Microsoft Research

ALPS
Algorithms and Libraries for Physics Simulations

Bela Bauer
Station Q, Santa Barbara

Outline

- First lecture:
  - Overview of ALPS, introduction to main concepts
  - Preparation for the tutorial (blackboard)

- Second lecture:
  - Hands-on

Outline

- First lecture:
  - Overview of ALPS, introduction to main concepts
  - Preparation for the tutorial (blackboard)

- Second lecture:
  - Hands-on

Please ask many questions!

The ALPS project

Provide open-source community-backed tools for the simulation of strongly correlated systems.

- **Replace status quo** of individual codes with reliable, tested, efficient and documented community codes relying on common file formats

The ALPS project

Provide open-source community-backed tools for the simulation of strongly correlated systems.

- **Replace status quo** of individual codes with reliable, tested, efficient and documented community codes relying on common file formats

1. Standard file formats
2. Libraries
3. Evaluation tools
4. Applications

The ALPS collaboration

Release 2.0:

- Switzerland
  - B. Bauer, L. Gamper, J. Gukelberger, A. Hehn, S. Isakov, P. N. Ma, J. D. Picon, B. Surer, M. Troyer, P. Werner
- United States
  - L. D. Carr, A. Feiguin, J. Freire, D. Koop, P. Mates, E. Gull, E. Santos, V. W. Scarola, C. Silva, M. L. Wall
- Germany
  - S. Fuchs, S. Gürtler, L. Pollet, U. Schollwöck, S. Trebst, S. Wessel
- Japan
  - R. Igarashi, H. Matsuo, S. Todo
- Austria
  - H. G. Evertz
- France
  - O. Parcollet
- Poland
  - G. Pawlowski

MPS applications:

- Switzerland
  - M. Dolfi, S. Keller, A. Kosenkov, T. Ewart, A. Kantian, T. Giamarchi, M. Troyer
- United States
  - B. Bauer

Key technologies

- Widely used object-oriented programming languages:
  - Performance-relevant parts implemented in **C++**
  - C++ code uses and follows coding practices of **Standard Library** and **Boost** libraries
  - Data analysis/evaluation code developed in **Python**
    - Extensive use of **NumPy/SciPy** and **Matplotlib** for computations and plotting
- Standard file formats:
  - **HDF5** (Hierarchical Data Format v5) binary format for efficient processing of large data sets
  - **XML** for human-readable input/output

Goals

- Provide **efficient implementation of standard algorithms** for solving physics problems, as well as testing and benchmarking new methods

- Facilitate **reproducibility** by providing **standardized file formats** and appropriate **provenance** mechanisms

- **Simplify** development of reliable simulation codes by providing **key libraries**

- Simplify data analysis by providing **powerful analysis tools**

Goals

- Provide **efficient implementation of standard algorithms** for solving physics problems, as well as testing and benchmarking new methods

- Facilitate **reproducibility** by providing **standardized file formats** and appropriate **provenance** mechanisms

- **Simplify** development of reliable simulation codes by providing **key libraries**

- Simplify data analysis by providing **powerful analysis tools**

ALPS applications

ALPS applications

- **Classical Monte Carlo:**
  - Local and cluster update algorithms for spin systems [spinmc]

ALPS applications

- **Classical Monte Carlo:**
  - Local and cluster update algorithms for spin systems [spinmc]
- **Exact Diagonalization**
  - Full [fulldiag] and sparse [sparsediag] diagonalization

ALPS applications

- **Classical Monte Carlo:**
  - Local and cluster update algorithms for spin systems [spinmc]
- **Exact Diagonalization**
  - Full [fulldiag] and sparse [sparsediag] diagonalization
- **Quantum Monte Carlo:**
  - Stochastic Series Expansion (SSE) [dirloop_sse]
  - Loop code [loop]
  - Continuous-time worm algorithm [worm and dwa]
  - Extended ensemble (quantum Wang-Landau) algorithm [qwl]

ALPS applications

- **Classical Monte Carlo:**
  - Local and cluster update algorithms for spin systems [spinmc]
- **Exact Diagonalization**
  - Full [fulldiag] and sparse [sparsediag] diagonalization
- **Quantum Monte Carlo:**
  - Stochastic Series Expansion (SSE) [dirloop_sse]
  - Loop code [loop]
  - Continuous-time worm algorithm [worm and dwa]
  - Extended ensemble (quantum Wang-Landau) algorithm [qwl]
- **Tensor network methods:**
  - DMRG [dmrg]
  - Time-evolving block decimation [tebd]
  - Matrix-product state implementations [mps_X]

ALPS applications

- **Classical Monte Carlo:**
  - Local and cluster update algorithms for spin systems [spinmc]
- **Exact Diagonalization**
  - Full [fulldiag] and sparse [sparsediag] diagonalization
- **Quantum Monte Carlo:**
  - Stochastic Series Expansion (SSE) [dirloop_sse]
  - Loop code [loop]
  - Continuous-time worm algorithm [worm and dwa]
  - Extended ensemble (quantum Wang-Landau) algorithm [qwl]
- **Tensor network methods:**
  - DMRG [dmrg]
  - Time-evolving block decimation [tebd]
  - Matrix-product state implementations [mps_X]
- **DMFT**

ALPS applications

- **Classical Monte Carlo:**
  - Local and cluster update algorithms for spin systems [spinmc]
- **Exact Diagonalization**
  - Full [fulldiag] and sparse [sparsediag] diagonalization
- **Quantum Monte Carlo:**
  - Stochastic Series Expansion (SSE) [dirloop_sse]
  - Loop code [loop]
  - Continuous-time worm algorithm [worm and dwa]
  - Extended ensemble (quantum Wang-Landau) algorithm [qwl]
- **Tensor network methods:**
  - DMRG [dmrg]
  - Time-evolving block decimation [tebd]
  - Matrix-product state implementations [mps_X]
- **DMFT**

Goals

- Provide **efficient implementation of standard algorithms** for solving physics problems, as well as testing and benchmarking new methods

- Facilitate **reproducibility** by providing **standardized file formats** and appropriate **provenance** mechanisms

- **Simplify** development of reliable simulation codes by providing **key libraries**

- Simplify data analysis by providing **powerful analysis tools**

Goals

- Provide **efficient implementation of standard algorithms** for solving physics problems, as well as testing and benchmarking new methods

- Facilitate **reproducibility** by providing **standardized file formats** and appropriate **provenance** mechanisms

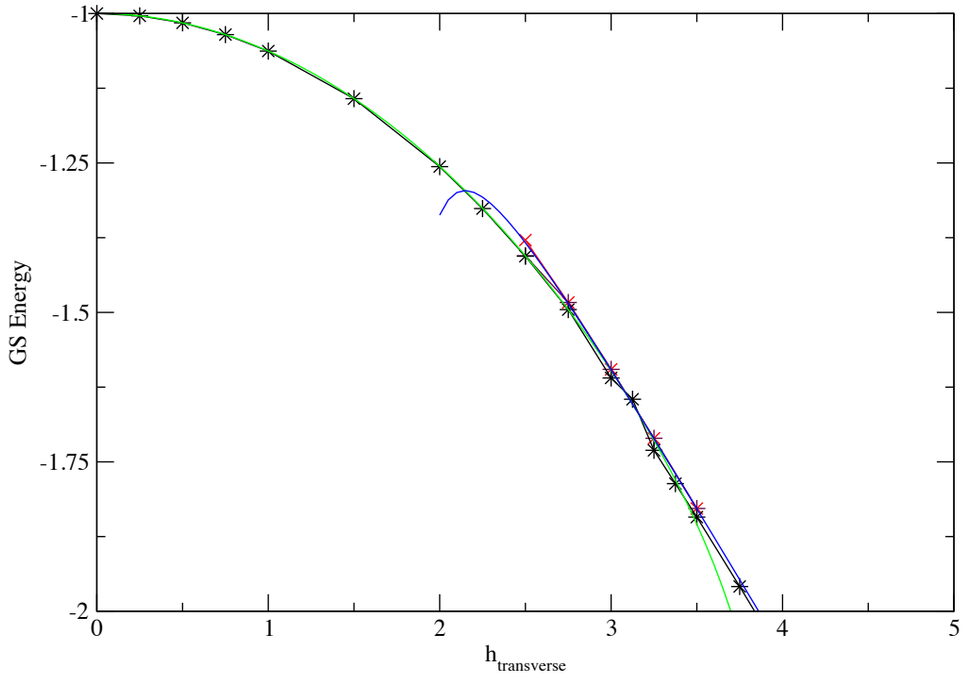
- **Simplify** development of reliable simulation codes by providing **key libraries**

- Simplify data analysis by providing **powerful analysis tools**

Reproducibility

- So you ran a simulation, stored the result in a text file and made a figure:

```
My-Mac:~ bela$ tail -n 4 /Users/bela/iPEPS/results/iPEPS_BLBQ/run2/M2_P30_s10/measurements.dat
a 0 energy -2.12543
a 1 energy -2.20704
a 2 energy -2.12558
a 3 energy -2.20719
```



- 3 years later, you want to reproduce it for your thesis…
  - Which version of your code did you run?
  - What are the parameters?
  - Was any post-processing applied to the data?

Provenance tracking

Provenance tracking

- Provenance on Wiktionary:

1 Place or source of origin. *Many supermarkets display the **provenance** of their food products.*
2 (*archaeology*) The place and time of origin of some artifact or other object. *This spear is of Viking **provenance**.*
3 (*art*) The history of ownership of a work of art *The picture is of royal **provenance**.*
4 (*computing*) The copy history of a piece of data, or the intermediate pieces of data utilized to compute a final data element, as in a database record or web site (data provenance)
5 (*computing*) The execution history of computer processes which were utilized to compute a final piece of data (process provenance)

Provenance tracking

- Provenance on Wiktionary:

1 Place or source of origin. *Many supermarkets display the **provenance** of their food products.*
2 (*archaeology*) The place and time of origin of some artifact or other object. *This spear is of Viking **provenance**.*
3 (*art*) The history of ownership of a work of art *The picture is of royal **provenance**.*
4 (*computing*) The copy history of a piece of data, or the intermediate pieces of data utilized to compute a final data element, as in a database record or web site (data provenance)
5 (*computing*) The execution history of computer processes which were utilized to compute a final piece of data (process provenance)

Provenance tracking

- Provenance on Wiktionary:

1 Place or source of origin. *Many supermarkets display the **provenance** of their food products.*
2 (*archaeology*) The place and time of origin of some artifact or other object. *This spear is of Viking **provenance**.*
3 (*art*) The history of ownership of a work of art *The picture is of royal **provenance**.*
4 (*computing*) The copy history of a piece of data, or the intermediate pieces of data utilized to compute a final data element, as in a database record or web site (data provenance)
5 (*computing*) The execution history of computer processes which were utilized to compute a final piece of data (process provenance)

- Preserve **information on the provenance of data**:
  - Who created this result, and when?
  - What application, and which version of it, was used?
  - What parameters were passed to the application?
  - What post-processing was performed?

Provenance tracking

- Provenance on Wiktionary:

1 Place or source of origin. *Many supermarkets display the **provenance** of their food products.*
2 (*archaeology*) The place and time of origin of some artifact or other object. *This spear is of Viking **provenance**.*
3 (*art*) The history of ownership of a work of art *The picture is of royal **provenance**.*
4 (*computing*) The copy history of a piece of data, or the intermediate pieces of data utilized to compute a final data element, as in a database record or web site (data provenance)
5 (*computing*) The execution history of computer processes which were utilized to compute a final piece of data (process provenance)

- Preserve **information on the provenance of data**:
  - Who created this result, and when?
  - What application, and which version of it, was used?
  - What parameters were passed to the application?
  - What post-processing was performed?
- **Facilitate the recording** of the above data:
  - Standard formats for input parameters and output data
  - Provide automated mechanisms to record provenance

Provenance tracking

- Provenance on Wiktionary:

1 Place or source of origin. *Many supermarkets display the **provenance** of their food products.*
2 (*archaeology*) The place and time of origin of some artifact or other object. *This spear is of Viking **provenance**.*
3 (*art*) The history of ownership of a work of art *The picture is of royal **provenance**.*
4 (*computing*) The copy history of a piece of data, or the intermediate pieces of data utilized to compute a final data element, as in a database record or web site (data provenance)
5 (*computing*) The execution history of computer processes which were utilized to compute a final piece of data (process provenance)

- Preserve **information on the provenance of data**:
  - Who created this result, and when?
  - What application, and which version of it, was used?
  - What parameters were passed to the application?
  - What post-processing was performed?
- **Facilitate the recording** of the above data:
  - Standard formats for input parameters and output data
  - Provide automated mechanisms to record provenance
- **Make data & provenance accessible** to publishers, referees & readers
  - Reproducible papers

Good provenance habits

Whatever software you use, practice good provenance habits!

- Use **descriptive** input formats, and well-defined & documented output formats
- **Store** input & output together
- **Don't hard-code** simulation parameters
- Use **versioning** software (SVN, Git) to keep track of source code revisions & record which version was used to perform simulations
- Keep track of **post-processing** steps
  - Perform all post-processing in **scripts rather than interactive tools**
  - **Store** post-processing information together with figure files and manuscripts

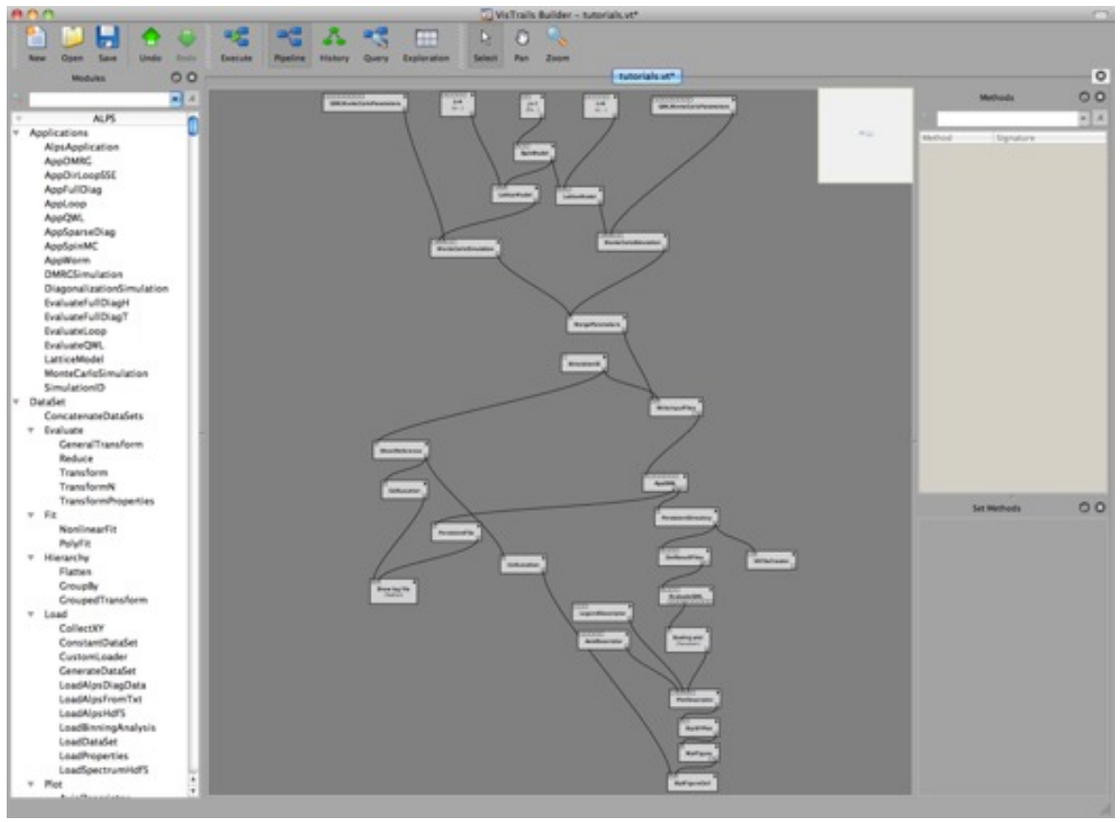
Advanced provenance tools

- VisTrails: **Visual workflow and provenance management**

Advanced provenance tools

- VisTrails: **Visual workflow and provenance management**

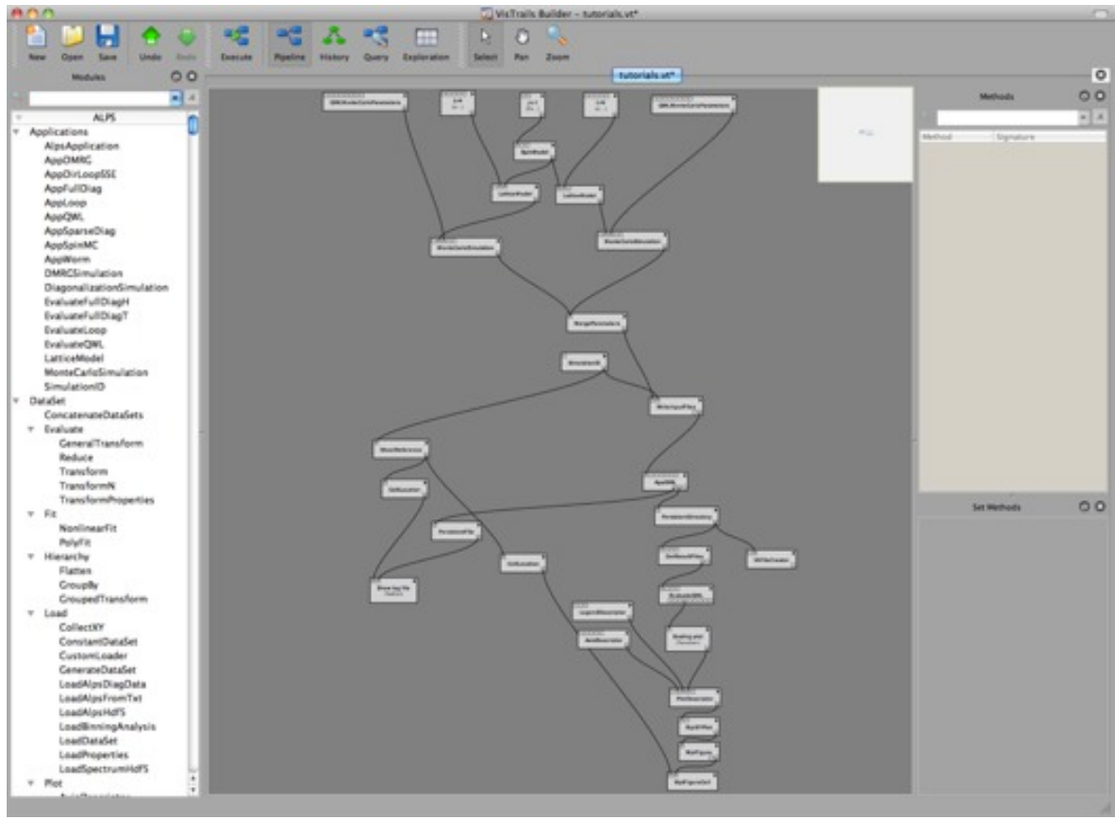
Visual workflows



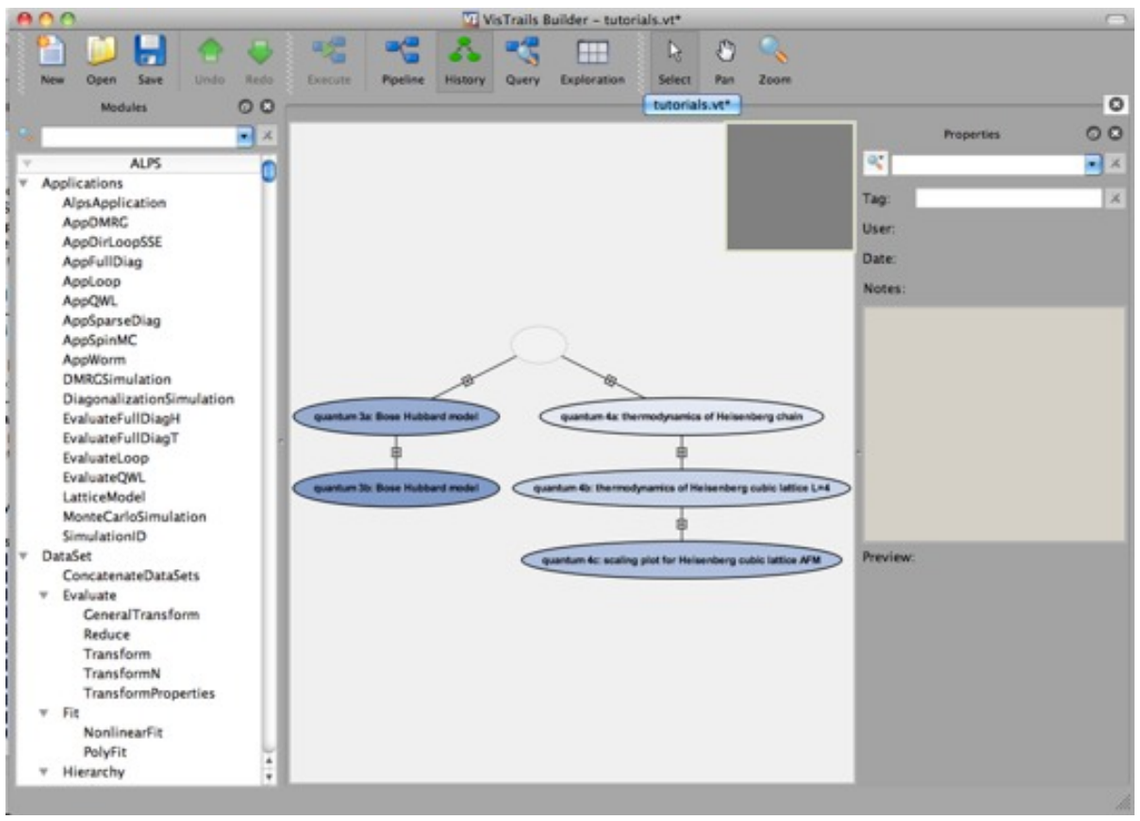
Advanced provenance tools

- VisTrails: **Visual workflow and provenance management**

Visual workflows

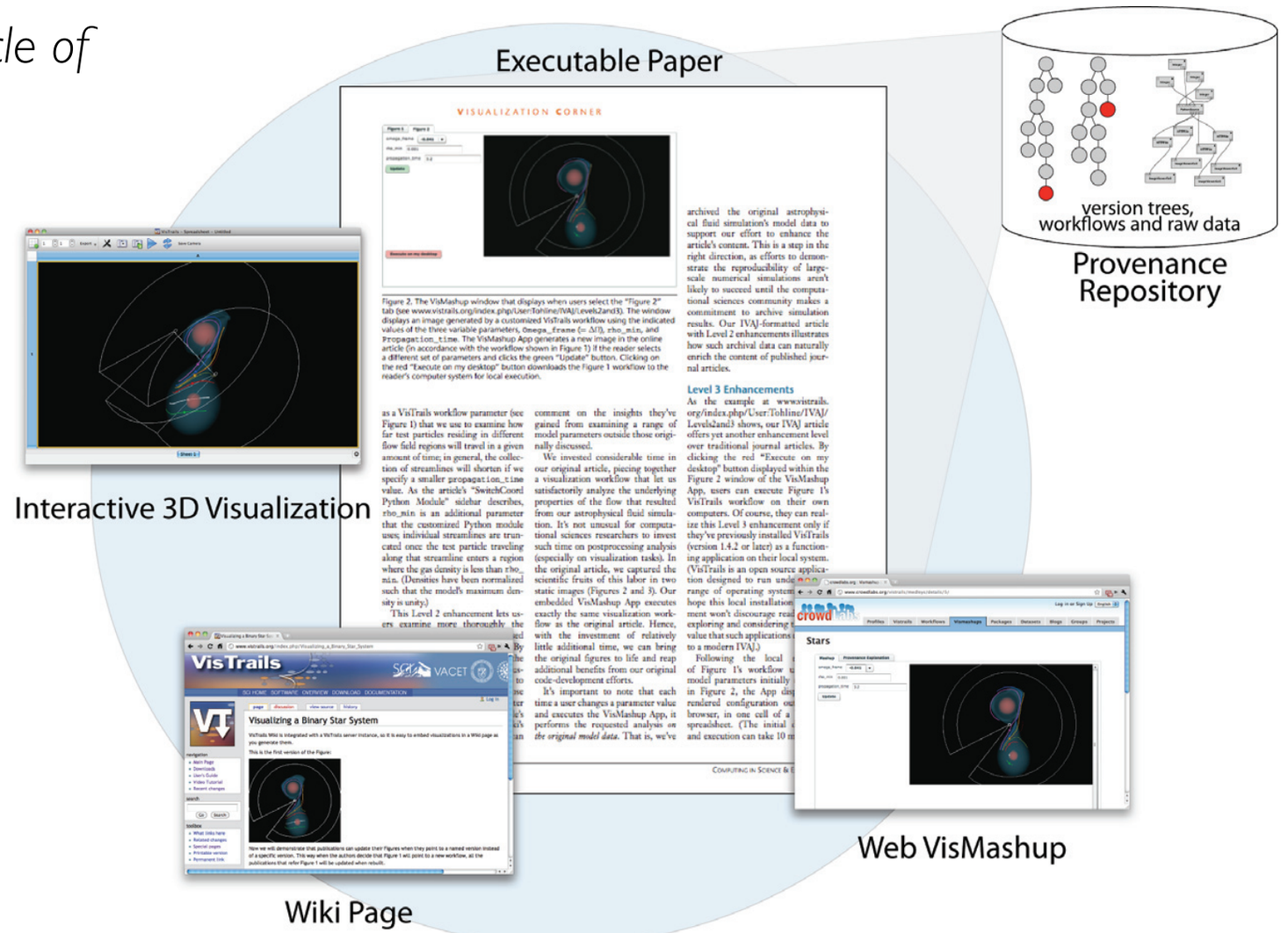


Exploration history

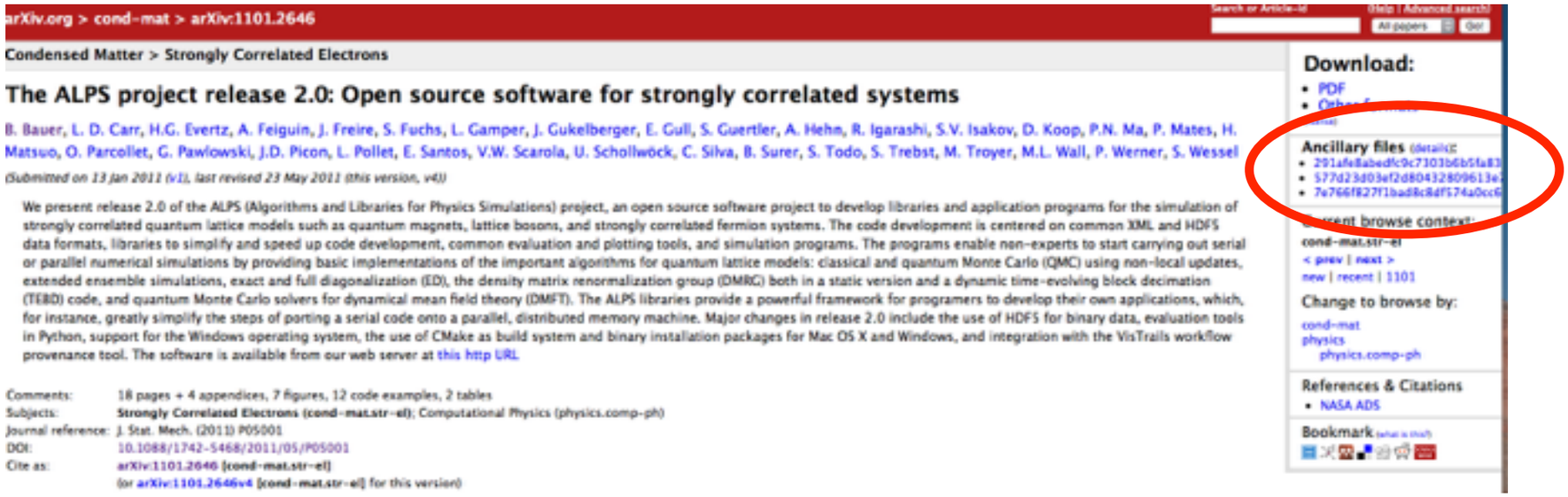


Executable papers

D. Koop et al, *A Provenance-Based Infrastructure to Support the Life Cycle of Executable Papers*, ICCS 2011



Executable papers



Executable papers

Journal of Statistical Mechanics: Theory and Experiment > Volume 2011 > May 2011

B Bauer *et al J. Stat. Mech.* (2011) P05001 doi:10.1088/1742-5468/2011/05/P05001

The ALPS project release 2.0: open source software for strongly correlated systems

B Bauer[1], L D Carr[2], H G Evertz[3], A Feiguin[4], J Freire[5], S Fuchs[6], L Gamper[1], J Gukelberger[1], E Gull[7], S Guertler[8], A Hehn[1], R Igarashi[9,10], S V Isakov[1], D Koop[5], P N Ma[1], P Mates[1,5], H Matsuo[11], O Parcollet[12], G Pawłowski[13], J D Picon[14], L Pollet[1,15], E Santos[5], V W Scarola[16], U Schollwöck[17], C Silva[5], B Surer[1], S Todo[10,11], S Trebst[18], M Troyer[1,21], M L Wall[2], P Werner[1] and S Wessel[19,20]

Show affiliations

Tag this article PDF (2.44 MB) View article

Abstract | References | Cited By | Figures | Tables | **Supplementary Data**

7e766f827f1bad8c8df574a0cc6135ce.vtl (45 kB file)
VisTrails file used to create figures 1 and 2.

Executable papers

Journal of Statistical Mechanics: Theory and Experiment > Volume 2011 > May 2011

B Bauer *et al J. Stat. Mech.* (2011) P05001 doi:10.1088/1742-5468/2011/05/P05001

The ALPS project release 2.0: open source software for strongly correlated systems

B Bauer[1], L D Carr[2], H G Evertz[3], A Feiguin[4], J Freire[5], S Fuchs[6], L Gamper[1], J Gukelberger[1], E Gull[7], S Guertler[8], A Hehn[1], R Igarashi[9,10], S V Isakov[1], D Koop[5], P N Ma[1], P Mates[1,5], H Matsuo[11], O Parcollet[12], G Pawłowski[13], J D Picon[14], L Pollet[1,15], E Santos[5], V W Scarola[16], U Schollwöck[17], C Silva[5], B Surer[1], S Todo[10,11], S Trebst[18], M Troyer[1,21], M L Wall[2], P Werner[1] and S Wessel[19,20]

Show affiliations

Tag this article PDF (2.44 MB) View article

Abstract | References | Cited By | Figures | Tables | Supplementary Data

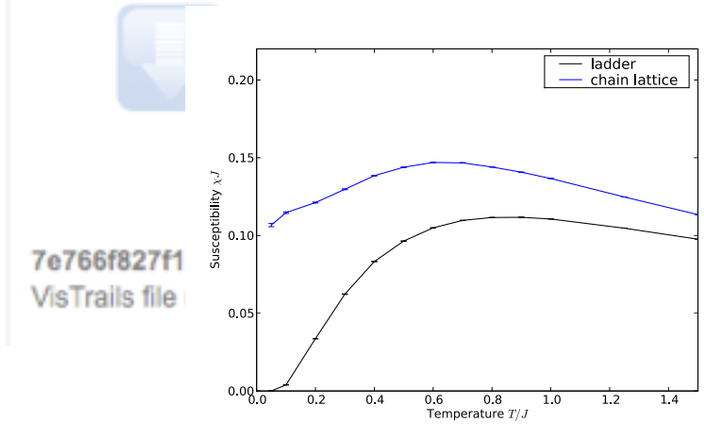


Figure 1. A figure produced by an ALPS VisTrails workflow: the uniform susceptibility of the Heisenberg chain and ladder. Clicking the figure retrieves the workflow used to create it. Opening that workflow on a machine with VisTrails and ALPS installed lets the reader execute the full calculation.

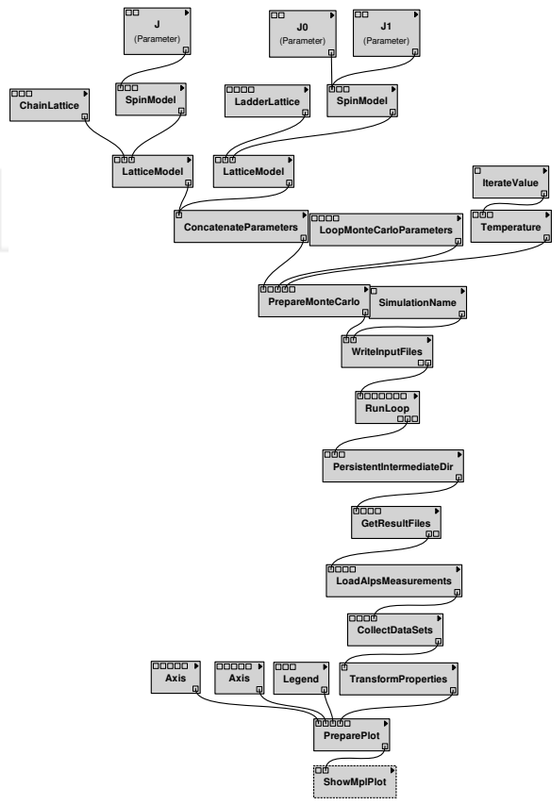


Figure 2. The workflow that created Fig. 1. The workflow image has been created by VisTrails for the specific workflow used to create the exact version shown in the figure. Clicking the figure retrieves the workflow.

Goals

- Provide **efficient implementation of standard algorithms** for solving physics problems, as well as testing and benchmarking new methods

- Facilitate **reproducibility** by providing **standardized file formats** and appropriate **provenance** mechanisms

- **Simplify** development of reliable simulation codes by providing **key libraries**

- Simplify data analysis by providing **powerful analysis tools**

Goals

- Provide **efficient implementation of standard algorithms** for solving physics problems, as well as testing and benchmarking new methods

- Facilitate **reproducibility** by providing **standardized file formats** and appropriate **provenance** mechanisms

- **Simplify** development of reliable simulation codes by providing **key libraries**

- Simplify data analysis by providing **powerful analysis tools**

ALPS Libraries

- ALPS **scheduler**: Easy and highly efficient MPI parallelization of Monte Carlo codes
- **Observables**: Efficient accumulation of MC measurement data using MPI, scalable to 1000's of nodes
- **Model & lattice** library: Descriptive input parameters for local Hamiltonians
- **Parameters**: Input from XML, plain-text and command line
- **HDF5**: Powerful C++ and Python interfaces to HDF5 library

ALPS Libraries

- ALPS **scheduler**: Easy and highly efficient MPI parallelization of Monte Carlo codes
- **Observables**: Efficient accumulation of MC measurement data using MPI, scalable to 1000's of nodes
- **Model & lattice** library: Descriptive input parameters for local Hamiltonians
- **Parameters**: Input from XML, plain-text and command line
- **HDF5**: Powerful C++ and Python interfaces to HDF5 library

Many of these can be used independently!

Goals

- Provide **efficient implementation of standard algorithms** for solving physics problems, as well as testing and benchmarking new methods

- Facilitate **reproducibility** by providing **standardized file formats** and appropriate **provenance** mechanisms

- **Simplify** development of reliable simulation codes by providing **key libraries**

- Simplify data analysis by providing **powerful analysis tools**

Goals

- Provide **efficient implementation of standard algorithms** for solving physics problems, as well as testing and benchmarking new methods

- Facilitate **reproducibility** by providing **standardized file formats** and appropriate **provenance** mechanisms

- **Simplify** development of reliable simulation codes by providing **key libraries**

- Simplify data analysis by providing **powerful analysis tools**

Data analysis tools

Data analysis tools

- Python package built around **NumPy**, **SciPy** and **Matplotlib**

Data analysis tools

- Python package built around **NumPy**, **SciPy** and **Matplotlib**
- Centered around `DataSet` structure:

```
import pyalps
help(pyalps.DataSet)

class DataSet(ResultProperties)
 |  The DataSet class stores a set of data, usually in XY format, along with all the properties
 |  describing the data, such as input parameters to the simulation etc.
 |
 |  Members are:
 |   * x, y - These contain the data and are expected to come as lists of Numpy arrays
 |            by many functions operating on DataSets. However, for user-supplied functions,
 |            other ways of representing data may be used.
 |   * props - This is a dictionary of properties describing the dataset.
```

Data analysis tools

- Python package built around **NumPy**, **SciPy** and **Matplotlib**
- Centered around `DataSet` structure:

```
import pyalps
help(pyalps.DataSet)

class DataSet(ResultProperties)
 |  The DataSet class stores a set of data, usually in XY format, along with all the properties
 |  describing the data, such as input parameters to the simulation etc.
 |
 |  Members are:
 |   * x, y - These contain the data and are expected to come as lists of Numpy arrays
 |            by many functions operating on DataSets. However, for user-supplied functions,
 |            other ways of representing data may be used.
 |   * props - This is a dictionary of properties describing the dataset.
```

- Provides many convenience functions:
  - Load data from XML or HDF5 into lists of `DataSet`s
  - Create X vs Y plots (`pyalps.collectXY()`)

Data analysis tools

- Python package built around **NumPy**, **SciPy** and **Matplotlib**
- Centered around `DataSet` structure:

```
import pyalps
help(pyalps.DataSet)

class DataSet(ResultProperties)
 |  The DataSet class stores a set of data, usually in XY format, along with all the properties
 |  describing the data, such as input parameters to the simulation etc.
 |
 |  Members are:
 |   * x, y - These contain the data and are expected to come as lists of Numpy arrays
 |            by many functions operating on DataSets. However, for user-supplied functions,
 |            other ways of representing data may be used.
 |   * props - This is a dictionary of properties describing the dataset.
```

- Provides many convenience functions:
  - Load data from XML or HDF5 into lists of `DataSet`s
  - Create X vs Y plots (`pyalps.collectXY()`)
- **Will be used extensively in Tutorials!**

Setting up ALPS simulations

Lattice

```
<LATTICE name="chain lattice" dimension="1">
  <PARAMETER name="a" default="1"/>
  <BASIS><VECTOR>a</VECTOR></BASIS>
</LATTICE>
```

Model

```
<HAMILTONIAN name="spin">
  <BASIS ref="spin"/>
  <SITETERM site="i">
    -h#*Sz(i)-Gamma#*Sx(i)+D#*Sz(i)*Sz(i)
  </SITETERM>
  <BONDTERM source="i" target="j">
    Jz#*Sz(i)*Sz(j)+Jxy#*exchange_xy(i,j)
  </BONDTERM>
</HAMILTONIAN>
```

Parameters

```
LATTICE = "chain lattice"
L = 100

MODEL = "spin"
J = 1
```

Setting up ALPS simulations

Lattice

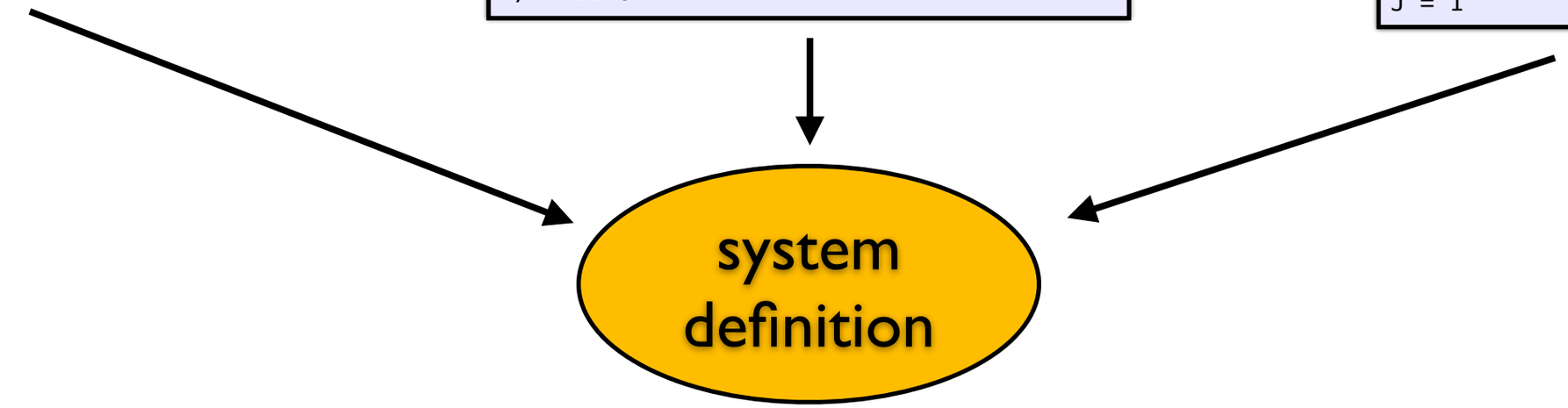
```
<LATTICE name="chain lattice" dimension="1">
  <PARAMETER name="a" default="1"/>
  <BASIS><VECTOR>a</VECTOR></BASIS>
</LATTICE>
```

Model

```
<HAMILTONIAN name="spin">
  <BASIS ref="spin"/>
  <SITETERM site="i">
    -h#*Sz(i)-Gamma#*Sx(i)+D#*Sz(i)*Sz(i)
  </SITETERM>
  <BONDTERM source="i" target="j">
    Jz#*Sz(i)*Sz(j)+Jxy#*exchange_xy(i,j)
  </BONDTERM>
</HAMILTONIAN>
```

Parameters

```
LATTICE = "chain lattice"
L = 100

MODEL = "spin"
J = 1
```



Setting up ALPS simulations

Lattice

```
<LATTICE name="chain lattice" dimension="1">
  <PARAMETER name="a" default="1"/>
  <BASIS><VECTOR>a</VECTOR></BASIS>
</LATTICE>
```

Model

```
<HAMILTONIAN name="spin">
  <BASIS ref="spin"/>
  <SITETERM site="i">
    -h#*Sz(i)-Gamma#*Sx(i)+D#*Sz(i)*Sz(i)
  </SITETERM>
  <BONDTERM source="i" target="j">
    Jz#*Sz(i)*Sz(j)+Jxy#*exchange_xy(i,j)
  </BONDTERM>
</HAMILTONIAN>
```

Parameters

```
LATTICE = "chain lattice"
L = 100

MODEL = "spin"
J = 1
```

system definition

Exact Diagonalization

Quantum Monte Carlo

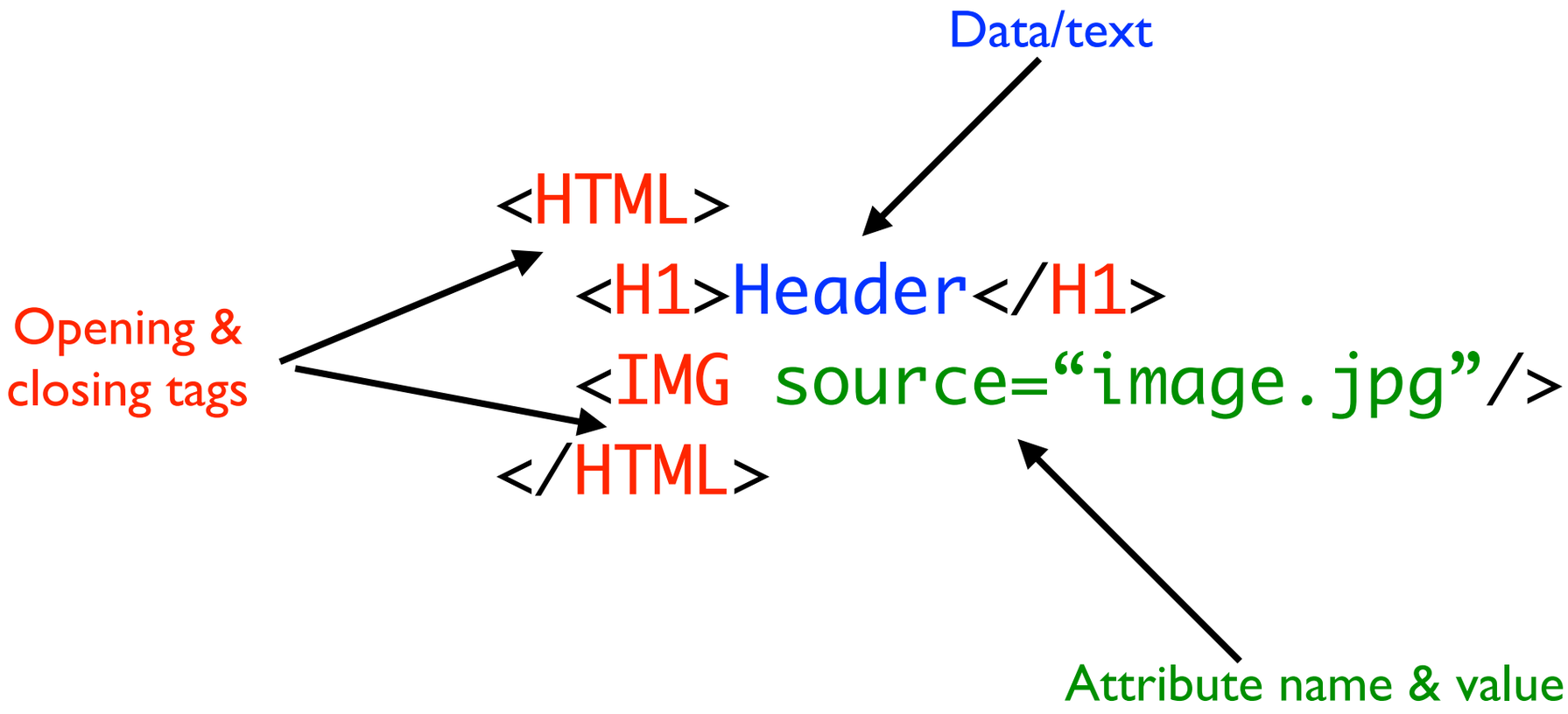
Tensor network methods

XML files

- Markup language: Text/data marked with tags
- Human and machine readable
- Example: HTML

XML files

- Markup language: Text/data marked with tags
- Human and machine readable
- Example: HTML



Lattice library

Lattice

```
<LATTICE name="chain lattice" dimension="1">
  <PARAMETER name="a" default="1"/>
  <BASIS><VECTOR>a</VECTOR></BASIS>
  <RECIPROCALBASIS><VECTOR>2*pi/a</VECTOR></RECIPROCALBASIS>
</LATTICE>

<UNITCELL name="simple1d" dimension="1">
  <VERTEX/>
  <EDGE>
    <SOURCE vertex="1" offset="0"/>
    <TARGET vertex="1" offset="1"/>
  </EDGE>
</UNITCELL>

<LATTICEGRAPH name = "chain lattice">
  <FINITELATTICE>
    <LATTICE ref="chain lattice"/>
    <EXTENT dimension="1" size ="L"/>
    <BOUNDARY type="periodic"/>
  </FINITELATTICE>
  <UNITCELL ref="simple1d"/>
</LATTICEGRAPH>
```

Lattice library

Lattice

↓

Unit cell

```
<LATTICE name="chain lattice" dimension="1">
  <PARAMETER name="a" default="1"/>
  <BASIS><VECTOR>a</VECTOR></BASIS>
  <RECIPROCALBASIS><VECTOR>2*pi/a</VECTOR></RECIPROCALBASIS>
</LATTICE>

<UNITCELL name="simple1d" dimension="1">
  <VERTEX/>
  <EDGE>
    <SOURCE vertex="1" offset="0"/>
    <TARGET vertex="1" offset="1"/>
  </EDGE>
</UNITCELL>

<LATTICEGRAPH name = "chain lattice">
  <FINITELATTICE>
    <LATTICE ref="chain lattice"/>
    <EXTENT dimension="1" size ="L"/>
    <BOUNDARY type="periodic"/>
  </FINITELATTICE>
  <UNITCELL ref="simple1d"/>
</LATTICEGRAPH>
```

- One vertex
- One edge

Lattice library

Lattice

↓

Unit cell

↓

Lattice graph

```
<LATTICE name="chain lattice" dimension="1">
  <PARAMETER name="a" default="1"/>
  <BASIS><VECTOR>a</VECTOR></BASIS>
  <RECIPROCALBASIS><VECTOR>2*pi/a</VECTOR></RECIPROCALBASIS>
</LATTICE>

<UNITCELL name="simple1d" dimension="1">
  <VERTEX/>
  <EDGE>
    <SOURCE vertex="1" offset="0"/>
    <TARGET vertex="1" offset="1"/>
  </EDGE>
</UNITCELL>

<LATTICEGRAPH name = "chain lattice">
  <FINITELATTICE>
    <LATTICE ref="chain lattice"/>
    <EXTENT dimension="1" size ="L"/>
    <BOUNDARY type="periodic"/>
  </FINITELATTICE>
  <UNITCELL ref="simple1d"/>
</LATTICEGRAPH>
```

- One vertex
- One edge

- One parameter: *L*
- Periodic boundary conditions

Lattice library – Example 2

```
<LATTICE name="square lattice" dimension="2">
  <PARAMETER name="a" default="1"/>
  <BASIS><VECTOR>a 0</VECTOR><VECTOR>0 a</VECTOR></BASIS>
  <RECIPROCALBASIS><VECTOR>2*pi/a 0</VECTOR><VECTOR>0 2*pi/a</VECTOR></RECIPROCALBASIS>
</LATTICE>

<UNITCELL name="simple2d" dimension="2">
  <VERTEX/>
  <EDGE><SOURCE vertex="1" offset="0 0"/><TARGET vertex="1" offset="0 1"/></EDGE>
  <EDGE><SOURCE vertex="1" offset="0 0"/><TARGET vertex="1" offset="1 0"/></EDGE>
</UNITCELL>

<LATTICEGRAPH name = "square lattice">
  <FINITELATTICE>
    <LATTICE ref="square lattice"/>
    <PARAMETER name="W" default="L"/>
    <EXTENT dimension="1" size="L"/>
    <EXTENT dimension="2" size="W"/>
    <BOUNDARY type="periodic"/>
  </FINITELATTICE>
  <UNITCELL ref="simple2d"/>
</LATTICEGRAPH>
```

Lattice library – Example 2

```
<LATTICE name="square lattice" dimension="2">
  <PARAMETER name="a" default="1"/>
  <BASIS><VECTOR>a 0</VECTOR><VECTOR>0 a</VECTOR></BASIS>
  <RECIPROCALBASIS><VECTOR>2*pi/a 0</VECTOR><VECTOR>0 2*pi/a</VECTOR></RECIPROCALBASIS>
</LATTICE>

<UNITCELL name="simple2d" dimension="2">
  <VERTEX/>
  <EDGE><SOURCE vertex="1" offset="0 0"/><TARGET vertex="1" offset="0 1"/></EDGE>
  <EDGE><SOURCE vertex="1" offset="0 0"/><TARGET vertex="1" offset="1 0"/></EDGE>
</UNITCELL>

<LATTICEGRAPH name = "square lattice">
  <FINITELATTICE>
    <LATTICE ref="square lattice"/>
    <PARAMETER name="W" default="L"/>
    <EXTENT dimension="1" size="L"/>
    <EXTENT dimension="2" size="W"/>
    <BOUNDARY type="periodic"/>
  </FINITELATTICE>
  <UNITCELL ref="simple2d"/>
</LATTICEGRAPH>
```

- One vertex
- Two edges

- Two sizes, L and W
- Periodic boundary conditions in both

Lattice library – Graph example

```
<GRAPH name="triangle" vertices="3">
  <EDGE type="0" source="1" target="2"/>
  <EDGE type="0" source="2" target="3"/>
  <EDGE type="0" source="3" target="1"/>
</GRAPH>
```

Lattice library – Graph example

```
<GRAPH name="triangle" vertices="3">
  <EDGE type="0" source="1" target="2"/>
  <EDGE type="0" source="2" target="3"/>
  <EDGE type="0" source="3" target="1"/>
</GRAPH>
```

- See `ALPS/lib/xml/lattices.xml`
- See https://alps.comp-phys.org/mediawiki/index.php/Tutorials:LatticeHOWTO
  - Simple graphs
  - Lattices with unit cells
  - Checking graphs & lattices
- Existing lattice: chain lattice, square, triangular, honeycomb, …

Model library

Site basis

```
<SITEBASIS name="spin">
  <PARAMETER name="local_S" default="1/2"/>
  <QUANTUMNUMBER name="S" min="local_spin" max="local_spin"/>
  <QUANTUMNUMBER name="Sz" min="-S" max="S"/>
  <OPERATOR name="Splus" matrixelement="sqrt(S*(S+1)-Sz*(Sz+1))">
    <CHANGE quantumnumber="Sz" change="1"/>
  </OPERATOR>
  <OPERATOR name="Sminus" matrixelement="sqrt(S*(S+1)-Sz*(Sz-1))">
    <CHANGE quantumnumber="Sz" change="-1"/>
  </OPERATOR>
  <OPERATOR name="Sz" matrixelement="Sz"/>
</SITEBASIS>
```

$|S, S^z\rangle$

$$S^+|S,S^z\rangle = \sqrt{S(S+1) - S^z(S^z+1)}|S,S^z+1\rangle$$
$$S^-|S,S^z\rangle = \sqrt{S(S+1) - S^z(S^z-1)}|S,S^z-1\rangle$$
$$S^z|S,S^z\rangle = S^z|S,S^z\rangle$$

Model library

Site basis

```
<SITEBASIS name="spin">
  <PARAMETER name="local_S" default="1/2"/>
  <QUANTUMNUMBER name="S" min="local_spin" max="local_spin"/>
  <QUANTUMNUMBER name="Sz" min="-S" max="S"/>
  <OPERATOR name="Splus" matrixelement="sqrt(S*(S+1)-Sz*(Sz+1))">
    <CHANGE quantumnumber="Sz" change="1"/>
  </OPERATOR>
  <OPERATOR name="Sminus" matrixelement="sqrt(S*(S+1)-Sz*(Sz-1))">
    <CHANGE quantumnumber="Sz" change="-1"/>
  </OPERATOR>
  <OPERATOR name="Sz" matrixelement="Sz"/>
</SITEBASIS>
```

$$|S, S^z\rangle$$

$$S^+|S,S^z\rangle = \sqrt{S(S+1) - S^z(S^z+1)}|S,S^z+1\rangle$$
$$S^-|S,S^z\rangle = \sqrt{S(S+1) - S^z(S^z-1)}|S,S^z-1\rangle$$
$$S^z|S,S^z\rangle = S^z|S,S^z\rangle$$

Basis

```
<BASIS name="spin">
  <SITEBASIS ref="spin">
    <PARAMETER name="local_S#" value="local_S"/>
    <PARAMETER name="local_S" value="1/2"/>
  </SITEBASIS>
  <CONSTRAINT quantumnumber="Sz" value="Sz_total"/>
</BASIS>
```

Model library

Site basis

↓

Basis

↓

Hamiltonian

```
<SITEBASIS name="spin">
  <PARAMETER name="local_S" default="1/2"/>
  <QUANTUMNUMBER name="S" min="local_spin" max="local_spin"/>
  <QUANTUMNUMBER name="Sz" min="-S" max="S"/>
  <OPERATOR name="Splus" matrixelement="sqrt(S*(S+1)-Sz*(Sz+1))">
    <CHANGE quantumnumber="Sz" change="1"/>
  </OPERATOR>
  <OPERATOR name="Sminus" matrixelement="sqrt(S*(S+1)-Sz*(Sz-1))">
    <CHANGE quantumnumber="Sz" change="-1"/>
  </OPERATOR>
  <OPERATOR name="Sz" matrixelement="Sz"/>
</SITEBASIS>

<BASIS name="spin">
  <SITEBASIS ref="spin">
    <PARAMETER name="local_S#" value="local_S"/>
    <PARAMETER name="local_S" value="1/2"/>
  </SITEBASIS>
  <CONSTRAINT quantumnumber="Sz" value="Sz_total"/>
</BASIS>

<HAMILTONIAN name="spin">
  <PARAMETER name="J" default="0"/>
  <PARAMETER name="Jz" default="J"/>
  <BASIS ref="spin"/>
  <SITETERM site="i">
    -h#*Sz(i)-Gamma#*Sx(i)+D#*Sz(i)*Sz(i)
  </SITETERM>
  <BONDTERM source="i" target="j">
    Jz#*Sz(i)*Sz(j)+Jxy#*exchange_xy(i,j)+K#*biquadratic(i,j)
  </BONDTERM>
</HAMILTONIAN>
```

$$|S, S^z\rangle$$

$$S^+|S,S^z\rangle = \sqrt{S(S+1) - S^z(S^z+1)}|S,S^z+1\rangle$$
$$S^-|S,S^z\rangle = \sqrt{S(S+1) - S^z(S^z-1)}|S,S^z-1\rangle$$
$$S^z|S,S^z\rangle = S^z|S,S^z\rangle$$

$$H_{\text{site}} = \sum_i \left(-hS_i^z - \Gamma S_i^x + DS_i^z S_i^z\right)$$

$$H_{\text{bond}} = \sum_{\langle i,j\rangle} \left(J_z S_i^z S_j^z + J_{xy} h_{ij}^{exc} + K h_{ij}^{blbq}\right)$$

Model library

Site basis

Basis

Hamiltonian

```
<SITEBASIS name="spin">
  <PARAMETER name="local_S" default="1/2"/>
  <QUANTUMNUMBER name="S" min="local_spin" max="local_spin"/>
  <QUANTUMNUMBER name="Sz" min="-S" max="S"/>
  <OPERATOR name="Splus" matrixelement="sqrt(S*(S+1)-Sz*(Sz+1))">
    <CHANGE quantumnumber="Sz" change="1"/>
  </OPERATOR>
  <OPERATOR name="Sminus" matrixelement="sqrt(S*(S+1)-Sz*(Sz-1))">
    <CHANGE quantumnumber="Sz" change="-1"/>
  </OPERATOR>
  <OPERATOR n
</SITEBASIS>

<BASIS name="
  <SITEBASIS
    <PARAMETE
    <PARAMETE
  </SITEBASIS
  <CONSTRAINT
</BASIS>

<HAMILTONIAN
  <PARAMETER name="J" default="0"/>
  <PARAMETER name="Jz" default="J"/>
  <BASIS ref="spin"/>
  <SITETERM site="i">
    -h#*Sz(i)-Gamma#*Sx(i)+D#*Sz(i)*Sz(i)
  </SITETERM>
  <BONDTERM source="i" target="j">
    Jz#*Sz(i)*Sz(j)+Jxy#*exchange_xy(i,j)+K#*biquadratic(i,j)
  </BONDTERM>
</HAMILTONIAN>
```

$$|S, S^z\rangle$$

$$= \sqrt{S(S+1) - S^z(S^z+1)}|S, S^z+1\rangle$$

$$= \sqrt{S(S+1) - S^z(S^z-1)}|S, S^z-1\rangle$$

$$= S^z|S, S^z\rangle$$

Existing models:

- Boson & fermion Hubbard
- Spinless fermions
- *t*-*J* model
- SU(2) spins

$$H_{\text{site}} = \sum_i (-hS_i^z - \Gamma S_i^x + DS_i^z S_i^z)$$

$$H_{\text{bond}} = \sum_{\langle i,j\rangle} \left(J_z S_i^z S_j^z + J_{xy} h_{ij}^{exc} + K h_{ij}^{blbq}\right)$$

3 Ways of running ALPS

- Command line:
  - Requires minimal external dependencies (can be built without Python)
  - Best suited for remote use on clusters etc.
  - Greatest flexibility
- Python
  - Both command-line and interactive use when used with IPython
  - Best suited for remote applications when Python is available
- VisTrails
  - Interactive use
  - Best tracking of provenance information

3 Ways of running ALPS

- Command line:
  - Requires minimal external dependencies (can be built without Python)
  - Best suited for remote use on clusters etc.
  - Greatest flexibility
- Python
  - Both command-line and interactive use when used with IPython
  - Best suited for remote applications when Python is available
- VisTrails
  - Interactive use
  - Best tracking of provenance information

[i]Python

- **Python**: easily accessible, portable and widely used interpreted programming language
  - Currently tested against Python 2.7
  - Many tutorials, e.g.:
    - Chris Laumann's lectures
    - https://wiki.python.org/moin/BeginnersGuide
    - https://docs.python.org/2/tutorial/
- **IPython**: improved Interactive Python shell
- **IPython Notebook**/Jupyter:
  - Browser interface for IPython
  - Offers full advantage of both interactive and scripted tools for data analysis
  - Similar interface to Mathematica: code cells
  - **Will be used in tutorial.**

iPython Notebook

jupyter DMRG_Chain_Spectrum_Entanglement Last Checkpoint: Last Monday at 10:38 AM (autosaved)

File Edit View Insert Cell Kernel Help | Python 2

Code | Cell Toolbar: None

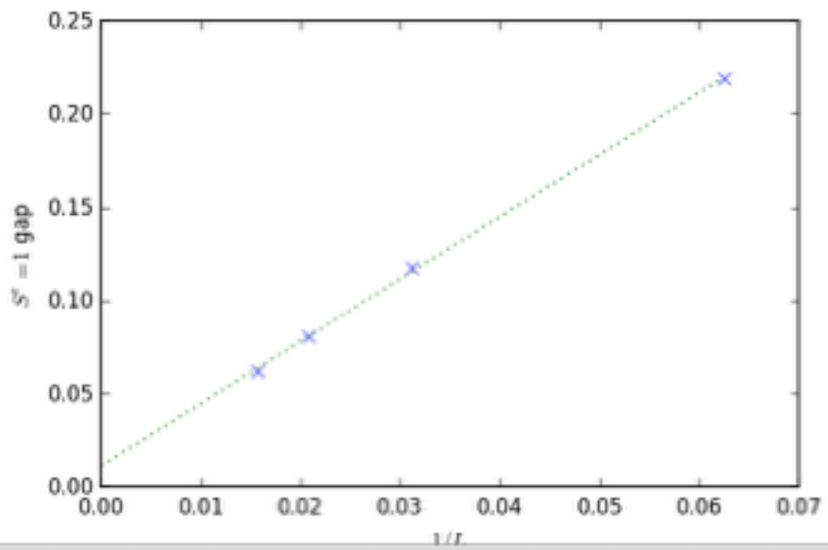
Spectrum

We first explore the spectrum of the Heisenberg chain. In the following panel, we plot the spin triplet gap, which is the energy difference between the ground state in the $S_z = 1$ and the $S_z = 0$ sector. We then perform a fit to a polynomial using a method from NumPy.

- Change the order of the polynomial used for the fit. What is the leading order?
- What does this imply about the spectral gap in the thermodynamic limit?

In [9]:
```
energy_data = pyalps.loadEigenstateMeasurements(pyalps.getResultFiles(prefix=chain_prefix), what=['Energy'])

p = pyalps.collectXY(energy_data, 'L', 'Energy', ['Sz_total'])

# Here we calculate the spectral gap by subtracting the energy for Sz=1 and Sz=0.
L = p[0].x
x = L**-1.0
gap = np.abs(p[0].y-p[1].y)

plt.plot(x, gap, 'x')

# We now perform a fit to the data.
# fit_order = 1 chooses a linear fit.
# For fit_order = 2, a quadratic fit is performed.
fit_order = 1
fit = np.polyfit(x, gap, fit_order)
fit_x = np.linspace(0, max(x), 100)
plt.plot(fit_x, np.polyld(fit)(fit_x), ':')

plt.xlabel('$1/L$')
plt.ylabel('$S^z=1$ gap')
```

Out[9]: <matplotlib.text.Text at 0x116465550>



Markup

- Double-click to edit, Shift-Enter to render
- Use LaTeX math annotations ($…$)

Source code

- Usual Python commands
- Execute with Shift-Enter

Inline figures from matplotlib (see commands at top of notebook)

ALPS applications

- **Classical Monte Carlo:**
  - Local and cluster update algorithms for spin systems [spinmc]
- **Exact Diagonalization**
  - Full [fulldiag] and sparse [sparsediag] diagonalization
- **Quantum Monte Carlo:**
  - Stochastic Series Expansion (SSE) [dirloop_sse]
  - Loop code [loop]
  - Continuous-time worm algorithm [worm and dwa]
  - Extended ensemble (quantum Wang-Landau) algorithm [qwl]
- **Tensor network methods:**
  - DMRG [dmrg]
  - Time-evolving block decimation [tebd]
  - Matrix-product state implementations [mps_X]
- **DMFT**

The QMC applications

- Which QMC code should you use?
  - https://alps.comp-phys.org/mediawiki/index.php/Comments:_which_code_to_choose_for_your_calculation
- Looper:
  - Only for highly symmetric models (U(1) x Z_2), but most efficient for those
- Directed Loop SSE:
  - More generic, high performance. Inefficient with a large number of states per site (soft-core bosons).
- Worm algorithm: Bose-Hubbard models
- Quantum Wang-Landau: Specific application for free energy & entropy calculations without thermodynamic integration

The DMRG applications

- DMRG:
  - Traditional DMRG code, good for simple *1d* Hamiltonians but inefficient in more general cases
- TEBD:
  - Specific to time evolution, special models, more dependencies
- MPS:
  - Generic, modern implementation in MPS language
  - Ground state & time evolution methods in the same infrastructure
  - High-performance (up to 10,000 states), efficient for two-dimensional systems

Code	**Time**
ALPS MPS (our new code)	16 sec
ALPS DMRG [1, 21]	73 sec
ITensor [27]	24 sec
OSMPS [28]	40 sec

- Reference: ***M. Dolfi et al, Comput. Phys. Commun. 185, 3430 (2014)***

Obtaining ALPS

http://alps.comp-phys.org/

Welcome to the ALPS project.

The **ALPS project** (**A**lgorithms and **L**ibraries for **P**hysics **S**imulations) is an open source effort aiming at providing high-end simulation codes for strongly correlated quantum mechanical systems as well as C++ libraries for simplifying the development of such code. ALPS strives to increase software reuse in the physics community.

Announcement:

ALPS 2.1 has been released

Community

Check back regularly to read the **latest news** and information on the **people** contributing to the project.

User Forum

Go here to **discuss** the ALPS libraries and applications with the community of developers. This is the place to address any questions you encounter while using any codes of the ALPS project.

Papers and Talks

Go here to **find papers** on the ALPS project, **talks** from the first user workshop and a list of papers citing the ALPS project.

Download and Installation

Useful information **how to install** the alps libraries and applications on workstations and clusters.

Tutorials

How to run ALPS applications

ALPS applications
latest/v2.2b - v2.1

Using ALPS libraries for your applications
latest/v2.2b - v2.1

Documentation

ALPS applications reference documentation

The ALPS libraries (2.1.0) API

The ALPS Python modules API

The HDF5 scheme documentation

Obtaining ALPS

http://alps.comp-phys.org/

Welcome to the ALPS project.

The **ALPS project** (**A**lgorithms and **L**ibraries for **P**hysics **S**imulations) is an open source effort aiming at providing high-end simulation codes for strongly correlated quantum mechanical systems as well as C++ libraries for simplifying the development of such code. ALPS strives to increase software reuse in the physics community.

Announcement:

ALPS 2.1 has been released

Community

Check back regularly to read the **latest news** and information on the **people** contributing to the project.

User Forum

Go here to **discuss** the ALPS libraries and applications with the community of developers. This is the place to address any questions you encounter while using any codes of the ALPS project.

Papers and Talks

Go here to **find papers** on the ALPS project, **talks** from the first user workshop and a list of papers citing the ALPS project.

Download and Installation

Useful information **how to install** the alps libraries and applications on workstations and clusters.

Tutorials

How to run ALPS applications

ALPS applications
latest/v2.2b - v2.1

Using ALPS libraries for your applications
latest/v2.2b - v2.1

Documentation

ALPS applications reference documentation
The ALPS libraries (2.1.0) API
The ALPS Python modules API
The HDF5 scheme documentation

Installation

Installation

- **Binary**:
  - Available for Linux, Mac OS X, and Windows
  - Relies on VisTrails for Python interpreter & libraries
  - Easiest installation, least flexibility

Installation

- **Binary**:
  - Available for Linux, Mac OS X, and Windows
  - Relies on VisTrails for Python interpreter & libraries
  - Easiest installation, least flexibility
- **Source installation**:
  - Available on Linux and Mac OS X, challenging on Windows
  - More flexibility:
    - Use your own version of Boost, Python, HDF5 and MPI
    - Recommended if you want to write code that links against ALPS libraries
    - Required on clusters

Installation

- **Binary**:
  - Available for Linux, Mac OS X, and Windows
  - Relies on VisTrails for Python interpreter & libraries
  - Easiest installation, least flexibility
- **Source installation**:
  - Available on Linux and Mac OS X, challenging on Windows
  - More flexibility:
    - Use your own version of Boost, Python, HDF5 and MPI
    - Recommended if you want to write code that links against ALPS libraries
    - Required on clusters
- **Versions**:
  - General recommendation: **most recent version possible!**
  - For the tutorials
    - For binary installation, use **pre-release 2.2.0b4**
    - For source installation, either **2.2.0b4 or SVN trunk**

License conditions

- ALPS license = variant of BSD license
- **Crucial condition**:
  - Use of **any ALPS** component requires citation of the ALPS papers: A.F. Albuquerque et al., J. of Magn. and Magn. Materials 310, 1187 (2007) and B. Bauer et al., J. Stat. Mech. (2011) P05001
  - Use of the DMFT codes additionally requires citation of E. Gull, P. Werner, S. Fuchs, B. Surer, T. Pruschke, and M. Troyer, Computer Physics Communications 182, 1078 (2011).
  - Use of the MPS codes requires additional citation of M. Dolfi et al., Computer Physics Communications 185, 3430 (2014).
- **For details, see CITATIONS.txt in source distribution**

Tutorials

- Will be done on lab workstations - **no installation necessary**
- **If you would like to continue working on it later, ask me for installation help**
- Today: **Two tutorials**
  - **Quantum Monte Carlo**
  - **Density Matrix Renormalization Group**

Welcome to the ALPS project.

The **ALPS project** (**A**lgorithms and **L**ibraries for **P**hysics **S**imulations) is an open source effort aiming at providing high-end simulation codes for strongly correlated quantum mechanical systems as well as C++ libraries for simplifying the development of such code. ALPS strives to increase software reuse in the physics community.

Announcement:

ALPS 2.1 has been released

Community

Check back regularly to read the **latest news** and information on the **people** contributing to the project.

User Forum

Go here to **discuss** the ALPS libraries and applications with the community of developers. This is the place to address any questions you encounter while using any codes of the ALPS project.

Papers and Talks

Go here to **find papers** on the ALPS project, **talks** from the first user workshop and a list of papers citing the ALPS project.

Download and Installation

Useful information **how to install** the alps libraries and applications on workstations and clusters.

Tutorials

How to run ALPS applications

ALPS applications
latest/v2.2b - v2.1

Using ALPS libraries for your applications
latest/v2.2b - v2.1

Documentation

ALPS applications reference documentation

The ALPS libraries (2.1.0) API

The ALPS Python modules API

The HDF5 scheme documentation

Tutorials

- Will be done on lab workstations - **no installation necessary**
- **If you would like to continue working on it later, ask me for installation help**
- Today: **Two tutorials**
  - **Quantum Monte Carlo**
  - **Density Matrix Renormalization Group**

Welcome to the ALPS project.

The **ALPS project** (**A**lgorithms and **L**ibraries for **P**hysics **S**imulations) is an open source effort aiming at providing high-end simulation codes for strongly correlated quantum mechanical systems as well as C++ libraries for simplifying the development of such code. ALPS strives to increase software reuse in the physics community.

Announcement:

ALPS 2.1 has been released

Community

Check back regularly to read the **latest news** and information on the **people** contributing to the project.

User Forum

Go here to **discuss** the ALPS libraries and applications with the community of developers. This is the place to address any questions you encounter while using any codes of the ALPS project.

Papers and Talks

Go here to **find papers** on the ALPS project, **talks** from the first user workshop and a list of papers citing the ALPS project.

Download and Installation

Useful information **how to install** the alps libraries and applications on workstation[illegible] clusters.

Tutorials

How to run ALPS applic[illegible]

ALPS applications
latest/v2.2b - v2.1

Using ALPS libraries fo[illegible]pplications
latest/v2.2b - v2.1

Documentation

ALPS applications reference documentation
The ALPS libraries (2.1.0) API
The ALPS Python modules API
The HDF5 scheme documentation

Tutorials

- DWA-01 Simulating the Bose Hubbard model using dwa QMC code (revisiting tutorial MC-05)
- DWA-02 Density profile of a 3D optical lattice in a harmonic trap
- DWA-03 Time-of-flight images of a 3D optical lattice in a harmonic trap

Exact diagonalization

- ED-01 Sparse Diagonalization (Lanczos)
- ED-02 Spin gaps of 1D quantum systems
- ED-03 Spectra of 1D quantum systems
- ED-04 Conformal field theory description of 1D critical spectra
- ED-05 Phase transition in a frustrated spin chain
- ED-06 Full Diagonalization

Density Matrix Renormalization Group (DMRG)

- DMRG-01 Density Matrix Renormalization Group Introduction
- DMRG-02 Calculating gaps
- DMRG-03 Calculating local observables
- DMRG-04 Calculating correlations

Dynamical Mean Field Theory (DMFT) solvers

- DMFT-01 An introduction to DMFT
- DMFT-02 CT-HYB: the CT-HYB QMC solver
- DMFT-03 CT-INT: the CT-INT QMC solver
- DMFT-04 Mott Transition
- DMFT-05 Orbitally Selective Mott Transition
- DMFT-06 Paramagnetic metal
- DMFT-07 The Hirsch-Fye solver
- DMFT-08 Setting a particular lattice

Time-Evolving Block Decimation (TEBD)

- TEBD-01 Quenches in the hardcore boson model
- TEBD-02 Time Evolution of a domain wall in the XX model

Trieste Summer School 2015 Tutorials

- Trieste tutorials

ALPS examples

The ALPS examples section provide examples of real simulations, which require more computing resources to run

/afs/ictp.it/public/b/bbauer/

Tutorials

- DWA-01 Simulating the Bose Hubbard model using dwa QMC code (revisiting tutorial MC-05)
- DWA-02 Density profile of a 3D optical lattice in a harmonic trap
- DWA-03 Time-of-flight images of a 3D optical lattice in a harmonic trap

Exact diagonalization

- ED-01 Sparse Diagonalization (Lanczos)
- ED-02 Spin gaps of 1D quantum systems
- ED-03 Spectra of 1D quantum systems
- ED-04 Conformal field theory description of 1D critical spectra
- ED-05 Phase transition in a frustrated spin chain
- ED-06 Full Diagonalization

Density Matrix Renormalization Group (DMRG)

- DMRG-01 Density Matrix Renormalization Group Introduction
- DMRG-02 Calculating gaps
- DMRG-03 Calculating local observables
- DMRG-04 Calculating correlations

Dynamical Mean Field Theory (DMFT) solvers

- DMFT-01 An introduction to DMFT
- DMFT-02 CT-HYB: the CT-HYB QMC solver
- DMFT-03 CT-INT: the CT-INT QMC solver
- DMFT-04 Mott Transition
- DMFT-05 Orbitally Selective Mott Transition
- DMFT-06 Paramagnetic metal
- DMFT-07 The Hirsch-Fye solver
- DMFT-08 Setting a particular lattice

Time-Evolving Block Decimation (TEBD)

- TEBD-01 Quenches in the hardcore boson model
- TEBD-02 Time Evolution of a domain wall in the XX model

Trieste Summer School 201[illegible] Tutorials

- Trieste tutorials

ALPS examples

The ALPS examples section provide examples of real simulations, which require more computing resources to run

/afs/ictp.it/public/b/bbauer/