table 6.2. The following section details how the cross-validation function was minimized.

#### Minimizing the cross-validation function

Initially, we attempted to use ROOT’s fit methods to minimize the cross-validation function. To keep the function smooth, minimization was performed on a continuous number of bins, and all but the last bin were equal in size. For example, to achieve 12.2 bins, the range was divided by 12 equal sized bins, and one extra bin that was 0.2 the size of the other bins. As a cross-check, to ensure that we were minimizing the proper function parameter, we also ran the minimization using continuous bin-widths, instead of the number of bins. Fortunately, the results of minimizing the cross-validation function using either the number of bins or the bin-width as the function parameter were identical, as one would expect.

Using ROOT’s fit methods and its default Minuit minimizer, we attempted to find the optimal number of bins between 10 and 1M. The cross-validation function for most observables descends rapidly to a nearly constant plateau after about 100 bins. This plateau is nearly flat, descending very slowly as the number of bins increase, wobbling through numerous local minima, to an ultimate global minimum, before rising very slowly afterwards. See Figure [REF] for an example minimization curve.

The nearly flat nature of this plateau, often resulted in ROOT’S fit function converging on some local minima. Hesse errors were enabled to make a parabolic estimate of the errors, and these were always close to the order of the entire allowed range of the number of bins. However, it was unclear whether we could rely on the error estimates at all, as they are dependent upon a Gaussian distribution around the converged upon parameter result, and the corresponding definition of the equivalent height in the cost function corresponding to the desired confidence level. The fit results were also fairly dependent upon the tolerance setting for the minimizer, the fit range, and the initial step.

We suspected that perhaps the issue is the continuous nature of the fit parameter, as opposed to the integer nature of the number of bins the original cross-validation objection function was derived with. In an attempt to solve this, we changed the objective function to return an interpolated result between the two nearest integer values for the number of bins. For example, if the minimizer requested 12.2 bins, we would calculate the cross-validation function for 12 and 13 bins, and return a weighted average using 80% of the 12 result and 20% of the 13 result. This helped smooth the minimization function somewhat, but not sufficiently that it helped solve the issues we were having in getting consistent fit results.

In the end, we concluded that using a minimizer to find the optimal number of bins was overly complicated and untrustworthy given the circumstances. Instead, we programmed a very simple scan of the cross-validation function across an allowed range of the number of bins using only integer values in steps that were humanly selected. For example, in the 10s we scanned by steps of 5, in the 100s by steps of 25, in the 1000s by steps of 250, and so on. We scanned from 10 to 1M bins and recorded the point with the minimal cross-validation value. The cross-validation results from this simple scan are shown in Table 6.2.

For the phase-space observables, reasonably low number of bins is chosen by this method. However, for the optimal observables, particularly the second-order, the number of bins is very high. The minimization plateau of two of second order optimal observables, was observed to continue descend past our upper limit of 1M bins. This casts some doubt on the usability of the cross-validation method for our data. The long flat plateau in its entirety can be seen as roughly the functions minimum, indicating that any number of bins from somewhere between 100 and 1000 to 1M could conceivably be used.

### Selection of the number of bins

The different results for optimal bins by Scott’s rule, the Freedman-Diaconis rule, and by minimizing the cross-validation function, are fairly varied. The broad minimization plateau for the cross-validation, suggests that a broad range of number of bins will be effectively the same. So despite all the mathematics to predict the optimal number of bins, we will end up choosing the number ourselves, guided by the mathematical suggestions.

The fewer bins we have the faster the performance of the model fits we will perform in the next section. In that interest, we have chosen to limit the number of bins to a maximum of 10000. Then considering the broad minimization plateau of the cross-validation function, we have chosen numbers of bins that are rounded to humanly pleasing numbers, that fit the range nicely, and that are close to the suggested values. The selected number of bins is shown in Table 6.2.

For the mean-value observables using profile histograms, the number of bins is selected to match the observable the mean is plotted against, which in our case is invariant mass, which is the same as M(WZ).

Table 6.2: Observables for parton-level WZ-production, optimal bins by different methods, and selected bins.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Observable** | **Scott’s Bins** | **Freedman- Diaconis Bins** | **Cross-Validation Bins** | **Selected Bins** |
| **Phase-space observables** | | | | | |
| 1 | PT(Z) | 952 | 1700 | 900 | 750 |
| 2 | M(WZ) = | 709 | 1510 | 1750 | 1500 |
| 3 | y(Z) | 139 | 153 | 175 | 200 |
| **Optimal observables** | | | | | |
| 4 | O1(cWWW) | 394 | 1220 | 2000 | 2000 |
| 5 | O2(cWWW) | 16300 | 1.68×107 | > 1E6 | 100000 |
| 6 | O1(cW) | 6110 | 39200 | 17500 | 10000 |
| 7 | O2(cW) | 16200 | 6.53×106 | > 1E6 | 100000 |
| 8 | O1(cB) | 338 | 812 | 1250 | 1000 |
| 9 | O2(cB) | 6190 | 74900 | 325000 | 100000 |
| **Mean optimal observables vs. invariant mass** | | | | | |
| 10 | vs | Same as M(WZ) | | | |
| 11 | vs |
| 12 | vs |
| 13 | vs |
| 14 | vs |
| 15 | vs |
| **Mean optimal observables divided by powers of invariant mass vs. invariant mass** | | | | | |
| 16 | vs | Same as M(WZ) | | | |
| 17 | vs |
| 18 | vs |
| 19 | vs |
| 20 | vs |
| 21 | vs |

## Binning for unbinned fits

TODO

# Fitting EFT to SM for parton-level WZ-production

Confident that we have the reweighting tools to construct the observable plots for any defined set of EFT model parameters, we will now measure the fit characteristics of reweighting EFT to SM data at the parton-level. Specifically, we measure how well reweighted EFT data fits the SM data for each selected observable, using a fit algorithm to determine the EFT model parameters that fit best. This is done using the same parton-level WZ-production pseudo-data generated by Sherpa used previously.

The SM and EFT data samples both have 1M events. Their total cross-sections are 18.554 ± 0.016 pb for SM and 42.805 ± 0.038 pb for EFT. The data samples are scaled to a luminosity of 10 fb–1, and Asimov bin errors are used as discussed in 4.2.1.

Log-likelihood minimization is used to fit to event counted observables, and -minimization is used to fit to mean-observables. The results are shown in Table XXX.

Observables

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Observable** | **Type** | **Bin Range** | **Bins** | **Fit Bins a** | **Underflow**  **Count b** | **Overflow**  **Count b** |
| **PT(Z)** | count | 0 to 750 GeV/c | 150 | 150 | 0 | 8.2 |
| **M(WZ) =** | count | 0 to 3000 GeV/c2 | 150 | 142 | 0 | 6.3 |
| **y(Z)** | count | −5 to +5 | 100 | 98 | 0 | 0 |
| **O1(cWWW)** | count | −60000 to 7000 | 1000 | 1000 | 6.7 | 3.4 |
| **O2(cWWW)** | count | 0 to 2×1012 | 1000 | 1000 | 0 | 5.9 |
| **O1(cW)** | count | (−200 to 2)×104 | 1000 | 997 | 2.7 | 0 |
| **O2(cW)** | count | 0 to 6×1011 | 1000 | 1000 | 0 | 7.6 |
| **O1(cB)** | count | −800 to 5000 | 1000 | 996 | 1.5 | 8.2 |
| **O2(cB)** | count | 0 to 2×108 | 1000 | 1000 | 0 | 3.7 |
| **O1(cWWW) vs** | mean | 0 to 3000 GeV | 100 | 76 | 0 | 6.3 |
| **O2(cWWW) vs** | mean | 0 to 3000 GeV | 100 | 76 | 0 | 6.3 |
| **O1(cW) vs** | mean | 0 to 3000 GeV | 100 | 76 | 0 | 6.3 |
| **O2(cW) vs** | mean | 0 to 3000 GeV | 100 | 76 | 0 | 6.3 |
| **O1(cB) vs** | mean | 0 to 3000 GeV | 100 | 76 | 0 | 6.3 |
| **O2(cB) vs** | mean | 0 to 3000 GeV | 100 | 76 | 0 | 6.3 |

1. Fit skips bins with a zero error value. For mean observables, fit also skips bins with less than 10 effective events in either the SM or EFT sample prior to any luminosity scaling.
2. SM sample at 10 fb−1.

Table 7.1: Fit results for binned (B) and unbinned (U) fits of cWWW

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Observable** | | **Fit Type** | **Fit cWWW** | | | **Fit All** | | |
| **Value**  **[10−6 GeV2]** | **Error**  **[10−6 GeV2]** |  | **Value**  **[10−6 GeV2]** | **Error**  **[10−6 GeV2]** |  |
| **Phase-space observables** | | | | | | | | |
| 1 | PT(Z) | B | −0.25 | 0.33 | 0.27 | 0.053 | 0.58 | 0.27 |
|  |  | U | −0.25 | 0.33 |  | 0.053 | 0.58 |  |
| 2 | M(WZ) = | B | −1.18 | 0.18 | 0.50 | 0.10 | 2.1 | 0.47 |
|  |  | U | −1.18 | 0.18 |  | 0.10 | 2.1 |  |
| 3 | y(Z) | B | −0.98 | 0.73 | 0.61 | 0.60 | 3.2 | 0.62 |
|  |  | U | −0.98 | 0.73 |  | 0.60 | 3.2 |  |
| **Optimal observables** | | | | | | | | |
| 4 | O1(cWWW) | B | −0.29 | 0.28 | 0.36 | 0.37 | 0.89 | 0.36 |
|  |  | U | −0.29 | 0.28 |  | 0.37 | 0.89 |  |
| 5 | O2(cWWW) | B | −0.30 | 0.25 | 0.058 | 0.42 | 0.48 | 0.057 |
|  |  | U | −0.30 | 0.25 |  | 0.42 | 0.48 |  |
| 6 | O1(cW) | B | −0.21 | 0.23 | 0.099 | 0.16 | 0.69 | 0.099 |
|  |  | U | −0.21 | 0.23 |  | 0.16 | 0.69 |  |
| 7 | O2(cW) | B | −0.35 | 0.23 | 0.076 | 0.51 | 0.44 | 0.074 |
|  |  | U | −0.35 | 0.23 |  | 0.51 | 0.44 |  |
| 8 | O1(cB) | B | −0.28 | 0.28 | 0.40 | 0.32 | 0.98 | 0.40 |
|  |  | U | −0.28 | 0.28 |  | 0.32 | 0.98 |  |
| 9 | O2(cB) | B | −0.24 | 0.24 | 0.086 | 0.17 | 0.67 | 0.087 |
|  |  | U | −0.24 | 0.24 |  | 0.17 | 0.67 |  |
| **Mean optimal observables vs. invariant mass** | | | | | | | | |
| 10 | vs | B | 0.0028 | 0.077 | 0.19 | −0.00027 | 0.11 | 0.065 |
| 11 | vs | B | 0.0029 | 0.018 | 3.3 | −0.00024 | 0.026 | 0.87 |
| 12 | vs | B | 0.0028 | 0.021 | 2.5 | −0.00024 | 0.029 | 0.69 |
| 13 | vs | B | 0.0028 | 0.020 | 2.7 | −0.00024 | 0.028 | 0.74 |
| 14 | vs | B | 0.0029 | 0.13 | 0.080 | −0.00032 | 0.17 | 0.035 |
| 15 | vs | B | 0.0028 | 0.12 | 0.091 | −0.00030 | 0.16 | 0.037 |
| **Mean optimal observables divided by powers of invariant mass vs. invariant mass** | | | | | | | | |
| 16 | vs | B |  |  |  |  |  |  |
| 17 | vs | B |  |  |  |  |  |  |
| 18 | vs | B |  |  |  |  |  |  |
| 19 | vs | B |  |  |  |  |  |  |
| 20 | vs | B |  |  |  |  |  |  |
| 21 | vs | B |  |  |  |  |  |  |

cW

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Observable** | **Fit cW** | | | **Fit All** | | |
| **Value**  **[10−6 GeV2]** | **Error**  **[10−6 GeV2]** |  | **Value**  **[10−6 GeV2]** | **Error**  **[10−6 GeV2]** |  |
| **PT(Z)** | −0.11 | 0.13 | 0.27 | −0.16 | 0.20 | 0.27 |
| **M(WZ)** | −1.1 | 0.24 | 0.46 | −1.42 | 0.37 | 0.47 |
| **y(Z)** | −0.41 | 0.37 | 0.61 | −1.6 | 3.2 | 0.62 |
| **O1(cWWW)** | −0.10 | 0.11 | 0.36 | 0.020 | 0.39 | 0.36 |
| **O2(cWWW)** | −0.10 | 0.12 | 0.059 | 0.18 | 0.50 | 0.057 |
| **O1(cW)** | −0.069 | 0.095 | 0.099 | 0.0081 | 0.24 | 0.099 |
| **O2(cW)** | −0.12 | 0.11 | 0.077 | 0.23 | 0.50 | 0.074 |
| **O1(cB)** | −0.095 | 0.11 | 0.40 | −0.0091 | 0.36 | 0.40 |
| **O2(cB)** | −0.086 | 0.10 | 0.087 | −0.012 | 0.25 | 0.087 |
| **O1(cWWW) vs** | 0.44 | 0.17 | 0.063 | 0.44 | 0.17 | 0.065 |
| **O2(cWWW) vs** | 0.457 | 0.041 | 0.85 | 0.456 | 0.041 | 0.87 |
| **O1(cW) vs** | 0.456 | 0.046 | 0.67 | 0.455 | 0.047 | 0.69 |
| **O2(cW) vs** | 0.456 | 0.045 | 0.72 | 0.455 | 0.045 | 0.74 |
| **O1(cB) vs** | 0.43 | 0.27 | 0.034 | 0.43 | 0.28 | 0.035 |
| **O2(cB) vs** | 0.43 | 0.25 | 0.036 | 0.43 | 0.25 | 0.037 |

cB

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Observable** | **Fit cB** | | | **Fit All** | | |
| **Value**  **[10−6 GeV2]** | **Error**  **[10−6 GeV2]** |  | **Value**  **[10−6 GeV2]** | **Error**  **[10−6 GeV2]** |  |
| **PT(Z)** | 2.4 | 5.3 | 0.26 | −2.3 | 10 | 0.27 |
| **M(WZ)** | 23 | 12 | 0.47 | −27 | 29 | 0.47 |
| **y(Z)** | 12 | 11 | 0.61 | −39 | 140 | 0.62 |
| **O1(cWWW)** | 3.8 | 3.8 | 0.36 | 7.3 | 17 | 0.36 |
| **O2(cWWW)** | 7.0 | 6.4 | 0.058 | 17 | 22 | 0.057 |
| **O1(cW)** | 3.4 | 3.9 | 0.099 | 5.0 | 13 | 0.099 |
| **O2(cW)** | 7.3 | 5.5 | 0.075 | 19 | 20 | 0.074 |
| **O1(cB)** | 3.4 | 3.8 | 0.40 | 5.7 | 17 | 0.40 |
| **O2(cB)** | 4.1 | 4.4 | 0.086 | 5.1 | 13 | 0.087 |
| **O1(cWWW) vs** | −6.2 | 12 | 0.19 | −0.97 | 17 | 0.065 |
| **O2(cWWW) vs** | −6.3 | 2.9 | 3.2 | −0.94 | 4 | 0.87 |
| **O1(cW) vs** | −6.3 | 3.3 | 2.5 | −0.94 | 4.5 | 0.69 |
| **O2(cW) vs** | −6.3 | 3.1 | 2.7 | −0.94 | 4.3 | 0.74 |
| **O1(cB) vs** | −6.2 | 20 | 0.079 | −0.91 | 27 | 0.035 |
| **O2(cB) vs** | −6.2 | 18 | 0.090 | −0.92 | 25 | 0.037 |

# Fitting EFT to SM for simulated measurement of WZ-production

# Conclusion

# Bibliography