Hands-on advanced numerical methods for quantum many-body dynamics

A tour through numerical methods for simulating large-scale quantum physics (classically)
Motto: Do it from scratch to better understand quantum physics and classical limitations
Structure: Theory lecture part + tutorial-style

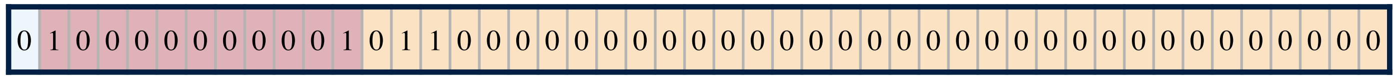
- Language used: Julia, **https://julialang.org/** (open source, easy, fast linear algebra)

- **Outline** (may vary)

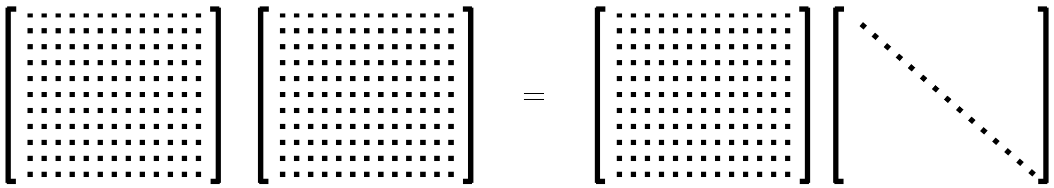
1. **20/10:** Basic concepts: Numbers on computers, basic exact diagonalization (ED)
   Tutorial: ED on a simple spin model
2. **21/10:** A better ED, sparse matrices, Krylov space. Open systems.
   Tutorial: Spin-model simulations using Krylov space
3. **22/10**: Mean-field, Runge-Kutta
   Tutorial: A mean-field simulation of the transverse Ising model
4. **23/10:** How to go beyond: Matrix product states

Recap

- We discussed how to represent numbers as bits, and associated relative machine precisions ~1e-16

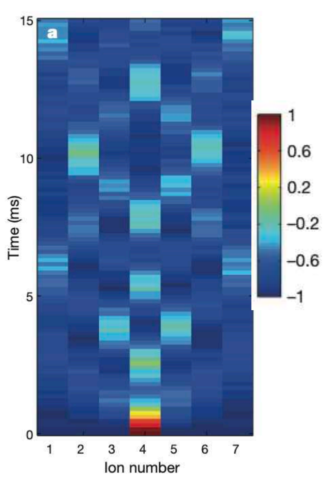


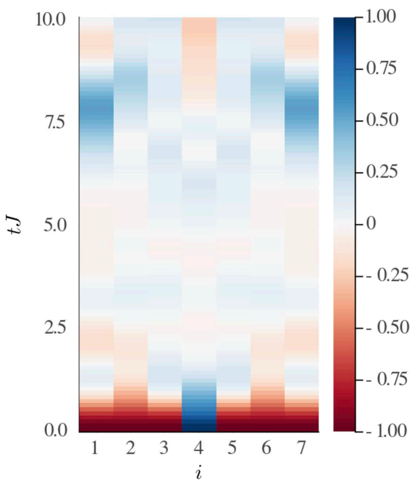
- We describe systems with state-vectors (not only in QM). If the problem is linear (QM), an exact diagonalization of the Hamiltonian allows to simulate arbitrary time-evolution



- In tutorial style we have seen how to construct many-body spin (or other) operators using Kronecker product functions. We used this to simulate dynamics in a transverse Ising spin model (which can e.g. be realized with trapped ions)

$$\hat{\sigma}_i^- = \begin{pmatrix} 1 & & \\ & 1 & \\ & & \ddots \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & & \\ & 1 & \\ & & \ddots \end{pmatrix}$$





- It’s important to use sparse matrices.

Tutorial: Constructing a Hamiltonian and simulating dynamics

Full version

```
julia> @time begin; sm = [0 1; 0 0]; sxs, szs = build_spin_ops(sm, 10); H = build_hamiltonian(1, 3, 1, sxs, szs); end;
  0.830853 seconds (821 allocations: 1.397 GiB, 48.33% gc time)
```

Sparse version

```
julia> @time begin; sxs, szs = build_sparse_spin_ops(sparse([0 1; 0 0]), 10); H = build_sparse_hamiltonian(1, 3, 1, sxs, szs); end;
  0.007831 seconds (2.78 k allocations: 10.354 MiB, 59.62% gc time)
```

- Much more memory and time-efficient to construct the Hamiltonian!
- **Problem:** A full diagonalization does not make use of the sparsity!

$$\hat{V}^\dagger \hat{H} \hat{V} = \hat{E} \quad \Leftrightarrow \quad \hat{H}\hat{V} = \hat{V}\hat{E} \quad \Leftrightarrow \quad \hat{H} = \hat{V}\hat{E}\hat{V}^\dagger$$

V would be a full matrix!

We have to be smarter to make use of sparse matrices in the time-evolution!

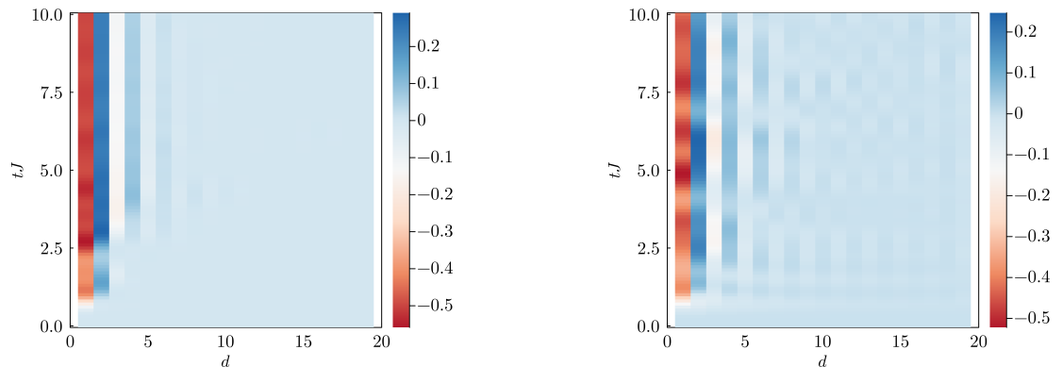
This time

- **Part 1:** Krylov space: Evolution with sparse Hamiltonians

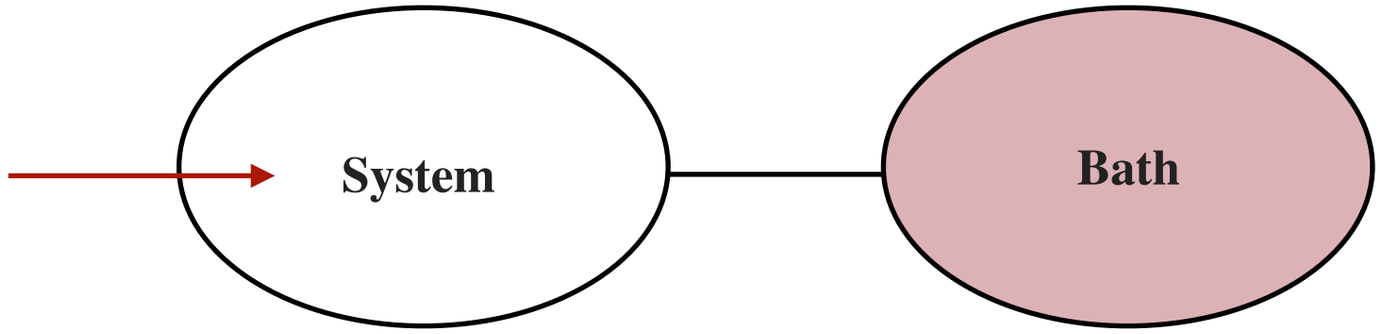
$$\text{span}\left(\hat{A}^0 |\psi_0\rangle, \hat{A}^1 |\psi_0\rangle, \hat{A}^2 |\psi_0\rangle, \ldots, \hat{A}^{m-1} |\psi_0\rangle\right)$$

Tutorials: Power method/implement an Arnoldi iteration

- **Tutorial:** Spin-model simulations using Krylov space



- **Part 2:** Basics for open-system simulations



Krylov space

- The goal is to compute dynamics of some initial state

$$\frac{d}{dt}|\psi\rangle = -\mathrm{i}\hat{H}|\psi\rangle \qquad |\psi(t)\rangle = \mathrm{e}^{-\mathrm{i}\hat{H}t}\,|\psi_0\rangle \qquad \ldots \textit{with a generally very } \textbf{\textit{sparse Hamiltonian}}$$

- We could just use standard ODE solvers (e.g. Runge-Kutta, later), but it would be overkill, since:

i) The problem is linear
ii) The Hamiltonian is time-independent

- It would be good to compute the matrix exponential directly
- **Problem:** A matrix exponentiation turns the sparse Hamiltonian into a full matrix

E.g.: Compute matrix exponential by full diagonalization

$$\hat{H}\,|E_n\rangle = E_n\,|E_n\rangle \qquad \hat{H} = \sum_n E_n\,|E_n\rangle\langle E_n| \qquad \mathrm{e}^{-\mathrm{i}\hat{H}t} = \sum_{k=0}\frac{(-\mathrm{i}t\hat{H})^k}{k!} = \sum_n \mathrm{e}^{-\mathrm{i}tE_n}\,|E_n\rangle\langle E_n|$$

sparse ~D elements — *D energy eigenstates* — *each ~D elements* — *in general: Full DxD matrix … not sparse :(*

- **On the other hand:** We're only interested in the state after applying the matrix exponential

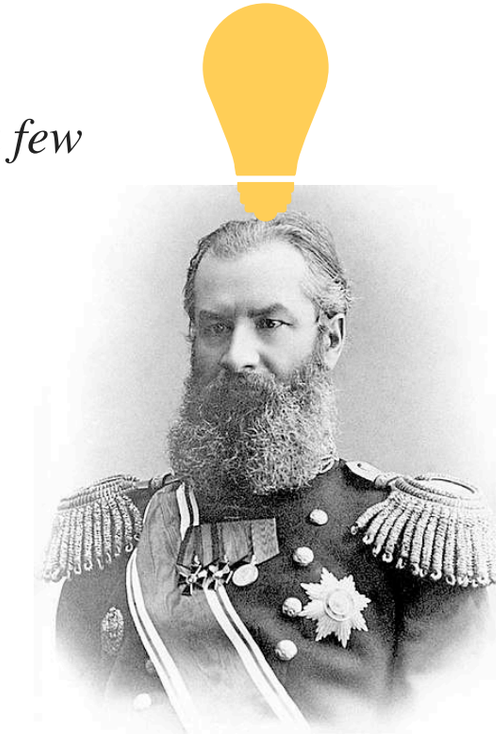
$$\mathrm{e}^{-\mathrm{i}\hat{H}t}\,|\psi_0\rangle = \sum_{k=0}\frac{(-\mathrm{i}t\hat{H})^k}{k!}\,|\psi_0\rangle \equiv \sum_{k=0}\frac{\hat{A}^k}{k!}\,|\psi_0\rangle$$

Maybe we only need a few terms in this sum?

Krylov space

$$e^{-i\hat{H}t}\,|\psi_0\rangle = \sum_{k=0} \frac{(-it\hat{H})^k}{k!}\,|\psi_0\rangle \equiv \sum_{k=0} \frac{\hat{A}^k}{k!}\,|\psi_0\rangle$$

Maybe we only need a few terms in this sum?



Aleksey Nikolaevich Krylov (1863 - 1945)

- Starting from the initial state, a good idea seems to be using the following space:

Krylov space: $\text{span}\left(\hat{A}^0\,|\psi_0\rangle\,, \hat{A}^1\,|\psi_0\rangle\,, \hat{A}^2\,|\psi_0\rangle\,, \ldots, \hat{A}^{m-1}\,|\psi_0\rangle\right)$

(here formulated in bra/ket notation, but concept is general)

Krylov wrote about 300 papers and books. They span a wide range of topics, including shipbuilding, magnetism, artillery, mathematics, astronomy, and geodesy.

- **Problem:** The states are not orthogonal (in fact with increasing k they become more and more parallel)

Example, let's for a sec take the matrix to be a hermitian Hamiltonian $\hat{A} = \hat{H}$

Take an initial state expanded in eigenstates:

$$|\psi_0\rangle = \sum_n c_n\,|E_n\rangle \qquad \hat{H}\,|E_n\rangle = E_n\,|E_n\rangle \qquad E_1 > E_2 > E_3 > \ldots$$

k-th Krylov state $|\psi_k\rangle \equiv \hat{H}^k\,|\psi_0\rangle = \sum_n c_n E_n^k\,|E_n\rangle$

All high k states will be parallel to $|E_1\rangle$

- When normalizing and $k \gg 1$

$$c_1 E_1^k \gg c_2 E_2^k$$

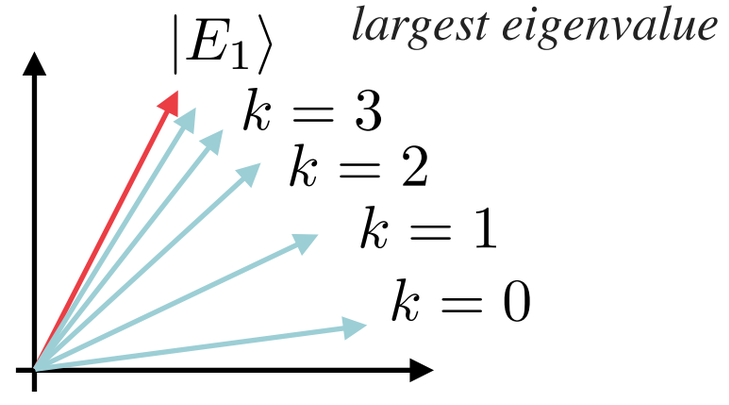
$$\frac{|\psi_k\rangle}{\||\psi_k\rangle\|} \approx |E_1\rangle$$

Also: *Repeated application gives the highest energy state! Known as* ***power method***

… assuming $c_1 \neq 0$

Power method

```
julia> ev = eigvecs(H)[:, end]
1024-element Vector{Float64}:
 0.4369482662387492
 0.1801701648057189 9
 0.1130179795824155 8
 0.1019360648791765 6
```




```
julia> vec_it = rand(1024); vec_it ./= norm(vec_it); abs(vec_it' * ev)^2
0.1988702689126 62
```

```
julia> vec_it = H * vec_it; vec_it ./= norm(vec_it); abs(vec_it' * ev)^2
0.3910949616254 2984
```

```
julia> vec_it = H * vec_it; vec_it ./= norm(vec_it); abs(vec_it' * ev)^2
0.9970319044046937
```

Krylov space - Arnoldi iteration

- **Krylov space:** $\text{span}\left(\hat{A}^0 |\psi_0\rangle, \hat{A}^1 |\psi_0\rangle, \hat{A}^2 |\psi_0\rangle, \ldots, \hat{A}^{m-1} |\psi_0\rangle\right)$

  (here formulated in bra/ket notation, but concept is general)

- **Problem:** The states are not orthogonal (in fact with increasing *k* they become more and more parallel)

- **Arnoldi iteration:** Create an orthonormal basis from the Krylov vectors (using improved/modified Gram-Schmidt)

$|\psi_0\rangle = |\psi_0\rangle / \| |\psi_0\rangle \|$ *(normalization just in case)*

1st iteration

$$|\psi_1\rangle \equiv \hat{A} |\psi_0\rangle$$
$$h_{1,1} \equiv \langle\psi_0|\psi_1\rangle \quad |\psi_1\rangle \equiv |\psi_1\rangle - h_{1,1} |\psi_0\rangle$$
$$h_{2,1} \equiv \| |\psi_1\rangle \| \quad |\psi_1\rangle \equiv |\psi_1\rangle / h_{2,1}$$

ith - iteration

$$|\psi_i\rangle \equiv \hat{A} |\psi_{i-1}\rangle$$
$$h_{1,i} \equiv \langle\psi_0|\psi_i\rangle \quad |\psi_i\rangle \equiv |\psi_i\rangle - h_{1,i} |\psi_0\rangle$$
$$h_{2,i} \equiv \langle\psi_1|\psi_i\rangle \text{★} \quad |\psi_i\rangle \equiv |\psi_i\rangle - h_{2,i} |\psi_1\rangle$$
$$\ldots$$
$$h_{i,i} \equiv \langle\psi_{i-1}|\psi_i\rangle \quad |\psi_i\rangle \equiv |\psi_i\rangle - h_{i+1,i} |\psi_{i-1}\rangle$$
$$h_{i+1,1} \equiv \| |\psi_i\rangle \| \quad |\psi_i\rangle \equiv |\psi_i\rangle / h_{i+1,1}$$

Walter Edwin Arnoldi
(American engineer/mathematician)
(1917 - 1995)

Notation: $\equiv$

means "define"

…as in code:

```
= # not ==
```

★ *modified GS:*
use already updated state

Krylov space - Arnoldi iteration

- **Krylov space:** $\text{span}\left(\hat{A}^0 |\psi_0\rangle, \hat{A}^1 |\psi_0\rangle, \hat{A}^2 |\psi_0\rangle, \ldots, \hat{A}^{m-1} |\psi_0\rangle\right)$

$$|\psi_i\rangle \equiv \hat{A} |\psi_{i-1}\rangle \qquad \textit{ith - iteration}$$

$$h_{1,i} \equiv \langle\psi_0|\psi_i\rangle \quad |\psi_i\rangle \equiv |\psi_i\rangle - h_{1,i} |\psi_0\rangle$$

$$h_{2,i} \equiv \langle\psi_1|\psi_i\rangle \quad |\psi_i\rangle \equiv |\psi_i\rangle - h_{2,i} |\psi_1\rangle$$

$$\ldots$$

$$h_{i,i} \equiv \langle\psi_{i-1}|\psi_i\rangle \quad |\psi_i\rangle \equiv |\psi_i\rangle - h_{i+1,i} |\psi_{i-1}\rangle$$

$$h_{i+1,1} \equiv \| |\psi_i\rangle \| \quad |\psi_i\rangle \equiv |\psi_i\rangle / h_{i+1,1}$$

after i iterations, we have i+1 orthogonal vectors, i.e. we have the "semi-unitary" matrix

$$\mathbf{Q}_i = \begin{pmatrix} |\psi_0\rangle & |\psi_1\rangle & \ldots & |\psi_i\rangle \end{pmatrix}$$

$$D \times (i+1) \qquad \mathbf{Q}_i^\dagger \mathbf{Q}_i = \mathbb{1}$$

- $\mathbf{Q}_i$ is a projection matrix to the Krylov basis

What is the matrix **h?**

$$\mathbf{h}_i = \begin{pmatrix} h_{1,1} & h_{1,2} & \ldots & h_{1,i} \\ h_{2,1} & h_{2,2} & \ldots & h_{2,i} \\ 0 & h_{3,2} & \ldots & h_{3,i} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & h_{i+1,i} \end{pmatrix}$$

$$(i+1) \times i$$

Upper Hessenberg form

The iteration can be written in the matrix form:

$$\hat{A}\mathbf{Q}_i = \mathbf{Q}_{i+1}\mathbf{h}_i \qquad \textit{(exercise: show)}$$

Example: i=1

$$|\psi_1\rangle = \left(\hat{A} |\psi_0\rangle - h_{1,1} |\psi_0\rangle\right) / h_{2,1}$$

$$\hat{A} |\psi_0\rangle = h_{1,1} |\psi_0\rangle + h_{2,1} |\psi_1\rangle$$

$$\hat{A}(|\psi_0\rangle) = \begin{pmatrix} |\psi_0\rangle & |\psi_1\rangle \end{pmatrix} \begin{pmatrix} h_{1,1} \\ h_{2,1} \end{pmatrix}$$

Krylov space - Arnoldi iteration

- **Krylov space:** $\text{span}\left(\hat{A}^0\left|\psi_0\right\rangle, \hat{A}^1\left|\psi_0\right\rangle, \hat{A}^2\left|\psi_0\right\rangle, \ldots, \hat{A}^{m-1}\left|\psi_0\right\rangle\right)$

ith - iteration

$$\left|\psi_i\right\rangle \equiv \hat{A}\left|\psi_{i-1}\right\rangle$$

$$h_{1,i} \equiv \langle\psi_0|\psi_i\rangle \quad \left|\psi_i\right\rangle \equiv \left|\psi_i\right\rangle - h_{1,i}\left|\psi_0\right\rangle$$

$$h_{2,i} \equiv \langle\psi_1|\psi_i\rangle \quad \left|\psi_i\right\rangle \equiv \left|\psi_i\right\rangle - h_{2,i}\left|\psi_1\right\rangle$$

$$\ldots$$

$$h_{i,i} \equiv \langle\psi_{i-1}|\psi_i\rangle \quad \left|\psi_i\right\rangle \equiv \left|\psi_i\right\rangle - h_{i+1,i}\left|\psi_{i-1}\right\rangle$$

$$h_{i+1,1} \equiv \| \left|\psi_i\right\rangle \| \quad \left|\psi_i\right\rangle \equiv \left|\psi_i\right\rangle / h_{i+1,1}$$

after i iterations, we have i+1 orthogonal vectors, i.e. we have the "semi-unitary" matrix

$$\mathbf{Q}_i = \begin{pmatrix} \left|\psi_0\right\rangle & \left|\psi_1\right\rangle & \ldots & \left|\psi_i\right\rangle \end{pmatrix}$$

$$D \times (i+1) \qquad \mathbf{Q}_i^\dagger \mathbf{Q}_i = \mathbb{1}$$

- What is the matrix **h?**

The iteration can be written in the matrix form:

$$\hat{A}\mathbf{Q}_i = \mathbf{Q}_{i+1}\mathbf{h}_i$$

Terminating means:

$$\mathbf{Q}_{i+1} = \begin{pmatrix} \left|\psi_0\right\rangle & \left|\psi_1\right\rangle & \ldots & \left|\psi_i\right\rangle & \cancel{\left|\psi_{i+1}\right\rangle} \end{pmatrix}$$

$$\mathbf{Q}_{i+1}\mathbf{h}_i \approx \mathbf{Q}_i\tilde{\mathbf{h}}_i$$

$$\tilde{\mathbf{h}}_i = \begin{pmatrix} h_{1,1} & h_{1,2} & \ldots & h_{1,i} \\ h_{2,1} & h_{2,2} & \ldots & h_{2,i} \\ 0 & h_{3,2} & \ldots & h_{3,i} \\ \vdots & \vdots & \ddots & \vdots \\ \cancel{0} & \cancel{0} & \ldots & h_{i+1,i} \end{pmatrix}$$

- Then: $\hat{A}\mathbf{Q}_i \approx \mathbf{Q}_i\tilde{\mathbf{h}}_i$ $\qquad \tilde{\mathbf{h}}_i \approx \mathbf{Q}_i^\dagger \hat{A}\mathbf{Q}_i$ $\qquad \hat{A} \approx \mathbf{Q}_i\tilde{\mathbf{h}}_i\mathbf{Q}_i^\dagger$

- The matrix **h** is an approximation of the matrix **A** in (the much smaller) Krylov space!

Krylov space - Arnoldi iteration

- **Krylov space:** $\text{span}\left(\hat{A}^0|\psi_0\rangle, \hat{A}^1|\psi_0\rangle, \hat{A}^2|\psi_0\rangle, \ldots, \hat{A}^{m-1}|\psi_0\rangle\right)$

$$|\psi_i\rangle \equiv \hat{A}|\psi_{i-1}\rangle \qquad \textit{ith - iteration}$$

$$h_{1,i} \equiv \langle\psi_0|\psi_i\rangle \quad |\psi_i\rangle \equiv |\psi_i\rangle - h_{1,i}|\psi_0\rangle$$

$$h_{2,i} \equiv \langle\psi_1|\psi_i\rangle \quad |\psi_i\rangle \equiv |\psi_i\rangle - h_{2,i}|\psi_1\rangle$$

$$\ldots$$

$$h_{i,i} \equiv \langle\psi_{i-1}|\psi_i\rangle \quad |\psi_i\rangle \equiv |\psi_i\rangle - h_{i+1,i}|\psi_{i-1}\rangle$$

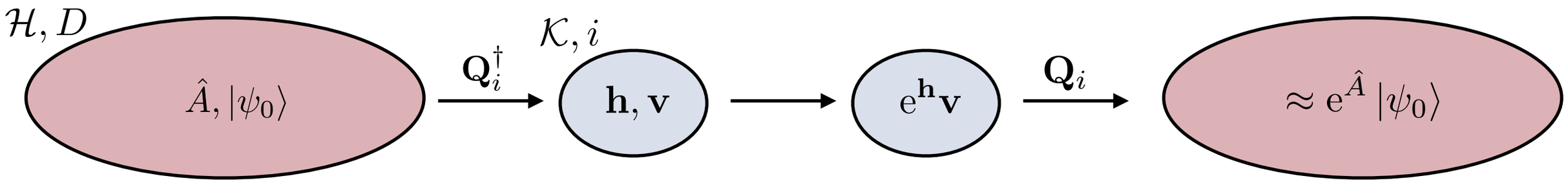
$$h_{i+1,1} \equiv \| |\psi_i\rangle \| \quad |\psi_i\rangle \equiv |\psi_i\rangle / h_{i+1,1}$$

after i iterations, we have i+1 orthogonal vectors, i.e. we have the "semi-unitary" matrix

$$\mathbf{Q}_i = \begin{pmatrix} |\psi_0\rangle & |\psi_1\rangle & \ldots & |\psi_i\rangle \end{pmatrix}$$

$$D \times (i+1) \qquad \mathbf{Q}_i^\dagger \mathbf{Q}_i = \mathbb{1}$$

- The matrix **h** is an approximation of the matrix **A** in (the much smaller) Krylov space $\mathcal{K}$!
- The idea is to now do exact diagonalization of the small matrix and then to transform it back to the full space:



Remark: For hermitian matrices, e.g. $\hat{A} = \hat{H}$ the matrix **h** is even only tri-diagonal. Then the iteration to obtain **h** is even simpler, it's called **Lanczos** iteration. It's very useful to compute a few eigenvalues and eigenstates of very large sparse Hamiltonians.

$$\mathbf{h}_i = \begin{pmatrix} h_{1,1} & h_{1,2} & 0 & 0 & \ldots \\ h_{2,1} & h_{2,2} & h_{2,3} & 0 & \ldots \\ 0 & h_{3,2} & h_{3,2} & h_{3,2} & \ldots \\ 0 & 0 & h_{4,3} & h_{4,3} & \ldots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

Tutorial: Arnoldi iteration

- **Goal:** compute $\mathrm{e}^{-\mathrm{i}\hat{H}t}\left|\psi_0\right\rangle = \mathrm{e}^{\hat{A}t}\left|\psi_0\right\rangle$

$$
\begin{aligned}
&\left|\psi_i\right\rangle \equiv \hat{A}\left|\psi_{i-1}\right\rangle \\
&\quad h_{1,i} \equiv \langle\psi_0|\psi_i\rangle \quad \left|\psi_i\right\rangle \equiv \left|\psi_i\right\rangle - h_{1,i}\left|\psi_0\right\rangle \\
&\quad h_{2,i} \equiv \langle\psi_1|\psi_i\rangle \quad \left|\psi_i\right\rangle \equiv \left|\psi_i\right\rangle - h_{2,i}\left|\psi_1\right\rangle \\
&\qquad\qquad \cdots \\
&\quad h_{i,i} \equiv \langle\psi_{i-1}|\psi_i\rangle \quad \left|\psi_i\right\rangle \equiv \left|\psi_i\right\rangle - h_{i+1,i}\left|\psi_{i-1}\right\rangle \\
&h_{i+1,1} \equiv \| \left|\psi_i\right\rangle \| \quad \left|\psi_i\right\rangle \equiv \left|\psi_i\right\rangle / h_{i+1,1}
\end{aligned}
$$

$$
\mathbf{Q}_i = \begin{pmatrix} \left|\psi_0\right\rangle & \left|\psi_1\right\rangle & \cdots & \left|\psi_i\right\rangle \end{pmatrix}
$$

For the matrix exponential applied to $\left|\psi_0\right\rangle$

… approximation is formally

$$
\mathrm{e}^{\hat{A}}\left|\psi_0\right\rangle \approx \mathbf{Q}_m \mathrm{e}^{\mathbf{h}} \mathbf{Q}_m^{\dagger}\left|\psi_0\right\rangle
$$

… but since $\left|\psi_0\right\rangle$ *is the first row in* $\mathbf{Q}_m^{\dagger}$*, and all other rows are orthogonal, we just need to take the first column of* $\mathrm{e}^{\mathbf{h}}$

```
# Compute approximation to exp(A)*psi0
# ... with m Krylov basis states
function arnoldi_exp(A, psi0, m)

    D = length(psi0)

    Q = similar(A, D, m) # the projection matrix
    h = zeros(eltype(A), m, m) # Krylov projection of A

    Q[:,1] = psi0   # assumed normalized
    for ii = 1:(m-1)
        psi_i = A*Q[:,ii]
        for jj = 1:ii
            h[jj,ii] = Q[:,jj]' * psi_i
            psi_i -= h[jj,ii] .* Q[:,jj]
        end
        h[ii+1, ii] = norm(psi_i)
        Q[:,ii+1] = psi_i ./ h[ii+1, ii]
    end

    # now return the matrix exponential
    return Q * exp(h)[:,1]

end
```

Tutorial: Arnoldi iteration

- Test with a random Hamiltonian $\hat{H}$

$$\hat{A} = -\mathrm{i}\Delta t \hat{H}$$

```
D = 1000; m = 20
H = rand(ComplexF64, D,D); H += H'
A = -1im .* 0.1 .* H

psi0 = rand(ComplexF64, D); psi0 ./= norm(psi0)

@time psi_ed = exp(A)*psi0
@time psi_krylov = arnoldi_exp(A, psi0, m)
@show abs(psi_ed' * psi_krylov)^2
```

```
  0.676812 seconds (16 allocations: 91.584 MiB)
  0.007632 seconds (831 allocations: 12.931 MiB)
abs(psi_ed' * psi_krylov) ^ 2 = 0.9999999999999936
```

Krylov approximation is already exact up to machine precision with m ~ 20. Speed-up of two orders of magnitude. For sparse matrices even more!

When does it go bad?

```
julia> psi_krylov = arnoldi_exp(A, psi0, 20); abs(psi_ed' * psi_krylov)^2
1.000000000000004

julia> psi_krylov = arnoldi_exp(A, psi0, 10); abs(psi_ed' * psi_krylov)^2
1.0000013088710726

julia> psi_krylov = arnoldi_exp(A, psi0, 4); abs(psi_ed' * psi_krylov)^2
1.1542019279919236
```

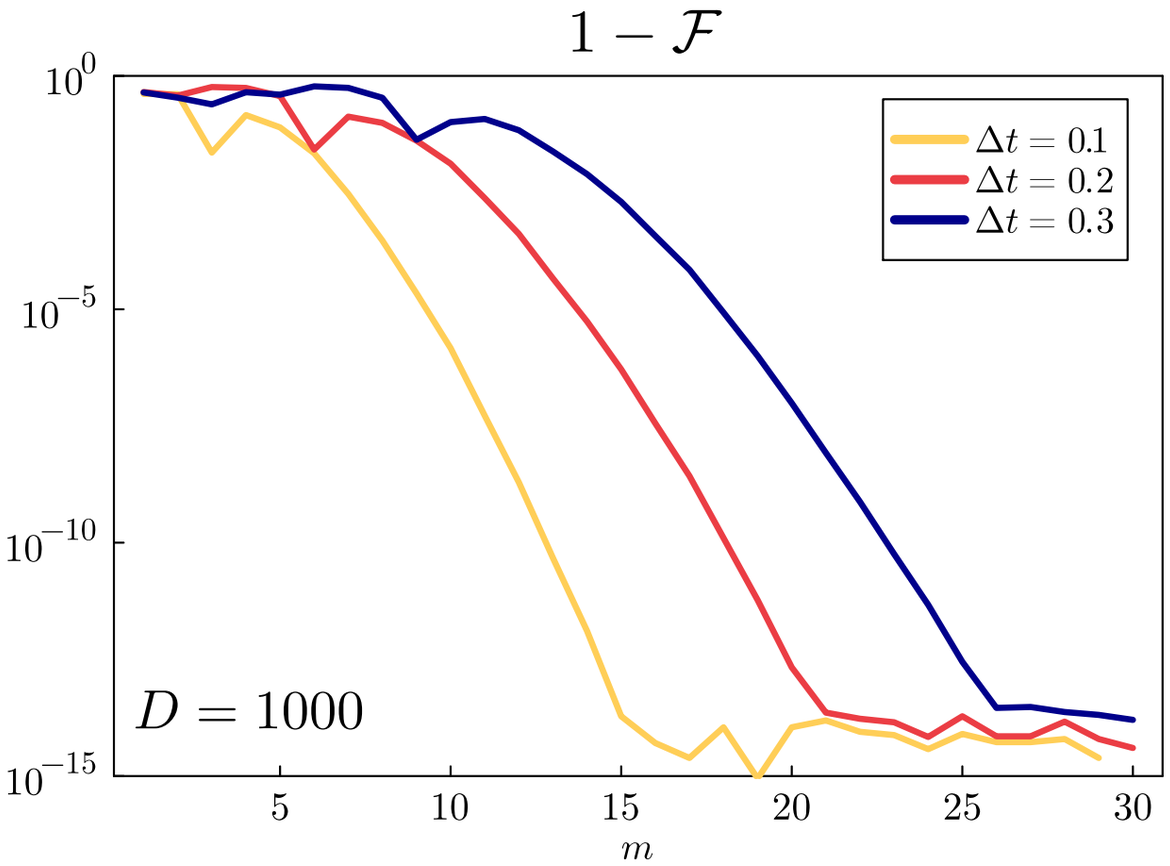
Smaller dt:

```
julia> A = -1im * 0.01 * H
```

```
julia> psi_krylov = arnoldi_exp(A, psi0, 20); abs(psi_ed' * psi_krylov)^2
0.9999999999999991

julia> psi_krylov = arnoldi_exp(A, psi0, 10); abs(psi_ed' * psi_krylov)^2
1.0

julia> psi_krylov = arnoldi_exp(A, psi0, 4); abs(psi_ed' * psi_krylov)^2
1.0000653363523324
```

Krylov space - Arnoldi iteration

- Test with a random Hamiltonian $\hat{H}$ $\quad \hat{A} = -\mathrm{i}\Delta t \hat{H}$



$$\mathcal{F} = |\langle\psi_0| (\mathrm{e}^{\hat{A}})^\dagger \mathbf{Q}_m \mathrm{e}^{\mathbf{h}} \mathbf{Q}_m^\dagger |\psi_0\rangle|^2$$

A larger "time-step" will need a larger Krylov space dimension.

- This makes sense, remember:

$$\mathrm{e}^{-\mathrm{i}\hat{H}t} |\psi_0\rangle = \sum_{k=0} \frac{(-\mathrm{i}t\hat{H})^k}{k!} |\psi_0\rangle \equiv \sum_{k=0} \frac{\hat{A}^k}{k!} |\psi_0\rangle$$

Maybe we only need a few terms in this sum?

The longer I evolve, the "more Hilbert space" I need

Krylov space - Arnoldi iteration

- **Remark:** The implementation on the right is very simple, we did not implement any stopping condition and didn't discuss any (smart) error estimation

There are many implementations out there, e.g. *KrylovKit* for Julia (or *ExpoKit*), etc.

```julia
using KrylovKit

psi, info = exponentiate(A, 1.0, psi0;
                         krylovdim=m, tol=1e-12)
```

```julia
# Compute approximation to exp(A)*psi0
# ... with m Krylov basis states
function arnoldi_exp(A, psi0, m)

    D = length(psi0)

    Q = similar(A, D, m) # the projection matrix
    h = zeros(eltype(A), m, m) # Krylov projection of A

    Q[:,1] = psi0   # assumed normalized
    for ii = 1:(m-1)
        psi_i = A*Q[:,ii]
        for jj = 1:ii
            h[jj,ii] = Q[:,jj]' * psi_i
            psi_i -= h[jj,ii] .* Q[:,jj]
        end
        h[ii+1, ii] = norm(psi_i)
        Q[:,ii+1] = psi_i ./ h[ii+1, ii]
    end

    # now return the matrix exponential
    return Q * exp(h)[:,1]

end
```

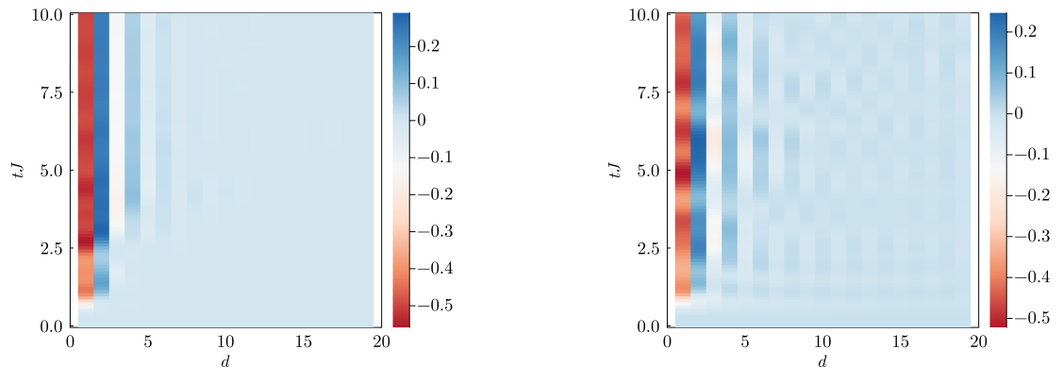
This time

- **Part 1:** Krylov space: Evolution with sparse Hamiltonians

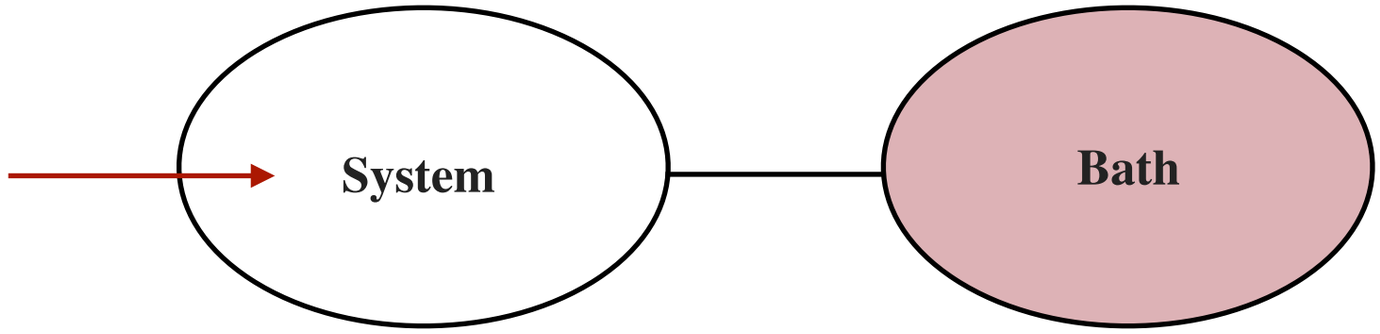
$$\text{span}\left(\hat{A}^0 |\psi_0\rangle, \hat{A}^1 |\psi_0\rangle, \hat{A}^2 |\psi_0\rangle, \ldots, \hat{A}^{m-1} |\psi_0\rangle\right)$$

Tutorials: Power method/implement an Arnoldi iteration

- **Tutorial:** Spin-model simulations using Krylov space



- **Part 2:** Basics for open-system simulations



Tutorial: Spin-model simulations using Krylov space

1. Construct the sparse Hamiltonian

2. Prepare initial state

3. Compute dynamics with Arnoldi exponentiation

4. Plot observable dynamics (also correlation functions)

Long-range transverse Ising model: $\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j} \hat{\sigma}_i^z \hat{\sigma}_j^z + h_x \sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha}$

- **Initial state:** This time different $|\psi_0\rangle = |{\downarrow}{\uparrow}{\downarrow}{\uparrow}\ldots\rangle = |{\downarrow}\rangle \otimes |{\uparrow}\rangle \otimes |{\downarrow}\rangle \otimes |{\uparrow}\rangle \ldots$

  Néel state

- **Look also at correlations:** $C_{i,j} = \langle \hat{\sigma}_i^z \hat{\sigma}_j^z \rangle - \langle \hat{\sigma}_i^z \rangle \langle \hat{\sigma}_j^z \rangle$

Tutorial: Spin-model simulations using Krylov space

Long-range transverse Ising model: $\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z + h_x \sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha}$

```julia
# Step 3: Function doing the time evolution from Néel state
function sparse_ti_simulation(N, J, alpha, hx, dt, steps)

    sm = sparse([0 1; 0 0])
    sxs, szs = build_sparse_spin_ops(sm, N)

    H = build_sparse_hamiltonian(J, alpha, hx, sxs, szs)

    # initial state
    psi = zeros(ComplexF64, 2^N)
    psi[1] = 1.0 # the all zero state

    for ii = 1:2:N
        psi = sxs[ii] * psi
    end

    return psi
```

Néel state

$$|\psi_0\rangle = |\downarrow\uparrow\downarrow\uparrow\downarrow\uparrow\ldots\rangle$$

```
julia> include("ti_model_simulation_sparse.jl");

julia> psi = sparse_ti_simulation(10,1,3,1,0.1,101);

julia> findall( abs.(psi) .> 0)
1-element Vector{Int64}:
 342
```

```
julia> bitstring(342-1)
"0000000000000000000000000000000000000000000000000000000101010101"
```

- So, better idea? *unsigned integer decomposition into n bits* $m = \sum_{i=0}^{n-1} d_i 2^i$

$$m_{\mathrm{Neel}} = \sum_{i=0}^{N/2-1} 2^{2i} \qquad \textit{geometric series!} \qquad r = 4$$

$$S = r^0 + r^1 + r^2 + \cdots + r^M \qquad S = \frac{r^{M+1}-1}{r-1}$$

$$m_{\mathrm{Neel}} = \frac{4^{N/2}-1}{3}$$

```
julia> div(4^cld(10,2) - 1, 3)
341

julia> div(4^cld(9,2) - 1, 3)
341
```

Tutorial: Spin-model simulations using Krylov space

Long-range transverse Ising model: $\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z + h_x \sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha}$

```julia
# Step 3: Function doing the time evolution from Néel state
function sparse_ti_simulation(N, J, alpha, hx, dt, steps)

    sm = sparse([0 1; 0 0])
    sxs, szs = build_sparse_spin_ops(sm, N)

    H = build_sparse_hamiltonian(J, alpha, hx, sxs, szs)

    # initial state
    psi = zeros(ComplexF64, 2^N)
    mneel = div(4^cld(N,2) - 1, 3) + 1
    psi[mneel] = 1.0 # the Néel state
```

Néel state

$|\psi_0\rangle = |\downarrow\uparrow\downarrow\uparrow\downarrow\uparrow\ldots\rangle$

Tutorial: Spin-model simulations using Krylov space

Long-range transverse Ising model: $\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z + h_x \sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha}$

```
    for tt = 1:steps
        println("Step $tt/$steps - norm(psi) = $(norm(psi))")

        for ii = 1:N
            out_sz[tt, ii] = psi' * szs[ii] * psi
        end

        psi, _ = exponentiate(H, -1im * dt, psi;
                              krylovdim = 20, tol = 1e-12)
    end

    return out_sz
end
```

```
Step 96/101 - norm(psi) = 0.9999999999999976
Step 97/101 - norm(psi) = 0.9999999999999974
Step 98/101 - norm(psi) = 0.9999999999999974
Step 99/101 - norm(psi) = 0.9999999999999976
Step 100/101 - norm(psi) = 0.9999999999999976
Step 101/101 - norm(psi) = 0.9999999999999976
  0.088565 seconds (23.70 k allocations: 75.769 MiB, 14.91% gc time)

julia> include("ti_model_simulation_sparse.jl"); @time info = sparse_ti_simulation(10, 1, 3, 1, 0.1, 101);
```

Play with Krylov dim

```
Step 100/101 - norm(psi) = 0.9999999999996546
Step 101/101 - norm(psi) = 0.9999999999996511
  0.131281 seconds (42.79 k allocations: 115.634 MiB, 10.65% gc time)

julia>
```

m=10 (slower)

fail: m=2

```
Step 101/101 - norm(psi) = 1.001604676019396
┌ Warning: expintegrate did not reach sufficiently small error after 100 iterations:
│ * total error = 7.82e-04
│ * number of operations = 300
└ @ KrylovKit ~/.julia/packages/KrylovKit/MlUlm/src/matrixfun/expintegrator.jl:278
  0.648713 seconds (406.09 k allocations: 549.648 MiB, 11.75% gc time)

julia> include("ti_model_simulation_sparse.jl"); @time info = sparse_ti_simulation(10, 1, 3, 1, 0.1, 101);
```

Tutorial:

Long-range transverse Ising model: $\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j} \hat{\sigma}_i^z \hat{\sigma}_j^z + h_x \sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha}$

Switch also dense code to Neel, compare runtimes up to N=11, push up to N=18

```
#println("Step $tt/$steps - norm(psi) = $(norm(psi))")

@time out_sz = sparse_ti_simulation(N, J, alpha, hx, dt, steps)
```

Compare own Arnoldi

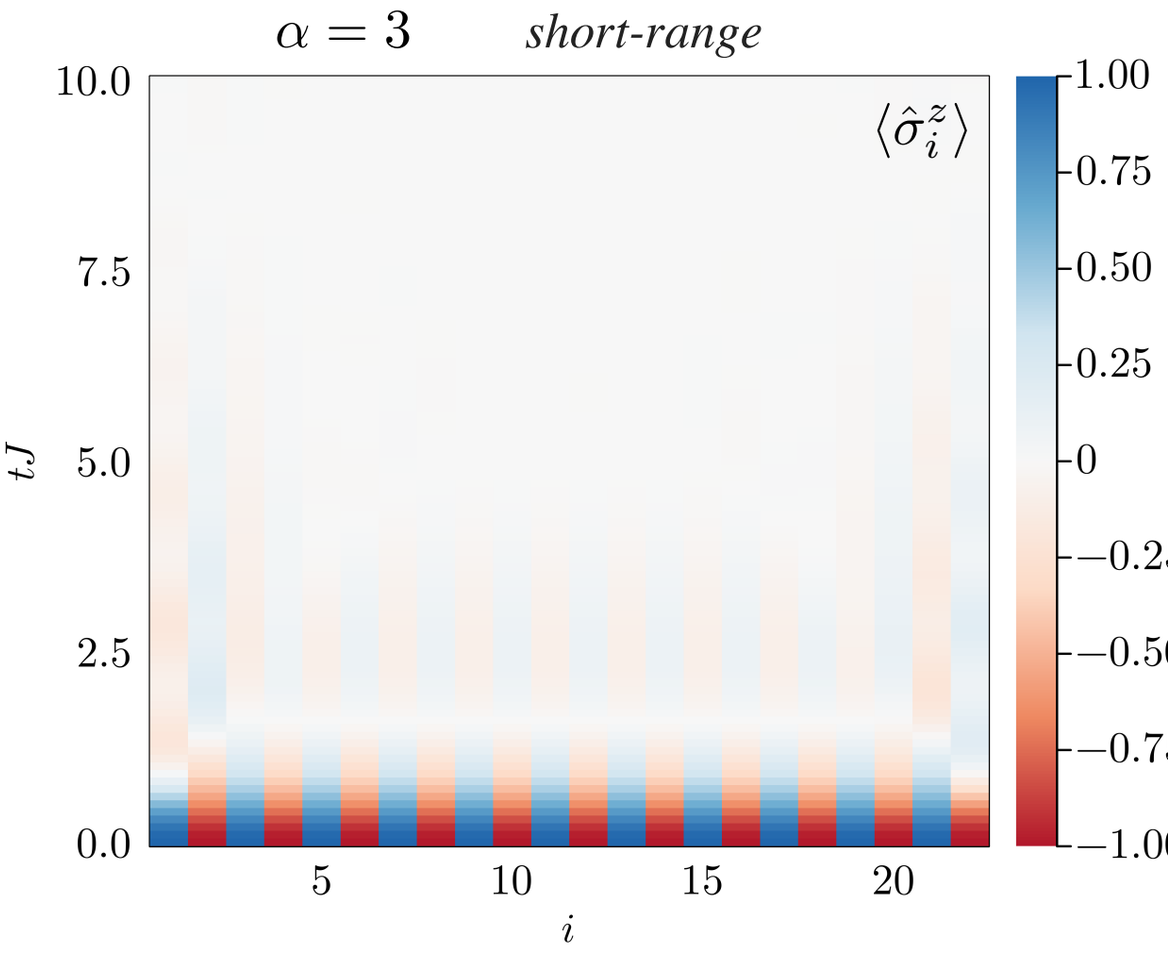
```
A = -1im * dt * H;
# initial state
psi = zeros(ComplexF64, 2^N)
mneel = div(4^cld(N,2) - 1, 3) + 1
psi[mneel] = 1.0 # the Néel state

out_sz = zeros(Float64, steps, N)
out_szsz = zeros(Float64, steps, N-4)

for tt = 1:steps
    #println("Step $tt/$steps - norm(psi) = $(norm(psi))")

    for ii = 1:N
        out_sz[tt, ii] = psi' * szs[ii] * psi
    end

    for cc = 4:N
        out_szsz[tt, cc - 3] = psi' * szs[3] * szs[cc] * psi'
        out_szsz[tt, cc - 3] -= out_sz[3] * out_sz[cc]
    end

    #psi = arnoldi_exp(A, psi, 20)
```

Tutorial: Spin-model simulations using Krylov space

- Evolution of local spin-z component

$$\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z + h_x\sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha} \qquad h_x = J$$

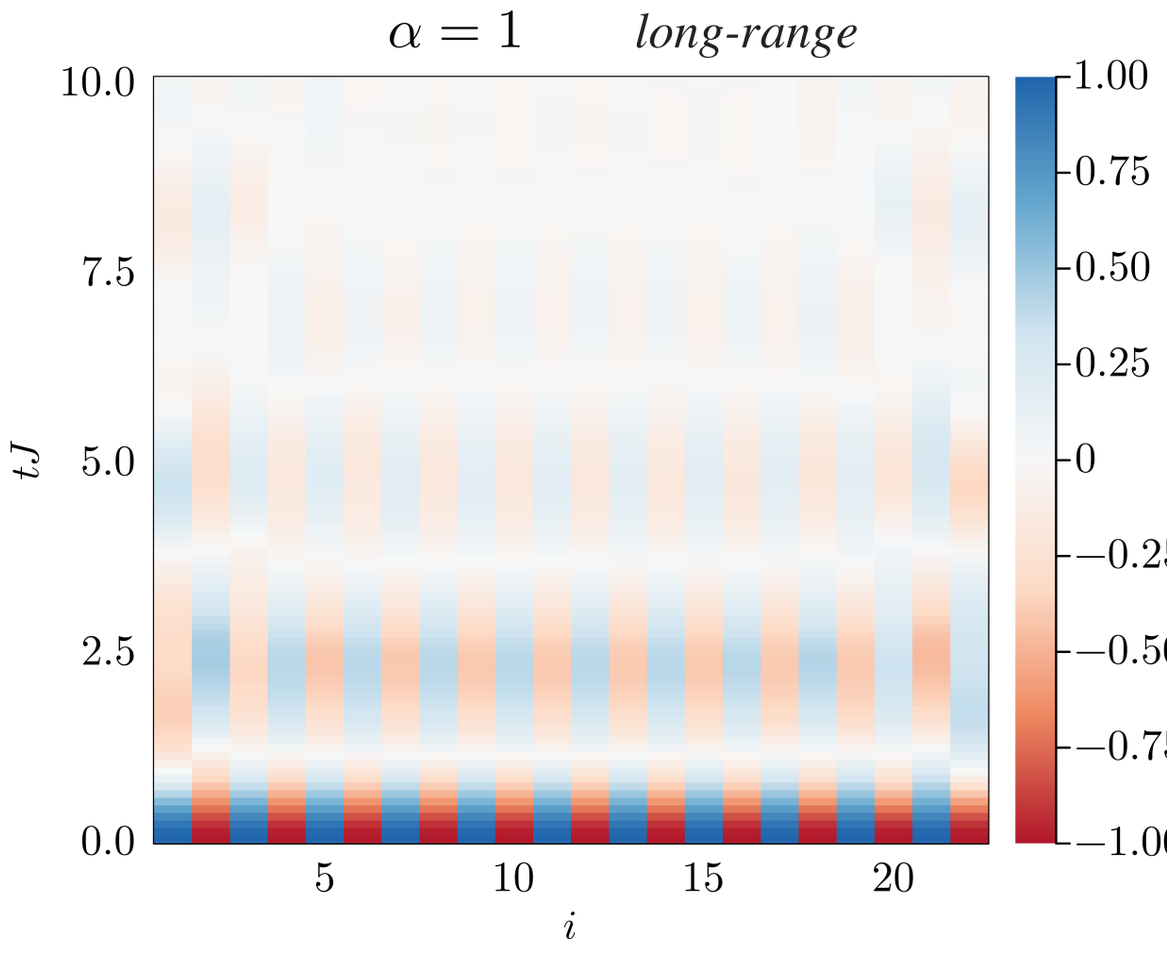
$N = 22$ spins



For the short range case, correlation build-up leads to local mixed states. For example, for sufficiently long times, the reduced density matrices will become fully mixed.

$$\hat{\rho}_i = \mathrm{tr}_{j\neq i}(|\psi(t)\rangle\langle\psi(t)|) \approx \begin{pmatrix} 1/2 & 0 \\ 0 & 1/2 \end{pmatrix} \qquad \mathrm{tr}(\hat{\rho}_i\hat{\sigma}_i^z) \approx 0$$

$$t \gg J^{-1}$$



In the long-range case, dynamics is more collective

$$\alpha = 0$$

$$\hat{H}_{\mathrm{TI}} = J\sum_{i<j} \hat{\sigma}_i^z\hat{\sigma}_j^z + h_x\sum_i \hat{\sigma}_i^x$$

Tutorial: Spin-model simulations using Krylov space

1. Construct the sparse Hamiltonian

2. Prepare initial state

3. Compute dynamics with Arnoldi exponentiation

4. Plot observable dynamics (also correlation functions)

Long-range transverse Ising model: $\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j} \hat{\sigma}_i^z \hat{\sigma}_j^z + h_x \sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha}$

- **Initial state:** This time different $|\psi_0\rangle = |{\downarrow}{\uparrow}{\downarrow}{\uparrow}\ldots\rangle = |{\downarrow}\rangle \otimes |{\uparrow}\rangle \otimes |{\downarrow}\rangle \otimes |{\uparrow}\rangle \ldots$

  Néel state

- **Look also at correlations:** $C_{i,j} = \langle \hat{\sigma}_i^z \hat{\sigma}_j^z \rangle - \langle \hat{\sigma}_i^z \rangle \langle \hat{\sigma}_j^z \rangle$

Tutorial: Spin-model simulations using Krylov space

Long-range transverse Ising model: $\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z + h_x \sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha}$

For correlation: $C_{i,j} = \langle\hat{\sigma}_i^z\hat{\sigma}_j^z\rangle - \langle\hat{\sigma}_i^z\rangle\langle\hat{\sigma}_j^z\rangle$

$C_{3,3+d}$

```
out_szsz = zeros(Float64, steps, N-3)

for tt = 1:steps
    println("Step $tt/$steps - norm(psi) = $(norm(psi))")

    for ii = 1:N
        out_sz[tt, ii] = psi' * szs[ii] * psi
    end

    for cc = 4:N
        out_szsz[tt, cc - 3] = psi' * szs[3] * szs[cc] * psi
        out_szsz[tt, cc - 3] -= out_sz[tt, 3] * out_sz[tt, cc]
    end
```

```
return out_sz, out_szsz
```

```
@time out_sz, out_szsz = sparse_ti_simulation(N, J, alpha, hx, dt, steps)
```

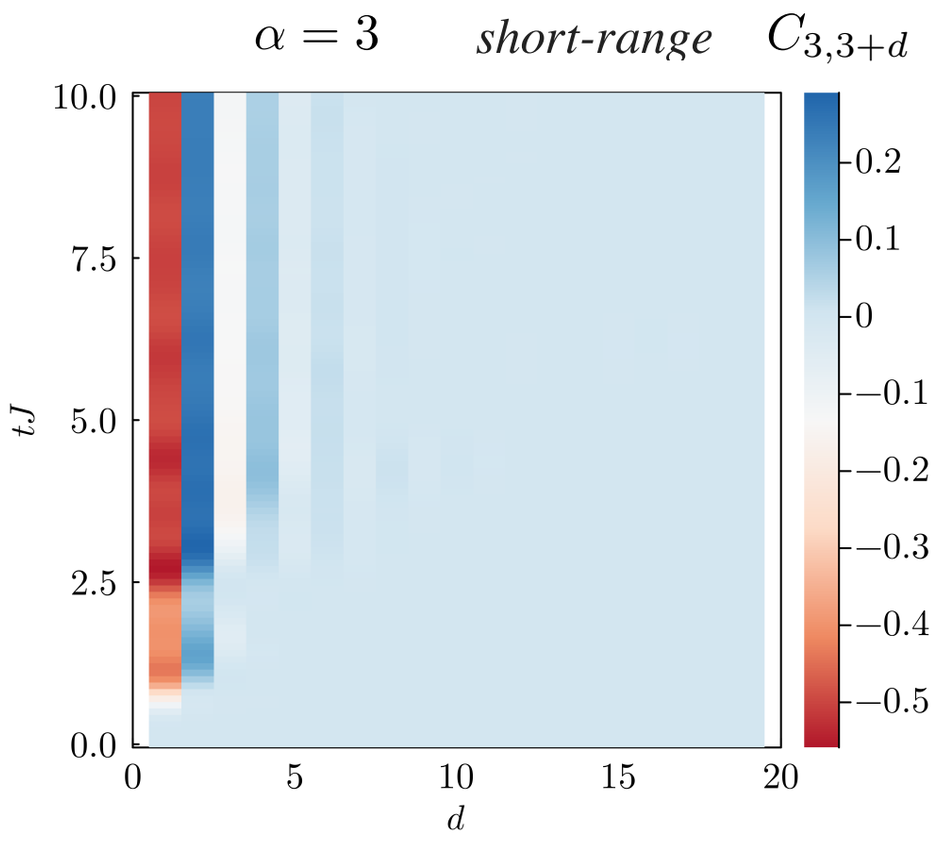
Tutorial: Spin-model simulations using Krylov space

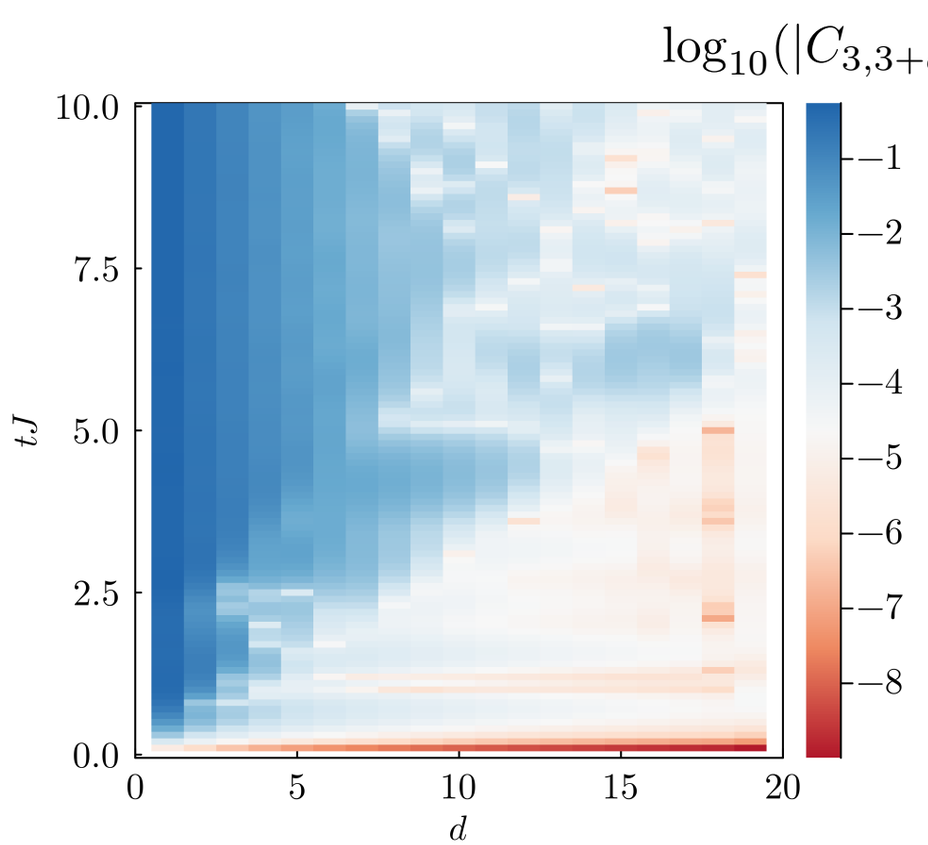
- Evolution of local spin-z component

$N = 22$ spins

$$\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z + h_x \sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha}$$

$$C_{i,j} = \langle\hat{\sigma}_i^z\hat{\sigma}_j^z\rangle - \langle\hat{\sigma}_i^z\rangle\langle\hat{\sigma}_j^z\rangle$$





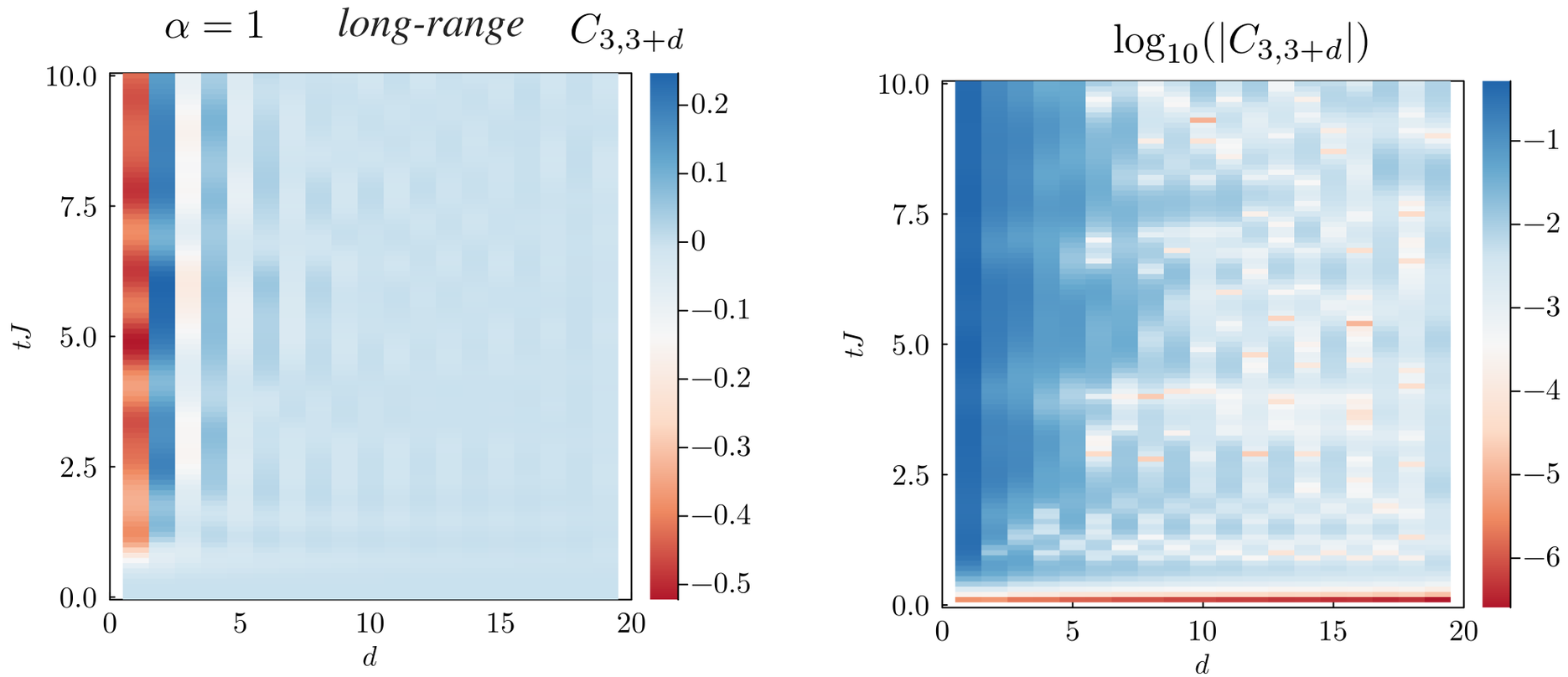
There is a spreading out in a "light-cone".
The two-spin correlations become very small over long-distances.

Tutorial: Spin-model simulations using Krylov space

- Evolution of two-spin correlations
- $N = 22$ spins

$$\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z + h_x \sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha}$$

$$C_{i,j} = \langle\hat{\sigma}_i^z\hat{\sigma}_j^z\rangle - \langle\hat{\sigma}_i^z\rangle\langle\hat{\sigma}_j^z\rangle$$



For long-range couplings: complicated interplay light-cone vs. all-to all

- Complex information spreading physics as competition between long and short-range terms … further reading:

P. Calabrese and J. Cardy, J. Stat. Mech, P04010 (2005)
J. Eisert, M. van den Worm, S. R. Manmana, M. Kastner, Phys. Rev. Lett. 111, 260401 (2013)
J. Schachenmayer, B. P. Lanyon, C. F. Roos, A. J. Daley, Phys. Rev. X 3, 031015 (2013)
Z.-X. Gong, M. Foss-Feig, Fernando G. S. L. Brandão, and A. V. Gorshkov, Phys. Rev. Lett. 119 050501 (2017)
L. Cevolani, J. Despres, G. Carleo, L. Tagliacozzo, L. Sanchez-Palencia, Phys. Rev. B 98, 024302 (2018)
I. Frérot, P. Naldesi, and T. Roscilde, Phys. Rev. Lett. 120, 050401 (2018), ….

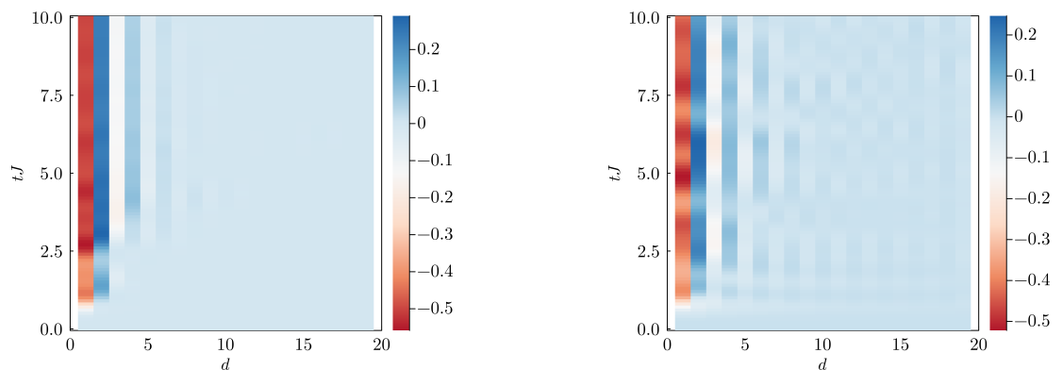
This time

- **Part 1:** Krylov space: Evolution with sparse Hamiltonians

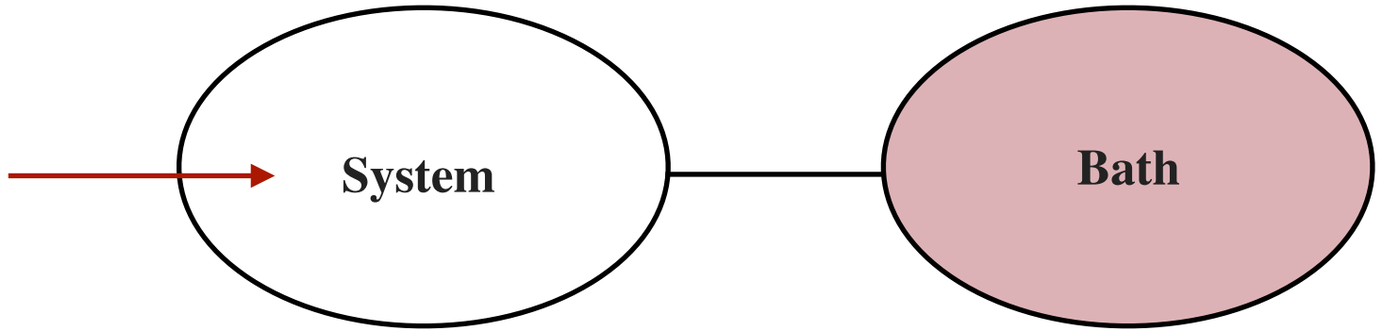
$$\text{span}\left(\hat{A}^0\left|\psi_0\right\rangle, \hat{A}^1\left|\psi_0\right\rangle, \hat{A}^2\left|\psi_0\right\rangle, \ldots, \hat{A}^{m-1}\left|\psi_0\right\rangle\right)$$

Tutorials: Power method/ **i**mplement an Arnoldi iteration

- **Tutorial:** Spin-model simulations using Krylov space

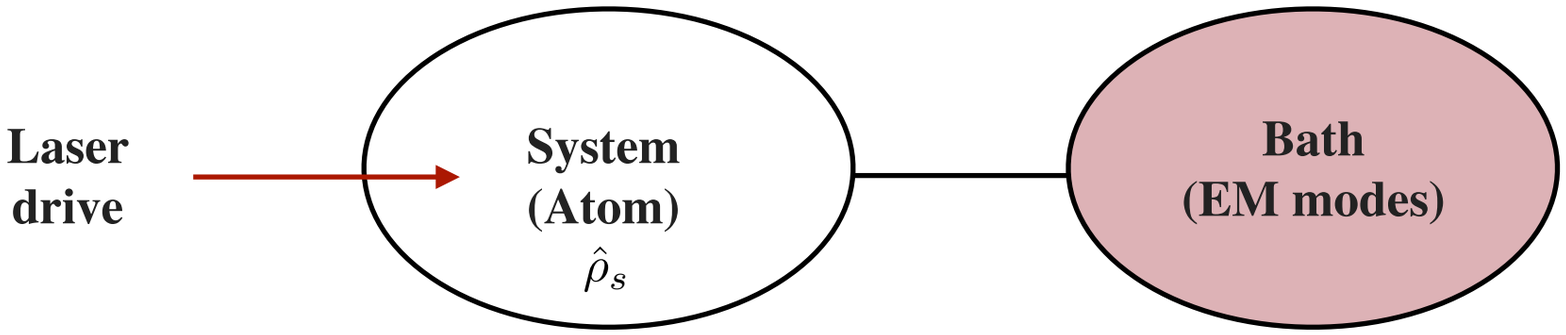


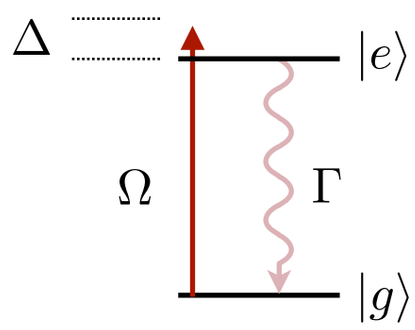
- **Part 2:** Basics for open-system simulations



Lindblad master equations - Review

- Here we will focus on Lindblad master equations: We have a **System** and a **large bath:**





Example: Consider an atom coupled to the electromagnetic vacuum modes

- Effectively a Lindblad master equation describes the dynamics of the system only (the bath is traced out), the system needs to be described by a density matrix $\hat{\rho}_s$

- Several approximations on the way:

  For the atomic decay problem: Dipole approximation, Rotating wave approximation, **Markov approximation**, …

$$\frac{d}{dt}\hat{\rho}_S = \mathcal{L}(\hat{\rho}_S) = -\mathrm{i}\left[\hat{H}_S, \hat{\rho}\right] + \sum_\eta \left(2\hat{L}_\eta \hat{\rho}_S \hat{L}_\eta^\dagger - \hat{L}_\eta^\dagger \hat{L}_\eta \hat{\rho}_S - \hat{\rho}_S \hat{L}_\eta^\dagger \hat{L}_\eta\right)$$

Lindblad jump operators

- E.g. for a single decaying atom "Optical Bloch equations" $\hat{L} = \sqrt{\frac{\Gamma}{2}}\hat{\sigma}^-$

$$\frac{d}{dt}\hat{\rho}_S = -\mathrm{i}\left[\hat{H}_{0A}, \hat{\rho}_S\right] + \frac{\Gamma}{2}\left(-\hat{\sigma}^+\hat{\sigma}^-\hat{\rho}_S - \hat{\rho}_S\hat{\sigma}^+\hat{\sigma}^- + 2\hat{\sigma}^-\hat{\rho}_S\hat{\sigma}^+\right) \quad \ldots \text{ decay of atom with rate } \Gamma$$

Free atomic Hamiltonian (e.g. with drive)

Lindblad master equations - Review

- General form of Lindblad master equation

$$\frac{d}{dt}\hat{\rho}_S = \mathcal{L}(\hat{\rho}_S) = -\mathrm{i}\left[\hat{H}_S, \hat{\rho}\right] + \sum_\eta \left(2\hat{L}_\eta \hat{\rho}_S \hat{L}_\eta^\dagger - \hat{L}_\eta^\dagger \hat{L}_\eta \hat{\rho}_S - \hat{\rho}_S \hat{L}_\eta^\dagger \hat{L}_\eta\right)$$

- Alternative equivalent forms

$$\hat{H}_{\text{eff}} = \hat{H}_S - \mathrm{i}\sum_\eta \hat{L}_\eta^\dagger \hat{L}_\eta$$

*effective **non-hermitian** Hamiltonian*

non-trace preserving

$$\frac{d}{dt}\hat{\rho}_S = -\mathrm{i}\left(\hat{H}_{\text{eff}}\hat{\rho}_S - \hat{\rho}_S \hat{H}_{\text{eff}}^\dagger\right) + \sum_\eta 2\hat{L}_\eta \hat{\rho}_S \hat{L}_\eta^\dagger$$

recycling term (fixes trace)

- In fact, the Lindblad form is the simplest way to write a linear evolution equation which is trace preserving

$$\mathrm{tr}\left[\frac{d}{dt}\hat{\rho}_S\right] = \mathrm{tr}\left\{\mathcal{L}(\hat{\rho}_S)\right\} = \sum_\eta \mathrm{tr}\left(2\hat{L}_\eta \hat{\rho}_S \hat{L}_\eta^\dagger\right) - \mathrm{tr}\left(\hat{L}_\eta^\dagger \hat{L}_\eta \hat{\rho}_S\right) - \mathrm{tr}\left(\hat{\rho}_S \hat{L}_\eta^\dagger \hat{L}_\eta\right) = 0$$

cyclic invariance of trace

Lindblad master equations - Review

- The right hand side of the master equation is just a function of the density matrix ("super-operator")

$$\frac{d}{dt}\hat{\rho}_S = \mathcal{L}(\hat{\rho}_S) = -\mathrm{i}\left[\hat{H}_S, \hat{\rho}\right] + \sum_\eta \left(2\hat{L}_\eta \hat{\rho}_S \hat{L}_\eta^\dagger - \hat{L}_\eta^\dagger \hat{L}_\eta \hat{\rho}_S - \hat{\rho}_S \hat{L}_\eta^\dagger \hat{L}_\eta\right)$$

- We can simply define a statevector of matrix elements, and implement the super-operator as function

$$\mathbf{y} = [\rho_{1,1}, \rho_{2,1}, \rho_{3,1}, \ldots, \rho_{1,2}, \rho_{1,2}, \ldots]^T$$

$$\boxed{\frac{d}{dt}\hat{\rho}_S = \mathcal{L}(\hat{\rho}_S) \quad \Leftrightarrow \quad \dot{\mathbf{y}}(t) = f(\mathbf{y}(t))}$$

…we could plug this into ODE solver!

- However, the **master equation** is also **linear!**

We can write it like $\dot{\mathbf{y}}(t) = f(\mathbf{y}(t)) = \mathbf{A} \cdot \mathbf{y}(t)$ *(next slide)*

Solution: $y(t) = \mathrm{e}^{t\mathbf{A}} y(0)$

If we construct the (sparse) matrix representing the whole "Liouvillian" matrix $\boldsymbol{A}$*, we can also use our Krylov space method again!*

Numerics: Density matrix as state-vector

- How to construct the super-operator matrix? One possibility …

$$\hat{\rho} = \underbrace{\begin{pmatrix} \rho_{1,1} & \rho_{1,2} & \rho_{1,3} \\ \rho_{2,1} & \rho_{2,2} & \rho_{2,3} \\ \rho_{3,1} & \rho_{3,2} & \rho_{3,3} \end{pmatrix}}_{D \times D} \rightarrow \mathbf{y} = \big(\underbrace{\rho_{1,1}, \rho_{2,1}, \rho_{3,1}}_{\mathbf{y}_1^T}, \underbrace{\rho_{1,2}, \rho_{2,2}, \rho_{3,2}}_{\mathbf{y}_2^T}, \underbrace{\rho_{1,3}, \rho_{2,3}, \rho_{3,3}}_{\mathbf{y}_3^T}\big)^T$$

$$\underbrace{\big(\mathbf{y}_1 \quad \mathbf{y}_2 \quad \mathbf{y}_3\big)}$$

$$\rho_{i,j} \quad \Leftrightarrow \quad y_v \qquad v = (j-1)D + i$$

- Now consider an operator $\hat{A}$

Kronecker product definition: $\hat{A} \otimes \hat{B} = \begin{pmatrix} A_{1,1}\hat{B} & A_{1,2}\hat{B} & \dots \\ A_{2,1}\hat{B} & A_{2,2}\hat{B} & \dots \\ \vdots & \vdots & \end{pmatrix}$

$$(\mathbb{1} \otimes \hat{A})\mathbf{y} = \underbrace{\begin{pmatrix} \hat{A} & 0 & 0 \\ 0 & \hat{A} & 0 \\ 0 & 0 & \hat{A} \end{pmatrix}}_{D^2 \times D^2} \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \end{pmatrix} = \hat{A}\begin{pmatrix} \mathbf{y}_1 & \mathbf{y}_2 & \mathbf{y}_3 \end{pmatrix} = \hat{A}\hat{\rho}$$

Multiplication from left side

- Equally, it's easy to show *(exercise)*

$$(\hat{B}^T \otimes \mathbb{1})\mathbf{y} = \hat{\rho}\hat{B}$$

Multiplication from right side

$$(\hat{B}^T \otimes \hat{A})\mathbf{y} = \hat{A}\hat{\rho}\hat{B}$$

Multiplication from left and right side

Liouvillian matrix can be fully constructed with Kronecker products!

Numerics: Density matrix as state-vector

$$\hat{\rho} = \begin{pmatrix} \rho_{1,1} & \rho_{1,2} & \rho_{1,3} \\ \rho_{2,1} & \rho_{2,2} & \rho_{2,3} \\ \rho_{3,1} & \rho_{3,2} & \rho_{3,3} \end{pmatrix} \rightarrow \mathbf{y} = \left(\rho_{1,1}, \rho_{2,1}, \rho_{3,1}, \rho_{1,2}, \rho_{2,2}, \rho_{3,2}, \rho_{1,3}, \rho_{2,3}, \rho_{3,3}\right)^T$$

"Column-major order in Julia"

```
rho = reshape(y, 2^N, 2^N)
```

```
y = rho[:] # memory ordering fits
```

$$(\mathbb{1} \otimes \hat{A})\mathbf{y} = \hat{A}\hat{\rho} \qquad (\hat{B}^T \otimes \mathbb{1})\mathbf{y} = \hat{\rho}\hat{B} \qquad (\hat{B}^T \otimes \hat{A})\mathbf{y} = \hat{A}\hat{\rho}\hat{B}$$

- Then:

$$\hat{H}_{\text{eff}} = \hat{H}_S - \mathrm{i}\sum_\eta \hat{L}_\eta^\dagger \hat{L}_\eta \qquad \frac{d}{dt}\hat{\rho}_S = -\mathrm{i}\left(\hat{H}_{\text{eff}}\hat{\rho}_S - \hat{\rho}_S \hat{H}_{\text{eff}}^\dagger\right) + \sum_\eta 2\hat{L}_\eta \hat{\rho}_S \hat{L}_\eta^\dagger$$

$$-\mathrm{i}\left[(\mathbb{1} \otimes \hat{H}_{\text{eff}}) - (\hat{H}_{\text{eff}}^* \otimes \mathbb{1})\right]\mathbf{y}$$

element-wise conjugate

$$2(\hat{L}_\eta^* \otimes \hat{L}_\eta)\mathbf{y}$$

```
Heff = H
for nn = 1:N
    Heff -= 1im .* (gam/2) .* (sms[nn]' * sms[nn])
end
```

```
idD = sparse(I, 2^N, 2^N)
```

Example:

$$\hat{L}_\eta = \sqrt{\frac{\Gamma}{2}}\hat{\sigma}_\eta^-$$

```
A = -1im .* (kron(idD, Heff) - kron(conj.(Heff), idD))
```

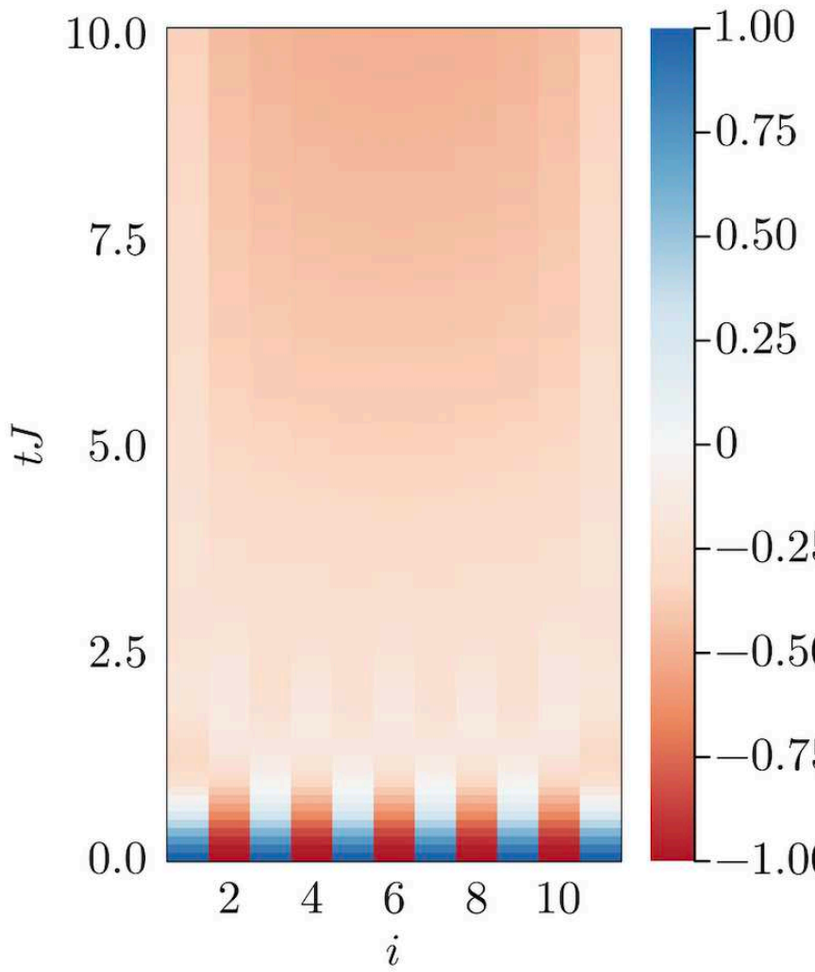
```
for nn = 1:N
    A += gam .* kron(sms[nn], sms[nn]) # sigmas are real
end
```

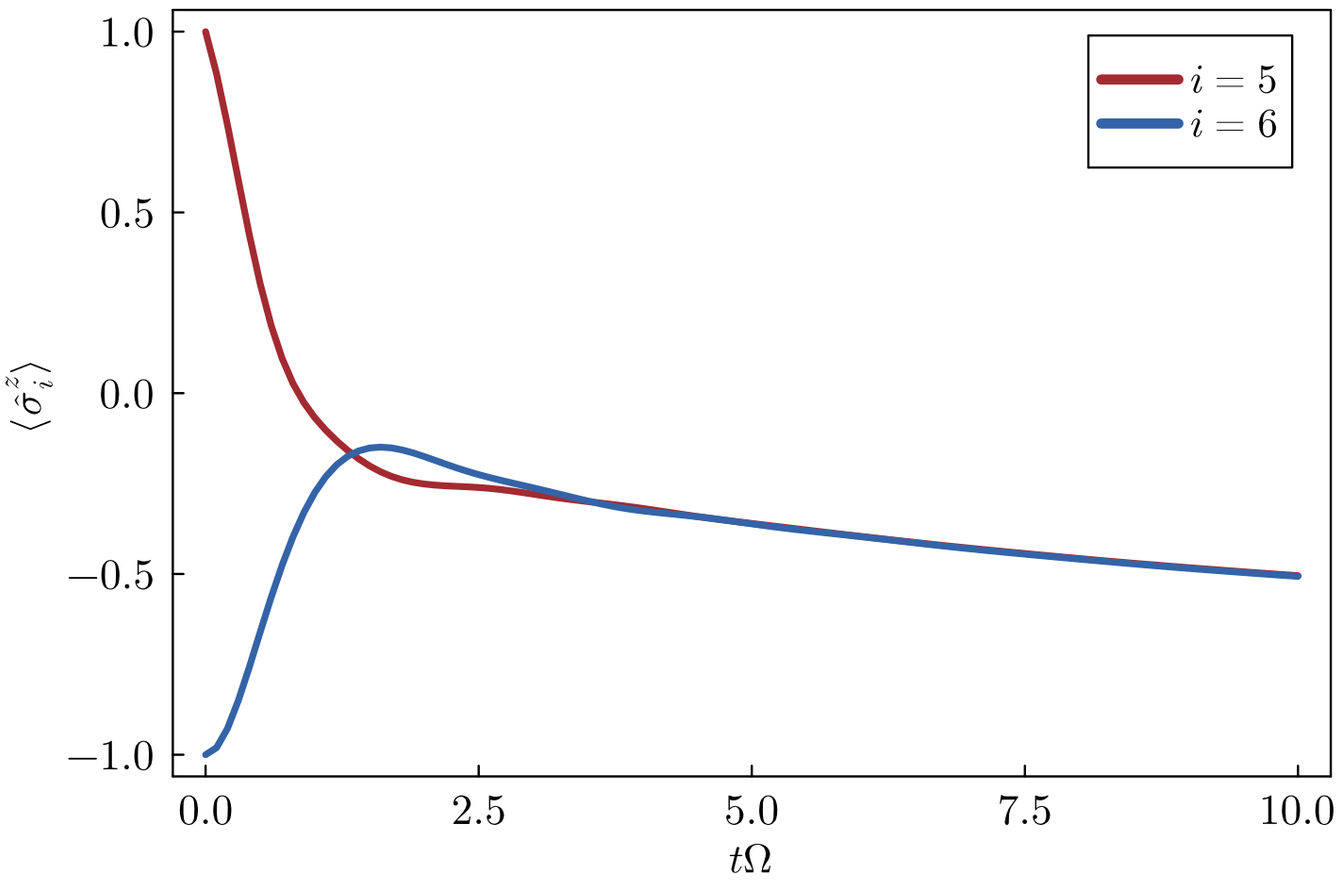
Numerics: Density matrix as state-vector

- Evolution of local spin-z component
- *N = 11* spins

$$\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z + h_x\sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha} \qquad h_x = J$$

$$\alpha = 3 \qquad \Gamma = \frac{J}{2}$$





- **Remark:** The Liouvillian matrix construction is not ideal, an alternative, expand in Pauli basis

$$\hat{\rho} = \sum_{\{i_n\}=1}^{4} r_{i_1,i_2,\ldots,i_N}\,\hat{\sigma}_1^{r_1}\otimes\hat{\sigma}_1^{r_2}\otimes\ldots\,\hat{\sigma}_N^{r_N}$$

$$\hat{\sigma}_i^{0,1,2,3} = \{\mathbb{1}_i, \hat{\sigma}_i^x, \hat{\sigma}_i^y, \hat{\sigma}_i^z\}$$

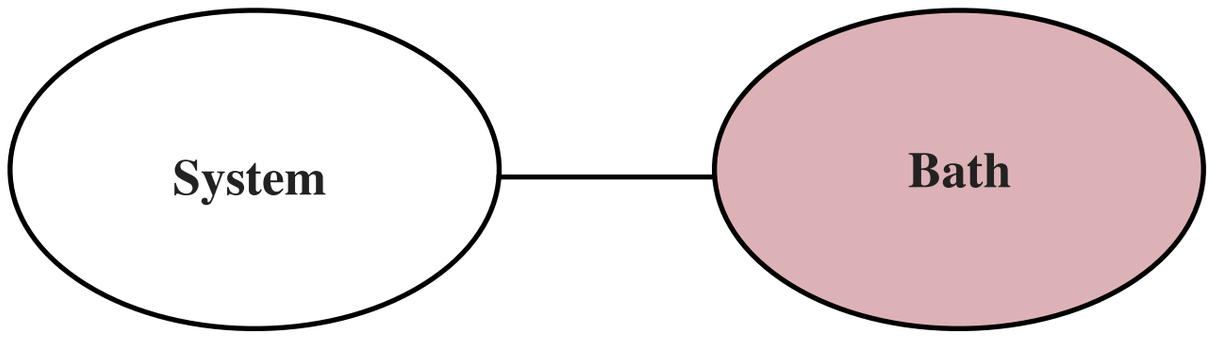
$$\mathrm{tr}\left(\hat{\sigma}_i^n\hat{\sigma}_i^m\right) = \delta_{n,m}$$

Advantage: The "Bloch" state vectors are real!

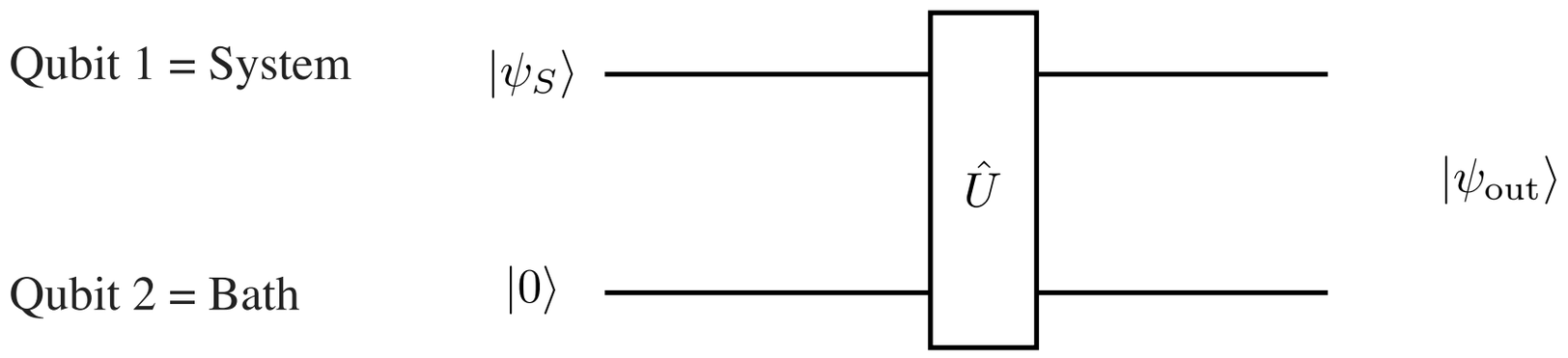
- **Remark:** Since we keep the exact full density matrix, we're now limited to ~half the size than before :(

More general framework: Kraus operators and quantum trajectories

- A more general framework for describing sub-system dynamics:



- Most minimalistic form:



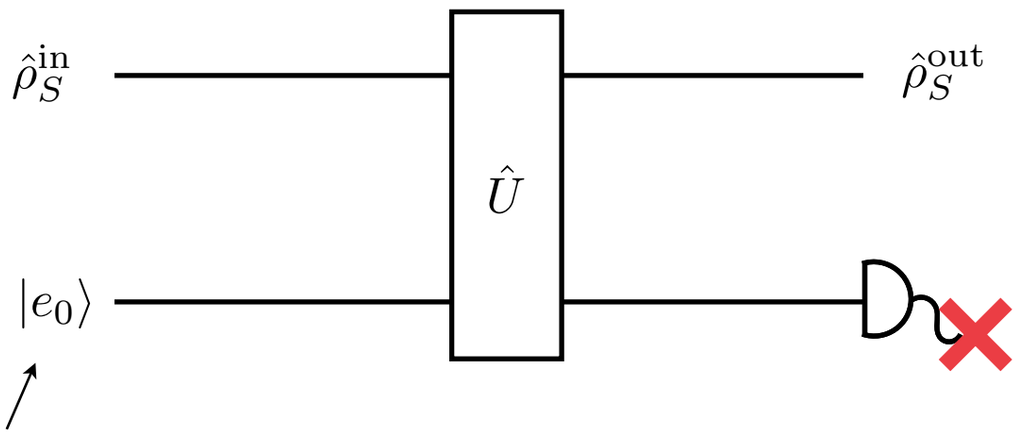
- We consider an initial product state $|\psi_{\text{in}}\rangle = |\psi_S\rangle \otimes |0\rangle$ (The two qubits never interacted with each other)

- Combined system still a pure state $|\psi_{\text{out}}\rangle = \hat{U}\,|\psi_{\text{in}}\rangle$ "reversible process, no information lost"

More general framework: Kraus operators and quantum trajectories

- General **quantum operation** $\hat{\rho}_S^{\text{out}} = \mathcal{E}(\hat{\rho}_S^{\text{in}})$ Mapping of any density matrix to another one

- One can think of it as …



Some state of a bath

$$\boxed{\mathcal{E}(\hat{\rho}_S^{\text{in}}) = \sum_k \hat{E}_k \hat{\rho}_S^{\text{in}} \hat{E}_k^\dagger}$$

Operator sum representation

$$= \sum_k \langle e_k | \hat{U} (\hat{\rho}_S^{\text{in}} \otimes |e_0\rangle \langle e_0|) \hat{U}^\dagger | e_k \rangle$$

Kraus operators

$$\hat{E}_k \equiv \langle e_k | \hat{U} | e_0 \rangle$$

(operators acting on system)
in general: non-unitary, non-hermitian

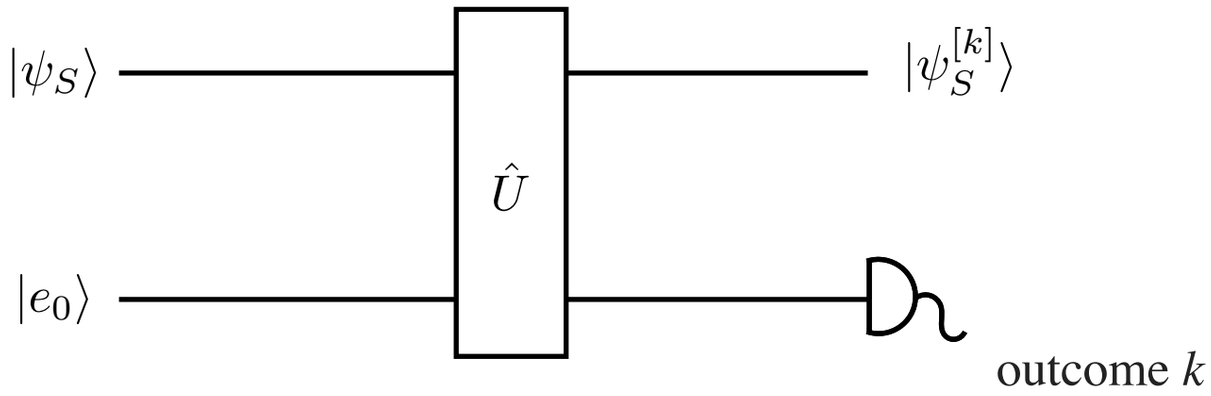
- Only condition on Kraus operators (quantum operations preserve the trace)

$$\text{tr}\mathcal{E}(\hat{\rho}_S^{\text{in}}) = 1 = \sum_k \text{tr}\left(\hat{E}_k \hat{\rho}_S^{\text{in}} \hat{E}_k^\dagger\right) = \sum_k \text{tr}\left(\hat{E}_k^\dagger \hat{E}_k \hat{\rho}_S^{\text{in}}\right) \quad \Leftrightarrow \quad \forall \hat{\rho}_S \quad \boxed{\sum_k \hat{E}_k \hat{E}_k^\dagger = \mathbb{1}}$$

see e.g. Nielsen & Chuang

More general framework: Kraus operators and quantum trajectories

- **Quantum trajectories = Natural stochastic interpretation!**



$$\hat{\rho}_S^{\text{out}} = \mathcal{E}(\hat{\rho}_S^{\text{in}})$$

$$\mathcal{E}(\hat{\rho}_S^{\text{in}}) = \sum_k \hat{E}_k \hat{\rho}_S^{\text{in}} \hat{E}_k^\dagger$$

- **Quantum trajectories:** Stochastically evolve a sample state from the density matrix, according to the rule:

- Randomly pick:

$$|\psi_S^{[k]}\rangle = \begin{cases} \hat{E}_1 |\psi_S\rangle / \sqrt{p_1} & \text{with probability} \quad p_1 = \langle\psi_S|\hat{E}_1^\dagger \hat{E}_1|\psi_S\rangle \\ \quad \dots & \\ \hat{E}_m |\psi_S\rangle / \sqrt{p_m} & \text{with probability} \quad p_m = \langle\psi_S|\hat{E}_m^\dagger \hat{E}_m|\psi_S\rangle \end{cases}$$

$$\hat{\rho}_S^{\text{out}} \approx \sum_k p_k \frac{1}{p_k} \hat{E}_k |\psi_S\rangle \langle\psi_S| \hat{E}_k^\dagger = \mathcal{E}(\hat{\rho}_S^{\text{in}})$$

More general framework: Kraus operators and quantum trajectories

- **Example:** from Lindblad master equation above (with single jump operator)

$$\hat{H}_{\text{eff}} = \hat{H}_S - \mathrm{i}\hat{L}^\dagger\hat{L} \qquad\qquad \frac{d}{dt}\hat{\rho}_S = -\mathrm{i}\left(\hat{H}_{\text{eff}}\hat{\rho}_S - \hat{\rho}_S\hat{H}_{\text{eff}}^\dagger\right) + 2\hat{L}\hat{\rho}_S\hat{L}^\dagger$$

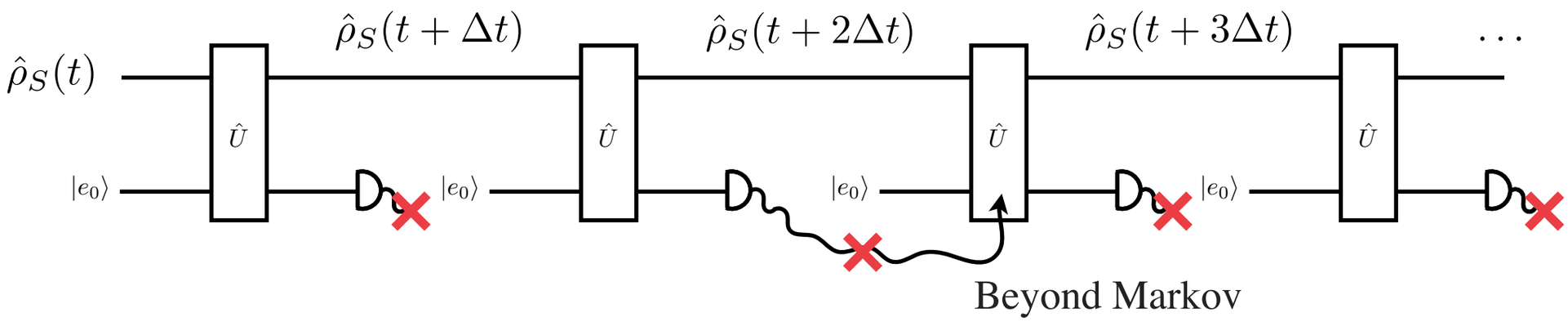
- It's a Markovian master equation … so how do Kraus operators look like?

- Integrate over small time-step: $\Delta t \to 0$

$$\begin{aligned}\hat{\rho}_S(t+\Delta t) &= \hat{\rho}_S(t) - \mathrm{i}\Delta t\left(\hat{H}_{\text{eff}}\hat{\rho}_S(t) - \hat{\rho}_S(t)\hat{H}_{\text{eff}}^\dagger\right) + 2\Delta t\hat{L}\hat{\rho}_S(t)\hat{L}^\dagger \\ &= \underbrace{(\mathbb{1} - \mathrm{i}\hat{H}_{\text{eff}}\Delta t)}_{\hat{E}_0}\,\hat{\rho}_S(t)\,(\mathbb{1} + \mathrm{i}\hat{H}_{\text{eff}}^\dagger\Delta t) + \underbrace{\sqrt{2\Delta t}\hat{L}}_{\hat{E}_1}\,\hat{\rho}_S(t)\,\sqrt{2\Delta t}\hat{L}^\dagger + \mathcal{O}(\Delta t^2)\end{aligned}$$

$$\mathcal{E}[\hat{\rho}_S(t+\Delta t)] = \sum_k \hat{E}_k\,\hat{\rho}_S(t)\,\hat{E}_k^\dagger$$

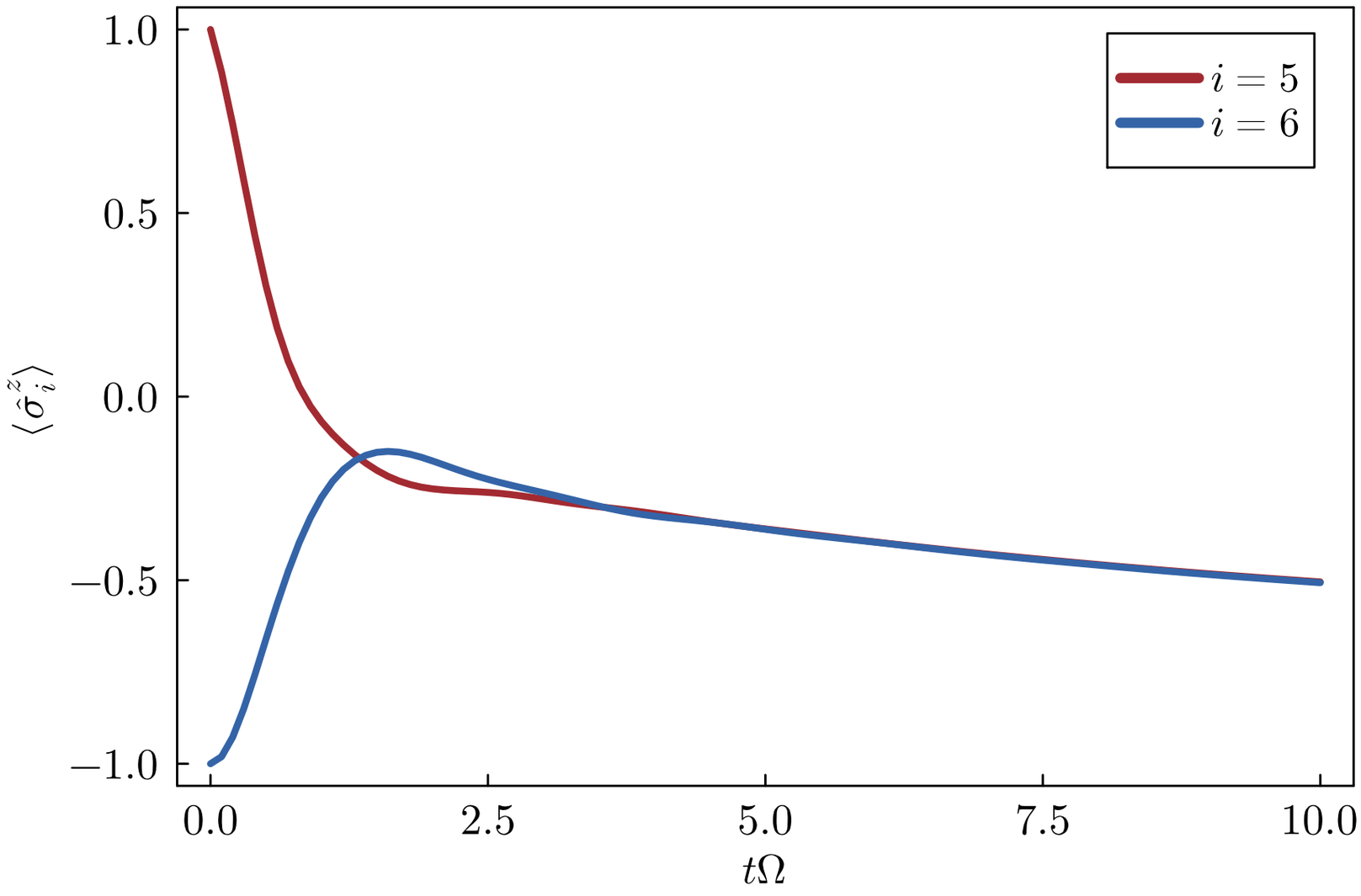
- **Note:** Markovian approximation in Master equation is now quite intuitive



Quantum trajectories

- Evolution of local spin-z component
- *N = 11* spins $\alpha = 3 \qquad \Gamma = \frac{J}{2}$
- Reminder - **full density matrix:**

$$\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z + h_x\sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha} \qquad h_x = J$$



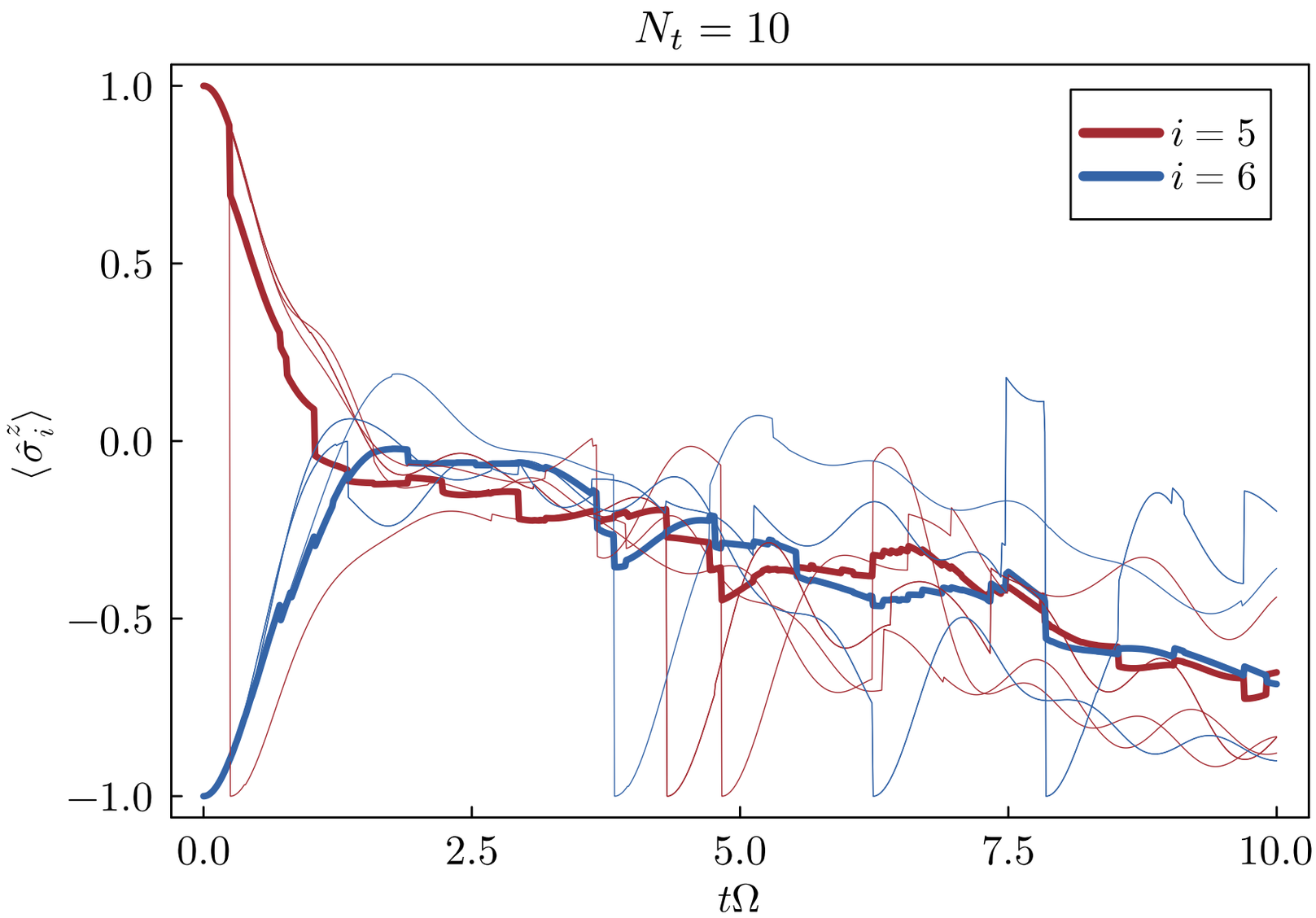
Quantum trajectories

- Evolution of local spin-z component
- *N = 11* spins

$$\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z + h_x\sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha} \qquad h_x = J$$

$$\alpha = 3 \qquad \Gamma = \frac{J}{2}$$

- **Trajectories:**



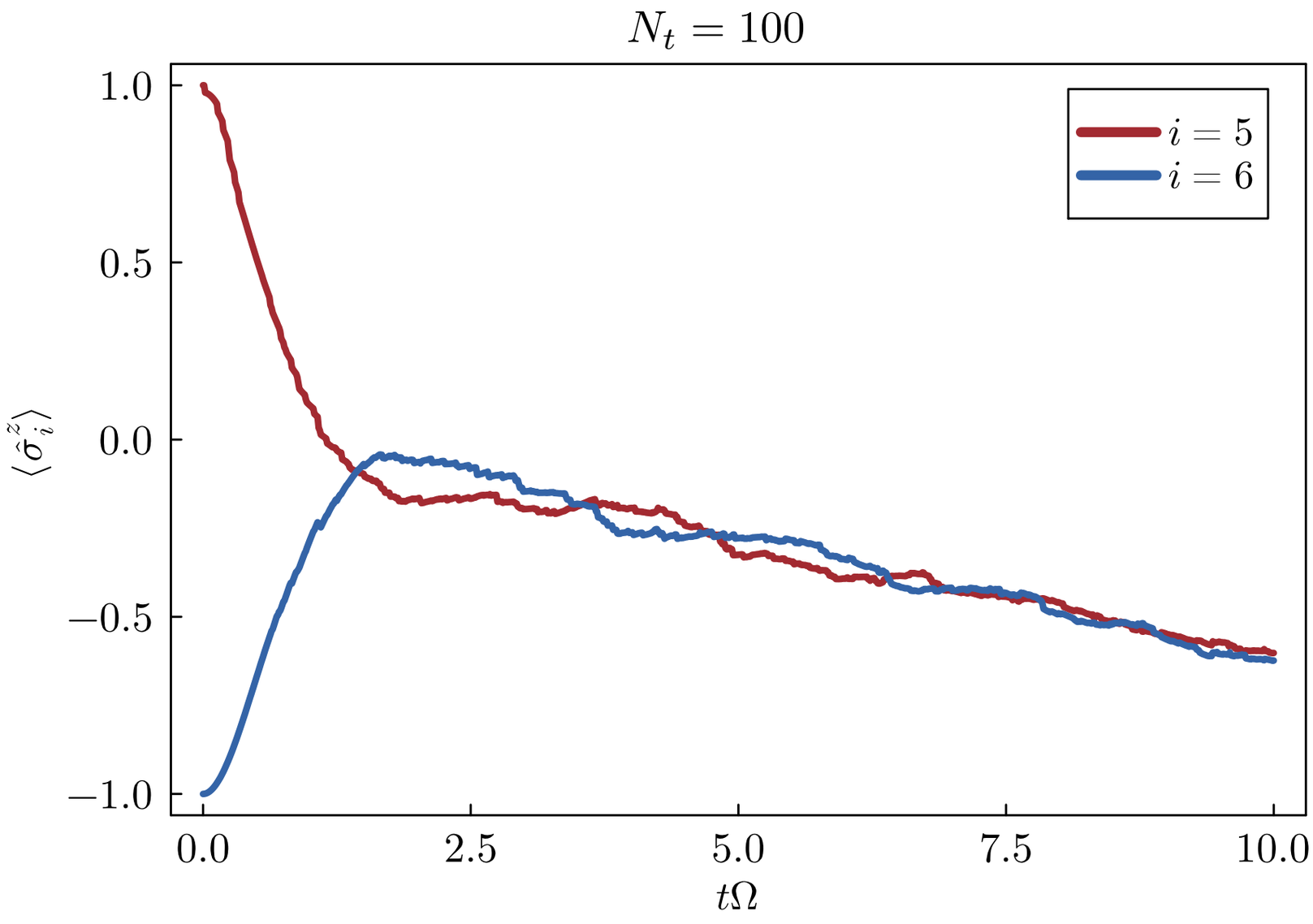
Quantum trajectories

- Evolution of local spin-z component
- *N = 11* spins

$$\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z + h_x\sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha} \qquad h_x = J$$

$$\alpha = 3 \qquad \Gamma = \frac{J}{2}$$

- **Trajectories:**

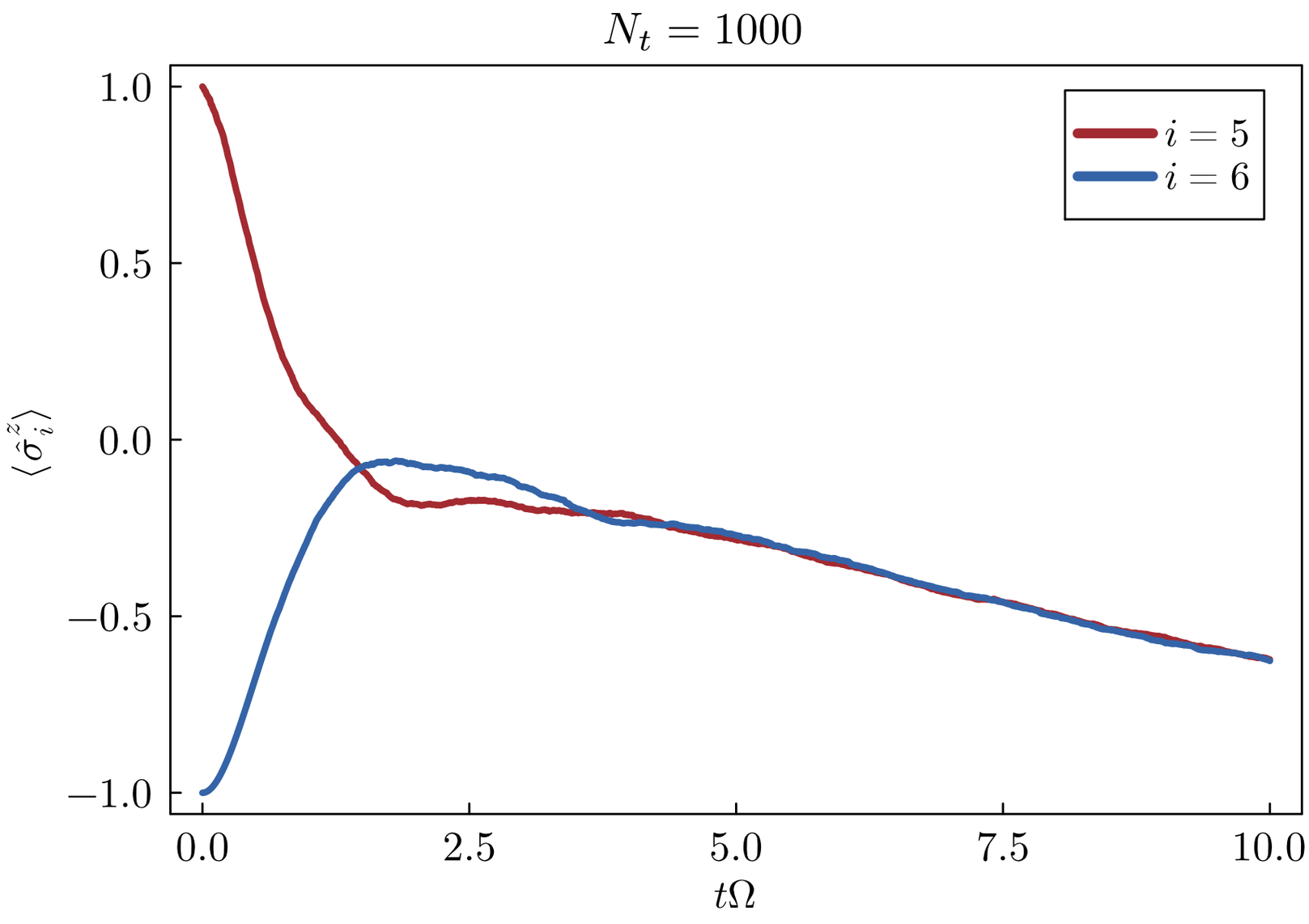


Quantum trajectories

- Evolution of local spin-z component
- *N = 11* spins $\alpha = 3 \qquad \Gamma = \frac{J}{2}$

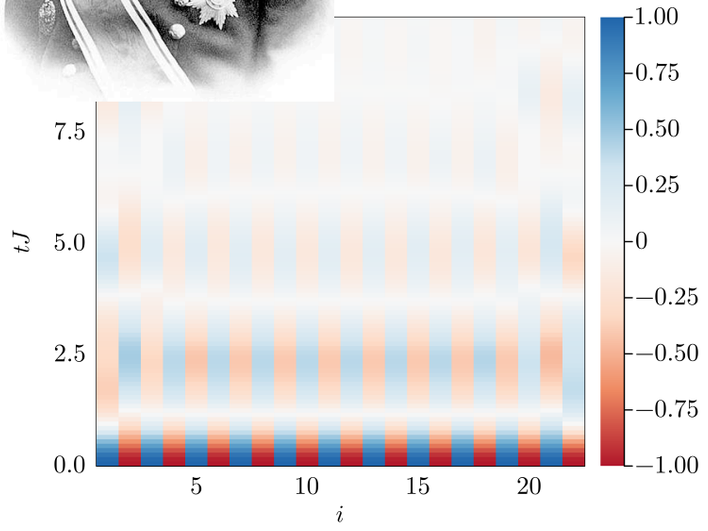
$$\hat{H}_{\mathrm{TI}} = \sum_{i<j} J_{i,j}\hat{\sigma}_i^z\hat{\sigma}_j^z + h_x \sum_i \hat{\sigma}_i^x \qquad J_{ij} = \frac{J}{|i-j|^\alpha} \qquad h_x = J$$

- **Trajectories:**



- **Remark:** This only works for simple observables. For correlations that are small, **many trajectories** needed. Statistical error bars only decrease as $1/\sqrt{N_t}$
- **On the other hand:** Many trajectories are easy to compute on clusters, since the method parallelizes trivially!

Recap



- We introduced a time-evolution algorithm for linear systems, based on: **Krylov space**. Krylov space is a vector-space constructed from an initial state and the evolution matrix:

$$\text{span}\left(\hat{A}^0\left|\psi_0\right\rangle, \hat{A}^1\left|\psi_0\right\rangle, \hat{A}^2\left|\psi_0\right\rangle, \ldots, \hat{A}^{m-1}\left|\psi_0\right\rangle\right)$$

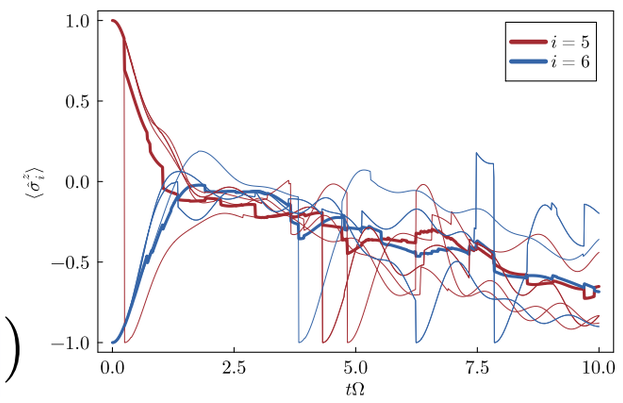
Eigenvectors need to be made orthonormal, using **Arnoldi** or **Lanczos** iterations, then diagonalizations and matrix exponentials can be performed very efficiently on the much smaller **Krylov space.** This allows to easily simulate quantum dynamics of ~22 spins/qubits on a laptop.

- We used Krylov space (Arnoldi exponentiation) to simulate correlation dynamics in long-range spin-models.

- Open system dynamics: Use **full density matrix simulations** or **Quantum trajectories**

$$\hat{\rho} = \overbrace{\begin{pmatrix} \rho_{1,1} & \rho_{1,2} & \rho_{1,3} \\ \rho_{2,1} & \rho_{2,2} & \rho_{2,3} \\ \rho_{3,1} & \rho_{3,2} & \rho_{3,3} \end{pmatrix}}^{D \times D} \rightarrow \underbrace{\begin{pmatrix} \mathbf{y}_1 & \mathbf{y}_2 & \mathbf{y}_3 \end{pmatrix}}_{\mathbf{y} = \begin{pmatrix} \rho_{1,1}, \rho_{2,1}, \rho_{3,1}, & \rho_{1,2}, \rho_{2,2}, \rho_{3,2}, & \rho_{1,3}, \rho_{2,3}, \rho_{3,3} \end{pmatrix}} \quad \begin{matrix} \mathbf{y}_1^T & \mathbf{y}_2^T & \mathbf{y}_3^T \end{matrix}$$

Master equation is linear, use Krylov space



- Quantum trajectories can be easily derived and simulated using the Kraus operator formalism