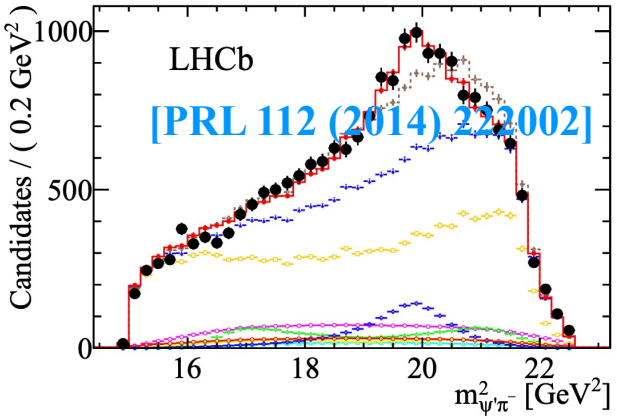
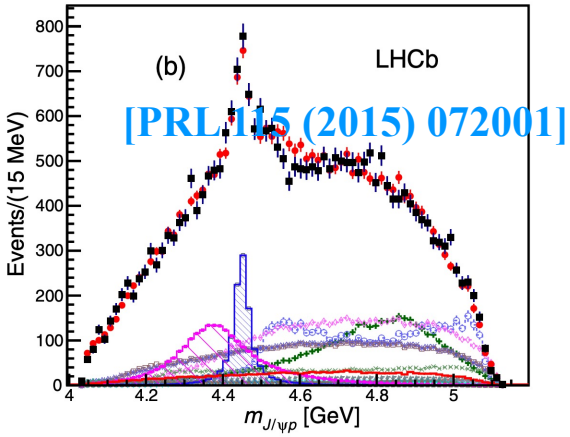
Amplitude analyses with Roofit fitter

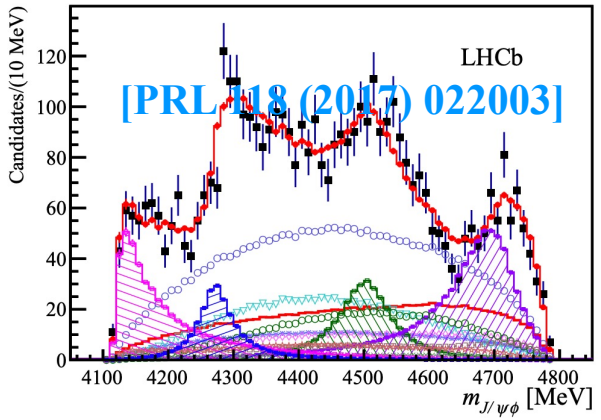
Many amplitude analyses in spectroscopy studies using RooFit fitter, e.g. :



Observation of $Z_c(4430)^-$



Observation of pentaquark



Observations of $X(4270)$ $X(4450)$ & $X(4700)$

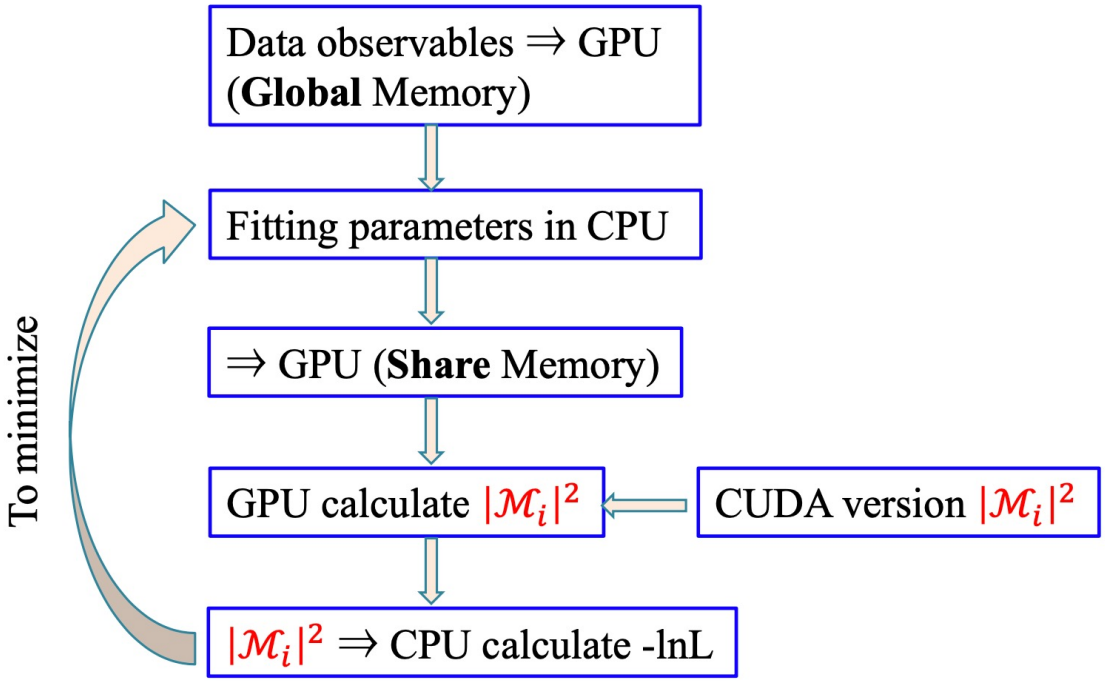
- RooFit widely used in HEP community
- CPU based codes could handle Run 1 statistic
  - ✓ e.g., ~30,000 signal in P_c study, multi-CPU fit
- Computation time related to many factors:
  - ✓ Data sample size
  - ✓ Initial values of fitting parameters
  - ✓ Number of floating parameters
  - ✓ ····

RooFit&CUDA based AmAn program

CPU based fit takes too much time with Run 2 data sample

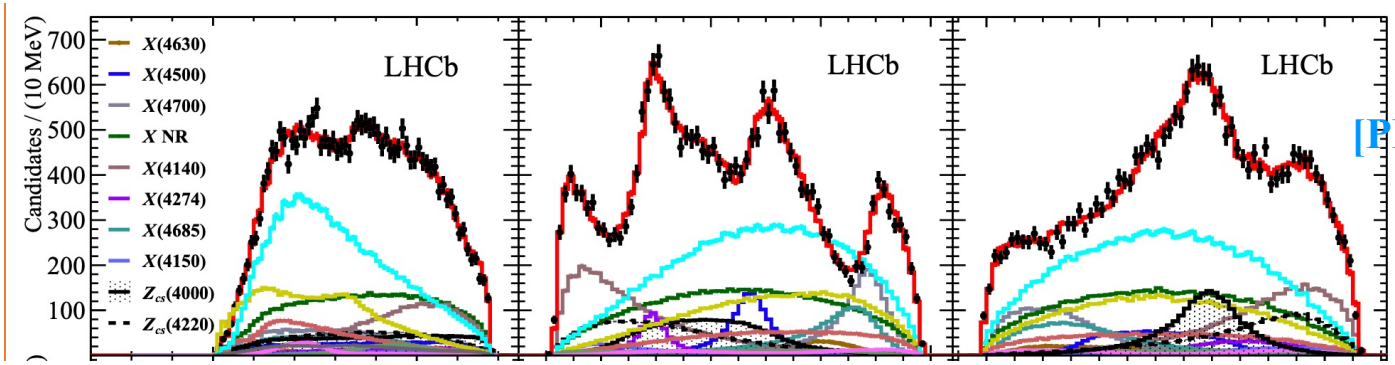
- Several hours ➔ acceptable
- Several days ➔ unacceptable

- Idea: Let GPU to calculate $|\mathcal{M}_i|^2$ for events of both data and MC
  - CUDA (GPU based C++) for computation of $|\mathcal{M}_i|^2$
  - CUDA memory transfers between CPU and CPU



Performance in $B \to J/\psi\phi K$ study

First publication using RooFit&CUDA fit technology at LHCb



[PRL 127 (2021) 082001]

Total signal number: ~24 k

Float parameter number: 144 (default model)

Computing time of once NLL: ~3ms

Computing time of one fit: few tens of minutes

- ✓ Closely related to the initial parameters
- ✓ Related to the float parameters number
- ✓ The complexity of different resonance line-shape
- ✓ ……

Theoretical Performa	TITAN V
Pixel Rate:	139.7 GPixel/s
Texture Rate:	465.6 GTexel/s
FP16 (half) performance:	29.80 TFLOPS (2:1)
FP32 (float) performance:	14.90 TFLOPS
FP64 (double) performance:	7.450 TFLOPS (1:2)

Our GPU card

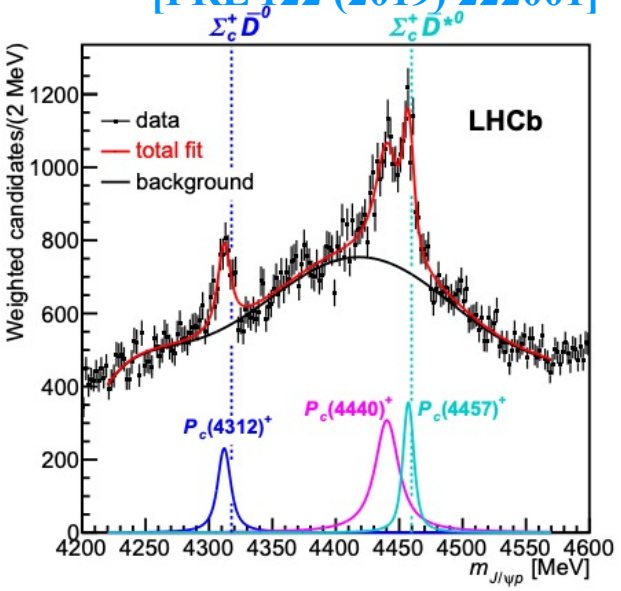
Tips in RooFit&CUDA AmAn program

- Reducing precision, double➔float
  - ✓ The final likelihood difference is negligible
  - ✓ GPU computing time reduced by half
  - ✓ Also dependent on the type of GPU card
- Malloc once and reuse memory
  - ✓ Reduce the data transfer between CPU & GPU
- Use less registers
  - ✓ Reducing the register pressure increases the number of working thread
- Split kernel function
  - ✓ Increase the GPU computing occupancy
- ……

Pentaquark in $\Lambda_b^0 \to J/\psi p K$

- Amplitude analysis with Run 1 & Run 2 ongoing
  (cannot show physical results)
- Total signal numbers : ~200k
- Longer computational time:
  - ✓ Larger statistic
  - ✓ More float parameters
  - ✓ Narrow structure, mass resolution considered

[PRL 122 (2019) 222001]



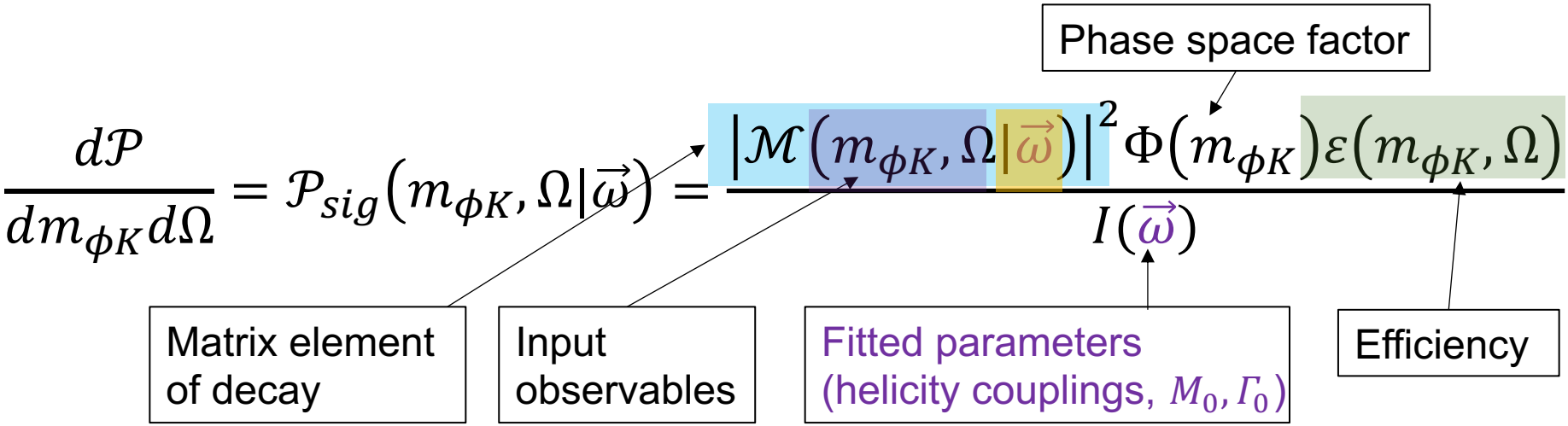
- Computing time of once NLL: ~0.17s
- Computing time of one complete fit: ~ 1-2 days
  - ✓ With good initial parameters, < 1 day
  - ✓ Many GPU cards used at same time
  - ✓ Not very convenient, computation is not the biggest obstacle for this study
  - ✓ Due to small fraction, not an ideal channel to determine J^P of pentaquark

Future

- RooFit & CUDA is not designed for widespread use, to optimize it more readable and extensible
- Computation timing smaller than several hours is acceptable in the view of physical analysis
- Many AmAn programs based on different frameworks are developed, e.g., RooFit&CUDA, GooFit, Tenserflow, etc., analysists prefer to choose the familiar one if their performances are similar
- More efficient fitter could boost the analysist work efficiency !

Backup

Signal PDF



$\vec{\omega}$: fitting parameters
Φ : phase-space factor = pq
ϵ : efficiency
$I(\vec{\omega})$: normalization integral

AmAn likelihood construction

Efficiency parameterization from MC sample

$$-\ln L(\vec{\omega}) = -\sum_i \ln\left[(1-\beta)\mathcal{P}_{sig}\left(m_{\phi K\, i}, \Omega_i\,|\vec{\omega}\right) + \beta\mathcal{P}_{bkg}\left(m_{\phi K\, i}, \Omega_i\right)\right]$$

$$= -\sum_i \ln\left[(1-\beta)\frac{\left|\mathcal{M}\left(m_{\phi K\, i}, \Omega_i\,|\vec{\omega}\right)\right|^2 \Phi(m_{\phi K\, i})\varepsilon(m_{\phi K\, i}, \Omega_i)}{I(\vec{\omega})} + \beta\frac{\mathcal{P}^u_{bkg}(m_{\phi K\, i}, \Omega_i)}{I_{bkg}}\right]$$

$$= -\sum_i \ln\left[\left|\mathcal{M}\left(m_{\phi K}, \Omega\,|\vec{\omega}\right)\right|^2 + \frac{\beta I(\vec{\omega})}{(1-\beta)I_{bkg}}\frac{\mathcal{P}^u_{bkg}(m_{\phi K\, i}, \Omega_i)}{\Phi(m_{\phi K\, i})\varepsilon(m_{\phi K\, i}, \Omega_i)}\right] + N\ln I(\vec{\omega}) + \text{constant}$$

Normalization factor

Background parameterization from sideband