The Binoth-Les Houches Accord

Rikkert Frederix
University of Zurich

MC4LHC readiness, CERN, March 30, 2010

Warning

- This talk will be about an interface between two contributions to NLO calculations
- So, do not expect much "physics" here...

Contents

- NLO calculations
- Motivation for the accord
- How does it work?
- Summary

CONTRIBUTIONS TO NLO CALCULATIONS

$$\sigma^{\mathrm{NLO}} = \int_{m+1} d^{(d)}\sigma^R + \int_m d^{(d)}\sigma^V + \int_m d^{(4)}\sigma^B$$

‘Real emission’ NLO corrections

‘Virtual’ or ‘one-loop’ NLO corrections

‘Born’ or ‘LO’ contribution

IR divergence

$$\sigma^{\rm NLO} = \int_{m+1} d^{(d)}\sigma^R + \int_m d^{(d)}\sigma^V + \int_m d^{(4)}\sigma^B$$

- Real emission -> IR divergent
- (UV-renormalized) virtual corrections -> IR divergent
  - After integration, the sum of all contributions is finite (for infrared-safe observables)
- Relative straightforward to get explicit poles for virtual corrections, because loop integrals (scalar integrals) are done analytically

Subtraction terms

$$\sigma^{\mathrm{NLO}} = \int_{m+1} d^{(d)}\sigma^R + \int_m d^{(d)}\sigma^V + \int_m d^{(4)}\sigma^B$$

Subtraction terms

$$\sigma^{\mathrm{NLO}} = \int_{m+1} d^{(d)}\sigma^R + \int_m d^{(d)}\sigma^V + \int_m d^{(4)}\sigma^B$$

$$\sigma^{\mathrm{NLO}} = \int_{m+1} \left[d^{(4)}\sigma^R - d^{(4)}\sigma^A\right] + \int_m \left[d^{(4)}\sigma^B + \int_{\mathrm{loop}} d^{(d)}\sigma^V + \int_1 d^{(d)}\sigma^A\right]_{\epsilon=0}$$

- Include subtraction terms to make real emission contributions and virtual contributions separately finite
- All can be integrated numerically

OLP's and MC codes

$$\sigma^{\mathrm{NLO}} = \int_{m+1} \left[d^{(4)}\sigma^{R} - d^{(4)}\sigma^{A} \right] + \int_{m} \left[d^{(4)}\sigma^{B} + \int_{\mathrm{loop}} d^{(d)}\sigma^{V} + \int_{1} d^{(d)}\sigma^{A} \right]_{\epsilon=0}$$

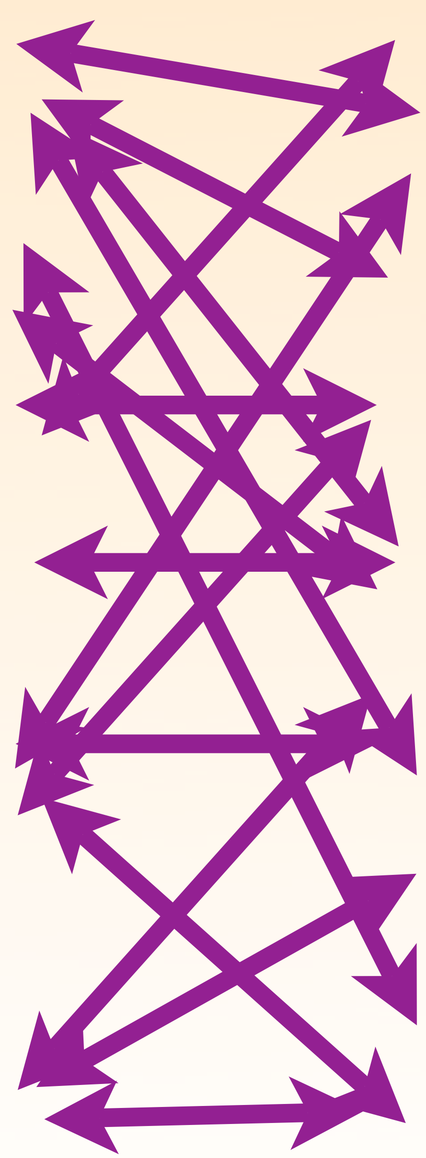
MC code	One-Loop Program (OLP)
Tree level	Loops
Subtraction of singularities	Numerical instabilities
Efficient Phase-space integration	Evaluation time per phase-space point

The codes

MC codes	One-Loop Programs
Sherpa	BlackHat
MadFKS	GOLEM
Helac-Dipoles	Rocket
YourPrivateCode	Helac-1Loop
...	YourPrivateCode
	...

The codes

MC codes	One-Loop Programs
Sherpa	BlackHat
MadFKS	GOLEM
Helac-Dipoles	Rocket
YourPrivateCode	Helac-1Loop
...	YourPrivateCode
	...



The idea

- **Facilitate the information exchange between the MC codes and the OLPs**
- It should NOT constrain the OLP (nor the MC code) in any way
  Not a standard on what kind of information*, but more on the way it should be passed.
- OLP and MC might work in completely different ways
  Amplitudes may be created on the fly, or read from a library of processes

The advantages

- Switching between codes becomes easy
  Model parameters etc. should be set automatically: checking codes becomes much simpler
- If you write your own OLP or MC code, you know how to link it to existing codes
  Modular problem/calculation allows for modular solutions
- Our (experimental) colleagues can still use their favorite MC code (e.g. Sherpa or MG/ME), but then at NLO, using the most efficient OLP

Binoth-Les Houches Accord

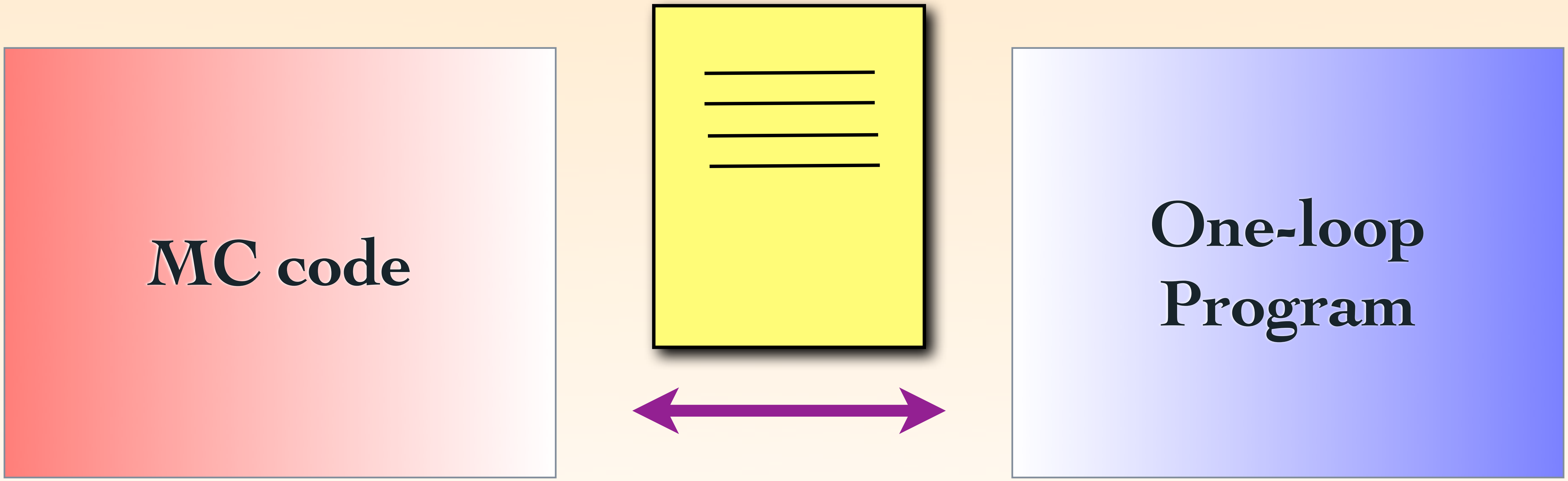
"Dedicated to the memory of, and in tribute to, Thomas Binoth, who led the effort to develop this proposal for Les Houches 2009"

- **Initialization phase**
  MC code communicates basic information about the process to the OLP. OLP answers if it can provide the loop corrections.

- **Run-time phase**
  MC code queries the OLP for the value of the one-loop contributions for each phase-space point.

arXiv:1001.1307 [hep-ph]

Initialization phase



MC code writes an order file
OLP replies with a contract file

```
# example order file

MatrixElementSquareType CHsummed
IRregularisation        CDR
OperationMode           LeadingColor
ModelFile               ModelInLHFormat.slh
SubdivideSubprocess     yes
AlphasPower             3
CorrectionType          QCD

#g g   ->   t tbar g
  21 21 -> 6 -6 21
#u ubar   ->   t tbar g
  2 -2 -> 6 -6 21
#u g   ->   t tbar u
  2 21 -> 6 -6 2
```

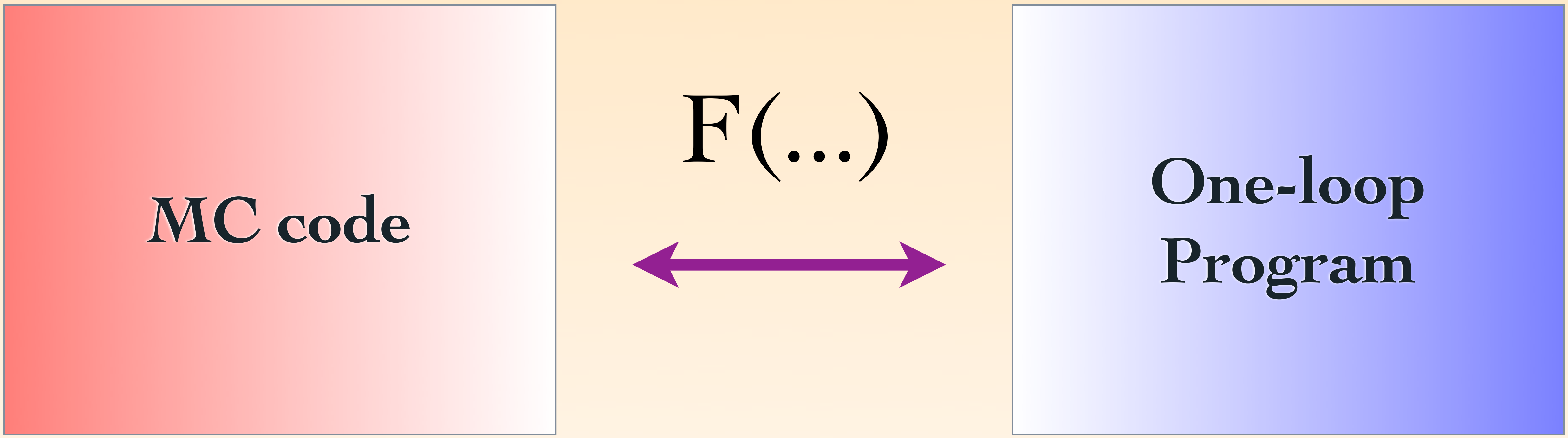
MC code → order file → **OLP**

```
# example contract file
# authors of OLP, citation policy, etc

MatrixElementSquareType CHsummed             | OK
IRregularisation        CDR                  | OK
OperationMode           LeadingColor         | OK
ModelFile               ModelInLHFormat.slh  | OK
SubdivideSubprocess     yes                  | OK
AlphasPower             3                    | OK
CorrectionType          QCD                  | OK

#g g   ->   t tbar g
  21 21 -> 6 -6 21      | 2 13 35
#u ubar   ->   t tbar g
  2 -2 -> 6 -6 21       | 1 29
#u g   ->   t tbar u
  2 21 -> 6 -6 2        | 3 8 23 57
```

MC code

OLP

Run-time phase




```
OLP_Start(..)

OLP_EvalSubProcess(..)
```

OLP_Start(..)

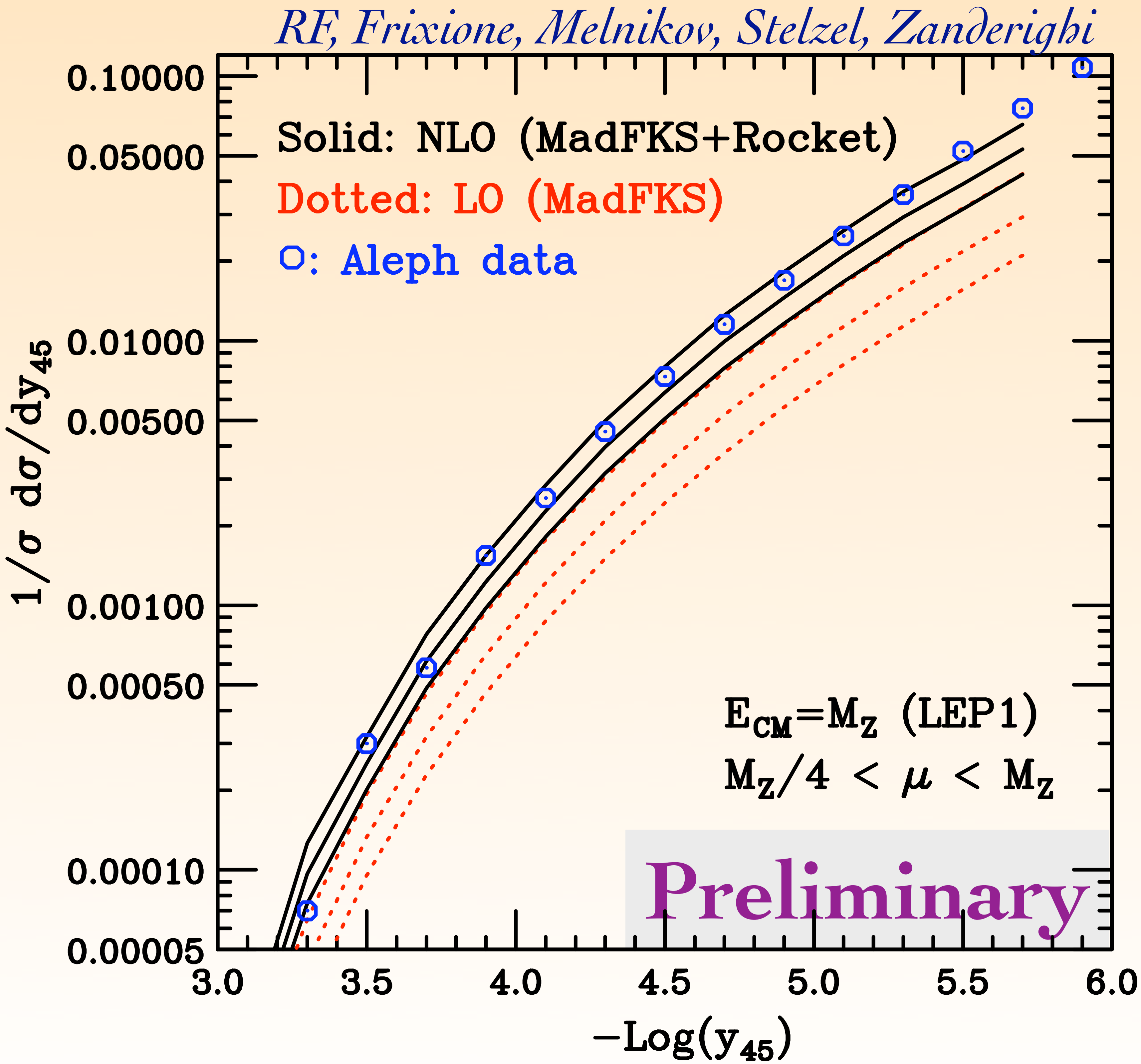
- Should be called once (from MC code) at start up, to confirm the contract and initialize the process
- Two arguments:
  - String with the location of the agreed contract file
  - OLP returns with integer: ‘1’ if all okay, ‘0’ if some error occurred

OLP_EvalSubProcess(..)

- Should be called (from MC code) for every phase-space point
- Five arguments:
  - Integer label of the process
  - Array of momenta and masses of the particles
  - Renormalization scale
  - Strong coupling at the above scale
  - OLP returns array of the results

AN EXAMPLE

- e+e- to 5 jets at NLO
- MadFKS + Rocket
- Results checked with MadFKS + BlackHat



Conclusions

- The Binoth-Les Houches Accord describes an interface between MC codes and One-Loop Programs
- As far as I know, the Binoth-Les Houches Interface has already been implemented and proven to work in
  - Sherpa & MadFKS
  - BlackHat & GOLEM & Rady & Rocket
  - Hopefully many others will follow
- This talk was focussed on QCD corrections, but the Accord also describes the interface for EW corrections
- More details on syntax etc. can be found in

arXiv:1001.1307 [hep-ph]