# 课前准备

- Login the Siyuan-1 server:

```
ssh ${usr_name}@sylogin.hpc.sjtu.edu.cn
```

- Create your personal directory:

```
mkdir ${your_name}
```

- Copy the examples to your personal directory:

```
cd ${your_name}
cp -r /dssg/home/acct-phyww/phyww/qazhang/training_camp_2023/class2_zhang/example .
```

- Upload and download:

```
scp –r ${usr_name}@sylogin.hpc.sjtu.edu.cn:${data_path} ${local_address}
scp –r ${locat_data} ${usr_name}@sylogin.hpc.sjtu.edu.cn:${server_address}
```

- Python environments：

```
export PATH="/dssg/home/acct-phyww/phyww/qazhang/packages/anaconda3/bin:$PATH"
export PYTHONPATH=$PYTHONPATH:/dssg/home/acct-phyww/phyww/.local/lib/python3.7/site-packages
```

- Bash基本操作：

cd：跳转到某个目录

ls (ll)：列出当前目录中的所有文件

rm -rf：删除文件/目录

mkdir：新建目录

mv：移动文件/目录

cp -r：拷贝文件/目录

……

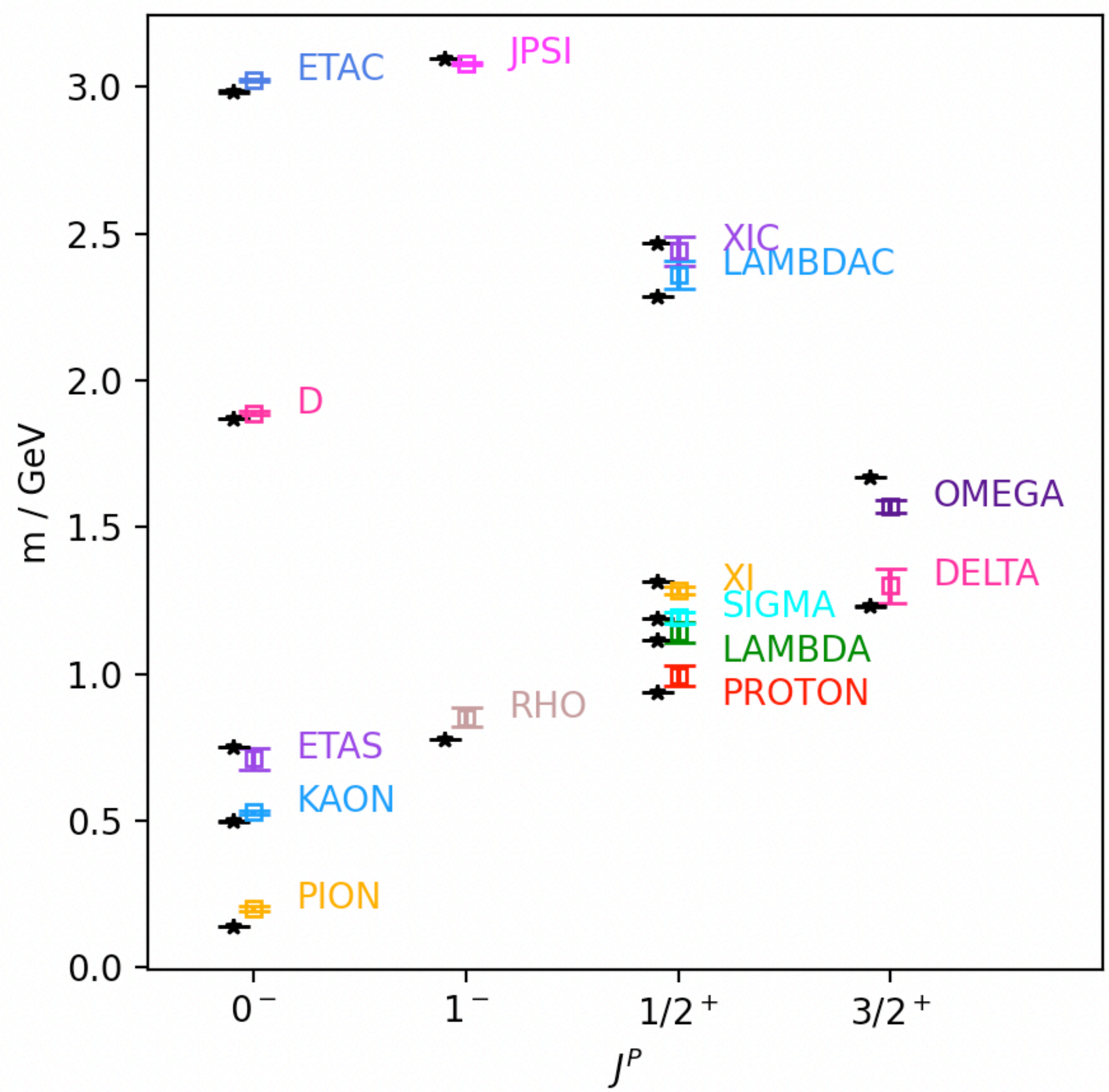
# 格点QCD中的强子谱学研究

# Hadron Spectroscopy from Lattice QCD

张其安 zhangqa@buaa.edu.cn

北京航空航天大学

Jul. 18, 2023




## Hadron spectrum: final results

- Generic hadron
- Ground state energy (effective mass)
- A coarse lattice
- Statistic and systematic uncertainties

NOT particularly accurate,

BUT CHEAP......

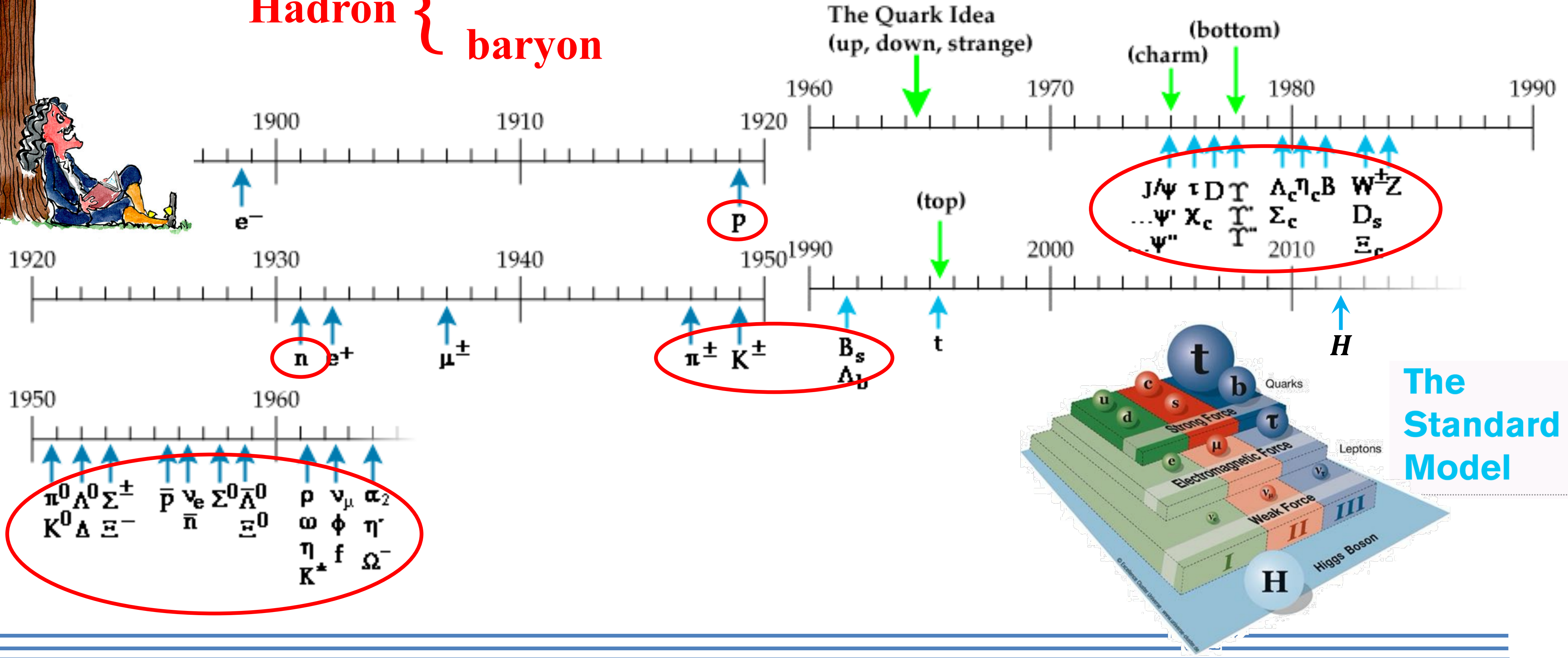
## OUTLOOK

- Introduction
- Meson interpolators and correlators
- Program implementation of meson 2-point correlators
- Numerical analysis and extracting pion effective mass
- Baryon 2-point correlators and proton effective mass
- Hadron spectrum
- Backup slides

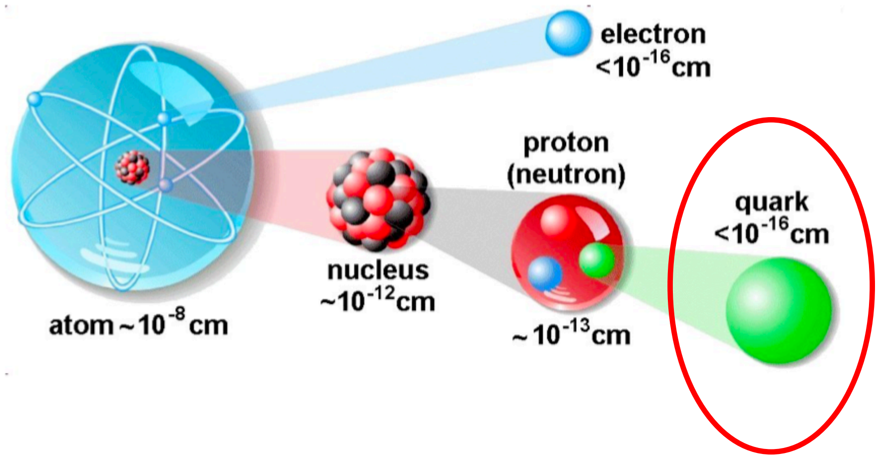
# Discovery of elementary particles

**Large mounts of, not so elementary……**

Hadron { meson, baryon




**The Standard Model**




creation operator |vacuum⟩

annihilation operator |vacuum⟩

⇒ states

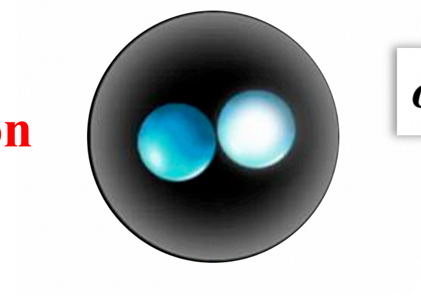
- **Quark fields: plane wave expansion**

$$\psi(x) = \int \frac{d^3\vec{p}}{(2\pi)^3}\frac{1}{2E_p}\sum_{\lambda=1,2}\left[b_\lambda(\vec{p})u^{(\lambda)}(\vec{p})e^{-i\vec{p}\cdot\vec{x}} + d^\dagger_\lambda(\vec{p})v^{(\lambda)}(\vec{p})e^{i\vec{p}\cdot\vec{x}}\right]$$

$$\bar{\psi}(x) = \int \frac{d^3\vec{p}}{(2\pi)^3}\frac{1}{2E_p}\sum_{\lambda=1,2}\left[b^\dagger_\lambda(\vec{p})\bar{u}^{(\lambda)}(\vec{p})e^{i\vec{p}\cdot\vec{x}} + d_\lambda(\vec{p})\bar{v}^{(\lambda)}(\vec{p})e^{-i\vec{p}\cdot\vec{x}}\right]$$

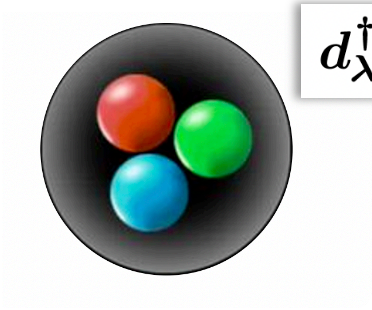
- **Hadron: bound state of fermion fields**

**meson**

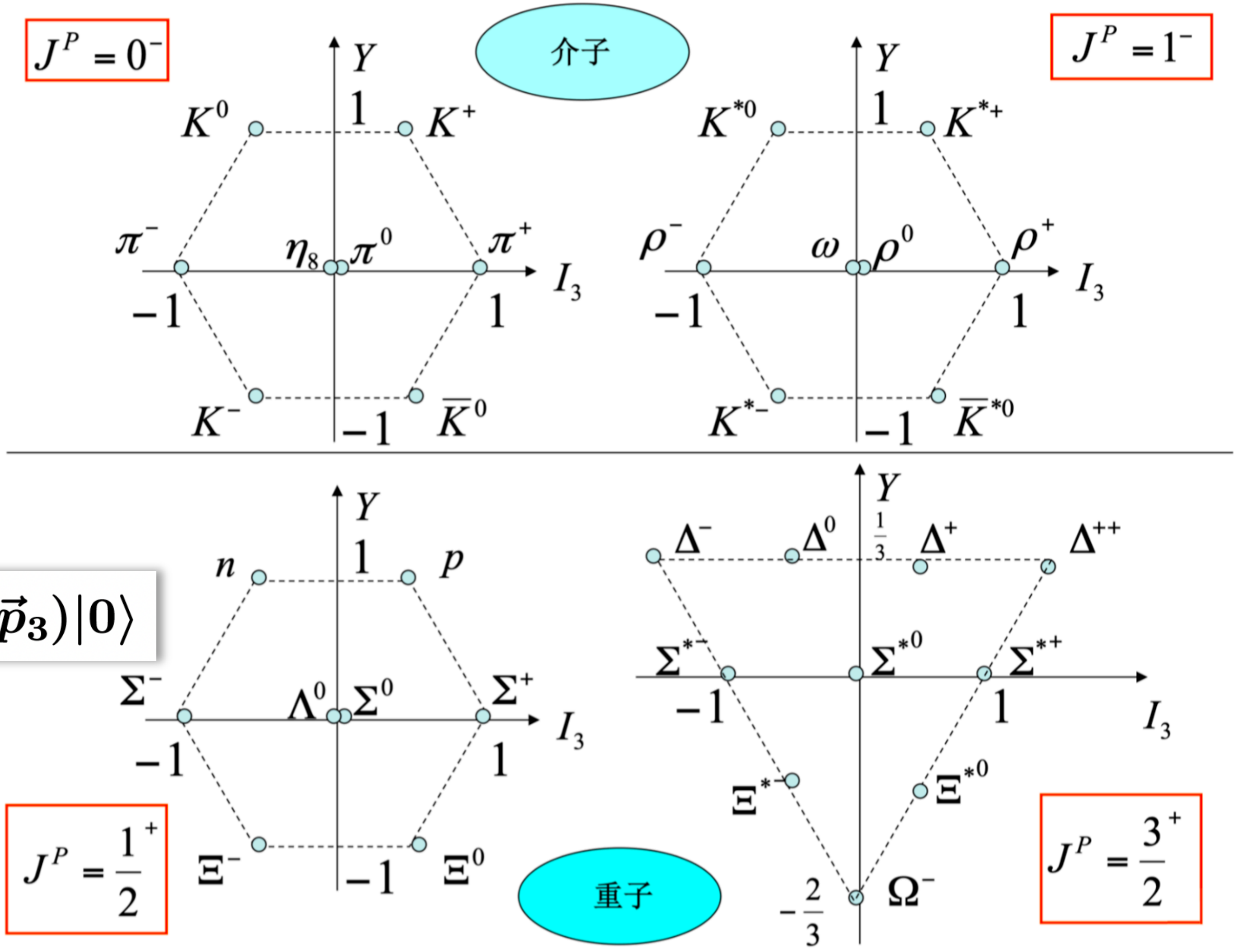


$$d^{\dagger}_{\lambda_1}(\vec{p}_1)b^{\dagger}_{\lambda_2}(\vec{p}_2)|0\rangle$$

**baryon**



$$d^{\dagger}_{\lambda_1}(\vec{p}_1)d^{\dagger}_{\lambda_2}(\vec{p}_2)d^{\dagger}_{\lambda_3}(\vec{p}_3)|0\rangle$$




constructed by quarks and gluons

gauge-invariant, color singlets

- **Meson interpolators:**

**Quantum number:**

isospin, charge } Quark constitution

spin, parity } Dirac structure

| State | $J^P$ | $\Gamma$ | Particles |
|---|---|---|---|
| Scalar | $0^+$ | $\mathcal{I},\ \gamma_4$ | $f_0,\ a_0,\ K_0^*,\ \cdots$ |
| Pseudoscalar | $0^-$ | $\gamma_5,\ \gamma_4\gamma_5$ | $\pi,\ \eta,\ K,\ \cdots$ |
| Vector | $1^-$ | $\gamma_i,\ \gamma_4\gamma_i$ | $\rho,\ \omega,\ K^*,\ \phi,\ \cdots$ |
| Axial vector | $1^+$ | $\gamma_i\gamma_5,\ \gamma_4\gamma_i\gamma_5$ | $a_1,\ f_1,\ \cdots$ |
| | | $\gamma_i\gamma_j,\ \gamma_i\gamma_j\gamma_5$ | $h_1,\ b_1,\ \cdots$ |

- **Meson interpolators**:

Quantum number:

isospin, charge } Quark constitution

spin, parity } Dirac structure

| State | $J^P$ | $\Gamma$ | Particles |
| --- | --- | --- | --- |
| Scalar | $0^+$ | $\mathcal{I},\ \gamma_4$ | $f_0,\ a_0,\ K_0^*,\ \cdots$ |
| Pseudoscalar | $0^-$ | $\gamma_5,\ \gamma_4\gamma_5$ | $\pi,\ \eta,\ K,\ \cdots$ |
| Vector | $1^-$ | $\gamma_i,\ \gamma_4\gamma_i$ | $\rho,\ \omega,\ K^*,\ \phi,\ \cdots$ |
| Axial vector | $1^+$ | $\gamma_i\gamma_5,\ \gamma_4\gamma_i\gamma_5$ | $a_1,\ f_1,\ \cdots$ |
| | | $\gamma_i\gamma_j,\ \gamma_i\gamma_j\gamma_5$ | $h_1,\ b_1,\ \cdots$ |

➢ **Pion**

$$O_{\pi^+}(x) = \bar{d}(x)\gamma_5 u(x) = \bar{\psi}_\alpha^{d,a}(x)(\gamma_5)_{\alpha\beta}\psi_\beta^{u,a}(x),$$
$$O_{\pi^-}(x) = \bar{u}(x)\gamma_5 d(x) = \bar{\psi}_\alpha^{u,a}(x)(\gamma_5)_{\alpha\beta}\psi_\beta^{d,a}(x),$$
$$O_{\pi^0}(x) = \frac{1}{\sqrt{2}}[\bar{u}(x)\gamma_5 u(x) - \bar{d}(x)\gamma_5 d(x)].$$

- **Meson interpolators**:

Quantum number:

isospin, charge } Quark constitution

spin, parity } Dirac structure

| State | $J^P$ | $\Gamma$ | Particles |
|---|---|---|---|
| Scalar | $0^+$ | $\mathcal{I},\ \gamma_4$ | $f_0,\ a_0,\ K_0^*,\ \cdots$ |
| Pseudoscalar | $0^-$ | $\gamma_5,\ \gamma_4\gamma_5$ | $\pi,\ \eta,\ K,\ \cdots$ |
| Vector | $1^-$ | $\gamma_i,\ \gamma_4\gamma_i$ | $\rho,\ \omega,\ K^*,\ \phi,\ \cdots$ |
| Axial vector | $1^+$ | $\gamma_i\gamma_5,\ \gamma_4\gamma_i\gamma_5$ | $a_1,\ f_1,\ \cdots$ |
| | | $\gamma_i\gamma_j,\ \gamma_i\gamma_j\gamma_5$ | $h_1,\ b_1,\ \cdots$ |

🤔

| Meson | $I$ | $I_3$ | $J^P$ |
|---|---|---|---|
| $\rho^{\pm,0}$ | 1 | $\pm1, 0$ | $1^-$ |
| $K^\pm$ | $1/2$ | $\pm1/2$ | $0^-$ |
| $D^\pm$ | $1/2$ | $\pm1/2$ | $0^-$ |
| $\eta_c$ | 0 | 0 | $0^-$ |
| $J/\psi$ | 0 | 0 | $1^-$ |

$\Rightarrow$ **Interpolators?**

- **Meson correlators in coordinate space**:

A general local meson interpolator has the form:

$$O_M(x) = \bar{\psi}^{f_1}(x)\Gamma\psi^{f_2}(x), \quad \bar{O}_M(x) = \bar{\psi}^{f_2}(x)\Gamma\psi^{f_1}(x)$$

The Euclidean 2-point correlation function (2pt):

$$\begin{aligned}
\langle O_M(y)\bar{O}_M(x)\rangle &= \langle \bar{\psi}^{f_1,a}_{\alpha}(y)\Gamma_{\alpha\beta}\psi^{f_2,a}_{\beta}(y)\bar{\psi}^{f_2,b}_{\alpha'}(x)\Gamma_{\alpha'\beta'}\psi^{f_1,b}_{\beta'}(x)\rangle \\
&= \Gamma_{\alpha\beta}\Gamma_{\alpha'\beta'}\langle \bar{\psi}^{f_1,a}_{\alpha}(y)\psi^{f_2,a}_{\beta}(y)\bar{\psi}^{f_2,b}_{\alpha'}(x)\psi^{f_1,b}_{\beta'}(x)\rangle \\
&= -\Gamma_{\alpha\beta}\Gamma_{\alpha'\beta'}\langle \psi^{f_1,b}_{\beta'}(x)\bar{\psi}^{f_1,a}_{\alpha}(y)\rangle\langle \psi^{f_2,a}_{\beta}(y)\bar{\psi}^{f_2,b}_{\alpha'}(x)\rangle \\
&= -\Gamma_{\alpha\beta}\Gamma_{\alpha'\beta'}G^{f_1,ba}_{\beta'\alpha}(x,y)G^{f_2,ab}_{\beta\alpha'}(y,x) \\
&= -\mathrm{tr}[\Gamma G^{f_2}(y,x)\Gamma G^{f_1}(x,y)]
\end{aligned}$$

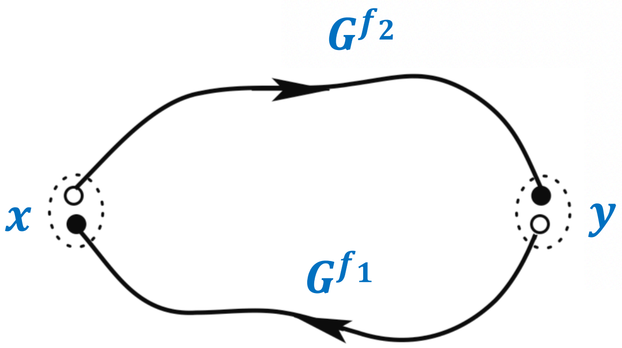
- **Meson correlators in coordinate space**:

A general local meson interpolator has the form:

$$O_M(x) = \bar{\psi}^{f_1}(x)\Gamma\psi^{f_2}(x), \quad \bar{O}_M(x) = \bar{\psi}^{f_2}(x)\Gamma\psi^{f_1}(x)$$

The Euclidean 2-point correlation function (2pt):

$$\begin{aligned}
\langle O_M(y)\bar{O}_M(x)\rangle &= \langle \bar{\psi}^{f_1,a}_{\alpha}(y)\Gamma_{\alpha\beta}\psi^{f_2,a}_{\beta}(y)\bar{\psi}^{f_2,b}_{\alpha'}(x)\Gamma_{\alpha'\beta'}\psi^{f_1,b}_{\beta'}(x)\rangle \\
&= \Gamma_{\alpha\beta}\Gamma_{\alpha'\beta'}\langle \bar{\psi}^{f_1,a}_{\alpha}(y)\psi^{f_2,a}_{\beta}(y)\bar{\psi}^{f_2,b}_{\alpha'}(x)\psi^{f_1,b}_{\beta'}(x)\rangle \\
&= -\Gamma_{\alpha\beta}\Gamma_{\alpha'\beta'}\langle \psi^{f_1,b}_{\beta'}(x)\bar{\psi}^{f_1,a}_{\alpha}(y)\rangle\langle \psi^{f_2,a}_{\beta}(y)\bar{\psi}^{f_2,b}_{\alpha'}(x)\rangle \\
&= -\Gamma_{\alpha\beta}\Gamma_{\alpha'\beta'}G^{f_1,ba}_{\beta'\alpha}(x,y)G^{f_2,ab}_{\beta\alpha'}(y,x) \\
&= -\mathrm{tr}[\Gamma G^{f_2}(y,x)\Gamma G^{f_1}(x,y)]
\end{aligned}$$




**Dirac structure**

**Propagator: $f, x \mapsto y$**

flavor, spinor, color, position, …

🤔 **2 propagators:** $f_1, y \mapsto x$ & $f_2, x \mapsto y$

- $\gamma_5$-hermiticity: (almost) all Dirac operators obey

$$\gamma_5 D \gamma_5 = D^\dagger$$

so we can reverse the propagator:

$$(\gamma_5)_{\alpha\alpha'} G^{f,ab}_{\alpha'\beta'}(x,y)(\gamma_5)_{\beta'\beta} = (G^{f,ba}_{\beta\alpha}(y,x))^*$$

⇒ **2 propagators have same start point (source) and end point (sink).**

- **2pt in momentum space:**

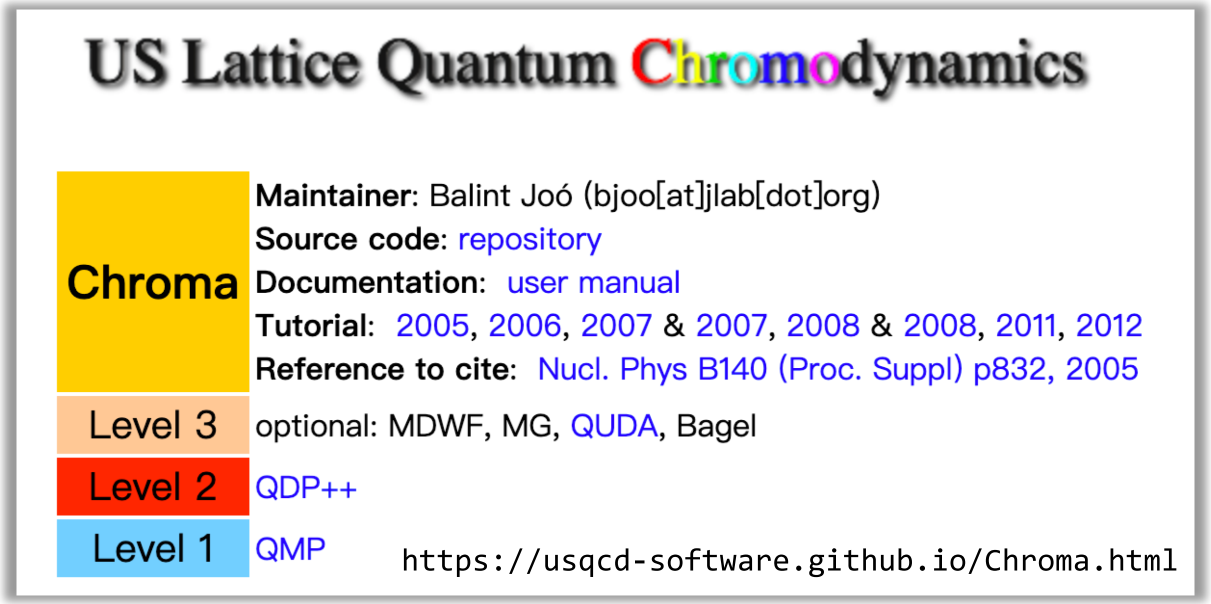
$$\begin{aligned}\Pi_2(t_0,t;\vec{p},\vec{x}_0) &= \sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)}\langle O_M(\vec{x},t)\bar{O}_M(\vec{x}_0,t_0)\rangle \\ &= -\sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)}\,\mathrm{tr}[\Gamma\, G^{f_2}(\vec{x},t;\vec{x}_0,t_0)\Gamma\,\gamma_5(G^{f_1}(\vec{x},t;\vec{x}_0,t_0))^*\gamma_5\,]\end{aligned}$$

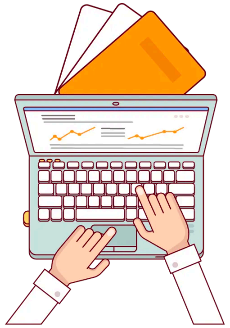
- **2pt in momentum space:**

$$
\begin{aligned}
\Pi_2(t_0, t; \vec{p}, \vec{x}_0) &= \sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)} \langle O_M(\vec{x}, t) \bar{O}_M(\vec{x}_0, t_0) \rangle \\
&= -\sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)} \mathrm{tr}[\Gamma G^{f_2}(\vec{x}, t; \vec{x}_0, t_0) \Gamma \gamma_5 (G^{f_1}(\vec{x}, t; \vec{x}_0, t_0))^* \gamma_5]
\end{aligned}
$$

**Translating to program language:**

1. **Read in propagators;**
2. **Do the trace;**
3. **Multiply the phase factor;**
4. **Sum over all space coordinate.**





# 请大家登陆思源一号服务器，一边听一边操作

```
ssh ${usr_name}@sylogin.hpc.sjtu.edu.cn
```

# 课前准备

- Login the Siyuan-1 server:

```
ssh ${usr_name}@sylogin.hpc.sjtu.edu.cn
```

- Create your personal directory:

```
mkdir ${your_name}
```

- Copy the examples to your personal directory:

```
cd ${your_name}
cp -r /dssg/home/acct-phyww/phyww/qazhang/training_camp_2023/class2_zhang/example .
```

- Upload and download:

```
scp -r ${usr_name}@sylogin.hpc.sjtu.edu.cn:${data_path} ${local_address}
scp -r ${locat_data} ${usr_name}@sylogin.hpc.sjtu.edu.cn:${server_address}
```

- Python environments：

```
export PATH="/dssg/home/acct-phyww/phyww/qazhang/packages/anaconda3/bin:$PATH"
export PYTHONPATH=$PYTHONPATH:/dssg/home/acct-phyww/phyww/.local/lib/python3.7/site-packages
```

- Bash基本操作：

cd: 跳转到某个目录
ls (ll): 列出当前目录中的所有文件
rm -rf: 删除文件/目录
mkdir: 新建目录
mv: 移动文件/目录
cp -r: 拷贝文件/目录
……

# A brief introduction to Chroma applications

**Measurements:**
(sequential) sources, smearings propagators spectroscopy, 3pt functions, hadron structure, wilson loops, eigenvalues

**Fermion Actions:**
wilson, tm, clover, 4D and 5D overlap, variety of coeffs, DWF, AsqTAD

**Monomials:**
two flavor 4D&5D,
one flavor rational 4D&5D,
Hasenbusch Term (4D),
LogDetEvenEven

**GaugeActions**
plaquette, rectangle, tree level and 1 loop LW, RG impr. plaq+rect, DBW2

**I/O Support:**
NERSC, CPPACS, UKY, SciDAC and ILDG Configurations

**Inverters:**
CG, CGNE, BiCGStab, Multi Shift CG, SUMR, GMRESR, MINRES

**Chronological Predictors:**
Zero Guess, Last Solution, Linear Extrapolation, Minimum Residual

**Eigensystems:**
Kalkreuter-Simma Ritz

**MD Integrators:**
Leapfrog, Omelyan (SW?) and Multi Time Scale versions of same

**Boundaries:**
(anti)periodic,Dirichlet, twisted, Schroedinger Functional

**Measurement (chroma)** | **HMC (hmc)** | **Pure Gauge Heatbath (purgaug)**

# A brief introduction to Chroma applications

**Measurement Applications**

- **Day2: Hadron spectrum**
- **Day4: Nucleon matrix elements**

**Inverter Applications**

- **Day3: Propagator generation**

**Gauge Generation Applications**

- **Day5: hybrid Monte Carlo**

**Measurements:**
(sequential) sources, smearings propagators spectroscopy, 3pt functions, hadron structure, wilson loops, eigenvalues

**Fermion Actions:**
wilson, tm, clover, 4D and 5D overlap, variety of coeffs, DWF, AsqTAD

**Monomials:**
two flavor 4D&5D, one flavor rational 4D&5D, Hasenbusch Term (4D), LogDetEvenEven

**I/O Support:**
NERSC, CPPACS, UKY, SciDAC and ILDG Configurations

**Inverters:**
CG, CGNE, BiCGStab, Multi Shift CG, SUMR, GMRESR, MINRES

**GaugeActions**
plaquette, rectangle, tree level and 1 loop LW, RG impr. plaq+rect, DBW2

**MD Integrators:**
Leapfrog, Omelyan (SW?) and Multi Time Scale versions of same

**Chronological Predictors:**
Zero Guess, Last Solution, Linear Extrapolation, Minimum Residual

**Eigensystems:**
Kalkreuter-Simma Ritz

**Boundaries:**
(anti)periodic,Dirichlet, twisted, Schroedinger Functional

Measurement (chroma) | HMC (hmc) | Pure Gauge Heatbath (purgaug)

- Chroma input file: XML driven programs

```
<?xml version="1.0"?>
<chroma>
  <Param>
    <InlineMeasurements>
      <elem>
          <Name>QIO_READ_NAMED_OBJECT</Name>
          <Frequency>1</Frequency>
          <NamedObject>
              <object_id>quark_propagator</object_id>
              <object_type>LatticePropagator</object_type>
          </NamedObject>
          <File>
              <file_name>$PATH_OF_PROPAGATOR</file_name>
              <parallel_io>true</parallel_io>
          </File>
      </elem>

      <elem>
        <Name>My_Measurements</Name>
        <Param>         </Param>
        <NamedObject>   </NamedObject>
      </elem>

    </InlineMeasurements>
      <nrow>24 24 24 72</nrow>
  </Param>

  <RNG>
    <Seed>
      <elem>11</elem>
      <elem>11</elem>
      <elem>11</elem>
      <elem>0</elem>
    </Seed>
  </RNG>

  <Cfg>
    <cfg_type>SCIDAC</cfg_type>
    <cfg_file>$PATH_OF_CONFIGURATION</cfg_file>
    <parallel_io>true</parallel_io>
  </Cfg>

</chroma>
```

# • Chroma input file: XML driven programs

```xml
<?xml version="1.0"?>
<chroma>
  <Param>
    <InlineMeasurements>
      <elem>
          <Name>QIO_READ_NAMED_OBJECT</Name>
          <Frequency>1</Frequency>
          <NamedObject>
              <object_id>quark_propagator</object_id>
              <object_type>LatticePropagator</object_type>
          </NamedObject>
          <File>
              <file_name>$PATH_OF_PROPAGATOR</file_name>
              <parallel_io>true</parallel_io>
          </File>
      </elem>

      <elem>
        <Name>My_Measurements</Name>
        <Param>         </Param>
        <NamedObject>   </NamedObject>
      </elem>

    </InlineMeasurements>
      <nrow>24 24 24 72</nrow>
  </Param>

  <RNG>
    <Seed>
      <elem>11</elem>
      <elem>11</elem>
      <elem>11</elem>
      <elem>0</elem>
    </Seed>
  </RNG>

  <Cfg>
    <cfg_type>SCIDAC</cfg_type>
    <cfg_file>$PATH_OF_CONFIGURATION</cfg_file>
    <parallel_io>true</parallel_io>
  </Cfg>

</chroma>
```

Array of Measurements (Tasks)

Task (array element)

Global Lattice Size

Random Number Generator

Input Configuration Details

# • Chroma input file: XML driven programs

```xml
<?xml version="1.0"?>
<chroma>
  <Param>
    <InlineMeasurements>
      <elem>
          <Name>QIO_READ_NAMED_OBJECT</Name>
          <Frequency>1</Frequency>
          <NamedObject>
              <object_id>quark_propagator</object_id>
              <object_type>LatticePropagator</object_type>
          </NamedObject>
          <File>
              <file_name>$PATH_OF_PROPAGATOR</file_name>
              <parallel_io>true</parallel_io>
          </File>
      </elem>

      <elem>
        <Name>My_Measurements</Name>
        <Param>        </Param>
        <NamedObject>    </NamedObject>
      </elem>

    </InlineMeasurements>
      <nrow>24 24 24 72</nrow>
  </Param>

  <RNG>
    <Seed>
      <elem>11</elem>
      <elem>11</elem>
      <elem>11</elem>
      <elem>0</elem>
    </Seed>
  </RNG>

  <Cfg>
    <cfg_type>SCIDAC</cfg_type>
    <cfg_file>$PATH_OF_CONFIGURATION</cfg_file>
    <parallel_io>true</parallel_io>
  </Cfg>

</chroma>
```

Array of Measurements (Tasks)

Task (array element)

YOUR WORK

Task Name

Task specific parameters

Named Objects (communicate between tasks)

Global Lattice Size

Random Number Generator

Input Configuration Details

- **Program implementation of 2pt**

```
<elem>
    <Name>QIO_READ_NAMED_OBJECT</Name>
    <Frequency>1</Frequency>
    <NamedObject>
        <object_id>quark_propagator</object_id>
        <object_type>LatticePropagator</object_type>
    </NamedObject>
    <File>
        <file_name>$PATH_OF_PROPAGATOR</file_name>
        <parallel_io>true</parallel_io>
    </File>
</elem>
```

```
<elem>
  <Name>My_Measurements</Name>
  <Param>          </Param>
  <NamedObject>    </NamedObject>
</elem>
```

1. **Read in propagators;**
2. **Do the trace;**
3. **Multiply the phase factor;**
4. **Sum over all space coordinate.**

- **Program implementation of 2pt**

```
<elem>
    <Name>QIO_READ_NAMED_OBJECT</Name>
    <Frequency>1</Frequency>
    <NamedObject>
        <object_id>quark_propagator</object_id>
        <object_type>LatticePropagator</object_type>
    </NamedObject>
    <File>
        <file_name>$PATH_OF_PROPAGATOR</file_name>
        <parallel_io>true</parallel_io>
    </File>
</elem>
```

1. **Read in propagators;**
2. **Do the trace;**
3. **Multiply the phase factor;**
4. **Sum over all space coordinate.**

```
<elem>
  <Name>My_Measurements</Name>
  <Param>          </Param>
  <NamedObject>    </NamedObject>
</elem>
```

**Parameters input**

```
> ls source_codes
```

(C++)

```
chroma      inline_myMeas.cc   io_general.cc         io_general_class.o   Makefile
chroma.cc   inline_myMeas.h    io_general_class.cc   io_general.h         make.sh
chroma.o    inline_myMeas.o    io_general_class.h    io_general.o
```

- **Program implementation of 2pt**

```
<elem>
    <Name>QIO_READ_NAMED_OBJECT</Name>
    <Frequency>1</Frequency>
    <NamedObject>
        <object_id>quark_propagator</object_id>
        <object_type>LatticePropagator</object_type>
    </NamedObject>
    <File>
        <file_name>$PATH_OF_PROPAGATOR</file_name>
        <parallel_io>true</parallel_io>
    </File>
</elem>
```

1. **Read in propagators;**
2. **Do the trace;**
3. **Multiply the phase factor;**
4. **Sum over all space coordinate.**

```
<elem>
  <Name>My_Measurements</Name>
  <Param>          </Param>
  <NamedObject>    </NamedObject>
</elem>
```

**Parameters input**

**An example of input**

```
<elem>
  <Name>My_Measurements</Name>
  <Param>
    <cfg_serial>4005</cfg_serial>
    <hadrons>
      <elem>PION</elem>
      <elem>KAON</elem>
      <elem>D</elem>
    </hadrons>
    <l_prop>L_quark_propagator</l_prop>
    <s_prop>S_quark_propagator</s_prop>
    <c_prop>C_quark_propagator</c_prop>
    <file_name>./data/2pt_PION-KAON-D_4005.dat.iog</file_name>
  </Param>
</elem>
```

```
> ls source_codes
```

(C++)

```
chroma        inline_myMeas.cc   io_general.cc         io_general_class.o   Makefile
chroma.cc     inline_myMeas.h    io_general_class.cc   io_general.h         make.sh
chroma.o      inline_myMeas.o    io_general_class.h    io_general.o
```

- **Parameters input for building 2pt**

$$\Pi_2(t_0, t; \vec{p}, \vec{x}_0) = \sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)} \langle O_M(\vec{x}, t) \bar{O}_M(\vec{x}_0, t_0) \rangle$$

$$= -\sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)} \, \mathrm{tr}[\Gamma G^{f_2}(\vec{x}, t; \vec{x}_0, t_0) \Gamma \gamma_5 (G^{f_1}(\vec{x}, t; \vec{x}_0, t_0))^* \gamma_5]$$

mass: static energy $\Rightarrow$ $\vec{p} = 0$

- **Parameters input for building 2pt**

$$\Pi_2(t_0, t; \vec{p}, \vec{x}_0) = \sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)} \langle O_M(\vec{x}, t) \bar{O}_M(\vec{x}_0, t_0) \rangle$$
$$= -\sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)} \mathrm{tr}[\Gamma G^{f_2}(\vec{x}, t; \vec{x}_0, t_0) \Gamma \gamma_5 (G^{f_1}(\vec{x}, t; \vec{x}_0, t_0))^* \gamma_5]$$

mass: static energy $\Rightarrow$ $\vec{p} = 0$

inline_myMeas.h

Declaration

- No. of configurations;
- Key of hadrons;
- Read in propagators;
- File name of output results.

```
struct InlineMyMeasIOGParams
{
  // Default constructor
  InlineMyMeasIOGParams();
  // Construct from XML
  InlineMyMeasIOGParams(XMLReader& xml_in, const std::string& path);
  // Write out the configuration of the parameters
  void write(XMLWriter& xml_out, const std::string& path);
  unsigned long frequency;

  // Holds the non-lattice parameters
  struct Param_t
  {
    int          cfg_serial;              /*!< The configuration serial number*/
    multi1d<std::string> hadrons;
    std::string l_prop;
    std::string file_name;

  } param;

  std::string xml_file; /*!< Alternate XML file pattern */
}; // end of struct InlineMyMeasIOGParams
```

<Param> xxx </Param>

inline_myMeas.cc

A long program……

## Step 1: Readin parameters

```
//! Reader for parameters
void read(                          XMLReader& xml,
                            const std::string& path,
         InlineMyMeasIOGParams::Param_t& param )
{
  XMLReader paramtop(xml, path);
  read(paramtop, "cfg_serial", param.cfg_serial);
  read(paramtop, "hadrons", param.hadrons);
  read(paramtop, "l_prop", param.l_prop);
  read(paramtop, "file_name", param.file_name);
}
```

inline_myMeas.cc

## Step 2: Grab propagators

```
//! Reader for parameters
void read(                          XMLReader& xml,
                           const std::string& path,
         InlineMyMeasIOGParams::Param_t& param )
{
  XMLReader paramtop(xml, path);
  read(paramtop, "cfg_serial", param.cfg_serial);
  read(paramtop, "hadrons", param.hadrons);
  read(paramtop, "l_prop", param.l_prop);
  read(paramtop, "file_name", param.file_name);
}
```

**std::string ==> A real propagator?**

## inline_myMeas.cc

### Step 2: Grab propagators

```
//! Reader for parameters
void read(                              XMLReader& xml,
                                 const std::string& path,
          InlineMyMeasIOGParams::Param_t& param )
{
  XMLReader paramtop(xml, path);
  read(paramtop, "cfg_serial", param.cfg_serial);
  read(paramtop, "hadrons", param.hadrons);
  read(paramtop, "l_prop", param.l_prop);
  read(paramtop, "file_name", param.file_name);
}
```

**std::string ==> A real propagator?**

- **Type structures on Chroma:** https://usqcd.jlab.org/usqcd-docs/qdp++/manual/index.html

| | *Lattice* | | *Color* | | *Spin* | | *Complexity* |
|---|---|---|---|---|---|---|---|
| Gauge fields : | Lattice | $\otimes$ | Matrix(Nc) | $\otimes$ | Scalar | $\otimes$ | Complex |
| Fermions : | Lattice | $\otimes$ | Vector(Nc) | $\otimes$ | Vector(Ns) | $\otimes$ | Complex |
| Scalars : | Scalar | $\otimes$ | Scalar | $\otimes$ | Scalar | $\otimes$ | Scalar |
| Propagators : | Lattice | $\otimes$ | Matrix(Nc) | $\otimes$ | Matrix(Ns) | $\otimes$ | Complex |
| Gamma : | Scalar | $\otimes$ | Scalar | $\otimes$ | Matrix(Ns) | $\otimes$ | Complex |

$\gamma_5 \Rightarrow$ Gamma(15)
$\gamma^x \Rightarrow$ Gamma(1)
$\gamma^y \Rightarrow$ Gamma(2)
$\gamma^z \Rightarrow$ Gamma(4)
$\gamma^t \Rightarrow$ Gamma(8)

```
typedef OScalar < PScalar     < PScalar<       RScalar <REAL>     >       > > Real;
typedef OLattice< PScalar     < PColorMatrix< RComplex<REAL>, Nc>       > > LatticeColorMatrix;
typedef OLattice< PSpinMatrix< PColorMatrix< RComplex<REAL>, Nc>, Ns> > LatticePropagator;
```

inline_myMeas.cc

## Step 2: Grab propagators

```
//! Reader for parameters
void read(                        XMLReader& xml,
                          const std::string& path,
         InlineMyMeasIOGParams::Param_t& param )
{
  XMLReader paramtop(xml, path);
  read(paramtop, "cfg_serial", param.cfg_serial);
  read(paramtop, "hadrons", param.hadrons);
  read(paramtop, "l_prop", param.l_prop);
  read(paramtop, "file_name", param.file_name);
}
```

std::string ==> A real propagator?

```
LatticePropagator L_prop;
QDPIO::cout << "Attempt to parse forward propagator" << std::endl;

try
{
  L_prop = TheNamedObjMap::Instance().getData<LatticePropagator>(params.param.l_prop);
}
catch (const std::string& e)
{
  QDPIO::cerr << InlineMyMeasIOGEnv::name
              << ": propagators: error message: " << e << std::endl;
  QDP_abort(1);
}
QDPIO::cout << "All propagators successfully parsed" << std::endl;
```

## inline_myMeas.cc

**Step 3:** Do the trace

Pion: pseudoscalar meson $\Gamma = \gamma_5$

$$\Pi_2(t_0, t; \vec{p}, \vec{x}_0) = \sum_{\vec{x}} \mathrm{tr}[\Gamma G^{f_2}(\vec{x}, t; \vec{x}_0, t_0) \Gamma \gamma_5 (G^{f_1}(\vec{x}, t; \vec{x}_0, t_0))^* \gamma_5]$$

**Lattice Wide Types: e.g. for real/complex number, Dirac/color matrix, or fermions, propagators……**

```
LatticeReal   Phases = 1.;

LatticeComplex corr = trace(adj(Gamma(15) * L_prop * Gamma(15))
                       * Gamma(15) * L_prop * Gamma(15));

multi1d<DComplex> hsum = sumMulti( Phases * corr, timeslice );
```

**multi1d<T>: 1D array of T (explicitly indexed)**

inline_myMeas.cc

Step 4: **Write data to disk**

```
multi1d<DComplex> hsum = sumMulti( Phases * corr, timeslice );
```

1-d complex data array

## inline_myMeas.cc

**Step 4:** Write data to disk

```
multi1d<DComplex> hsum = sumMulti( Phases * corr, timeslice );
```

1-d complex data array

- Dimensions of output data

Save file: *.iog

```
general_data_base res(params.param.file_name.c_str());
res.add_dimension(dim_conf, 1, &params.param.cfg_serial);
res.add_dimension(dim_operator, operator_no);
res.add_dimension(dim_t,tlen);
res.add_dimension(dim_complex, 2);
if(Layout::primaryNode()) res.initialize();
```

- **Save the data on each configuration as a single file;**
- **Calculate several hadrons at same time, index as "operator";**
- $\Pi_2(t)$ **with** $t \in [0, n_t)$;
- **Complex number: real & imag.**

**Size of output data:**

**1 × (No. of hadrons) ×** $n_t$ **× 2 float number**

- `1-dimensional array for output data`

| pion, t=0, real | pion, t=0, imag | pion, t=1, real | pion, t=1, imag | ······ | pion, t=$n_t$-1, real | pion, t=$n_t$-1, imag | proton, t=0, real | proton, t=0, imag | ······ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | ······ | $2n_t$-2 | $2n_t$-1 | $2n_t$ | $2n_t$+1 | ······ |

← **array elements**

← **array indexes**

- **1-dimensional array for output data**

| pion, t=0, real | pion, t=0, imag | pion, t=1, real | pion, t=1, imag | ...... | pion, t=$n_t$-1, real | pion, t=$n_t$-1, imag | proton, t=0, real | proton, t=0, imag | ...... |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | ...... | $2n_t$-2 | $2n_t$-1 | $2n_t$ | $2n_t$+1 | ...... |

← **array elements**

← **array indexes**

**Size of output data:** **$1 \times$ (No. of hadrons) $\times n_t \times 2$ float number**

```
multi1d<DComplex> hsum = sumMulti( Phases * corr, timeslice );

if(Layout::primaryNode())
for (int t=0; t < tlen; ++t)
{
  res.data[offset*tlen*2 + 2*t]  = hsum[t].elem().elem().elem().real();
  res.data[offset*tlen*2 + 2*t + 1] = hsum[t].elem().elem().elem().imag();
}
offset++;
```

**Offset of the indexes with various dimensions**

- **1-dimensional array for output data**

| pion, t=0, real | pion, t=0, imag | pion, t=1, real | pion, t=1, imag | …… | pion, t=$n_t$-1, real | pion, t=$n_t$-1, imag | kaon, t=0, real | kaon, t=0, imag | …… | ← **array elements** |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | …… | $2n_t$-2 | $2n_t$-1 | $2n_t$ | $2n_t$+1 | …… | ← **array indexes** |

**Size of output data:** **$1 \times$ (No. of hadrons) $\times n_t \times 2$ float number**

```
multi1d<DComplex> hsum = sumMulti( Phases * corr, timeslice );

if(Layout::primaryNode())
for (int t=0; t < tlen; ++t)
{
  res.data[offset*tlen*2 + 2*t]  = hsum[t].elem().elem().elem().real();
  res.data[offset*tlen*2 + 2*t + 1] = hsum[t].elem().elem().elem().imag();
}
offset++;
```

**Offset of the indexes with various dimensions**

**Done!** For more details, see the manual of Chroma and QDP++.

# Build your codes and run Chroma

- **Build your code:**
  - You need compile your source codes after update;
  - Run `sbatch sub.sh` to submit your compiling task to the compute node;
  - Use `squeue` to see the status of your submitted job;
  - After a few minutes, you will get the executable file `chroma`.

executable file    Your measurements

```
Makefile    inline_myMeas.cc  io_general.h        io_general_class.o    sub.sh
chroma      inline_myMeas.h   io_general.o        make.sh
chroma.cc   inline_myMeas.o   io_general_class.cc make_source_code.err
chroma.o    io_general.cc     io_general_class.h  make_source_code.out
```

logs and errors of your job

- **Submit your Chroma task:**

Typical command line in `sub.sh`:

- **Slurm Workload Manager**

```
#SBATCH --job-name=2pt_qaz
#SBATCH --partition=64c512g
#SBATCH --output=test_2pt.sh.out
#SBATCH --error=test_2pt.sh.err
#SBATCH --array=0
#SBATCH -N 1
#SBATCH --ntasks-per-node=64
#SBATCH --cpus-per-task=1
#SBATCH --time=00:120:00
```

Job Name

Output and error files

Nodes and CPUs on each node

**Totally 64 CPUs allocated for this job**

- **Submit your Chroma task:**

Typical command line in `sub.sh`:

- **Slurm Workload Manager**

```
#SBATCH --job-name=2pt_qaz
#SBATCH --partition=64c512g
#SBATCH --output=test_2pt.sh.out
#SBATCH --error=test_2pt.sh.err
#SBATCH --array=0
#SBATCH -N 1
#SBATCH --ntasks-per-node=64
#SBATCH --cpus-per-task=1
#SBATCH --time=00:120:00
#SBATCH --exclusive
```

Job Name

Output and error files

Nodes and CPUs on each node

**Totally 64 CPUs allocated for this job**

- **Perl script to generate the input XML file**

```
./2pt.pl  ${conf} $prefix > ./ini_out_file/ini_${conf}.xml
```

**Totally 50 configurations**

- **Submit your Chroma task:**

Typical command line in `sub.sh`:

```
mpirun -mca btl_tcp_if_include ib0 -n 64
    ../source_codes/chroma
    -i ./ini_out_file/ini_${conf}.xml
    -o ./ini_out_file/out_${conf}.xml
    > ./log_file/log_${conf}   2>&1
```

**64 CPUs task**

**Executable Chroma file you compiled**

**Input parameter file;**

**Output XML file;**

**Log file.**

- Run `sbatch sub.sh` to submit your jobs to the compute node;
- Use `squeue` to see the status of your submitted job;
- When you see these at the end of `log_file/log_${conf}`, your jobs were successfully run:

```
CHROMA: total time = 1.429768 secs
CHROMA: ran successfully
```

# Numerical results of pion 2pt

- **Read the iog data:**

```
res.add_dimension(dim_conf, 1, &params.param.cfg_serial);
res.add_dimension(dim_operator, operator_no);
res.add_dimension(dim_t,tlen);
res.add_dimension(dim_complex, 2);
```

| cfg | hadrons | t | Re | Im |
|---|---|---|---|---|
| 10000 | 0 | 0 | 379116.4304037094 | -2.673714808221206e-06 |
| 10000 | 0 | 1 | 286375.731112957 | 6.193346642646702e-06 |
| 10000 | 0 | 2 | 228801.41831445694 | -2.767021394234348e-07 |
| 10000 | 0 | 3 | 184919.00905895233 | -2.0718254489793253e-06 |
| 10000 | 0 | 4 | 152465.3666498661 | -2.9396259080272102e-06 |
| 10000 | 0 | 5 | 134413.50410485268 | -1.1056700648381934e-06 |
| 10000 | 0 | 6 | 118356.19234442711 | 1.913735434566324e-06 |
| 10000 | 0 | 7 | 97286.86232805252 | -6.839579205220048e-07 |
| 10000 | 0 | 8 | 76100.68012273312 | 1.614602146204902e-06 |
| 10000 | 0 | 9 | 59145.15935754776 | -1.109869838367139e-06 |
| 10000 | 0 | 10 | 46103.60806834698 | -2.334704649609165e-07 |
| 10000 | 0 | 11 | 37906.13109225035 | -4.332245748805974e-07 |
| 10000 | 0 | 12 | 32229.44473797083 | -2.415140017975048e-07 |
| 10000 | 0 | 13 | 27475.266874611378 | -1.272796494866668e-07 |
| 10000 | 0 | 14 | 22694.08443546295 | 3.874264117298 4094e-07 |
| 10000 | 0 | 15 | 17927.174669593573 | 1.401173795878563e-07 |
| 10000 | 0 | 16 | 14262.828596234322 | 4.038727519084517e-08 |
| 10000 | 0 | 17 | 11772.581270948052 | 2.261288581384413e-08 |
| 10000 | 0 | 18 | 9570.323621958494 | -2.638852990011209e-07 |
| 10000 | 0 | 19 | 8038.1801244318485 | 1.0683004819656006e-07 |
| 10000 | 0 | 20 | 7157.956188388169 | -2.533002501636794e-07 |

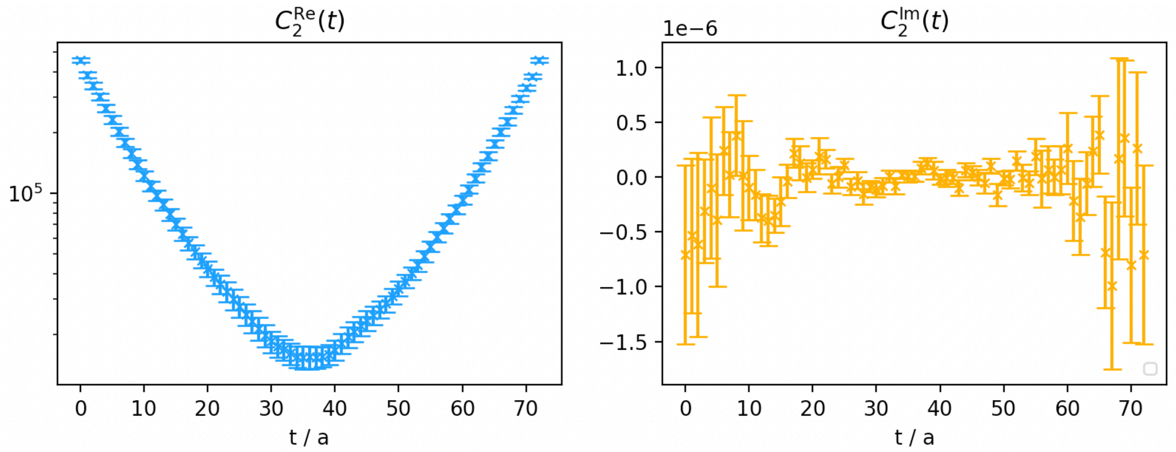
# Numerical results of pion 2pt

- **Read the iog data:**

| cfg | hadrons | t | Re | Im |
|---|---|---|---|---|
| 10000 | 0 | 0 | 379116.4304037094 | -2.673714808221206e-06 |
| 10000 | 0 | 1 | 286375.731112957 | 6.193346642646702e-06 |
| 10000 | 0 | 2 | 228801.41831445694 | -2.767021394234348e-07 |
| 10000 | 0 | 3 | 184919.00905895233 | -2.0718254489793253e-06 |
| 10000 | 0 | 4 | 152465.3666498661 | -2.9396259080272102e-06 |
| 10000 | 0 | 5 | 134413.50410485268 | -1.1056700648381934e-06 |
| 10000 | 0 | 6 | 118356.19234442711 | 1.9137354345666324e-06 |
| 10000 | 0 | 7 | 97286.86232805252 | -6.839579205220048e-07 |
| 10000 | 0 | 8 | 76100.68012273312 | 1.614602146204902e-06 |
| 10000 | 0 | 9 | 59145.15935754776 | -1.109869838367139e-06 |
| 10000 | 0 | 10 | 46103.60806834698 | -2.334704649609165e-07 |
| 10000 | 0 | 11 | 37906.13109225035 | -4.332245748805974e-07 |
| 10000 | 0 | 12 | 32229.44473797083 | -2.415140017975048e-07 |
| 10000 | 0 | 13 | 27475.266874611378 | -1.272796494866668e-07 |
| 10000 | 0 | 14 | 22694.08443546295 | 3.8742641172984094e-07 |
| 10000 | 0 | 15 | 17927.174669593573 | 1.401173795878563e-07 |
| 10000 | 0 | 16 | 14262.828596234322 | 4.038727519084517e-08 |
| 10000 | 0 | 17 | 11772.581270948052 | 2.261288581384413e-08 |
| 10000 | 0 | 18 | 9570.323621958494 | -2.638852990011209e-07 |
| 10000 | 0 | 19 | 8038.1801244318485 | 1.0683004819656006e-07 |
| 10000 | 0 | 20 | 7157.956188388169 | -2.533002501636794e-07 |

```
res.add_dimension(dim_conf, 1, &params.param.cfg_serial);
res.add_dimension(dim_operator, operator_no);
res.add_dimension(dim_t,tlen);
res.add_dimension(dim_complex, 2);
```

**Illustration of 50 cfgs results:**




- How to explain the exponential behavior of the real part?
- Why the imaginary part no signal?

# Analysis the 2pt data: A brief reviewing of the first day's talk

- Revisiting the theoretical form of 2pt:

$$\Pi_2(t_0, t; \vec{p}, \vec{x}_0) \; = \; a^6 \sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)} \langle O_M(\vec{x}, t) O_M^\dagger(\vec{x}_0, t_0)\rangle$$

inserting a complete set of intermediate states (completeness relation for the 1-particle states):

$$\mathcal{I} = \int \frac{d^3\vec{k}}{(2\pi)^3} |\vec{k}\rangle \frac{1}{2E_{\vec{k}}} \langle \vec{k}| \quad \Rightarrow \quad \mathcal{I} = \frac{1}{V} \sum_n |\vec{k}_n\rangle \frac{1}{2E_{\vec{k}_n}} \langle \vec{k}_n|$$

2pt becomes:

$$\Pi_2(t_0, t; \vec{p}, \vec{x}_0) \; = \; a^6 \sum_n \sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)} \langle 0|O_M(\vec{x}, t)|\vec{k}_n\rangle \frac{1}{2VE_{\vec{k}_n}} \langle \vec{k}_n|O_M^\dagger(\vec{x}_0, t_0)|0\rangle$$

# Analysis the 2pt data: A brief reviewing of the first day's talk

- Revisiting the theoretical form of 2pt:

$$\Pi_2(t_0,t;\vec{p},\vec{x}_0) \;=\; a^6\sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)}\langle O_M(\vec{x},t)O_M^\dagger(\vec{x}_0,t_0)\rangle$$

inserting a complete set of intermediate states (completeness relation for the 1-particle states):

$$\mathcal{I}=\int\frac{d^3\vec{k}}{(2\pi)^3}|\vec{k}\rangle\frac{1}{2E_{\vec{k}}}\langle\vec{k}| \quad\Rightarrow\quad \mathcal{I}=\frac{1}{V}\sum_n|\vec{k}_n\rangle\frac{1}{2E_{\vec{k}_n}}\langle\vec{k}_n|$$

2pt becomes:

$$\Pi_2(t_0,t;\vec{p},\vec{x}_0) \;=\; a^6\sum_n\sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)}\langle 0|O_M(\vec{x},t)|\vec{k}_n\rangle\frac{1}{2VE_{\vec{k}_n}}\langle\vec{k}_n|O_M^\dagger(\vec{x}_0,t_0)|0\rangle$$

$$\phi(x)=e^{i\hat{p}\cdot x}\phi(0)e^{-i\hat{p}\cdot x} \quad\Rightarrow\quad \langle 0|e^{\hat{E}(t-t_0)-i\hat{p}\cdot(\vec{x}-\vec{x}_0)}O_M(\vec{x}_0,t_0)e^{-\hat{E}(t-t_0)+i\hat{p}\cdot(\vec{x}-\vec{x}_0)}|\vec{k}_n\rangle$$

Minkovski, continuous space-time

Euclidean, discrete space-time

- The Euclidean time dependence of 2pt:

$$
\begin{aligned}
\Pi_2(t_0, t; \vec{p}, \vec{x}_0) &= a^6 \sum_n \sum_{\vec{x}} e^{i(\vec{k}_n - \vec{p})\cdot(\vec{x}-\vec{x}_0)} \langle 0|O_M(\vec{x}_0, t_0)|\vec{k}_n\rangle \frac{e^{-E_{\vec{k}_n}(t-t_0)}}{2VE_{\vec{k}_n}} \langle \vec{k}_n|O_M^\dagger(\vec{x}_0, t_0)|0\rangle \\
&= a^3 \sum_n |\langle 0|O_M(\vec{x}_0, t_0)|\vec{k}_n\rangle|^2 \frac{e^{-E_{\vec{k}_n}(t-t_0)}}{2E_{\vec{k}_n}} \delta^{(3)}_{\vec{k}_n, \vec{p}}\, e^{-i(\vec{k}_n - \vec{p})\cdot\vec{x}_0} \\
&= a^3 \sum_n \frac{|\langle 0|O_M(\vec{x}_0, t_0)|\vec{p}_n\rangle|^2}{2E_{\vec{p}_n}} e^{-E_{\vec{p}_n}(t-t_0)} \\
(\text{suppose } t_0 = 0) &= \sum_n A_n e^{-E_n t}
\end{aligned}
$$

- The Euclidean time dependence of 2pt:

$$\begin{aligned}
\Pi_2(t_0, t; \vec{p}, \vec{x}_0) &= a^6 \sum_n \sum_{\vec{x}} e^{i(\vec{k}_n - \vec{p})\cdot(\vec{x}-\vec{x}_0)} \langle 0|O_M(\vec{x}_0, t_0)|\vec{k}_n\rangle \frac{e^{-E_{\vec{k}_n}(t-t_0)}}{2VE_{\vec{k}_n}} \langle \vec{k}_n|O_M^\dagger(\vec{x}_0, t_0)|0\rangle \\
&= a^3 \sum_n |\langle 0|O_M(\vec{x}_0, t_0)|\vec{k}_n\rangle|^2 \frac{e^{-E_{\vec{k}_n}(t-t_0)}}{2E_{\vec{k}_n}} \delta^{(3)}_{\vec{k}_n,\vec{p}}\, e^{-i(\vec{k}_n-\vec{p})\cdot\vec{x}_0} \\
&= a^3 \sum_n \frac{|\langle 0|O_M(\vec{x}_0, t_0)|\vec{p}_n\rangle|^2}{2E_{\vec{p}_n}} e^{-E_{\vec{p}_n}(t-t_0)} \\
(\text{suppose } t_0 = 0) &= \sum_n A_n e^{-E_n t}
\end{aligned}$$

**Real, exponential decay with t。**

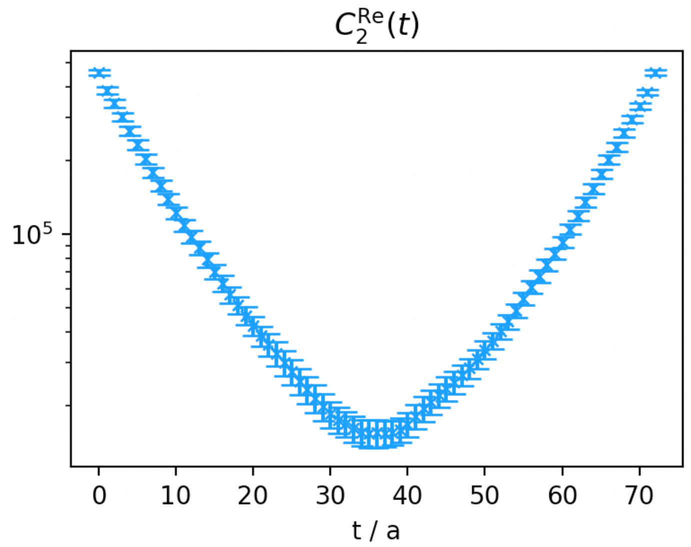
**Ground state + excited states ……**

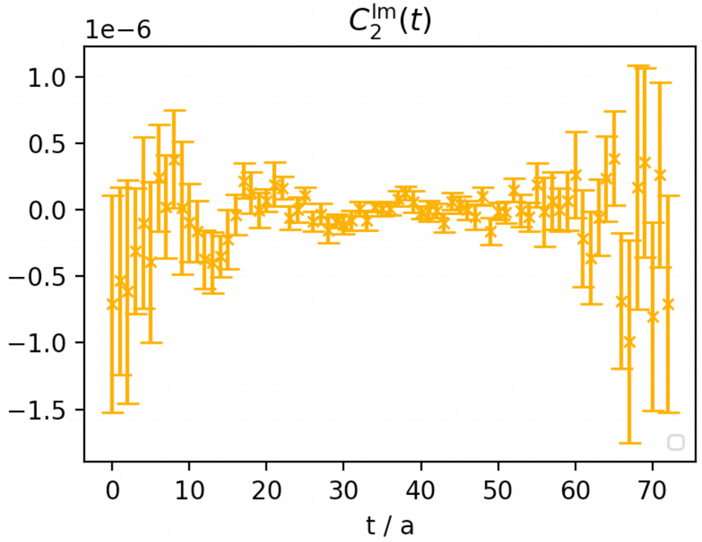
- The Euclidean time dependence of 2pt:

$$\begin{aligned}
\Pi_2(t_0,t;\vec{p},\vec{x}_0) &= a^6\sum_n\sum_{\vec{x}} e^{i(\vec{k}_n-\vec{p})\cdot(\vec{x}-\vec{x}_0)}\langle 0|O_M(\vec{x}_0,t_0)|\vec{k}_n\rangle\frac{e^{-E_{\vec{k}_n}(t-t_0)}}{2VE_{\vec{k}_n}}\langle\vec{k}_n|O_M^\dagger(\vec{x}_0,t_0)|0\rangle \\
&= a^3\sum_n |\langle 0|O_M(\vec{x}_0,t_0)|\vec{k}_n\rangle|^2\frac{e^{-E_{\vec{k}_n}(t-t_0)}}{2E_{\vec{k}_n}}\delta^{(3)}_{\vec{k}_n,\vec{p}}\,e^{-i(\vec{k}_n-\vec{p})\cdot\vec{x}_0} \\
&= a^3\sum_n\frac{|\langle 0|O_M(\vec{x}_0,t_0)|\vec{p}_n\rangle|^2}{2E_{\vec{p}_n}}e^{-E_{\vec{p}_n}(t-t_0)} \\
(\text{suppose } t_0=0) &= \sum_n A_n e^{-E_n t}
\end{aligned}$$

**Real, exponential decay with t。**

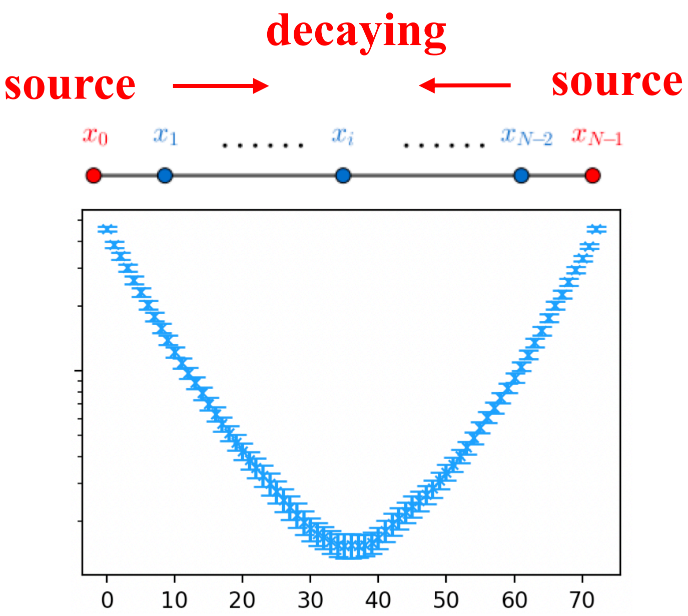
**Ground state + excited states ……**

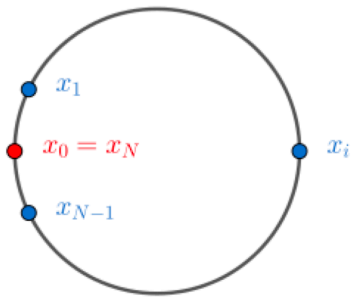



**The decreasing of 2pt depends on the effective energies.**

- **Periodic/Anti-periodic boundary condition:**




Considering the periodic boundary condition:

$$\Pi_2(t) = \sum_n A_n e^{-E_n t} + A_n e^{-E_n(n_t - t)}$$
$$= \sum_n 2A_n e^{-E_n \frac{n_t}{2}} \cosh\left[E_n\left(t - \frac{n_t}{2}\right)\right]$$

- **Extract the hadron effective mass/energy:**

**neglect the excited state contributions**

  - **From the parametrization of 2pt ("log" form):**

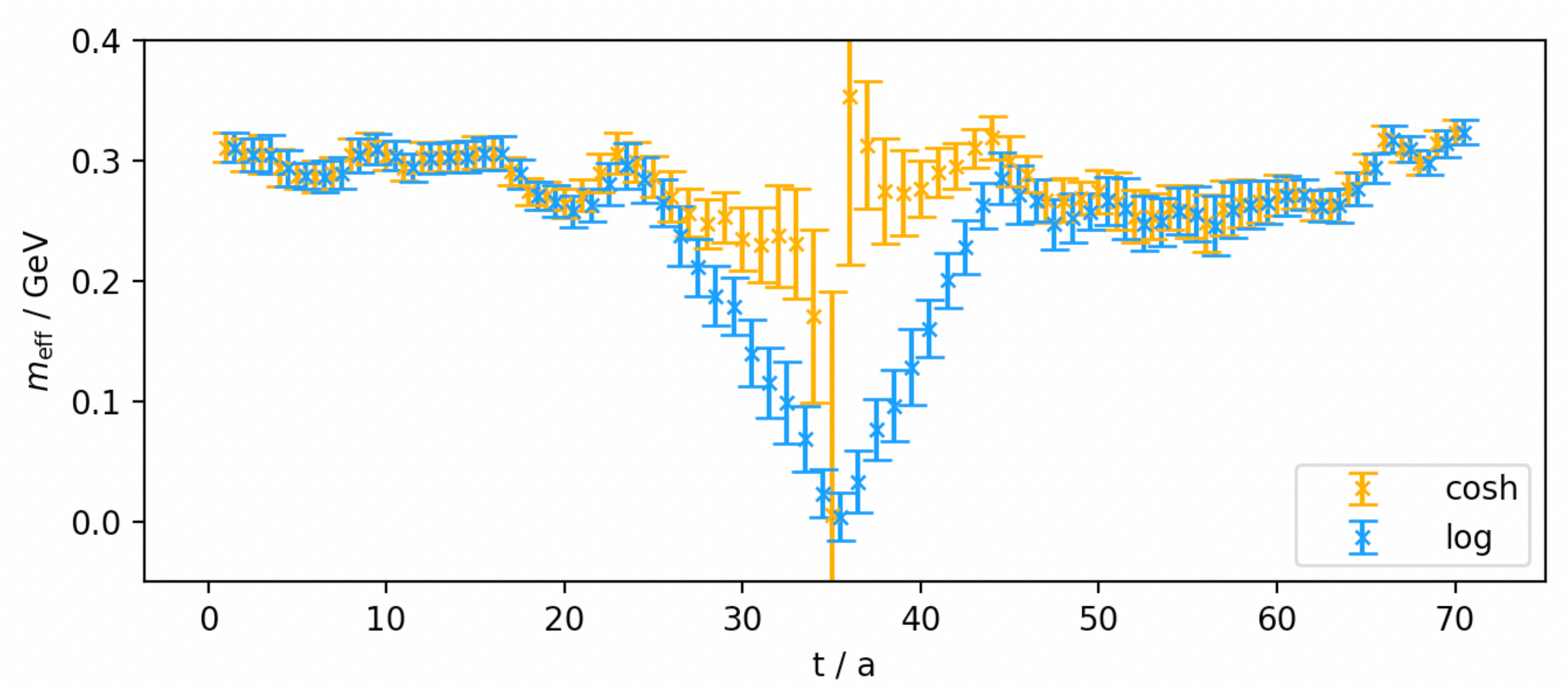
$$\Pi_2(t) \simeq A_0 e^{-m_{\rm eff} t} \quad \Rightarrow \quad m_{\rm eff} \simeq \ln\frac{\Pi_2(t)}{\Pi_2(t+1)}$$

  - **Considering the periodic boundary condition ("cosh" form):**

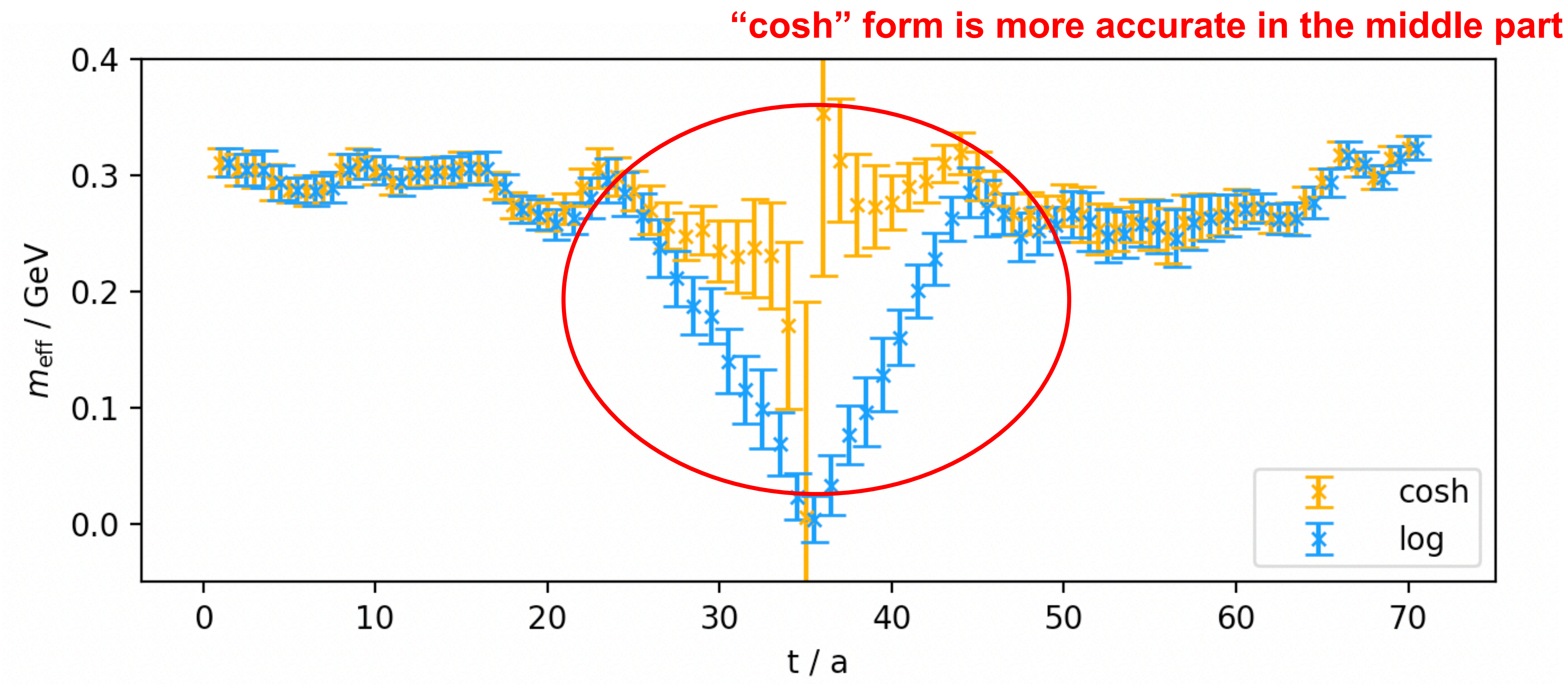
$$\Pi_2(t) = 2A_0 e^{-m_{\rm eff}\frac{n_t}{2}} \cosh\left[m_{\rm eff}\left(t-\frac{n_t}{2}\right)\right]$$

$$\Rightarrow \quad \frac{\Pi_2(t)}{\Pi_2(t+1)} = \frac{\mathrm{Cosh}[m_{\rm eff}(t-n_t/2)]}{\mathrm{Cosh}[m_{\rm eff}(t+1-n_t/2)]}$$

- **Extract the hadron effective mass/energy:**
  - **Comparison of the “log” and “cosh” form:**

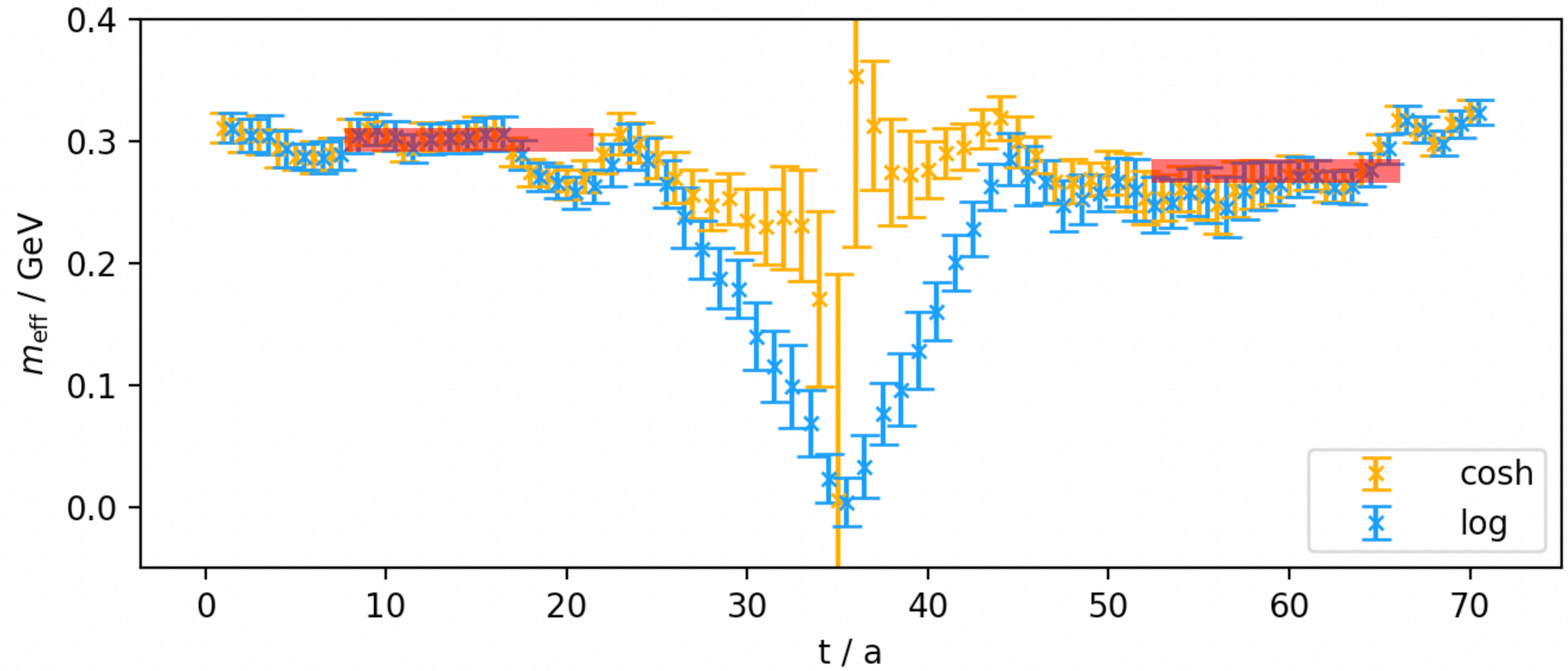


- **Extract the hadron effective mass/energy:**
  - **Comparison of the "log" and "cosh" form:**



- **Effective mass can be extracted from the plateau at:**

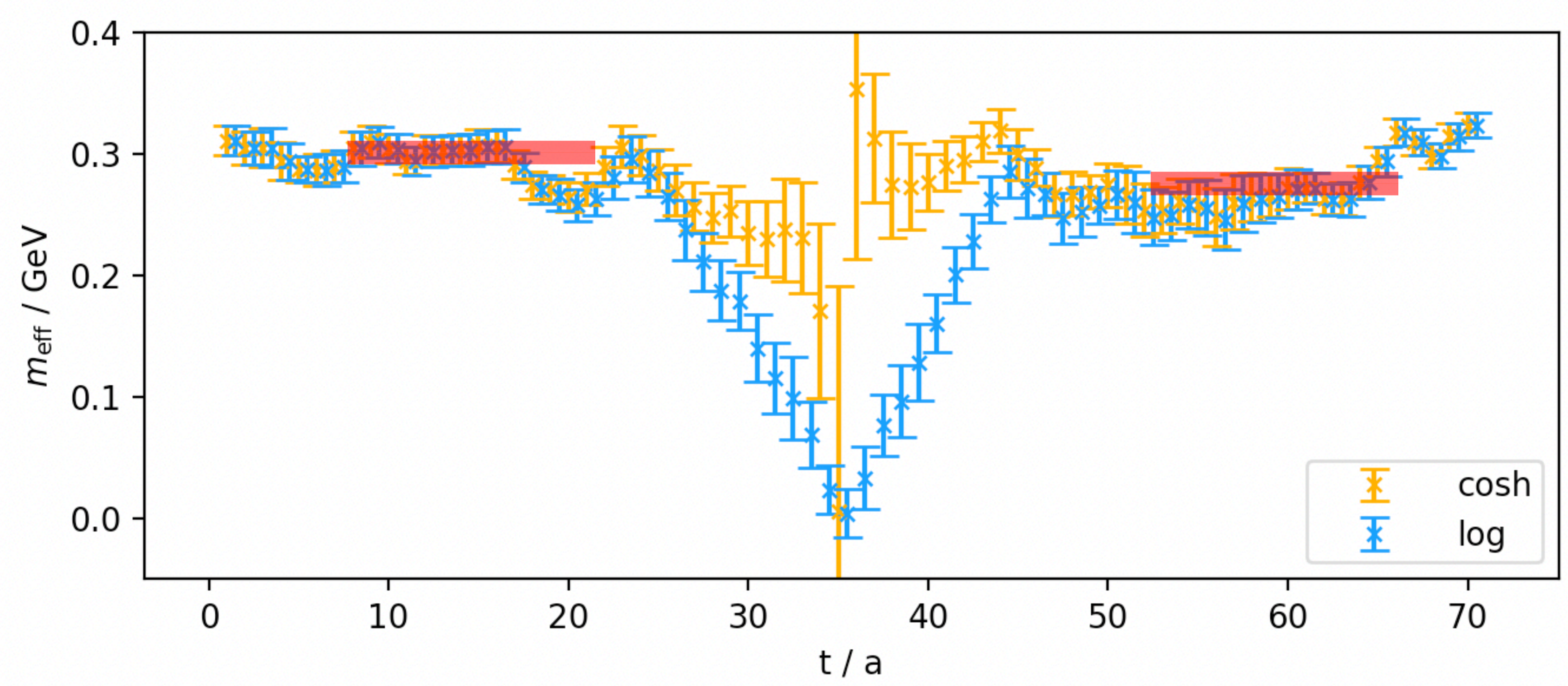
$$\begin{cases} t \gg 0 \text{ or } t \ll n_t \text{ , } e^{-E_0 t} > e^{-E_1 t}\text{, ground state dominant;} \\ t < n_t/2 \text{ or } t > n_t/2\text{, forward or backward dominant.} \end{cases}$$




About here…

- **Effective mass can be extracted from the plateau at:**

$$\begin{cases} t \gg 0 \text{ or } t \ll n_t\ ,\ e^{-E_0 t} > e^{-E_1 t}, \text{ ground state dominant;} \\ t < n_t/2 \text{ or } t > n_t/2, \text{ forward or backward dominant.} \end{cases}$$

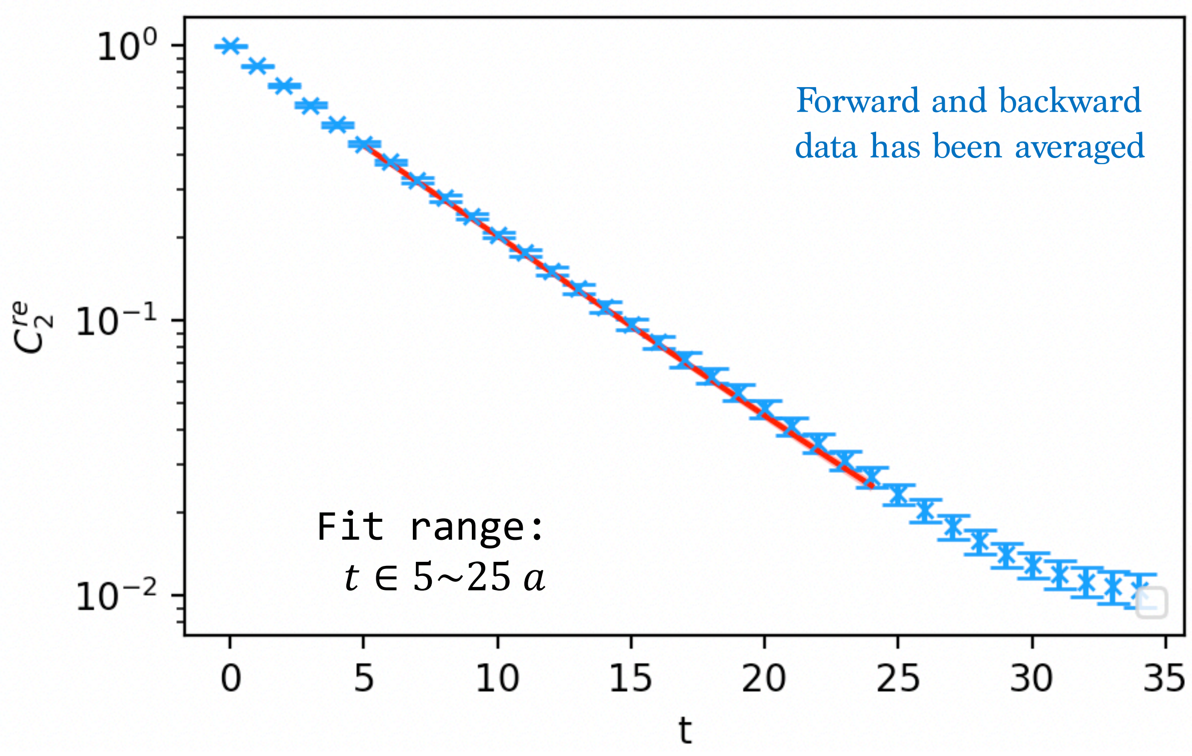



But how to exactly determining?

- **Extract the hadron effective mass from fitting 2pt:**

  - **With "cosh" form**

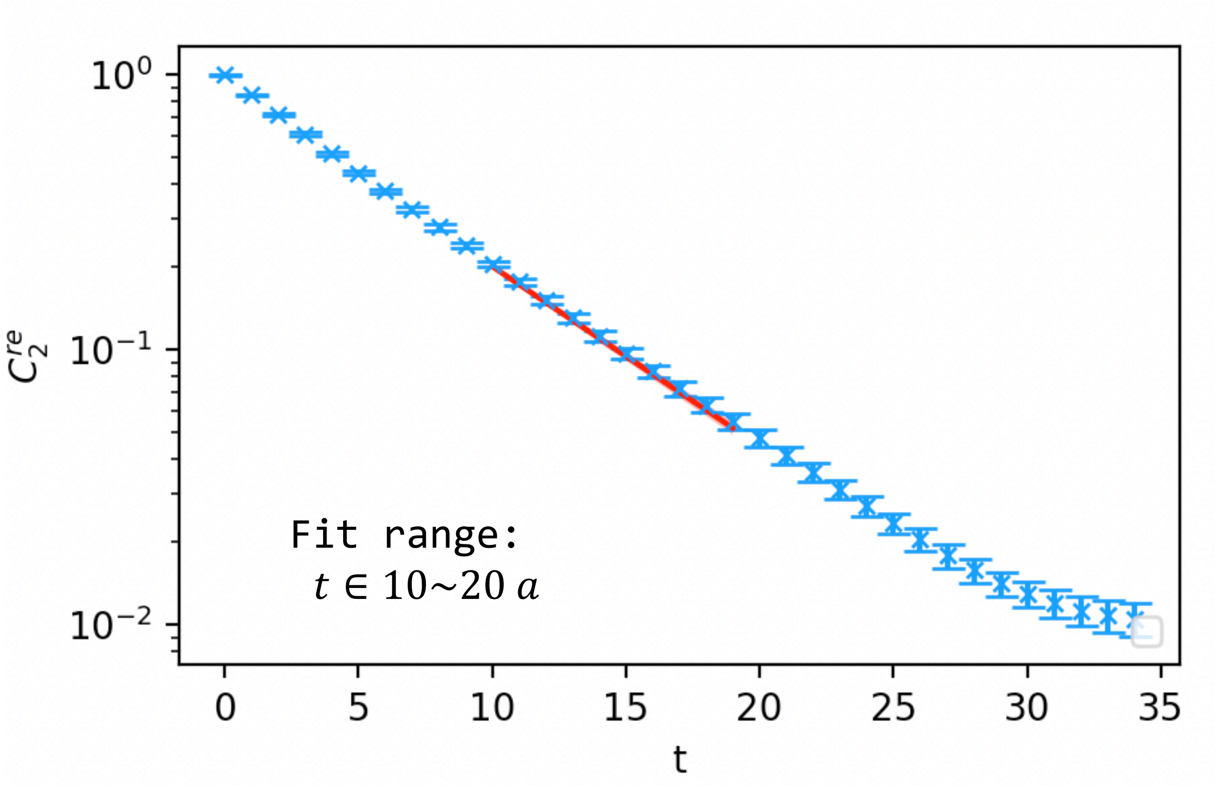
$$\Pi_2(t) = A_0 e^{-E_0 \frac{n_t}{2}} \cosh\left[E_0\left(t - \frac{n_t}{2}\right)\right]$$




```
====== PION ======
Least Square Fit:
  chi2/dof [dof] = 1.6 [20]    Q = 0.052    logGBF = 116.01

Parameters:
             E0   0.1512 (26)     [  0.20 (50) ]
             A0    1.844 (35)     [  0.0 (5.0) ]

Fit:
              key          y[key]       f(p)[key]
-----------------------------------------------
    2pt_forward 0    0.4408 (81)    0.4329 (55)
                1    0.3775 (77)    0.3722 (47)
                2    0.3245 (72)    0.3200 (42)
                3    0.2789 (66)    0.2751 (39)
                4    0.2394 (59)    0.2365 (37)
                5    0.2053 (53)    0.2033 (35)
                6    0.1765 (50)    0.1748 (33)
                7    0.1517 (47)    0.1503 (32)
                8    0.1305 (46)    0.1293 (30)
                9    0.1124 (45)    0.1112 (28)
               10    0.0971 (45)    0.0956 (27)
               11    0.0836 (43)    0.0822 (25)
               12    0.0723 (41)    0.0708 (23)
               13    0.0630 (39)    0.0609 (21)
               14    0.0549 (36)    0.0524 (20)
               15    0.0478 (34)    0.0452 (18)
               16    0.0415 (31)    0.0389 (17)
               17    0.0360 (28)    0.0336 (15)
               18    0.0311 (25)    0.0290 (14)
               19    0.0270 (23)    0.0251 (13)

Settings:
  svdcut/n = 1e-08/0    tol = (1e-08,1e-10,1e-10*)    (itns,
  fitter = scipy_least_squares    method = trf

m=0.2833(48)GeV
```

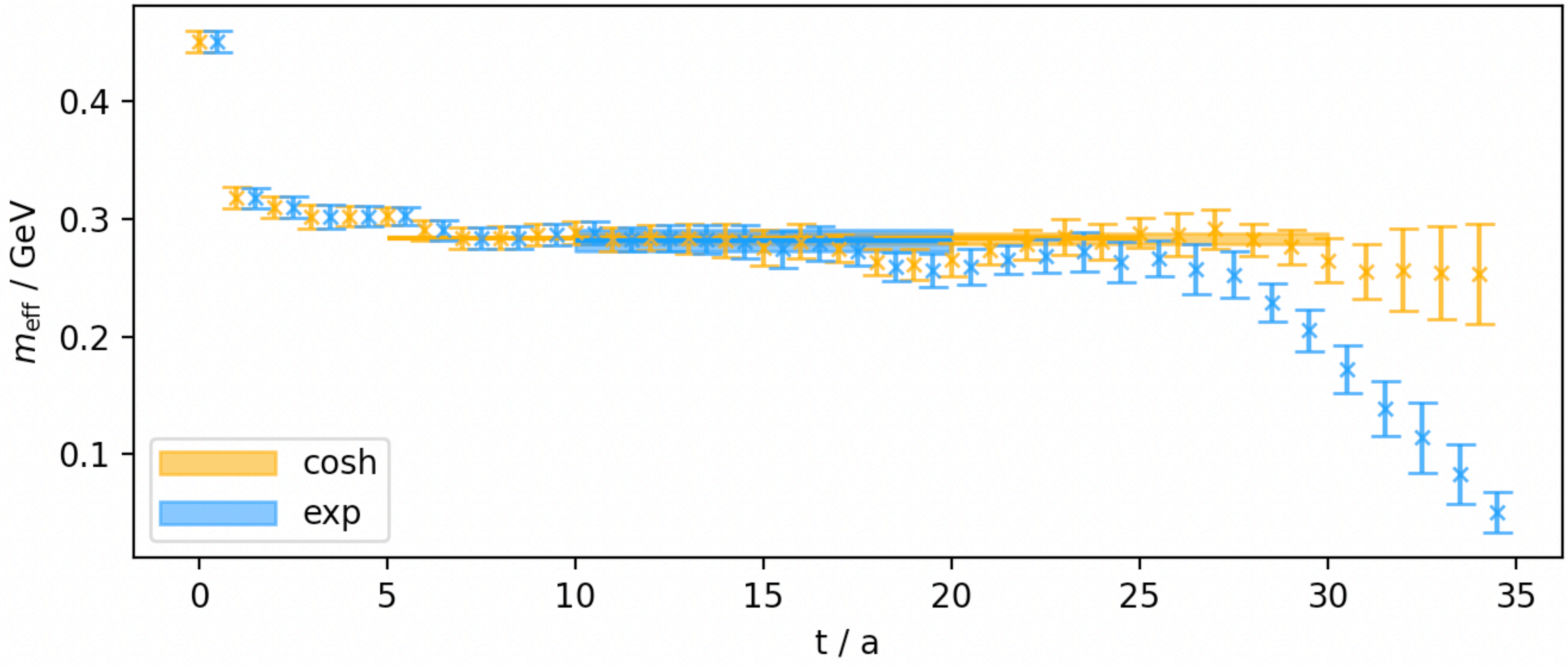
- **Extract the hadron effective mass from fitting 2pt:**
  - **With ground state "exp" form**

$$\Pi_2(t) \;=\; A_0 e^{-E_0 t}$$

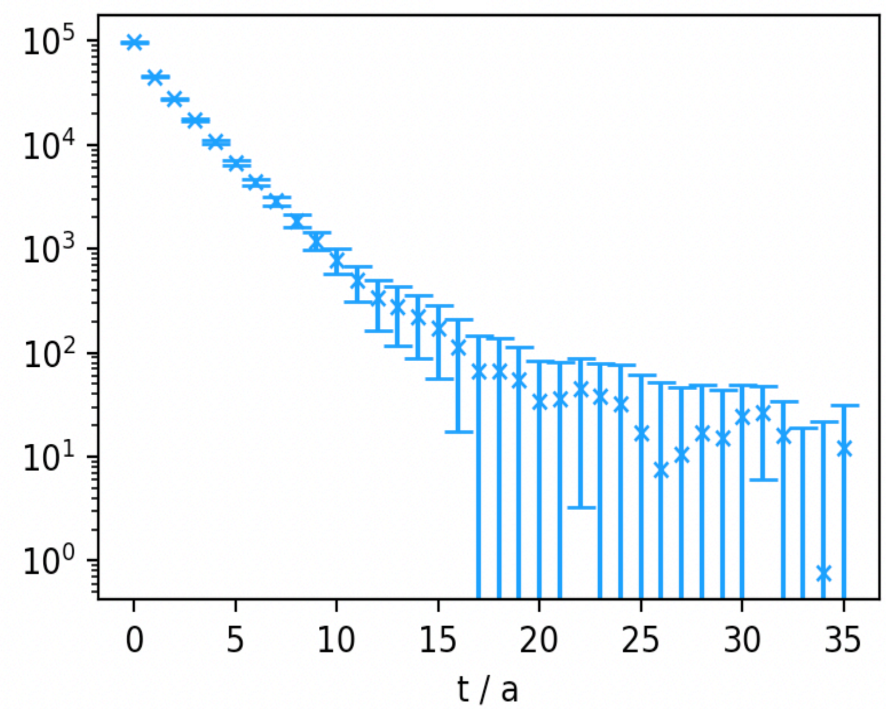



```
====== PION ======
Least Square Fit:
  chi2/dof [dof] = 0.99 [10]     Q = 0.45     logGE

Parameters:
             E0    0.1504 (47)      [  0.0 (2.0) ]
             A0     0.905 (39)      [  0.0 (5.0) ]

Fit:
      key         y[key]       f(p)[key]
---------------------------------------
    2pt 0     0.2053 (53)     0.2011 (45)
          1     0.1765 (50)     0.1730 (43)
          2     0.1517 (47)     0.1488 (41)
          3     0.1305 (46)     0.1281 (39)
          4     0.1124 (45)     0.1102 (38)
          5     0.0971 (45)     0.0948 (36)
          6     0.0836 (43)     0.0816 (34)
          7     0.0723 (41)     0.0702 (32)
          8     0.0630 (39)     0.0604 (30)
          9     0.0549 (36)     0.0520 (28)

Settings:
  svdcut/n = 1e-08/0     tol = (1e-08,1e-10,1e-10
  fitter = scipy_least_squares     method = trf

m=0.2818(88)GeV
```

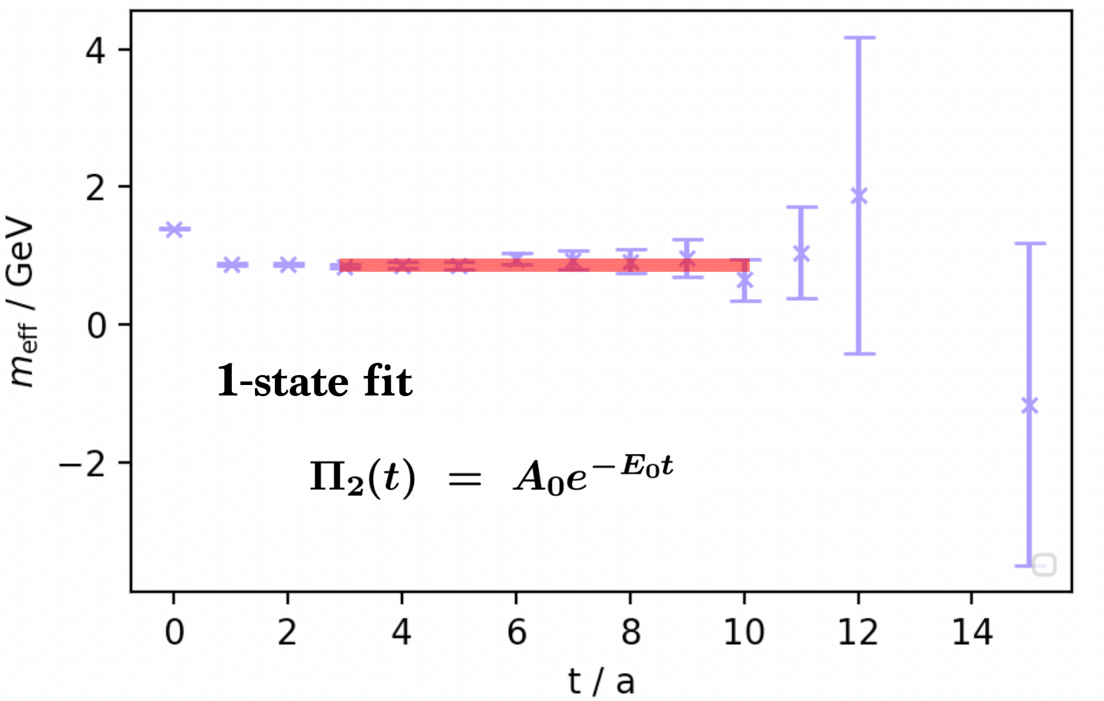
- **Comparison of different fit formula:**



⚠ **Except for the pseudoscalar meson, the signal-to-noisy ratio of 2pt will exponentially increasing:**

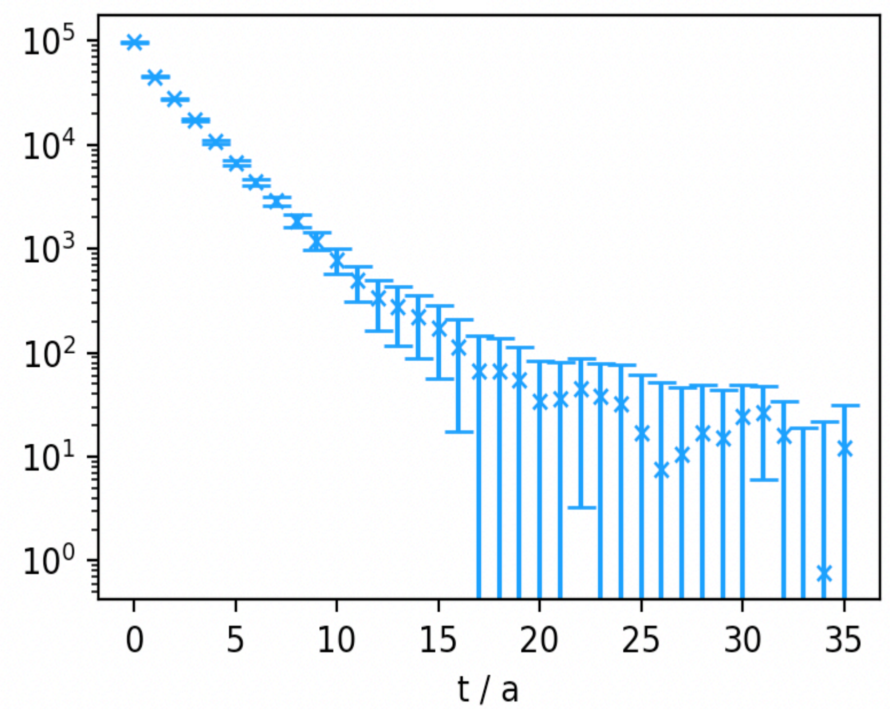



**2pt, $\rho$ meson**

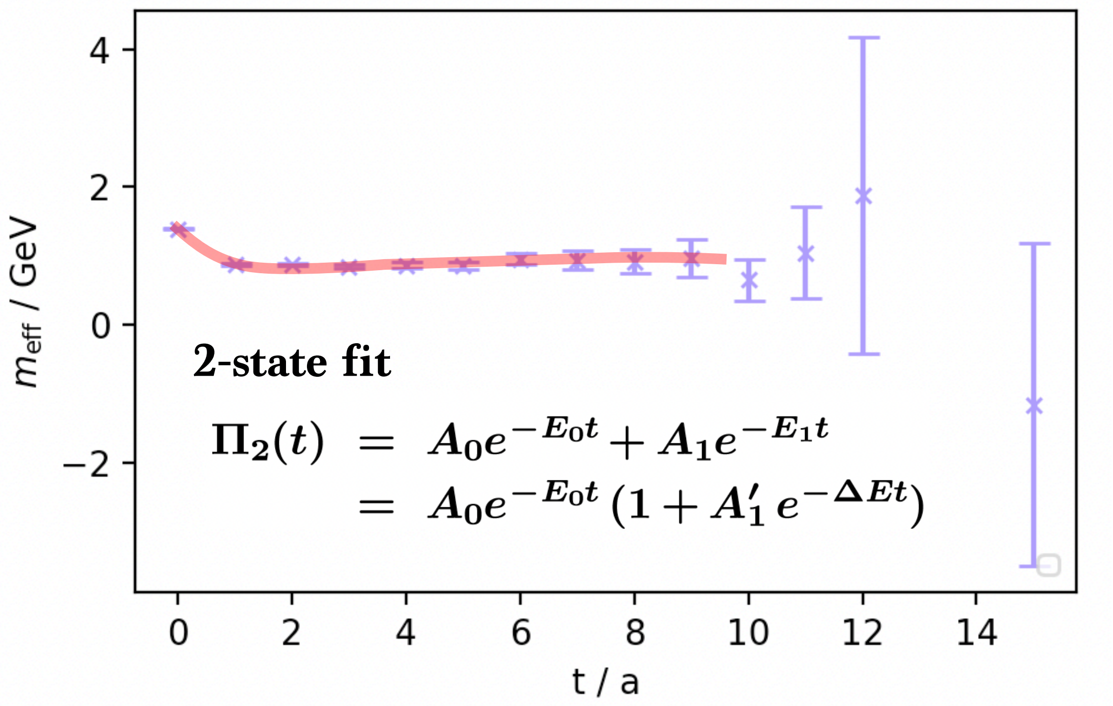



**Effective mass curve, $\rho$ meson**

⚠️ **Except for the pseudoscalar meson, the signal-to-noisy ratio of 2pt will exponentially increasing:**




2pt, $\rho$ meson




Effective mass curve, $\rho$ meson

## 🤔 **Exercise:**

1. Repeat the generation and analysis of pion 2pt;
2. Use Chroma to calculate the numerically results of kaon and $\rho$ 2pt, and extract their effective masses.

Hints:

- $K\ (u\bar{s})$: $J^P = 0^-$, $\Gamma = \gamma_5$;
- ρ $(u\bar{d})$: $J^P = 1^-$, $\Gamma = \vec{\gamma}$.

## Baryon 2pt and proton effective mass

- Interpolators for spin-1/2 baryons:

$$O_\gamma^B(x) = \epsilon^{abc} P_\pm \psi_\alpha^{f_1,a}(x)(C\gamma_5)_{\alpha\beta}\psi_\beta^{f_2,b}(x)\psi_\gamma^{f_3,c}(x)$$

- $f_i$: flavor of the $i$-th quark, related to the quantum numbers like isospin, strangeness……
- $a/b/c$: color indices.

  Color singlet, all color indices anti-symmetric

- $\alpha/\beta/\gamma$: spinor indices.

  Diquark ($I = J = 0$) + single quark ($I = J = 1/2$, dominant the quantum number of baryon)

- $P_\pm = (1 \pm \gamma^t)/2$: parity projector, make sure the right parity of baryon.

# Baryon 2pt and proton effective mass

- Interpolators for spin-1/2 baryons:

$$O_\gamma^B(x) = \epsilon^{abc} P_\pm \psi_\alpha^{f_1,a}(x)(C\gamma_5)_{\alpha\beta}\psi_\beta^{f_2,b}(x)\psi_\gamma^{f_3,c}(x)$$

- $f_i$: flavor of the $i$-th quark, related to the quantum numbers like isospin, strangeness……
- $a/b/c$: color indices.

  Color singlet, all color indices anti-symmetric

- $\alpha/\beta/\gamma$: spinor indices.

  Diquark ($I = J = 0$) + single quark ($I = J = 1/2$, dominant the quantum number of baryon)

- $P_\pm = (1 \pm \gamma^t)/2$: parity projector, make sure the right parity of baryon.

➢ Example:

proton: $uud$, $I(J^P) = \frac{1}{2}\left(\frac{1}{2}\right)^+ \Rightarrow f_1 = f_3 = u,\ f_2 = d,\ P_+ = (1 + \gamma_t)/2$

- Interpolators of proton:

$$
\begin{aligned}
O^{p}_{\gamma}(x) &= \epsilon^{abc} P_{+} \psi^{u,a}_{\alpha}(x) (C\gamma_5)_{\alpha\beta} \psi^{d,b}_{\beta}(x) \psi^{u,c}_{\gamma}(x) \\
\bar{O}^{p}_{\gamma}(x) &= -\epsilon^{abc} P_{+} \bar{\psi}^{d,a}_{\alpha}(x) (C\gamma_5)_{\alpha\beta} \bar{\psi}^{u,b}_{\beta}(x) \bar{\psi}^{u,c}_{\gamma}(x)
\end{aligned}
$$

- Interpolators of proton:

$$
\begin{aligned}
O^p_\gamma(x) &= \epsilon^{abc} P_+ \psi^{u,a}_\alpha(x) (C\gamma_5)_{\alpha\beta} \psi^{d,b}_\beta(x) \psi^{u,c}_\gamma(x) \\
\bar{O}^p_\gamma(x) &= -\epsilon^{abc} P_+ \bar{\psi}^{d,a}_\alpha(x) (C\gamma_5)_{\alpha\beta} \bar{\psi}^{u,b}_\beta(x) \bar{\psi}^{u,c}_\gamma(x)
\end{aligned}
$$

- Proton 2pt: *more details see the backup slides*

$$
\begin{aligned}
\langle O^p_\gamma(x) \bar{O}^p_{\gamma'}(x_0) \rangle &= \mathrm{traceColor}(G^u * \mathrm{traceSpin}(\mathrm{quarkContract13}(G^u C\gamma_5, C\gamma_5 G^d)))_{\gamma\gamma'} \\
&+ \mathrm{traceColor}(\mathrm{transposeSpin}(\mathrm{quarkContract24}(C\gamma_5 G^d, G^u C\gamma_5)) * G^u)_{\gamma\gamma'}
\end{aligned}
$$

- Interpolators of proton:

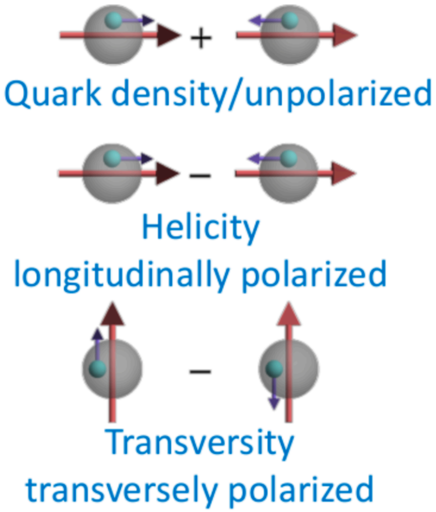
$$O^{p}_{\gamma}(x) = \epsilon^{abc} P_{+} \psi^{u,a}_{\alpha}(x)(C\gamma_5)_{\alpha\beta}\psi^{d,b}_{\beta}(x)\psi^{u,c}_{\gamma}(x)$$

$$\bar{O}^{p}_{\gamma}(x) = -\epsilon^{abc} P_{+} \bar{\psi}^{d,a}_{\alpha}(x)(C\gamma_5)_{\alpha\beta}\bar{\psi}^{u,b}_{\beta}(x)\bar{\psi}^{u,c}_{\gamma}(x)$$

- Proton 2pt: *more details see the backup slides*

$$\langle O^{p}_{\gamma}(x)\bar{O}^{p}_{\gamma'}(x_0)\rangle = \mathrm{traceColor}(G^u * \mathrm{traceSpin}(\mathrm{quarkContract13}(G^u C\gamma_5, C\gamma_5 G^d)))_{\gamma\gamma'} + \mathrm{traceColor}(\mathrm{transposeSpin}(\mathrm{quarkContract24}(C\gamma_5 G^d, G^u C\gamma_5)) * G^u)_{\gamma\gamma'}$$

Polarization of the external proton




For simplicity, we choose the unpolarized projection:

$$T_{\mathrm{unpol}} = (1 + \gamma^t)/2$$

then we can obtain the 2pt on momentum space:

$$\Pi_2(t_0, t; \vec{p}, \vec{x}_0) = \sum_{\vec{x}} e^{-i\vec{p}\cdot(\vec{x}-\vec{x}_0)} T_{\mathrm{unpol}} \langle O^p(x)\bar{O}^p(x_0)\rangle$$

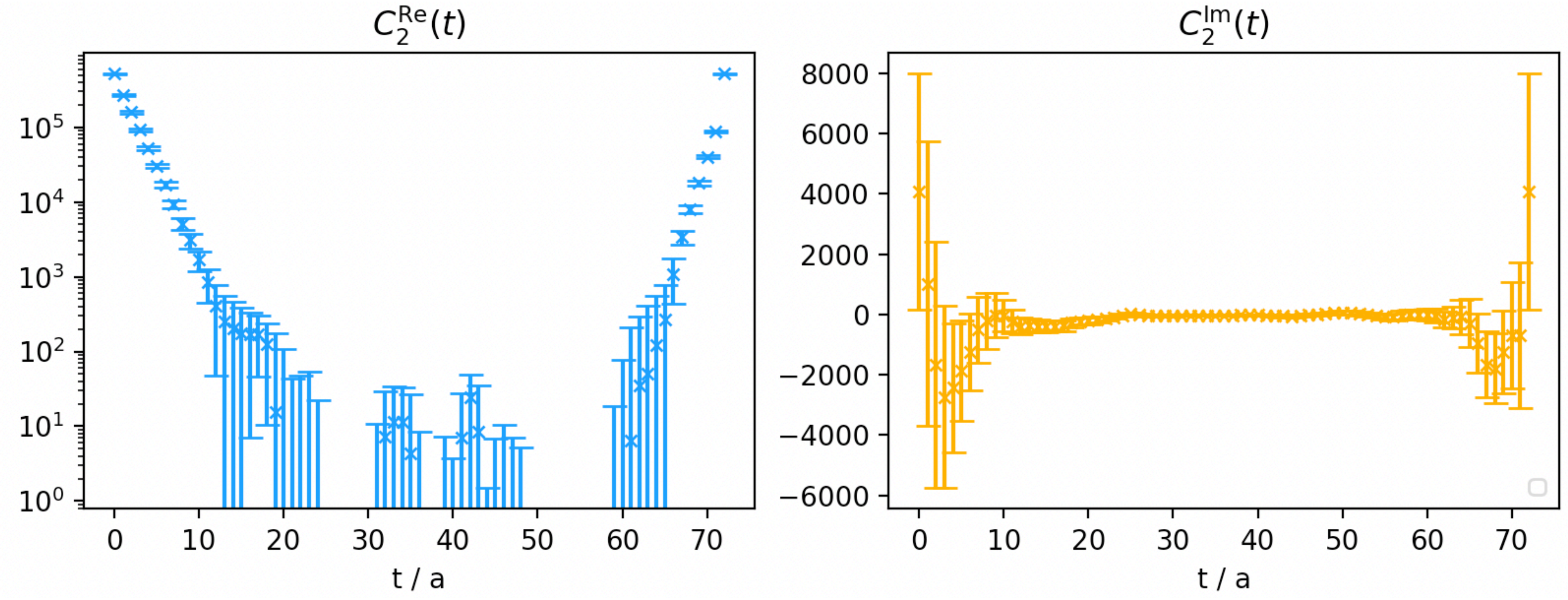
- Program realization of the proton 2pt:

$$T^{\text{unpol}}_{\gamma'\gamma}\langle O^p_\gamma(x)\bar{O}^p_{\gamma'}(x_0)\rangle = \mathbf{trace}(T*\mathbf{traceColor}(G^u*\mathbf{traceSpin}(\mathbf{quarkContract13}(G^uC\gamma_5, C\gamma_5G^d))))$$
$$+\mathbf{trace}(T*\mathbf{traceColor}(\mathbf{transposeSpin}(\mathbf{quarkContract24}(C\gamma_5G^d, G^uC\gamma_5))*G^u))$$

```
else if (hadron_list[i] == "PROTON")
{
  corr = LatticeComplex( trace(T_unpol * traceColor(L_prop * traceSpin(quarkContract13(L_prop * Cg5, Cg5 * L_prop))))
          + trace(T_unpol * traceColor(transposeSpin(quarkContract24(Cg5 * L_prop, L_prop * Cg5)) * L_prop)) );
}
```

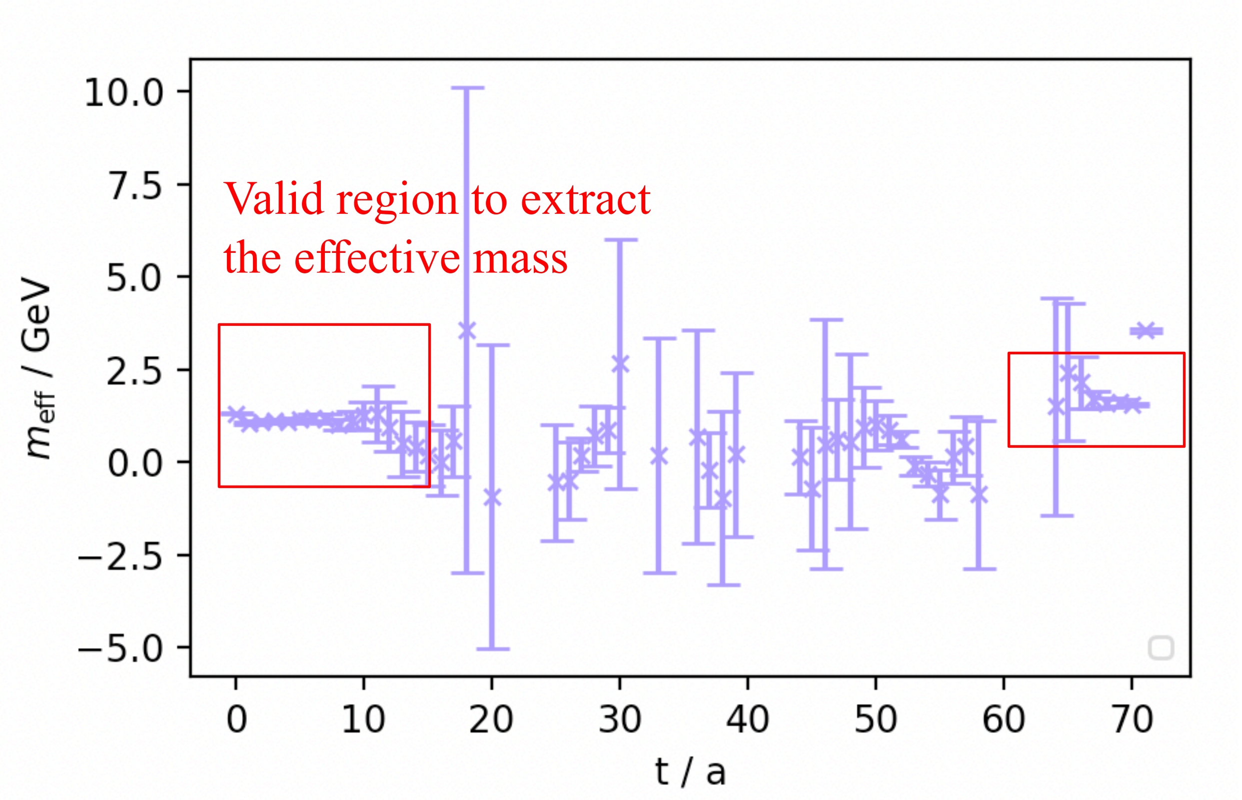
**Compile, submit, read, ……**

- Numerical results of proton 2pt:

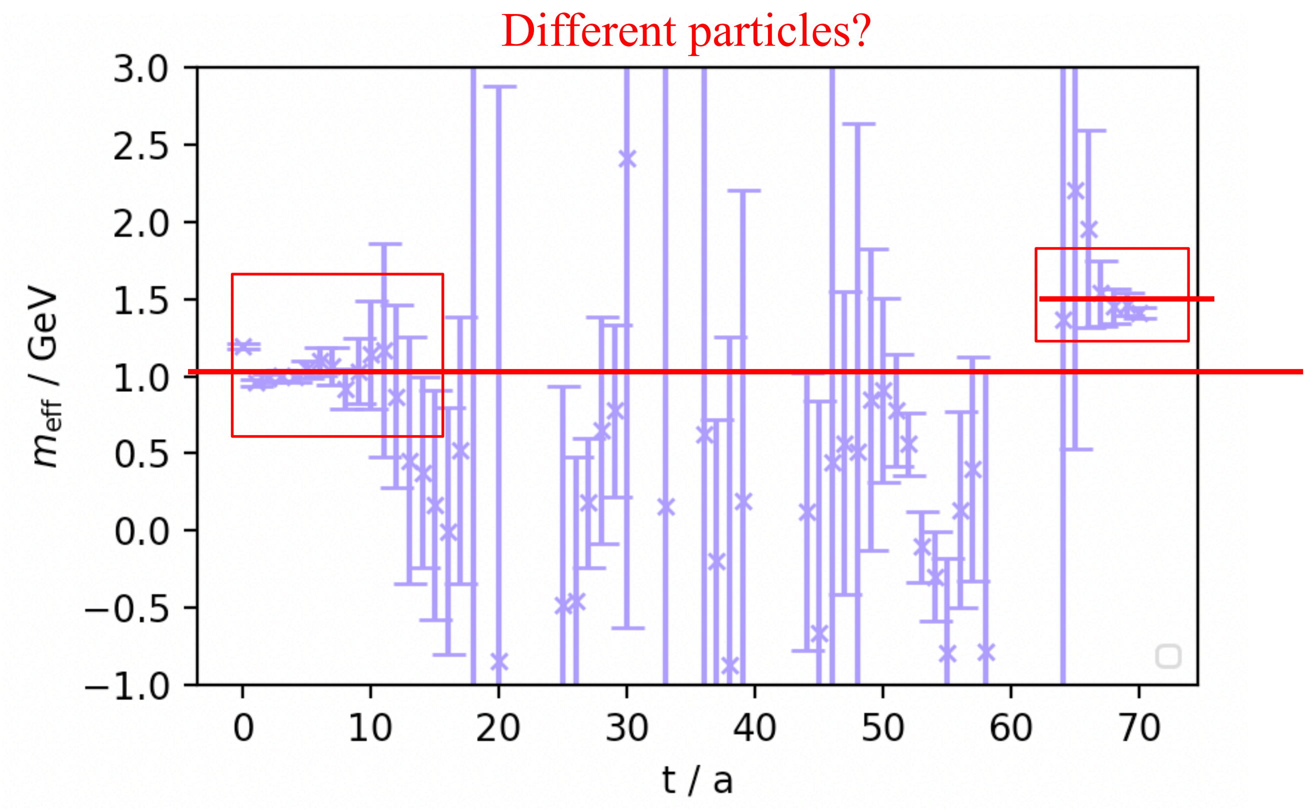



Uncertainties exponentially increasing with propagating

- Effective mass of proton:



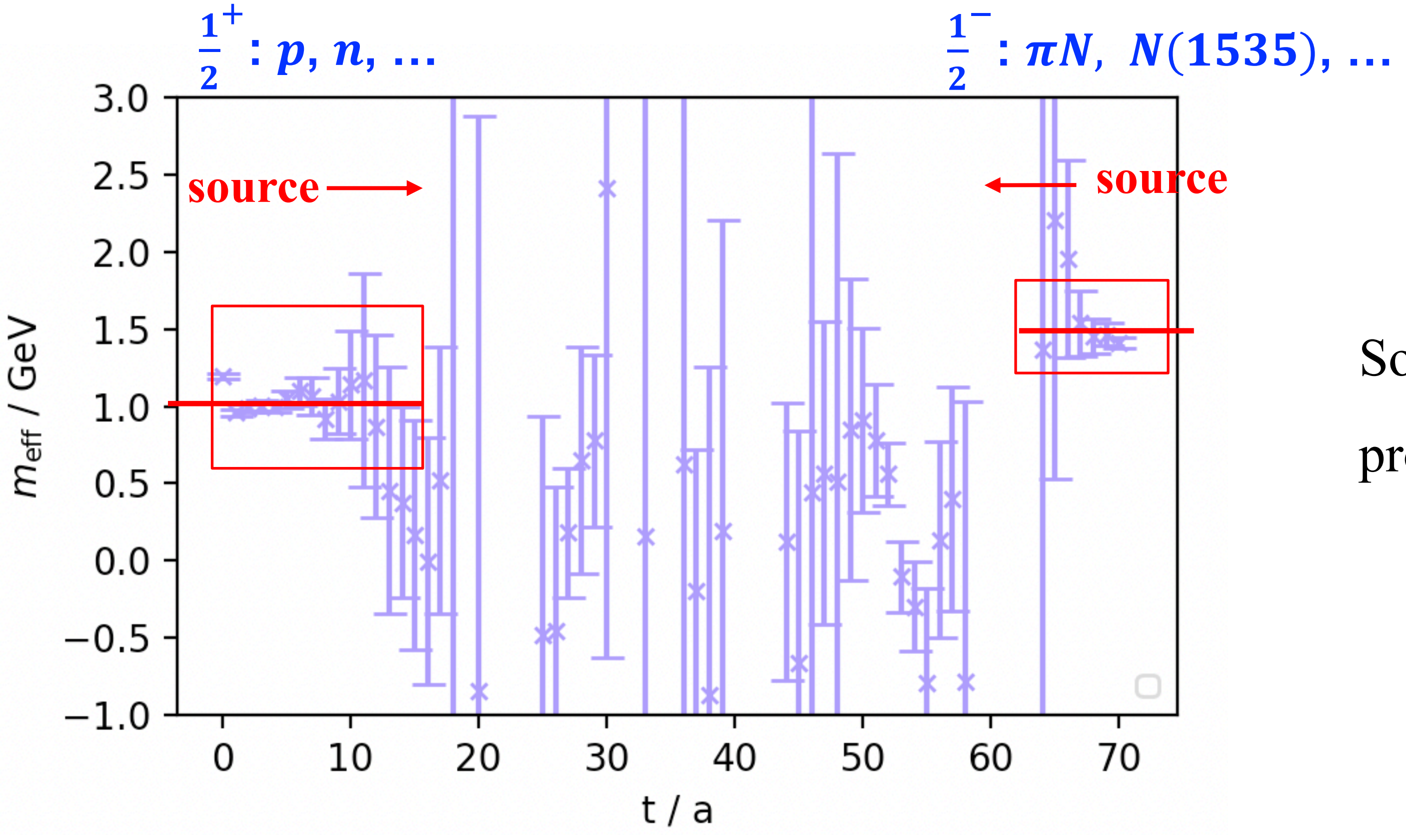
- Effective mass of proton:



- Effective mass of proton:

Parity: forward propagating => +

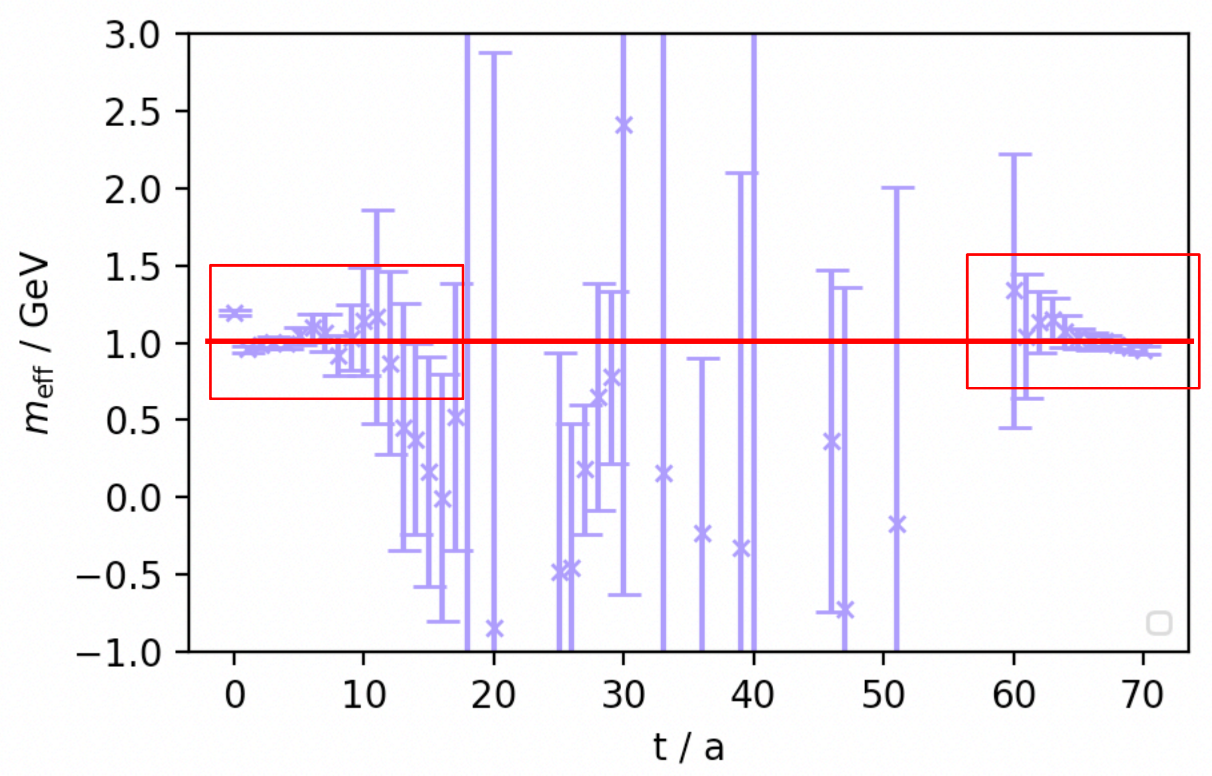
backward propagating => -




So we can extract the effective mass of proton from the first window, or ……

……or use $P_{-}$ for the backward propagation:

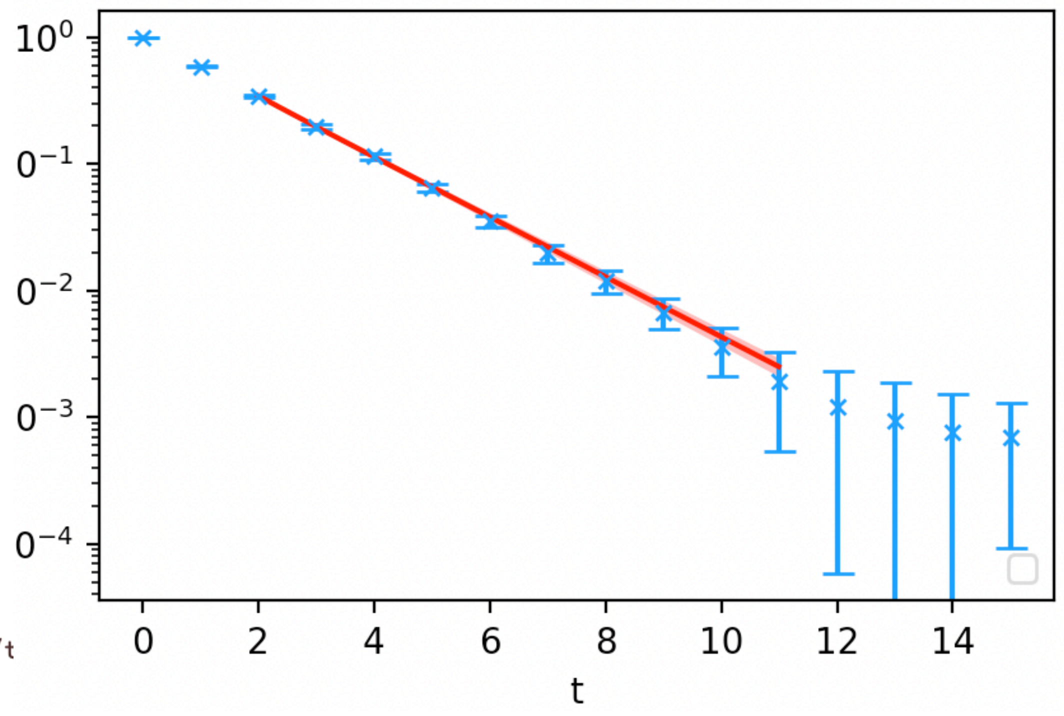
```
int nt=Layout::lattSize()[3];
int t0=0;
SpinMatrix prj_p(0.5 * (g_one + (g_one * Gamma(8))));
SpinMatrix prj_m(0.5 * (g_one - (g_one * Gamma(8))));
LatticeSpinMatrix T_unpol =
    where(((Layout::latticeCoordinate(3))-t0+nt)%nt<nt/2,
        prj_p,prj_m);
```



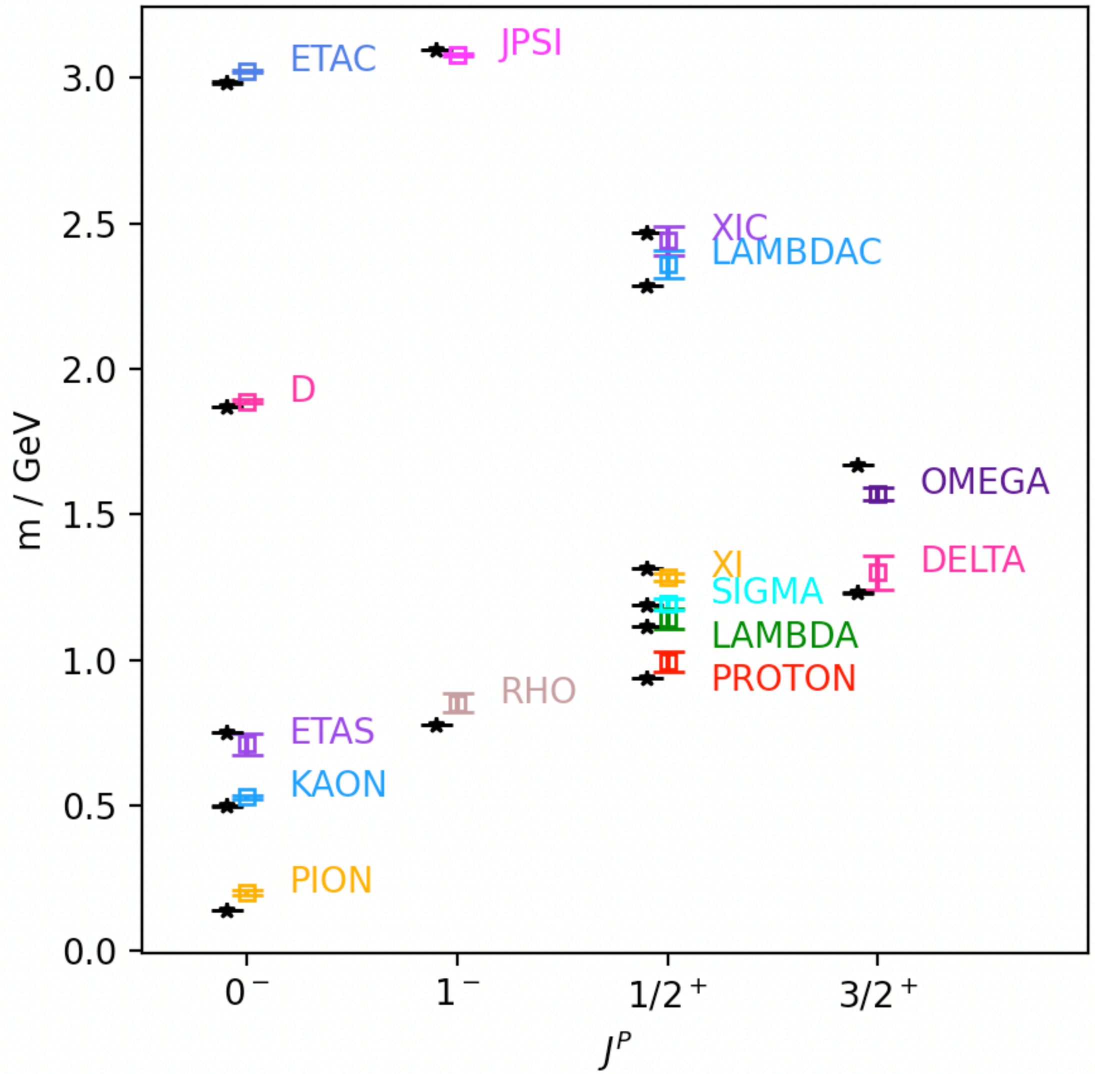
- Extract effective mass from joint fit

```
====== PROTON ======
Least Square Fit:
  chi2/dof [dof] = 0.8 [10]     Q = 0.63     logGBF = 42.745

Parameters:
             E0    0.545 (19)      [  2.0 (2.0) ]
             A0    1.008 (64)      [  0.0 (5.0) ]
             A1     0.4 (1.3)      [  0.0 (5.0) ]
         deltaE     1.5 (1.9)      [  1.0 (2.0) ]

Fit:
      key          y[key]        f(p)[key]
----------------------------------------
    2pt 0      0.3442 (85)      0.3456 (75)
        1      0.1981 (80)      0.1972 (71)
        2      0.1147 (62)      0.1139 (54)
        3      0.0647 (46)      0.0660 (41)
        4      0.0355 (39)      0.0382 (30)
        5      0.0198 (32)      0.0222 (21)
        6      0.0120 (24)      0.0129 (14)
        7      0.0068 (19)     0.00745 (97)
        8      0.0036 (15)     0.00432 (64)
        9      0.0019 (14)     0.00250 (42)

Settings:
  svdcut/n = 1e-08/0    tol = (1e-08,1e-10,1e-10*)    (itns/t
  fitter = scipy_least_squares    method = trf

m=0.995(35)GeV
```

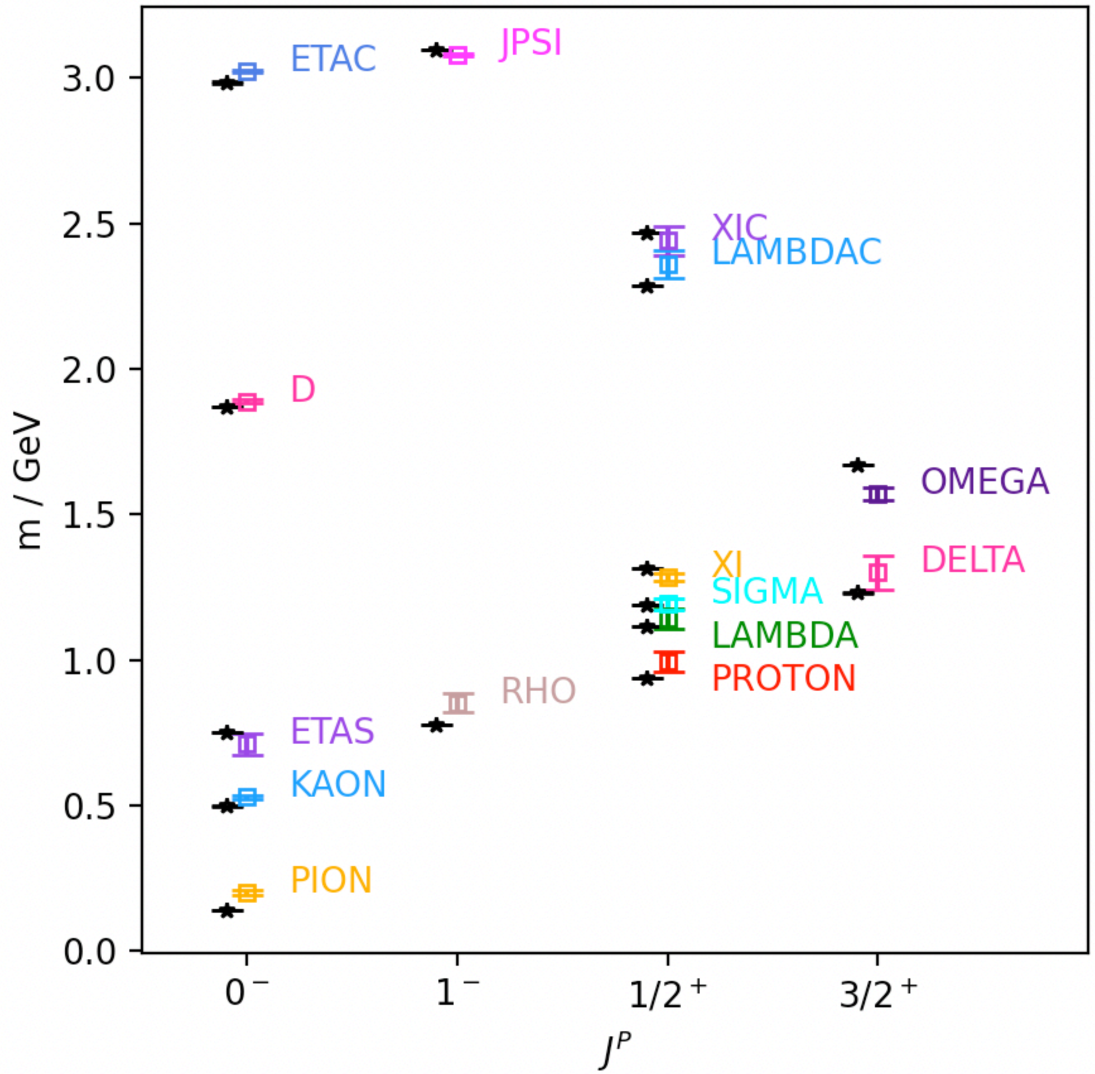
$$\Pi_2(t) = A_0\, e^{-E_0 t} + A_1'\, e^{-E_1 t} + \cdots$$
$$= A_0\, e^{-E_0 t}\,(1 + e^{-\Delta E t})$$



# Hadron Spectrum



# Hadron Spectrum




| Name | $n_s^3 \times n_t$ | $a$ | $m_\pi$ | $m_{\eta_s}$ |
|---|---|---|---|---|
| C24P34 | $24^3 \times 64$ | 0.10530 | 340.5(1.7) | 748.7(0.9) |
| C24P29 | $24^3 \times 72$ | | 292.7(1.2) | 657.4(0.6) |
| C32P29 | $32^3 \times 64$ | | 292.4(1.1) | 658.0(0.7) |
| C32P23 | $32^3 \times 64$ | | 228.0(1.2) | 643.5(0.5) |
| C48P23 | $48^3 \times 96$ | | 225.61(86) | 643.2(0.5) |
| C48P14 | $48^3 \times 96$ | | 135.5(1.6) | 706.3(0.3) |
| F32P30 | $32^3 \times 96$ | 0.07746 | 303.2(1.3) | 681.6(0.9) |
| F48P30 | $48^3 \times 96$ | | 303.44(86) | 679.9(0.6) |
| F32P21 | $32^3 \times 64$ | | 210.9(2.2) | 665.6(0.7) |
| F48P21 | $48^3 \times 96$ | | 207.2(1.1) | 667.7(0.7) |
| H48P32 | $48^3 \times 144$ | 0.05187 | 321.44(79) | 709.0(0.5) |

- Finite size effects and infinite volume limit $V \to \infty$;
- Scaling in the continuum limit $a \to 0$;
- Chiral extrapolation $m \to m_{\exp}$.

THANKS FOR YOUR ATTENTION

# Backup slides

- **Notations and conventions**

In the Minkovski space, the Dirac matrix defined as

$$\{\gamma^\mu, \gamma^\nu\} = 2g^{\mu\nu}, \qquad g^{\mu\nu} = g_{\mu\nu} = \mathrm{Diag}(+1, -1, -1, -1),$$

$$\{\gamma^\mu, \gamma_5\} = 0, \qquad \gamma_5 = i\gamma^0\gamma^1\gamma^2\gamma^3,$$

with

$$\gamma^0 = \begin{pmatrix} I & 0 \\ 0 & -I \end{pmatrix}, \quad \gamma^i = \begin{pmatrix} 0 & \sigma^i \\ -\sigma^i & 0 \end{pmatrix}, \quad \gamma_5 = \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix},$$

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \sigma^1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma^2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma^3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

and the Pauli matrix following

$$[\sigma_i, \sigma_j] = 2i\epsilon_{ijk}\sigma_k, \quad \{\sigma_i, \sigma_j\} = 2\delta_{ij}, \quad \sigma_i\sigma_j = i\epsilon_{ijk}\sigma_k,$$
$$\mathrm{Tr}\,(\sigma_i\sigma_j) = 2\delta_{ij}, \quad \epsilon_{ijm}\epsilon_{klm} = \delta_{ik}\delta_{jl} - \delta_{il}\delta_{jk}.$$

And switch to Euclidean space, we have the relations

$$\gamma_i^{(E)} = -i\gamma_i^{(M)}, \qquad \gamma_0^{(E)} = \gamma_0^{(M)}, \qquad \gamma_5^{(E)} = \gamma_0^{(E)}\gamma_1^{(E)}\gamma_2^{(E)}\gamma_3^{(E)},$$

so all $\gamma$-matrix in the Euclidean space are Hermitian:

$$\left(\gamma_\mu^{(E)}\right)^\dagger = \gamma_\mu^{(E)},$$

and satisfies the anti-commute relation:

$$\left\{\gamma_\mu^{(E)}, \gamma_\nu^{(E)}\right\} = 2\delta_{\mu\nu}, \qquad \left\{\gamma_\mu^{(E)}, \gamma_5^{(E)}\right\} = 0.$$

- **Contract rules of spin and color indices**

  - `quarkContract13(a,b)`
    Effectively, one can use this to construct an anti-quark from a di-quark contraction. In explicit index form, the operation `quarkContract13` does

$$\mathrm{qC}_{13}\left[G^{(1)}, G^{(2)}\right]_{\alpha\beta}^{c'c} = \epsilon^{abc}\epsilon^{a'b'c'} * G_{\rho\alpha}^{(1),aa'} * G_{\rho\beta}^{(2),bb'},$$

  - `quarkContract14(a,b)`

$$\mathrm{qC}_{14}\left[G^{(1)}, G^{(2)}\right]_{\alpha\beta}^{c'c} = \epsilon^{abc}\epsilon^{a'b'c'} * G_{\rho\alpha}^{(1),aa'} * G_{\beta\rho}^{(2),bb'},$$

  - `quarkContract23(a,b)`

$$\mathrm{qC}_{23}\left[G^{(1)}, G^{(2)}\right]_{\alpha\beta}^{c'c} = \epsilon^{abc}\epsilon^{a'b'c'} * G_{\alpha\rho}^{(1),aa'} * G_{\rho\beta}^{(2),bb'},$$

  - `quarkContract24(a,b)`

$$\mathrm{qC}_{24}\left[G^{(1)}, G^{(2)}\right]_{\alpha\beta}^{c'c} = \epsilon^{abc}\epsilon^{a'b'c'} * G_{\alpha\rho}^{(1),aa'} * G_{\beta\rho}^{(2),bb'},$$

- `quarkContract12(a,b)`

$$\mathrm{qC}_{12}\left[G^{(1)}, G^{(2)}\right]^{c'c}_{\alpha\beta} = \epsilon^{abc}\epsilon^{a'b'c'} * G^{(1),aa'}_{\rho\rho} * G^{(2),bb'}_{\alpha\beta},$$

- `quarkContract34(a,b)`

$$\mathrm{qC}_{34}\left[G^{(1)}, G^{(2)}\right]^{c'c}_{\alpha\beta} = \epsilon^{abc}\epsilon^{a'b'c'} * G^{(1),aa'}_{\alpha\beta} * G^{(2),bb'}_{\rho\rho}.$$

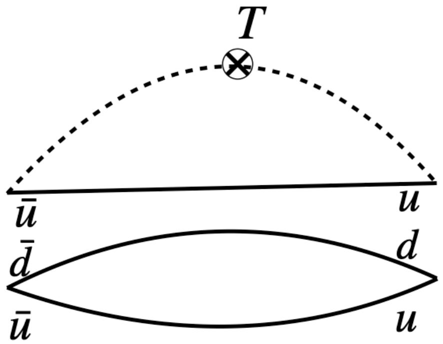
- `colorContract(a,b,c)`
  Epsilon contract 3 color primitives and return a primitive scalar. The sources and targets must all be of the same primitive type (a matrix or vector) but not necessarily of the same lattice type. In explicit index form, the operation colorContract does

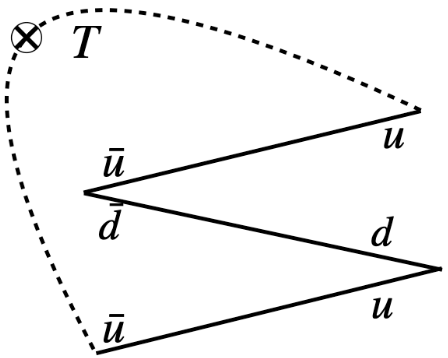
$$\mathrm{cC}\left[G^{(1)}, G^{(2)}, G^{(3)}\right] = \epsilon^{abc}\epsilon^{a'b'c'} * G^{(1),aa'} * G^{(2),bb'} * G^{(3),cc'}$$

  or

$$\mathrm{cC}\left[G^{(1)}, G^{(2)}, G^{(3)}\right] = \epsilon^{abc} * G^{(1),a} * G^{(2),b} * G^{(3),c}.$$

- **Proton 2pt**




$$
\begin{aligned}
\left\langle \chi^p_\gamma(x)\bar\chi^p_{\gamma'}(0)\right\rangle &= -\epsilon^{abc}\epsilon^{a'b'c'}\left\langle \psi^{u,a}_\alpha(x)\left(C\gamma^5\right)_{\alpha\beta}\psi^{d,b}_\beta(x)\psi^{u,c}_\gamma(x)\bar\psi^{u,a'}_{\alpha'}(0)\left(C\gamma^5\right)_{\alpha'\beta'}\bar\psi^{d,b'}_{\beta'}(0)\bar\psi^{u,c'}_{\gamma'}(0)\right\rangle \\
&= -\epsilon^{abc}\epsilon^{a'b'c'}\left\langle \psi^{u,a}_\alpha(x)\left(C\gamma^5\right)_{\alpha\beta}\psi^{d,b}_\beta(x)\psi^{u,c}_\gamma(x)\bar\psi^{u,a'}_{\alpha'}(0)\left(C\gamma^5\right)_{\alpha'\beta'}\bar\psi^{d,b'}_{\beta'}(0)\bar\psi^{u,c'}_{\gamma'}(0)\right\rangle \\
&\quad -\epsilon^{abc}\epsilon^{a'b'c'}\left\langle \psi^{u,a}_\alpha(x)\left(C\gamma^5\right)_{\alpha\beta}\psi^{d,b}_\beta(x)\psi^{u,c}_\gamma(x)\bar\psi^{u,a'}_{\alpha'}(0)\left(C\gamma^5\right)_{\alpha'\beta'}\bar\psi^{d,b'}_{\beta'}(0)\bar\psi^{u,c'}_{\gamma'}(0)\right\rangle \\
&= \epsilon^{abc}\epsilon^{a'b'c'}\left\langle G^{u,aa'}_{\alpha\alpha'}(x,0)\left(C\gamma^5\right)_{\alpha\beta}G^{d,bb'}_{\beta\beta'}(x,0)G^{u,cc'}_{\gamma\gamma'}(x,0)\left(C\gamma^5\right)_{\alpha'\beta'}\right\rangle \\
&\quad -\epsilon^{abc}\epsilon^{a'b'c'}\left\langle G^{u,ac'}_{\alpha\gamma'}(x,0)\left(C\gamma^5\right)_{\alpha\beta}G^{d,bb'}_{\beta\beta'}(x,0)G^{u,ca'}_{\gamma\alpha'}(x,0)\left(C\gamma^5\right)_{\alpha'\beta'}\right\rangle \\
&= \epsilon^{abc}\epsilon^{a'b'c'}\left\langle \left(G^u(x,0)C\gamma^5\right)^{aa'}_{\alpha\beta'}\left(C\gamma^5G^d(x,0)\right)^{bb'}_{\alpha\beta'}G^{u,cc'}_{\gamma\gamma'}(x,0)\right\rangle \\
&\quad -\epsilon^{abc}\epsilon^{a'b'c'}\left\langle G^{u,ac'}_{\alpha\gamma'}(x,0)\left(C\gamma^5G^d(x,0)\right)^{bb'}_{\alpha\beta'}\left(G^u(x,0)C\gamma^5\right)^{ca'}_{\gamma\beta'}\right\rangle \\
&= \mathrm{Tr}_s\left[\mathrm{qC}_{13}\left[G^u(x,0)C\gamma^5, C\gamma^5G^d(x,0)\right]\right]^{c'c}G^{u,cc'}_{\gamma\gamma'}(x,0) \\
&\quad +\epsilon^{bca}\epsilon^{b'a'c'}\left\langle G^{u,ac'}_{\alpha\gamma'}(x,0)\left(C\gamma^5G^d(x,0)\right)^{bb'}_{\alpha\beta'}\left(G^u(x,0)C\gamma^5\right)^{ca'}_{\gamma\beta'}\right\rangle \\
&= \mathrm{Tr}_c\left[G^u(x,0)*\mathrm{Tr}_s\left[\mathrm{qC}_{13}\left[G^u(x,0)C\gamma^5, C\gamma^5G^d(x,0)\right]\right]\right]_{\gamma\gamma'} \\
&\quad +\left\langle G^{u,ac'}_{\alpha\gamma'}(x,0)*\mathrm{qC}_{24}\left[C\gamma^5G^d(x,0), G^u(x,0)C\gamma^5\right]^{c'a}_{\alpha\gamma}\right\rangle \\
&= \mathrm{Tr}_c\left[G^u(x,0)*\mathrm{Tr}_s\left[\mathrm{qC}_{13}\left[G^u(x,0)C\gamma^5, C\gamma^5G^d(x,0)\right]\right]\right]_{\gamma\gamma'} \\
&\quad +\left\langle G^{u,ac'}_{\alpha\gamma'}(x,0)*\mathrm{transposeSpin}\left[\mathrm{qC}_{24}\left[C\gamma^5G^d(x,0), G^u(x,0)C\gamma^5\right]^{c'a}_{\gamma\alpha}\right]\right\rangle \\
&= \mathrm{Tr}_c\left[G^u(x,0)*\mathrm{Tr}_s\left[\mathrm{qC}_{13}\left[G^u(x,0)C\gamma^5, C\gamma^5G^d(x,0)\right]\right]\right]_{\gamma\gamma'} \\
&\quad +\mathrm{Tr}_c\left[\mathrm{transposeSpin}\left[\mathrm{qC}_{24}\left[C\gamma^5G^d(x,0), G^u(x,0)C\gamma^5\right]\right]*G^u(x,0)\right]_{\gamma\gamma'},
\end{aligned}
$$