Propagator generation with Chroma+QUDA for various fermion actions

2021.07.28

Zhang Kuan

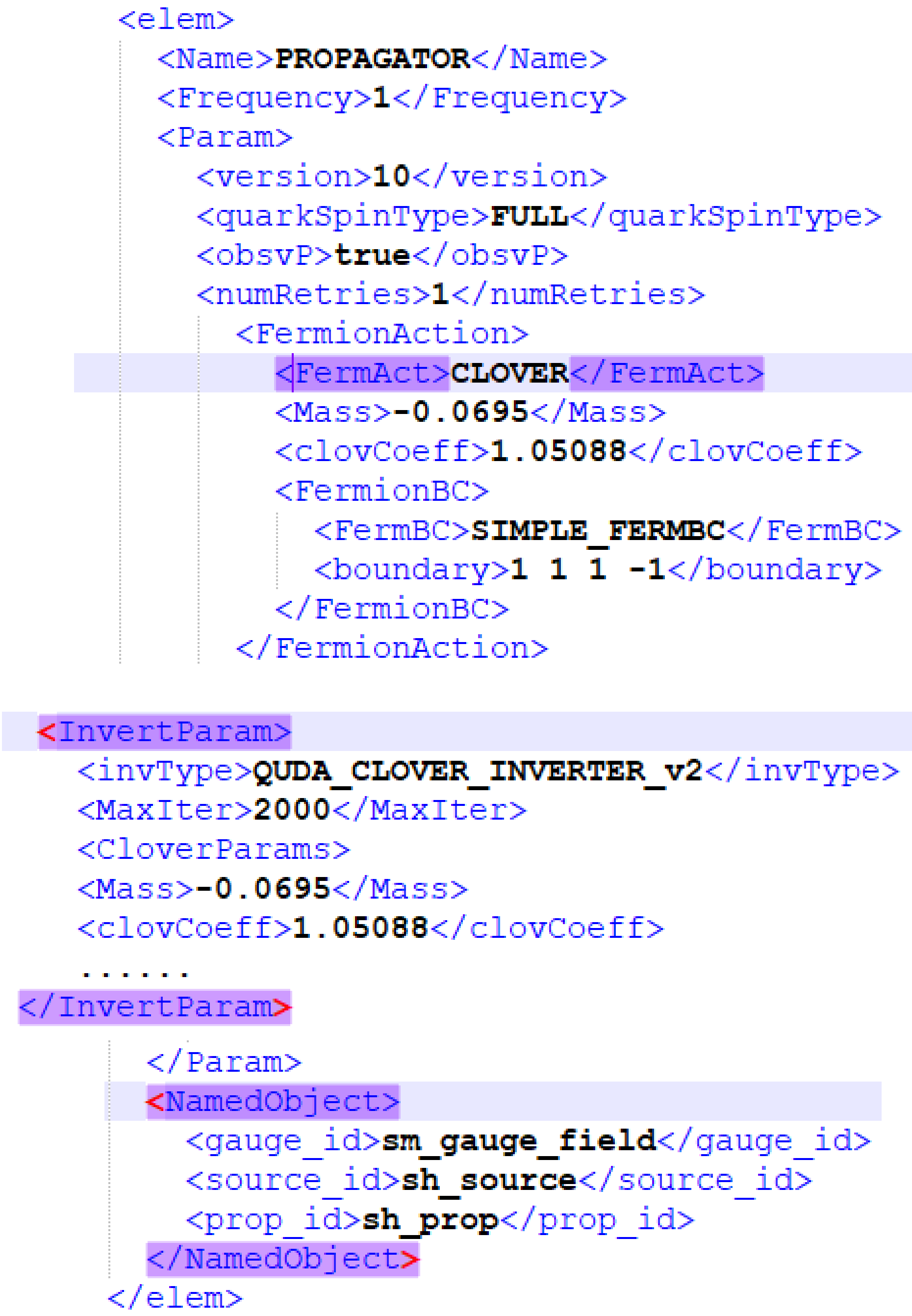
Collaborate with **Yang Yibo** ···

Chroma+QUDA

- Chroma from Jefferson Lab
  efficient, flexible, user friendly

- accelerate D-slash on GPU

- QUDA is a library for performing calculations in lattice QCD on GPUs

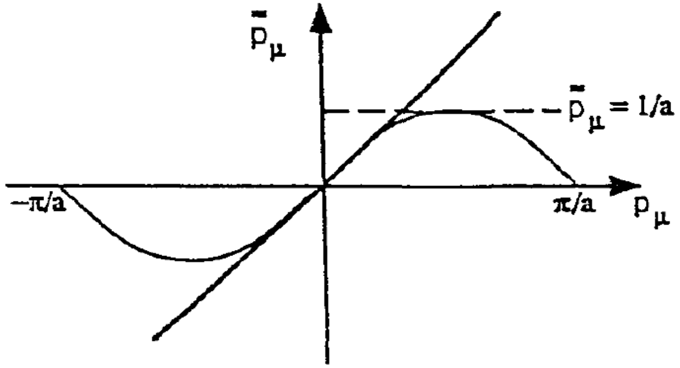


```xml
<elem>
  <Name>PROPAGATOR</Name>
  <Frequency>1</Frequency>
  <Param>
    <version>10</version>
    <quarkSpinType>FULL</quarkSpinType>
    <obsvP>true</obsvP>
    <numRetries>1</numRetries>
      <FermionAction>
        <FermAct>CLOVER</FermAct>
        <Mass>-0.0695</Mass>
        <clovCoeff>1.05088</clovCoeff>
        <FermionBC>
          <FermBC>SIMPLE_FERMBC</FermBC>
          <boundary>1 1 1 -1</boundary>
        </FermionBC>
      </FermionAction>

<InvertParam>
  <invType>QUDA_CLOVER_INVERTER_v2</invType>
  <MaxIter>2000</MaxIter>
  <CloverParams>
  <Mass>-0.0695</Mass>
  <clovCoeff>1.05088</clovCoeff>
  ......
</InvertParam>

      </Param>
      <NamedObject>
        <gauge_id>sm_gauge_field</gauge_id>
        <source_id>sh_source</source_id>
        <prop_id>sh_prop</prop_id>
      </NamedObject>
    </elem>
```

Fermion action on lattice



- naïve fermion $S = \sum_x \overline{\psi_x}\left[\frac{1}{2}(\partial_\mu + \partial_\mu^*)\gamma^\mu + m_0\right]\psi_x$ fermion doubling!
- Wilson-type fermion
  - Wilson fermion √ $S = \sum_x \overline{\psi_x}\left[\frac{1}{2}(\partial_\mu + \partial_\mu^*)\gamma^\mu - \frac{1}{2}(\partial^\mu\partial_\mu^*) + m_0\right]\psi_x$
  - clover fermion √
  - twist mass fermion
- overlap fermion
- HISQ(highly improved staggered quark) fermion
- domain wall fermion

Twist mass fermion

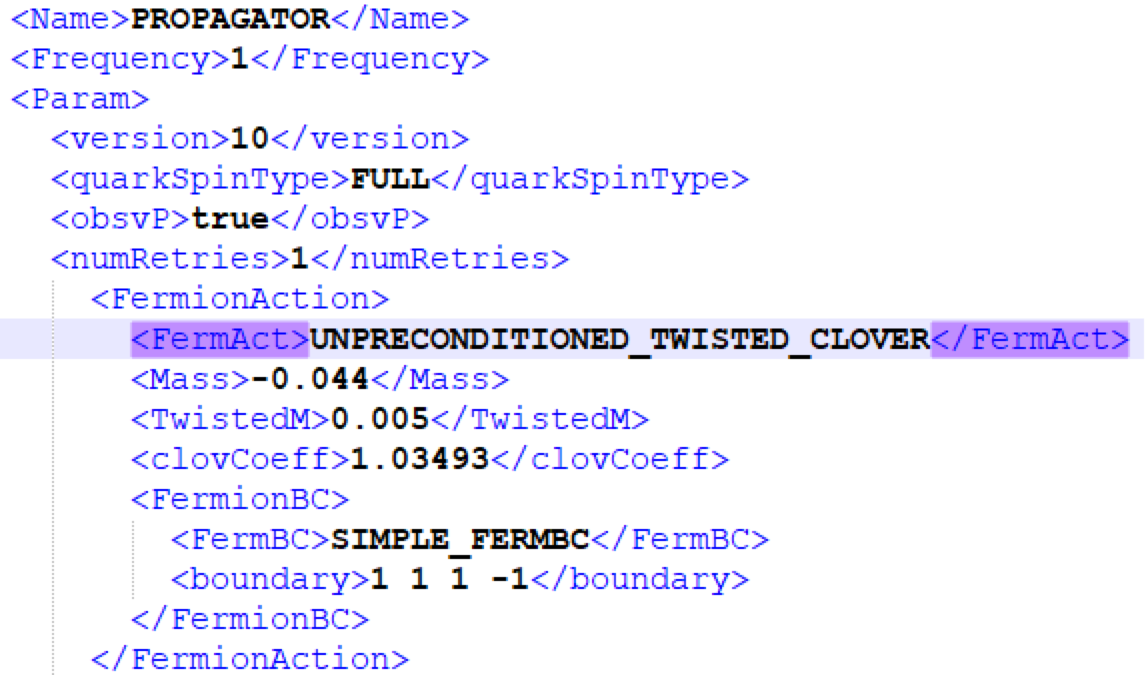
$$\chi(x) = e^{-\frac{i}{2}\omega\gamma_5\tau_3}\psi(x)$$

$$\bar{\chi}(x) = \bar{\psi}(x)e^{-\frac{i}{2}\omega\gamma_5\tau^3}$$

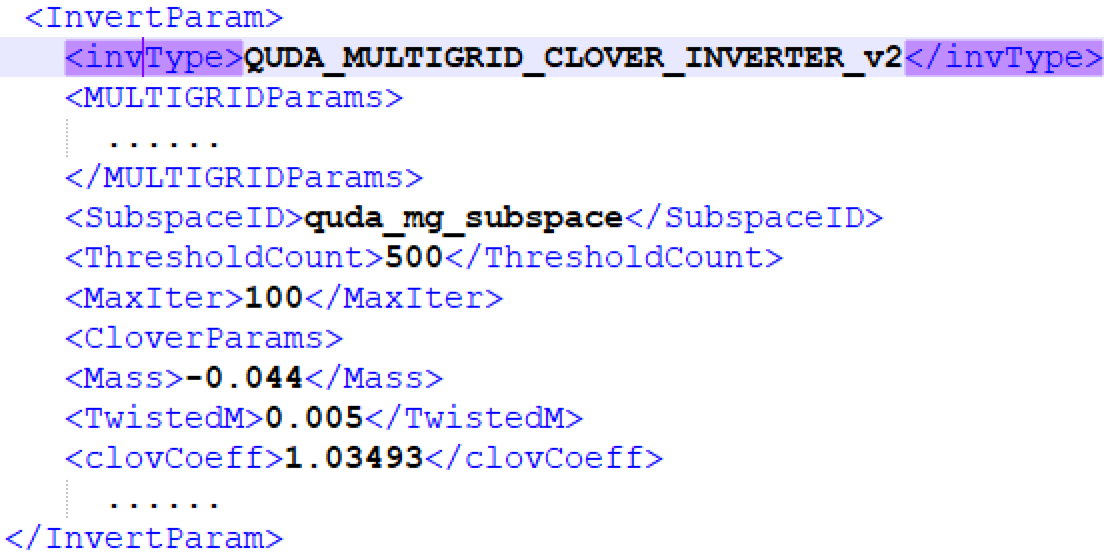
$$S_F = \int d^4x\bar{\chi}(x)[\gamma_\mu D_\mu + m + i\mu\gamma_5\tau^3]\chi(x)$$

$$Me^{i\alpha\gamma_5\tau^3} = m + i\mu\gamma_5\tau^3$$

- avoid the exceptional condition due to the unstablity of the critical point
- Automatic O(a) improvement if $\omega = \frac{\pi}{2}$ $\alpha \to \omega$

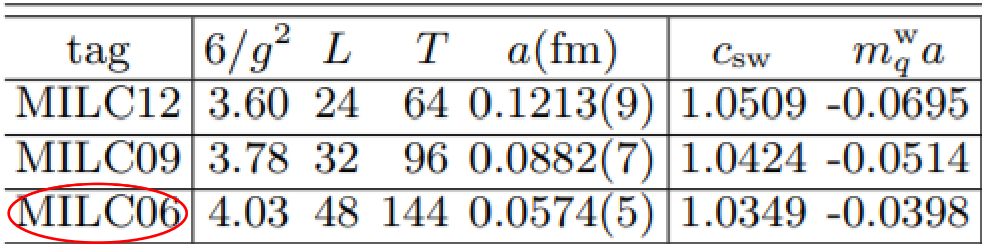


```
<Name>PROPAGATOR</Name>
<Frequency>1</Frequency>
<Param>
  <version>10</version>
  <quarkSpinType>FULL</quarkSpinType>
  <obsvP>true</obsvP>
  <numRetries>1</numRetries>
    <FermionAction>
      <FermAct>UNPRECONDITIONED_TWISTED_CLOVER</FermAct>
      <Mass>-0.044</Mass>
      <TwistedM>0.005</TwistedM>
      <clovCoeff>1.03493</clovCoeff>
      <FermionBC>
        <FermBC>SIMPLE_FERMBC</FermBC>
        <boundary>1 1 1 -1</boundary>
      </FermionBC>
    </FermionAction>
```



```
 <InvertParam>
  <invType>QUDA_MULTIGRID_CLOVER_INVERTER_v2</invType>
  <MULTIGRIDParams>
    ......
  </MULTIGRIDParams>
  <SubspaceID>quda_mg_subspace</SubspaceID>
  <ThresholdCount>500</ThresholdCount>
  <MaxIter>100</MaxIter>
  <CloverParams>
  <Mass>-0.044</Mass>
  <TwistedM>0.005</TwistedM>
  <clovCoeff>1.03493</clovCoeff>
    ......
</InvertParam>
```

acceleration

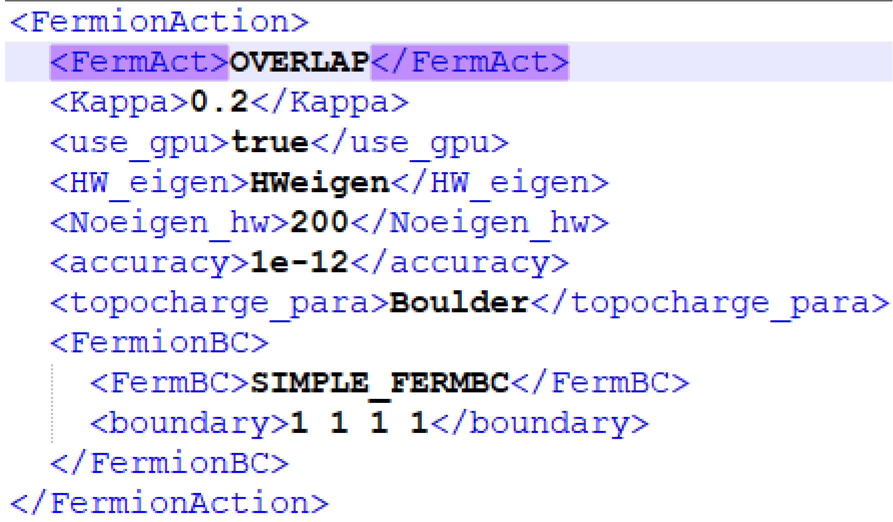


tag	$6/g^2$	L	T	$a(\mathrm{fm})$	c_{sw}	$m_q^{\mathrm{w}}a$
MILC12	3.60	24	64	0.1213(9)	1.0509	-0.0695
MILC09	3.78	32	96	0.0882(7)	1.0424	-0.0514
MILC06	4.03	48	144	0.0574(5)	1.0349	-0.0398

$m_\pi \sim$ 300MeV

residual = 2e-6

	ensembles	mass	processes	time(subspace)
CPU	MILC06	-0.0398	36	6992s
GPU	MILC06	-0.0398	36	652s
GPU with multi-grid	MILC06	-0.0398	36	99s(515s)
GPU with multi-grid	MILC06	-0.044+0.005i	36	78s(154s)



```
<FermionAction>
  <FermAct>OVERLAP</FermAct>
  <Kappa>0.2</Kappa>
  <use_gpu>true</use_gpu>
  <HW_eigen>HWeigen</HW_eigen>
  <Noeigen_hw>200</Noeigen_hw>
  <accuracy>1e-12</accuracy>
  <topocharge_para>Boulder</topocharge_para>
  <FermionBC>
    <FermBC>SIMPLE_FERMBC</FermBC>
    <boundary>1 1 1 1</boundary>
  </FermionBC>
</FermionAction>
```

Overlap fermion action

- Ginsparg-Wilson relation $\{\gamma_5, D\} = D\gamma_5 D$

- Overlap $S_q^{\text{ov}} = \sum_{x,y} \bar{\psi}(x)\,\rho D_{\text{ov}}(x,y)\psi(y)$ $\quad D_{ov}(\rho) = 1 + \gamma_5 \varepsilon(\gamma_5 D_{\text{w}}(\rho))$

$$\frac{1}{D_c + m_q} = \frac{1}{\frac{D_{\text{ov}}}{1-\frac{1}{2\rho}D_{\text{ov}}} + m_q} = \frac{1 - \frac{1}{2\rho}D_{\text{ov}}}{D_{\text{ov}} + m(1 - \frac{1}{2\rho}D_{\text{ov}})} \qquad \rho = 1.5/a$$

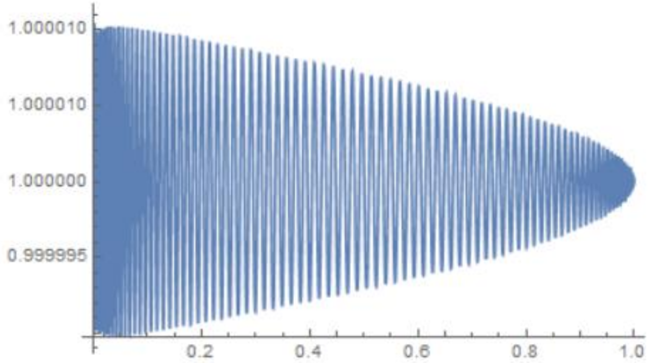
- Sign function polynomial $\varepsilon(\gamma_5 D_{\text{w}}(\rho))$ ~ polynomial of Hwilson D-slash

$$D = V\,\Lambda\,V^{-1} \qquad D^n = V\,\Lambda^n\,V^{-1} \qquad f(D) = V\,f(\Lambda)\,V^{-1}$$

$$D(\psi_1, \psi_2, \cdots, \psi_k) = (c_1\psi_1, c_2\psi_2, \cdots, c_{\text{k}}\psi_k) = (\psi_1, \psi_2, \cdots, \psi_k)\begin{pmatrix} c_1 & & & \\ & c_2 & & \\ & & \ddots & \\ & & & c_{\text{k}} \end{pmatrix} \Rightarrow DV = V\Lambda \Rightarrow D = V\,\Lambda\,V^{-1}$$

Chebyshev polynomial

$$\varepsilon(\gamma_5 D_{\mathrm{w}}(\rho))$$



- Definition converge faster

$$h(y) = 1 - \sqrt{y}p(y)$$

$$p(y) = \sum_i c_i T_i(z), z = (2y - 1 - \epsilon)/(1 - \epsilon)$$

$$\epsilon \leq y \leq 1$$

$$T_0(x) = 1$$

$$T_1(x) = x$$

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x).$$

- Chebyshev coefficients

- ε :chebyshev cut

- Hwilson low mode

Chebyshev coefficients

$$p(y) = \sum_i^n c_i T_i(z), z = (2y - 1 - \epsilon)/(1 - \epsilon)$$

$$\epsilon \leq y \leq 1$$

$$h(y) = 1 - \sqrt{y}p(y)$$

- Estimate the ranks of polynomial: n according to ε, δ

- Solve the equations $\sqrt{y_i}\sum_j c_j T_j(z_i) = 1$

$z = \cos(\pi\frac{i}{n})$ initial value

- Use extreme points as the new y(z)

meet the tolerance δ

- Clenshaw recursion less round-off error

Hwilson Eigen solver

- Chebyshev acceleration → reduce high mode

  $$f(D) = V\, f(\Lambda)\, V^{-1}$$

- Arnoldi factorization:
  Krylov array $\quad A\mathbf{x} = \mathbf{b} \qquad \{\mathbf{b}, A\mathbf{b}, ..., A^{k-1}\mathbf{b}\}$

  Schmidt orthogonalization

  $$A\mathbf{q}_l = \begin{pmatrix} \mathbf{q}_1 & \cdots & \mathbf{q}_l \end{pmatrix} \begin{pmatrix} h_{1l} \\ \vdots \\ h_{ll} \end{pmatrix} \qquad AQ_k = \begin{pmatrix} A\mathbf{q}_1 & A\mathbf{q}_2 & \cdots & A\mathbf{q}_k \end{pmatrix} = \begin{pmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_k \end{pmatrix} \begin{pmatrix} h_{11} & h_{12} & h_{13} & \cdots & h_{1k} \\ h_{12} & h_{22} & h_{23} & \cdots & h_{2k} \\ 0 & h_{32} & h_{33} & \cdots & h_{3k} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & h_{k,k-1} & h_{kk} \end{pmatrix}$$

  Diagonalize Hessenberg matrix $\quad AQ = QT\Lambda\, T^{-1} \qquad A = QT\Lambda(QT)^{-1}$

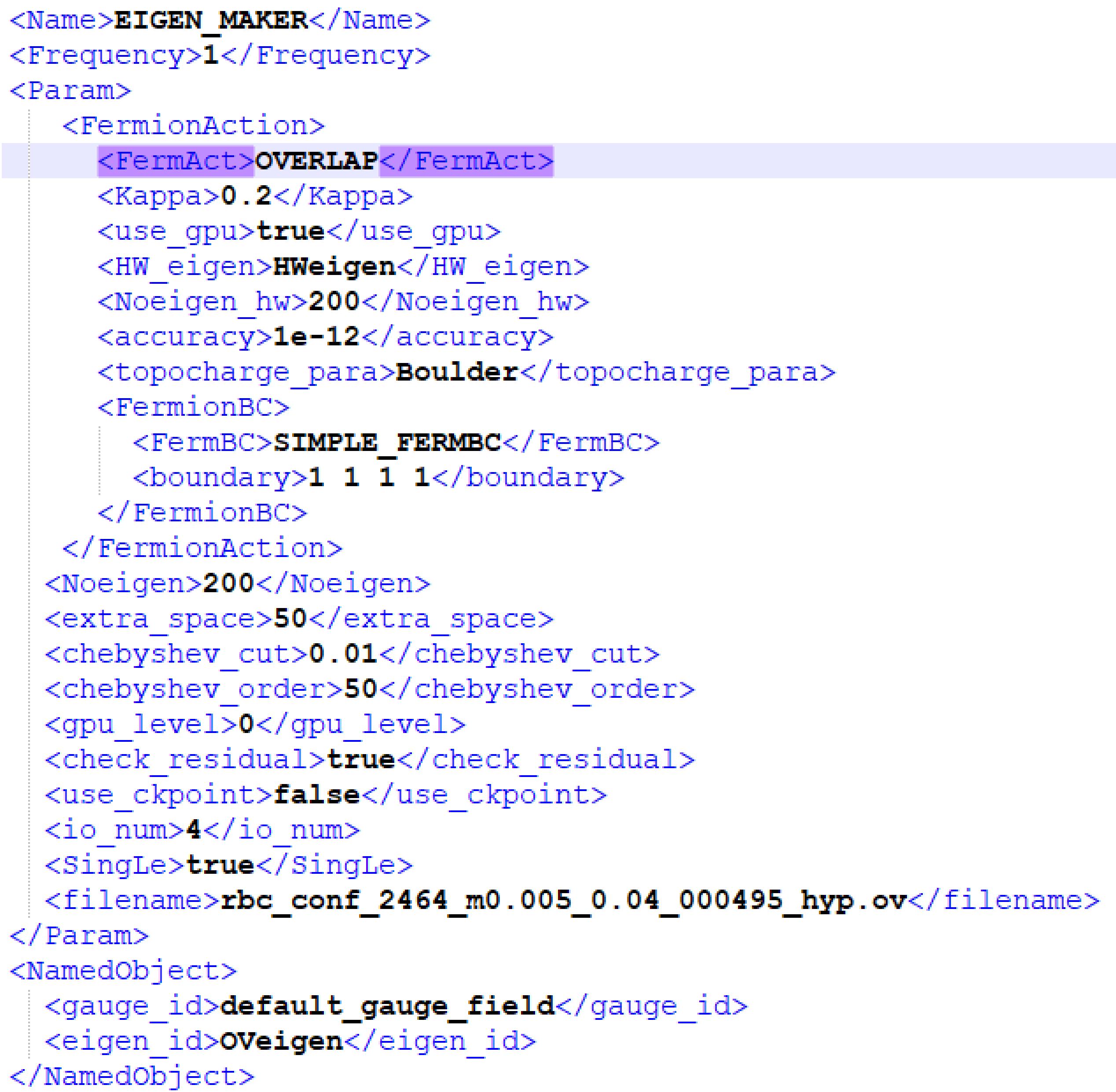
Overlap Eigen solver

- deflation $D = V\,\Lambda\,V^{-1}$ $D^{-1} = V\Lambda^{-1}\,V^{-1}$
  overlap low node

- complex eigenvalue×
  arnoldi needs real !

- half of spinor
  $D\psi = \lambda$ $D\gamma^5\psi = \lambda^*\gamma^5\psi$
  $(1\pm\gamma^5)\,D(1\pm\gamma^5)\psi = \;(\lambda+\lambda^*)(1\pm\gamma^5)\psi$

- reconstruct



```
<Name>EIGEN_MAKER</Name>
<Frequency>1</Frequency>
<Param>
   <FermionAction>
     <FermAct>OVERLAP</FermAct>
     <Kappa>0.2</Kappa>
     <use_gpu>true</use_gpu>
     <HW_eigen>HWeigen</HW_eigen>
     <Noeigen_hw>200</Noeigen_hw>
     <accuracy>1e-12</accuracy>
     <topocharge_para>Boulder</topocharge_para>
     <FermionBC>
       <FermBC>SIMPLE_FERMBC</FermBC>
       <boundary>1 1 1 1</boundary>
     </FermionBC>
   </FermionAction>
  <Noeigen>200</Noeigen>
  <extra_space>50</extra_space>
  <chebyshev_cut>0.01</chebyshev_cut>
  <chebyshev_order>50</chebyshev_order>
  <gpu_level>0</gpu_level>
  <check_residual>true</check_residual>
  <use_ckpoint>false</use_ckpoint>
  <io_num>4</io_num>
  <SingLe>true</SingLe>
  <filename>rbc_conf_2464_m0.005_0.04_000495_hyp.ov</filename>
</Param>
<NamedObject>
  <gauge_id>default_gauge_field</gauge_id>
  <eigen_id>OVeigen</eigen_id>
</NamedObject>
```

Overlap propagator

- deflation: accelerate inversion

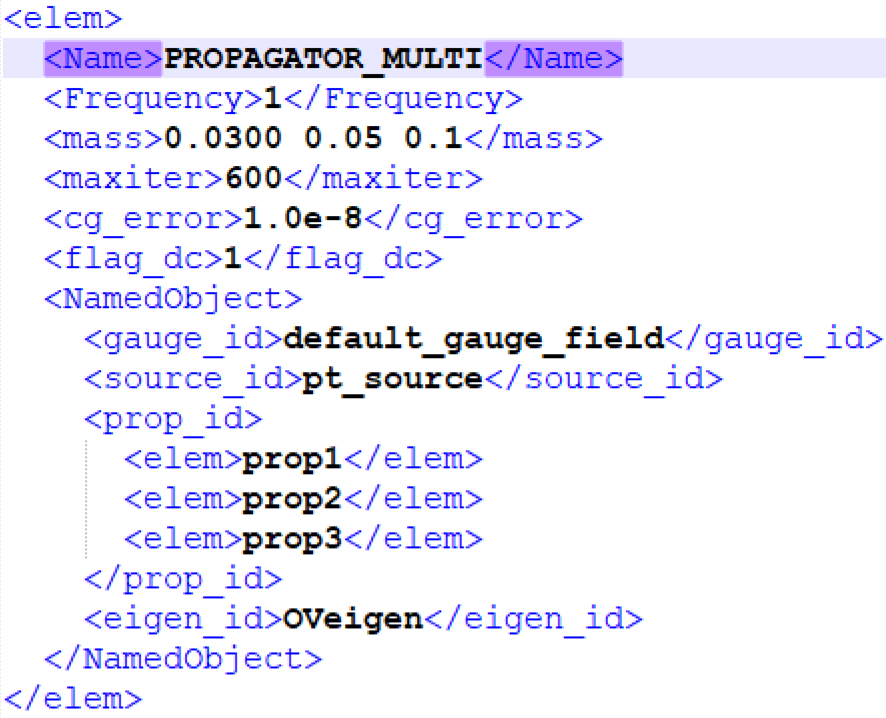
$$D(m)|X^H_{L,R}\rangle = (1 - P_L)|\eta_{L,R}\rangle$$

$$|X^H\rangle = |X^H_L\rangle + |X^H_R\rangle$$

PHYSICAL REVIEW D **82,** 114501 (2010)

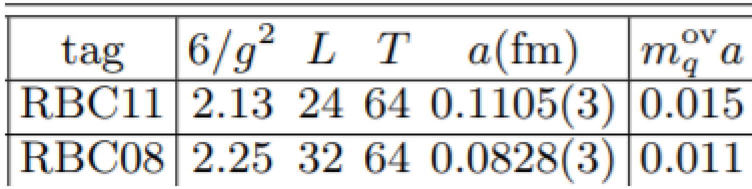
- multi-mass:
  calculate propagators with different mass simultaneously

arXiv:hep-lat/9612014v1



```
<elem>
  <Name>PROPAGATOR_MULTI</Name>
  <Frequency>1</Frequency>
  <mass>0.0300 0.05 0.1</mass>
  <maxiter>600</maxiter>
  <cg_error>1.0e-8</cg_error>
  <flag_dc>1</flag_dc>
  <NamedObject>
    <gauge_id>default_gauge_field</gauge_id>
    <source_id>pt_source</source_id>
    <prop_id>
      <elem>prop1</elem>
      <elem>prop2</elem>
      <elem>prop3</elem>
    </prop_id>
    <eigen_id>OVeigen</eigen_id>
  </NamedObject>
</elem>
```

GPU acceleration



tag	$6/g^2$	L	T	a(fm)	$m_q^{\rm ov}a$
RBC11	2.13	24	64	0.1105(3)	0.015
RBC08	2.25	32	64	0.0828(3)	0.011

m_π ~ 300MeV

$$\rho = 1.5/a$$

- Hwilson eigen system

residual = 1e-15

	ensembles	eigenvectors	processes	time(chroma)
CPU	RBC11	200	4	8070s
GPU	RBC11	200	4	491s
CPU	RBC08	200	16	4704s
GPU	RBC08	200	16	635s

- Overlap eigen system

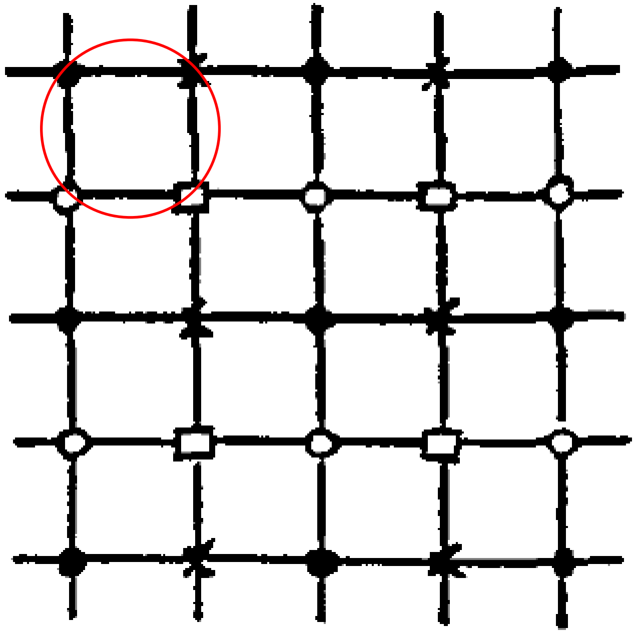
residual = 1e-12

	ensembles	eigenvectors	processes	time(chroma)	time(gwu-code)
CPU	RBC11	200	4	>12h	-
GPU	RBC11	200	4	7600s	16110s
CPU	RBC08	200	16	>12h	-
GPU	RBC08	200	16	10236s	14370s

Intel Xeon Gold 6142

NVIDIA Tesla V100 32GB

Staggered fermion



- Fermion double: 2^4=16

- Staggered transformation: 4

$$\psi'(x) = \gamma_0^{x_0}\gamma_1^{x_1}\gamma_2^{x_2}\gamma_3^{x_3}\psi(x)$$

- Residual 4 freedom

- HISQ(highly improved staggered quark)
  long link   fat link

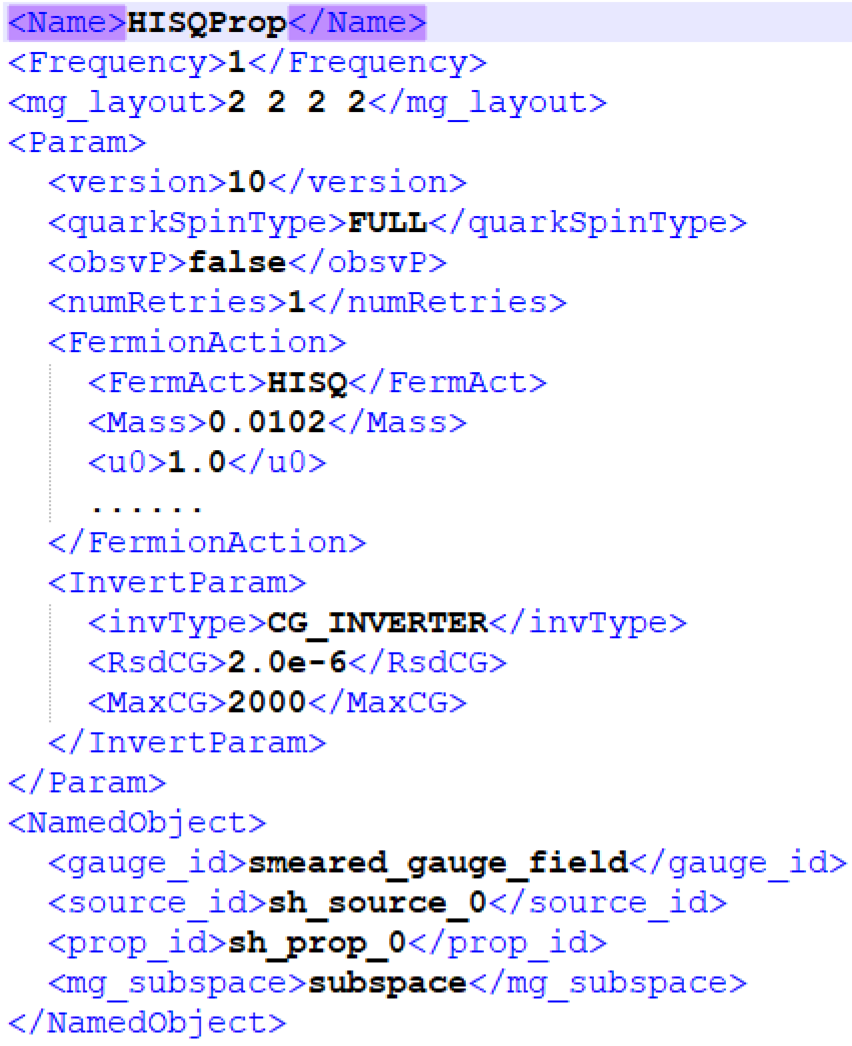
Cheap!

HISQ inverter

- CPU version in Chroma √
- GPU version in QUDA √
- Interface between CHROMA and QUDA

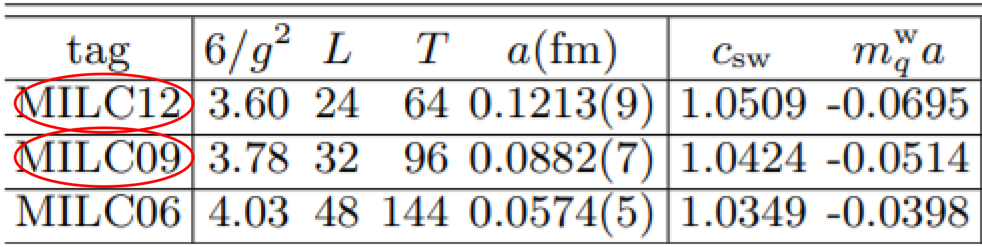
  set parameters for inverter
  calculate&load long&fat link

  invertQuda(***);



```
<Name>HISQProp</Name>
<Frequency>1</Frequency>
<mg_layout>2 2 2 2</mg_layout>
<Param>
  <version>10</version>
  <quarkSpinType>FULL</quarkSpinType>
  <obsvP>false</obsvP>
  <numRetries>1</numRetries>
  <FermionAction>
    <FermAct>HISQ</FermAct>
    <Mass>0.0102</Mass>
    <u0>1.0</u0>
    ......
  </FermionAction>
  <InvertParam>
    <invType>CG_INVERTER</invType>
    <RsdCG>2.0e-6</RsdCG>
    <MaxCG>2000</MaxCG>
  </InvertParam>
</Param>
<NamedObject>
  <gauge_id>smeared_gauge_field</gauge_id>
  <source_id>sh_source_0</source_id>
  <prop_id>sh_prop_0</prop_id>
  <mg_subspace>subspace</mg_subspace>
</NamedObject>
```

GPU acceleration



tag	$6/g^2$	L	T	a(fm)	$c_{\rm sw}$	$m_q^{\rm w}a$
MILC12	3.60	24	64	0.1213(9)	1.0509	-0.0695
MILC09	3.78	32	96	0.0882(7)	1.0424	-0.0514
MILC06	4.03	48	144	0.0574(5)	1.0349	-0.0398

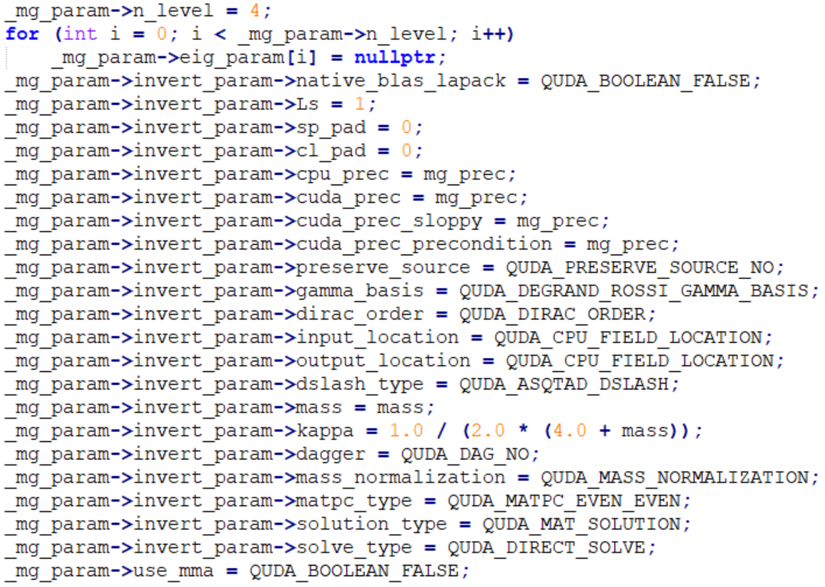
m_π ~ 300MeV

residual = 2e-6

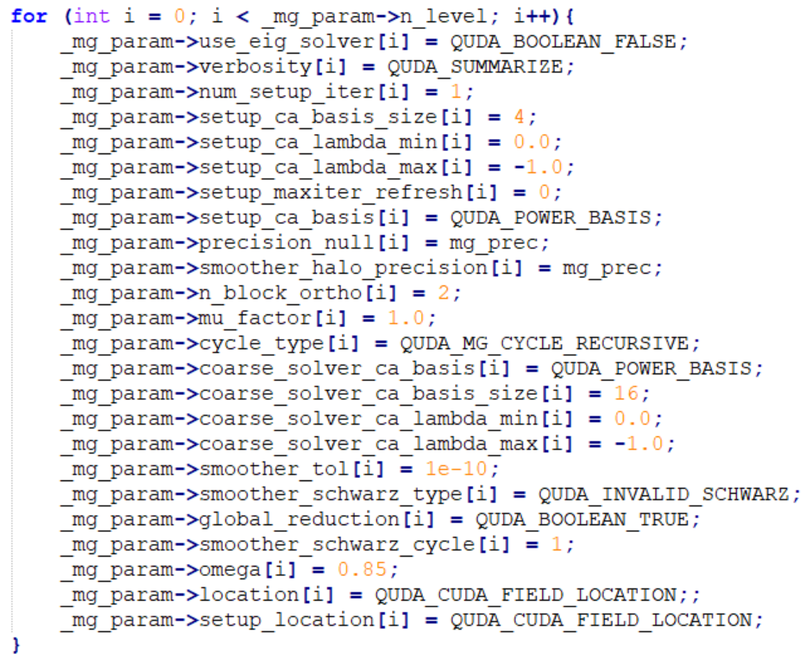
	ensembles	Mass	processes	time(subspace)
CPU	MILC12	0.0102	4	523s
GPU	MILC12	0.0102	4	17s
GPU with multi-grid	MILC12	0.0102	4	26s(334s)
CPU	MILC09	0.0074	12	1086s
GPU	MILC09	0.0074	12	23s
GPU with multi-grid	MILC09	0.0074	12	42s(311s)

Intel Xeon Gold 6142

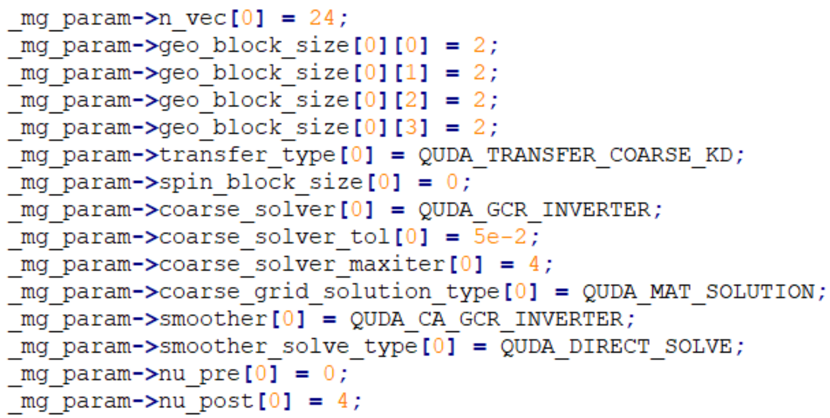
NVIDIA Tesla V100 32GB



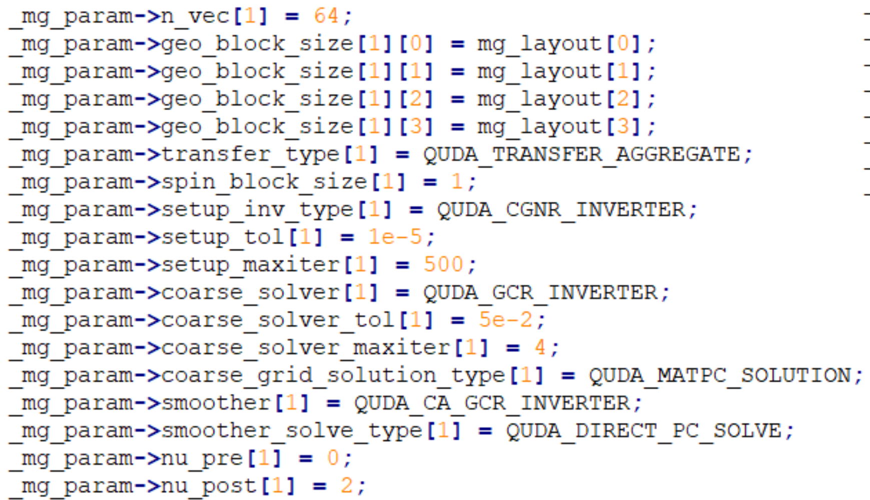
```
_mg_param->n_level = 4;
for (int i = 0; i < _mg_param->n_level; i++)
    _mg_param->eig_param[i] = nullptr;
_mg_param->invert_param->native_blas_lapack = QUDA_BOOLEAN_FALSE;
_mg_param->invert_param->Ls = 1;
_mg_param->invert_param->sp_pad = 0;
_mg_param->invert_param->cl_pad = 0;
_mg_param->invert_param->cpu_prec = mg_prec;
_mg_param->invert_param->cuda_prec = mg_prec;
_mg_param->invert_param->cuda_prec_sloppy = mg_prec;
_mg_param->invert_param->cuda_prec_precondition = mg_prec;
_mg_param->invert_param->preserve_source = QUDA_PRESERVE_SOURCE_NO;
_mg_param->invert_param->gamma_basis = QUDA_DEGRAND_ROSSI_GAMMA_BASIS;
_mg_param->invert_param->dirac_order = QUDA_DIRAC_ORDER;
_mg_param->invert_param->input_location = QUDA_CPU_FIELD_LOCATION;
_mg_param->invert_param->output_location = QUDA_CPU_FIELD_LOCATION;
_mg_param->invert_param->dslash_type = QUDA_ASQTAD_DSLASH;
_mg_param->invert_param->mass = mass;
_mg_param->invert_param->kappa = 1.0 / (2.0 * (4.0 + mass));
_mg_param->invert_param->dagger = QUDA_DAG_NO;
_mg_param->invert_param->mass_normalization = QUDA_MASS_NORMALIZATION;
_mg_param->invert_param->matpc_type = QUDA_MATPC_EVEN_EVEN;
_mg_param->invert_param->solution_type = QUDA_MAT_SOLUTION;
_mg_param->invert_param->solve_type = QUDA_DIRECT_SOLVE;
_mg_param->use_mma = QUDA_BOOLEAN_FALSE;
```



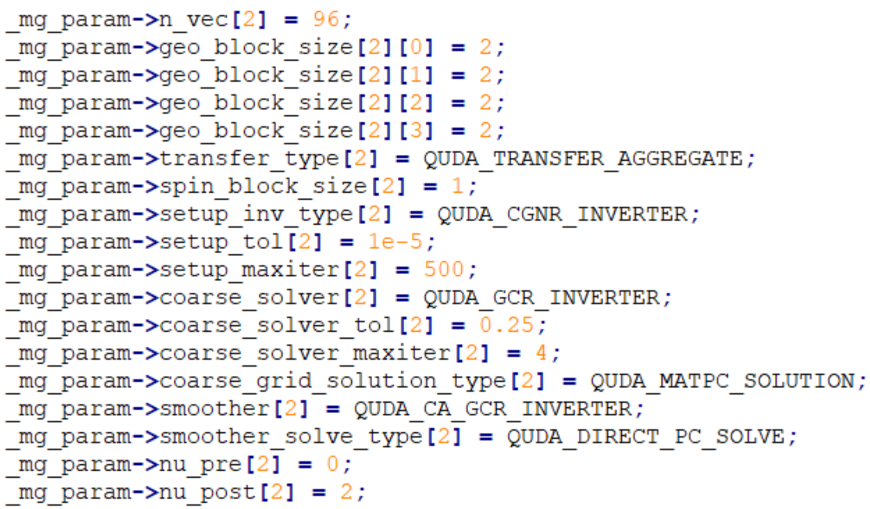
```
for (int i = 0; i < _mg_param->n_level; i++){
    _mg_param->use_eig_solver[i] = QUDA_BOOLEAN_FALSE;
    _mg_param->verbosity[i] = QUDA_SUMMARIZE;
    _mg_param->num_setup_iter[i] = 1;
    _mg_param->setup_ca_basis_size[i] = 4;
    _mg_param->setup_ca_lambda_min[i] = 0.0;
    _mg_param->setup_ca_lambda_max[i] = -1.0;
    _mg_param->setup_maxiter_refresh[i] = 0;
    _mg_param->setup_ca_basis[i] = QUDA_POWER_BASIS;
    _mg_param->precision_null[i] = mg_prec;
    _mg_param->smoother_halo_precision[i] = mg_prec;
    _mg_param->n_block_ortho[i] = 2;
    _mg_param->mu_factor[i] = 1.0;
    _mg_param->cycle_type[i] = QUDA_MG_CYCLE_RECURSIVE;
    _mg_param->coarse_solver_ca_basis[i] = QUDA_POWER_BASIS;
    _mg_param->coarse_solver_ca_basis_size[i] = 16;
    _mg_param->coarse_solver_ca_lambda_min[i] = 0.0;
    _mg_param->coarse_solver_ca_lambda_max[i] = -1.0;
    _mg_param->smoother_tol[i] = 1e-10;
    _mg_param->smoother_schwarz_type[i] = QUDA_INVALID_SCHWARZ;
    _mg_param->global_reduction[i] = QUDA_BOOLEAN_TRUE;
    _mg_param->smoother_schwarz_cycle[i] = 1;
    _mg_param->omega[i] = 0.85;
    _mg_param->location[i] = QUDA_CUDA_FIELD_LOCATION;;
    _mg_param->setup_location[i] = QUDA_CUDA_FIELD_LOCATION;
}
```



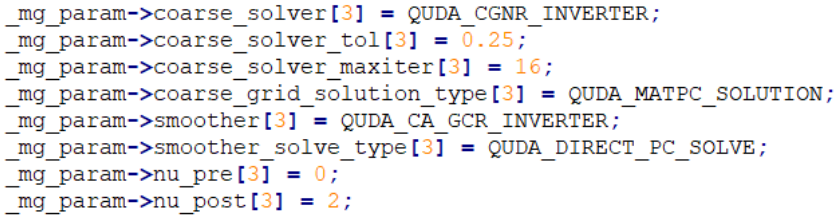
```
_mg_param->n_vec[0] = 24;
_mg_param->geo_block_size[0][0] = 2;
_mg_param->geo_block_size[0][1] = 2;
_mg_param->geo_block_size[0][2] = 2;
_mg_param->geo_block_size[0][3] = 2;
_mg_param->transfer_type[0] = QUDA_TRANSFER_COARSE_KD;
_mg_param->spin_block_size[0] = 0;
_mg_param->coarse_solver[0] = QUDA_GCR_INVERTER;
_mg_param->coarse_solver_tol[0] = 5e-2;
_mg_param->coarse_solver_maxiter[0] = 4;
_mg_param->coarse_grid_solution_type[0] = QUDA_MAT_SOLUTION;
_mg_param->smoother[0] = QUDA_CA_GCR_INVERTER;
_mg_param->smoother_solve_type[0] = QUDA_DIRECT_SOLVE;
_mg_param->nu_pre[0] = 0;
_mg_param->nu_post[0] = 4;
```



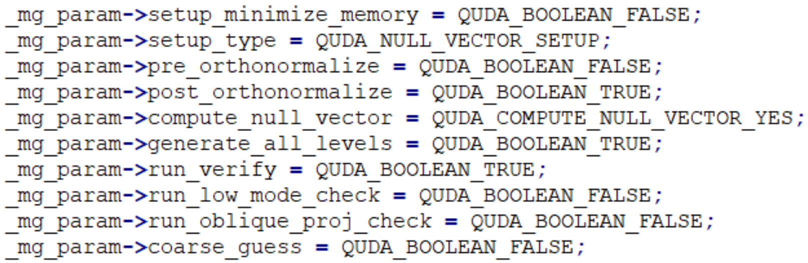
```
_mg_param->n_vec[1] = 64;
_mg_param->geo_block_size[1][0] = mg_layout[0];
_mg_param->geo_block_size[1][1] = mg_layout[1];
_mg_param->geo_block_size[1][2] = mg_layout[2];
_mg_param->geo_block_size[1][3] = mg_layout[3];
_mg_param->transfer_type[1] = QUDA_TRANSFER_AGGREGATE;
_mg_param->spin_block_size[1] = 1;
_mg_param->setup_inv_type[1] = QUDA_CGNR_INVERTER;
_mg_param->setup_tol[1] = 1e-5;
_mg_param->setup_maxiter[1] = 500;
_mg_param->coarse_solver[1] = QUDA_GCR_INVERTER;
_mg_param->coarse_solver_tol[1] = 5e-2;
_mg_param->coarse_solver_maxiter[1] = 4;
_mg_param->coarse_grid_solution_type[1] = QUDA_MATPC_SOLUTION;
_mg_param->smoother[1] = QUDA_CA_GCR_INVERTER;
_mg_param->smoother_solve_type[1] = QUDA_DIRECT_PC_SOLVE;
_mg_param->nu_pre[1] = 0;
_mg_param->nu_post[1] = 2;
```



```
_mg_param->n_vec[2] = 96;
_mg_param->geo_block_size[2][0] = 2;
_mg_param->geo_block_size[2][1] = 2;
_mg_param->geo_block_size[2][2] = 2;
_mg_param->geo_block_size[2][3] = 2;
_mg_param->transfer_type[2] = QUDA_TRANSFER_AGGREGATE;
_mg_param->spin_block_size[2] = 1;
_mg_param->setup_inv_type[2] = QUDA_CGNR_INVERTER;
_mg_param->setup_tol[2] = 1e-5;
_mg_param->setup_maxiter[2] = 500;
_mg_param->coarse_solver[2] = QUDA_GCR_INVERTER;
_mg_param->coarse_solver_tol[2] = 0.25;
_mg_param->coarse_solver_maxiter[2] = 4;
_mg_param->coarse_grid_solution_type[2] = QUDA_MATPC_SOLUTION;
_mg_param->smoother[2] = QUDA_CA_GCR_INVERTER;
_mg_param->smoother_solve_type[2] = QUDA_DIRECT_PC_SOLVE;
_mg_param->nu_pre[2] = 0;
_mg_param->nu_post[2] = 2;
```



```
_mg_param->coarse_solver[3] = QUDA_CGNR_INVERTER;
_mg_param->coarse_solver_tol[3] = 0.25;
_mg_param->coarse_solver_maxiter[3] = 16;
_mg_param->coarse_grid_solution_type[3] = QUDA_MATPC_SOLUTION;
_mg_param->smoother[3] = QUDA_CA_GCR_INVERTER;
_mg_param->smoother_solve_type[3] = QUDA_DIRECT_PC_SOLVE;
_mg_param->nu_pre[3] = 0;
_mg_param->nu_post[3] = 2;
```



```
_mg_param->setup_minimize_memory = QUDA_BOOLEAN_FALSE;
_mg_param->setup_type = QUDA_NULL_VECTOR_SETUP;
_mg_param->pre_orthonormalize = QUDA_BOOLEAN_FALSE;
_mg_param->post_orthonormalize = QUDA_BOOLEAN_TRUE;
_mg_param->compute_null_vector = QUDA_COMPUTE_NULL_VECTOR_YES;
_mg_param->generate_all_levels = QUDA_BOOLEAN_TRUE;
_mg_param->run_verify = QUDA_BOOLEAN_TRUE;
_mg_param->run_low_mode_check = QUDA_BOOLEAN_FALSE;
_mg_param->run_oblique_proj_check = QUDA_BOOLEAN_FALSE;
_mg_param->coarse_guess = QUDA_BOOLEAN_FALSE;
```

My parameters for HISQ multi-grid

Copied from MILC interface

Summary

- We wrote interfaces of twist-mass, overlap, and HISQ fermion actions between Chroma and QUDA.
  The GPU acceleration effect is obvious.

- We need to try better parameters for HISQ multi-grid.

- We plan to write the interface of domain wall fermion in the future.