Matrix-Element Method Tutorial: MadWeight

Olivier Mattelaer
IPPP/Durham

PLAN

- Instruction
  - ➡ Theoretical reminder
  - ➡ description of the exercise
- Installation
- Tutorial
- Conclusion

- Associate to each experimental event characterised by $\boldsymbol{p}^{vis}$, the probability $\mathcal{P}(\boldsymbol{p}^{vis}|\alpha)$ to be produced and observed following a theoretical assumption α

$$\mathcal{P}(\boldsymbol{p}^{vis}|\alpha) = \frac{1}{\sigma_\alpha^{vis}} \int d\Phi dx_1 dx_2 |M_\alpha(\boldsymbol{p})|^2 W(\boldsymbol{p}, \boldsymbol{p}^{vis})$$

- $|M_\alpha(\boldsymbol{p})|^2$ is the squared matrix element
- $W(\boldsymbol{p}, \boldsymbol{p}^{vis})$ is the transfer function
- $\int d\Phi dx_1 dx_2$ is the phase-space integral
- σ_α^{vis} is the cross-section (after cuts)

Weight Evaluation

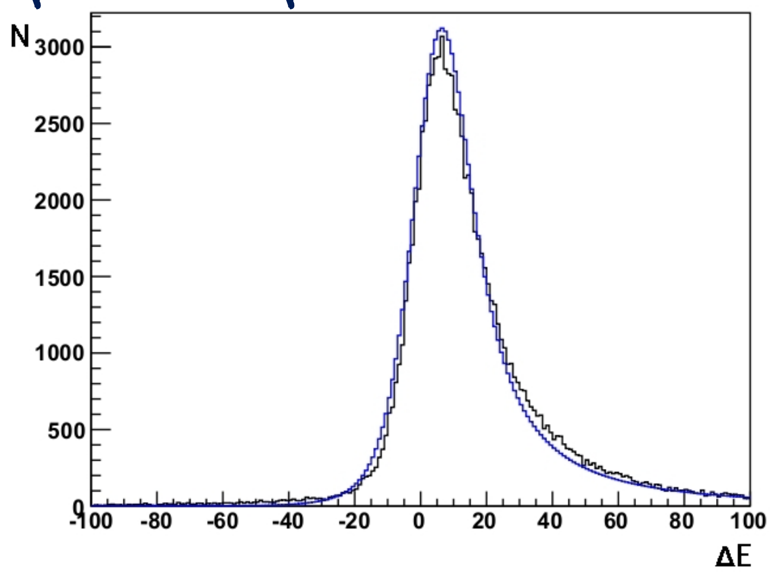
$$\mathcal{P}(\boldsymbol{p}^{vis}|\alpha) = \frac{1}{\sigma_\alpha}\int d\Phi dx_1 dx_2 |M_\alpha(\boldsymbol{p})|^2 W(\boldsymbol{p},\boldsymbol{p}^{vis})$$

Four Elements:

- cross-section
- matrix-element
- transfer function
- integration

Computed via

- MadGraph5
- MadGraph5
- fitted from MC



- MadWeight

MadWeight

MadWeight returns:

$$\mathcal{P}(\boldsymbol{p}^{vis}|\alpha) = \cancel{\frac{1}{\sigma_\alpha}} \int d\Phi dx_1 dx_2 |M_\alpha(\boldsymbol{p})|^2 W(\boldsymbol{p}, \boldsymbol{p}^{vis})$$

- Likelihood need to be correctly normalized

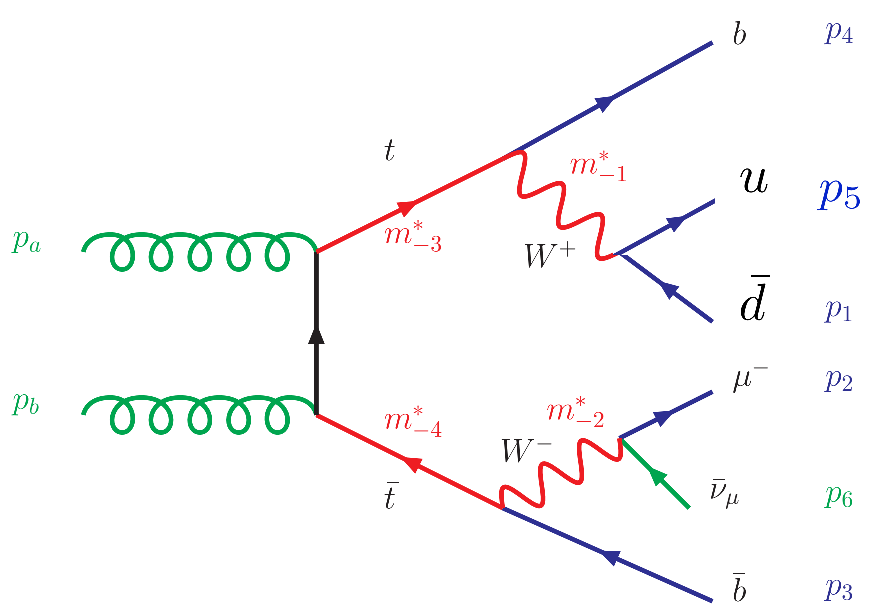
$$L(\alpha) = \prod_{i=1}^{N} \mathcal{P}(\boldsymbol{p}_i^{vis}|\alpha)$$

$$-ln(L(\alpha)) = -\sum_{i=1}^{N} ln(P_i^{MW}) + N * ln(\sigma_\alpha)$$

History

- 2009: MadGraph4 implementation
- 2011: Private implementation in MadGraph5
  - Initial State Radiation Support
  - SubProcess grouping (speed)
  - NWA (speed)
- 2013: MadWeight5
  - Improve cluster support (speed)
  - MC over jet/parton assignment (speed)
  - pre-training (speed)
  - better multi-channel (speed)
- 2014: MadWeight5
  - Support for multi-transfert function estimated on the same phase-space point (speed)
  - Module of preselection of the jet/parton assignment (speed)

Jet-Parton assignment




```
0            12   3587
1     1   0.935  5.230   83.80    0.00   1.0   0.0    0.00   0.0   0.0
2     4  -0.161  1.878   85.60    9.66   7.0   0.0    1.10   0.0   0.0
3     4  -0.223  5.295   45.64    5.43   3.0   0.0    0.30   0.0   0.0
4     4   0.695  2.208   37.99    7.68   8.0   0.0    3.63   0.0   0.0
5     4   1.164  3.357   49.01    6.95  13.0   0.0    2.66   0.0   0.0
6     6   0.000  6.035   39.48    0.00   0.0   0.0    0.00   0.0   0.0
```

Which jet correspond to which parton?

MW4: one integral per permutation
MW5: Monte-Carlo over the permutation

process	tf	permutation	Sum/MonteCarlo
tt semi leptonic	delta	24	7.5
tt semi leptonic	gauss	24	2
tt di leptonic	gauss	2	0.6
w+ j j	delta	2	1.5
tth (semi lept)	gauss	720	20

Speed benchmark

process	perm	MW4	MW5
tt semi lept	24	1h16	41s
tt fully lept	2	46s	10s
tth semi lept	720	> 2 days	10min
tth semi lept	48	> 3h	6min
tth fully lept	24	>1h	1min
h> w+ w- > 1lept	2	59s	<5s
h> w+ w- > 2lept	1	8s	<5s
z b b	24	39m	18s
zh	24	43m	<5s

running on 1core of a Intel core i7 2.3Ghz

Exercises

Installation:

- Download the tar-ball from indico
  - everything is inside
  - you are ready to go.

Exercise 1:

- Extract the top-quark mass from a sample of tt~ events (fully leptonic decay) at parton level
- Use the likelihood method (do not forget to normalize it)

Exercise 2:

- Same but at detector level
- Show the sensitivity to the transfer function

Exercise 3:

- Add background

Installation

- Download the package from indico
- $> tar -xzpf Tutorial.tgz
- You have
  - ➡ Tutorial/samples/: samples of events that we will use
  - ➡ Tutorial/MG5_aMC: code of MadGraph/ MadWeight
  - ➡ Tutorial/MG5_aMC/bin/mg5_amc : executable
  - ➡ Tutorial/mg5: symbolic link to the previous executable

Exercises

Installation:

- Download the tar-ball from indico
    - everything is inside
    - you are ready to go.

Exercise 1:

- Extract the top-quark mass from a sample of tt~ events (fully leptonic decay) at parton level
- Use the likelihood method (do not forget to normalize it)

Exercise 2:

- Same but at detector level
- Show the sensitivity to the transfer function

Exercise 3:

- Add background

Exercise I: parton level

creation of the dedicated code:

- run the executable
  - $> cd Tutorial
  - $> ./mg5 (or ./MG5_aMC/bin/mg5_amc) [open a prompt]
  - mg5_amc> generate p p > t t~, t > b e+ ve, t~ > b~ mu- vm~
  - mg5_amc> output madweight MW_TT_FULL_LEPT

result:

- Now you have a directory in your current directory: MW_TT_FULL_LEPT

comments:

- All LO MG5 syntax are supported (but no NLO)
- For semi-leptonic:
  - mg5_amc> generate p p > t t~, t > b e+ ve, t~ > b~ j j
  - mg5_amc> add process p p > t t~, t > b j j, t~ > b~ e- ve~

Exercise I: parton level

computing the weight:

- mg5_amc> launch MW_TT_FULL_LEPT

OR

- $> cd MW_TT_FULL_LEPT
- ./bin/madweight.py

output:

```
Do you want to edit a card (press enter to bypass editing)?
  1 / param      : param_card.dat
  2 / run        : run_card.dat
  3 / madweight  : madweight_card.dat
  4 / transfer   : transfer_card.dat
  5 / lhco       : input.lhco
 you can also
   - enter the path to a valid card or banner.
   - use the 'set' command to modify a parameter directly.
     The set option works only for param_card and run_card.
     Type 'help set' for more information on this command.
   - call an external program (ASperGE/MadWidth/...).
     Type 'help' for the list of available command
   - use the 'change_tf' command to set a transfer functions.
 [0, done, 1, param, 2, run, 3, madweight, 4, enter path, ... ][60s to answer]
```

comments:

- param: default model parameter
- run: accelerator/cut (if used)
- madweight: see later
- transfer: definition of TF
- lhco: input data for events

Exercise I: parton level

define transfer function:

- answer the question with
  - change_tf all_delta

output:

- The same question is re-asked.

comments:

- This defines ONLY the functional form of the transfer function
- The numerical coefficient can changed in transfer_card.dat
- They are no numerical coefficient for this transfer function
  - neutrino are not constraint
  - all other particles are assumed to be perfectly measured
- You can define your own functional form if needed (see backup)

Exercise I: parton level

define the input file / number of events /…

- answer the question with
  - madweight

output

- open the madweight_card within an editor (typically vi)

action

- change
  - the number of event to 25 (nb_exp_events)
  - the input path to .(.)/samples/parton.lhco.gz (inputfile)
  - define a scan over the top mass from 163 to 183 (by step of 5GeV
  - name of the run to “parton" (name)

comment

- Note that this card de-activate the cut of the run_card. (use_cut)
- you can change each of those parameters with “set name VALUE”

MadWeight_card

other interesting parameters

- nb_event_by_node: for running on cluster. (option for cluster define in MG5_aMC/input/mg5_configuration.txt)
- use_cut: defines if the cut of the run_card are applied or not. (in principle no cut should be applied)
- permutation: using only one (or all) jet-parton assignment
- montecarlo: using MC over the jet-parton assignment
- more info on the card: see the online wiki.

Exercise I: parton level

launch the run

- answer the question with
  - done (or just press enter)

What's going on

- The parametrization of the phase-space is chosen
- The actual computation takes places (N_event*Nth)

output

- in Events/parton/
  - un-normalized_likelihood.out: -ln(L) without the cross-section
  - weights.out: column format with the weights
  - output.xml: structured format contains details of the computation
  - The amount of information depends of the "log_level" parameter (can contain weight for each permutation/full log/ ...)

Exercise I: parton level

Compute the normalization factor

- we will use mg5_amc for that
- $> ./mg5 (or ./MG5_aMC/bin/mg5_amc) [open a prompt]
- mg5_amc> generate p p > t t~, t > b e+ ve, t~ > b~ mu- vm~
- mg5_amc> output MG_TT_FULL_LEPT
- mg5_amc>launch

output:

```
The following switches determine which programs are run:
 1 Run the pythia shower/hadronization:                        pythia=OFF
 2 Run PGS as detector simulator:                                 pgs=OFF
 3 Run Delphes as detector simulator:                         delphes=OFF
 4 Decay particles with the MadSpin module:                   madspin=OFF
 5 Add weights to the events based on changing model parameters:   reweight=OFF
  Either type the switch number (1 to 5) to change its default setting,
  or set any switch explicitly (e.g. type 'madspin=ON' at the prompt)
  Type '0', 'auto', 'done' or just press enter when you are done.
 [0, 1, 2, 3, 4, 5, auto, done, pythia=ON, ... ][60s to answer]
```

output:

- we do not need any of this (shower/...) so just press enter

Exercise I: parton level

output:

```
Do you want to edit a card (press enter to bypass editing)?
  1 / param        : param_card.dat
  2 / run          : run_card.dat
  9 / plot         : plot_card.dat
 you can also
   - enter the path to a valid card or banner.
   - use the 'set' command to modify a parameter directly.
     The set option works only for param_card and run_card.
     Type 'help set' for more information on this command.
   - call an external program (ASperGE/MadWidth/...).
     Type 'help' for the list of available command
 [0, done, 1, param, 2, run, 9, plot, enter path][60s to answer]
```

action:

- edit the run_card.dat to remove ALL cut. (type run to edit the card)
- then type “set mt scan:range(163,185,5)” [this defines a scan on the top mass]
- then press enter

output:

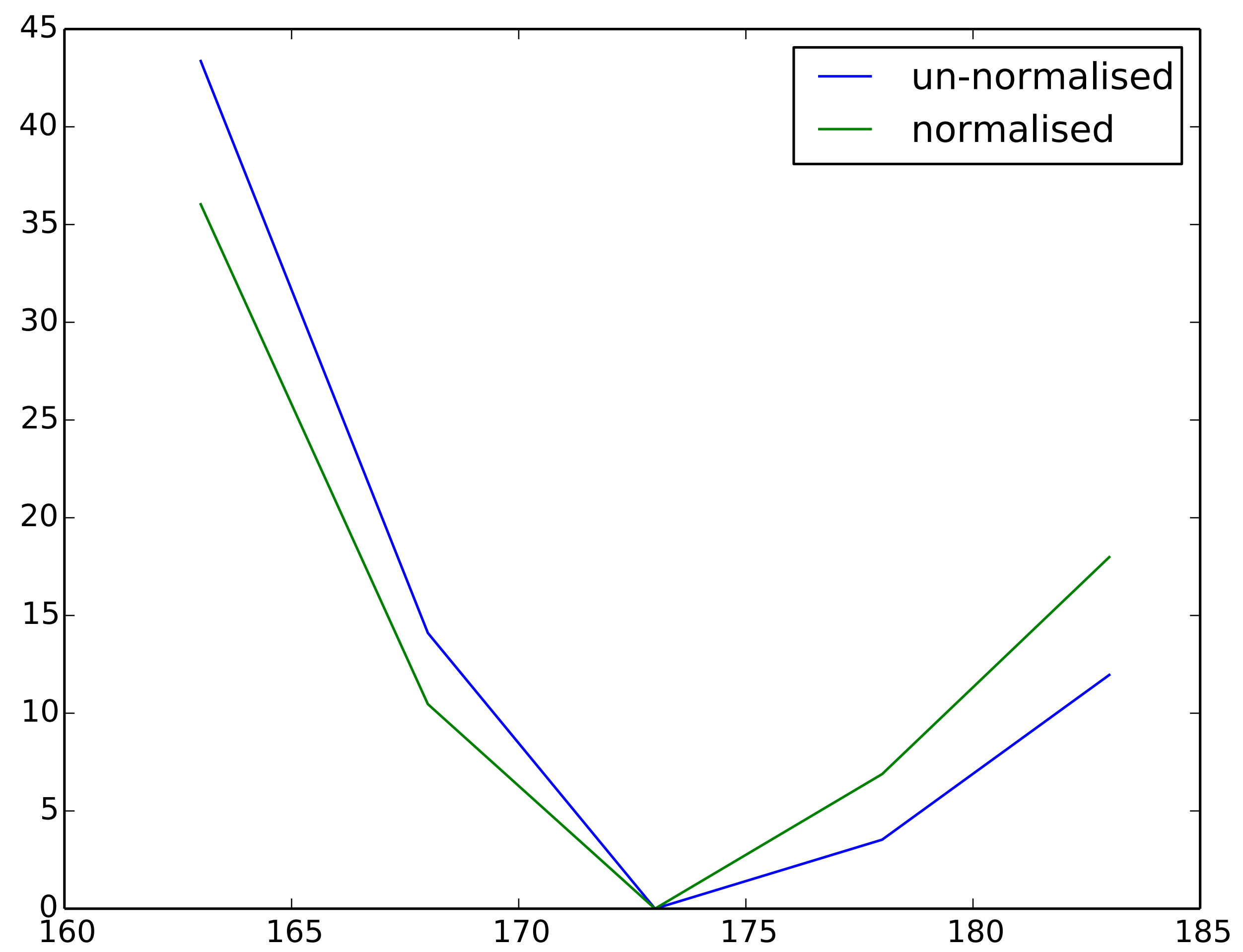
- MG_TT_FULL_LEPT/Events/scan_run_[01-05].txt
  - contains the cross-section for each top mass value

Exercise I: parton level

make the plot

- use your favorite program for that:

Expected output



Exercises

Installation:

- Download the tar-ball from indico
  - everything is inside
  - you are ready to go.

Exercise 1:

- Extract the top-quark mass from a sample of tt~ events (fully leptonic decay) at parton level
- Use the likelihood method (do not forget to normalize it)

Exercise 2:

- Same but at detector level
- Show the sensitivity to the transfer function

Exercise 3:

- Add background

Exercise II: detector

creation of the dedicated code:

- $> cd MW_TT_FULL_LEPT

comments:

- Since the process is the same (but not the transfer function) we can reuse the same directory/code

configure:

- run ./bin/madweight.py
- at the question type “change_tf dbl_gauss_pt_jet”

comment:

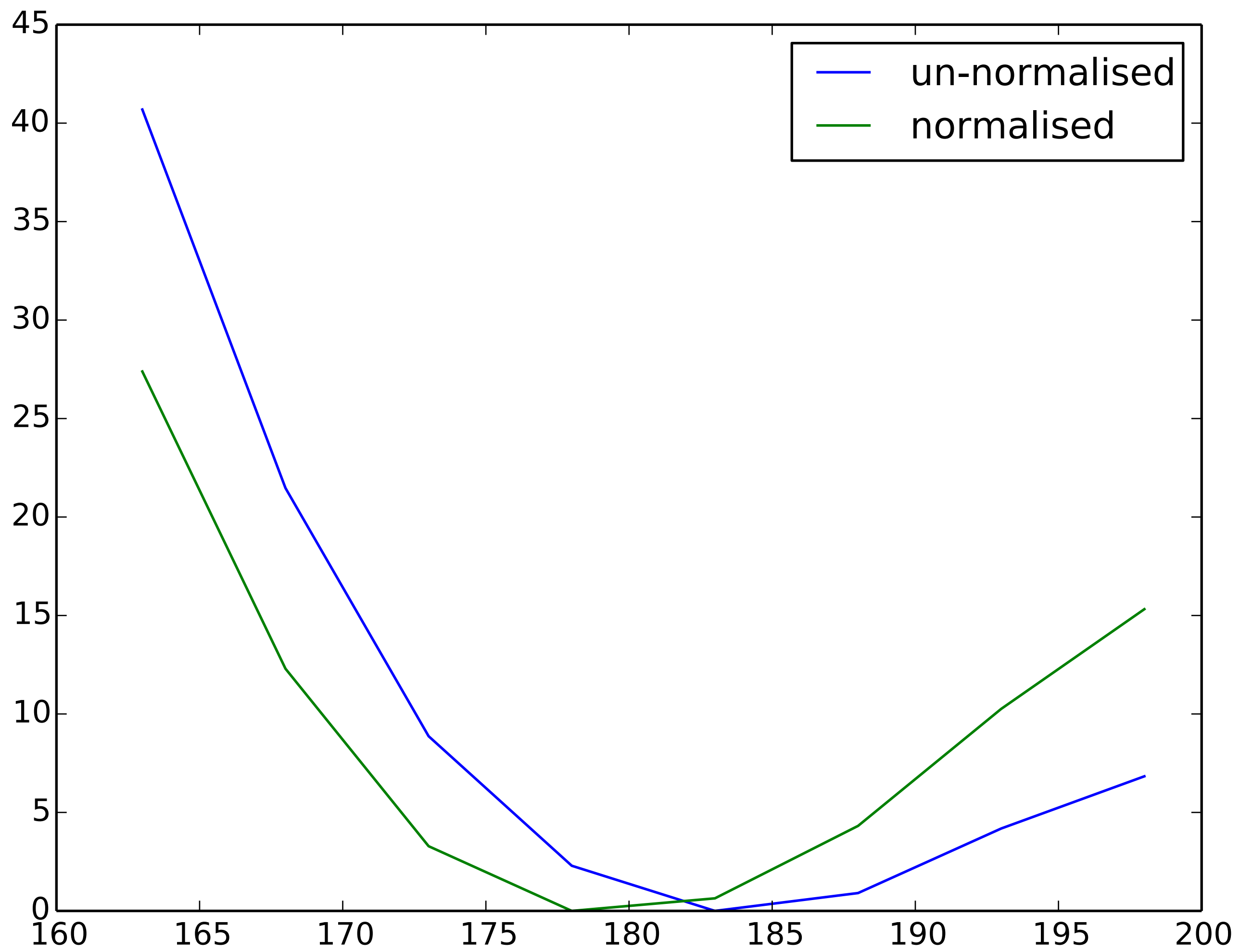
- the same as all_delta but for jet where the pt is related to a resolution function (double gaussian where each parameter is a function of the pt)
- Now you can edit the value of various parameter in the transfer_card

Exercise II: detector

change the path to the event file:

- pass to ../samples/detector.lhco.gz
- change the name to “detector”

expected output:



question:

- explain the bias (the correct value is 173)

Exercise II: detector

study the impact of transfer function

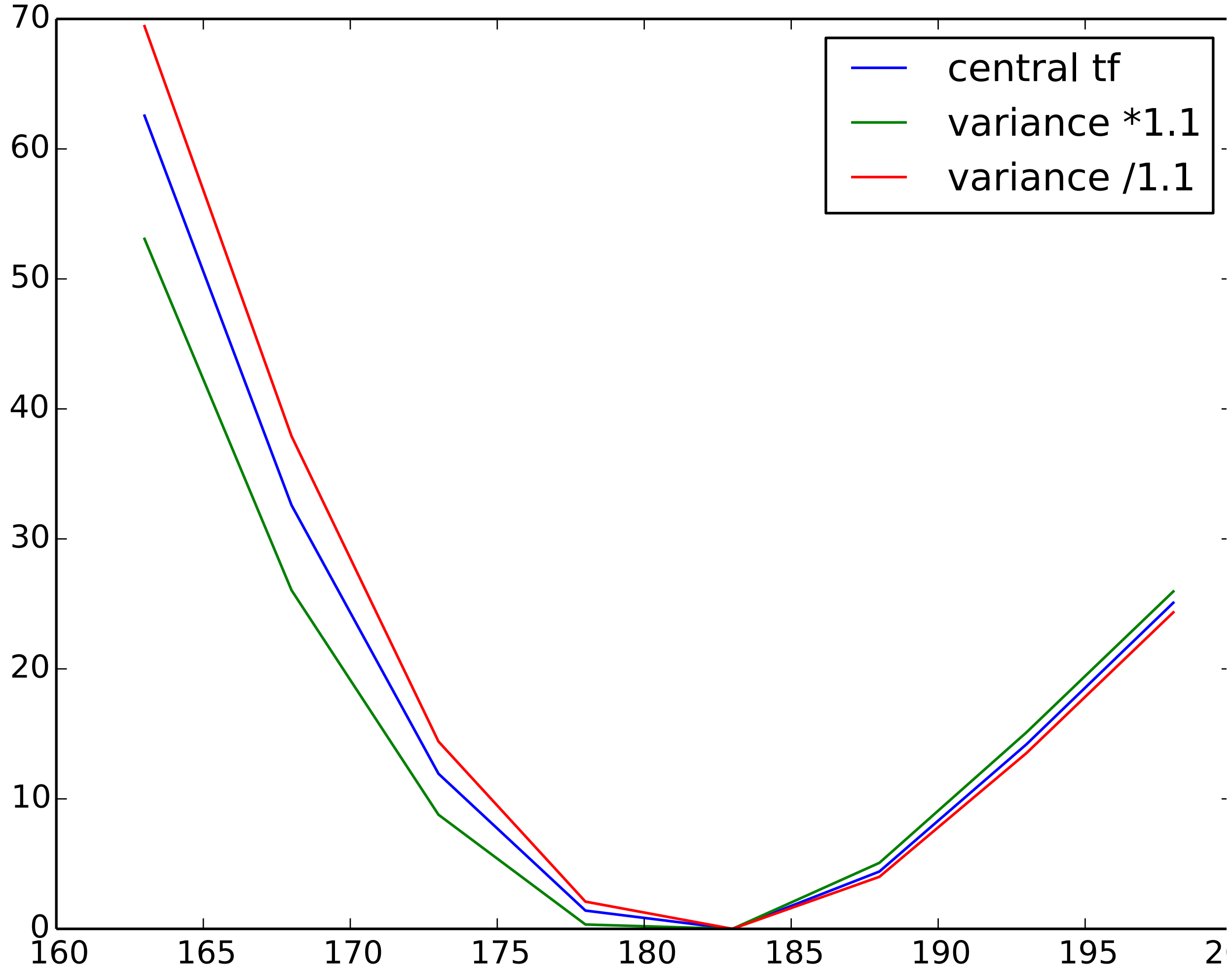
- rerun the same computation but with the variance of the transfer function divide and multiply by two.
- do it in a single run.

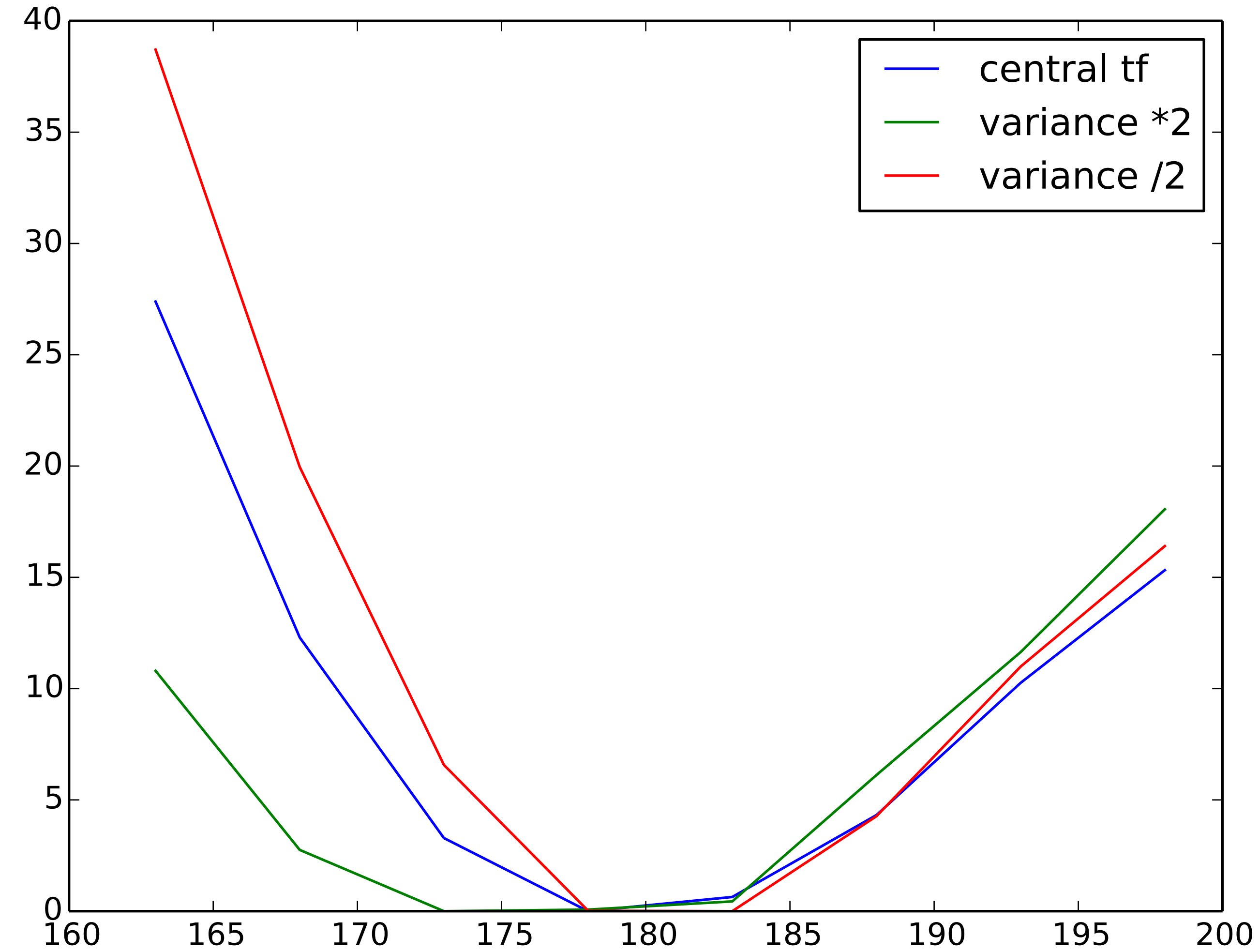
edit transfer_card.dat

- all entry should have three entries:
  - 4 0.9518d0 # sigma first gaussian
  - 4 1.8d0 # for the second tf
  - 4 0.47 # for the third tf
  - unmodified parameter should also have three entries.

run and compare the running time

Exercise II: detector





Exercises

Installation:

- Download the tar-ball from indico
  - everything is inside
  - you are ready to go.

Exercise 1:

- Extract the top-quark mass from a sample of tt~ events (fully leptonic decay) at parton level
- Use the likelihood method (do not forget to normalize it)

Exercise 2:

- Same but at detector level
- Show the sensitivity to the transfer function

Exercise 3:

- Add background

Exercise III: background

run and compare the running time

- background is here w+ w- + 2j
- in samples

Conclusion

- Thanks for your participation, questions and remarks.

Define transfer function

```
##**********************************************************************##
##                          TF JET                                      ##
##**********************************************************************##
<block name='jet'>   #name can be anything
<info> doubel gaussian with parameter depending of the energy </info>
<particles> u,d,s,c,b,g </particles>
# this defined when this tf will be used.the letter correspond to the label in
#        particles.dat
<width_type> large </width_type>
# width_type should be thin or large (thin is for energy acuurate up to 5-10%)
<variable name='E'>
<tf>
        prov1=(#1+#2*dsqrt(pexp(0))+#3*pexp(0))
        prov2=(#4+#5*dsqrt(pexp(0))+#6*pexp(0))
        prov3=(#7+#8*dsqrt(pexp(0))+#9*pexp(0))
        prov4=(#10+#11*dsqrt(pexp(0))+#12*pexp(0))
        prov5=(#13+#14*dsqrt(pexp(0))+#15*pexp(0))

        tf=(exp(-(p(0)-pexp(0)-prov1)**2/2d0/prov2**2))              !first gaussian
        tf=tf+prov3*exp(-(p(0)-pexp(0)-prov4)**2/2d0/prov5**2)      !second gaussian
        tf=tf*((1d0/dsqrt(2d0*pi))/(prov2+prov3*prov5))             !normalisation
</tf>
<width>
        prov2=(#4+#5*dsqrt(pexp(0))+#6*pexp(0))
        prov5=(#13+#14*dsqrt(pexp(0))+#15*pexp(0))

        width=max(prov2,prov5)
</width>
</variable>

 # in this case THETA/PHI are not defined because they are considered
 # in delta (=default)
 # The same syntax apply
</block>
```