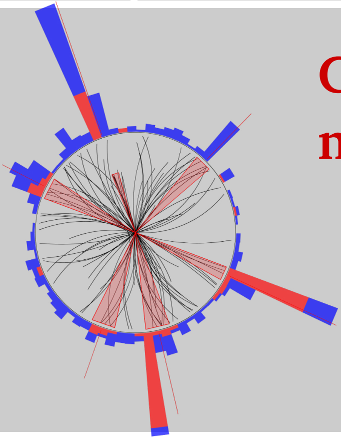
UHH PD group meeting

Wednesday 23rd January, 2019



Combining top quark mass measurements

SPONSORED BY THE

Federal Ministry of Education and Research

Christoph Garbers

Johannes Lange
johannes.lange@cern.ch

Peter Schleper

Hartmut Stadie

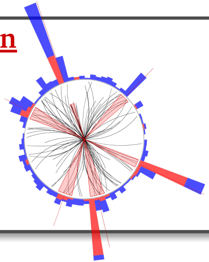
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Reminder: m_t measurement

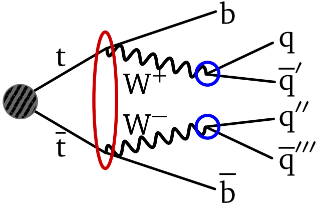
Event selection

all-jets final state

- 2016 dataset
- six jets, $p_T > 40\,\text{GeV}$
- two b tags

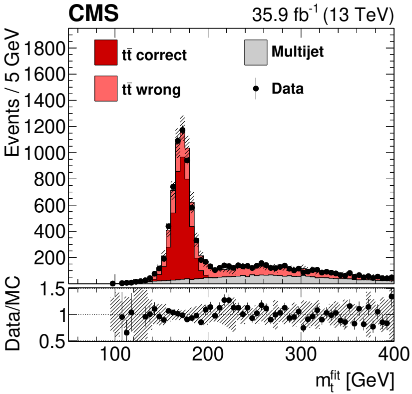


- kinematic fit with constraints:
  $m_{W^+} = m_{W^-} = 80.4\,\text{GeV}, m_t = m_{\bar{t}}$
- multijet background estimated from data
- mass extraction: ideogram method

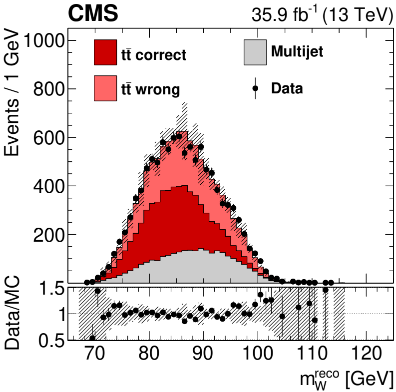


TOP-17-008, submitted to EPJC

Final selection



sensitive to m_t



sensitive to additional jet scale factor (JSF)

Mass extraction: ideogram method

Estimate m_t and additional jet scale factor (JSF)

$$P\left(m_\mathrm{t}, \mathrm{JSF} \middle| \mathrm{sample}\right) \propto P(\mathrm{JSF}) \cdot \mathcal{L}\left(\mathrm{sample} \middle| m_\mathrm{t}, \mathrm{JSF}\right)$$

$$\mathcal{L}\left(\mathrm{sample} \middle| m_\mathrm{t}, \mathrm{JSF}\right) = \prod_{\mathrm{events}} \mathcal{L}\left(\mathrm{event} \middle| m_\mathrm{t}, \mathrm{JSF}\right) = \prod_{\mathrm{events}} P\left(m_\mathrm{t}^\mathrm{fit}, m_\mathrm{W}^\mathrm{reco} \middle| m_\mathrm{t}, \mathrm{JSF}\right)$$

Mass extraction: ideogram method

Estimate m_t and additional jet scale factor (JSF)

$$P\left(m_t, \mathrm{JSF}|\mathrm{sample}\right) \propto P(\mathrm{JSF}) \cdot \mathcal{L}\left(\mathrm{sample}|m_t, \mathrm{JSF}\right)$$

$$\mathcal{L}\left(\mathrm{sample}|m_t, \mathrm{JSF}\right) = \prod_{\mathrm{events}} \mathcal{L}\left(\mathrm{event}|m_t, \mathrm{JSF}\right) = \prod_{\mathrm{events}} P\left(m_t^{\mathrm{fit}}, m_W^{\mathrm{reco}} \middle| m_t, \mathrm{JSF}\right)$$

Three versions of ideogram fit:

- m_t and JSF free (2D)
- fixed JSF = 1 (1D)
- Gaussian JSF constraint (hybrid)

Result (hybrid)

$$m_t = 172.34 \pm 0.20\,(\mathrm{stat{+}JSF}) \pm 0.70\,(\mathrm{syst})\ \mathrm{GeV}$$

Want to combine all-jets and ℓ+jets measurements (at 13 TeV)

Best Linear Unbiased Estimate

given: n measurements $y_1 \ldots y_n$ with
covariance matrix E (i.e., known correlations)

linear combination:

$$\hat{y} = \sum \alpha_i y_i$$

unbiased

$$\sum \alpha_i = 1$$

best: find α that minimize

$$\sigma^2 = \alpha^{\mathrm{T}} E \alpha = \sum_i \sum_j E_{ij} \alpha_i \alpha_j$$

The BLUE method

$$\hat{y} = \sum \alpha_i y_i$$

- minimal σ^2 solution (algebraic!):

$$\alpha = E^{-1}U/(U^{\mathrm{T}}E^{-1}U)$$

  U: n-component vector with $U_i = 1$
- if uncorrelated: E diagonal, simple $1/\sigma_i^2$-weighted mean

- equivalent to χ^2 minimization (considering correlations):

$$\chi^2 = \sum_{ij} (\hat{y} - y_i)(\hat{y} - y_j)(E^{-1})_{ij}$$

Python package `blue_combine`

- very simple python package without huge dependencies (no ROOT, Fortran libs, last-century cernlib, . . .)
- source code and examples: https://github.com/jolange/BLUE-py
- PIP-installable:

```
pip install blue_combine
```

or

```
git clone git@github.com:jolange/BLUE-py.git
pip install [-e] .
```

blue_combine example

```python
import blue_combine as blue

measurements = blue.Measurements([9.5, 11.9, 11.1, 8.9])
E = blue.CovarianceMatrix(
    [[2.74, 1.15, 0.86, 1.31],
     [1.15, 1.67, 0.82, 1.32],
     [0.86, 0.82, 2.12, 1.05],
     [1.31, 1.32, 1.05, 2.93]]
)

print('Measurements:', measurements)
print('Covariance Matrix:')
print(E)

comb = blue.BLUE(measurements, E)
print()
print('BLUE result:')
print(comb)
```

output:

```
Measurements: [[ 9.5 11.9 11.1  8.9]]
Covariance Matrix:
[[2.74 1.15 0.86 1.31]
 [1.15 1.67 0.82 1.32]
 [0.86 0.82 2.12 1.05]
 [1.31 1.32 1.05 2.93]]

BLUE result:
weights = [[0.14507476 0.46957738 0.34729705 0.03805081]]
result = 11.1598 +- 1.13404
chi2/Ndf = 6.0/3, p=0.111
```

blue_combine other features

```python
# giving errors and correlations
errors = blue.Errors([2, 3])
E = blue.CovarianceMatrix.from_correlation_matrix(
    errors,
    [[1, .5],
     [.5, 1]])
```

```python
# build covariance matrices for different uncertainty sources
components['Stat'] = blue.CovarianceMatrix.from_correlation_matrix
    ([0.20, 0.30], UNCORR)
components['JES']  = blue.CovarianceMatrix.from_correlation_matrix
    ([0.15, 0.16], CORR)
components['ISR']  = blue.CovarianceMatrix.from_correlation_matrix
    ([0.25, 0.32], CORR)

# build total covariance matrix
E = blue.CovarianceMatrix(np.zeros((n, n)))
for comp in components.values():
    E += comp
```

m_t BLUE combination

- ℓ+jets: $m_t = 172.25 \pm 0.08\,(\text{stat+JSF}) \pm 0.62\,(\text{syst})$ GeV
- all-jets: $m_t = 172.34 \pm 0.20\,(\text{stat+JSF}) \pm 0.70\,(\text{syst})$ GeV
- 22 correlation groups (each with $\rho = 100\%$ or 0%)
  following world-combination assumptions as close as possible
  $\Rightarrow$ overall correlation of 0.86

BLUE result

- weights: 1.196, -0.196
- result $= 172.232 \pm 0.627$
- chi2/Ndf $= 0.1/1$, p=0.823

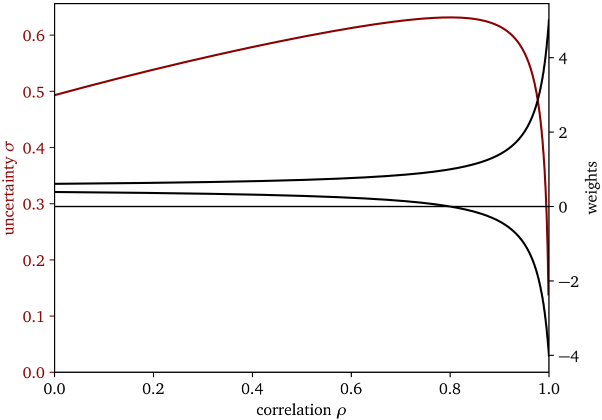
m_t BLUE combination

- ℓ+jets: $m_t = 172.25 \pm 0.08\,(\text{stat+JSF}) \pm 0.62\,(\text{syst})$ GeV
- all-jets: $m_t = 172.34 \pm 0.20\,(\text{stat+JSF}) \pm 0.70\,(\text{syst})$ GeV
- 22 correlation groups (each with $\rho = 100\%$ or 0%) following world-combination assumptions as close as possible $\Rightarrow$ overall correlation of 0.86

BLUE result

- weights: 1.196, -0.196
- result = 172.232 ± 0.627
- chi2/Ndf = 0.1/1, p=0.823

Correlation scan



- strong correlations: BLUE weights may become negative
- often considered as a sign that the correlations are overestimated
- frequently used modification: “reduced correlations” ρ_{XY}^{red}

“Reduced correlations” ρ_{XY}^{red}

- for measurements X and Y, the correlation of an uncertainty component is

$$\rho_{XY} = \text{Cov}_{XY} / (\sigma_X \sigma_Y)$$

- labelling such that $\sigma_X \leq \sigma_Y$
- splitting σ_Y into $(\sigma_X) + (\sigma_Y - \sigma_X)$
- only (σ_X) part is assumed to be fully correlated with X, remaining $(\sigma_Y - \sigma_X)$ part treated as uncorrelated

reduced covariance:

$$\text{Cov}_{XY}^{\text{red}} = \rho_{XX} \sigma_X \sigma_X = \sigma_X^2$$

corresponding reduced correlation:

$$\rho_{XY}^{\text{red}} = \text{Cov}_{XY}^{\text{red}} / (\sigma_X \sigma_Y) = \sigma_X^2 / (\sigma_X \sigma_Y) = \sigma_X / \sigma_Y \, .$$

```
CovarianceMatrix.from_correlation_matrix(cls, errors, rho, reduce_correlations=False)
```

m_t BLUE combination

- ℓ+jets: $m_t = 172.25 \pm 0.08\,(\text{stat+JSF}) \pm 0.62\,(\text{syst})$ GeV
- all-jets: $m_t = 172.34 \pm 0.20\,(\text{stat+JSF}) \pm 0.70\,(\text{syst})$ GeV

BLUE result

- overall correlation of 0.86
- weights: 1.196, -0.196
- result = 172.232 ± 0.627
- chi2/Ndf = 0.1/1, p=0.823

BLUE result, red. correlations

- overall correlation of 0.64
- weights: 0.795, 0.205
- result = 172.268 ± 0.619
- chi2/Ndf = 0.0/1, p=0.884

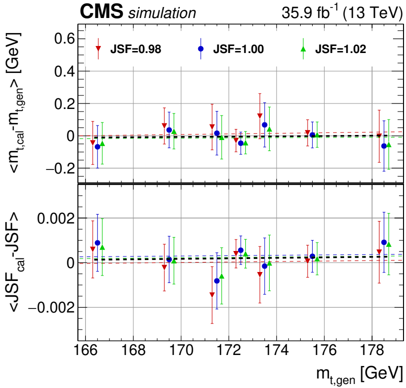
“Real” combination with ℓ+jets

measurement in the ℓ+jets channel uses the same mass extraction method
→ can perform a **single** mass extraction with the total likelihood:

$$\mathcal{L}(m_t, \text{JSF}) = \mathcal{L}_A(m_t, \text{JSF})\mathcal{L}_L(m_t, \text{JSF})$$

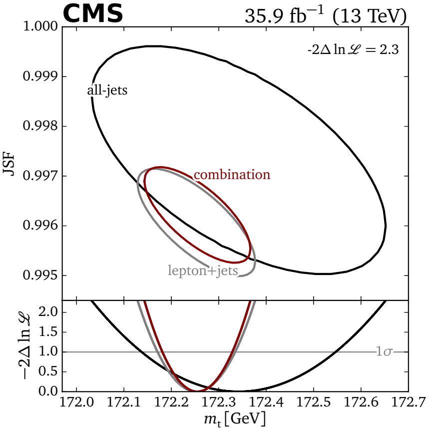
no assumption on correlations necessary

keeping the “single-channel calibration curves” and just check:

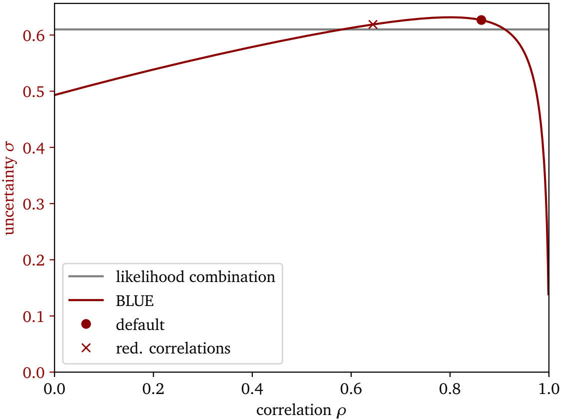


Combination result

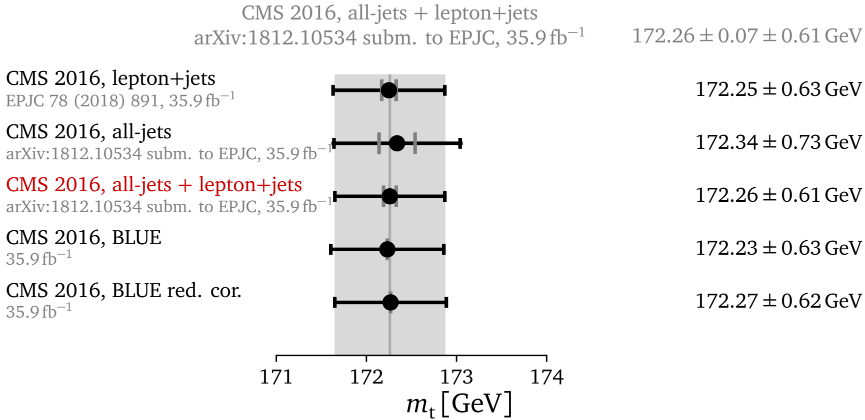
$$m_t = 172.26 \pm 0.07\,(\text{stat+JSF}) \pm 0.61\,(\text{syst})\ \text{GeV}$$



Comparison with BLUE



Summary & outlook



- combined measurement in all-jets and ℓ+jets channel
- exploiting full information
- BLUE combination(s) for comparison
- simultaneous nuisance parameter fit using both channels might provide useful constraints

BACKUP

BLUE input

```
#                                 LJ    AJ
components['FitCal']  = source([0.05, 0.06], UNCORR, reduce_corr)
components['intJES']  = source([0.04, 0.04], CORR,   reduce_corr)
components['MPFJES']  = source([0.07, 0.08], CORR,   reduce_corr)
components['uncJES']  = source([0.16, 0.12], CORR,   reduce_corr)
components['JER']     = source([0.12, 0.04], CORR,   reduce_corr)
components['BTag']    = source([0.03, 0.02], CORR,   reduce_corr)
components['PU']      = source([0.05, 0.04], CORR,   reduce_corr)
components['BKG']     = source([0.02, 0.07], UNCORR, reduce_corr)
components['Trigger'] = source([0.00, 0.02], UNCORR, reduce_corr)
components['flJES']   = source([0.39, 0.34], CORR,   reduce_corr)
components['bMod']    = source([0.12, 0.09], CORR,   reduce_corr)
components['PDF']     = source([0.02, 0.01], CORR,   reduce_corr)
components['Q']       = source([0.01, 0.04], CORR,   reduce_corr)
components['ME/PS']   = source([0.07, 0.24], CORR,   reduce_corr)
components['MCGen']   = source([0.20, 0.31], CORR,   reduce_corr)
components['ISR']     = source([0.07, 0.14], CORR,   reduce_corr)
components['FSR']     = source([0.13, 0.18], CORR,   reduce_corr)
components['topPt']   = source([0.01, 0.03], CORR,   reduce_corr)
components['UE']      = source([0.07, 0.17], CORR,   reduce_corr)
components['ERD']     = source([0.07, 0.24], CORR,   reduce_corr)
components['CR']      = source([0.31, 0.36], CORR,   reduce_corr)
components['Stat']    = source([0.08, 0.20], UNCORR, reduce_corr)
```