Multi-baryon systems and nuclear forces

Optimizing Hadronic Blocks

1

Hadronic building blocks facilitate the calculation of the correlation functions between initial and final nuclear systems. The simplest example is the propagation of a 3-quark state from the source, $(\mathbf{x}_0, t_0)$, to the sink, $(\mathbf{x}, t)$ where it is annihilated by a baryon interpolating field prescribed by the reduced weights $\tilde{w}_b(c_1, c_2, c_3|k)$ Orginos & Detmold

$$\mathcal{B}_b^{a_1,a_2,a_3}(\mathbf{p}, t; x_0) = \sum_{\mathbf{x}} e^{i\mathbf{p}\cdot\mathbf{x}} \sum_{k=1}^{N_{B(b)}} \tilde{w}_b^{(c_1,c_2,c_3),k} \sum_{\mathbf{i}} \epsilon^{i_1,i_2,i_3} S(c_{i_1}, x; a_1, x_0) S(c_{i_2}, x; a_2, x_0) S(c_{i_3}, x; a_3, x_0)$$

Parallelize/Vectorize

Goal

The interpolating fields for baryons with a given set of quantum numbers at the quark level are not unique. Exploiting symmetries and redundancies in the system enables access to more realistic and necessarily more complex – hence computationally more expensive – constructions at reasonable cost.

Components for parallization/vectorization

- Multiple baryon blocks
  - Λ, Σ, Ξ, proton, $\cdots \longleftarrow \mathcal{B}_b$
  - 1,728 combinations $\longleftarrow \{a_1, a_2, a_3\}$
- Multiple momentum projections

Features/constraints

- Common terms/sub-expressions
- Colour-singlet combination ϵ^{i_1,i_2,i_3}
- Temporal/Spatial locality
- Vectorization of products $S\,S\,S$

Strategy

Analyse all the terms required, perform the minimal required computations and organize the calculations to optimize locality of reference.

Example

Consider degenerate up/down quarks and two sets of interpolating fields — Setup 1 & 2. The latter contains interpolating fields involving more quark-level terms.

Original:

Setup	#SSS	multiplication	comment
1	228	2, $\mathbb{C}$, *Scalar*	*minimal setup required*
2	2,172	2, $\mathbb{C}$, *Scalar*	*superset of* Setup 1

Improved: *assume 4-wide vector*

Setup	#SSS	multiplication	comment
1	36	2, $\mathbb{C}$, *Vector*	
	24 + 36	1, $\mathbb{C}$, *Vector*	perform SSS in two stages
2	174	2, $\mathbb{C}$, *Vector*	
	67 + 174	1, $\mathbb{C}$, *Vector*	perform SSS in two stages

Summary

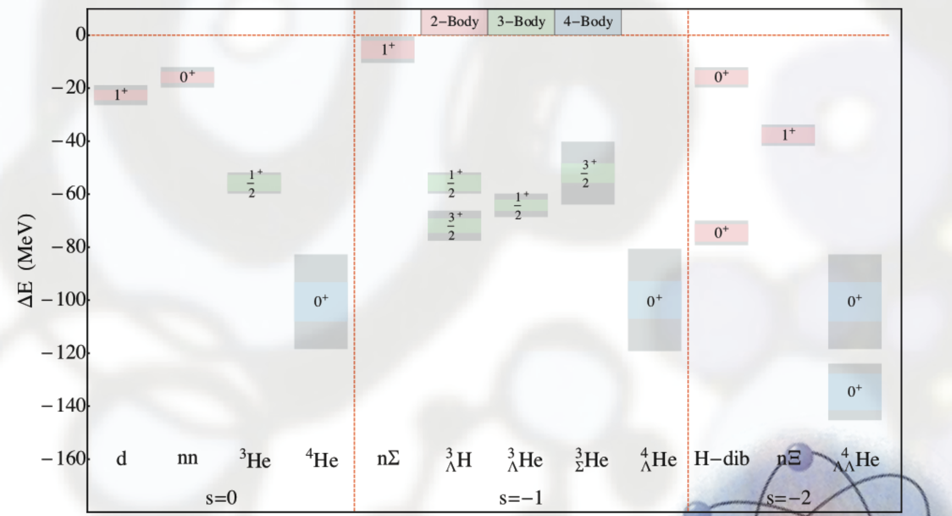
assume the cost of a scalar multiplication $\sim$ vector multiplication, ***or*** *scalar operations are performed by vector units*

Saving in the number of $\mathbb{C}$ multiplication:

85% for Setup 1 — 92% for Setup 2

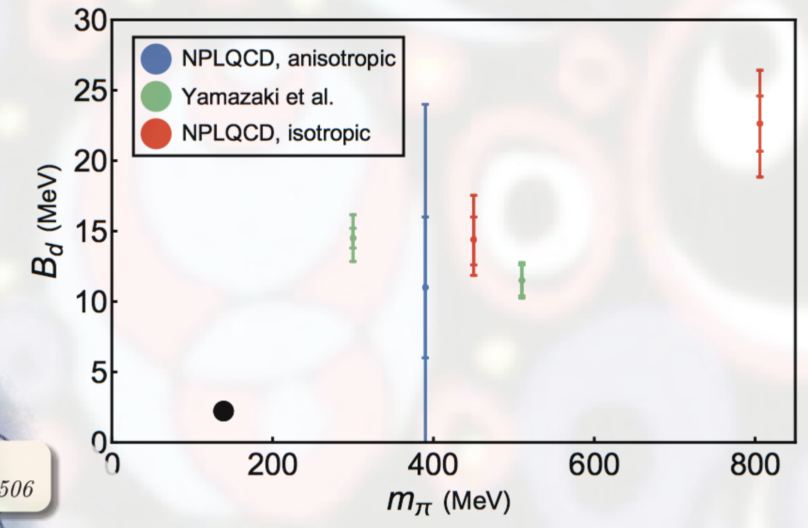
despite the great difference in the number of $S\,S\,S$ products involved (228 versus 2172). *In general, the greater the complexity of the interpolating fields, the greater the saving of computational effort.*

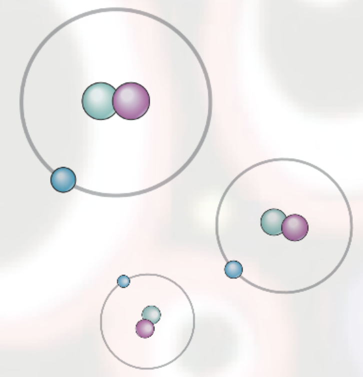
Nuclear Physics with Lattice QCD

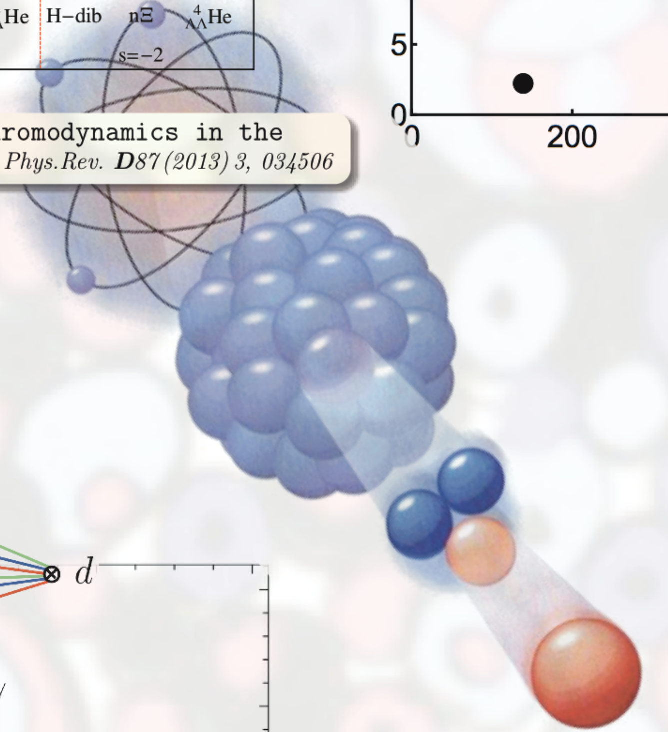


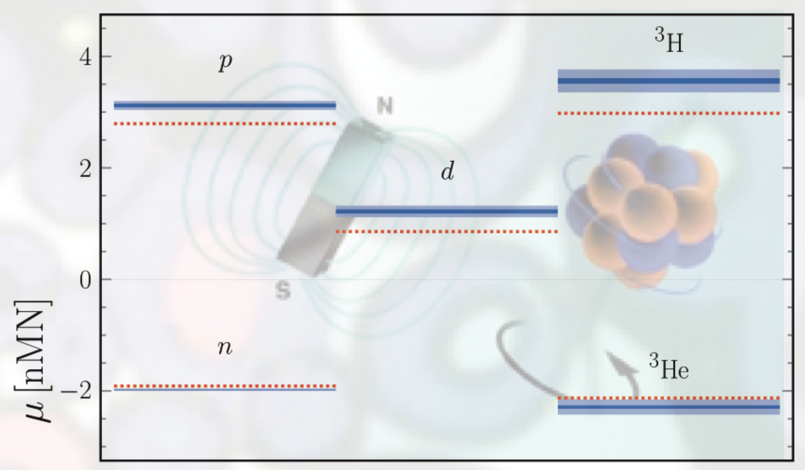
Light Nuclei and Hypernuclei from Quantum Chromodynamics in the limit of SU(3) Flavour Symmetry *Silas Beane et al, Phys.Rev.* ***D87*** *(2013) 3, 034506*

Two Nucleon Systems at pion mass of 450 MeV from Lattice QCD
William Detmold et al, July 2015

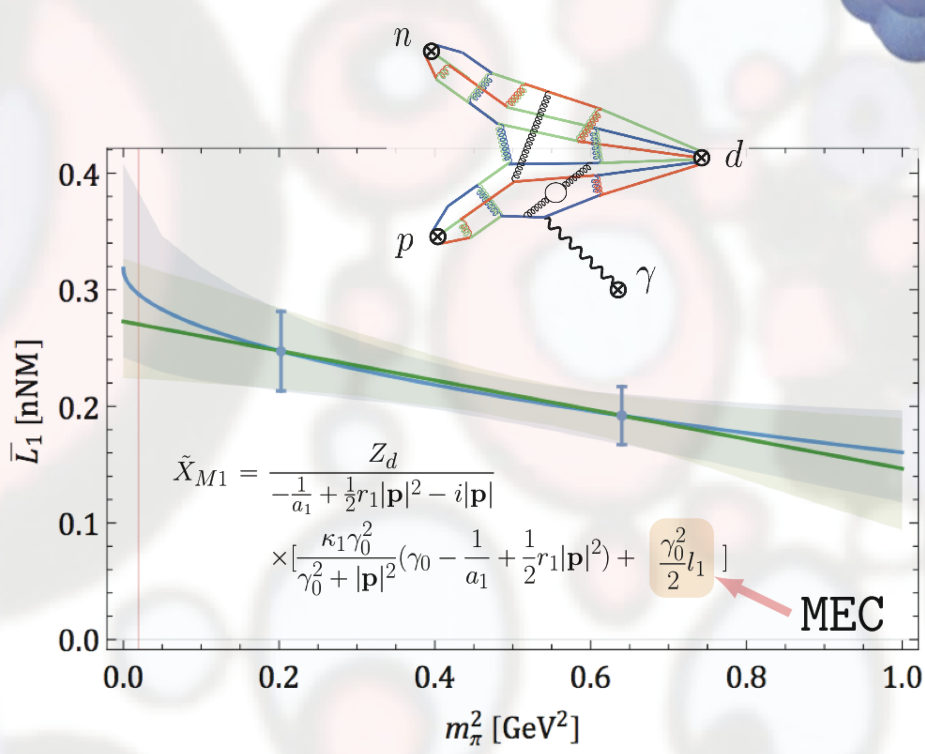




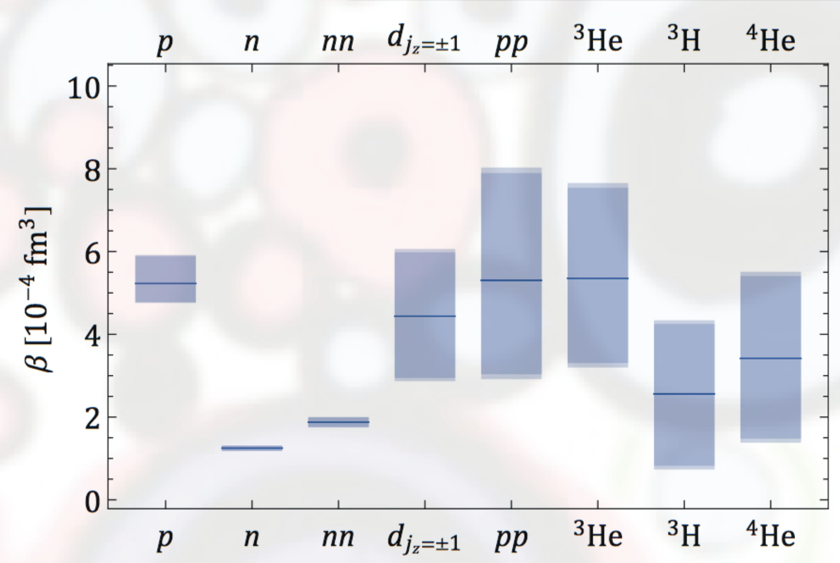




Magnetic moments of light nuclei from lattice quantum chromodynamics *Silas Beane et al, Phys.Rev.Lett. 113 (2014) 25, 252001*



Ab initio calculation of the $np \to d\gamma$ radiative capture process *Silas Beane et al,* ***arXiv****:1505.02422*



The Magnetic Structure of Light Nuclei from Lattice QCD *E. Chang et al,* ***arXiv****:1506.05518*

Author: Emmanuel Chang

Background image: psychedelic_wallpaper_by_7lady7maria7.jpg

Institute for Nuclear Theory

NPLQCD

SciDAC Scientific Discovery through Advanced Computing

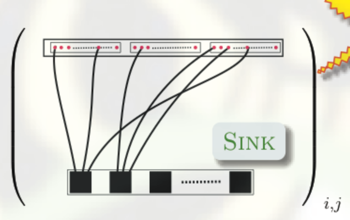
USQCD — US Lattice Quantum Chromodynamics

Optimizing Correlation Function Calculation

2

To calculate a correlation function between nuclear states, contractions are performed between a source (quark-level) and a sink (hadronic) interpolating field using *hadronic blocks*. 3-quark states are propagated from source to sink in all possible ways weighted by $\{ W^i_{Quark/Hadron} \}$ which specifies a particular source/sink interpolating field with a given set of quantum numbers Orginos & Detmold.

$$\langle \text{Sink(t)}\ \text{Source(0)} \rangle = \sum_{i,j}^{N_Q,N_H} W^i_{Quark} \times W^j_{Hadron} \left(\cdots \right)_{i,j}$$



QHCAP

Parallelize/Vectorize

Goal

A large number of contractions are performed with increasing complexity as the atomic number of the nucleus increases and the number of possible permutations grows exponentially from propagating permissible 3-quark state in all possible ways from source to sink. *Exploiting the large number of terms and relations between them leads to speedier and higher throughput calculations by parallelizing them over a maximum possible number of computation units.*

...the alternative $\longrightarrow$ **excessive disk usage**

Without a careful analysis of all the terms, the only option is to conduct the calculation sequentially. This implies a long latency between production of the propagators, hence hadronic blocks, and the contraction while independent tasks are gathered to fill the capacity of each computing node leading to excessive disk usage which becomes a limiting factor in large scale production.

Components for parallization/vectorization

- Time slices at sink can be distributed to
  - vector lanes
  - CPU cores, computes nodes
- $\{N_Q, N_H\}$ loops can be split/distributed to
  - CPU cores, computes nodes

Features/constraints

- The QHCAP
  - Common terms/sub-expressions
  - Parallelization/Vectorization
- Temporal/Spatial locality

...options ?

Treat QHCAP as an opaque object

- Split the $\{N_Q, N_H\}$ loops and load balance, using measured serial runtime, across multiple CPU cores and/or compute nodes reaching a maximum of **128** cores limited by the runtime of the most complicated correlator with $A = 5$.
- Distribute the time slices across multiple nodes.

Expand terms in $\sum W_Q \times W_H$ QHCAP explicitly

- Distribute the large number of terms, $\sim 10^9$, across multiple CPU cores and/or compute nodes. No longer limited by the complexity of an individual correlator, giving a better load balancing across cores.
- Distribute the time slices across vector lanes/CPU cores/compute nodes.

Summary

The large number of terms in the expansion of the correlators do not lend themselves easily to vectorization – instead time slices are threaded into each SIMD lane. However they do make it possible to scale the calculation to a few thousand cores and to pipeline the calculation of propagtors with that of the contractions to fully utilize leadership computing facilities.

Data management

- Requirements:
  - compact ⟵ compressed, binary, ..
  - easy of retrieval, search – by Q-num – for analysis
- Adopt SQLite3 as new application data format
  - develop new user interface (Mathematica)
  - requires new data extraction or *stripping*

Extract from the YAML specification for the new *stripper*


```
hadspec/xml:
'@Wilson_hadron_measurements':
...
Forward_prop_headers/First_forward_prop/PropSource/Source/t_srce:

 xyzt

SourceSinkType/(source|sink)_type_1:

 - - smearing/.key.matched => .value

   - .value: { POINT_SINK: P, POINT_SOURCE: P,
               SHELL_SINK: S, SHELL_SOURCE: S }

Mass_(1|2):
 [ [ mass/.key.matched , .key.matched: { 1: u/d , 2: s } ] ]

'@Shell_(Shell|Point)_Wilson_Mesons':

 '@momenta':
  sink_mom: &momentum
   momentum
  '@mesprop': &correlator
   [ correlator ,
    [ group => .key , .key: { mesprop: meson , barprop: baryon } ]
   ]

 gamma_value:
  - - op_id
    - .value: [
        1,  1,1,  1,
              1,  1,1,
        2,
        2,  2,2,  1,
              2,  1,1,
        1
       ]
  - - particle
    - .value: [
          a0,  [ rho , x ], [ rho , y ],   [ b1 , z ],
                            [ rho , z ],   [ b1 , y ], [ b1 , x ],
        pion,
          a0,  [ rho , x ], [ rho , y ],   [ a1 , z ],
                            [ rho , z ],   [ a1 , y ], [ a1 , x ],
        pion
       ]
```


Column $x\,y\,z\,t$: x, y, z, t coordinate of the source of the propagator.

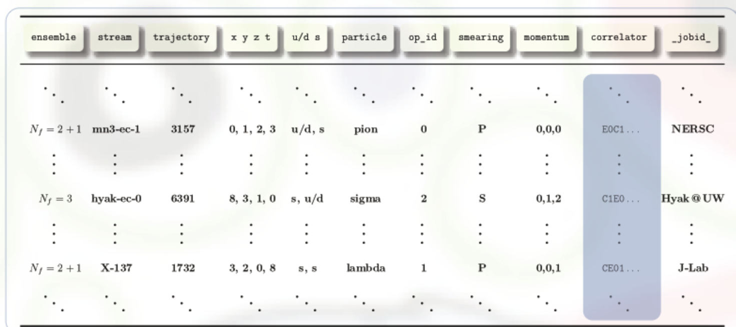
Column smearing

Column u/d s . Contractions are evaluated over all four combinations of u/d & s values. This specific ordering allows identification of the correct correlator.

Columns op_id , momentum and particle . *Particle* refers to the actual contraction function used.

These corresponds to the inner most loop in the source code. The nested groups of the XML data file is expressed by @ and indentation in the YAML file.

This YAML specification can be extended to be used with all our XML data files greatly simplyfing the task of extracting simulation data.



1. SQLite3 allows us to implement tables to record the results of simulation. Look-up and combining with other tables – of the same type – are *easy* using standard SQL queries.
2. It unifies our two current data formats:
   - XML data format (Text)
   - SDB (Binary, Key/Value)
3. The complexity of XML data format, as seen on the left, and the specificity of the SDB makes SQLite3 an very attractive option.

- (Above) An example of a table from XML data set – *hadspec*.
- (Below) The SQL schema in our YAML specification used to construct the table after stripping.

```
hadspec/sql:

 .primary.key: { .exclude: correlator }
 .schema:

  - TEXT:       ensemble[64] stream[16]
  - INTEGER:    trajectory
  - INTEGER:    x y z t

  - TEXT[4]:    [ { u/d: mass/'u/d' } , { s: mass/s } ]
  - TEXT[32]:   particle
  - INTEGER:    op_id

  - TEXT:       { 'smearing[2]': smearing/sink , 'momentum[16]': }

  - TEXT[128]:  _jobid_

  - BINARY(64): { _digest_: { .digest: correlator , .algorithm: BLAKE2B } }
  - BLOB:       correlator
```

Multi-baryon correlators

These have more complicated table definitions and actual require multiple tables to describe. The information from the tables are used to build the *catalogue* shown below to allow selection of desired correlators with given quantum numbers.

Further below is our standard Mathematica interface to retrieve the correlators. It is designed so that the same notation is used for both the XML and SDB data set as far as possible.