QUnfold: Quantum Annealing for Distributions Unfolding in High-Energy Physics

France-Berkeley PHYSTAT Conference on Unfolding *10-13 June 2024*

Gianluca Bianco, *PhD student in Physics*

Simone Gasperini, *PhD student in Data Science and Computation*
Marco Lorusso, *PhD student in Physics*

University of Bologna & INFN

Introduction

Outline of the talk:

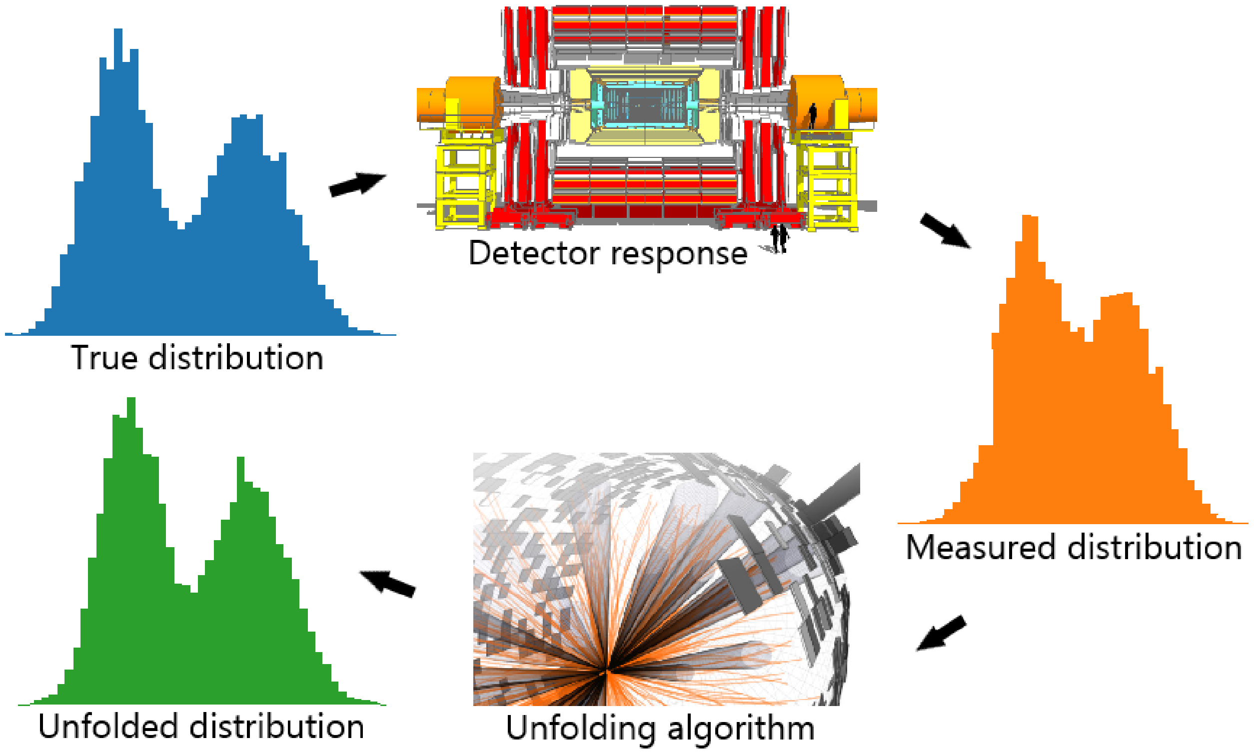
- Brief unfolding introduction
- Quantum computing and quantum annealing
- QUBO problems formulation
- D-Wave quantum annealer
- QUnfold: unfolding with quantum annealing
- Implementation strategy
- Tests on simulated data
- Differential cross-sections measurement
- Conclusions and outlooks

Team members:

- Gianluca Bianco (me, PhD) - Bologna
- Simone Gasperini (PhD) - Bologna
- Marco Lorusso (PhD) - Bologna

What is unfolding?

- In **High-Energy Physics** (HEP) experiments each measurement apparatus has a unique signature in terms of *detection efficiency*, *resolution,* and *geometric acceptance*
- The overall effect is that the distribution of some measured observable in a given physical process is *biased* and *distorted*
- **Unfolding** is the mathematical technique to correct for this distortion and recover the original distribution



$$\vec{\mu} = R\vec{x}$$

"reco" distribution

response matrix

unfolded distribution

Unfolding techniques

Classical unfolding methods in HEP:

- Standard matrix inversion (never used in practice)
- Bin-by-bin unfolding (never used in practice)
- Likelihood-based unfolding (SVD)
- Iterative Bayesian unfolding (IBU)

$$\vec{\mu} = R\vec{x}$$

$\vec{\mu}$: "reco" distribution; R: response matrix; $\vec{x}$: unfolded distribution

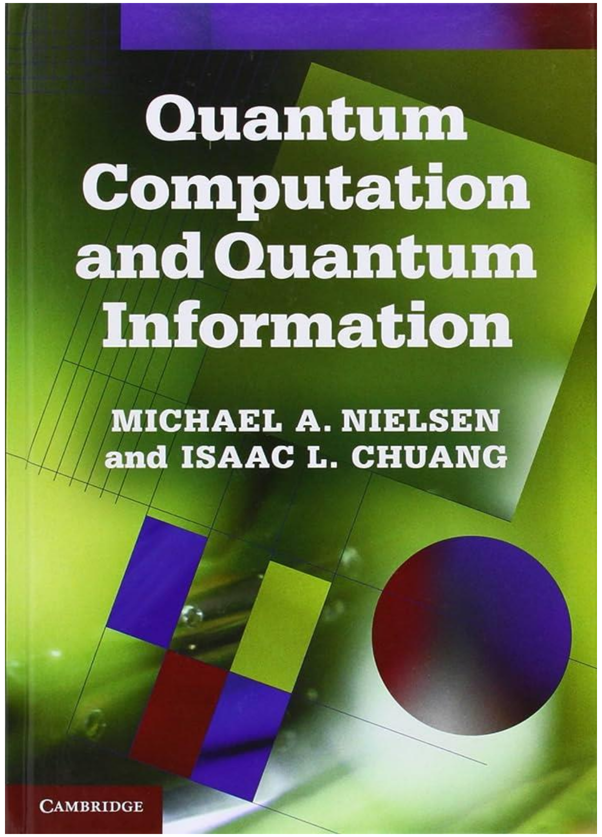
"Quantum" unfolding methods:

- First proof-of-concept by *R. Di Sipio* et al in 2019: the model worked only on really small-sized problems (very few bins and entries) using the D-Wave 2000Q quantum annealer machine
- Our open-source experimental proposal is *QUnfold*

Quantum computing - introduction

Key concepts:

- Quantum computing is a science based on principles of *information theory* and *quantum mechanics*
- Quantum algorithms are particular algorithms that exploit some properties deriving from quantum mechanics (ex: Shor, Groover, etc…)
- In order to work, quantum algorithms need to operate through computers capable of manipulating objects in which the quantum component is sufficiently manifest
- Such computers are called **Quantum Computers**



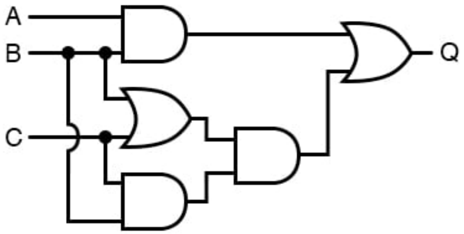
Quantum computing is based on **3 fundamental quantum concepts**:

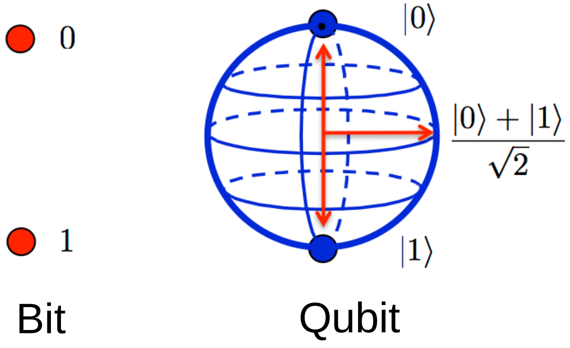
- *Superposition principle*
- *Quantum entanglement*
- *Tunneling effect*

Quantum computing - architecture

Classical computing:

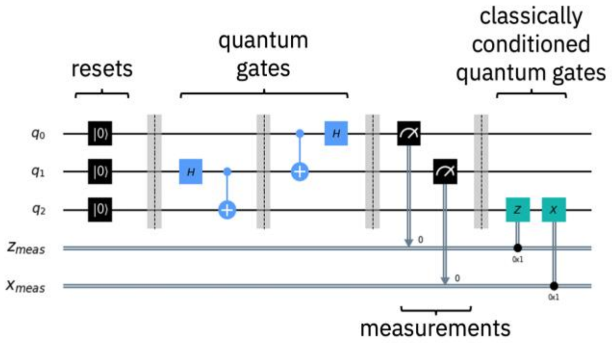
- Unit of measurement of classical information is the classical bit
- Bit can assume values 0 or 1
- Classical computing is performed by creating classical circuits
- Circuits are processed by the CPU





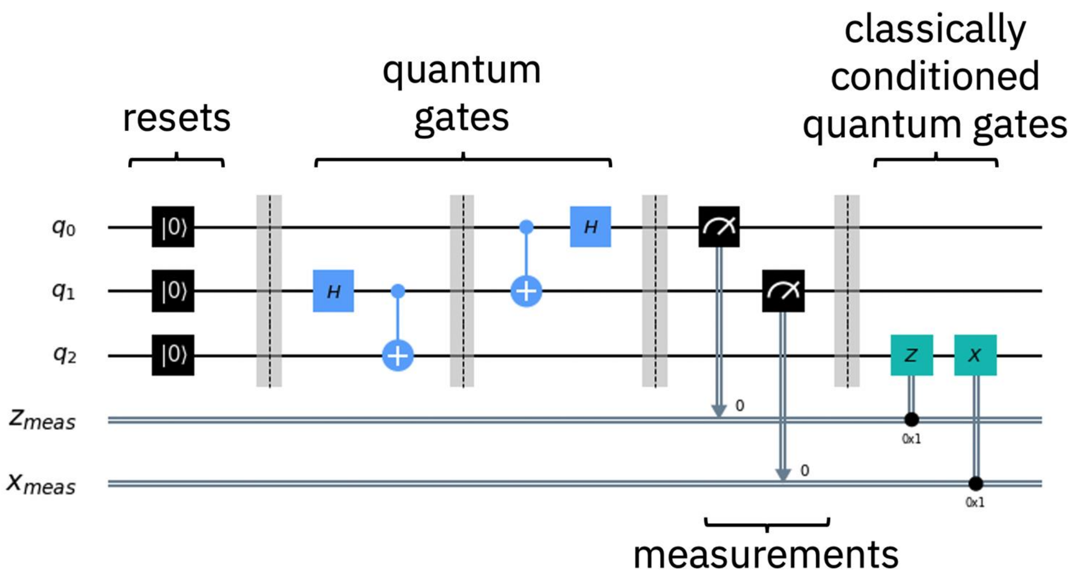
Quantum computing:

- Unit of measurement of quantum information is the qubit (mix of 0 and 1 states)
- Qubit manifests evidence of quantum behaviors like superposition
- Qubits can be represented by an atom, a trapped ion, etc …
- Quantum computing is performed by creating quantum circuits
- Circuits are processed by the QPU



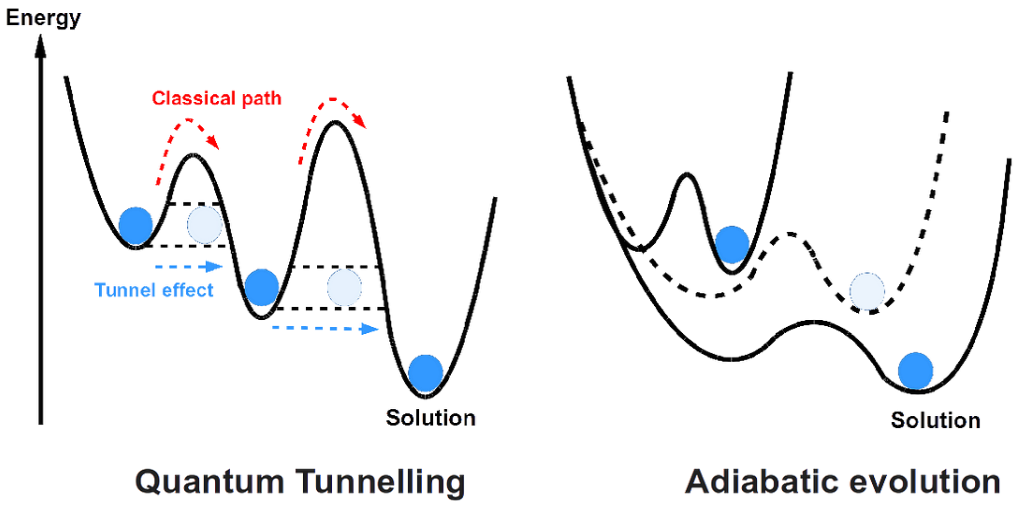
Quantum computing and quantum annealing

Gate-based quantum computing
(eg: IBM, Google)



Universal

Quantum-annealing-based quantum computing
(eg: D-Wave)



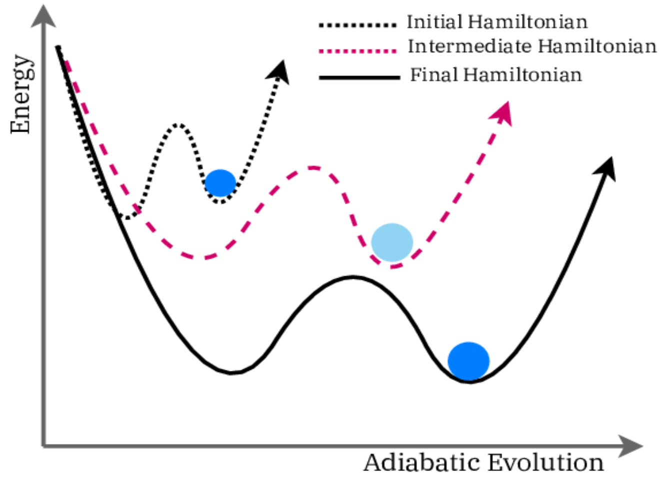
Only quantum annealing can be performed

What is quantum annealing?

- The quantum-mechanical system is prepared in the known ground-state of an **initial Hamiltonian** H_{init}
- The target solution is encoded in the ground-state of a **final Hamiltonian** H_{fin}, written as the energy/cost function of a Quadratic Unconstrained Binary Optimization problem (QUBO problem)
- The system evolution is controlled by the following time-dependent Hamiltonian:

$$H(t) = A(t)H_{init} + B(t)H_{fin} \qquad A(t)\downarrow \quad B(t)\uparrow$$

- **Quantum Adiabatic theorem**:
«if the evolution is slow enough, the quantum-mechanical system stays close the ground-state of the istantaneous Hamiltonian»



QUBO problem → $H(\vec{x}) = \sum_i a_i x_i + \sum_{i,j} b_{ij} x_i x_j \qquad x_i \in \{0,1\} \quad a_i, b_{ij} \in \mathbb{R}$

QUBO problems formulation

Quadratic Unconstrained Binary Optimization **(QUBO)** problem, minimizing:

$$H(\vec{x}) = \sum_{i} a_i x_i + \sum_{i,j} b_{ij} x_i x_j$$

x: vector of binary variables
a: linear term
b: quadraric term

Equivalent to the **Ising Model** (variables are “spin up” and “spin down” states):

$$\mathrm{E}_{ising}(s) = \sum_{i=1}^{N} h_i s_i + \sum_{i=1}^{N} \sum_{j=i+1}^{N} J_{i,j} s_i s_j$$

QUBO problem ⟷ Ising model

More resources here

D-Wave quantum annealer

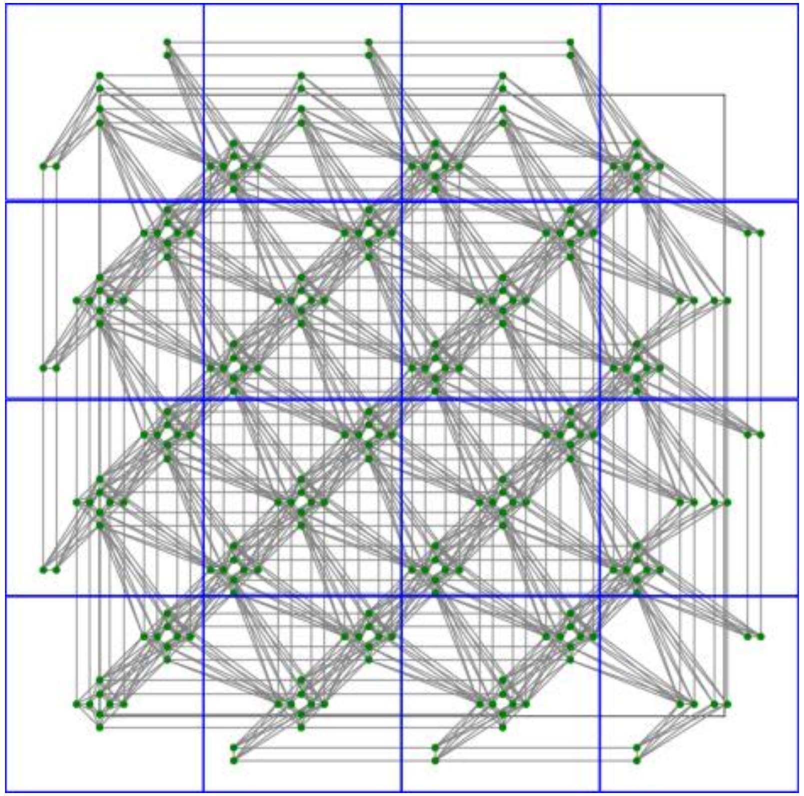
- The D-Wave company is the only commerical quantum annealing machines provider so far (1 min/month QPU access time for free)

- The D-Wave QPU is a **lattice of interconnected superconducting qubits** operating at around 15 mK and with a fixed limited topology

D-Wave Advantage is currently their best quantum annealer:

- 5000+ qubits
- 35000+ couplers
- *Pegasus* topology

Pegasus topology in D-Wave Advantage



D-Wave
The Quantum Computing Company™

D-Wave quantum annealer

Current limits:

- Too few qubits, but this number grows exponentially over the years
- Bad qubits quality and not optimal topology
- Quantum and hybrid solvers are unstable and results may oscillate for these cases

Usual standard workflow:

- Do large-scale studies using simulated annealing (slow)
- Test also the hybrid solver (medium)
- Do small-scale studies using quantum annealing (fast)
- With the increasing in the number of qubits this workflow will probably change

QUnfold - Mathematical formulation

Log-likelihood maximization unfolding: $\max_{\vec{x}} \left(\log \mathcal{L}(\vec{\mu}|\vec{d}) + \lambda \mathcal{S}(\vec{\mu}) \right)$

- $\mathcal{L}(\vec{\mu}|\vec{d})$: measured distribution
- $\lambda \mathcal{S}(\vec{\mu})$: regularization term

$$\vec{\mu} = R\vec{x}$$

- $\vec{\mu}$: "reco" distribution
- R: detector response matrix
- $\vec{x}$: unfolded distribution

Quadratic model minimization: $\min_{\vec{x}} \left(||R\vec{x} - \vec{d}||^2 + \lambda||G\vec{x}||^2 \right)$

$\lambda||G\vec{x}||^2$ → **Tikhonov regularization**: discrete 2nd order derivative (Laplacian operator G)

- Classical optimization problem, **nothing quantum yet**!
- $\vec{x}$ is the vector of integer numbers representing the unfolded histogram

QUnfold - Mathematical formulation

$$\min_{\vec{x}} \left(||R\vec{x} - \vec{d}||^2 + \lambda ||G\vec{x}||^2 \right)$$

$$\vec{a} = -2R^T\vec{d}$$
$$B = R^T R + \lambda G^T G$$

Minimization of:

$$H(\vec{x}) = \sum_i a_i x_i + \sum_{i,j} b_{ij} x_i x_j = \vec{a} \cdot \vec{x} + \vec{x}^T B \vec{x}$$

QUBO hamiltonian

QUnfold - Mathematical formulation

$$\vec{a} = -2R^T\vec{d}$$
$$B = R^T R + \lambda G^T G$$

$$H(\vec{x}) = \sum_i a_i x_i + \sum_{i,j} b_{ij} x_i x_j = \vec{a}\cdot\vec{x} + \vec{x}^T B\vec{x}$$

To get the QUBO model from this integer-variables quadratic problem, a “binarization” process based on the **logarithmic encoding** of the variables is needed

Their total number, which represents also the number of required logical qubits, scales as:

$$N_{\text{qubits}} \propto n_{\text{bins}} \cdot \log_2(n_{\text{entries}})$$

N_{bins} is the number of bins of the histogram
N_{entries} is the vector of the number of entries in each bin of the histogram

Example:
- Gaussian distribution
- 20 histogram bins
- 5M histogram entries

→ *N* qubits ≈ 350

QUnfold - Software package

- Implemented using *NumPy* and *D-Wave Ocean SDK* but fully compatible with ROOT
- Designed to address real-scale HEP applications
- Very simple and intuitive **Python** interface
- Public repository and documentation on GitHub
- Available on PyPI and easy to install via *pip*:

```
pip install Qunfold
```

Solver methods:

- Simulated annealing sampler (CPU only)
- Hybrid sampler (CPU + QPU)
- Quantum annealing sampler (QPU only)

QUnfold Public

A module to perform the statistical unfolding / deconvolution / matrix-inversion problem using quantum annealing with D-Wave quantum computer.

statistics python3 quantum-computing quantum-annealing

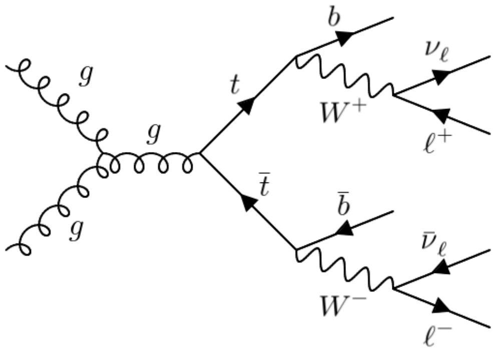
Python 18 2 MIT License 2 issues need help Updated 5 days ago

```
unfolder = QUnfoldQUBO(response, measured, lam=0.1)
unfolder.initialize_qubo_model()
unfolded_SA, error_SA = unfolder.solve_simulated_annealing(num_reads=10)
```

QUnfold - Tests on simulated data

Dataset

- $\boldsymbol{t\bar{t}}$ **process** in the *dileptonic channel* (2 leptons and at least 2 *b*-jets required in the final state)
- $\approx$ 2.5M **truth-level** events generated using the *MadGraph* generator (*truth* distribution)
- **Detector-level** data generated using the *Delphes* simulator (*measured* distribution)

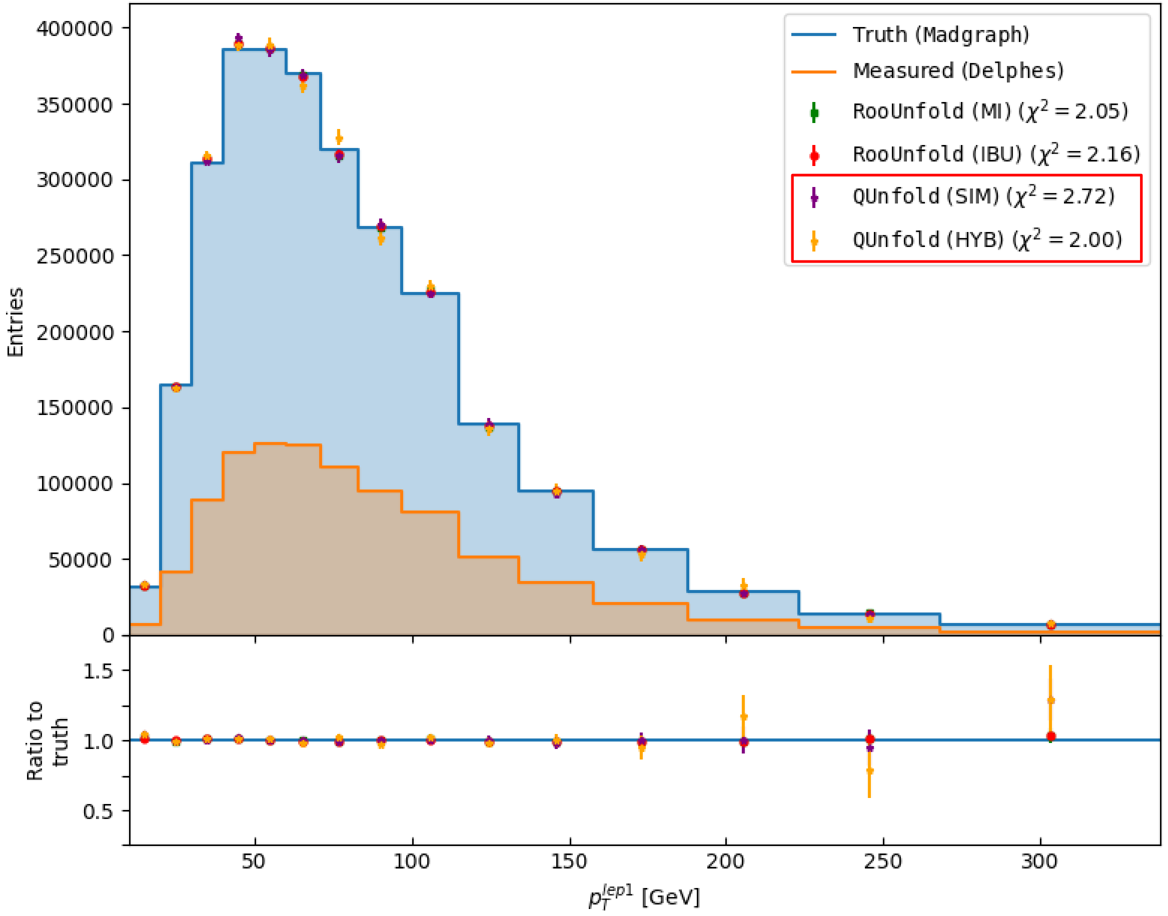


Technique

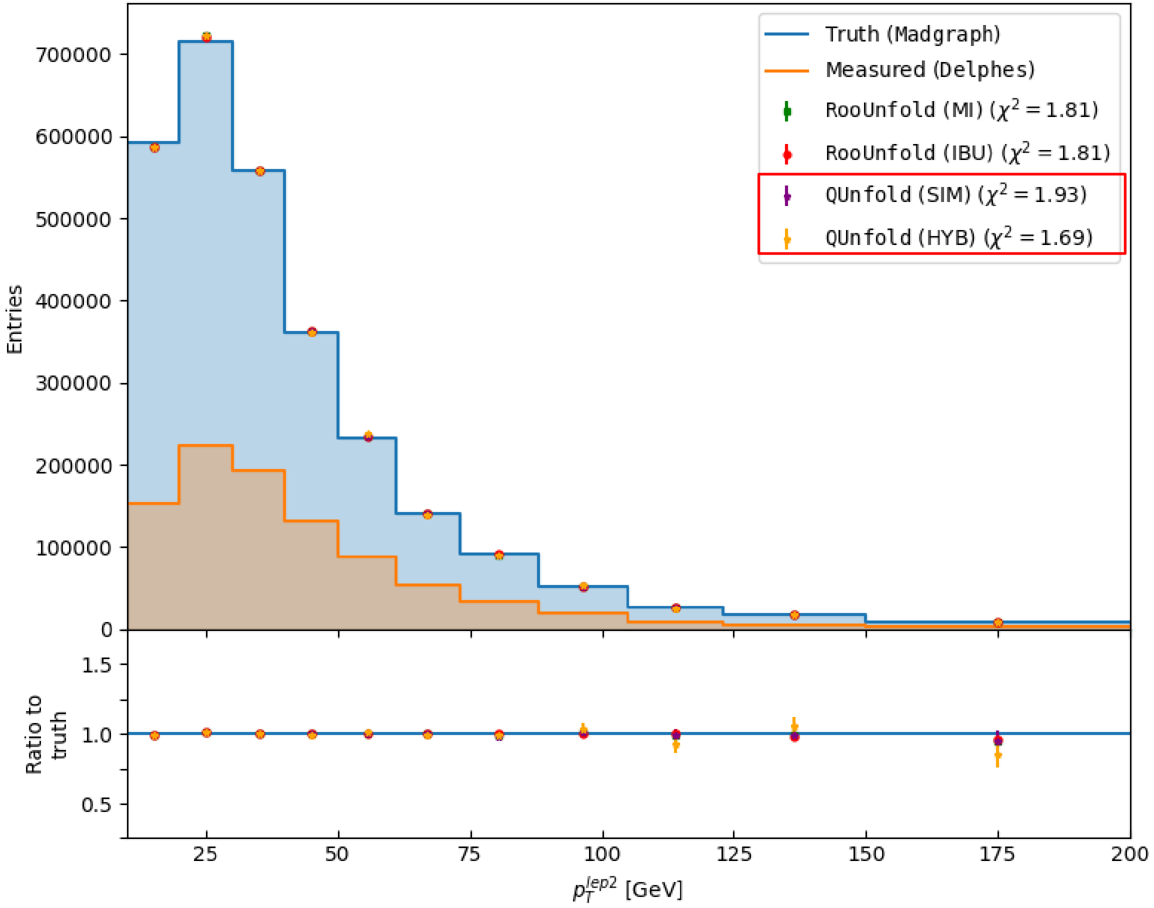
- Simulated annealing and hybrid solvers are used (quantum annealing solver is work in progress)
- Results are compared to the classical HEP unfolding methods *MI* and *IBU* (RooUnfold framework)
- **Toy Monte Carlo experiments** are run to compute the covariance matrix for evaluating the quality of the result (X^2 test) and estimating the statistical errors associated to the unfolding method

QUnfold - Preliminary results

Leading lepton p_T

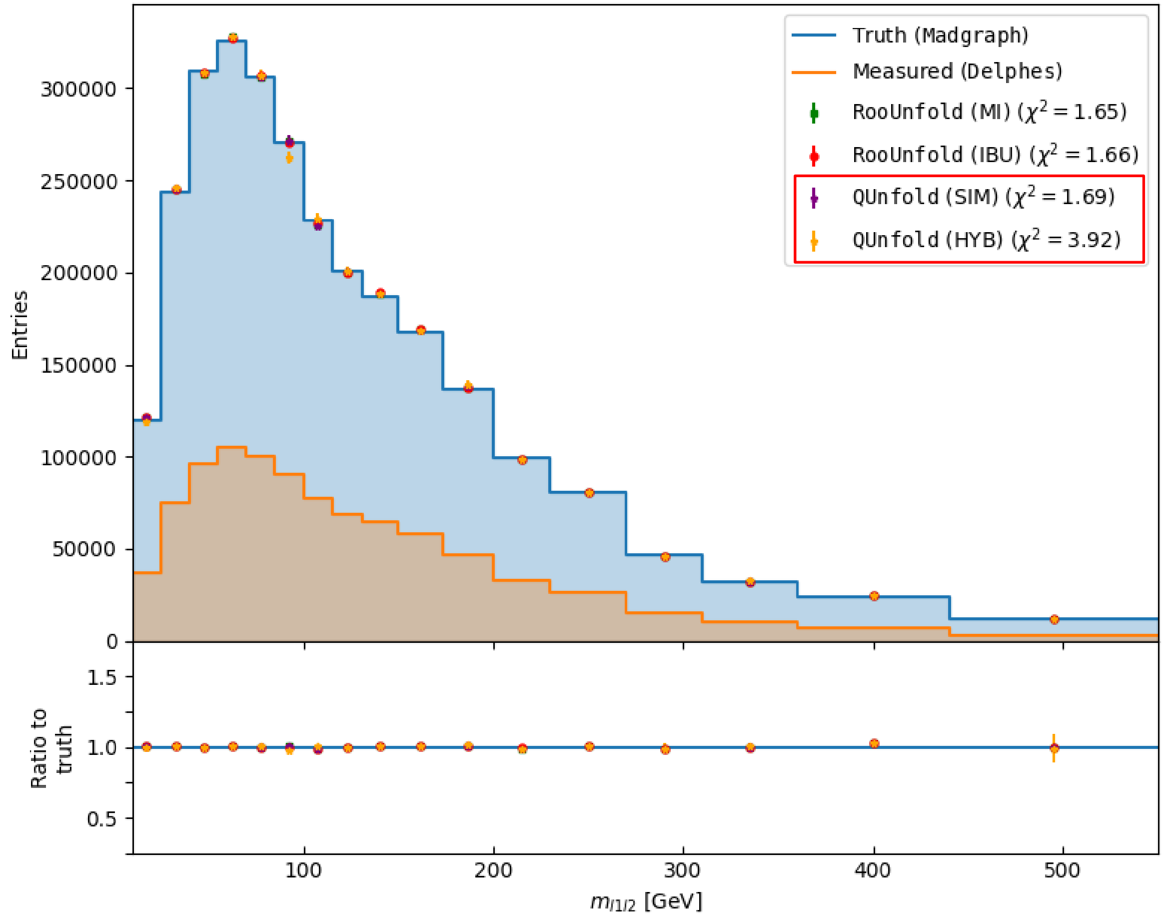


Subleading lepton p_T

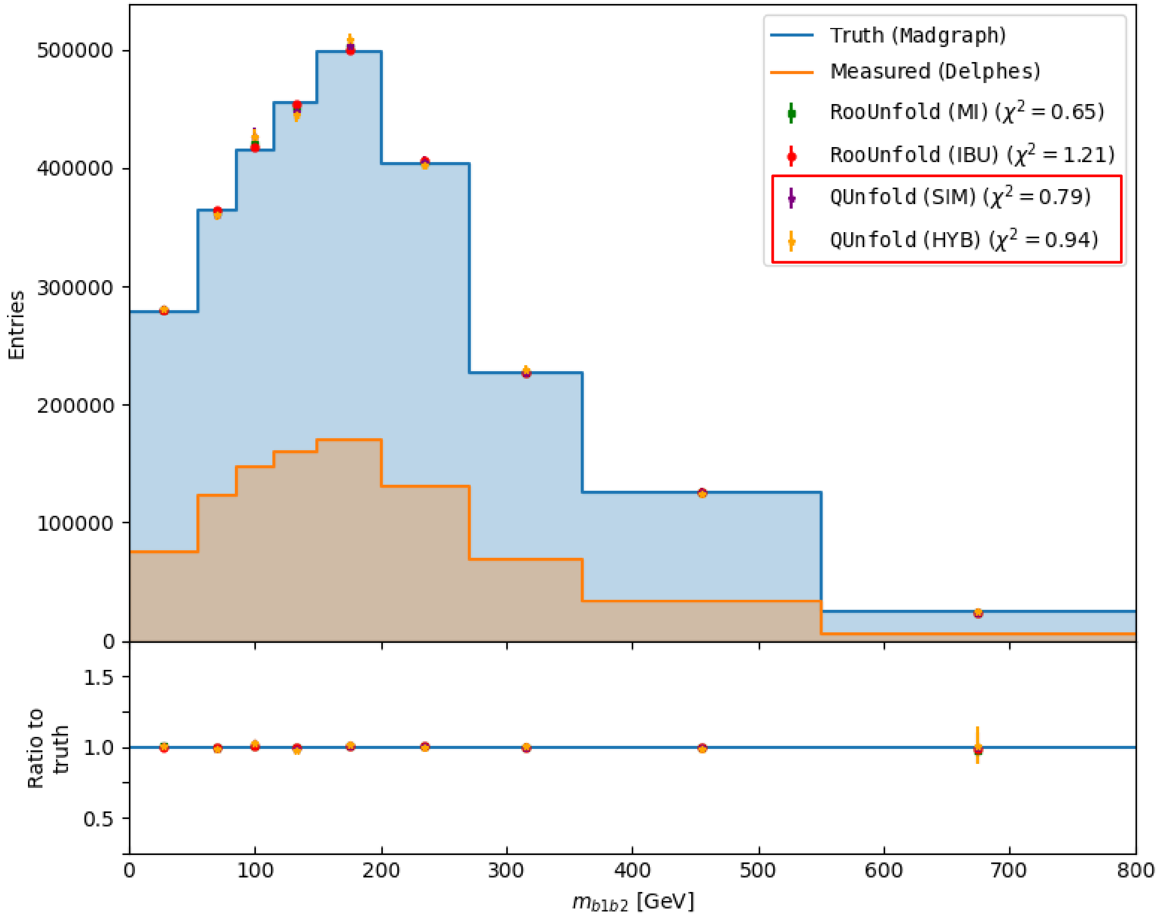


QUnfold - Preliminary results

Leptons invariant mass



***b*-jets invariant mass**



Measurement of differential cross-sections

Current standard method used in ATLAS and CMS:

- Particle/parton-level phase space
- Iterative Bayesian unfolding (IBU)
- In ATLAS Top TTbarUnfold is used (written in C++ by Marino)

$$\frac{d\sigma^{\text{fid}}}{dX^i} \equiv \frac{1}{\mathcal{L} \cdot \Delta X^i} \cdot \frac{1}{\epsilon^i} \cdot \sum_j M^{-1} \cdot f^j_{\text{acc}} \cdot \left(N^j_{\text{obs}} - N^j_{\text{bkg}}\right)$$

$$\frac{d\sigma^{\text{norm}}}{dX^i} = \frac{1}{\sigma^{\text{fid}}} \cdot \frac{d\sigma^{\text{fid}}}{dX^i}$$

Our idea:

- Working on a new general framework called PyXSec (open-source on GitHub), based on TTbarUnfold but written in Python
- Add full support to cross-sections measurements by using both **RooUnfold** classical methods and **QUnfold** quantum algorithms

Conclusion

Conclusions

- New unfolding approach based on the **QUBO formulation** of the problem and ***quantum annealing***
- Model implemented and tested in the **QUnfold** Python package, very easy to install and start using

Future steps

- Further optimize the algorithm (integer model *binarization*, QUBO matrix *pre-conditioning*, etc.)
- Perform more experiments on real quantum hardware (D-Wave resources by CINECA)
- Develop PyXSec: a new framework to measure differential cross-sections of HEP processes
- Design, implement and test a **gate-based approach** for the same problem (we started a collaboration with CERN QTI and IONQ)

Conclusion

Thanks for the attention!

https://github.com/JustWhit3/QUnfold

Backup

X^2 and errors computation

Covariance matrices and **errors** are computed through *MC pseudo-experiments*:

- A random *Poissonian smearing* is added to the measured distribution
- Unfolding is performed
- Procedure is repeated for *N* iterations (**toys**)
- Covariance matrix is computed considering the ensemble of the unfolding solution at each iteration:

$$c_{ij} = \langle (x_i - \langle x_i \rangle)(x_j - \langle x_j \rangle) \rangle$$

- Errors are computed as the square-root of the diagonal of the covariance matrix

$\mathbf{X}^2$ are computed with:

$$\mathrm{X}^2 = V^T \times Cov^{-1} \times V$$

Where V is the vector of *residuals*, defined as the difference between measurement and prediction

Preliminary results with Numpy