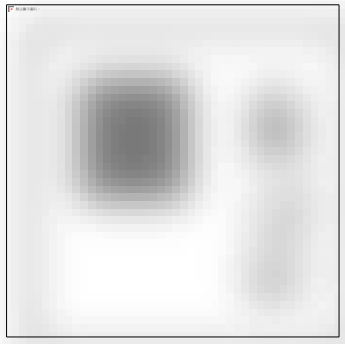


DY Cross Section Systematics 20250624

Chia-Yu Hsieh

Academia Sinica, Taiwan

Input Data

icol=0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
	M	qT	xF	xsec (boot)	err(stat)	error_syst(purity)	error_syst(acc1)	error_syst(acc2)	error_syst(lumi)	err_syst(acc)	xsec (boot)	err(stat)	error_syst(purity)	error_syst(acc1)	error_syst(acc2)	error_syst(lumi)	err_sy
irow=0	4.3 - 4.7	0 - 0.7	-0.2 : -0.1	0.0968	0.0327	0.0000	0.0032	0.0106	0.0032	0.0110							
1	4.3 - 4.7	0 - 0.7	-0.1 : 0.0	0.0668	0.0095	0.0013	0.0008	0.0032	0.0022	0.0033							
2	4.3 - 4.7	0 - 0.7	0.0 : 0.1	0.0879	0.0073	0.0015	0.0006	0.0027	0.0029	0.0027							
3	4.3 - 4.7	0 - 0.7	0.1 : 0.2	0.0864	0.0061	0.0015	0.0005	0.0047	0.0029	0.0047	0.0794	0.0214	0.0005	0.0017	0.0040	0.0026	
4	4.3 - 4.7	0 - 0.7	0.2 : 0.3	0.0812	0.0056	0.0015	0.0005	0.0028	0.0027	0.0028	0.0881	0.0108	0.0027	0.0009	0.0029	0.0029	
5	4.3 - 4.7	0 - 0.7	0.3 : 0.4	0.0641	0.0053	0.0009	0.0004	0.0019	0.0021	0.0020	0.0772	0.0064	0.0023	0.0006	0.0023	0.0026	
6	4.3 - 4.7	0 - 0.7	0.4 : 0.5	0.0611	0.0062	0.0005	0.0005	0.0020	0.0020	0.0021	0.0631	0.0047	0.0009	0.0004	0.0021	0.0021	
7	4.3 - 4.7	0 - 0.7	0.5 : 0.6	0.0568	0.0073	0.0003	0.0007	0.0023	0.0019	0.0024	0.0631	0.0045	0.0003	0.0004	0.0026	0.0021	
8	4.3 - 4.7	0 - 0.7	0.6 : 0.7	0.0391	0.0079	0.0000	0.0007	0.0022	0.0013	0.0023	0.0445	0.0038	0.0004	0.0004	0.0026	0.0015	
9	4.3 - 4.7	0 - 0.7	0.7 : 0.8	0.0324	0.0099	0.0003	0.0010	0.0015	0.0011	0.0018	0.0283	0.0031	0.0000	0.0003	0.0016	0.0009	
10	4.3 - 4.7	0 - 0.7	0.8 : 0.9								0.0149	0.0023	0.0000	0.0002	0.0009	0.0005	
11	4.3 - 4.7	0 - 0.7	0.9 : 1.0														

```python
# ---------- read excel file -----------------
# pd.read_excel reads an Excel file into a DataFrame (f).
# file_path is the path to your Excel file.
file_path = 'COMPASS_2.xlsx'
df = pd.read_excel(file_path)

col_indices = {
    'LL_xsc' : 5,
    'LL_stat': 6,
    'LL_sysP': 7,
    'LL_sysL': 10,
    'LL_sysA': 11,
    'LO_xsc' : 12,
    'LO_stat': 13,
    'LO_sysP': 14,
    'LO_sysL': 17,
    'LO_sysA': 18,
    'xf'     : 22,
    'xf_err' : 23,
    'M'      : 2,
    'qT'     : 3
}
```

Educated Guess (To be Discussed) : Covariance Matrix of Purity

```
purCorr_crossDia_LL     = 1.0 # = 1.0, okay, same bin same trigger
purCorr_crossOff_LL     = 0.0 # = 0.0, but one should test (0, 1), different bins same trigger
purCorr_crossDia_LO     = 1.0 # = 1.0, okay, same bin same trigger
purCorr_crossOff_LO     = 0.0 # = 0.0, but one should test (0, 1), different bins same trigger
purCorr_crossDia_LL_LO  = 0.0 # = 0, okay?, same bins different triggers
purCorr_crossOff_LL_LO  = 0.0 # = 0, okay?, different bins different triggers
```

```
#--- Systematic Source 2: Purity(?) -------------
#[[1.0 0.0]
# [0.0 1.0]]
diag_val     = purCorr_crossDia_LL  # (=1.0)
off_diag_val = purCorr_crossOff_LL  # (=0.0)
LL_corrP = generate_custom_matrix(N, diag_val, off_diag_val) # LL : diagonal covariance matrix of purity for LL, no correlation between bins
LO_corrP = generate_custom_matrix(N, diag_val, off_diag_val) # LO : diagonal covariance matrix of purity for LO, no correlation between bins
#print("LL purity covariance matrix, LL_corrP = ", LL_corrP)
#print("LO purity covariance matrix, LO_corrP = ", LO_corrP)
diag_val     = purCorr_crossDia_LL_LO # = 0.0
off_diag_val = purCorr_crossOff_LL_LO # = 0.0
cross_corrP = generate_custom_matrix(N, diag_val, off_diag_val) # cross correlation of purity between LL and LO, no correlation between bins
#print("cross correlation of purity between LL and LO, cross_corr2 = ", cross_corrP)
```

```
#- -- generate_custom_matrix ------------------
def generate_custom_matrix(N, diagonal_value, off_diagonal_value):
    matrix = np.full((N, N), off_diagonal_value)
    np.fill_diagonal(matrix, diagonal_value)
    return matrix
```

Educated Guess (To be Discussed) : Covariance Matrix of Acceptance

```
accCorr_scale_LL        = 2.0 # all random for both diagonal and off-diagonal? not realistic   <=========
accCorr_scale_LO        = 0.8 # all random for both diagonal and off-diagonal? not realistic   <=========
accCorr_crossDia_LL_LO  = 0.5 # = 0.5, okay, cross means different bins and different triggers
accCorr_crossOff_LL_LO  = 0.5 # = 0.5, but one should test (0, 1)
```

```
#--- Systematic Source 1: Acceptance -------------
# The covariance matrix for the acceptance systematic error is generated using a Gaussian correlation function.
# The function takes the number of variables (N) and a scale parameter (scale) as inputs.
# The correlation matrix is generated using a Gaussian function, which is commonly used in statistics and machine learning to model correlations between variables.
# The scale parameter controls the width of the Gaussian function, affecting how quickly the correlation decreases with distance.
# The correlation matrix is symmetric and positive definite, which is a requirement for covariance matrices.
# Dynamic Range 'scale' (s):
# Big s (e.g., 5): slow correlation falloff, distant bins correlated
# Small s (e.g., 0.5): rapid correlation falloff, only nearby bins correlated
LL_corrA = gaussian_corr(N, scale=accCorr_scale_LL) # LL : random covariance matrix for LL, weak correlation in nearby bins but still correlated in distant bins
LO_corrA = gaussian_corr(N, scale=accCorr_scale_LO) # LO : random covariance matrix for LO, stronger correlation in nearby bins but weak correlated in distant bins
#print("LL acceptance covariance matrix, LL_corrA = ", LL_corrA)
#print("LO acceptance covariance matrix, LO_corrA = ", LO_corrA)

# correlation of different bins, same bin = 1.0 and different bins = 0.5
#[[0.5 0.5]
# [0.5 0.5]]
diag_val     = accCorr_crossDia_LL_LO     #(=0.5)
off_diag_val = accCorr_crossOff_LL_LO #(=0.5)
cross_corrA  = generate_custom_matrix(N, diag_val, off_diag_val)
#print("cross correlation of acceptance between LL and LO, cross_corrA = ", cross_corrA)
```

```
# -- generate_correlation_matrix ------------------
def gaussian_corr(N, scale):
    # Create an array of bin indices: [0, 1, 2, ..., N-1]
    idx = np.arange(N)

    # Create a matrix of differences between bin indices
    # idx[:, None] makes it a column vector (N x 1)
    # idx[None, :] makes it a row vector (1 x N)
    # The subtraction builds an (N x N) matrix where element (i,j) = i - j
    diff_matrix = idx[:, None] - idx[None, :]

    # Normalize the differences by 'scale' and square them
    normalized_squared = (diff_matrix / scale) ** 2

    # Apply the Gaussian formula:
    # correlation(i,j) = exp( -0.5 * (normalized difference)^2 )
    corr_matrix = np.exp(-0.5 * normalized_squared)

    return corr_matrix
```

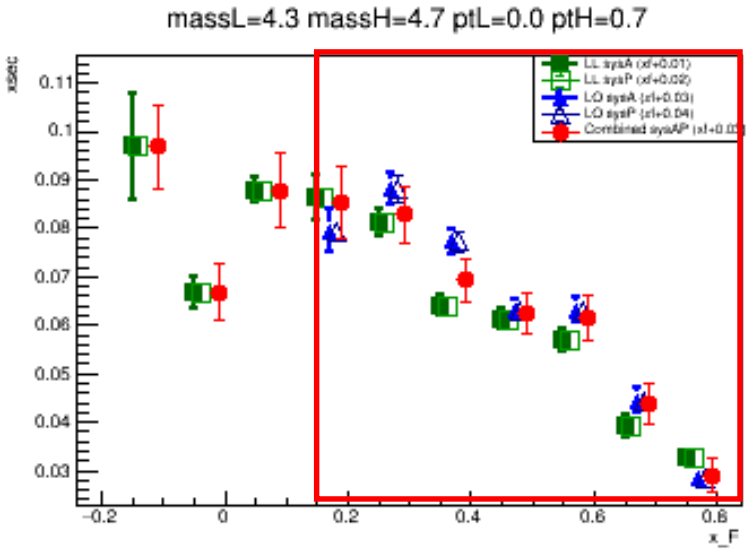
Build up Full Covariance Matrix
(Only xf Overlapped Region)

```python
# ----  combine purity and acc covariance matrix ----
filtered_LL_sys = [filtered_LL_both_sysA, filtered_LL_both_sysP] # 1D = [array(sys_A_LL), array(sys_P_LL)]
filtered_LO_sys = [filtered_LO_both_sysA, filtered_LO_both_sysP] # 1D = [array(sys_A_LO), array(sys_P_LO)]
#print("LL systematic sources, sys_sources_LL = ", LL_sys)
#print("LO systematic sources, sys_sources_LO = ", LO_sys)

LL_corr = [LL_corrA, LL_corrP] # 2D = [array(cov_A_LL), array(cov_P_LL)]
LO_corr = [LO_corrA, LO_corrP] # 2D = [array(cov_A_LO), array(cov_P_LO)]
cross_corr = [cross_corrA, cross_corrP] # 2D = [array(cov_A_LL_LO), array(cov_P_LL_LO)]
#print("LL covariance matrices, LL_corr = ", LL_corr)
#print("LO covariance matrices, LO_corr = ", LO_corr)
#print("cross covariance matrices, cross_corr = ", cross_corr)

#-------- Build Full Covariance Matrix ------------------
# The full covariance matrix is built using the function build_cov.
# It takes the systematic uncertainties, correlation matrices, and statistical uncertainties as inputs.
# The function returns the full covariance matrix, as well as the self-covariance matrices for each measurement and the cross-covariance matrix.
# The self-covariance matrices are the covariance matrices for each measurement, while the cross-covariance matrix represents the covariance between the two measurements.
# The full covariance matrix is a 2N x 2N matrix, where N is the number of bins.
full_cov, cov1, cov2, cross = build_cov(filtered_LL_sys, filtered_LO_sys, LL_corr, LO_corr, cross_corr, filtered_LL_both_stat, filtered_LO_both_stat)
np.set_printoptions(precision=6, suppress=True)
#print("Full 4x4 covariance matrix:\n", full_cov)
#print("\nCovariance of Measurement 1 (cov1):\n", cov1)
#print("\nCovariance of Measurement 2 (cov2):\n", cov2)
#print("\nCross-covariance matrix:\n", cross)
```

```python
def build_cov(LL_sys, LO_sys, LL_corr, LO_corr, cross_corr, LL_stat, LO_stat):
    """
    Build the full covariance matrix for two measurements with shared systematic sources.

    Parameters:
    - LL_sys: list of 1D arrays, each of shape (N,), representing uncertainties from each shared systematic source for measurement 1.
    - LO_sys: list of 1D arrays, same as LL_sys but for measurement 2.
    - LL_corr: list of 2D arrays (N x N), bin-to-bin correlation matrices for measurement 1.
    - LO_corr: list of 2D arrays (N x N), bin-to-bin correlation matrices for measurement 2.
    - cross_corr: list of 2D arrays (N x N), bin-to-bin correlation matrices between measurement 1 and 2.
    - LL_stat: 1D array of statistical uncertainties for measurement 1 (length N).
    - LO_stat: 1D array of statistical uncertainties for measurement 2 (length N).

    Returns:
    - cov: full 2N x 2N covariance matrix
    - cov1: N x N covariance matrix for measurement 1 (self-covariance)
    - cov2: N x N covariance matrix for measurement 2 (self-covariance)
    - cross: N x N cross-covariance matrix between LL and LO
    - cross.T  (N x N): Transpose of cross-covariance to ensure matrix symmetry
    # +-----------------+-----------------+
    # |   cov1 (N x N)  |   cross (N x N)    |
    # +-----------------+-----------------+
    # | cross.T (N x N) |   cov2 (N x N)     |
    # +-----------------+-----------------+
    """
```

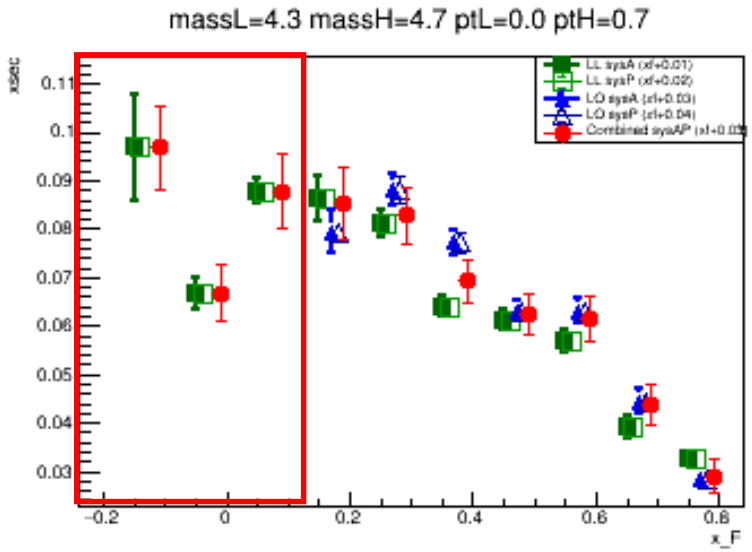
Merge LL and LO
(Only xf Overlapped Region)

```
# --------- Merge ----------
# The merge_measurements function combines two measurements (LL and LO) into a single measurement (m_comb) using the full covariance matrix (full_cov).
# The function returns the combined measurement and its covariance matrix.
# The combined measurement is a weighted average of the two measurements, where the weights are determined by the uncertainties in each measurement.
# The covariance matrix of the combined measurement is calculated using the inverse of the full covariance matrix.
# The combined measurement and its covariance matrix are returned as outputs.
# The combined measurement is a 2N x 1 vector, where N is the number of bins.
# The covariance matrix of the combined measurement is a 2N x 2N matrix.

m_comb, cov_comb = merge_measurements(filtered_LL_both_xsc, filtered_LO_both_xsc, full_cov)
combined_both_xsc    = m_comb
combined_both_sysAP  = np.sqrt(np.diag(cov_comb))
combined_both_ratio  = combined_both_sysAP / combined_both_xsc
combined_both_stat   = np.sqrt(filtered_LL_both_stat**2 + filtered_LO_both_stat**2)
combined_both_sysAPL = np.sqrt(combined_both_sysAP**2 + filtered_LL_both_sysL**2 + filtered_LO_both_sysL**2)
```

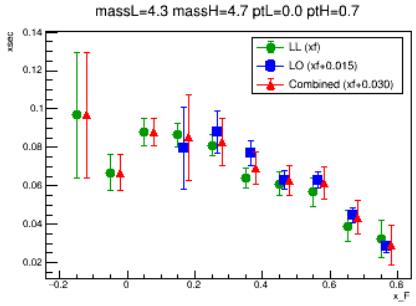


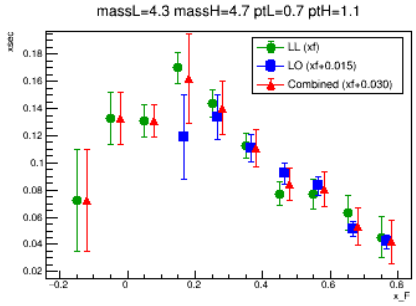
Merge LL and LO
(Bins with LL or LO Only)

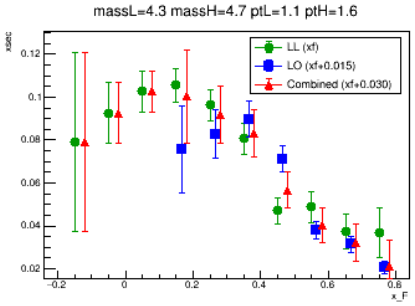
```
# --- Filter Data for the bins has LL only ---
N_ll_only = len(ll_only_indices)

filtered_LL_ll_only_xsc  = LL_xsc [ll_only_indices]
filtered_LL_ll_only_stat = LL_stat[ll_only_indices]
filtered_LL_ll_only_sysA = LL_sysA[ll_only_indices]
filtered_LL_ll_only_sysP = LL_sysP[ll_only_indices]
filtered_LL_ll_only_sysL = LL_sysL[ll_only_indices]


combined_ll_only_xsc     = filtered_LL_ll_only_xsc
combined_ll_only_sysAP   = filtered_LL_ll_only_xsc * combined_both_ratio[0]  # Use the first ratio for LL only
combined_ll_only_stat    = filtered_LL_ll_only_stat
combined_ll_only_sysAPL  = np.sqrt(combined_ll_only_sysAP**2 + filtered_LL_ll_only_sysL**2)
```

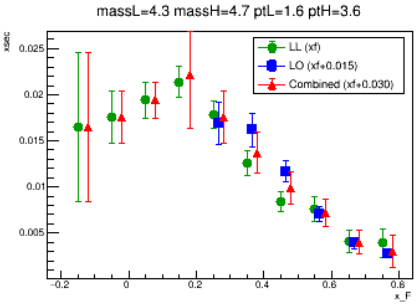


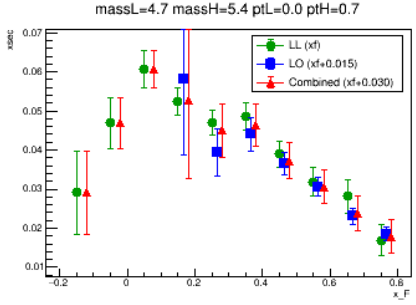
Statistic Error

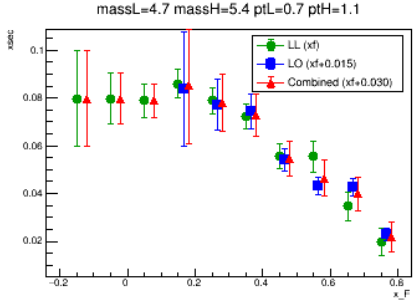


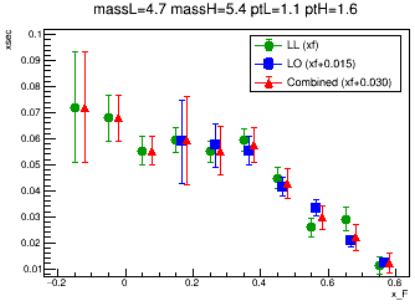


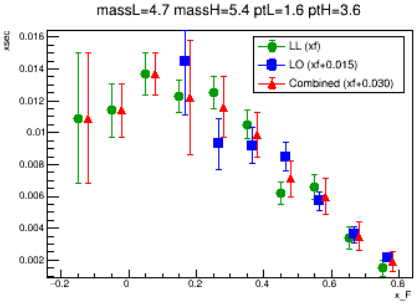


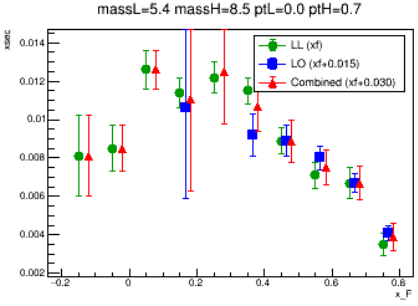


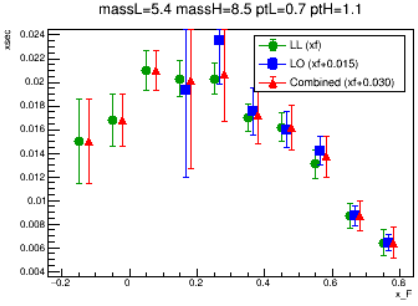


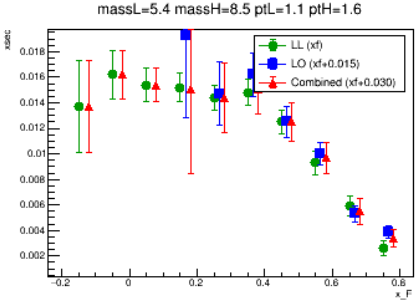


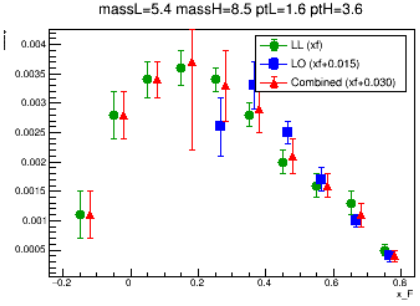




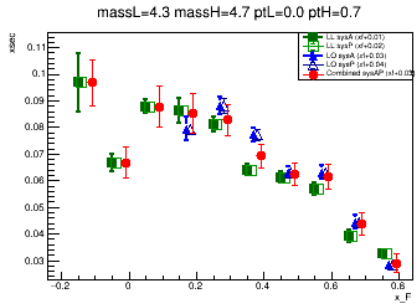


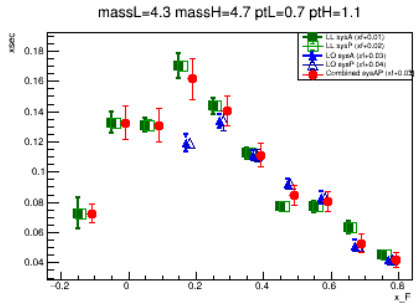


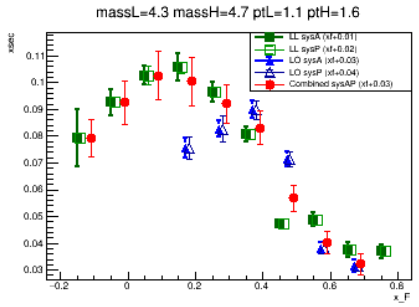


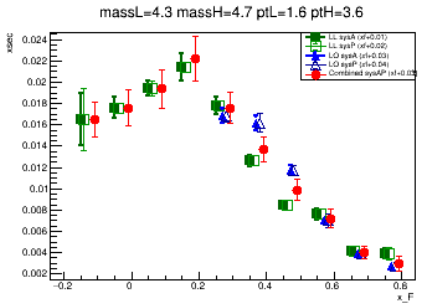


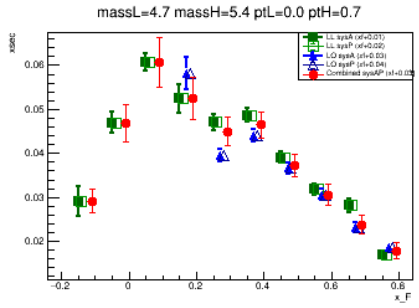
Systematic Error (Acc + Purity, w/o Luminosity Sys.)

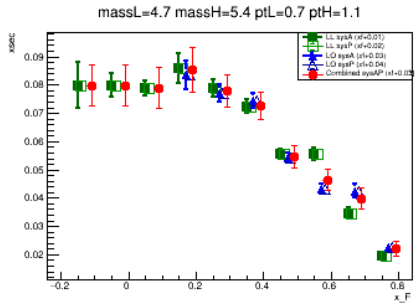


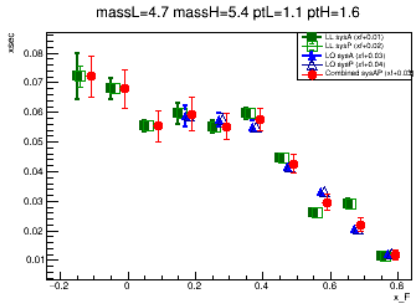


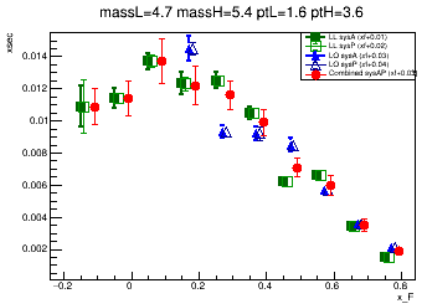


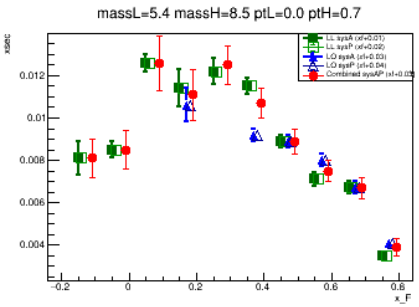


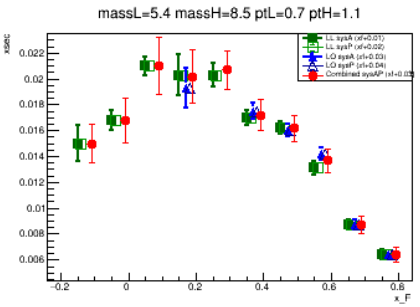


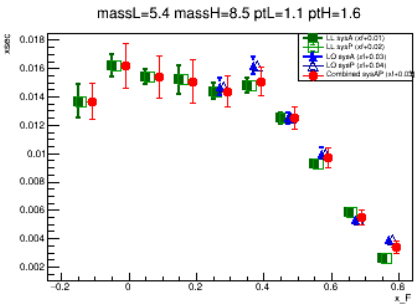


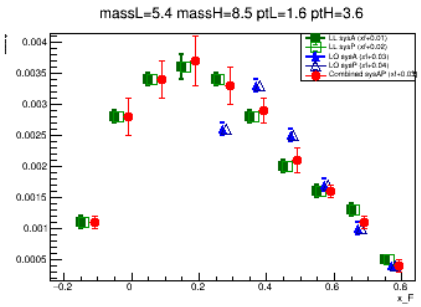




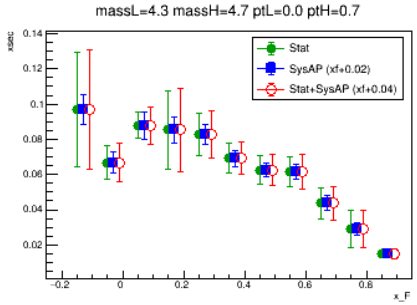


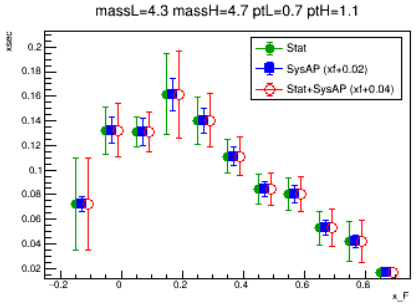


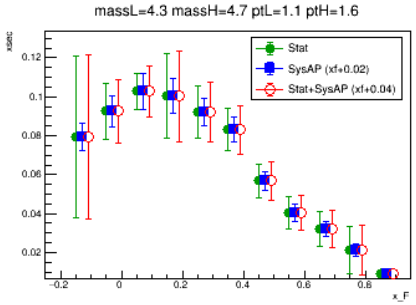


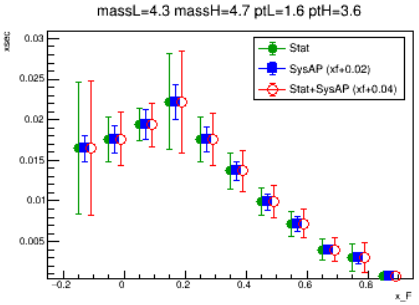


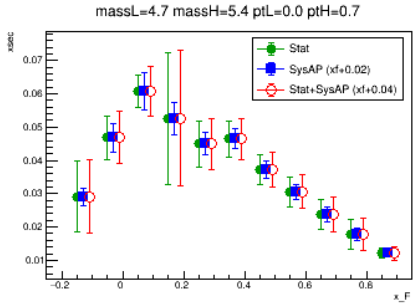
Final Results (w/ Luminosity Sys.)

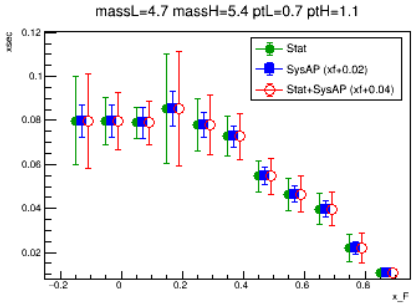


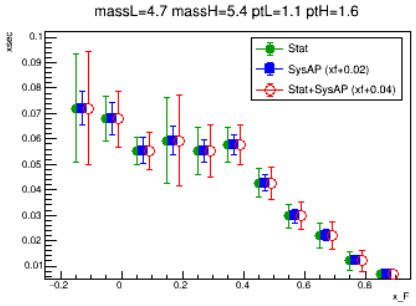


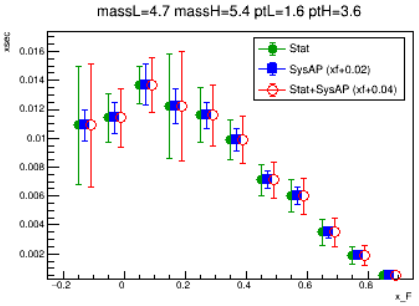


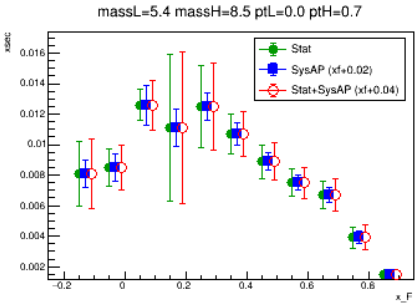


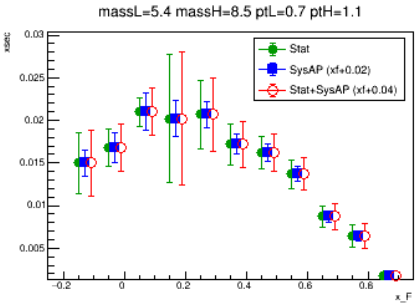


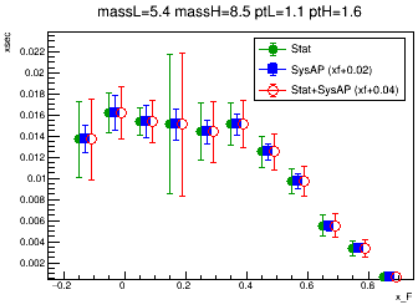


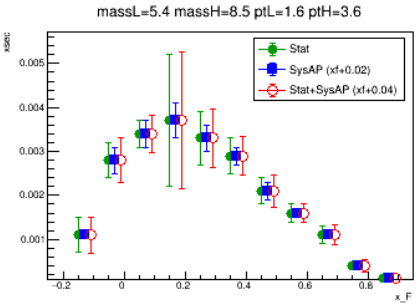












To Do

- Anything need to be improved in terms of method?
- Test correlated values for purity

```
purCorr_crossDia_LL     = 1.0 # = 1.0, okay, same bin same trigger
purCorr_crossOff_LL     = 0.0 # = 0.0, but one should test (0, 1), different bins same trigger
purCorr_crossDia_LO     = 1.0 # = 1.0, okay, same bin same trigger
purCorr_crossOff_LO     = 0.0 # = 0.0, but one should test (0, 1), different bins same trigger
purCorr_crossDia_LL_LO  = 0.0 # = 0, okay?, same bins different triggers
purCorr_crossOff_LL_LO  = 0.0 # = 0, okay?, different bins different triggers
```

(0, 0.5, 1.0)

(0, 0.5, 1.0)

- Test correlated values for acceptance

```
accCorr_scale_LL        = 2.0 # all random for both diagonal and off-diagonal? not realistic   <=========
accCorr_scale_LO        = 0.8 # all random for both diagonal and off-diagonal? not realistic   <=========
accCorr_crossDia_LL_LO  = 0.5 # = 0.5, okay, cross means different bins and different triggers
accCorr_crossOff_LL_LO  = 0.5 # = 0.5, but one should test (0, 1)
```

?

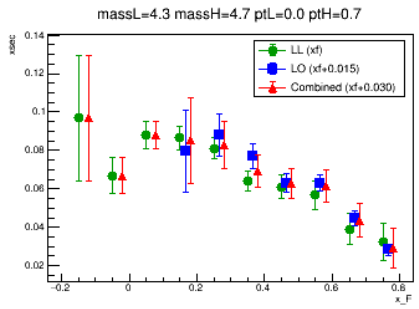
Test Correlated of Purity Sys.

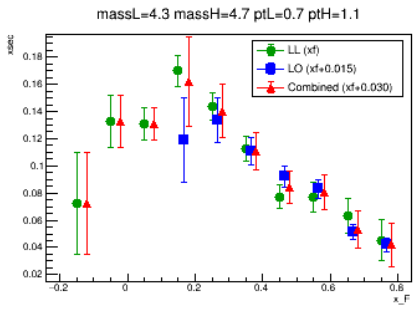
```
purCorr_crossDia_LL     = 1.0 # = 1.0, okay, same bin same trigger
purCorr_crossOff_LL     = 0.0 # = 0.0, but one should test (0, 1), different bins same trigger
purCorr_crossDia_LO     = 1.0 # = 1.0, okay, same bin same trigger
purCorr_crossOff_LO     = 0.0 # = 0.0, but one should test (0, 1), different bins same trigger
purCorr_crossDia_LL_LO  = 0.0 # = 0, okay?, same bins different triggers
purCorr_crossOff_LL_LO  = 0.0 # = 0, okay?, different bins different triggers
```

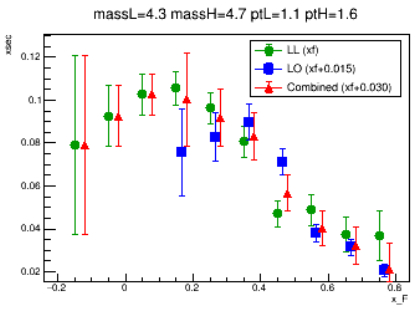
(0, 0.5, 1.0)

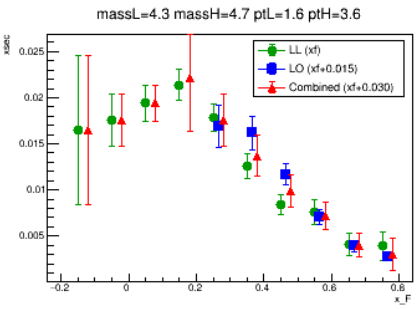
(0, 0.5, 1.0)

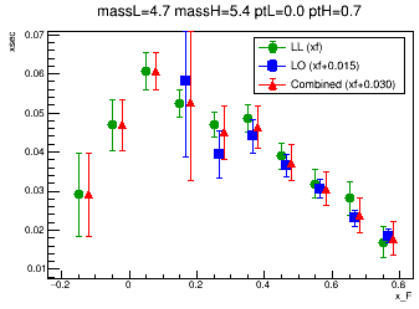
Stat Err : 0.0

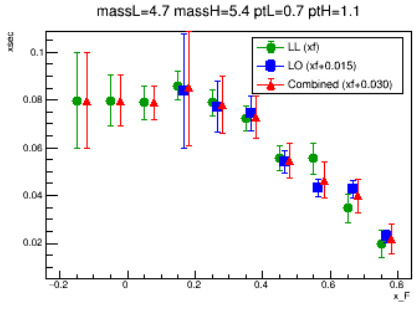

massL=4.3 massH=4.7 ptL=0.0 pthH=0.7
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

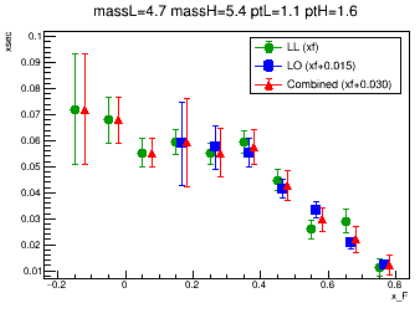

massL=4.3 massH=4.7 ptL=0.7 pthH=1.1
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

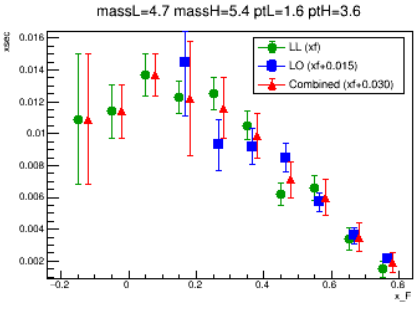

massL=4.3 massH=4.7 ptL=1.1 pthH=1.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

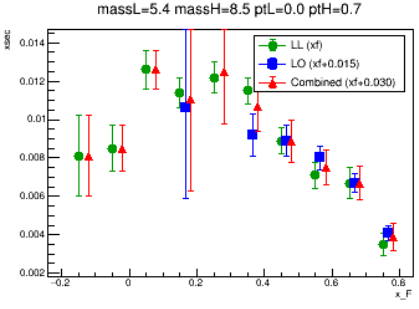

massL=4.3 massH=4.7 ptL=1.6 pthH=3.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

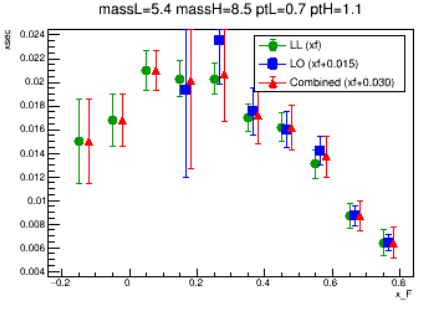

massL=4.7 massH=5.4 ptL=0.0 pthH=0.7
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

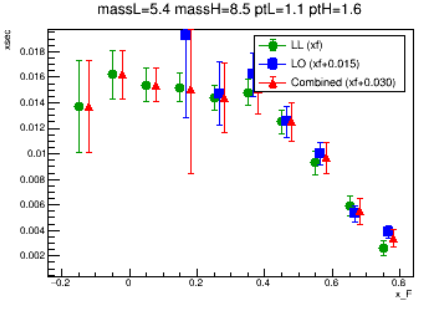

massL=4.7 massH=5.4 ptL=0.7 pthH=1.1
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

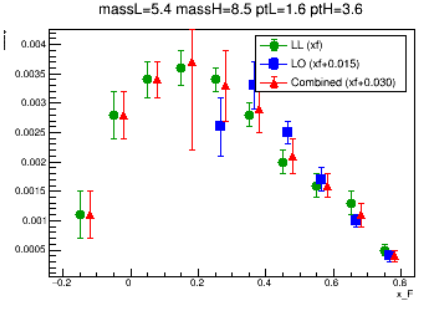

massL=4.7 massH=5.4 ptL=1.1 pthH=1.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F


massL=4.7 massH=5.4 ptL=1.6 pthH=3.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

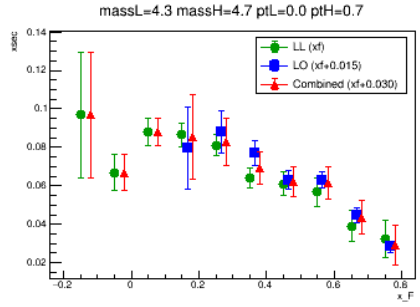

massL=5.4 massH=8.5 ptL=0.0 pthH=0.7
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

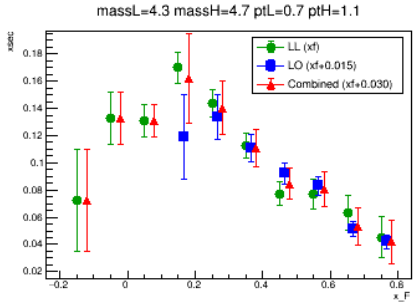

massL=5.4 massH=8.5 ptL=0.7 pthH=1.1
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

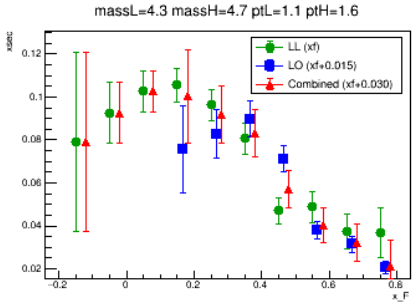

massL=5.4 massH=8.5 ptL=1.1 pthH=1.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

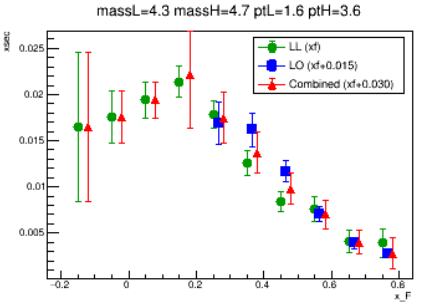

massL=5.4 massH=8.5 ptL=1.6 pthH=3.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

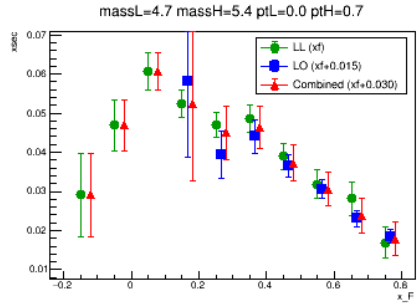
Stat Err : 0.5

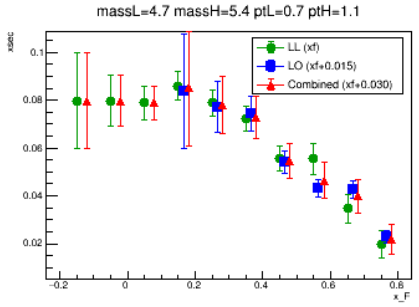

massL=4.3 massH=4.7 ptL=0.0 pthH=0.7
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

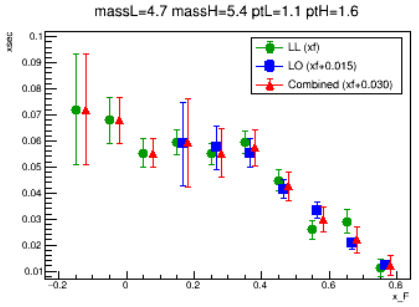

massL=4.3 massH=4.7 ptL=0.7 pthH=1.1
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

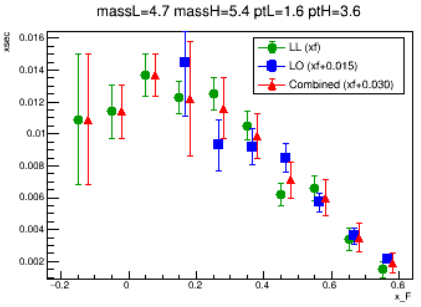

massL=4.3 massH=4.7 ptL=1.1 pthH=1.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

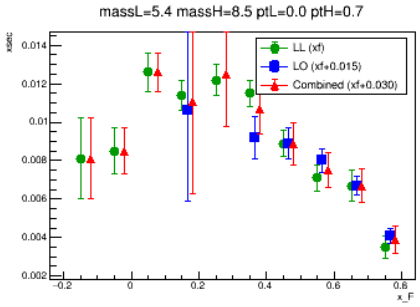

massL=4.3 massH=4.7 ptL=1.6 pthH=3.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

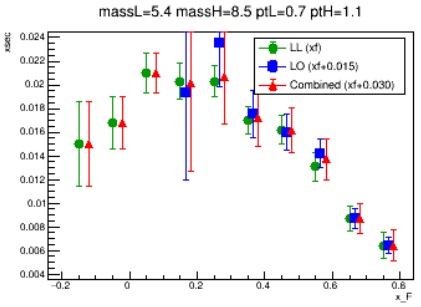

massL=4.7 massH=5.4 ptL=0.0 pthH=0.7
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

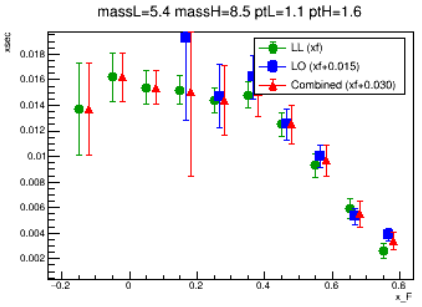

massL=4.7 massH=5.4 ptL=0.7 pthH=1.1
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

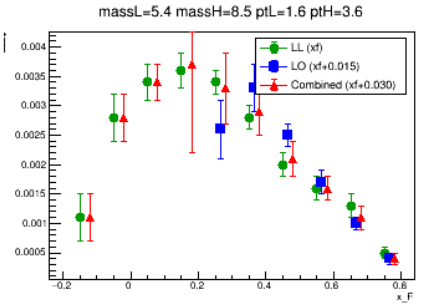

massL=4.7 massH=5.4 ptL=1.1 pthH=1.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F


massL=4.7 massH=5.4 ptL=1.6 pthH=3.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

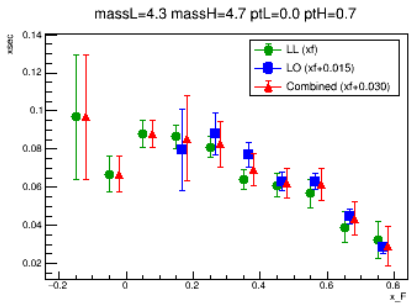

massL=5.4 massH=8.5 ptL=0.0 pthH=0.7
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

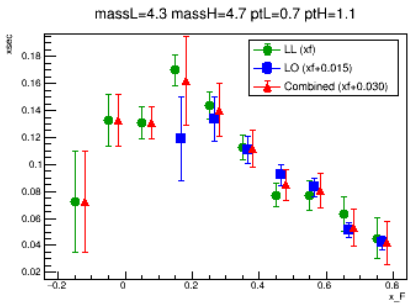

massL=5.4 massH=8.5 ptL=0.7 pthH=1.1
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

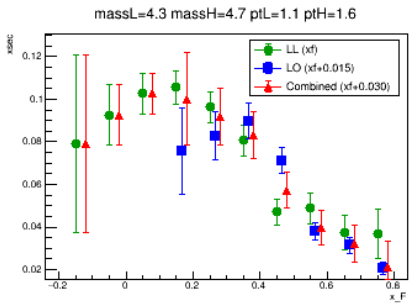

massL=5.4 massH=8.5 ptL=1.1 pthH=1.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

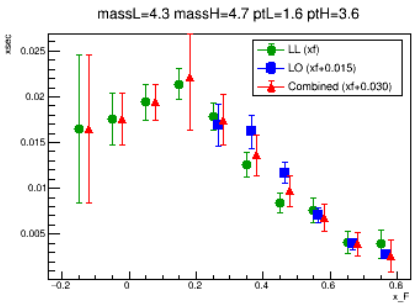

massL=5.4 massH=8.5 ptL=1.6 pthH=3.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

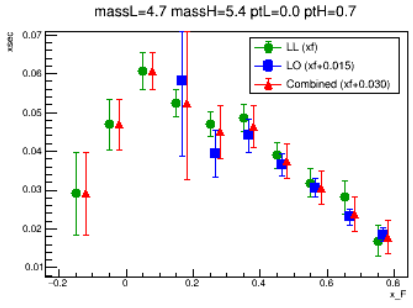
Stat Err : 1.0

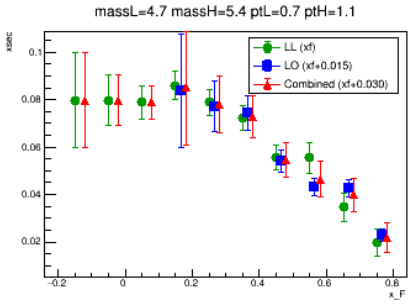

massL=4.3 massH=4.7 ptL=0.0 ptH=0.7
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

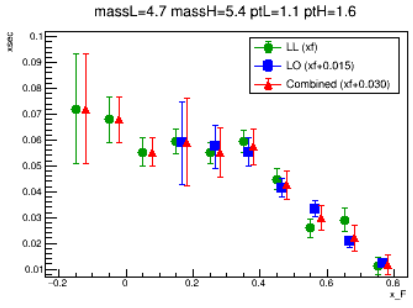

massL=4.3 massH=4.7 ptL=0.7 ptH=1.1
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

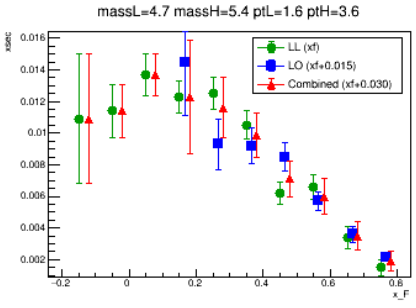

massL=4.3 massH=4.7 ptL=1.1 ptH=1.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

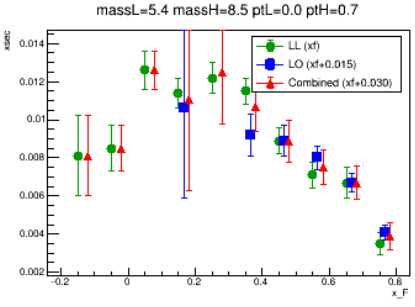

massL=4.3 massH=4.7 ptL=1.6 ptH=3.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

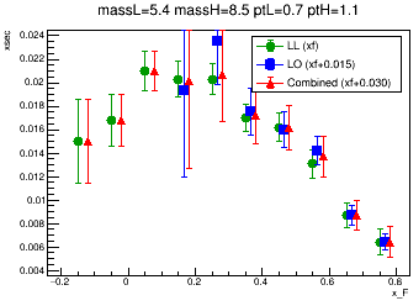

massL=4.7 massH=5.4 ptL=0.0 ptH=0.7
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

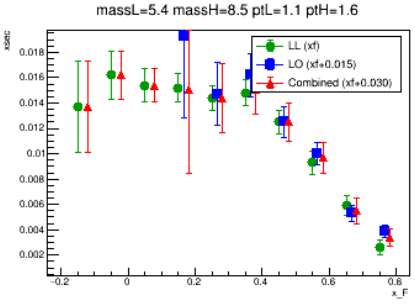

massL=4.7 massH=5.4 ptL=0.7 ptH=1.1
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

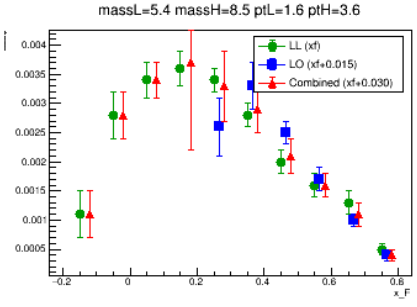

massL=4.7 massH=5.4 ptL=1.1 ptH=1.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F


massL=4.7 massH=5.4 ptL=1.6 ptH=3.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

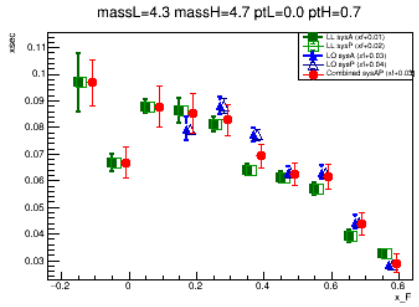

massL=5.4 massH=8.5 ptL=0.0 ptH=0.7
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

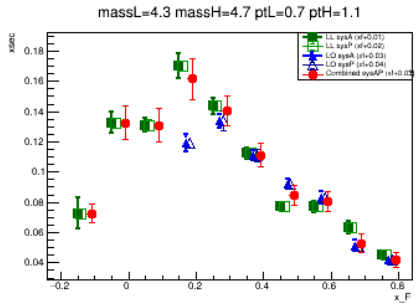

massL=5.4 massH=8.5 ptL=0.7 ptH=1.1
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

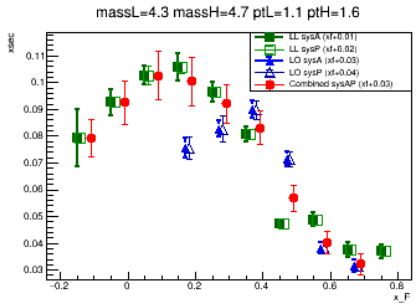

massL=5.4 massH=8.5 ptL=1.1 ptH=1.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

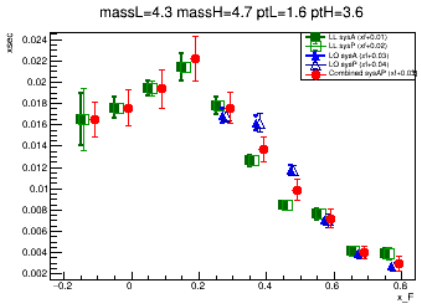

massL=5.4 massH=8.5 ptL=1.6 ptH=3.6
LL (xf)
LO (xf+0.015)
Combined (xf+0.030)
x_F

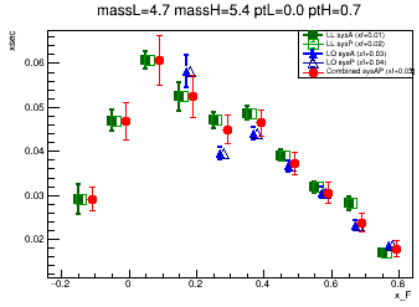
Sys Err : 0.0

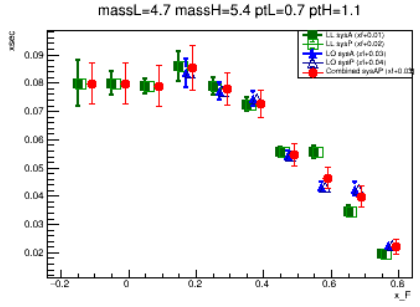

massL=4.3 massH=4.7 ptL=0.0 pthH=0.7

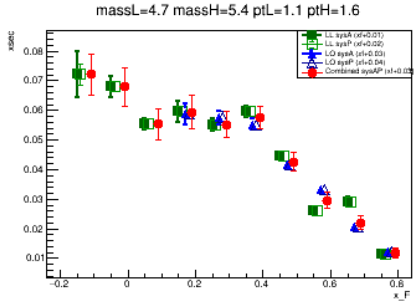

massL=4.3 massH=4.7 ptL=0.7 pthH=1.1

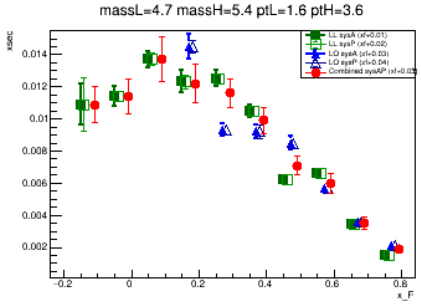

massL=4.3 massH=4.7 ptL=1.1 pthH=1.6

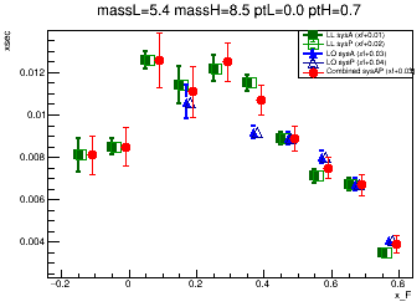

massL=4.3 massH=4.7 ptL=1.6 pthH=3.6

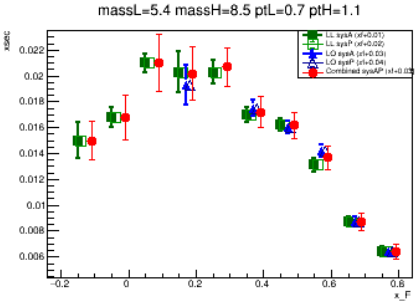

massL=4.7 massH=5.4 ptL=0.0 pthH=0.7

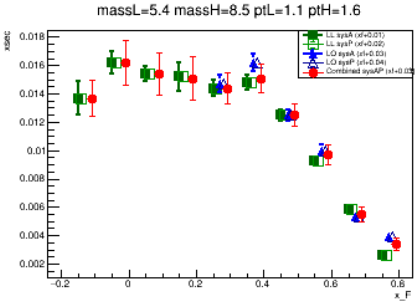

massL=4.7 massH=5.4 ptL=0.7 pthH=1.1

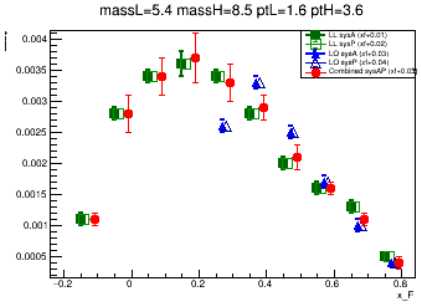

massL=4.7 massH=5.4 ptL=1.1 pthH=1.6


massL=4.7 massH=5.4 ptL=1.6 pthH=3.6

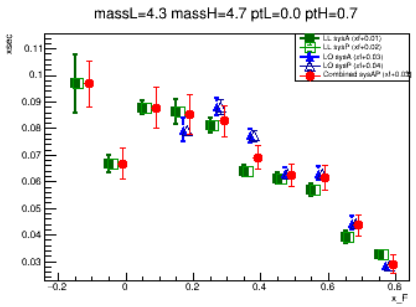

massL=5.4 massH=8.5 ptL=0.0 pthH=0.7

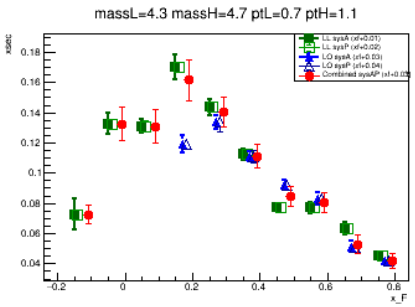

massL=5.4 massH=8.5 ptL=0.7 pthH=1.1

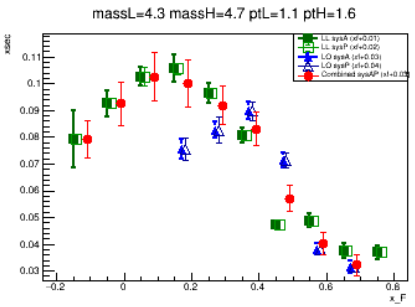

massL=5.4 massH=8.5 ptL=1.1 pthH=1.6

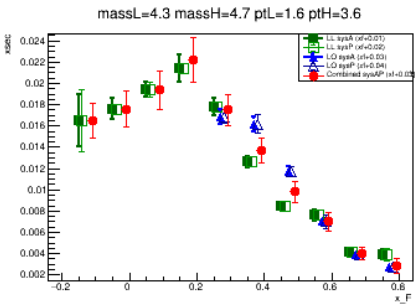

massL=5.4 massH=8.5 ptL=1.6 pthH=3.6

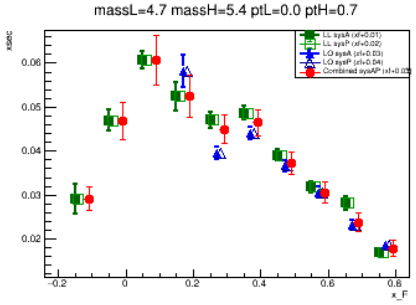
Sys Err : 0.5

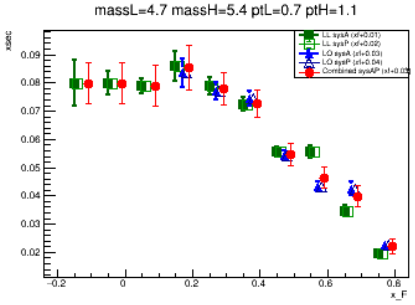

massL=4.3 massH=4.7 ptL=0.0 pthH=0.7

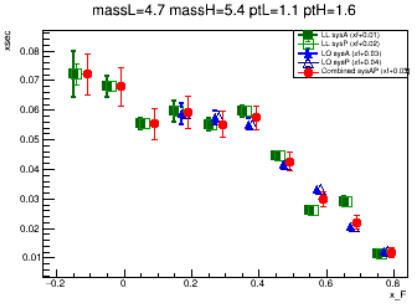

massL=4.3 massH=4.7 ptL=0.7 pthH=1.1

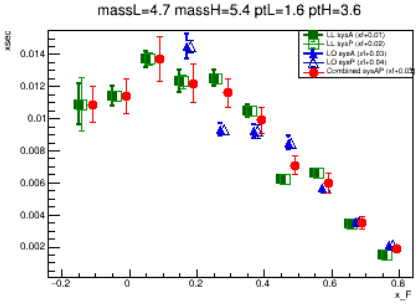

massL=4.3 massH=4.7 ptL=1.1 pthH=1.6

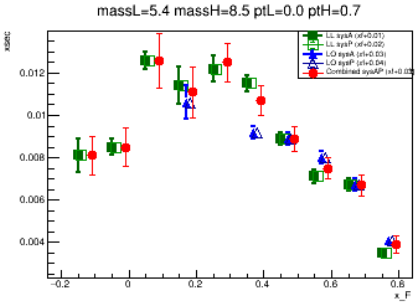

massL=4.3 massH=4.7 ptL=1.6 pthH=3.6

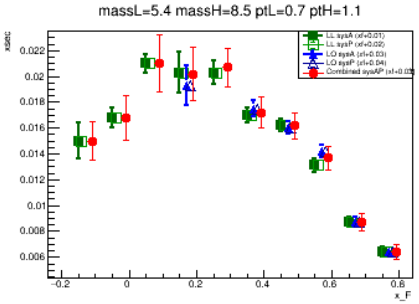

massL=4.7 massH=5.4 ptL=0.0 pthH=0.7

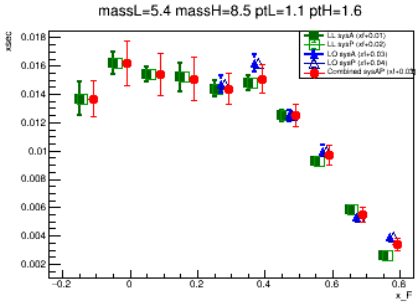

massL=4.7 massH=5.4 ptL=0.7 pthH=1.1

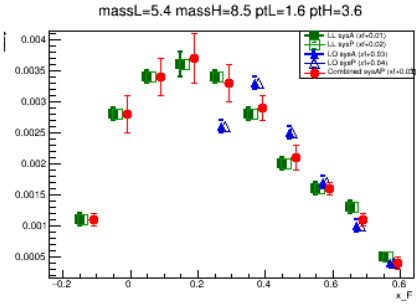

massL=4.7 massH=5.4 ptL=1.1 pthH=1.6


massL=4.7 massH=5.4 ptL=1.6 pthH=3.6


massL=5.4 massH=8.5 ptL=0.0 pthH=0.7

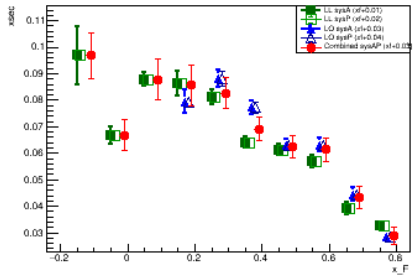

massL=5.4 massH=8.5 ptL=0.7 pthH=1.1


massL=5.4 massH=8.5 ptL=1.1 pthH=1.6

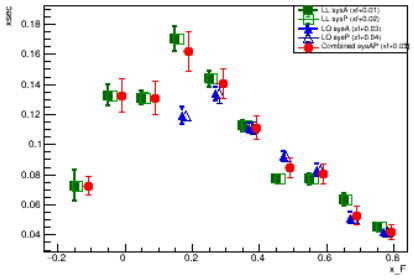

massL=5.4 massH=8.5 ptL=1.6 pthH=3.6

Sys Err : 1.0

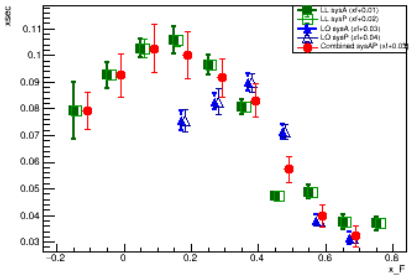
massL=4.3 massH=4.7 ptL=0.0 pthH=0.7



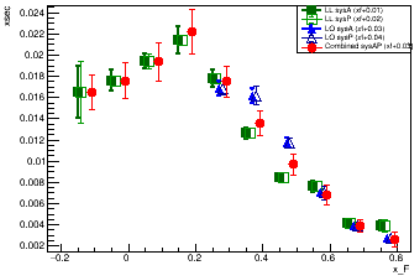
massL=4.3 massH=4.7 ptL=0.7 pthH=1.1



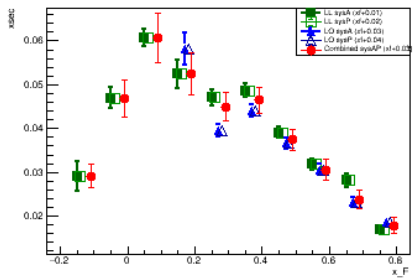
massL=4.3 massH=4.7 ptL=1.1 pthH=1.6



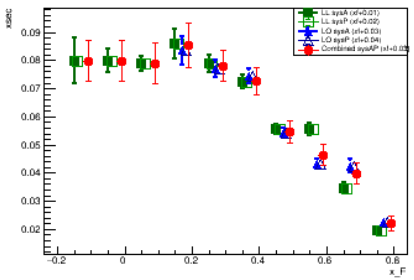
massL=4.3 massH=4.7 ptL=1.6 pthH=3.6



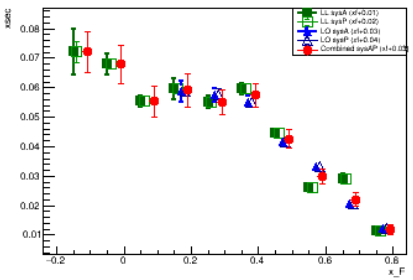
massL=4.7 massH=5.4 ptL=0.0 pthH=0.7



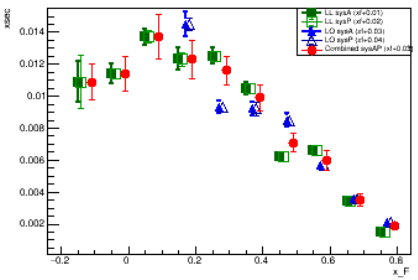
massL=4.7 massH=5.4 ptL=0.7 pthH=1.1



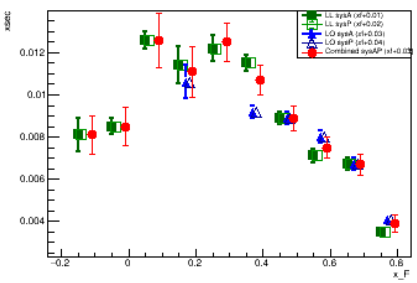
massL=4.7 massH=5.4 ptL=1.1 pthH=1.6



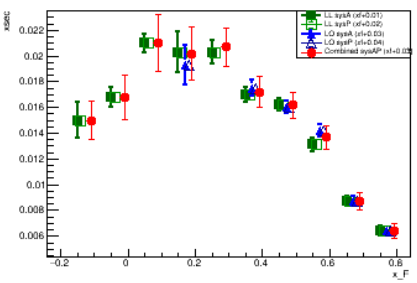
massL=4.7 massH=5.4 ptL=1.6 pthH=3.6



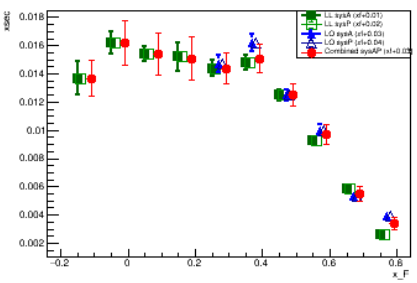
massL=5.4 massH=8.5 ptL=0.0 pthH=0.7



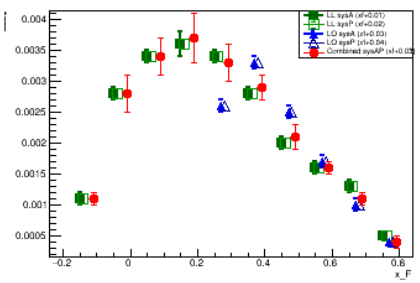
massL=5.4 massH=8.5 ptL=0.7 pthH=1.1



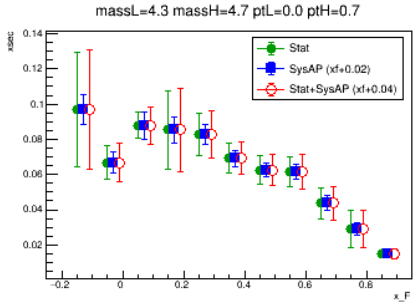
massL=5.4 massH=8.5 ptL=1.1 pthH=1.6

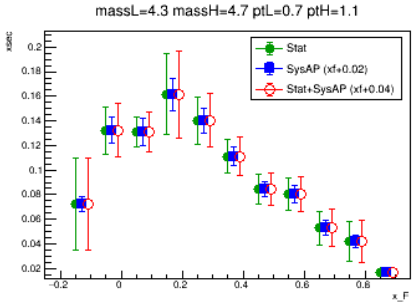


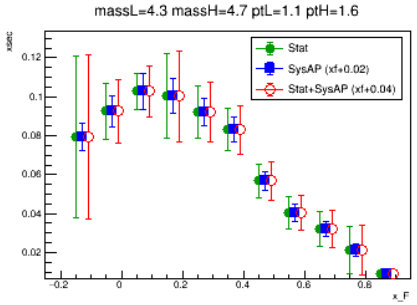
massL=5.4 massH=8.5 ptL=1.6 pthH=3.6

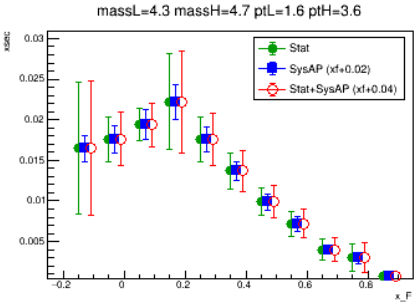


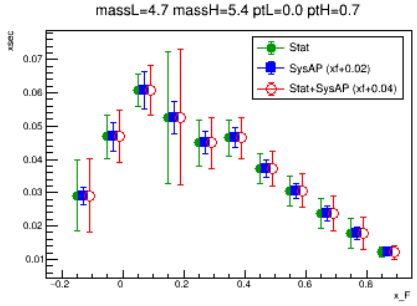
Final results : 0.0

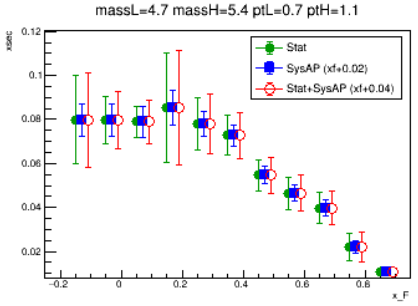


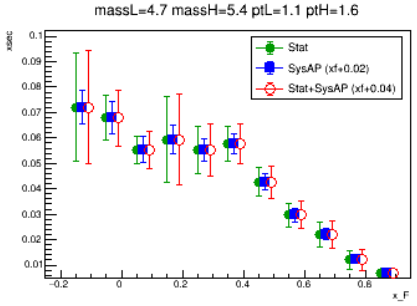


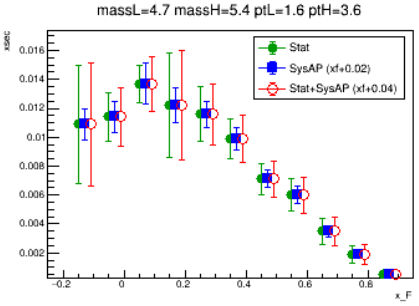


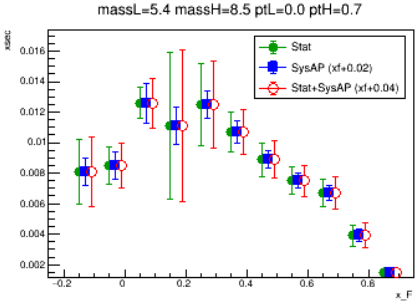


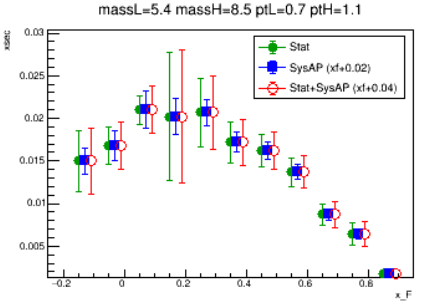


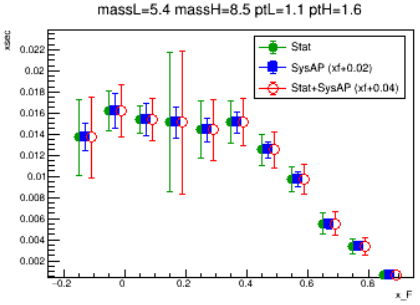


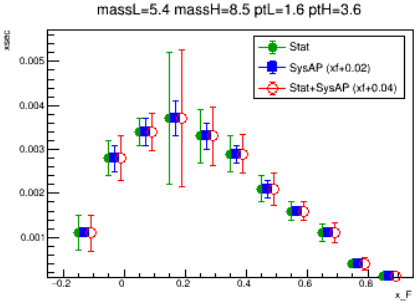




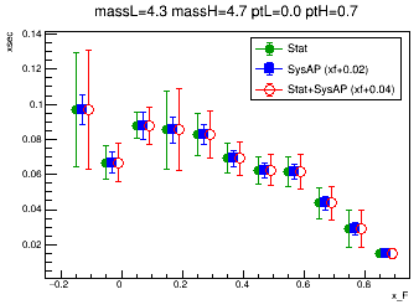


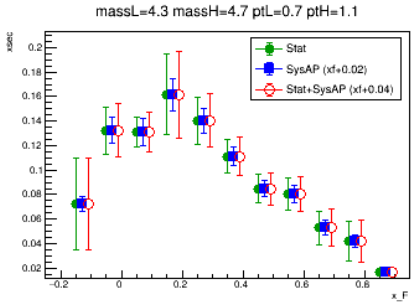


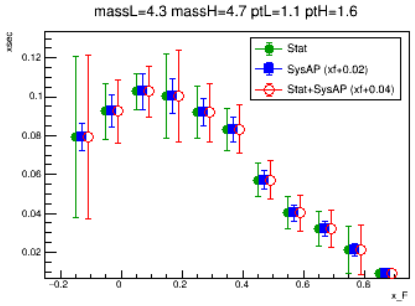


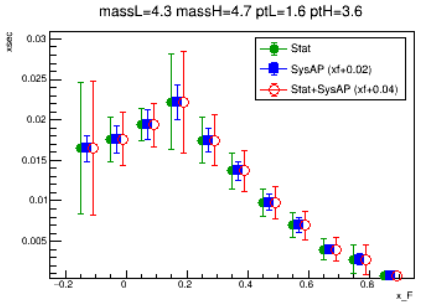


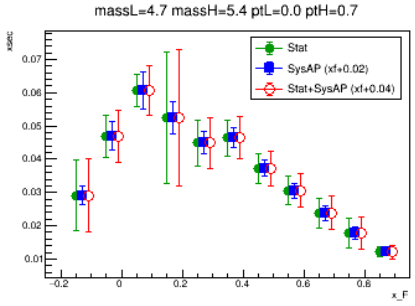
Final results : 0.5

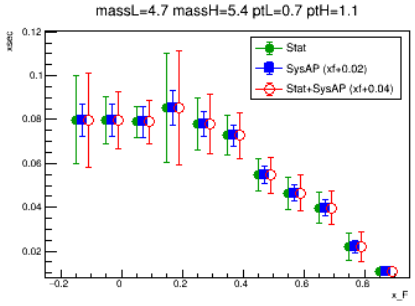


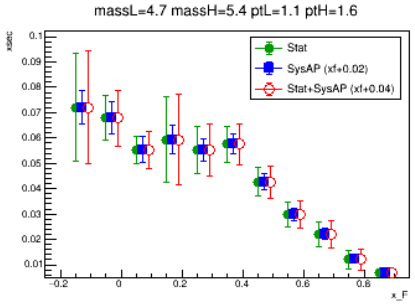


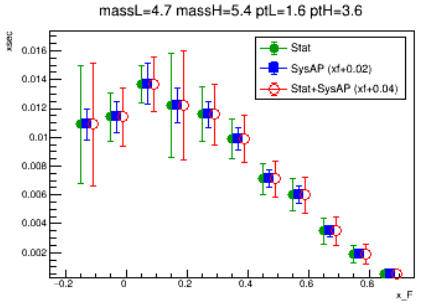


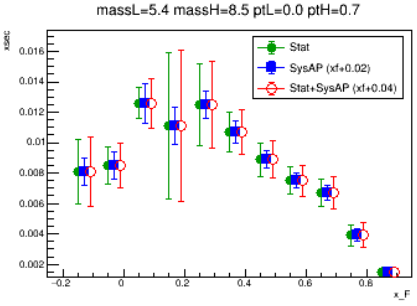


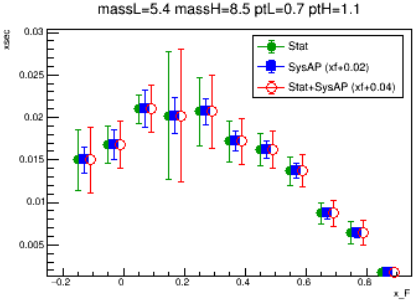


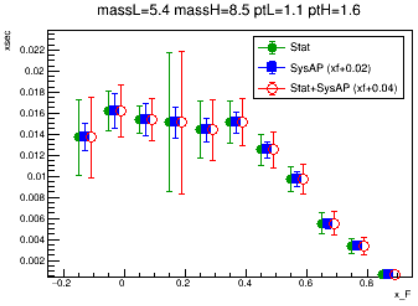


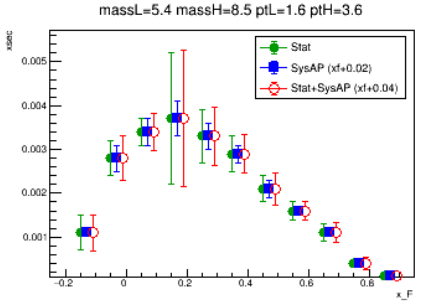




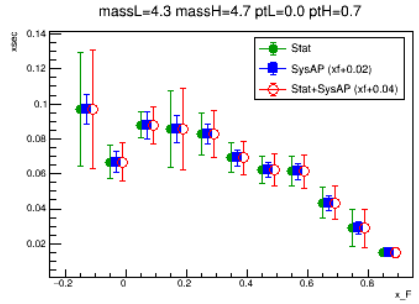


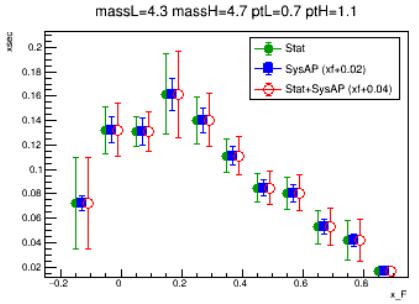


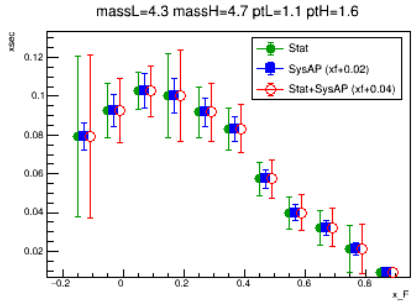


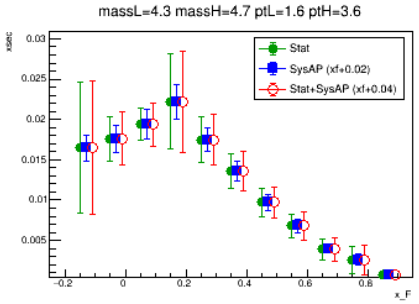


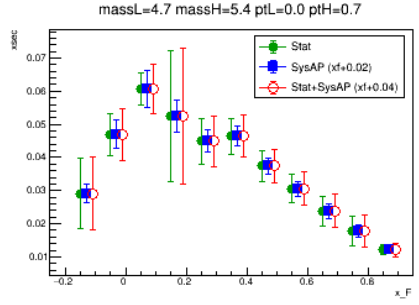
Final results : 1.0

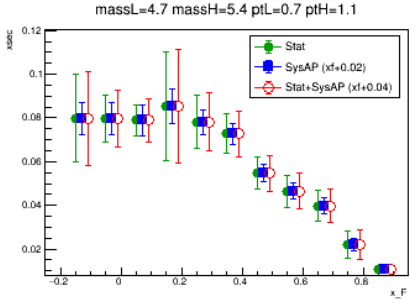


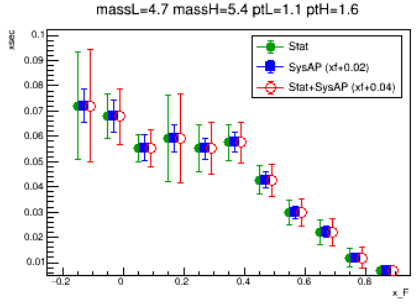


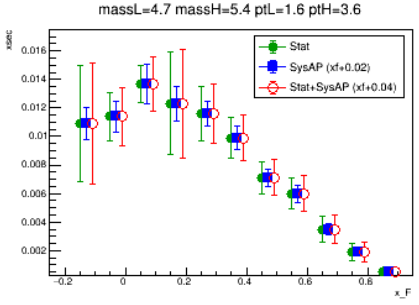


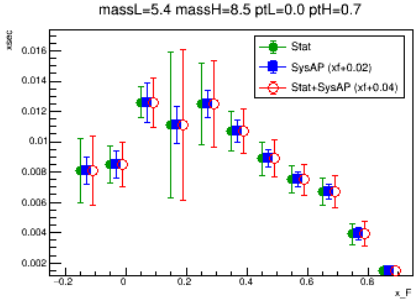


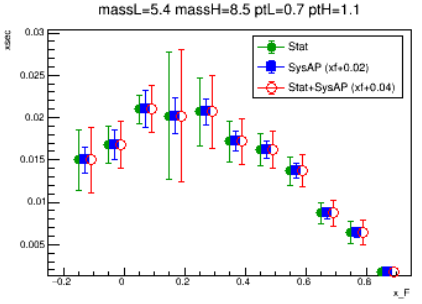


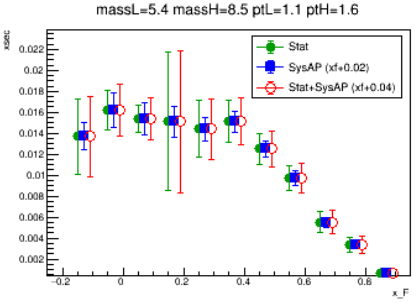


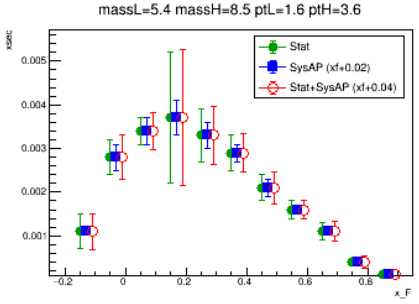








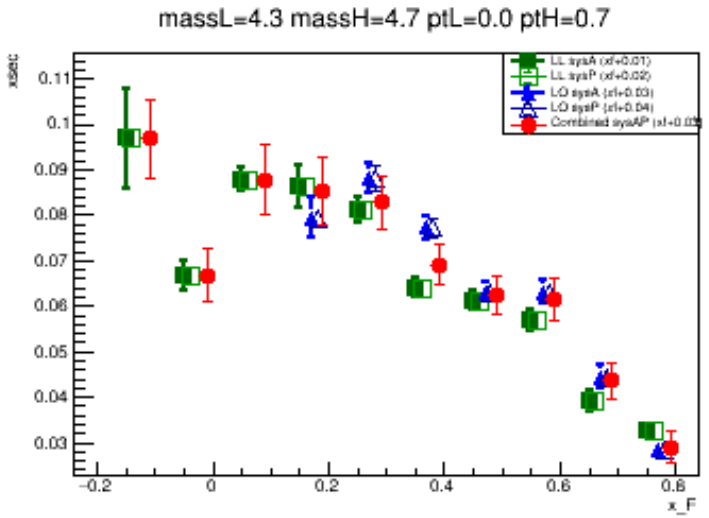




Difference between 0.5 and 1.0

```
3,5c3,5
< 4.3 4.7 0 0.7   0.05   0.0879   0.0073   0.0027   0.0015   0.0029   0.0000   0.0000   0.0000   0.0000   0.0000   0.0879   0.0077   0.0083   0.0073
< 4.3 4.7 0 0.7   0.15   0.0864   0.0061   0.0047   0.0015   0.0029   0.0794   0.0214   0.0044   0.0005   0.0026   0.0855   0.0075   0.0085   0.0222
< 4.3 4.7 0 0.7   0.25   0.0812   0.0056   0.0028   0.0015   0.0027   0.0881   0.0108   0.0031   0.0027   0.0029   0.0828   0.0058   0.0070   0.0121
---
> 4.3 4.7 0 0.7   0.05   0.0879   0.0073   0.0027   0.0015   0.0029   0.0000   0.0000   0.0000   0.0000   0.0000   0.0879   0.0077   0.0082   0.0073
> 4.3 4.7 0 0.7   0.15   0.0864   0.0061   0.0047   0.0015   0.0029   0.0794   0.0214   0.0044   0.0005   0.0026   0.0856   0.0075   0.0085   0.0222
> 4.3 4.7 0 0.7   0.25   0.0812   0.0056   0.0028   0.0015   0.0027   0.0881   0.0108   0.0031   0.0027   0.0029   0.0827   0.0058   0.0070   0.0121
```

Difference is too small to be observed.
Correlation of purity systematics doesn't play a big role.



Systematics of acceptance is much larger than purity.

Test Correlated of Acc. Sys.

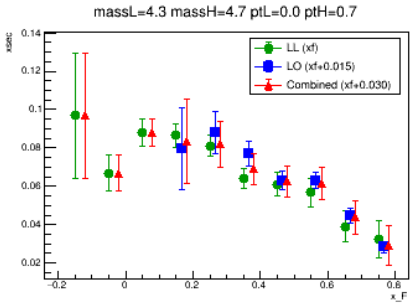
```
#double check the numbers!
purCorr_crossDia_LL     = 1.0 # = 1.0, okay, same bin same trigger
purCorr_crossOff_LL     = 0.0 # = 0.0, but one should test (0, 1), different bins same trigger <=========
purCorr_crossDia_LO     = 1.0 # = 1.0, okay, same bin same trigger
purCorr_crossOff_LO     = 0.0 # = 0.0, but one should test (0, 1), different bins same trigger <=========
purCorr_crossDia_LL_LO  = 0.0 # = 0, okay?, same bins different triggers                       <=========
purCorr_crossOff_LL_LO  = 0.0 # = 0, okay?, different bins different triggers                  <=========

accCorr_scale_LL        = 5 # all random for both diagonal and off-diagonal? not realistic   <=========
accCorr_scale_LO        = 0.8 # all random for both diagonal and off-diagonal? not realistic   <=========
accCorr_crossDia_LL_LO  = 0.5 # = 0.5, okay, cross means different bins and different triggers
accCorr_crossOff_LL_LO  = 0.5 # = 0.5, but one should test (0, 1)
```

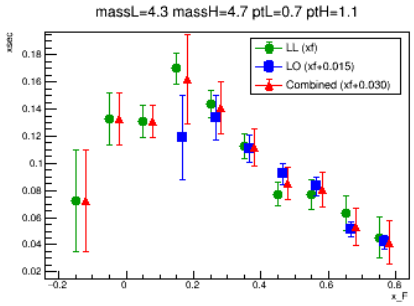
2

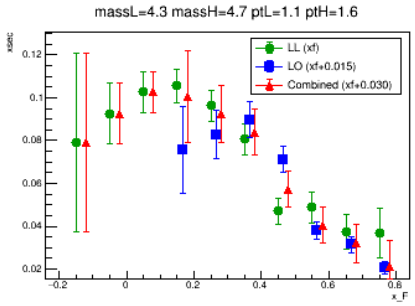
[0.5, 5, 50]

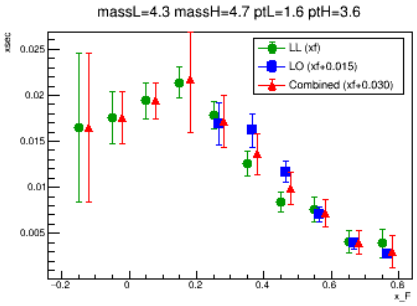
Small values : nearby bins are strongly correlated

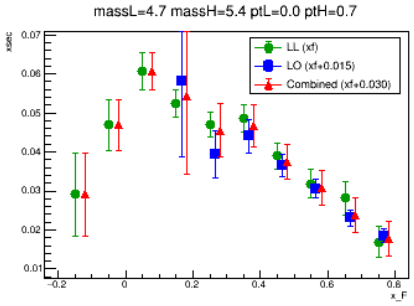
(Scale_LL, Scale_LO) = (0.5, 0.5) ➔ Stat. Err.

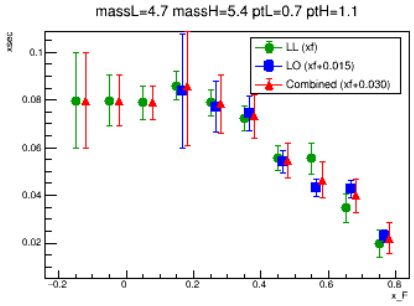


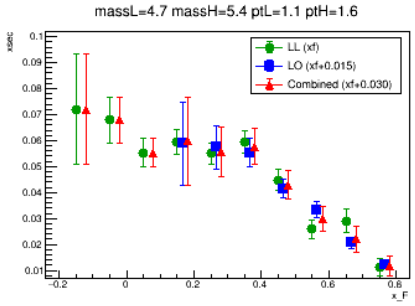


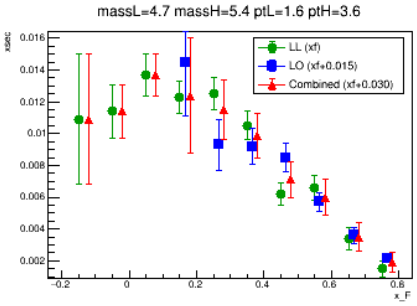


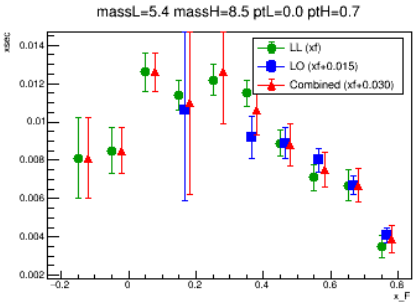


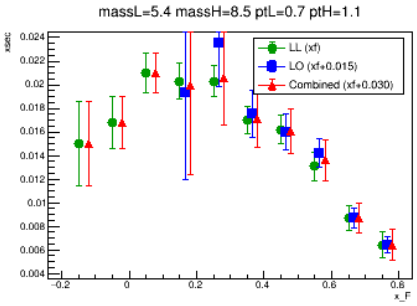


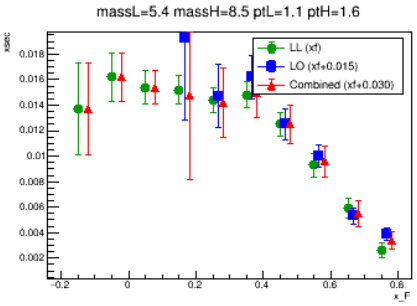


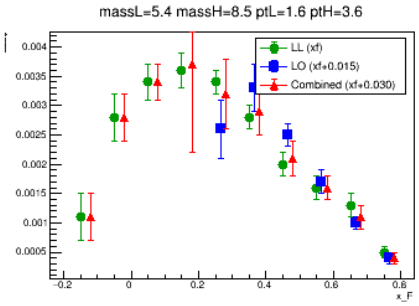




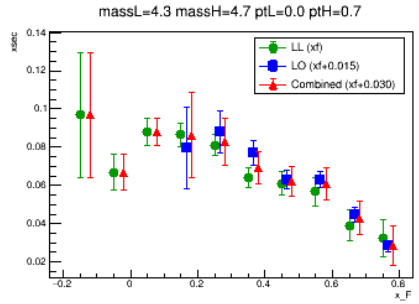


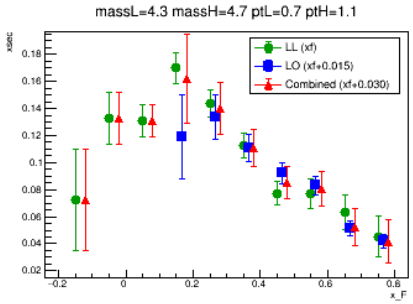


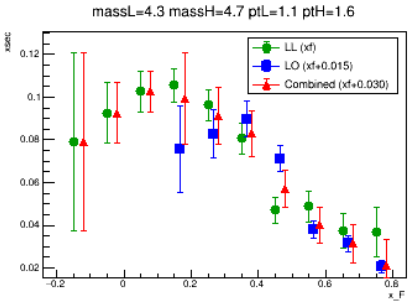


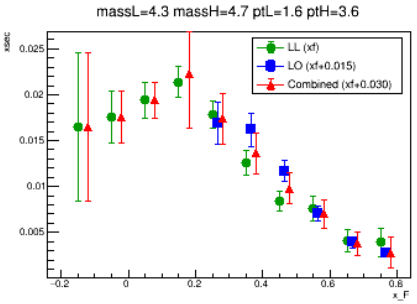


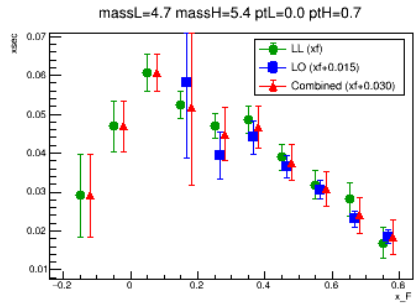
(Scale_LL, Scale_LO) = (5.0, 5.0)

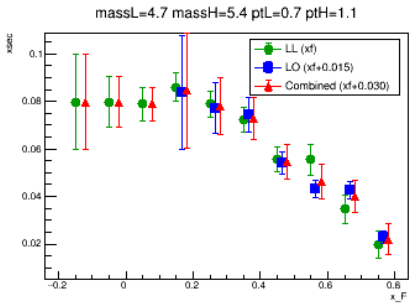


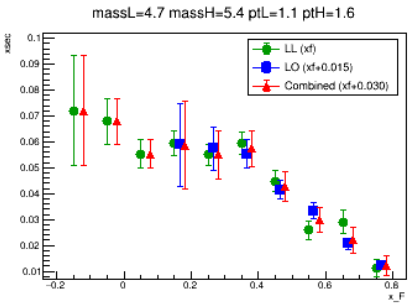


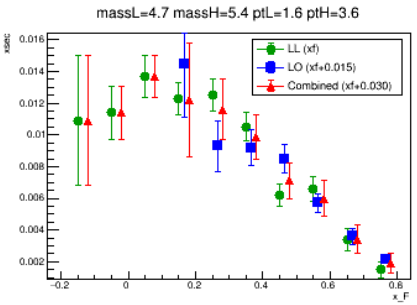


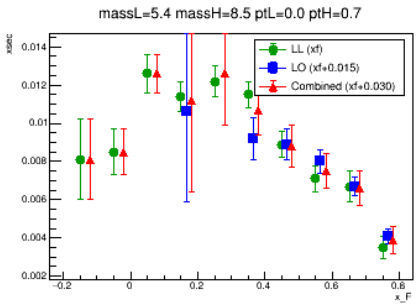


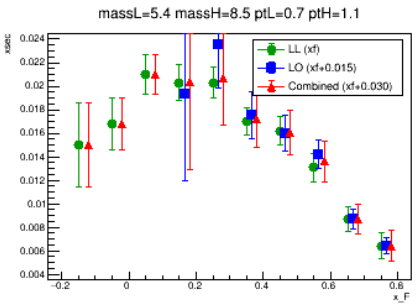


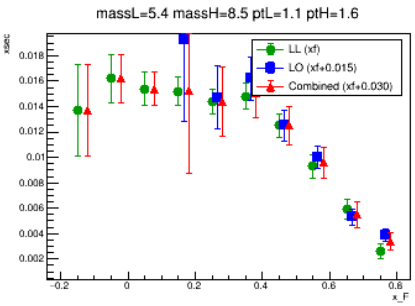


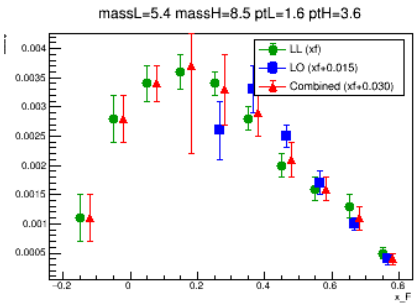




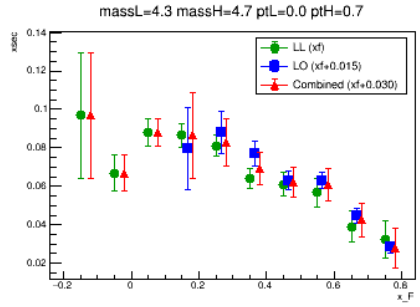


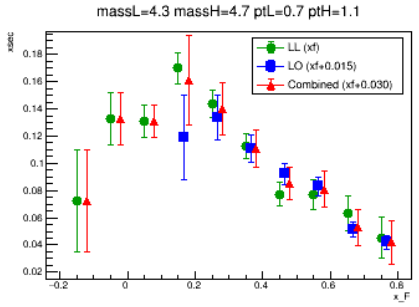


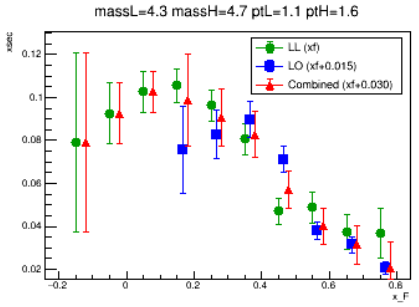


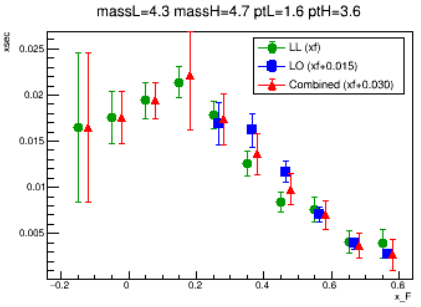


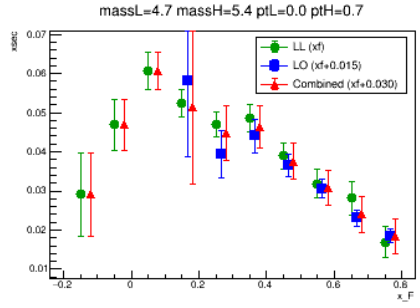
(Scale_LL, Scale_LO) = (50, 50)

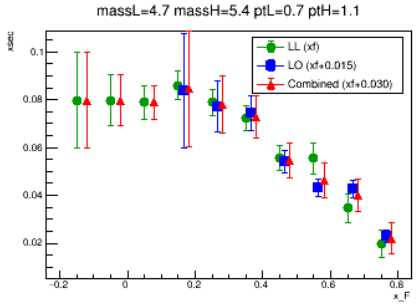


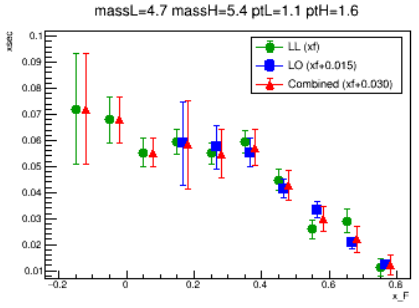


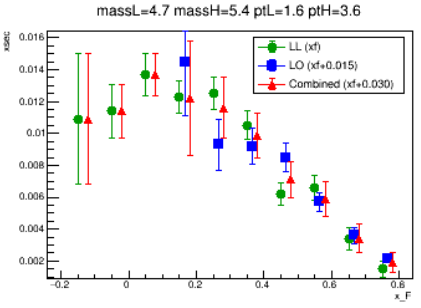


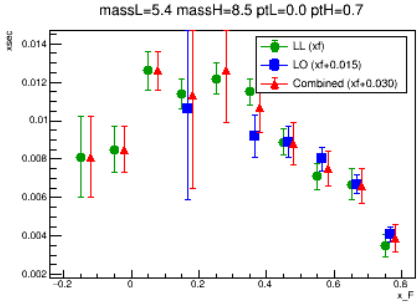


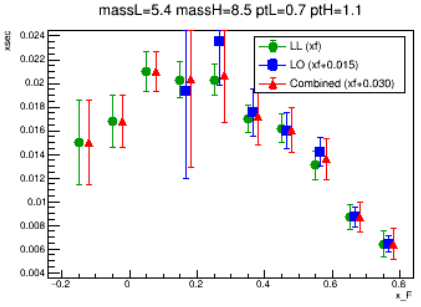


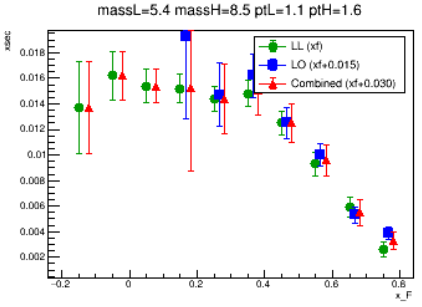


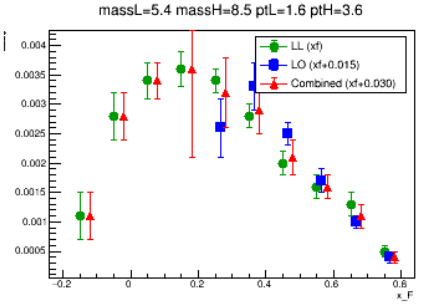




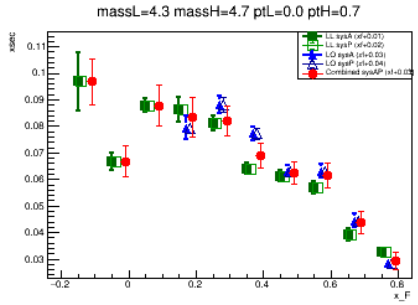


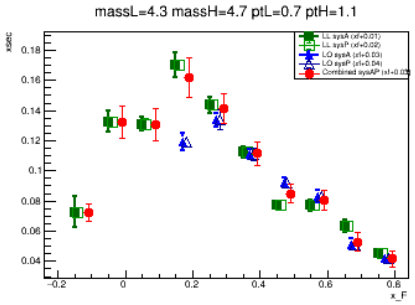


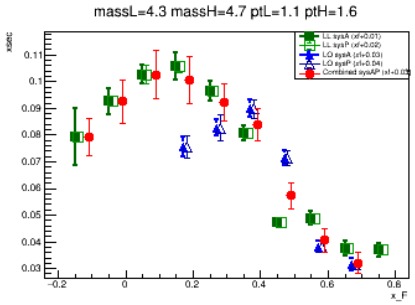


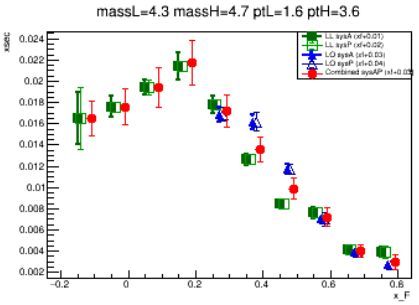


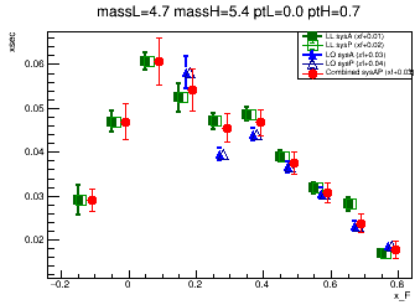
(Scale_LL, Scale_LO) = (0.5, 0.5) ➔ Sys. Err.

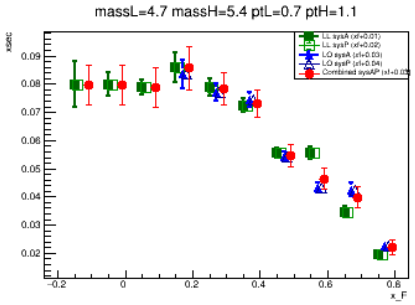


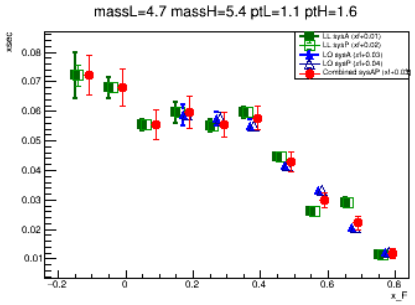


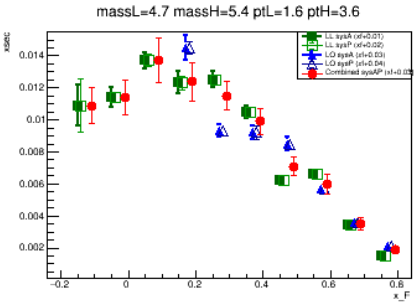


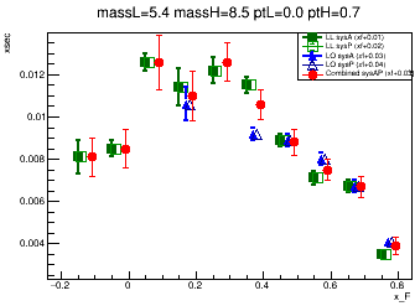


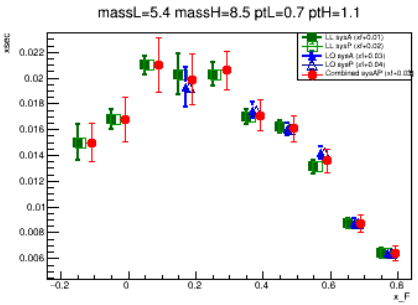


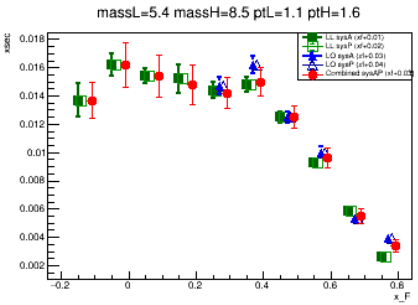


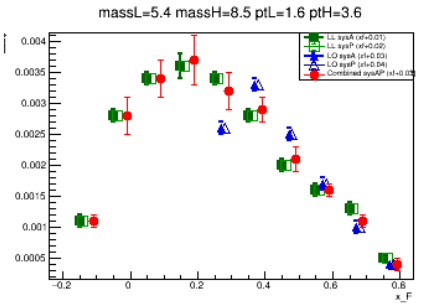




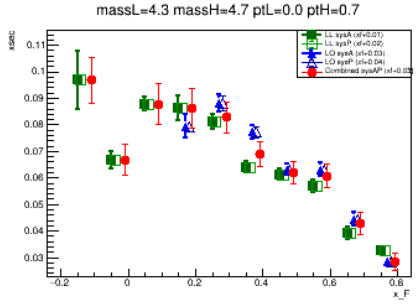


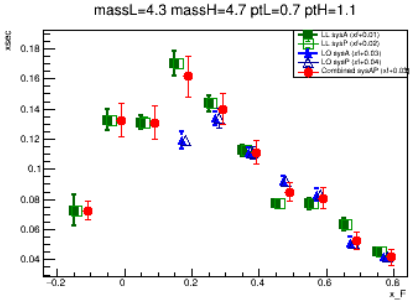


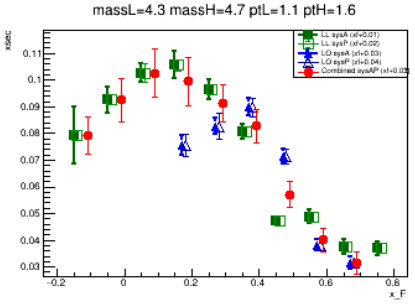


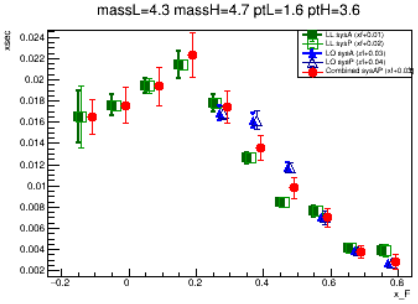


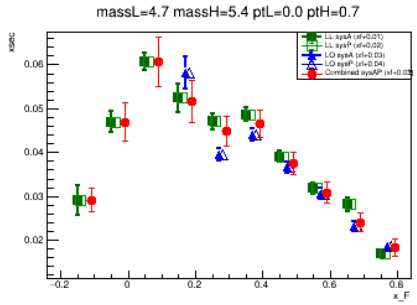
(Scale_LL, Scale_LO) = (5.0, 5.0)

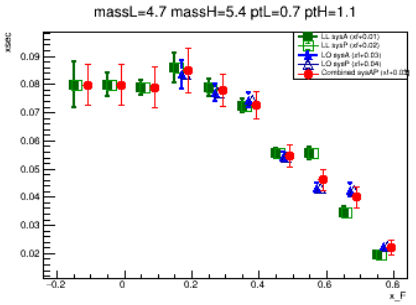


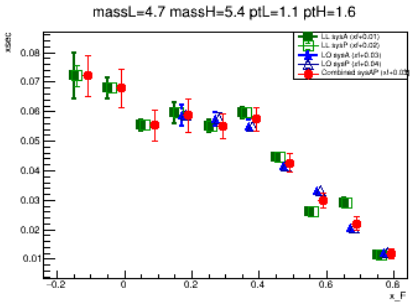


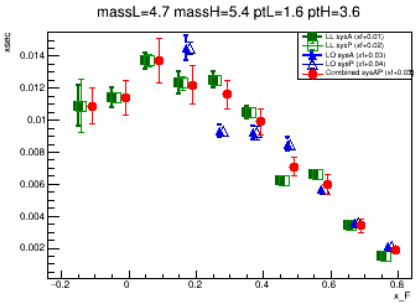


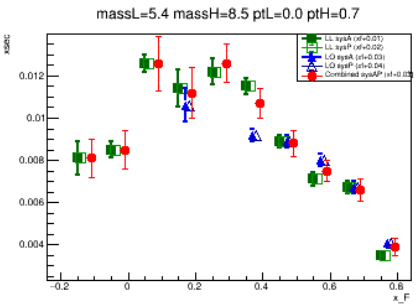


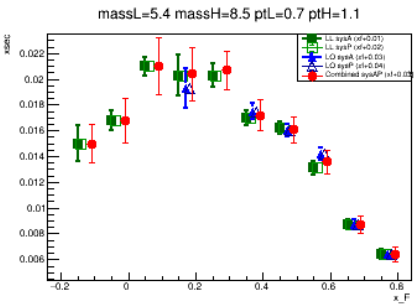


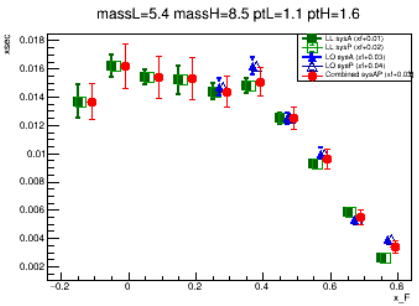


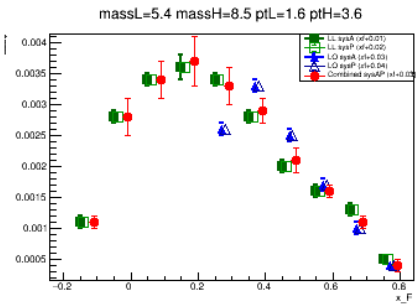




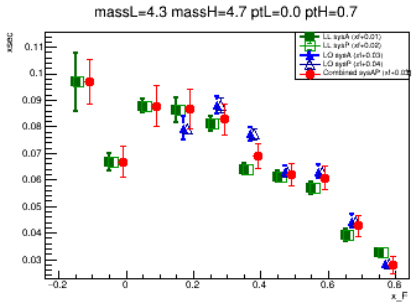


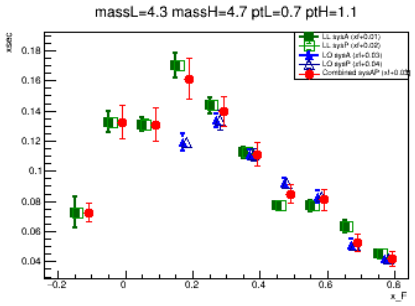


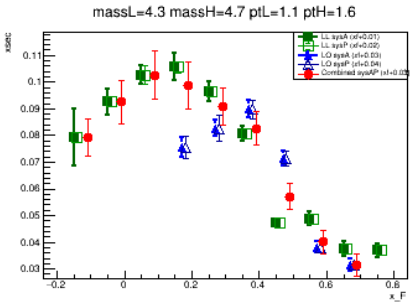


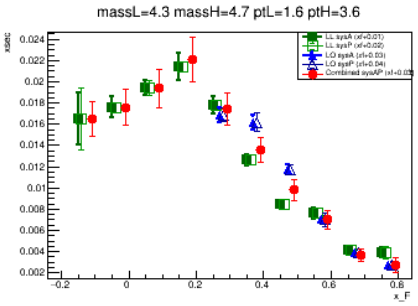


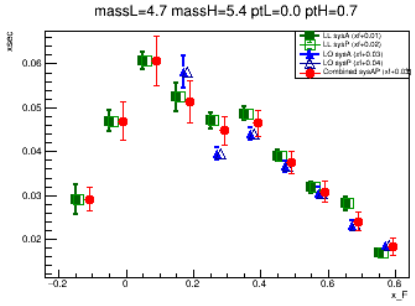
(Scale_LL, Scale_LO) = (50, 50)

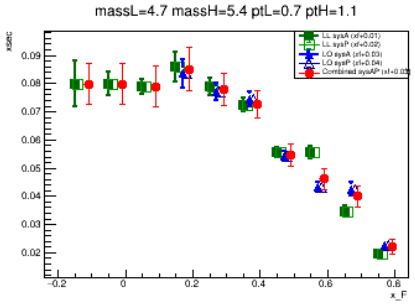


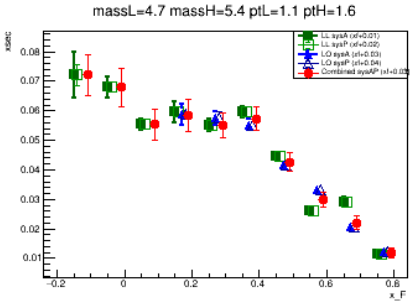


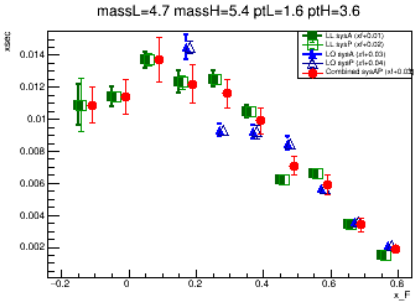


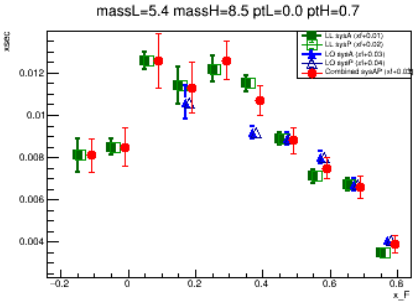


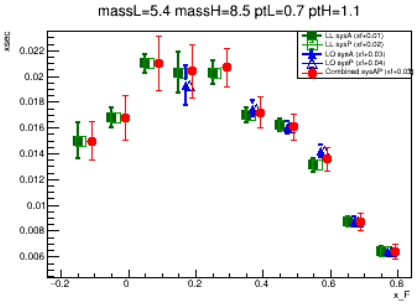


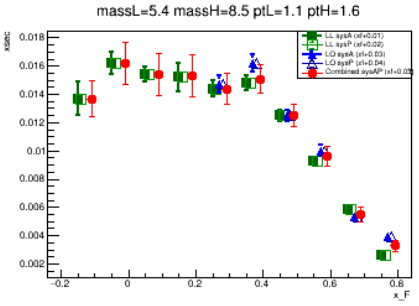


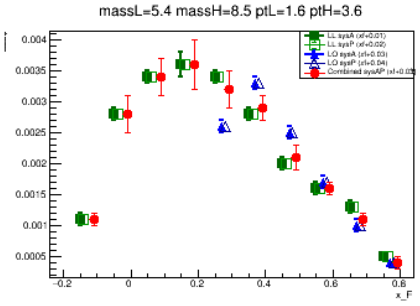




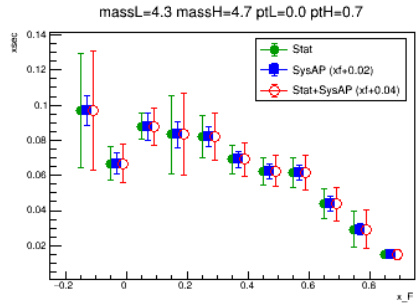


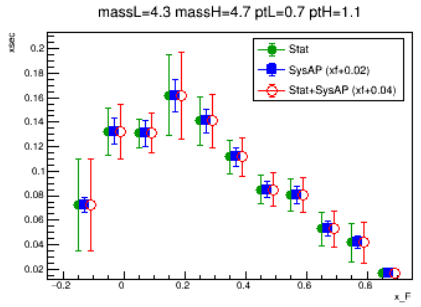


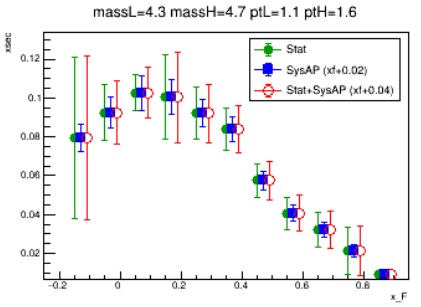


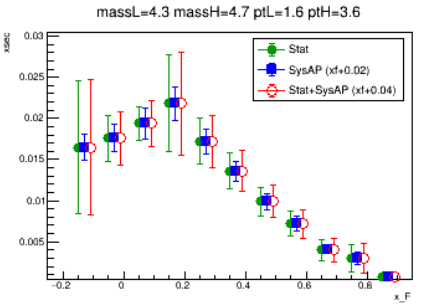


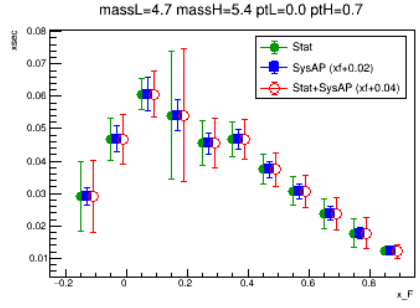
(Scale_LL, Scale_LO) = (0.5, 0.5) ➔ Final

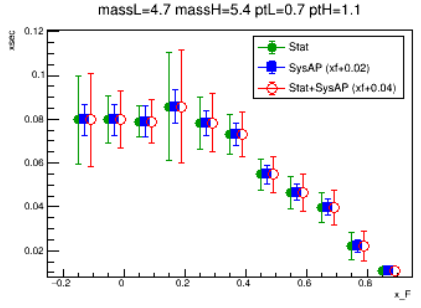


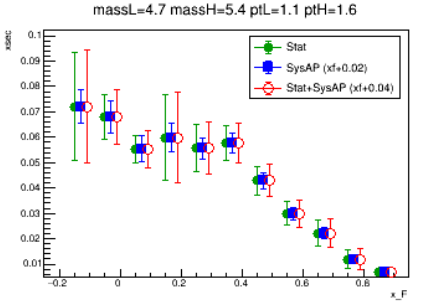


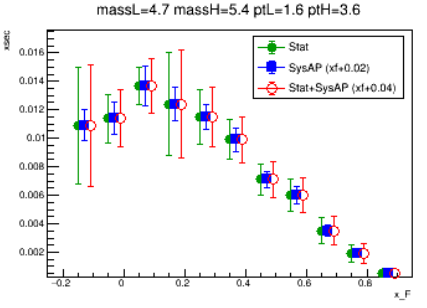


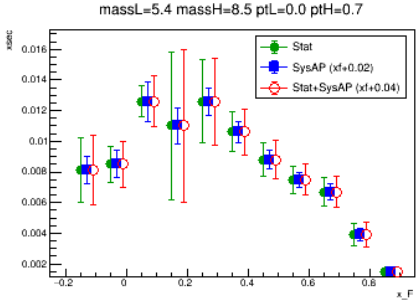


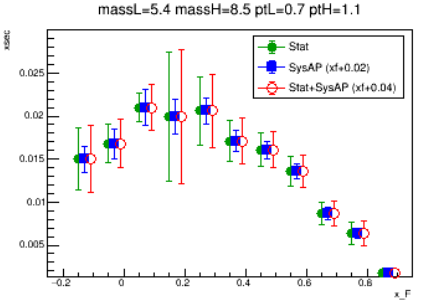


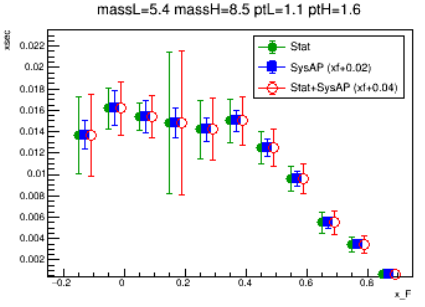


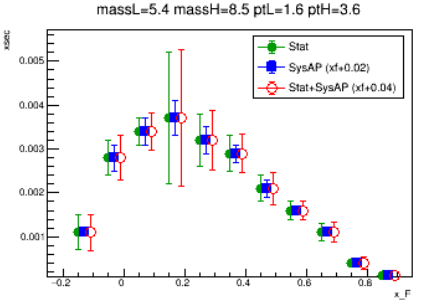




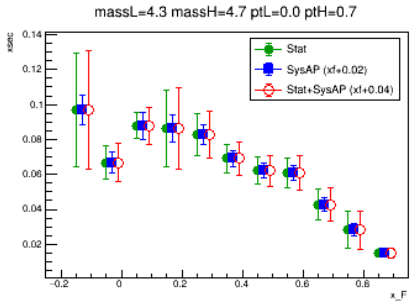


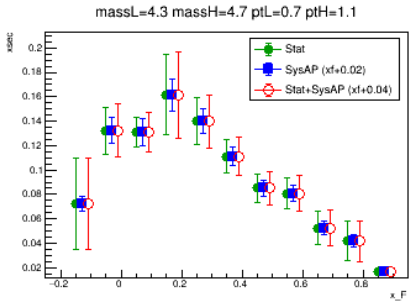


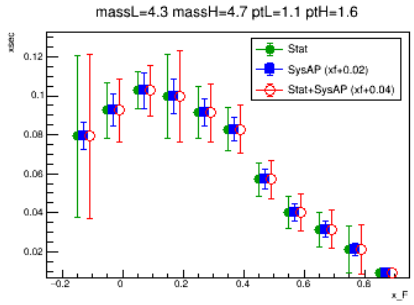


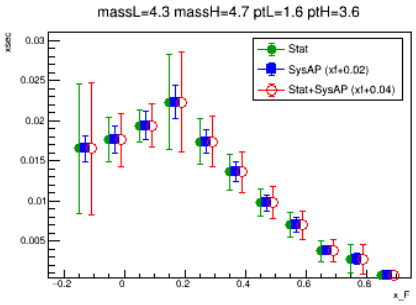


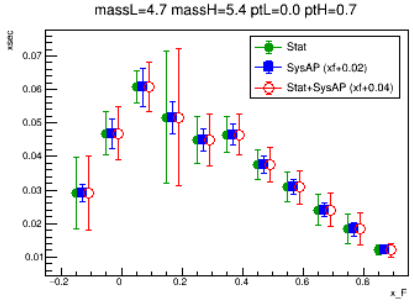
(Scale_LL, Scale_LO) = (5.0, 5.0)

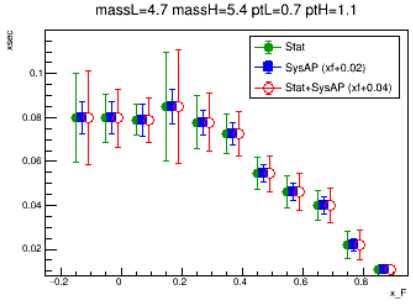


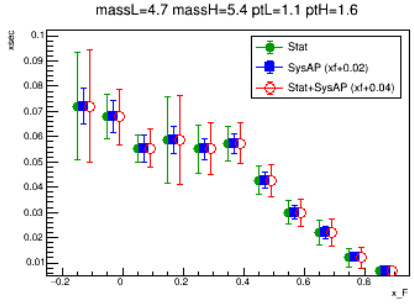


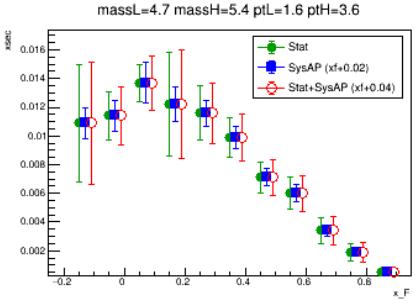


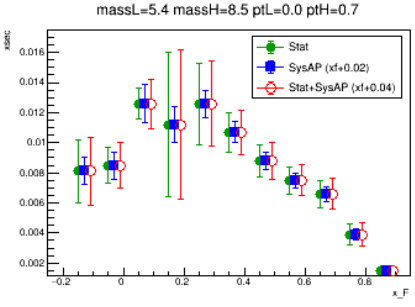


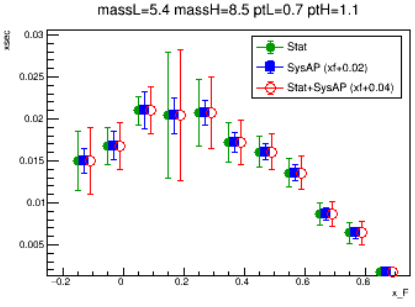


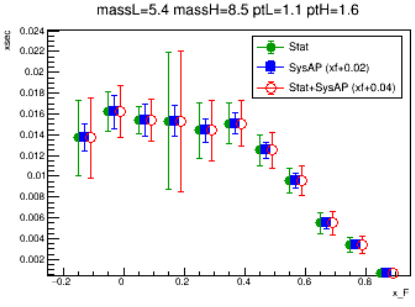


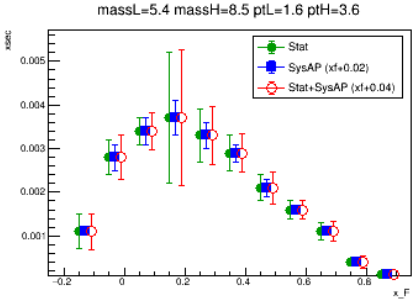




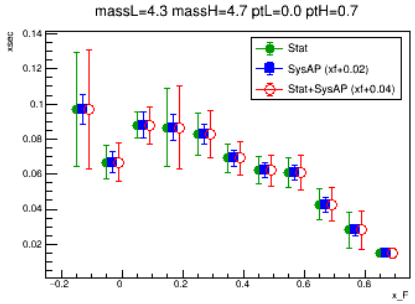


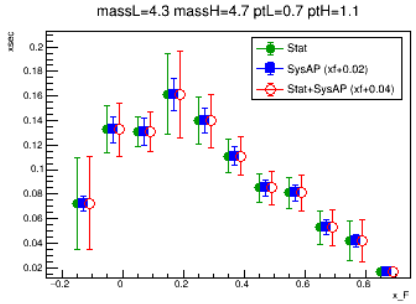


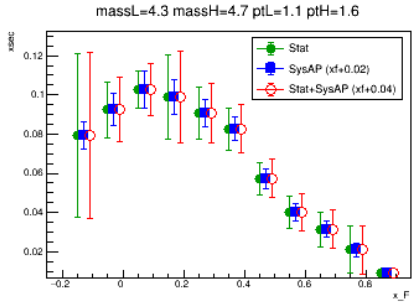


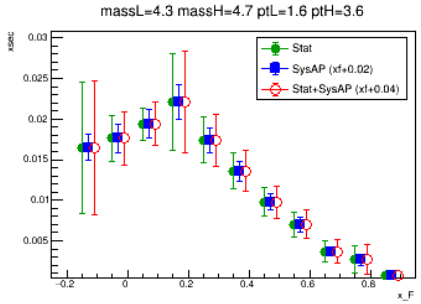


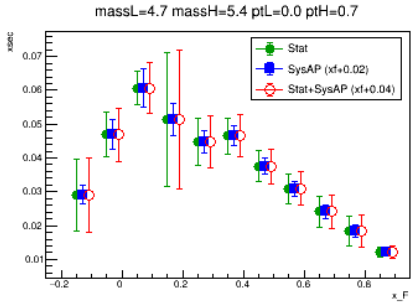
(Scale_LL, Scale_LO) = (50, 50)

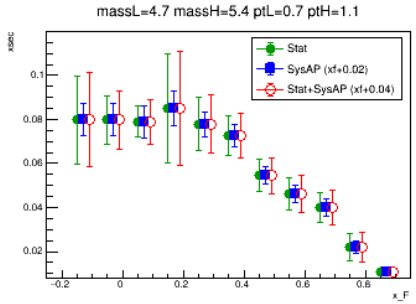


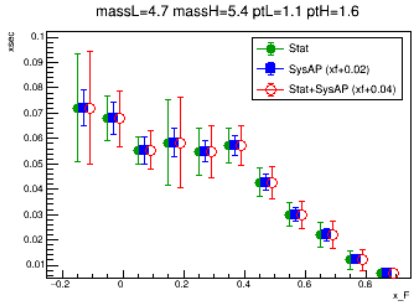


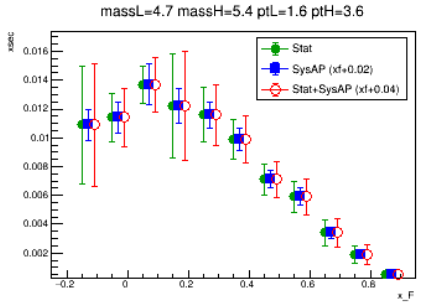


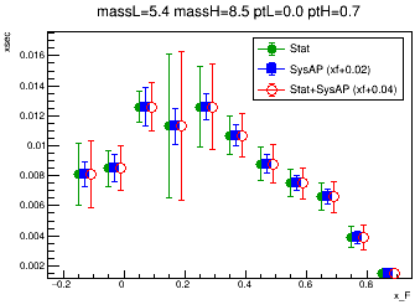


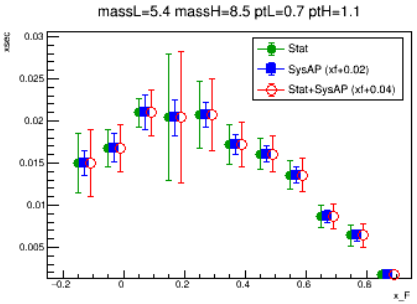


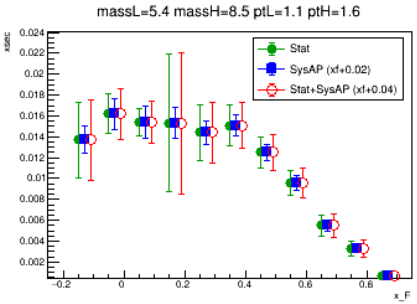


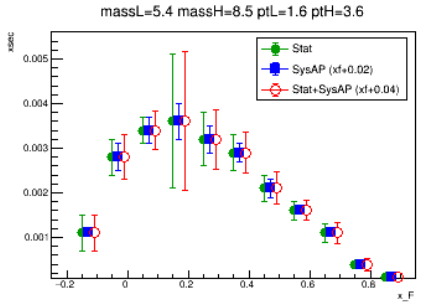












Back up