GPU QMC
OPTIMIZATION

Ming Wong
Tyler McDaniel
Quantum Monte Carlo Simulation

Slater Determinant for N-electrons system

\[ \Psi(x_1, x_2, \ldots, x_N) = \frac{1}{\sqrt{N!}} \begin{vmatrix} \chi_1(x_1) & \chi_2(x_1) & \cdots & \chi_N(x_1) \\ \chi_1(x_2) & \chi_2(x_2) & \cdots & \chi_N(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \chi_1(x_N) & \chi_2(x_N) & \cdots & \chi_N(x_N) \end{vmatrix} \]
What is QMCPACK?

• Software framework for quantum Monte Carlo simulation

• Written in C++ w/Cuda kernels

• Utilizes Cuda (acceleration) and openMP (parallelization)
Current QMC implementation

[while not all change vectors evaluated] matrix $A$, change vectors $U^*V^*$

Factorize $A$
- Find LU s.t. $A = LU$

Inverse $A$, determinant $(A)$, change vector $u^*v^*$

S.M. Formula
- Find $\text{inverse}(A + u^*v^*)$

Determinant Check
- Test: $\text{det}(A + u^*v^*) / \text{det}(A)$

Inverse Check
- $l = A \times \text{inverse}(A)$

Update $A$
- $A = (A + u^*v^*)$

[ratio < 1 and fail random check: evaluate next change]
Current QMPC implementation

[while not all change vectors evaluated] matrix A, change vectors U*v'

Factorize A
Find LU s.t. A = LU

inverse(A), determinant(A), change vector u*v'

S.M. Formula
Find inverse(A + u*v')

determinant(A + u*v')

Determinant Check
Test:
det(A + u*v') / det(A)

Inverse Check
l = A * inverse(A)

Update A
A = (A + u*v')

[ratio < 1 and fail random check: evaluate next change]

[fail]

[pass]

[ratio >= 1 || pass random check]
LU Decomposition

\[
\begin{bmatrix}
a_{11} & a_{12} & a_{13} \\
a_{21} & a_{22} & a_{23} \\
a_{31} & a_{32} & a_{33}
\end{bmatrix} = \begin{bmatrix}
1 & 0 & 0 \\
L_{21} & 1 & 0 \\
L_{31} & L_{32} & 1
\end{bmatrix}\begin{bmatrix}
U_{11} & U_{12} & U_{13} \\
0 & U_{22} & U_{23} \\
0 & 0 & U_{33}
\end{bmatrix}
\]

\[
A = L \ast U
\]
Current QMPC implementation

[while not all change vectors evaluated] matrix A, change vectors \( u^*v' \)

**Factorize A**
- Find LU s.t. \( A = LU \)
- inverse(A), determinant(A), change vector \( u^*v' \)

**S.M. Formula**
- Find inverse(A + \( u^*v' \))
- determinant(A + \( u^*v' \))

**Determinant Check**
- Test: \( \frac{\text{det}(A + u^*v')}{{\text{det}(A)}} \)
- [ratio >= 1 || pass random check]

**Inverse Check**
- \( I = A \times \text{inverse}(A) \)
- [pass]

**Update A**
- \( A = (A + u^*v') \)
Current QMPC implementation

[while not all change vectors evaluated]
matrix A, change vectors $u^*v'$

**Factorize A**
Find LU s.t. $A = LU$

inverses($A$),
determinant($A$),
change vector $u^*v'$

**S.M. Formula**
Find $\text{inverse}(A + u^*v')$

$\text{determinant}(A + u^*v')$

**Determinant Check**
Test:
$\text{det}(A + u^*v') / \text{det}(A)$

[ratio $< 1$ and fail random check: evaluate next change]

[ratio $\geq 1$ || pass random check]

**Inverse Check**
$\text{inv}(A)$

$A = (A + u^*v')$

**Update A**
Current QMPC implementation

[while not all change vectors evaluated]

matrix A, change vectors $u^*v'$

---

Factorize A

Find LU s.t. $A = LU$

---

inverse($A$), determinant($A$), change vector $u^*v'$

---

S.M. Formula

Find inverse($A + u^*v'$)

---

Determinant Check

determinant($A + u^*v'$)

Test:
$$\det(A + u^*v') / \det(A)$$

---

Inverse Check

$I = A \ast \text{inverse}(A)$

---

Update A

$A = (A + u^*v')$

---

[ratio < 1 and fail random check: evaluate next change]

---

[pass]

[ratio >= 1 || pass random check]
Current QMPC implementation

[while not all change vectors evaluated]
matrix A, change vectors $U^*V'$

**Factorize A**
Find LU s.t. $A = LU$

[fail]

inverse$(A)$, determinant$(A)$, change vector $U^*V'$

**S.M. Formula**
Find inverse$(A + U^*V')$

[pass]

**Inverse Check**
$I = A * \text{inverse}(A)$

[ratio < 1 and fail random check: evaluate next change]

determinant$(A + U^*V')$

**Determinant Check**
Test:

\[
\text{det}(A + U^*V') / \text{det}(A)
\]

[ratio >= 1 || pass random check]

\[
A = (A + U^*V')
\]

**Update A**
Proposed Implementation

- Using QR factorization
- Rank-k update
- Triangular solve
Proposed Implementation

- Factorize $A$
  - Find $QR$ s.t. $A = QR$ 
- Triangular Solve
  - Find $v' \cdot \text{inverse}(R)$: solve $R \cdot y = v'$ 
- Update $R$
  - $R = R + w^*v'$, reform triangular matrix 
- M.D. Lemma
  - Find $\det(R + w^*v')$ 
- Determinant Check
  - Test: $\det(R + w^*v') / \det(R)$ 

[while not all change vectors evaluated]
matrix $A$, change vectors $U^*V'$

[evaluate next change]

[ratio < 1 && fail random check: evaluate next change]

[ratio >= 1 || pass random check]
QR Decomposition

\[ A = QR, \]

\[
\begin{pmatrix}
 q_1 & \ldots & q_n \\
\end{pmatrix}
\begin{pmatrix}
 r_{11} & r_{12} & \cdots & r_{1n} \\
 0 & r_{22} & \cdots & r_{2n} \\
 \vdots & \ddots & \ddots & \vdots \\
 0 & \cdots & 0 & r_{nn}
\end{pmatrix}
\]

Note that Q is orthonormal and R is upper triangular.
Proposed Implementation

[while not all change vectors evaluated] matrix A, change vectors U*V'

**Factorize A**
- Find QR s.t.
- A = QR

**Triangular Solve**
- Find v' * inverse(R): solve R * y = v'

**Update R**
- R = R + w*V', reform triangular matrix

**M.D. Lemma**
- Find det(R + w*V')
- det(R + w*V')

**Determinant Check**
- Test:
  - det(R + w*V') / det(R)
Proposed Implementation

[while not all change vectors evaluated] matrix A, change vectors U*V'

Factorize A
Find QR s.t. A = QR
Q, R, v', w = Q' * u

Triangular Solve
Find v' * inverse(R): solve R * y = v'
v' * inverse(R)

M.D. Lemma
Find det(R + w*v')
det(R + w*v')

Update R
R = R + w*v', reform triangular matrix

[evaluate next change]
[ratio < 1 && fail random check: evaluate next change]
[ratio >= 1 || pass random check]

Determinant Check
Test:
det(R + w*v') / det(R)
Proposed Implementation

[while not all change vectors evaluated]
matrix A, change vectors U*V'

Factorize A
Find QR s.t.
A = QR

Triangular Solve
Find v' * inverse(R):
solve R * y = v'

M.D. Lemma
Find det(R + w*v')

det(R + w*v')

Update R
R = R + w*v',
reform triangular matrix

Determinant Check
Test:
det(R + w*v') / det(R)

[evaluate next change]
[ratio < 1 && fail random check: evaluate next change]
[ratio >= 1 || pass random check]
Proposed Implementation

[Diagram details]
- **Factorize A**: Find QR s.t. $A = QR$
- **Triangular Solve**: Find $v' \cdot \text{inverse}(R)$: solve $R \cdot y = v'$
- **M.D. Lemma**: Find $\det(R + w^*v')$
- **Determinant Check**: Test: $\det(R + w^*v') / \det(R)$
- **Update R**: $R = R + w^*v'$, reform triangular matrix
- Flowchart includes conditions and operations described in the text.
Given’s Rotation

\[
\begin{bmatrix}
c & -s \\
s & c
\end{bmatrix}^T \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}, \quad r = \sqrt{a^2 + b^2}.
\]

**Figure 4.3:** Rotation of \( \mathbf{x} \) in a plane by an angle \( \alpha \).
Householder Reflection

\[ u(u^T x) \]

\[ x \]

\[ u \]

\[ -2u(u^T x) \]

\[ Hx = (I - 2uu^T)x \]

\[ = x - 2u(u^T x) \]
Completed Work

- Design our algorithms, background work, version control repository
- Serial MATLAB implementations for rank-1 and rank-k updates
- Serial C/LAPACK/BLAS for rank-1 update
- cuBLAS for rank-1 update

In Progress

- Serial C/LAPACK/BLAS for rank k
- C++/Cuda kernels for rank-1/rank k update
Acknowledgements

• We greatly appreciate help from our mentors:
  ➢ Dr. Ed D'Azevedo from ORNL
  ➢ Dr. Ying Wai Li from ORNL
  ➢ Dr. Kwai Wong from UTK