Observables and correlations: new analysis concepts in ESPRESSO 3.1

Peter Košovan, Stefan Kesselheim

Institute for Computational Physics, University of Stuttgart

Espresso Summer School, 9.10.2012

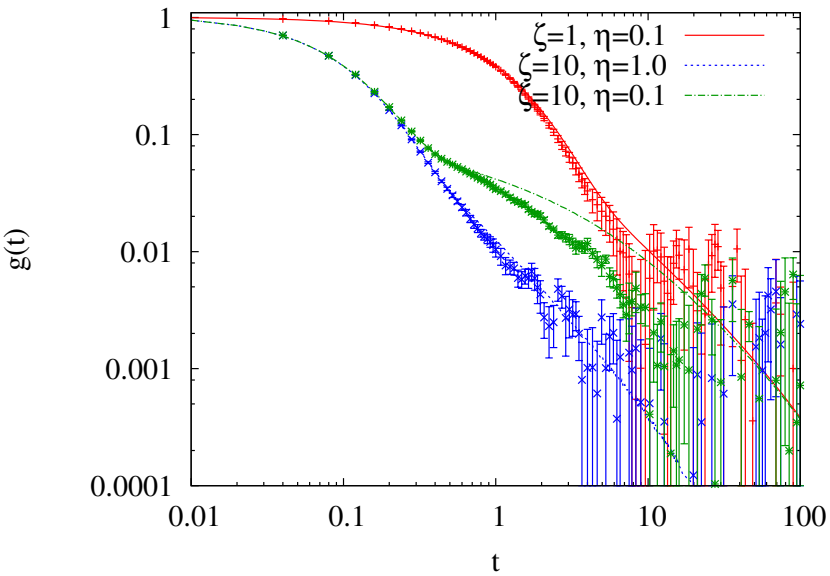
Green-Kubo method for Diffusion coefficent

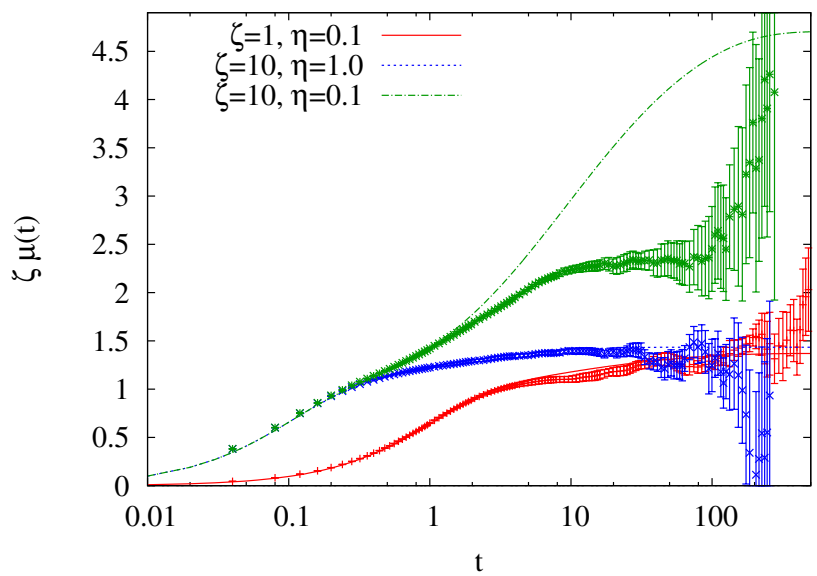
Velocity autocorrelation function:

$$g(\tau) = \langle \vec{v}(t) \cdot \vec{v}(t+\tau) \rangle_t$$

Diffusion coefficient:

$$D \simeq \int_0^\infty g(\tau)\, \mathrm{d}\tau$$





Analysis concepts before ESPRESSO 3.1

```
for {set i 0} {$i < $n_steps} {incr i} {
  ...
  integrate $int_steps
  set Rg [analyze rg]
  analyze append
  store_configuration
  ...
}
set Rg_av [analyze <rg>]
analyze_stored
```

Problems

- Analysis at TCL level slow
- Saving configurations can be unaffordable
- Observables can't be used by other functions in ESPRESSO core

Wishlist

- Create a generic concept of observables which can be used by different parts of the simulation kernel
- Update observables when needed without explicit user call
- Enable passing of observables to other routines for further processing, without recalculation
- Enable efficient computation of time-correlation functions
- ESPRESSO philosophy
  - extensibility
  - interface for implementaiton of new observables
- On-the-fly analysis: averaging, robust error estimation and automated runtime control

Time correlation functions

$$C(\tau) = \langle O(F_1(t), F_2(t+\tau)) \rangle_t$$

- F_1, F_2: Observables
  "Rule to calculate an n-dimensional vector from the state of a simulation"
- $O(x, y)$: Correlation operation.
  "Function that calculates an m-dimensional vector from two n-dimensional vectors"

Examples

- Velocity Autocorrelation function
  F = "particle velocity" O = "scalar product"
- Mean-Square-Displacement
  F = "particle positions" O = "square distance"

Core observable concepts in ESPResSo 3.1

- Define an observable:

```
set pos_id [observable new particle_positions type 0]
```

- Get its value:

```
set current_positions [observable $pos_id print]
```

- Create a correlation:

```
set msd [corelation new obs $pos_id dt $dt \
corr_operation square_distance tau_max 10]
```

- Update correlation manually:

```
correlation $msd update
```

- Or – tell ESPResSo to update it automatically:

```
correlation $msd autoupdate start
```

Calculating correlations

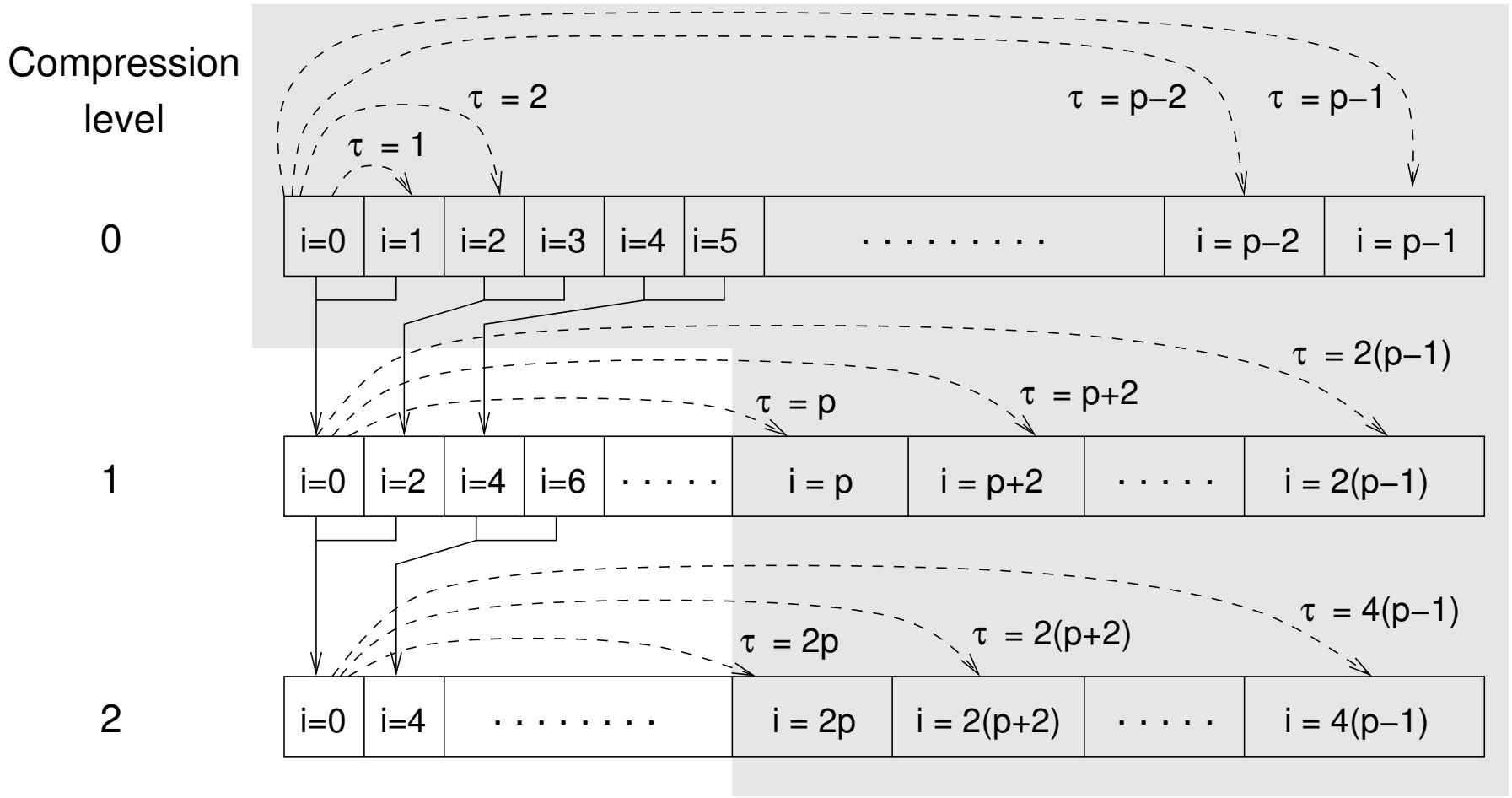
Naive implementation

- Sample every time step
- Compute $C(\tau)$ for $\tau \in [\tau_{\text{min}}, \tau_{\text{max}}]$
- Correlate all available pairs of observables

Problems

- Complexity $\mathcal{O}(\tau_{\text{max}}/\tau_{\text{min}}^2)$
- Feasible for $(\tau_{\text{max}}/\tau_{\text{min}}) \lesssim 10^2$
- Implemented *e. g.* in `analyze msd`
- Requires storing the whole history

Multiple tau correlator algorithm



Features of the ESPRESSO correlator

Efficient, robust, flexible

- Generic interface:
  - Can use any implemented observable
  - Number of correlation operations
- Almost no overhead
- Complexity $\mathcal{O}(\log(\tau_{\max}/\tau_{\min}))$
- Space for history buffers $\mathcal{O}(\log(\tau_{\max}/\tau_{\min}))$
- Correlator parameters critically effect efficiency, statistical quality and density of data in the result (see user manual for details)

A glimpse at the future

Modifications under development

- Generalized observables:
  - Depend on other observables
  - Depend on history
- History-dependent observables:
  average, variance, error . . . correlation as a special case

Example: Average of an observable

```
set av [observable new average $original_obs_id]
observable $av update
```

- `update` Recalc original observable and add it to the buffer
- `recalculate` Compute average from the current buffer
- `reset` Empty the history buffer

Last but not least

Observables are still a developing project

- We appreciate any feedback, especially on
  - Intuitiveness of use (or lack of it)
  - Suggestions for further features
- If you think of implementing a new analysis method for ESPRESSO, consider implementing it as an observable.

Have fun with observables and correlations!