A method for gauge transforming quark propagators on the lattice ("regauging")

Vincent Morénas

Laboratoire de Physique Corpusculaire
Université Blaise Pascal - IN2P3

Autrans ETMC Meeting - March 2009

Introduction

What we want

convert quark propagators calculated in one gauge into another gauge (for instance : the Landau gauge)

Ingredients

- link variables in both gauges
- propagator in the original gauge

Roadmap

1. look for the gauge transformation function $g(x)$ at each point of the lattice
2. do the propagator conversion

Outline

1. Determination of the gauge transformation function $g(x)$
2. Quark propagator conversion
3. A few benchmarks

Property of the gauge transformation function

$$U^{\text{out}}(y,x) = g^{\dagger}(y)\cdot U^{\text{in}}(y,x)\cdot g(x)$$

$$\Longleftrightarrow \quad g(y) = U^{\text{in}}(y,x)\cdot g(x)\cdot U^{\text{out}\,\dagger}(y,x)$$

where :

- in (out) represents the original (final) gauge
- U can be the link variable, or some curve in space-time going from x to y

⇒ if we know $g(x)$ at some lattice point x_o, we can get $g(x)$ for the whole lattice

⇒ First secondary problem : $g(x_o)$?

A new object

We need something gauge dependent whose transformation law involves one lattice point only...

For instance : a Wilson loop

$$W(x_o) \;=\; U_\mu(x_o)\, U_\nu(x_o + a\hat{\mu})\, U_\mu^\dagger(x_o + a\hat{\mu} + a\hat{\nu})\, U_\nu^\dagger(x_o + a\hat{\nu})$$

which statisfies

$$\boxed{W^{\text{out}}(x_o) \;=\; g^\dagger(x_o) \cdot W^{\text{in}}(x_o) \cdot g(x_o)}$$

Notes :

- no trace will be taken when using $W(x)$
- the choice of the 2D lattice slice for $W(x_o)$ is arbitrary
- matrices involved are $SU(3)$ matrices

First secondary problem

1. We choose a point x_o.
2. We calculate W^{in} and W^{out} at this lattice point.
3. We diagonalize the $SU(3)$ matrices W^{in} and W^{out} :

$$
\begin{aligned}
W^{\text{in}} &= M_{\text{in}}^{-1} \cdot D^{\text{in}} \cdot M_{\text{in}} \\
W^{\text{out}} &= M_{\text{out}}^{-1} \cdot D^{\text{out}} \cdot M_{\text{out}}
\end{aligned}
$$

where the | D's are $SU(3)$ diagonal matrices
| M's are $SU(3)$ matrices

First secondary problem

4. Using "$W^{\text{out}} = g^\dagger(x_o) \cdot W^{\text{in}} \cdot g(x_o)$", we obtain :

$$\begin{cases} D^{\text{out}} = \mathscr{P}^{-1} \cdot D^{\text{in}} \cdot \mathscr{P} \\ \quad \text{where} \quad \mathscr{P} = M_{\text{in}} \cdot g(x_o) \cdot M_{\text{out}}^{-1} \end{cases}$$

Consequence

If we can somehow determine $\mathscr{P}$, then we can get :

$$\boxed{g(x_o) = M_{\text{in}}^{-1} \cdot \mathscr{P} \cdot M_{\text{out}}}$$

$\Rightarrow$ Second secondary problem : $\mathscr{P}$?

Second secondary problem

Important result :

because

$$D^{\text{out}} = \mathscr{P}^{-1} \cdot D^{\text{in}} \cdot \mathscr{P}$$

then for $SU(3)$ matrices, if (for instance) D^{in} has three different non-vanishing diagonal terms, then :

- ⇝ $D^{\text{out}} = D^{\text{in}}$
- ⇝ $\mathscr{P}$ is a $SU(3)$ matrix with the following structure :

$$\mathscr{P}(\alpha, \beta) = \begin{pmatrix} e^{i\alpha} & 0 & 0 \\ 0 & e^{i\beta} & 0 \\ 0 & 0 & e^{-i(\alpha+\beta)} \end{pmatrix}$$

(It is always possible to change the lattice slice or the lattice point if we get a D^{in} which does not fulfill this requirement...)

Second secondary problem

Quick summary

We now know : M_{in}, M_{out} and the structure of $\mathscr{P}(\alpha, \beta)$ so that :

$$g(x_o) = M_{\text{in}}^{-1} \cdot \mathscr{P}(\alpha, \beta) \cdot M_{\text{out}}$$

$\Rightarrow$ we need to find α and β numerically

Second secondary problem : α and β

Method :

- We use 2 different slices at x_o and calculate :

$$\begin{cases} g^{(1)}(x_o,\,\alpha_1,\,\beta_1) = \; {M_{\text{in}}^{(1)}}^{-1} \cdot \mathscr{P}(\alpha_1,\,\beta_1) \cdot M_{\text{out}}^{(1)} \\ g^{(2)}(x_o,\,\alpha_2,\,\beta_2) = \; {M_{\text{in}}^{(2)}}^{-1} \cdot \mathscr{P}(\alpha_2,\,\beta_2) \cdot M_{\text{out}}^{(2)} \end{cases}$$

Of course : $g^{(1)}(x_o,\,\alpha_1,\,\beta_1) \; = \; g^{(2)}(x_o,\,\alpha_2,\,\beta_2)$

- We define the real scalar quantity :

$$S(\alpha_1,\,\beta_1,\,\alpha_2,\,\beta_2) \; = \sum_{i,j} |g_{ij}^{(1)}(x_o,\,\alpha_1,\,\beta_1) - g_{ij}^{(2)}(x_o,\,\alpha_2,\,\beta_2)|^2$$

- We look for $(\alpha_1,\,\beta_1)$ (or $(\alpha_2,\,\beta_2)$) that minimizes S

Finally

Gauge transformation function at x_o

$g(x_o)$ can now be numerically calculated using :

$$g(x_o) \;=\; M_{\text{in}}^{-1}(x_o) \cdot \mathscr{P}(\alpha,\, \beta) \cdot M_{\text{out}}(x_o)$$

Gauge transformation function at x

We can reach all the lattice sites by applying repeatedly :

$$U^{\text{out}}(y,\, x) \;=\; g^{\dagger}(y)\, U^{\text{in}}(y,\, x)\, g(x)$$

Remark

This method gives the "true" gauge transformation function up to a **global** phase factor $\exp(in\varphi)$ with $\varphi = 2\pi/3$ (center of $SU(3)$) :

$$\forall x, \quad g_{\text{calc}}(x) \;=\; e^{i\,n\,\varphi} \cdot g_{\text{true}}(x)$$

We will see that this global factor is harmless for the propagator conversion.

Description

Method

We use the gauge transformation property of the propagators :

$$\text{Prop}^{\text{out}}(y, x) \;=\; g^{\dagger}(y) \cdot \text{Prop}^{\text{in}}(y, x) \cdot g(x)$$

Global phase factor

It is clear that the unknown global phase factor of g_{calc} cancels itself out in that relation....

Just to give an idea... as a conclusion

In Orsay (typical PC cluster) :

$24^3 \times 48$ lattice | $g(x)$ calculation $\sim 30\,\text{min}$ (could be improved)
| propagator conversion $\sim 10\,\text{min}$

to be compared with the time to do a complete calculation of a propagator from scratch.

Tricky part : managing huge arrays