University of Chinese Academy of Sciences

GENERAL PARTIAL WAVE ANALYSIS TOOL TF-PWA AND ITS APPLICATIONS

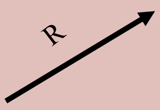
Yi Jiang (蒋艺)
University of Chinese Academy of Sciences

Homepage:
https://github.com/jiangyi15/tf-pwa
Or scan the QR

Abstract: Using simple configuration file, partial wave analysis (PWA) can be processed automatically and customable. Benefit from the powerful GPU calculation and Automatic Differentiation (AD) in TesorFlow2, the procedure is also fast and efficient. TF-PWA is our approach for general partial wave analysis tools. It have already used in real analysis.

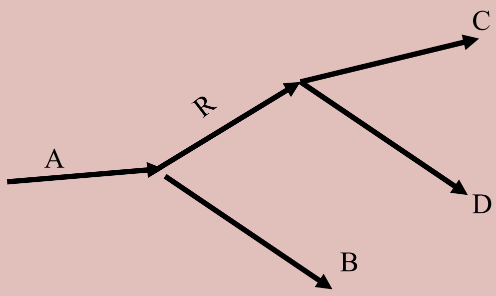
1. Configuration: YAML format, easy to understand and to modify.

Particle: Basic physic objects, with spin, parity, mass, width, and so on. Custom model is also allowed.

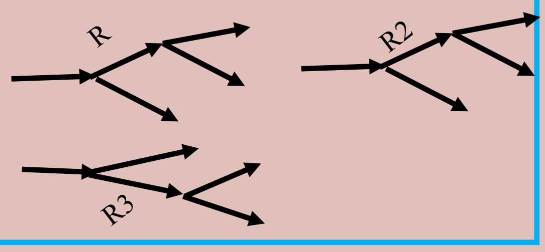



```yaml
rho:
  J: 1
  P: -1
  mass: 0.77511
  width: 0.1491
  model: GS_rho
  ...
```

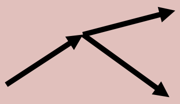
Decay Chain: A list of decay from initial particle to final particles.



Decay Group: A list of all possible Decay Chains



Decay: Connections in particles. Simple templates are provided. Also support custom model. Provide simplify replacement rules for complex decays.



```yaml
Lambda_c: [
  [Lambda, rho],
  [Sigma_starp, pi0 ],
  [Sigma_star0, pip ],
]
```

Data and MC: direct 4-momenta input. more information can be added such as weights and charge.

```yaml
data:
    dat_order: [p, pim, pip, pi0]
    data: [data.dat]
    bg: [sideband.dat]
    bg_weight: [0.13858078]
    phsp: [pshp.dat]
```

2. Rule based Amplitude Formula.

probability: $|\mathcal{A}|$^2

Decay Group: $\mathcal{A} = \tilde{A}_1 + \tilde{A}_2 + \cdots$

Decay Chain: $\tilde{A} = A_1 R A_2 \cdots$

Decay: Wigner D-matrix, $A = HD^{*J}(\phi, \theta, 0)$

Particle: Breit-Wigner: $R(m)$, user defined

$$\mathcal{A}^R_{\lambda_A,\lambda'_B,\lambda'_C,\lambda'_D} = \sum_{\lambda} H_{\lambda_R\lambda_B} D^{j_A\star}_{\lambda_A,\lambda_R-\lambda_B}(\varphi_1,\theta_1,0) R(M) H_{\lambda_C,\lambda_D} D^{j_R\star}_{\lambda_R,\lambda_C-\lambda_D}(\varphi_2,\theta_2,0)$$
$$D^{j_B\star}_{\lambda_B,\lambda_{B'}}(\alpha_B,\beta_B,\gamma_B) D^{j_C\star}_{\lambda_C,\lambda_{C'}}(\alpha_C,\beta_C,\gamma_C) D^{j_D\star}_{\lambda_D,\lambda_{D'}}(\alpha_D,\beta_D,\gamma_D)$$

$$\frac{d\sigma}{d\Phi} \propto \sum_{\lambda_A} \sum_{\lambda_B,\lambda_C,\lambda_D} \left| \sum_R \mathcal{A}^R_{\lambda_A,\lambda_B,\lambda_C,\lambda_D} \right|^2$$

4. Likelihood fit: (cFit as an example)

$$-\ln L = -\sum \ln\left[\frac{|\mathcal{A}|^2}{\mathcal{N}} + f_{bg}\right], \qquad \mathcal{N} = \frac{1}{N}\sum|\mathcal{A}|^2$$

$$\frac{\partial \ln L}{\partial \vartheta} = \frac{\partial}{\partial \vartheta}\sum \ln\left[\frac{|\mathcal{A}|^2}{N} + f_{bg}\right] + \frac{\partial}{\partial N}\sum \ln\left[\frac{|\mathcal{A}|^2}{\mathcal{N}} + f_{bg}\right]\frac{\partial \mathcal{N}}{\partial \vartheta},$$

$$\frac{\partial \mathcal{N}}{\partial \vartheta} = \frac{1}{N}\frac{\partial}{\partial \vartheta}\sum|\mathcal{A}|^2$$

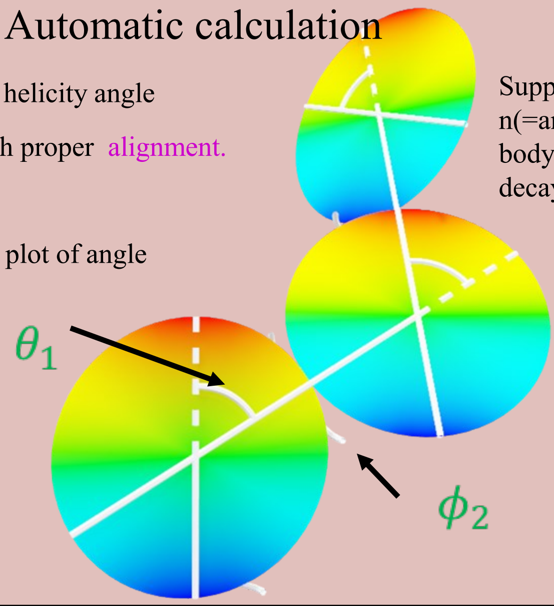
3. Automatic calculation

for helicity angle

with proper alignment.

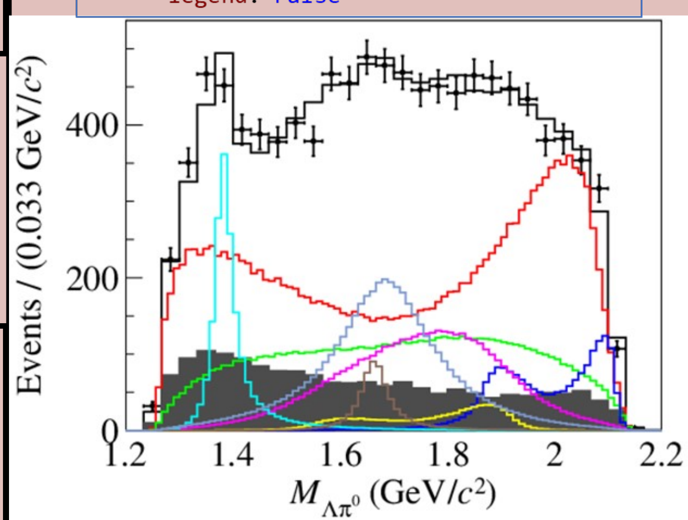
3D plot of angle

Support n(=any)-body decays.



Automatic Plot generated thought configuration

```yaml
plot:
  mass:
    Sigma_star0:  # name in Decay
      display: "$M_{\\Lambda\\pi^{0}}$"
      bins: 30
      range: [1.2, 2.2]
      legend: False
```



JHEP12(2022)033

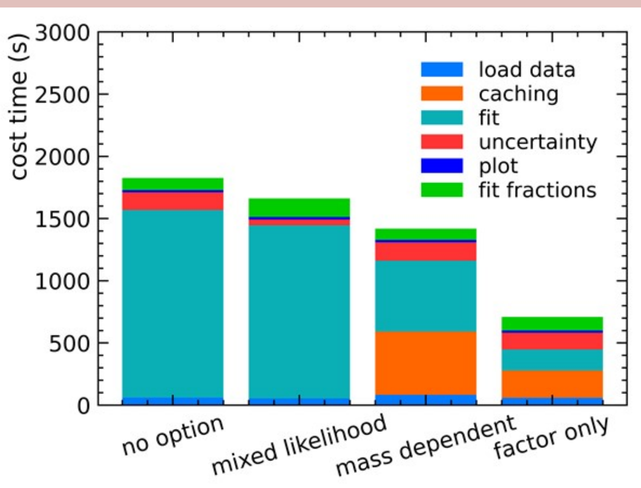
5. TensorFlow 2

GPU calculation: Vectorization and high parallelism.

AD: Fast gradients evaluations for fits and modified for supporting large size data (the red parts in 4 can be calculated in small batches)

6. Performances

Time for FULL procedure in different options. Achieve high performance for different use cases.



More for AD

User-friendly

error propagation

$$\sigma_f^2 = \frac{\partial f}{\partial \vartheta_i} V_{ij} \frac{\partial f}{\partial \vartheta_j}$$

```python
with config.params_trans() as pt:
    # g1 is fixed to 1
    g2_r = pt["Lmdc->piz.Sigma(1385)p_g_ls_1r"]
    g2_phi = pt["Lmdc->piz.Sigma(1385)p_g_ls_1i"]
    alpha = 2*g2_r*tf.cos(g2_phi) / (1 + g2_r * g2_r)
print(alpha, pt.get_error(alpha))
```

$$\alpha_{\Sigma(1385)\pi} = \frac{|H^{\Sigma(1385)}_{0,\frac{1}{2}}|^2 - |H^{\Sigma(1385)}_{0,\frac{-1}{2}}|^2}{|H^{\Sigma(1385)}_{0,\frac{1}{2}}|^2 + |H^{\Sigma(1385)}_{0,\frac{-1}{2}}|^2} = \frac{2\Re\left(g^{\Sigma(1385)}_{1,\frac{3}{2}} \cdot \bar{g}^{\Sigma(1385)}_{2,\frac{3}{2}}\right)}{|g^{\Sigma(1385)}_{1,\frac{3}{2}}|^2 + |g^{\Sigma(1385)}_{2,\frac{3}{2}}|^2}$$

7. Other functions:

(1) Constrains on parameters.

(2) Toy generation, with additional importance sampling.

(3) Bidirected transform between angle and momenta.

(4) Partial waves factorized extraction.

(5) Numerical method for lineshape calculation. (3)+(4)

And more.