Numerical approaches to quantum many-body non-equilibrium

www.schachenmayer.fr

www.cesq.eu

Johannes Schachenmayer: schachenmayer@unistra.fr

The "many-body quantum frontier"

Challenge: Controlled study of macroscopic coherent quantum superposition states

Microscopic world

Macroscopic world

Quantum physics

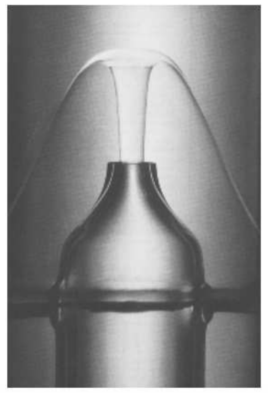
$$\frac{d}{dt}|\psi\rangle = -\frac{\mathrm{i}}{\hbar}\hat{H}|\psi\rangle$$

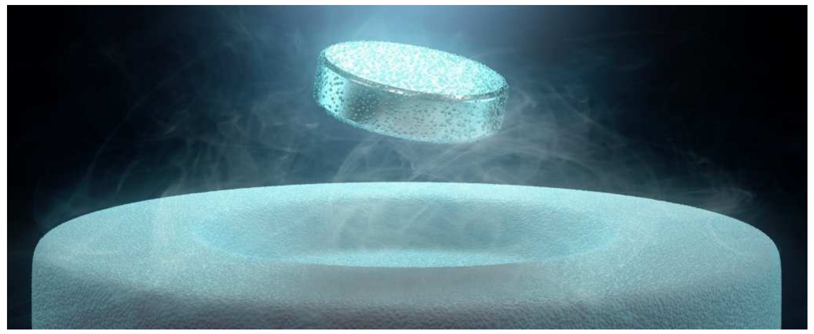
(decoherence)

Classical physics

$$\boldsymbol{F} = m\boldsymbol{a}$$

Macroscopic quantum effects



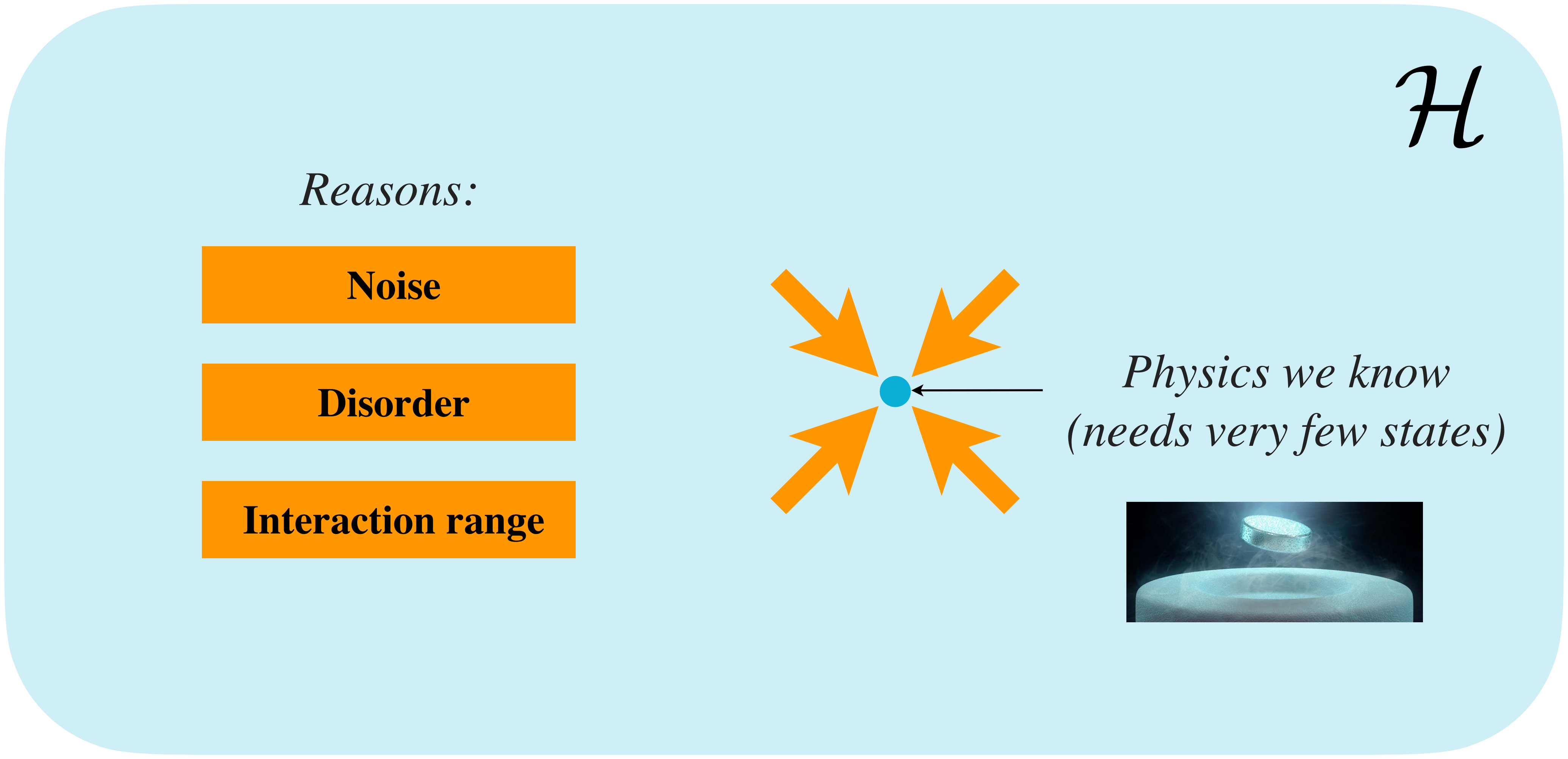


very rare, low temperatures

The “many-body quantum frontier”

Challenge: Controlled study of macroscopic coherent quantum superposition states

- **Many-body system:** *N particles* *Gigantic Hilbert space* $\dim(\mathcal{H}) \sim \exp(N)$



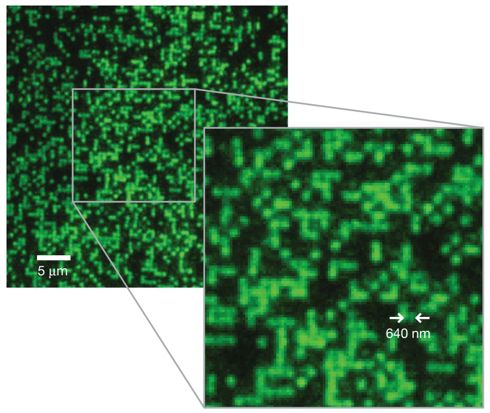
If instead we could create much larger superpositions controllably:

- **Fundamental question:** *Does quantum physics hold on the large scale?*
- **Applications:** *Engineering material properties, enhanced sensing, computing … ?*

The "many-body quantum frontier" - cold atom experiments

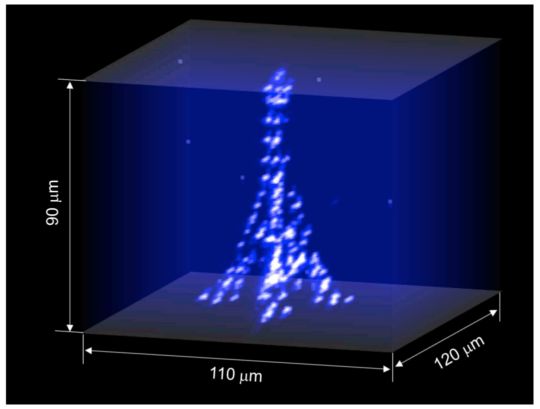
- Experimental platforms available with **cold atom physics:**

Optical lattices



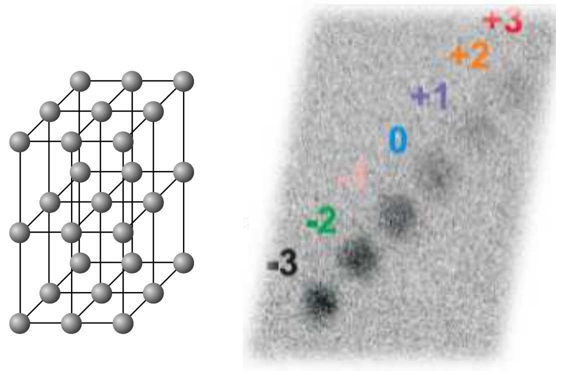
Munich, Harvard, Innsbruck, … many more

Optical tweezers/Rydberg



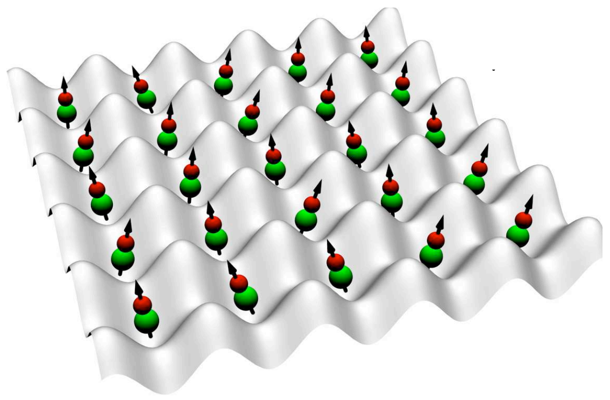
Paris, Harvard, … many more, ***Strasbourg***

Magnetic atoms

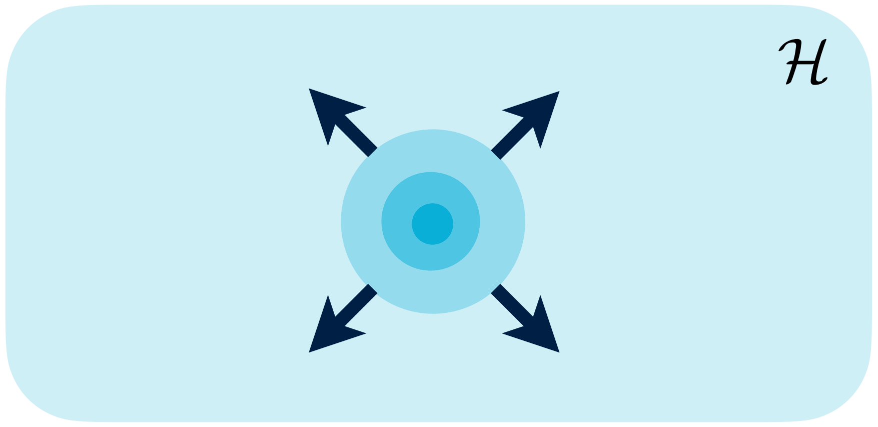


Innsbruck, Paris, Stuttgart … many more

Polar molecules



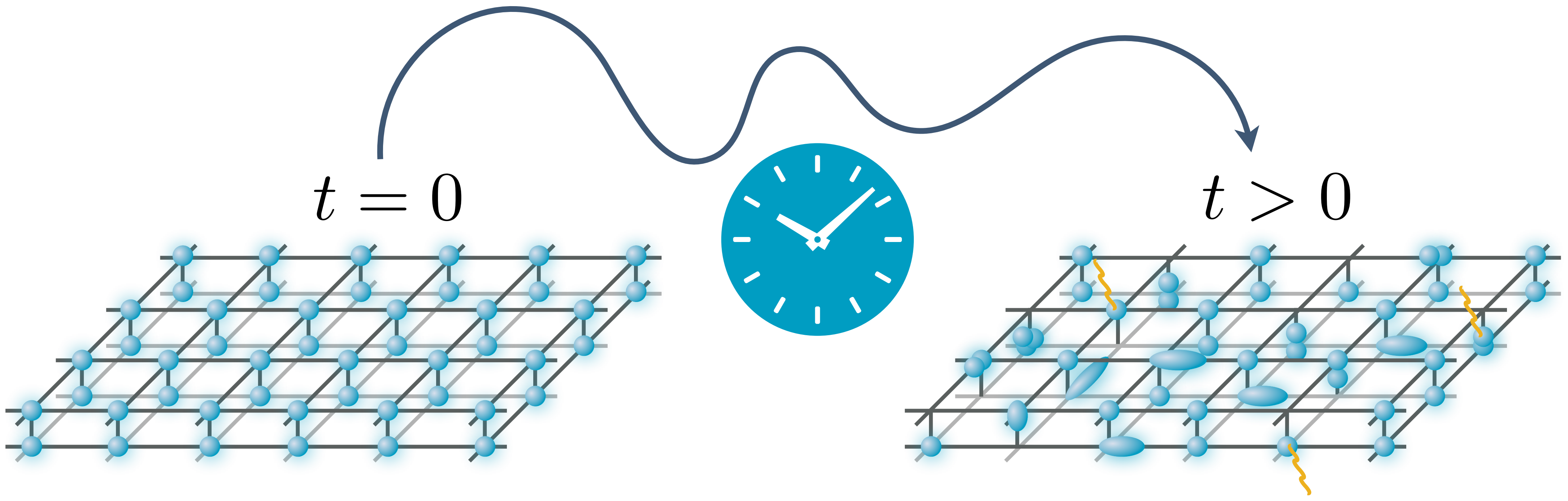
Boulder, Innsbruck, Ulm, … many more



Large Hilbert space access (long coherence times)

… or any quantum computing platform

Non-equilibrium quantum many-body physics: The problem



$|\psi_0\rangle$ *…far out-of equilibrium* $|\psi(t)\rangle$

$$\frac{d}{dt}|\psi\rangle = -\mathrm{i}\hat{H}|\psi\rangle$$

$$\frac{d}{dt}\hat{\rho} = -\mathrm{i}[\hat{H}, \hat{\rho}] + \sum_i \mathcal{L}^{[i]}\hat{\rho}$$

No decoherence

Master-equation noise modeling

(time-evolution & steady states)

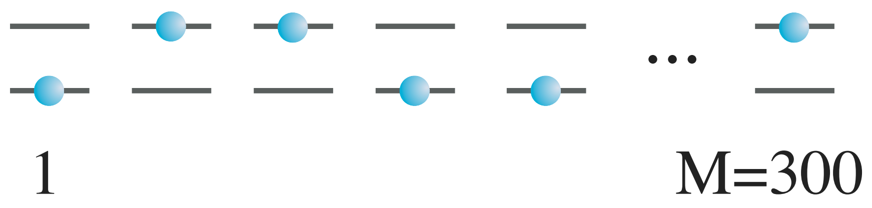
$\hbar \equiv 1$

Numerics: A fundamentally interesting problem

When do numerical simulations become impossible?

$|\psi\rangle \in \mathcal{H}$

$\dim(\mathcal{H}) = 2^M$



$\sim 10^{82}$ gigabyte

Atoms in universe (estimated): $\sim 10^{80}$

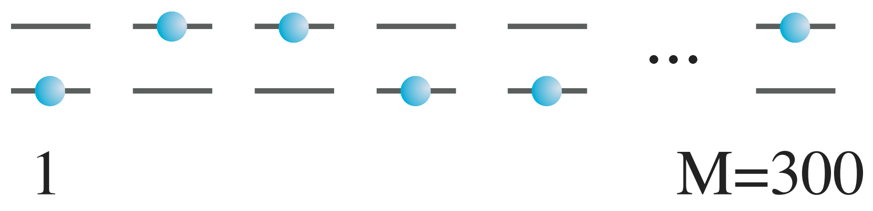
So we need a ***quantum computer*** *to simulate this?*

Lecture 5 - Why is this a fundamentally interesting problem?

When do numerical simulations become impossible?

$|\psi\rangle \in \mathcal{H}$

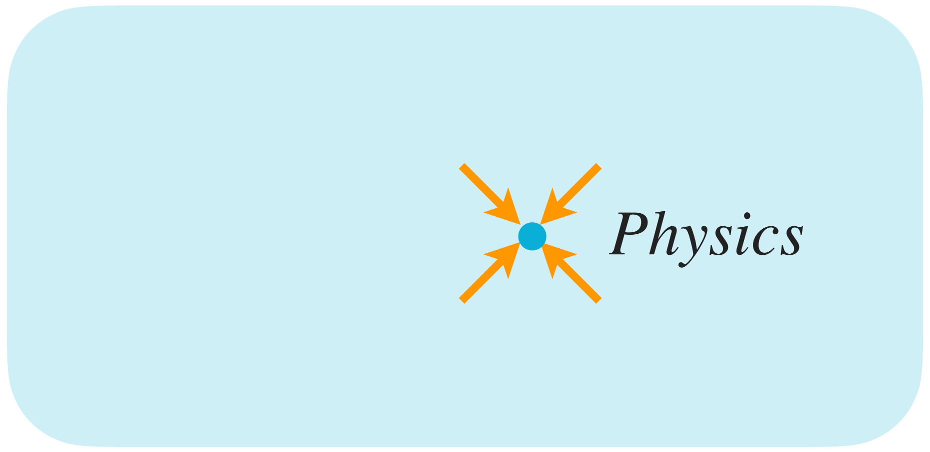
$\dim(\mathcal{H}) = 2^M$



$\sim 10^{82}$ gigabyte

Atoms in universe (estimated): $\sim 10^{80}$

We don't always need a quantum computer to simulate this!



Numerical simulations: A practical tool to understand quantum complexity better!

Numerical simulations of the many-body problem

Quantum many-body nonequilibrium dynamics

$$\frac{d}{dt}|\psi\rangle = -\mathrm{i}\hat{H}|\psi\rangle$$

$$\frac{d}{dt}\hat{\rho} = -\mathrm{i}[\hat{H},\hat{\rho}] + \sum_i \mathcal{L}^{[i]}\hat{\rho}$$

No decoherence

Master-equation noise modeling

(time-evolution & steady states)

$\hbar \equiv 1$

Motivation:

- Model experiments & benchmark the status of quantum platforms (quantum advantage?)
- Find new emergent macroscopic phenomena
- Fundamentally understand quantum many-body from a classical complexity perspective

Numerical approaches to quantum many-body non-equilibrium

Goal:

A tour through some numerical methods for simulating large quantum many-body non-equilibrium dynamics, with examples. Learning physics by simulating it.

Lecture 1: Foundations (QM on a computer), Runge-Kutta, Applications to ultra-cold bosonic systems

Lecture 2: Spin-model physics, Krylov space approaches, Open system methods

Lecture 3: Large systems: Matrix Product States (DMRG), Applications to spin-models and Bose-Hubbard

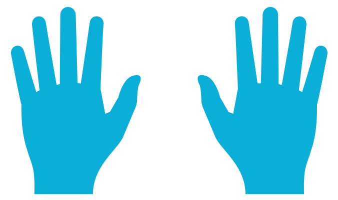
- Some text recommendations:
  - *Numerical recipes - The Art of Scientific Computing (a classic), on-line: https://numerical.recipes*
  - *General references to openly available publications in class*
- Language recommendation (used for examples): Julia, https://julialang.org/ (open source, easy, fast linear algebra)
- What these lectures are **not:**
  - *Complete: Many techniques are not discussed (e.g. Monte-Carlo, Fermions, Phase space methods, …)*
  - *Computer science class: No proofs of complexity etc.*
  - *Numerical tutorial: There will be code snippets … incentive to do it yourself*

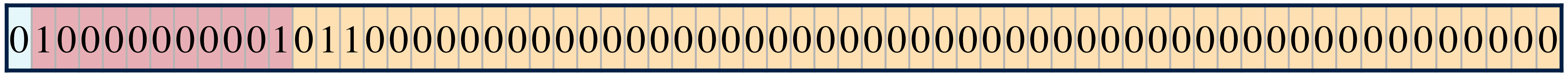
$$\hbar \equiv 1$$

… always!

Lecture 1 - Plan for today

- **Part 1.1:** Some fundamentals about numbers in digital memory and the linear algebra of quantum mechanics



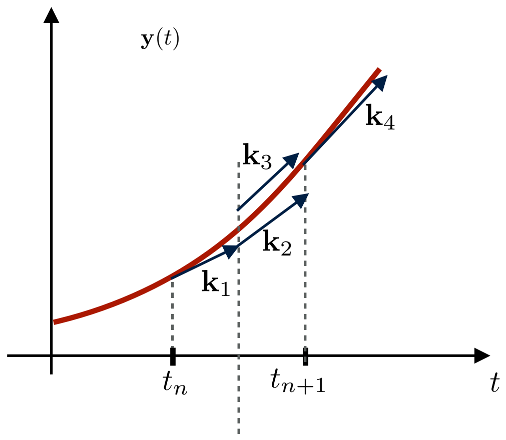


- **Part 1.2:** The many-body problem of the day: Ultra-cold bosons in mean-field approximation (Gross-Pitaevskii, GP)

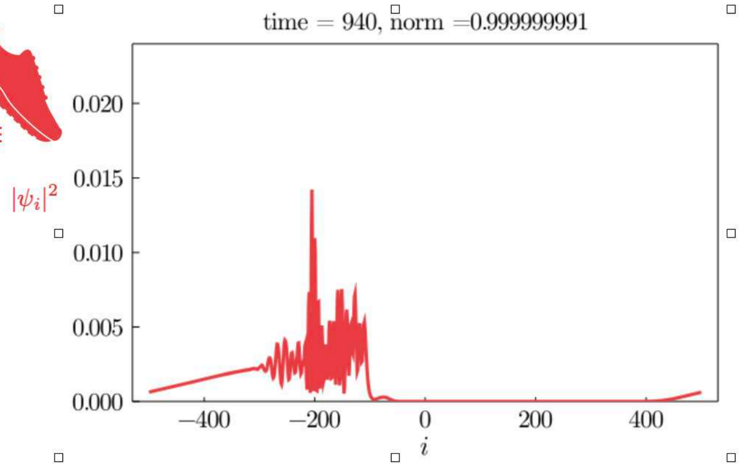
$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

- **Part 1.3:** Runge-Kutta (RK) time-evolution methods: A swiss army knife

$$\begin{aligned}
\mathbf{k}_1 &= f(t_n, \mathbf{y}_n) \\
\mathbf{k}_2 &= f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_1\right) \\
\mathbf{k}_3 &= f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_2\right) \\
\mathbf{k}_4 &= f\left(t_n + h, \mathbf{y}_n + h\mathbf{k}_3\right) \\
\mathbf{y}_{n+1} &\approx \mathbf{y}_n + \frac{h}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)
\end{aligned}$$

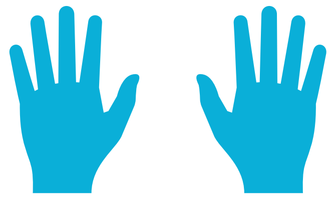


- **Part 1.4:** Applying Runge-Kutta to GP time-evolution



Lecture 1 - Integers on a computer

Latin: digitus = finger



- Computers are digital, they work with bit representations of numbers
- Integers (signed, example with 4 bits)

$m=+0$	$m=+1$	$m=+2$	$m=+3$	$m=+4$	$m=+5$	$m=+6$	$m=+7$
0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1

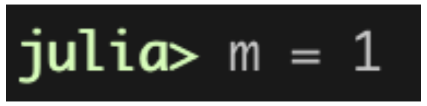
Sign bit *Bit vector:* d_i $d_2\,d_1\,d_0$ $m = d_2 2^2 + d_1 2^1 + d_0 2^0 = \sum_{i=0}^{2} d_i 2^i$

$m=-0$	$m=-1$	$m=-2$	$m=-3$	$m=-4$	$m=-5$	$m=-6$	$m=-7$
1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1

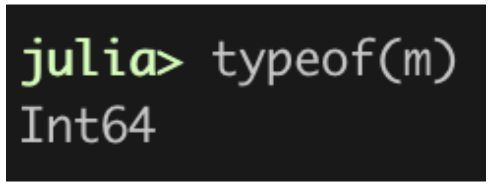
- To not store the double zeros, usual convention is:

$m=-8$	$m=-7$	$m=-6$	$m=-5$	$m=-4$	$m=-3$	$m=-2$	$m=-1$
1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1

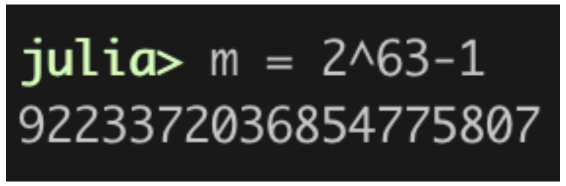
- Representable range of numbers (*n* bits): $(-2^{(n-1)}),\ldots,(2^{(n-1)}-1) \ \sim -10^{18},\ldots 10^{18}$ *(n=64)*



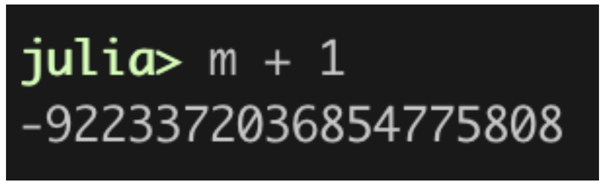
```
julia> m = 1
```



```
julia> typeof(m)
Int64
```



```
julia> m = 2^63-1
9223372036854775807
```



```
julia> m + 1
-9223372036854775808
```

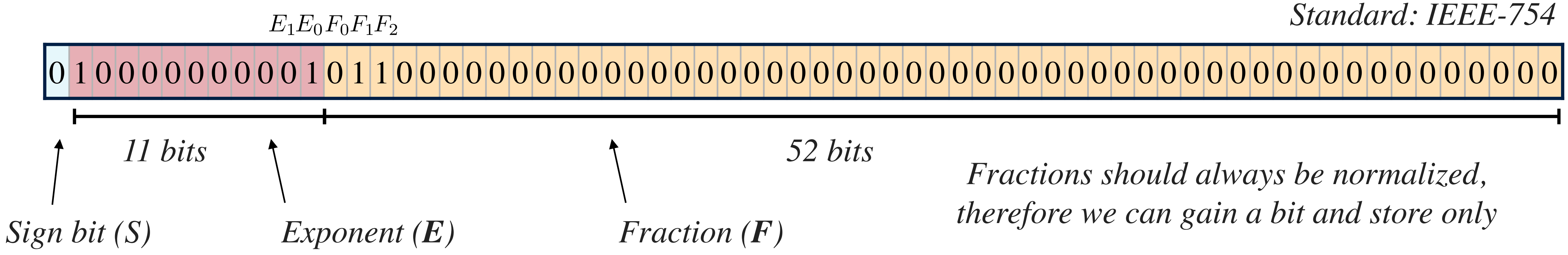
- **Pro-Tip:** When really pushing code performance it can sometimes help to use bitwise operations

Example: Figuring out qubit basis-state numbers

Lecture 1 - Floating point numbers on a computer

- Floating point numbers: $\pm 1.\text{fraction} \times 10^{\text{exponent}}$ … using **64 bits** (“double precision”)

Standard: IEEE-754



Fractions should always be normalized, therefore we can gain a bit and store only

$$1.F$$

- Then, in binary represented numbers are

fraction: $f = 1 + F_0\,2^{-1} + F_1\,2^{-2} + F_2\,2^{-3} + \cdots = 1 + \sum_{i=0}^{51} F_i\,2^{-i-1}$

exponent convention: $\mathcal{E} = E - 1023 = \left(\sum_{i=1}^{11} E_i\,2^i\right) - 1023$

$$r = (-1)^S \times f \times 2^{\mathcal{E}}$$

- **Example:** $\mathcal{E} = 2^{10} + 2^0 - (2^{10} - 1) = 2$ $\quad \mathcal{F} = 2^0 + 2^{-2} + 2^{-3} = 1.375$

  (above)

$$r = (-1)^0 \times 1.375 \times 2^2 = +5.5$$

- **Important:** Relative precision $1.0 \approx 1.0 \pm 2^{-53} \approx 1.0 \pm 10^{-16}$

- This means in practice: **Keep units normalized and consider everything below 1e-16 zero** $\hbar \equiv 1$ *definitely!*

In practice, pick a time unit by normalizing an energy, e.g.: $\hat{H} = -J \sum_i (\hat{b}_i \hat{b}^\dagger_{i+1} + \hat{b}^\dagger_i \hat{b}_{i+1})$ $\quad J \equiv 1$

Lecture 1 - Linear algebra of quantum mechanics

State = Vector

$$\begin{bmatrix} \vdots \\ \psi_i \\ \vdots \end{bmatrix} \quad D \times 1$$

In general: Complex elements

$$D \times (2 \times 64)\,\text{Bits} = D \times 16\,\text{Bytes}$$

Operators = Matrix

$$\begin{bmatrix} \vdots & \vdots & \vdots \\ \vdots & h_{i,j} & \vdots \\ \vdots & \vdots & \vdots \end{bmatrix} \quad D \times D$$

Hamiltonian, Observables, Time-evolution operator

- Let's define a state-vector as a general concept (not limited to linear quantum mechanics)

$$\mathbf{y}(t) \quad \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix}$$

Ordinary differential equation

$$\dot{\mathbf{y}}(t) = f(t, \mathbf{y}(t))$$

Schrödinger equation $\quad \frac{d}{dt}|\psi\rangle = -\mathrm{i}\hat{H}|\psi\rangle$

$$\mathbf{y} = |\psi\rangle \qquad f(t, \mathbf{y}(t)) = \hat{H}\,|\psi\rangle$$ *(linear)*

- The state-vector can be anything:

Linearized density matrix

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

Two classical particles

$$\mathbf{y} = [x_1, p_1, x_2, p_2]^T$$

$\ldots$

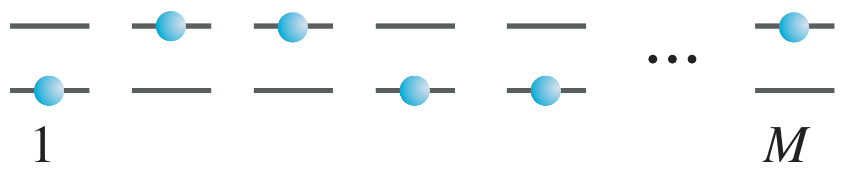
Lecture 1 - Linear algebra of quantum mechanics

- Fundamental limit, how far can we go
- Ultimately this will mean a memory limitation

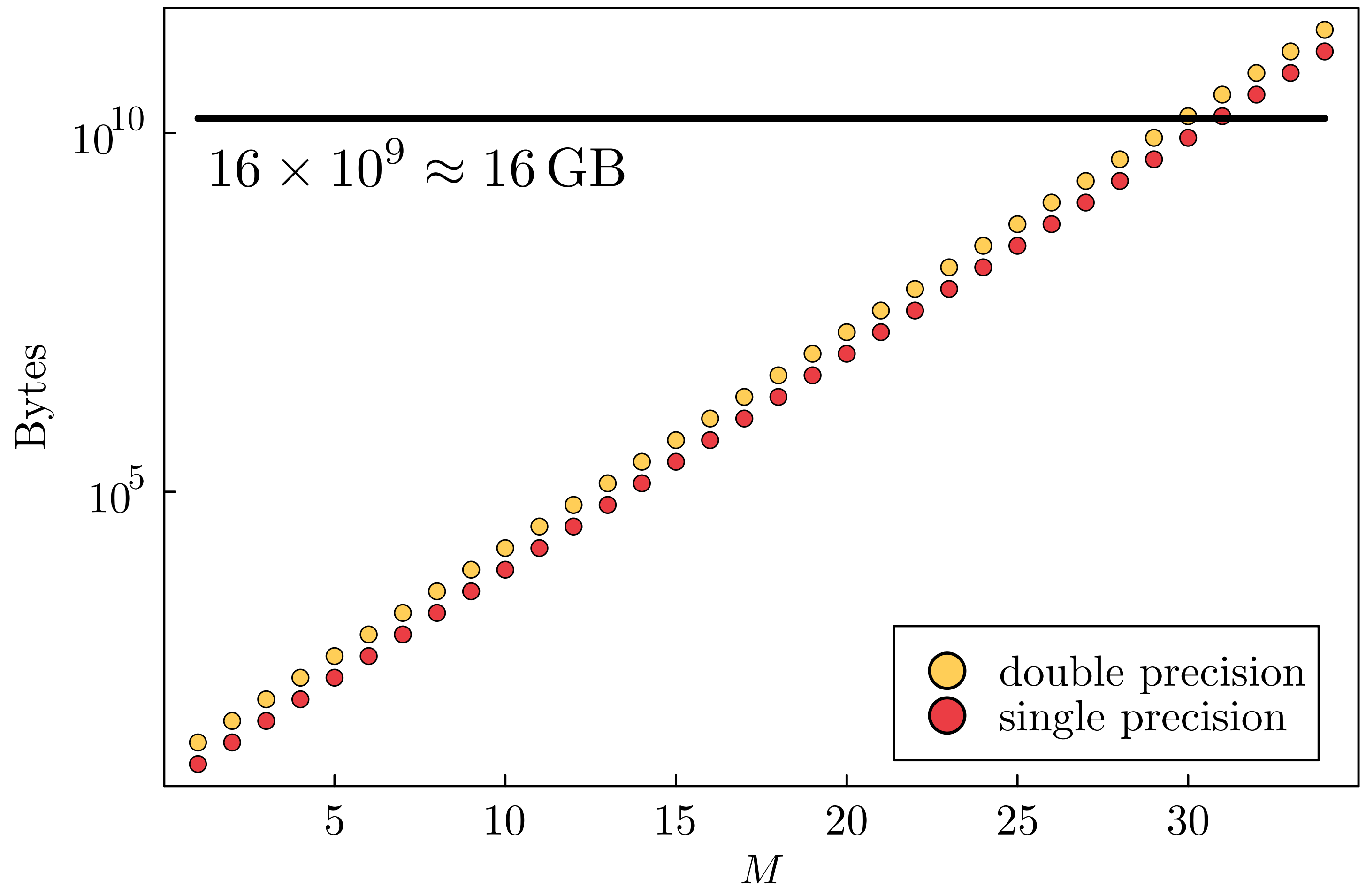
Double precision $D \times (2 \times 64)\,\text{Bits} = D \times 16\,\text{Bytes}$

Single precision $D \times 8\,\text{Bytes}$

- System of qubits



$D = 2^M$



$M \approx 30$

doable on current hardware

- **Remark:** Of course the Hamiltonian would be twice the size, so we have to be smart about that (see lecture 2)
- **Remark:** Later we will see that we can easily simulate larger systems, using "state compression" (see lecture 3)

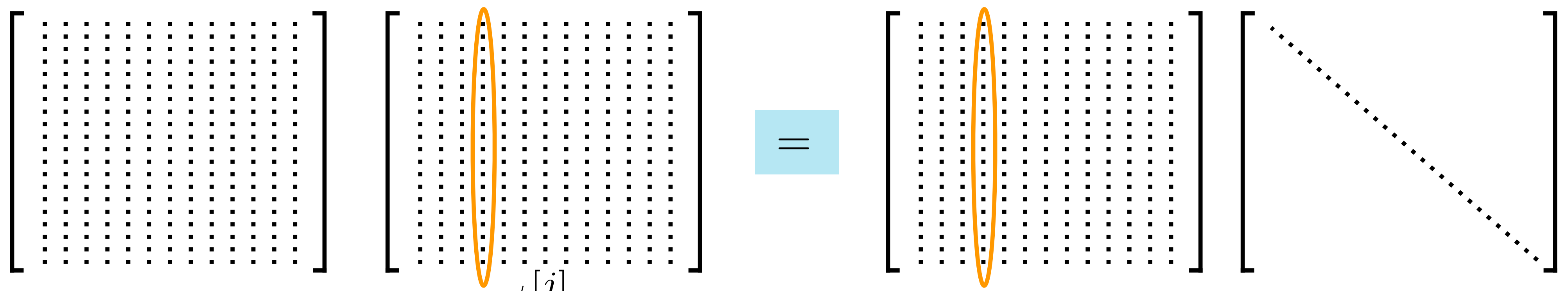
Lecture 1 - Exact Diagonalization

- Exact simulations of quantum mechanics often loosely described as: **Exact diagonalization (ED)**
- **Summary:** Diagonalization

Hamiltonian is Hermitian: real eigenvalues, unitary transformation $\hat{H}^\dagger = \hat{H}$ $\quad D \times D$ *matrix*

Unitary matrix $\hat{V}^\dagger \hat{V} = \mathbb{1} \quad \hat{V}^\dagger = \hat{V}^{-1} \quad \hat{V}^\dagger \hat{H} \hat{V} = \hat{E} \Leftrightarrow \hat{H}\hat{V} = \hat{V}\hat{E} \Leftrightarrow \hat{H} = \hat{V}\hat{E}\hat{V}^\dagger$

$(\hat{H})_{i,j} \equiv h_{i,j} \quad (\hat{V})_{i,j} \equiv v_{i,j} \quad (\hat{V})_{i,j} \equiv v_{i,j} \quad (\hat{E})_{i,j} \equiv e_{i,j}$



$\phi_i^{[j]} \equiv v_{i,j}$

Diagonal matrix

$e_{i,j} \equiv \delta_{i,j} E_j$

- The j-th column of the matrix *v* is the eigenvector corresponding to the j-th eigenvalue:

$$\sum_k h_{i,k} v_{k,j} = \sum_k v_{i,k} e_{k,j} = E_j v_{i,j} \qquad \phi_i^{[j]} \equiv v_{i,j} \qquad \sum_k h_{i,k} \phi_k^{[j]} = E_j \phi_i^{[j]}$$

- The columns of V are the "eigenkets" $(|\phi_j\rangle)_i \equiv v_{i,j}$ $\quad \hat{H}|\phi_j\rangle = E_j |\phi_j\rangle$
- … the rows are the "eigenbras" (complex conjugated eigenvectors), which follows from $\hat{V}^\dagger \hat{H} = \hat{E}\hat{V}^\dagger$

Lecture 1 - Exact Diagonalization

- Exact simulations of quantum mechanics often loosely described as: **Exact diagonalization (ED)**
- **Summary:** Diagonalization

Hamiltonian is Hermitian: real eigenvalues, unitary transformation $\hat{H}^\dagger = \hat{H}$ $\quad D \times D$ *matrix*

Unitary matrix $\hat{V}^\dagger \hat{V} = \mathbb{1} \quad \hat{V}^\dagger = \hat{V}^{-1} \quad \hat{V}^\dagger \hat{H} \hat{V} = \hat{E} \Leftrightarrow \boxed{\hat{H}\hat{V} = \hat{V}\hat{E}} \Leftrightarrow \boxed{\hat{H} = \hat{V}\hat{E}\hat{V}^\dagger}$

$$(\hat{E})_{i,j} \equiv \delta_{i,j} E_j$$

- A diagonalization gives us a full spectral decomposition: $\boxed{\hat{H} = \sum_k E_k |\phi_k\rangle \langle\phi_k|}$

- With this we can solve Schrödinger equation time-evolution

$$\frac{d}{dt}|\psi\rangle = -\mathrm{i}\hat{H}|\psi\rangle \qquad |\psi(t)\rangle = \underset{\uparrow}{\mathrm{e}^{-\mathrm{i}t\hat{H}}}\, |\psi(t=0)\rangle$$

Time-evolution operator (matrix exponential)

$$\mathrm{e}^{-\mathrm{i}\hat{H}t} = \sum_{n=0} \frac{(-\mathrm{i}t\hat{H})^n}{n!} = \sum_k \mathrm{e}^{-\mathrm{i}tE_k} |E_k\rangle \langle E_k|$$

- Diagonalization allows to compute the matrix exponential, and to compute exact evolution (no time-stepping needed)

$$|\psi(t)\rangle = \sum_k \mathrm{e}^{-\mathrm{i}tE_k} |E_k\rangle \langle E_k|\psi(t=0)\rangle$$

Lecture 1 - Exact Diagonalization

- Create random Hamiltonian

```
julia> H = rand(ComplexF64, 100, 100); H += H'
```

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

- Compute V and E

```
julia> using LinearAlgebra

julia> E, V = eigen(H);
```

E comes out as vector

- Check equations:

```
julia> norm(H*V .- V*Diagonal(E))
2.3421977872625636e-13
```

```
julia> norm(H .- V*Diagonal(E)*V')
7.79771749285727e-13
```

Precision of diagonalization algorithm (not quite double, but enough)

- Time evolution with matrix exponential

$$|\psi(t)\rangle = \sum_k \mathrm{e}^{-\mathrm{i}tE_k} |E_k\rangle \langle E_k|\psi(t=0)\rangle$$

```
psi0 = zeros(ComplexF64, D)
psi0[1] = 1.0
psit = zeros(ComplexF64, D)
for kk = 1:D
    ovl = V[:, kk]' * psi0
    psit += exp(-1im * t * E[kk]) * ovl .* V[:, kk]
end
```

- Also possible, build matrix exponential directly with built-in routines:

```
julia> U = exp(-1im .* t .* H); psit = U*psi0
100-element Vector{ComplexF64}:
```

Warning: Make sure exp is not element-wise exponential!

Lecture 1 - Plan for today

- **Part 1.1:** Some fundamentals about numbers in digital memory and the linear algebra of quantum mechanics

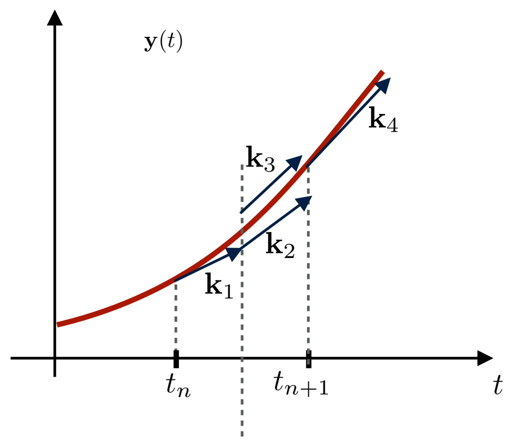
0 10000000001 0110000000000000000000000000000000000000000000000000

- **Part 1.2:** The many-body problem of the day: Ultra-cold bosons in mean-field approximation (Gross-Pitaevskii, GP)

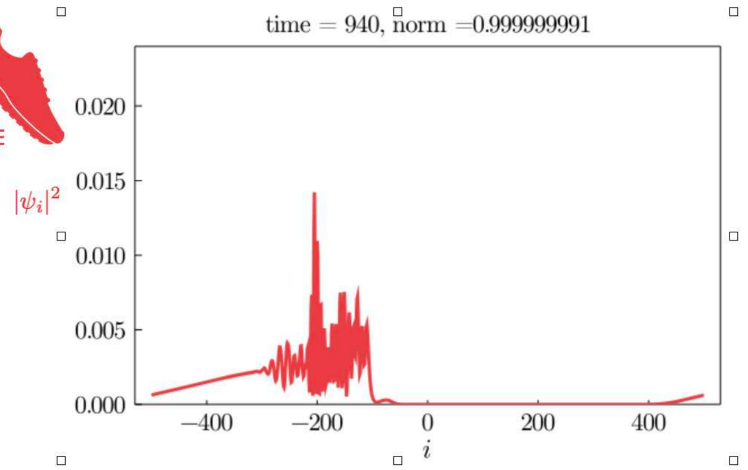
$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

- **Part 1.3:** Runge-Kutta (RK) time-evolution methods: A swiss army knife

$$\begin{aligned}
\mathbf{k}_1 &= f(t_n, \mathbf{y}_n)\\
\mathbf{k}_2 &= f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_1\right)\\
\mathbf{k}_3 &= f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_2\right)\\
\mathbf{k}_4 &= f\left(t_n + h, \mathbf{y}_n + h\mathbf{k}_3\right)\\
\mathbf{y}_{n+1} &\approx \mathbf{y}_n + \frac{h}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)
\end{aligned}$$



- **Part 1.4:** Applying Runge-Kutta to GP time-evolution



Lecture 1 - Bose-Einstein condensation physics

- **Starting point:** A dilute gas of *N* ultra-cold bosons trapped in some potential, contact-type scattering interactions

$$\hat{H} = \hat{H}_0 + \hat{H}_{\text{int}} \qquad \hat{H}_{\text{int}} = \frac{g}{2}\int dx_1 \int dx_2\, \delta(x_1 - x_2)\hat{\psi}^\dagger_{x_1}\hat{\psi}^\dagger_{x_2}\hat{\psi}_{x_2}\hat{\psi}_{x_1} = \frac{g}{2}\int dx\, \hat{\psi}^\dagger_x\hat{\psi}^\dagger_x\hat{\psi}_x\hat{\psi}_x$$

see e.g. *Yvan Castin,* http://www.arxiv.org/abs/cond-mat/0105058

Written in second quantization with field operators $\hat{\psi}_x$ $\hat{\psi}^\dagger_x\,|\text{vac}\rangle = |x\rangle$ *… creates particle at position x*

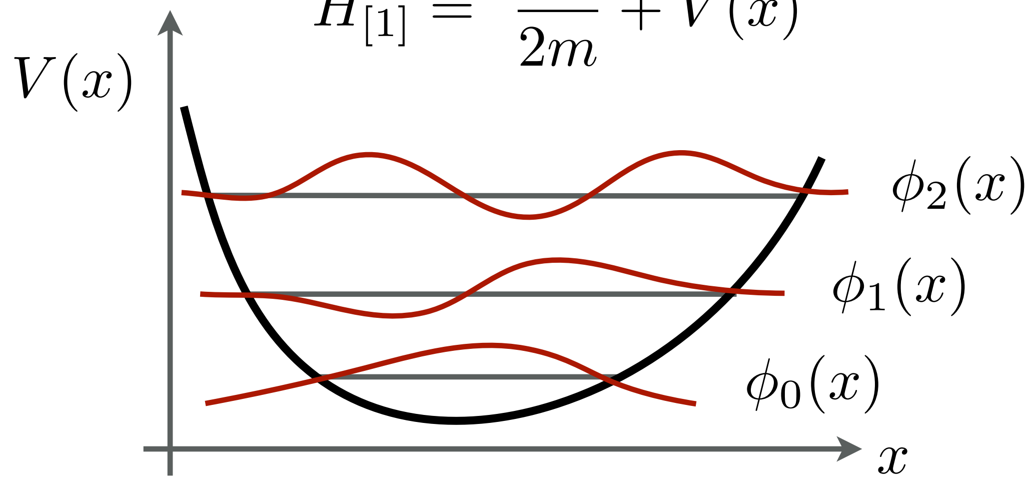
- Bosons: $[\hat{\psi}_x, \hat{\psi}^\dagger_{x'}] = \delta(x - x')$ $[\hat{\psi}_x, \hat{\psi}_{x'}] = [\hat{\psi}^\dagger_x, \hat{\psi}^\dagger_{x'}] = 0$

Technically this should be 3D (but 1D for simplicity)

single particle Hamiltonian

$$\hat{H}_{[1]} = \frac{\hat{p}^2}{2m} + V(x)$$

- … writing the field operators in the basis of the non-interacting problem



$$\hat{\psi}_x = \sum_n \phi_n(x)\hat{a}_n \qquad \hat{H}_0 = \sum_n E_n \hat{a}^\dagger_n \hat{a}_n$$

- Time-evolution of field operator: $\frac{d}{dt}\hat{\psi}_x = \mathrm{i}\big[\hat{H}_0 + \hat{V}, \hat{\psi}_x\big] = -\mathrm{i}\hat{H}_{[1]}\hat{\psi}_x + \mathrm{i}\left[\hat{H}_{\text{int}}, \hat{\psi}_x\right]$

Lecture 1 - Bose-Einstein-condensation physics

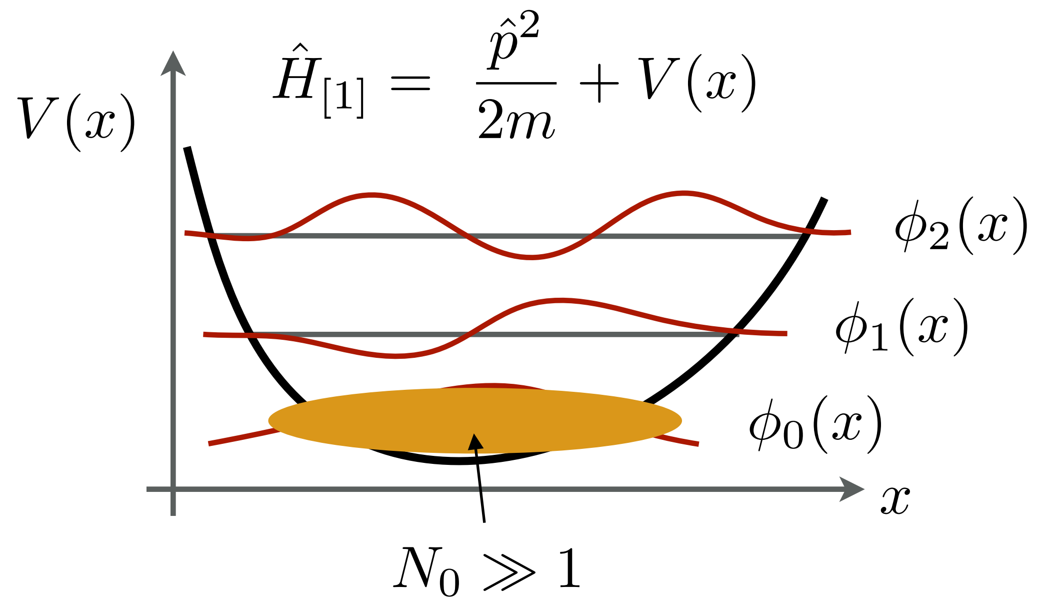
- Time-evolution of field operator: $\frac{d}{dt}\hat{\psi}_x = \mathrm{i}[\hat{H}_0 + \hat{V}, \hat{\psi}_x] = -\mathrm{i}\hat{H}_{[1]}\hat{\psi}_x + \mathrm{i}\left[\hat{H}_{\mathrm{int}}, \hat{\psi}_x\right]$

$$\left[\hat{H}_{\mathrm{int}}\hat{\psi}_x\right] = \frac{g}{2}\int dx' \left[\hat{\psi}^\dagger_{x'}\hat{\psi}^\dagger_{x'}\hat{\psi}_{x'}\hat{\psi}_{x'}, \hat{\psi}_x\right] = \frac{g}{2}\left[\hat{\psi}^\dagger_x\hat{\psi}^\dagger_x\hat{\psi}_x\hat{\psi}_x, \hat{\psi}_x\right] = \frac{g}{2}\left[\hat{\psi}^\dagger_x\hat{\psi}^\dagger_x, \hat{\psi}_x\right]\hat{\psi}_x\hat{\psi}_x$$

$$[\hat{A}\hat{B}, \hat{C}] = \hat{A}[\hat{B}, \hat{C}] + [\hat{A}, \hat{C}]\hat{B} \qquad [\hat{\psi}_x, \hat{\psi}^\dagger_{x'}] = \delta(x - x')$$

$$= \frac{g}{2}\left(\hat{\psi}^\dagger_x\left[\hat{\psi}^\dagger_x, \hat{\psi}_x\right] + \left[\hat{\psi}^\dagger_x, \hat{\psi}_x\right]\hat{\psi}^\dagger_x\right)\hat{\psi}_x\hat{\psi}_x = -g\hat{\psi}^\dagger_x\hat{\psi}_x\,\hat{\psi}_x$$

$$\frac{d}{dt}\hat{\psi}_x = -\mathrm{i}\left(\hat{H}_{[1]} + g\hat{\psi}^\dagger_x\hat{\psi}_x\right)\hat{\psi}_x$$



- Make a strong (mean-field) approximation for *N* particles

$$\hat{a}_0|\psi_0\rangle \approx \sqrt{N_0}|\psi_0\rangle \approx \sqrt{N_0 - 1}|\psi_0\rangle \approx \sqrt{N_0 + 1}|\psi_0\rangle$$

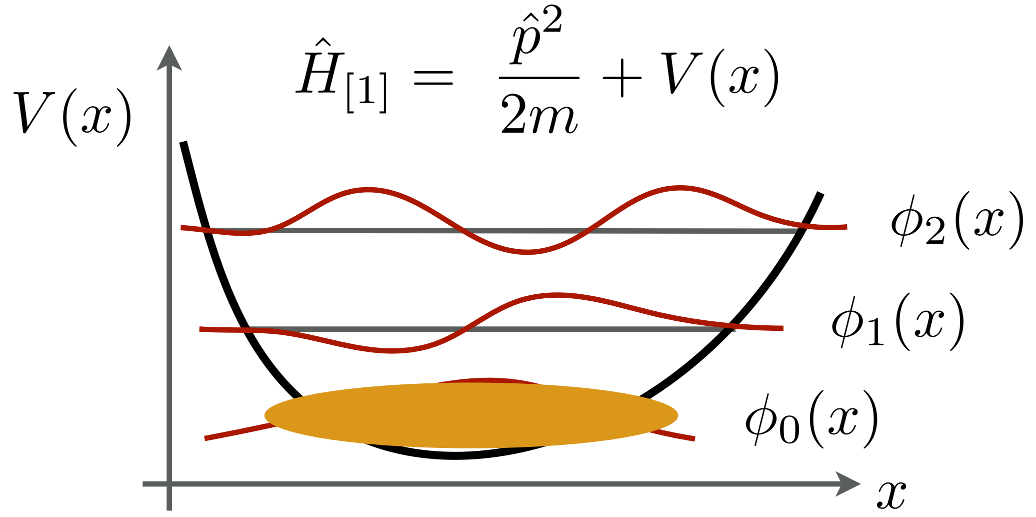
$$\hat{a}_0 \approx \hat{a}_0^\dagger \approx \sqrt{N_0} \approx \sqrt{N}$$

$$\hat{\psi}_x = \sum_n \phi_n(x)\hat{a}_n = \phi_0(x)\hat{a}_0 + \cancel{\sum_{n>0}\hat{a}_n} \approx \sqrt{N}\phi_0(x) \equiv \psi(x)$$

Let's assume a many-body (low energy) state $|\psi_0\rangle$, where almost all particles are in the lowest orbital

Lecture 1 - Bose-Einstein-condensation physics

$$\frac{d}{dt}\hat{\psi}_x = -\mathrm{i}\left(\hat{H}_{[1]} + g\hat{\psi}_x^\dagger\hat{\psi}_x\right)\hat{\psi}_x$$



Let's assume a many-body (low energy) state $|\psi_0\rangle$, where almost all particles are in the lowest orbital

- Make a strong (mean-field) approximation for *N* particles

$$\hat{\psi}_x = \sum_n \phi_n(x)\hat{a}_n = \phi_0(x)\hat{a}_0 + \cancel{\sum_{n>0}\hat{a}_n} \approx \sqrt{N}\phi_0(x) \equiv \psi(x)$$

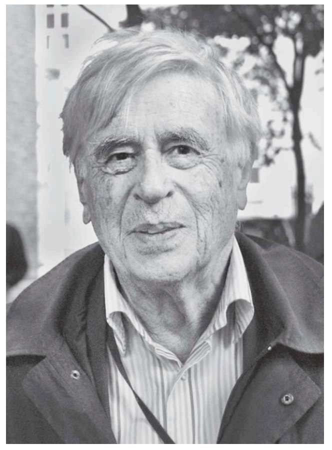
Make the quantum field operator a classical field!

- Then

$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

Time-dependent Gross-Pitaevskii equation!

Eugene P. Gross (1926-1991)


Lev Pitaevskii (1933-2022)

- A **non-linear** equation for a **classical field** $\psi(x,t)$ describing a condensate of many particles
- Also known as non-linear Schrödinger equation, but bad phraseology!
- The equation is derived in a limit where all particles are in the same state (**product state**), 1st quantization:

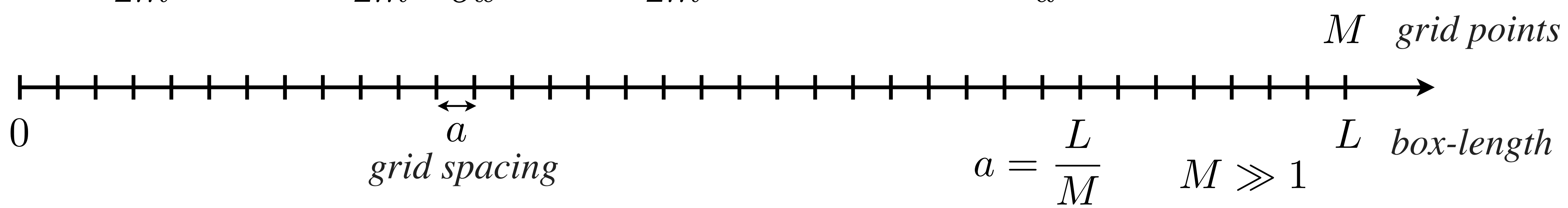
$$\phi(x_1, x_2, \ldots, x_N) = \bigotimes_{i=1}^{N}\phi_0(x_i)$$

Lecture 1 - GP equation - state space considerations

$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

- Let's solve GP numerically!
- For a continuous space, we need to introduce a discretization

$$\frac{\hat{p}^2}{2m}\psi(x) = -\frac{1}{2m}\frac{\partial^2\psi(x)}{\partial x^2} = -\frac{1}{2m}\lim_{a\to 0}\frac{\psi(x+a) - 2\psi(x) + \psi(x-h)}{a^2}$$



$$a = \frac{L}{M} \qquad M \gg 1$$

- Then the matrix of the operator

$$\frac{1}{2ma^2}\begin{pmatrix} -1 & 0 & -1 & & & & \\ & -1 & 0 & -1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -1 & 0 & -1 & \\ & & & & -1 & 0 & -1 \end{pmatrix} + \text{const.} \qquad \dots\textit{acting on}\dots \qquad \begin{pmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_{M-1} \\ \psi_M \end{pmatrix}$$

- **Note:** Kinetic energy is just nearest-neighbor hopping on a grid with amplitude $J = \frac{1}{2ma^2}$
- Indeed, the physics in continuous space is **identical to lattice physics** $\hat{H} = -J\sum_i \left(|i\rangle\langle i+1| + |i+1\rangle\langle i|\right)$

Lecture 1 - Bose-Einstein-condensation physics

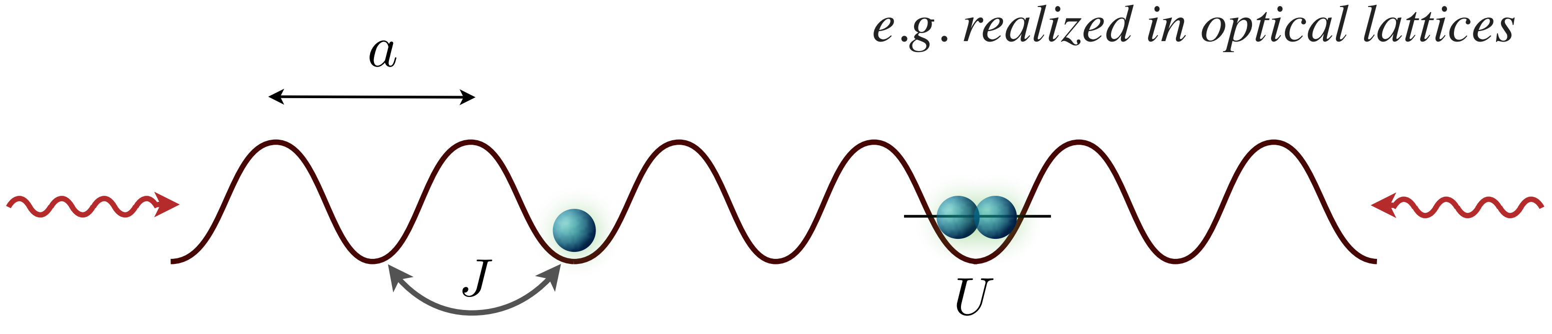
- **In fact:** The physics of the continuous model is identical to those on a lattice

Bose-Hubbard model

e.g. realized in optical lattices

free space (artificial grid)

$$a \to 0 \qquad \Leftrightarrow$$



$$\hat{H} = \hat{H}_0 + \frac{g}{2}\int dx\, \hat{\psi}_x^\dagger \hat{\psi}_x^\dagger \hat{\psi}_x \hat{\psi}_x$$

$$\hat{H} = -J\sum_i (\hat{b}_i \hat{b}_{i+1}^\dagger + \hat{b}_i^\dagger \hat{b}_{i+1}) + \frac{U}{2}\sum_i \hat{b}_i^\dagger \hat{b}_i^\dagger \hat{b}_i \hat{b}_i$$

- **Hilbert space size:** N particles on M sites:

$$D = \frac{(M+N-1)!}{(M-1)!N!}$$

Examples

$M = N = 16$	$D \approx 3 \times 10^8$	$\approx 5\,\mathrm{GB}$
$M = 100, N = 5$	$D \approx 1 \times 10^8$	$\approx 1.5\,\mathrm{GB}$

- **… in the mean-field approximation:** State-vector size is only $\tilde{D} = N$

$$\begin{pmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_{M-1} \\ \psi_M \end{pmatrix}$$

We can now treat huge systems!

- **… but:** The price we pay: **I. We made a strong approximation**, **II. The equations are now non-linear**

Lecture 1 - Plan for today

- **Part 1.1:** Some fundamentals about numbers in digital memory and the linear algebra of quantum mechanics

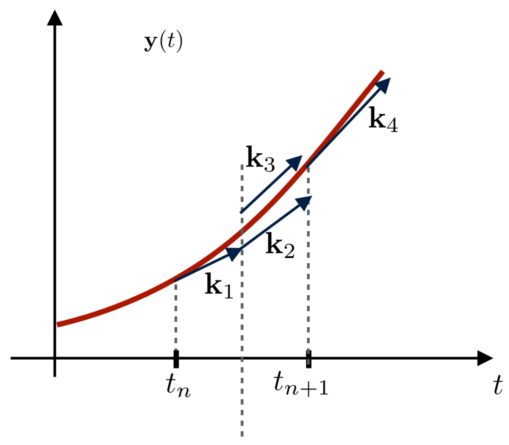
0100000000010110000000000000000000000000000000000000000000000000

- **Part 1.2:** The many-body problem of the day: Ultra-cold bosons in mean-field approximation (Gross-Pitaevskii, GP)

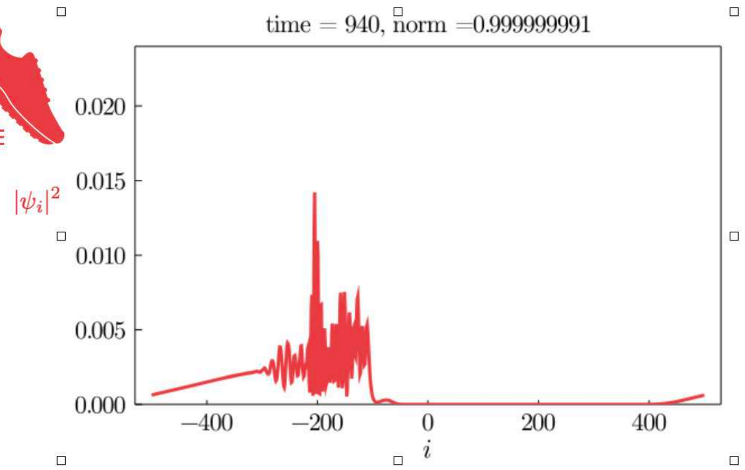
$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

- **Part 1.3:** Runge-Kutta (RK) time-evolution methods: A swiss army knife

$$\begin{aligned}
\mathbf{k}_1 &= f(t_n, \mathbf{y}_n) \\
\mathbf{k}_2 &= f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_1\right) \\
\mathbf{k}_3 &= f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_2\right) \\
\mathbf{k}_4 &= f\left(t_n + h, \mathbf{y}_n + h\mathbf{k}_3\right) \\
\mathbf{y}_{n+1} &\approx \mathbf{y}_n + \frac{h}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)
\end{aligned}$$



- **Part 1.4:** Applying Runge-Kutta to GP time-evolution



Runge-Kutta Methods

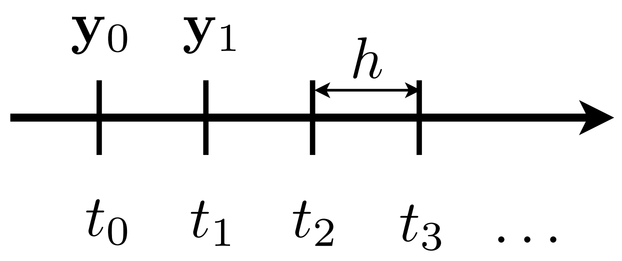
- A general class of standard methods for initial value problems (“Swiss army knife”)

$$\dot{\mathbf{y}}(t) = f(t, \mathbf{y}(t)) \qquad \mathbf{y}(t_n) = \mathbf{y}_n \qquad \textit{Find:} \quad \mathbf{y}_{n+1} \approx \mathbf{y}(t_n + h)$$

$\mathbf{y}(\dots)$ *“exact"*

$\mathbf{y}_n$ *“numerical approximation”*

- Time-discretization:



- Note: This includes linear Schrödinger equation, but also non-linear (e.g. GP) problems

Schrödinger equation

$$\frac{d}{dt}|\psi\rangle = -\mathrm{i}\hat{H}|\psi\rangle$$

$$f(t, \mathbf{y}(t)) = \mathbf{A} \cdot \mathbf{y}(\mathbf{t})$$

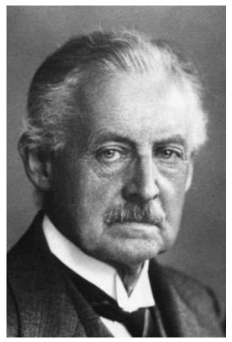
GP equation

$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

discretized $\psi(x,t) = \psi_i(t)$

$$f(t, \mathbf{y}(t)) = \mathbf{A}(\mathbf{y}(t)) \cdot \mathbf{y}(\mathbf{t})$$

$$\mathbf{y} = \begin{pmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_{M-1} \\ \psi_M \end{pmatrix}$$



Carl David Tolmé Runge (1856-1927)

Martin Kutta (1867–1944)

Runge-Kutta Methods: 1st order - explicit Euler

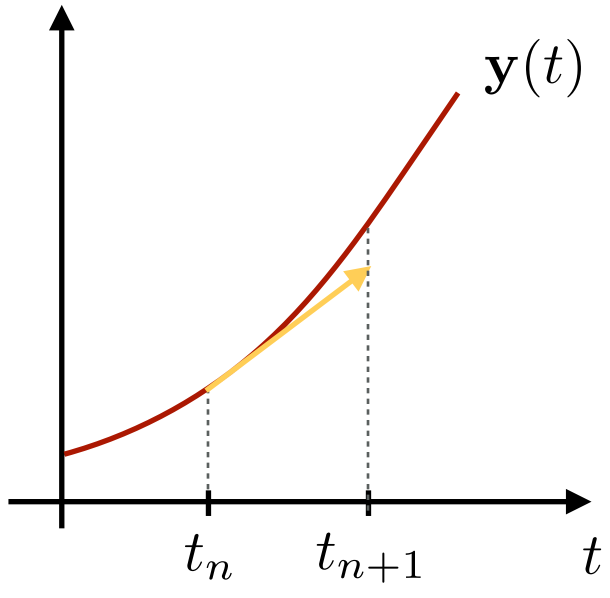
$$\dot{\mathbf{y}}(t) = f(t, \mathbf{y}(t)) \qquad \mathbf{y}(t_n) = \mathbf{y}_n \qquad \textit{Find:} \quad \mathbf{y}_{n+1} \approx \mathbf{y}(t_n + h)$$

- Let's find a method from Taylor expansion of $\mathbf{y}(t)$

$$\mathbf{y}(t_n + h) = \mathbf{y}(t_n) + h\dot{\mathbf{y}}(t_n) + \frac{h^2}{2}\ddot{\mathbf{y}}(t_n) + \cdots = \mathbf{y}(t_n) + hf(t_n, \mathbf{y}_n) + \mathcal{O}(h^2)$$

$$\mathbf{y}_{n+1} \approx \mathbf{y}_n + hf(t_n, \mathbf{y}_n)$$

"explicit Euler method"

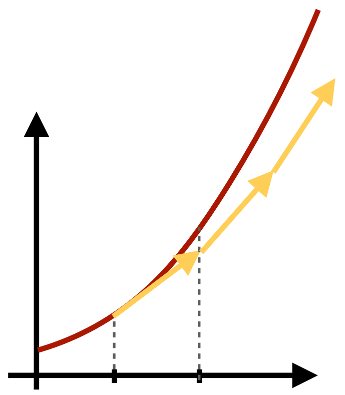


- Not a good method for several reasons

Error is large $\mathcal{O}(h^2)$

… need tiny h

Solution is often not stable!



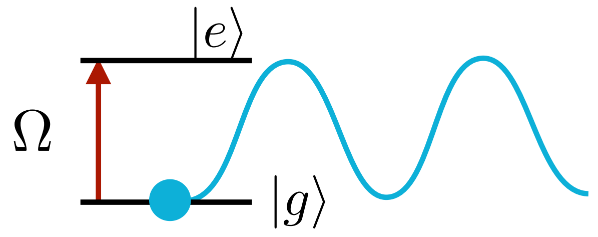
"error grows in same direction"

Runge-Kutta Methods: 1st order

$$\dot{\mathbf{y}}(t) = f(t, \mathbf{y}(t)) \qquad \mathbf{y}(t_n) = \mathbf{y}_n \qquad \textit{Find:}\quad \mathbf{y}_{n+1} \approx \mathbf{y}(t_n + h)$$

- … very simple example (Rabi oscillations)

$$\hat{H} = \begin{pmatrix} 0 & \Omega \\ \Omega & 0 \end{pmatrix} \qquad \frac{d}{dt}|\psi\rangle = -\mathrm{i}\hat{H}|\psi\rangle \qquad |\psi_0\rangle = \begin{array}{cc} \begin{pmatrix} 1 \\ 0 \end{pmatrix} & \begin{array}{c} |g\rangle \\ |e\rangle \end{array} \end{array}$$



Compute:

$$n_e(t) = |\,\langle\psi(t)|e\rangle\,|^2$$

Exact:

$$n_e(t) = \sin^2(t\Omega)$$

$\Omega \equiv 1$

```
h = 0.05
steps = 200

psi = [1;0]
ne = zeros(steps+1)
ne[1] = abs(psi[2])^2
for tt = 1:steps
    psi = rk1(H, psi, h)
    ne[tt+1] = abs(psi[2])^2
end
```

- Explicit Euler method:

$$\mathbf{y}_{n+1} \approx \mathbf{y}_n + hf(t_n, \mathbf{y}_n)$$

"explicit Euler method"

```
function rk1(H, y, h)
    y += h .* (-1im .* H * y)
    return y
end
```

Runge-Kutta Methods: 1st order

$$\dot{\mathbf{y}}(t) = f(t, \mathbf{y}(t)) \qquad \mathbf{y}(t_n) = \mathbf{y}_n \qquad \textit{Find:}\quad \mathbf{y}_{n+1} \approx \mathbf{y}(t_n + h)$$

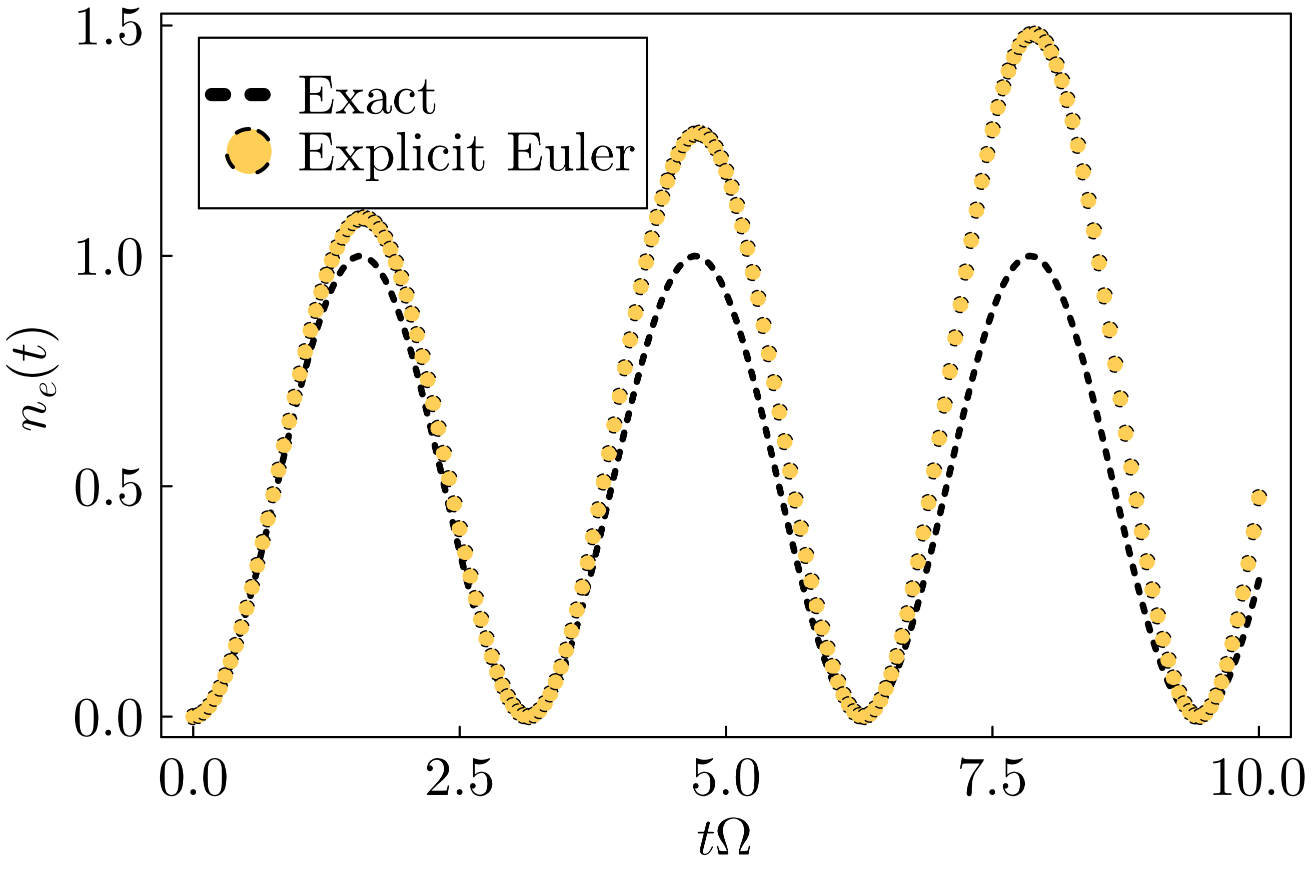
- Explicit Euler method: $\mathbf{y}_{n+1} \approx \mathbf{y}_n + h f(t_n, \mathbf{y}_n)$

"explicit Euler method"

```
function rk1(H, y, h)
    y += h .* (-1im .* H * y)
    return y
end
```

- … very simple example (Rabi oscillations) $\hat{H} = \begin{pmatrix} 0 & \Omega \\ \Omega & 0 \end{pmatrix}$ $\quad \frac{d}{dt}|\psi\rangle = -\mathrm{i}\hat{H}|\psi\rangle \quad |\psi_0\rangle = \begin{array}{cc} \begin{pmatrix} 1 \\ 0 \end{pmatrix} & \begin{array}{c} |g\rangle \\ |e\rangle \end{array} \end{array}$

$$n_e(t) = |\langle\psi(t)|e\rangle|^2$$



Fundamental problem:
Norm keeps increasing!

$$|\psi_{n+1}\rangle = |\psi_n\rangle - \mathrm{i}h\hat{H}\,|\psi_n\rangle$$

$$\langle\psi_{n+1}|\psi_{n+1}\rangle = \langle\psi_n|\psi_n\rangle + h^2\,\langle\psi_n|\,\hat{H}^2\,|\psi_n\rangle$$

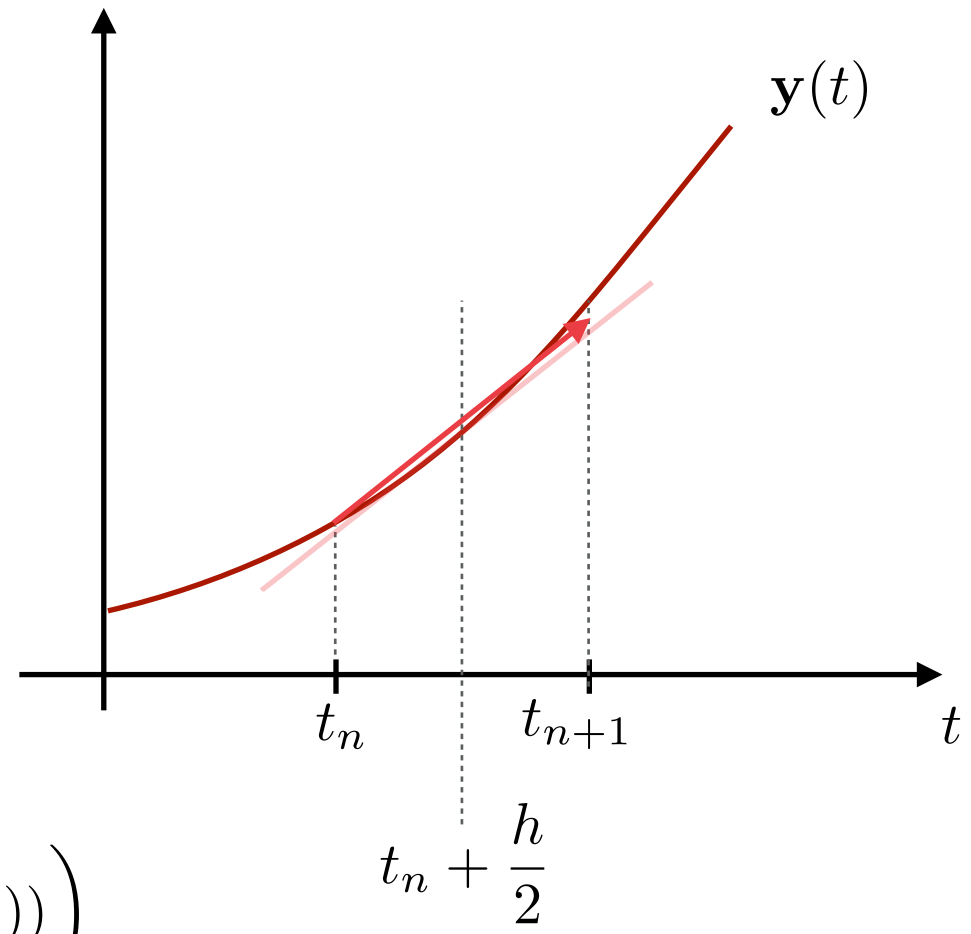
Runge-Kutta Methods: 2nd order - Midpoint methods

$$\dot{\mathbf{y}}(t) = f(t, \mathbf{y}(t)) \qquad \mathbf{y}(t_n) = \mathbf{y}_n \qquad \textit{Find:} \quad \mathbf{y}_{n+1} \approx \mathbf{y}(t_n + h)$$

- Let's find a better method:

$$\mathbf{y}_{n+1} \approx \mathbf{y}_n + hf\left(t_n + \frac{1}{2}, \frac{1}{2}(\mathbf{y}_n + \mathbf{y}_{n+1})\right)$$

"take slope at middle point"



- Then, in exact Taylor expansion, the error is

$$\epsilon_{n+1} \equiv \mathbf{y}(t_n + h) - \mathbf{y}(t_n) - hf\left(t_n + \frac{h}{2}, \frac{1}{2}(\mathbf{y}(t_n) + \mathbf{y}(t_{n+1}))\right)$$
$$= \cdots = 0 + \boxed{\mathcal{O}(h^3)}$$

(exercise)

- This is called "implicit midpoint method"

Implicit, meaning: The right hand-side has already the solution at n+1, so one generally needs to resolve the equation for the n+1 value or use some iteration.

Runge-Kutta Methods: 2nd order - Midpoint methods

$$\dot{\mathbf{y}}(t) = f(t, \mathbf{y}(t)) \qquad \mathbf{y}(t_n) = \mathbf{y}_n \qquad \textit{Find:}\quad \mathbf{y}_{n+1} \approx \mathbf{y}(t_n + h)$$

$$\mathbf{y}_{n+1} \approx \mathbf{y}_n + hf\left(t_n + \frac{1}{2}, \frac{1}{2}(\mathbf{y}_n + \mathbf{y}_{n+1})\right)$$

"implicit midpoint method"

- Zero order iteration gives:

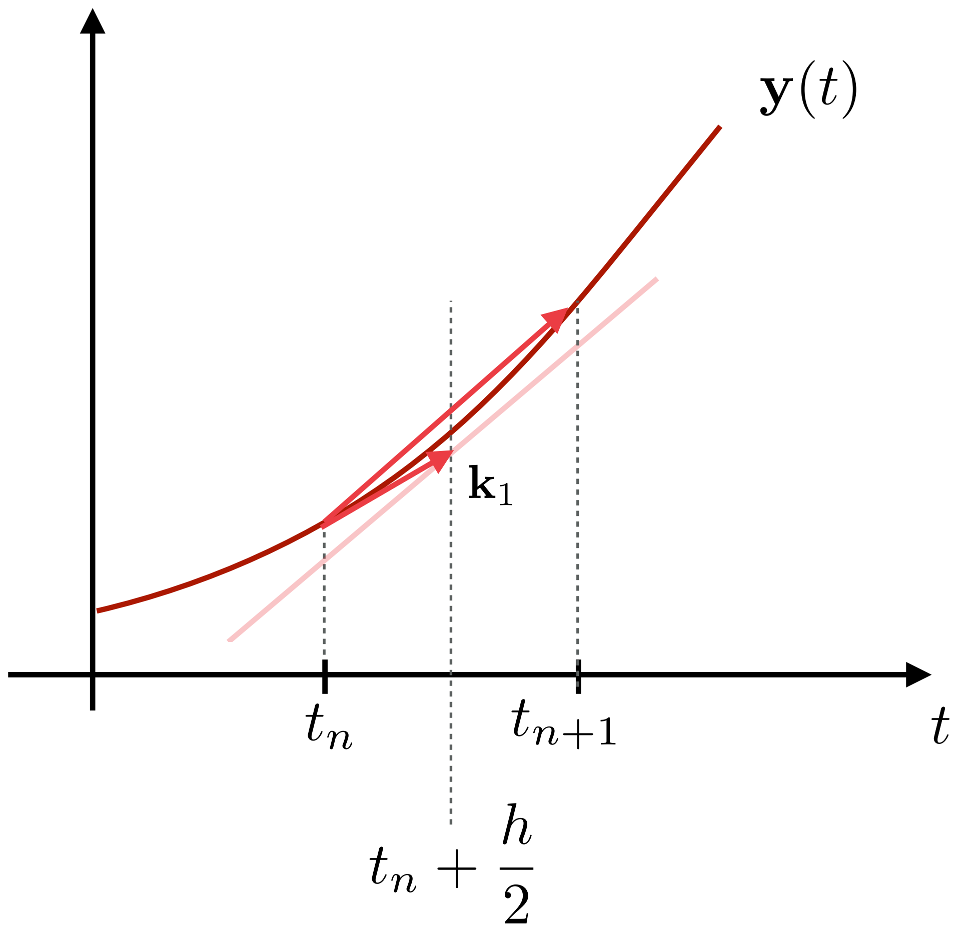
$$\mathbf{y}_{n+1} \approx \mathbf{y}_n + hf\left(t_n + \frac{1}{2}, \mathbf{y}_n\right) \qquad \mathbf{k}_1 \equiv \frac{\mathbf{y}_n + \mathbf{y}_{n+1}}{2} \approx \mathbf{y}_n + \frac{h}{2} f\left(t_n + \frac{1}{2}, \mathbf{y}_n\right)$$

Explicit Euler estimate for mid-point

$$\mathbf{k}_1 = \mathbf{y}_n + \frac{h}{2} f(t_n + \mathbf{y}_n)$$

$$\mathbf{y}_{n+1} \approx \mathbf{y}_n + hf(t_n + \frac{h}{2}, \mathbf{k}_1)$$

"explicit midpoint method"



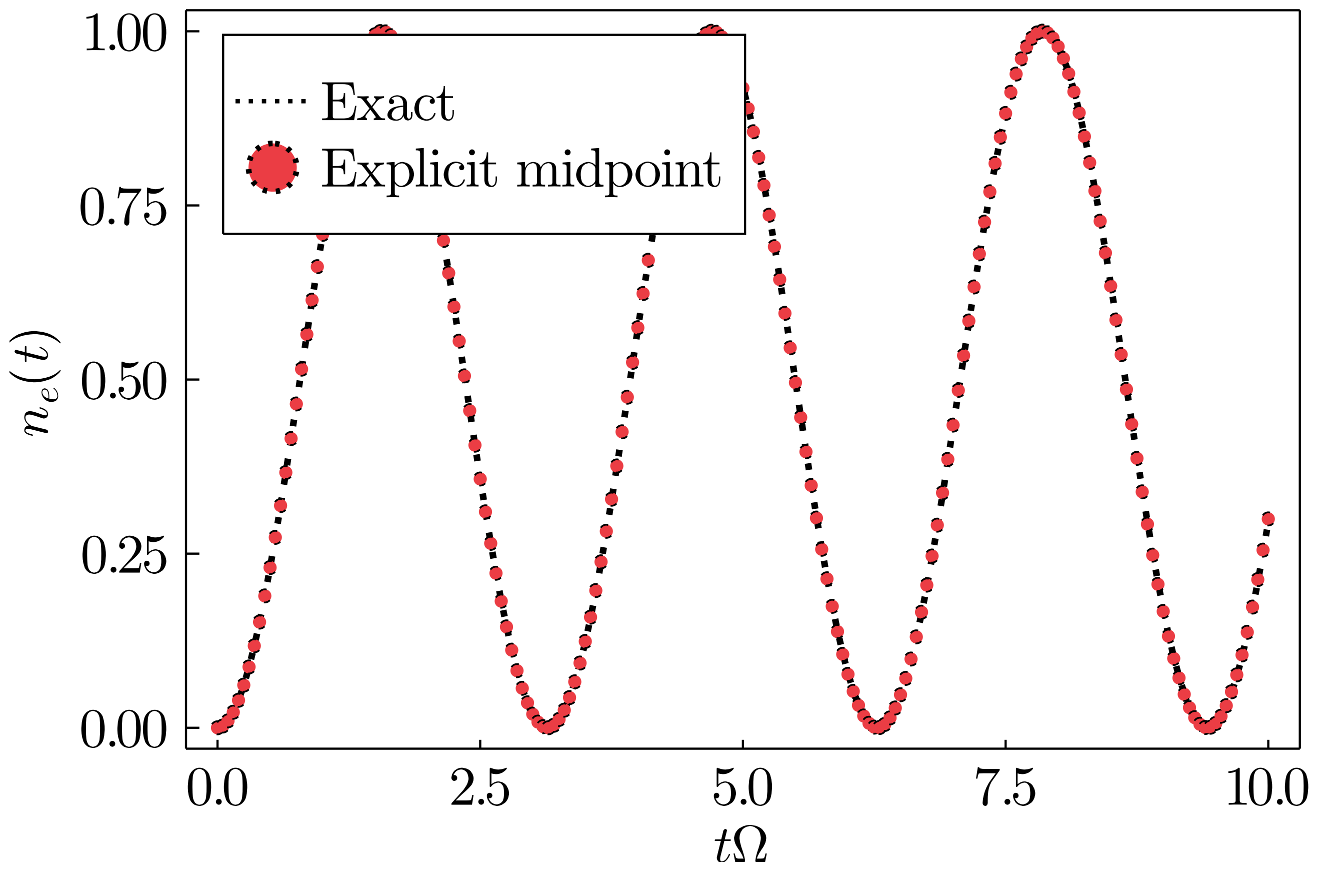
Runge-Kutta Methods: 2nd order - Midpoint methods

$$\dot{\mathbf{y}}(t) = f(t, \mathbf{y}(t)) \qquad \mathbf{y}(t_n) = \mathbf{y}_n \qquad \textit{Find:} \quad \mathbf{y}_{n+1} \approx \mathbf{y}(t_n + h)$$

- … very simple example (Rabi oscillations)

$$\hat{H} = \begin{pmatrix} 0 & \Omega \\ \Omega & 0 \end{pmatrix} \qquad \frac{d}{dt}|\psi\rangle = -\mathrm{i}\hat{H}|\psi\rangle \qquad |\psi_0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{matrix} |g\rangle \\ |e\rangle \end{matrix}$$

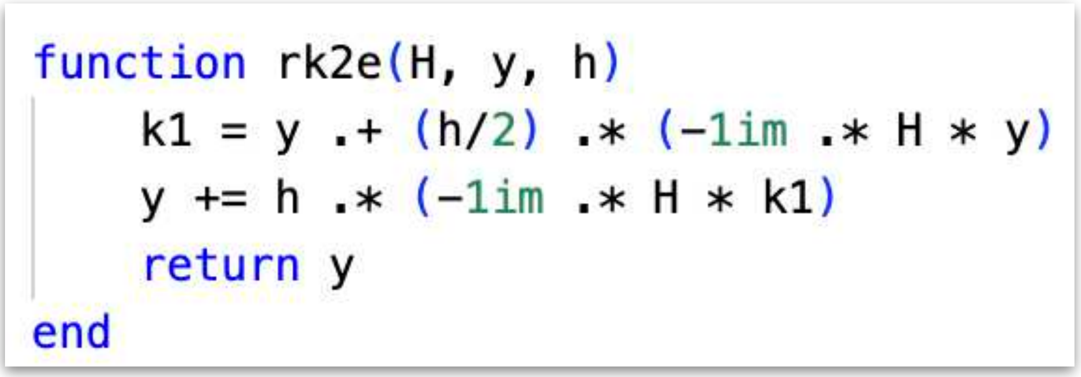
$$n_e(t) = |\langle\psi(t)|e\rangle|^2$$



$$\mathbf{k}_1 = \mathbf{y}_n + \frac{h}{2} f(t_n + \mathbf{y}_n)$$

$$\mathbf{y}_{n+1} \approx \mathbf{y}_n + h f(t_n + \frac{h}{2}, \mathbf{k}_1)$$

"explicit midpoint method"



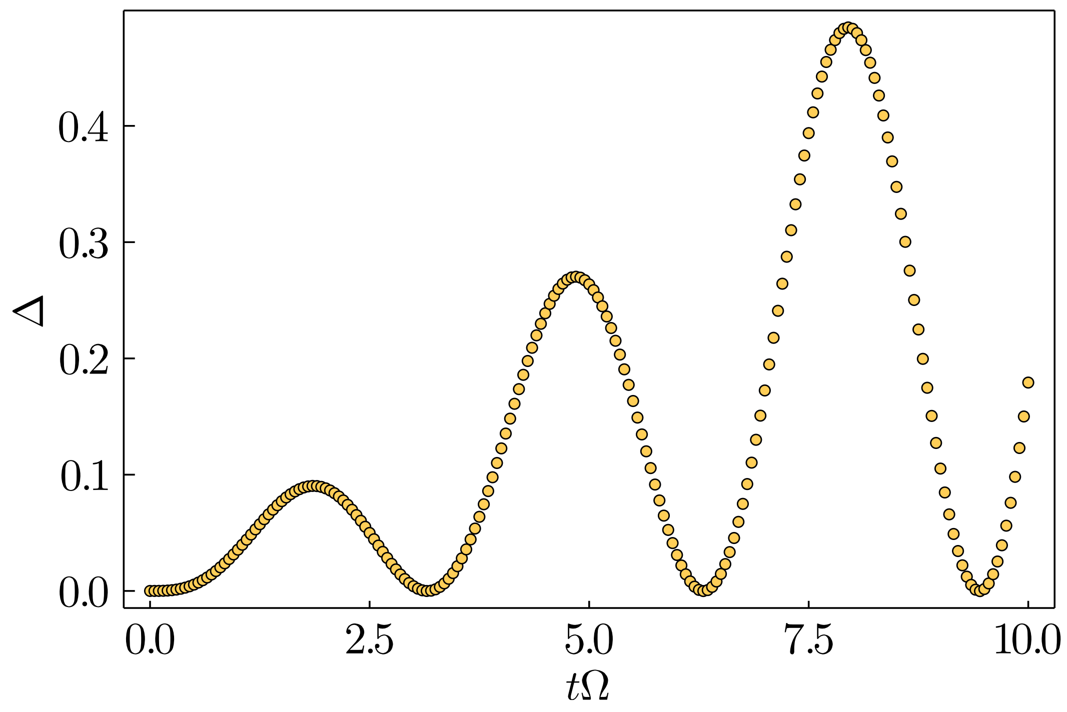
```
function rk2e(H, y, h)
    k1 = y .+ (h/2) .* (-1im .* H * y)
    y += h .* (-1im .* H * k1)
    return y
end
```

Runge-Kutta Methods: 2nd order

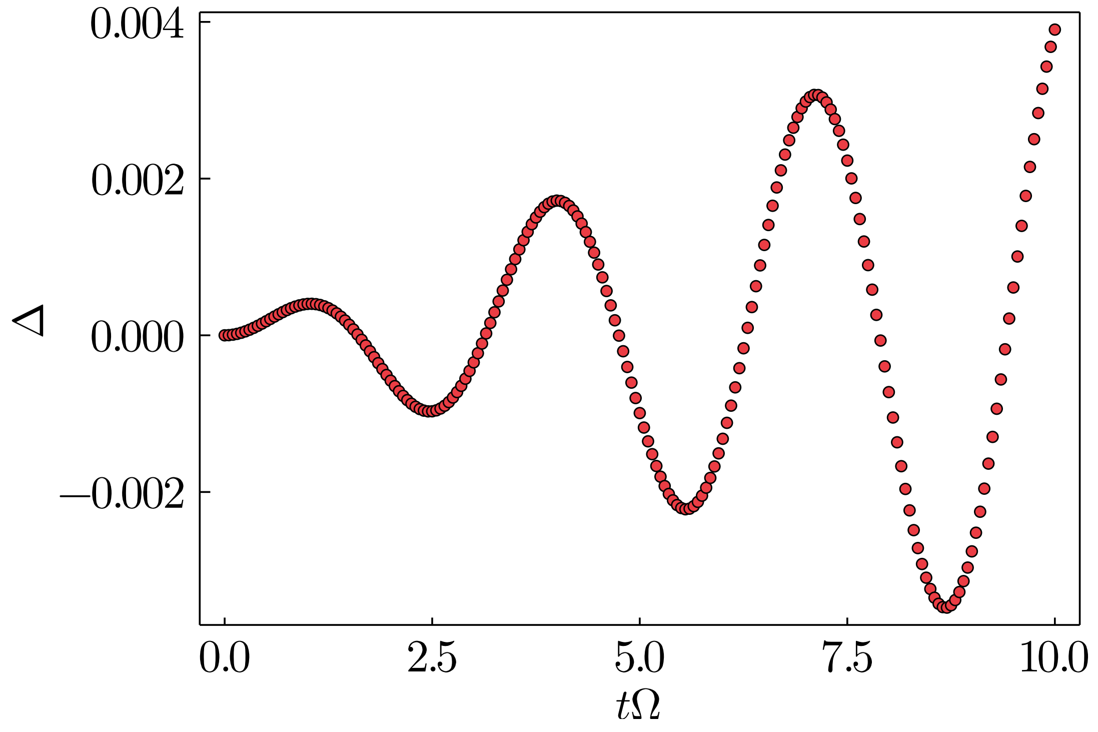
$$\dot{\mathbf{y}}(t) = f(t, \mathbf{y}(t)) \qquad \mathbf{y}(t_n) = \mathbf{y}_n \qquad \textit{Find:} \quad \mathbf{y}_{n+1} \approx \mathbf{y}(t_n + h)$$

- Comparisons $\Delta \equiv n_e(t) - \sin^2(t\Omega)$ $h\Omega = 0.05$

"explicit Euler"



"explicit midpoint"



Stable!

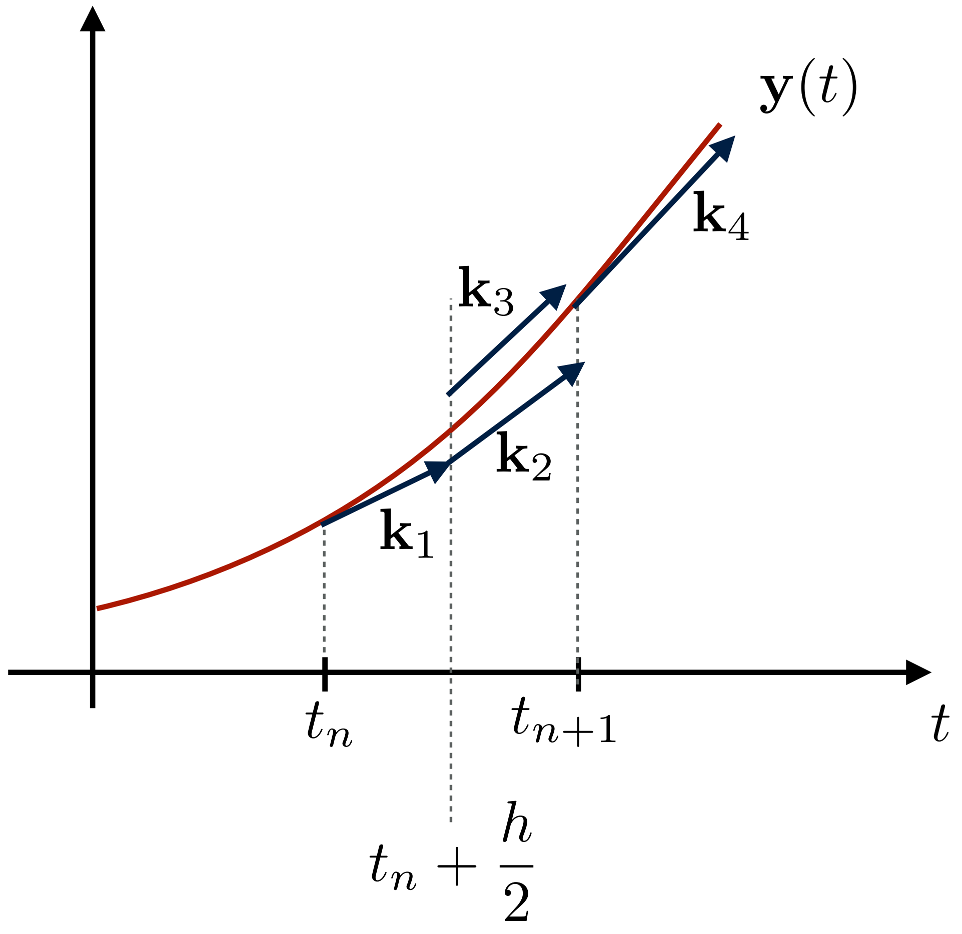
Runge-Kutta Methods: 4th order

$$\dot{\mathbf{y}}(t) = f(t, \mathbf{y}(t)) \qquad \mathbf{y}(t_n) = \mathbf{y}_n \qquad \textit{Find:} \quad \mathbf{y}_{n+1} \approx \mathbf{y}(t_n + h)$$

- In practice often the most convenient method

$$\mathbf{k}_1 = f(t_n, \mathbf{y}_n)$$
$$\mathbf{k}_2 = f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_1\right)$$
$$\mathbf{k}_3 = f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_2\right)$$
$$\mathbf{k}_4 = f\left(t_n + h, \mathbf{y}_n + h\mathbf{k}_3\right)$$
$$\mathbf{y}_{n+1} \approx \mathbf{y}_n + \frac{h}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)$$

"4th order Runge-Kutta"



- **Remarks:**

  - *Local error* $\epsilon_{n+1} = \mathcal{O}(h^5)$
  - *Note: n-th order = n function evaluations … higher order pays off!*
  - *In practice n=4 is convenient: e.g. 100 steps for plots, typical timescales ~10, time-step ~0.1 ideal*

Runge-Kutta Methods: 4th order

$$\dot{\mathbf{y}}(t) = f(t, \mathbf{y}(t)) \qquad \mathbf{y}(t_n) = \mathbf{y}_n \qquad \textit{Find:} \quad \mathbf{y}_{n+1} \approx \mathbf{y}(t_n + h)$$

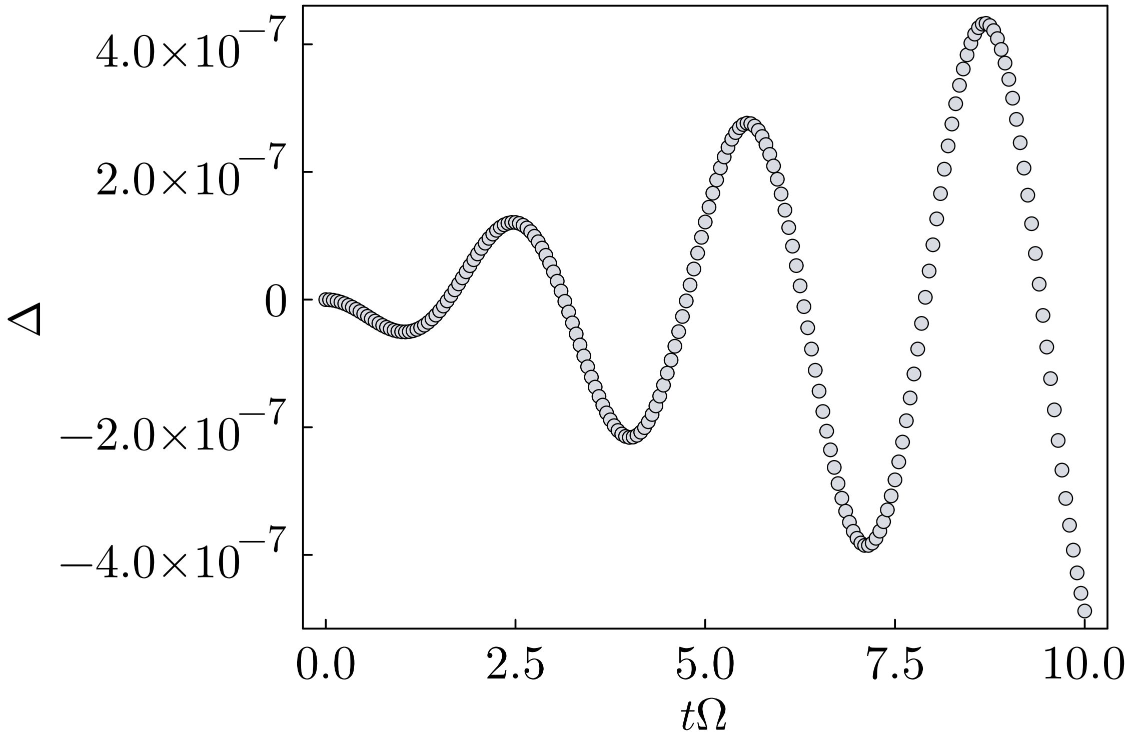
$$\mathbf{k}_1 = f(t_n, \mathbf{y}_n)$$
$$\mathbf{k}_2 = f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_1\right)$$
$$\mathbf{k}_3 = f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_2\right)$$
$$\mathbf{k}_4 = f\left(t_n + h, \mathbf{y}_n + h\mathbf{k}_3\right)$$
$$\mathbf{y}_{n+1} \approx \mathbf{y}_n + \frac{h}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)$$

"4th order Runge-Kutta"

```
function rk4(H, y, h)
    h2 = h/2
    imH = -1im .* H
    k1 = imH * y
    k2 = imH * (y .+ h2 .* k1)
    k3 = imH * (y .+ h2 .* k2)
    k4 = imH * (y .+ h .* k3)
    y += (h/6) .* (k1 .+ 2 .* k2 .+ 2 .* k3 .+ k4)
    return y
end
```

- … very simple example (Rabi oscillations)

$$\Delta \equiv n_e(t) - \sin^2(t\Omega)$$

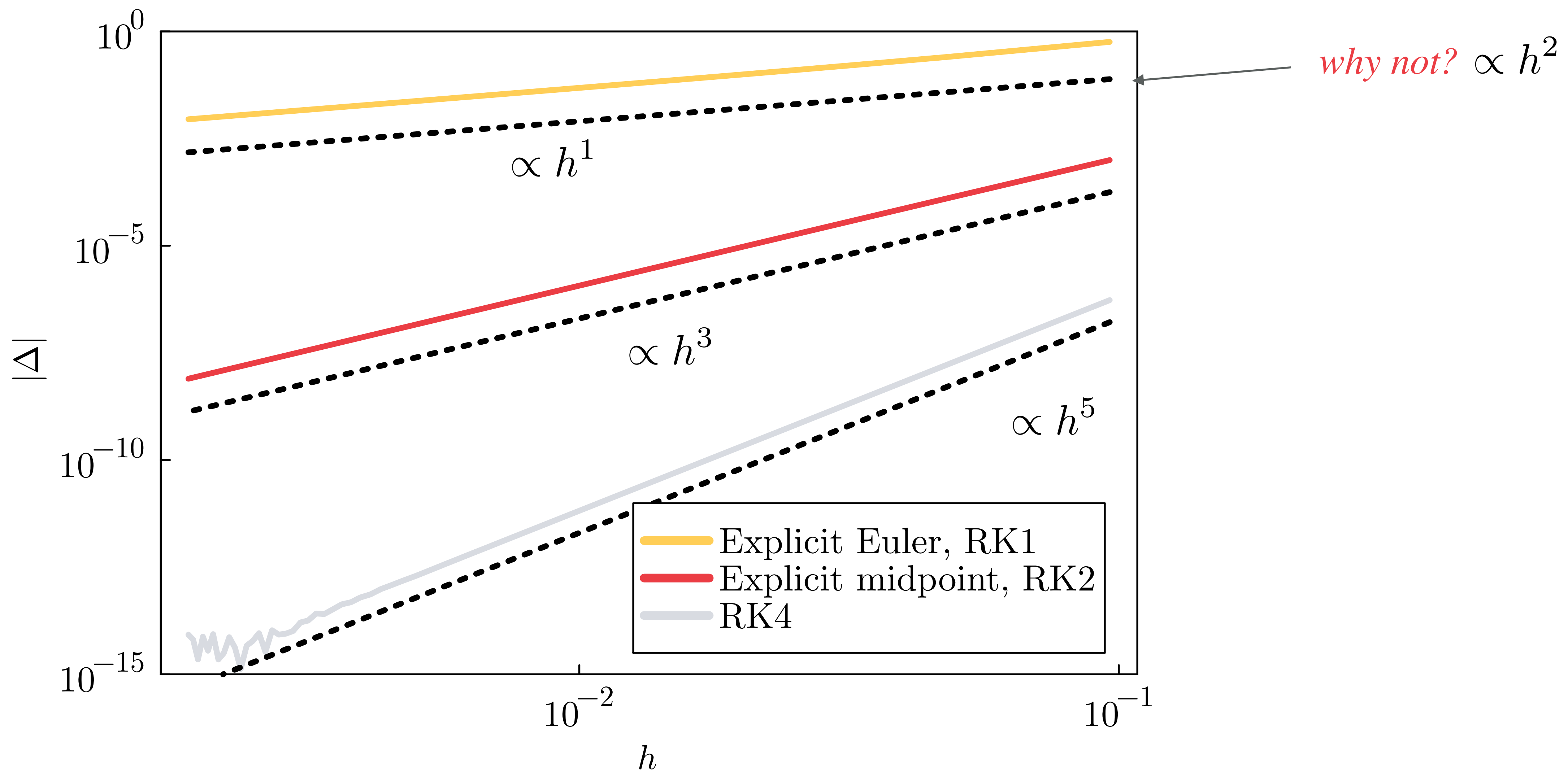


Runge-Kutta Methods: Sanity checks

- … very simple example (Rabi oscillations) $\hat{H} = \begin{pmatrix} 0 & \Omega \\ \Omega & 0 \end{pmatrix}$ $\frac{d}{dt}|\psi\rangle = -\mathrm{i}\hat{H}|\psi\rangle$ $|\psi_0\rangle = \begin{array}{cc} \begin{pmatrix} 1 \\ 0 \end{pmatrix} & \begin{array}{c} |g\rangle \\ |e\rangle \end{array} \end{array}$

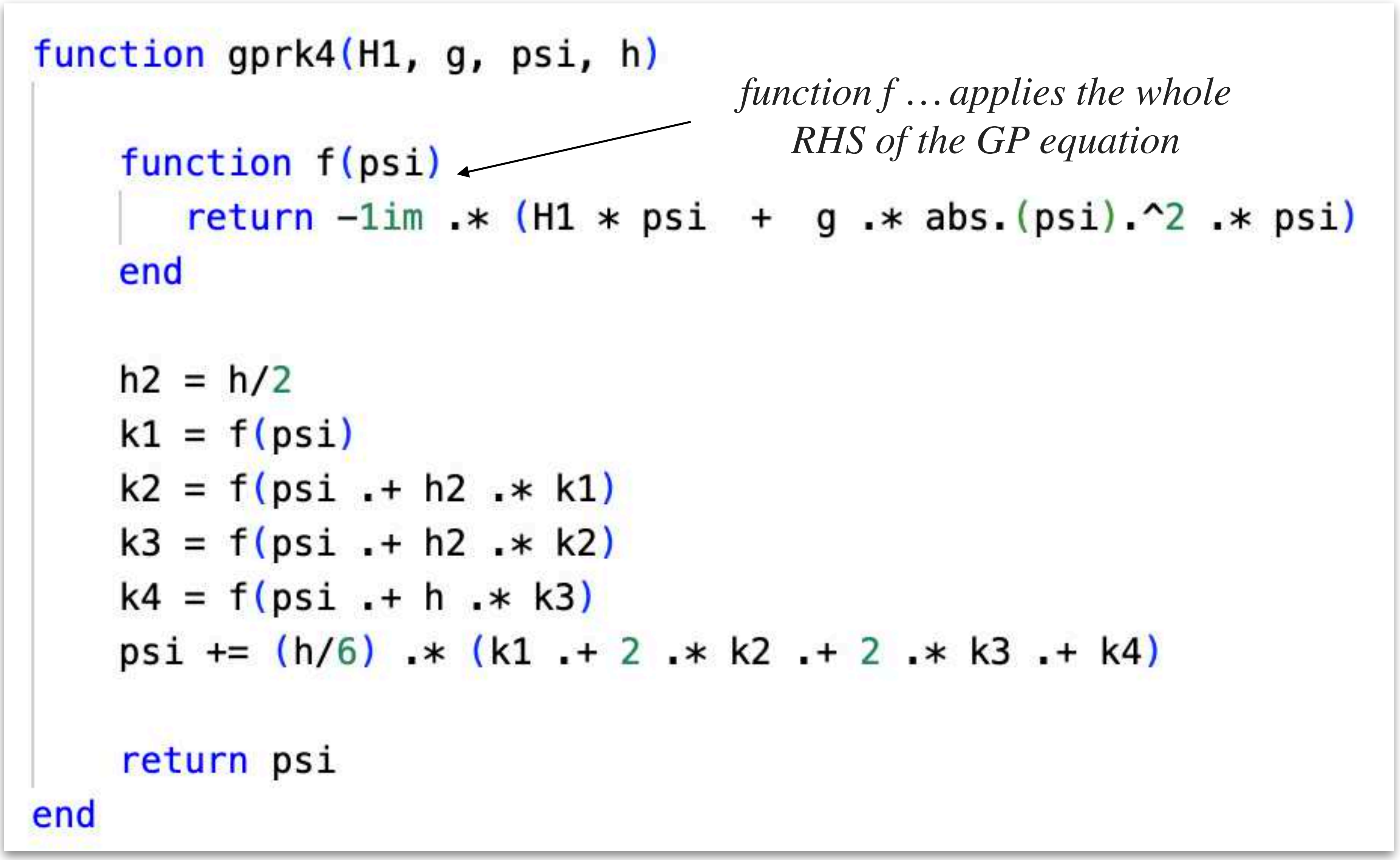
$$\Delta \equiv n_e(t) - \sin^2(t\Omega)$$

- Error at time fixed time, compare methods: $t\Omega = 3\frac{\pi}{2}$ $\sin^2(t\Omega) = 1$



Runge-Kutta Methods: 4th order applied to GP

$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\underbrace{\frac{\hat{p}^2}{2m} + V(x)}_{\hat{H}_1} + g|\psi(x,t)|^2\right)\psi(x,t)$$



```
function gprk4(H1, g, psi, h)

    function f(psi)
        return -1im .* (H1 * psi  +  g .* abs.(psi).^2 .* psi)
    end

    h2 = h/2
    k1 = f(psi)
    k2 = f(psi .+ h2 .* k1)
    k3 = f(psi .+ h2 .* k2)
    k4 = f(psi .+ h .* k3)
    psi += (h/6) .* (k1 .+ 2 .* k2 .+ 2 .* k3 .+ k4)

    return psi
end
```

function f … applies the whole RHS of the GP equation

Lecture 1 - Plan for today

- **Part 1.1:** Some fundamentals about numbers in digital memory and the linear algebra of quantum mechanics

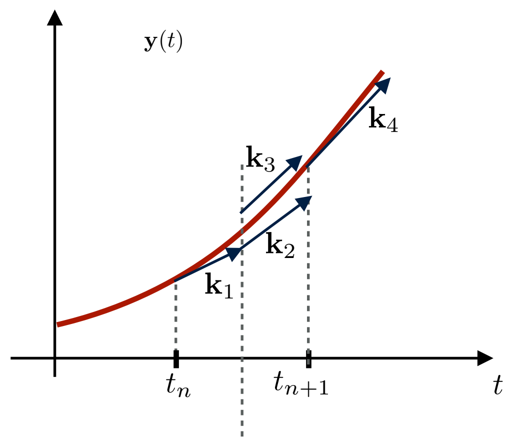
0100000000010110000000000000000000000000000000000000000000000000

- **Part 1.2:** The many-body problem of the day: Ultra-cold bosons in mean-field approximation (Gross-Pitaevskii, GP)

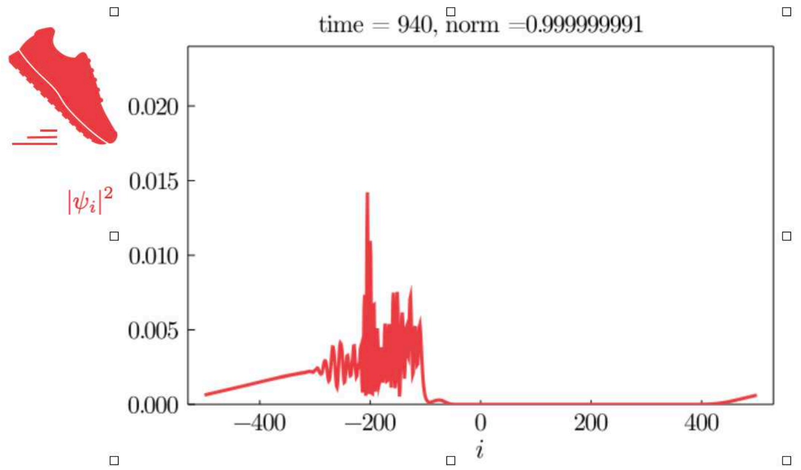
$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

- **Part 1.3:** Runge-Kutta (RK) time-evolution methods: A swiss army knife

$$\begin{aligned}
\mathbf{k}_1 &= f(t_n, \mathbf{y}_n)\\
\mathbf{k}_2 &= f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_1\right)\\
\mathbf{k}_3 &= f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_2\right)\\
\mathbf{k}_4 &= f\left(t_n + h, \mathbf{y}_n + h\mathbf{k}_3\right)\\
\mathbf{y}_{n+1} &\approx \mathbf{y}_n + \frac{h}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)
\end{aligned}$$



- **Part 1.4:** Applying Runge-Kutta to GP time-evolution



Lecture 1 - GP simulations with Runge-Kutta

$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

$$J = \frac{1}{2ma^2} \equiv 1$$

Solving it with RK4

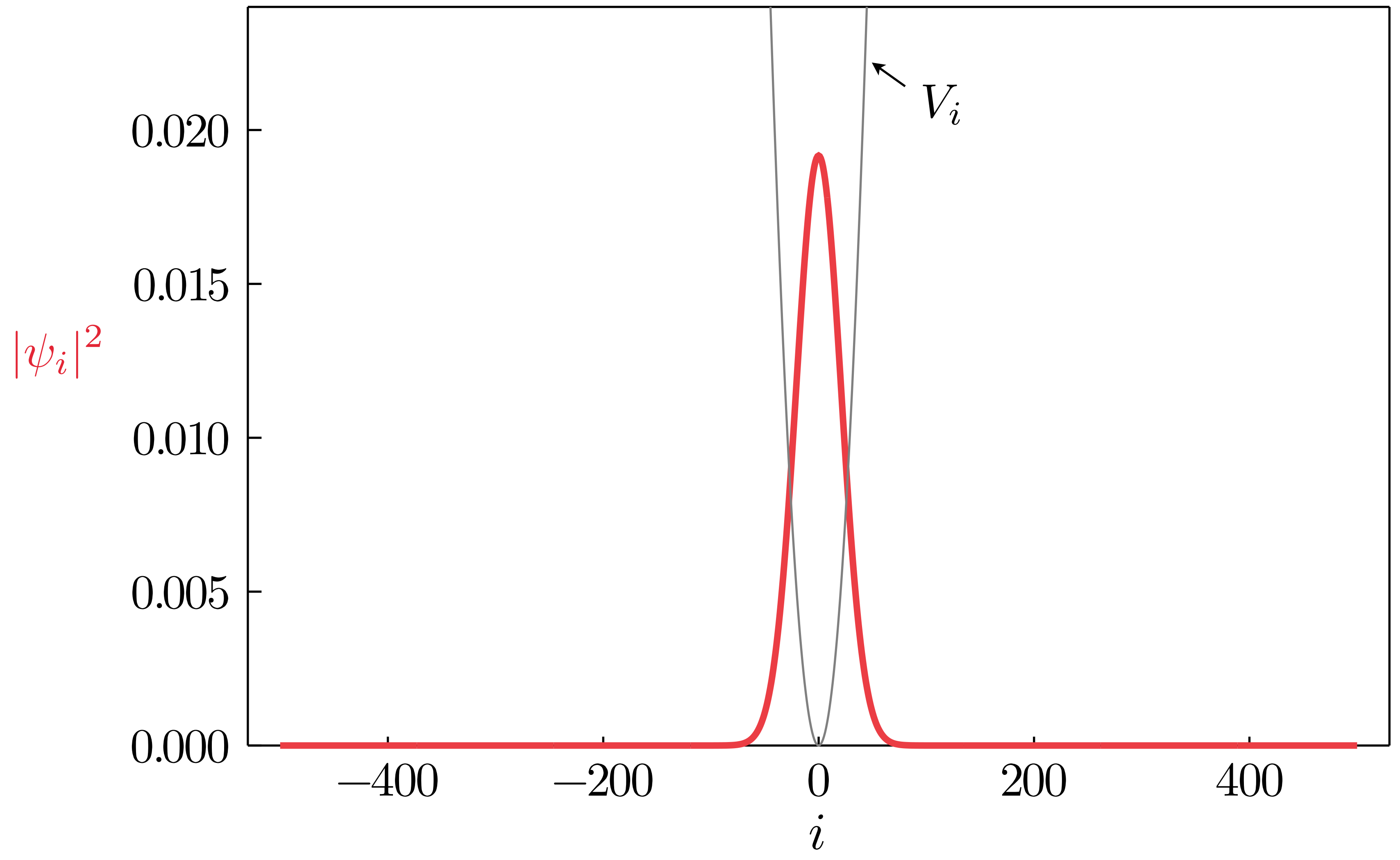
$hJ = 0.02$

1001 grid points

periodic boundaries

- Initial trap: $V_i = 1.2 \times 10^{-5} \times i^2$
- First we compute the ground state.

…we do this by evolving in imaginary time (will see how that works later)



$g = 0$

No interactions, we just prepare a standard Gaussian wave-packet (Standard QM for g=0)

Lecture 1 - GP simulations with Runge-Kutta

$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

$$J = \frac{1}{2ma^2} \equiv 1$$

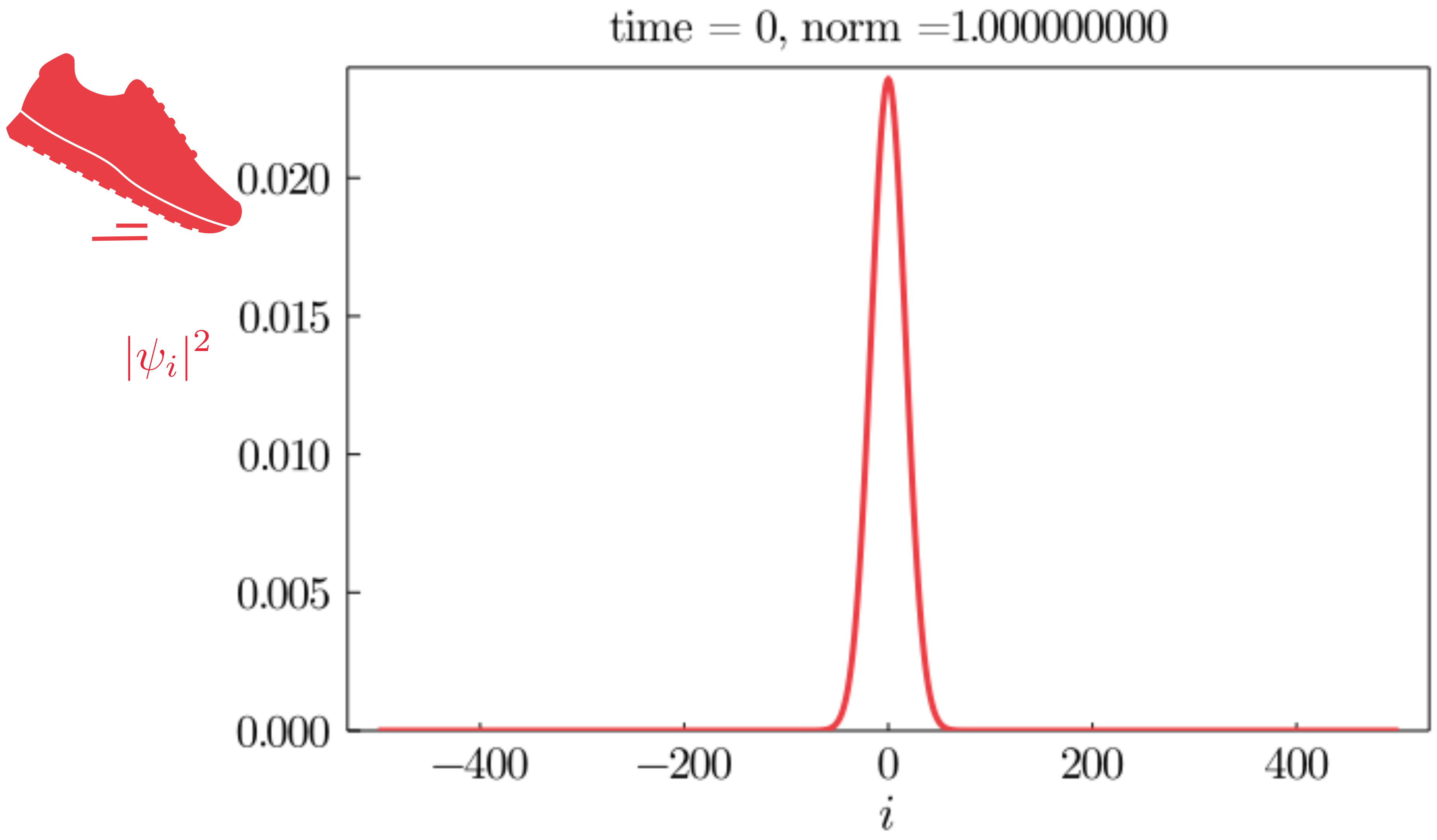
Solving it with RK4

$hJ = 0.02$

1001 grid points

periodic boundaries

- We remove the trap $V_i = 0$
- … and kick the system! This can be done by applying a phase-gradient $\psi_i \to \psi_i \mathrm{e}^{\mathrm{i}(ka)i}$



$g = 0$

$ka = 0.2\pi$

Standard evolution of quantum wave-packet with diffusion.

Lecture 1 - GP simulations with Runge-Kutta

$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

$$J = \frac{1}{2ma^2} \equiv 1$$

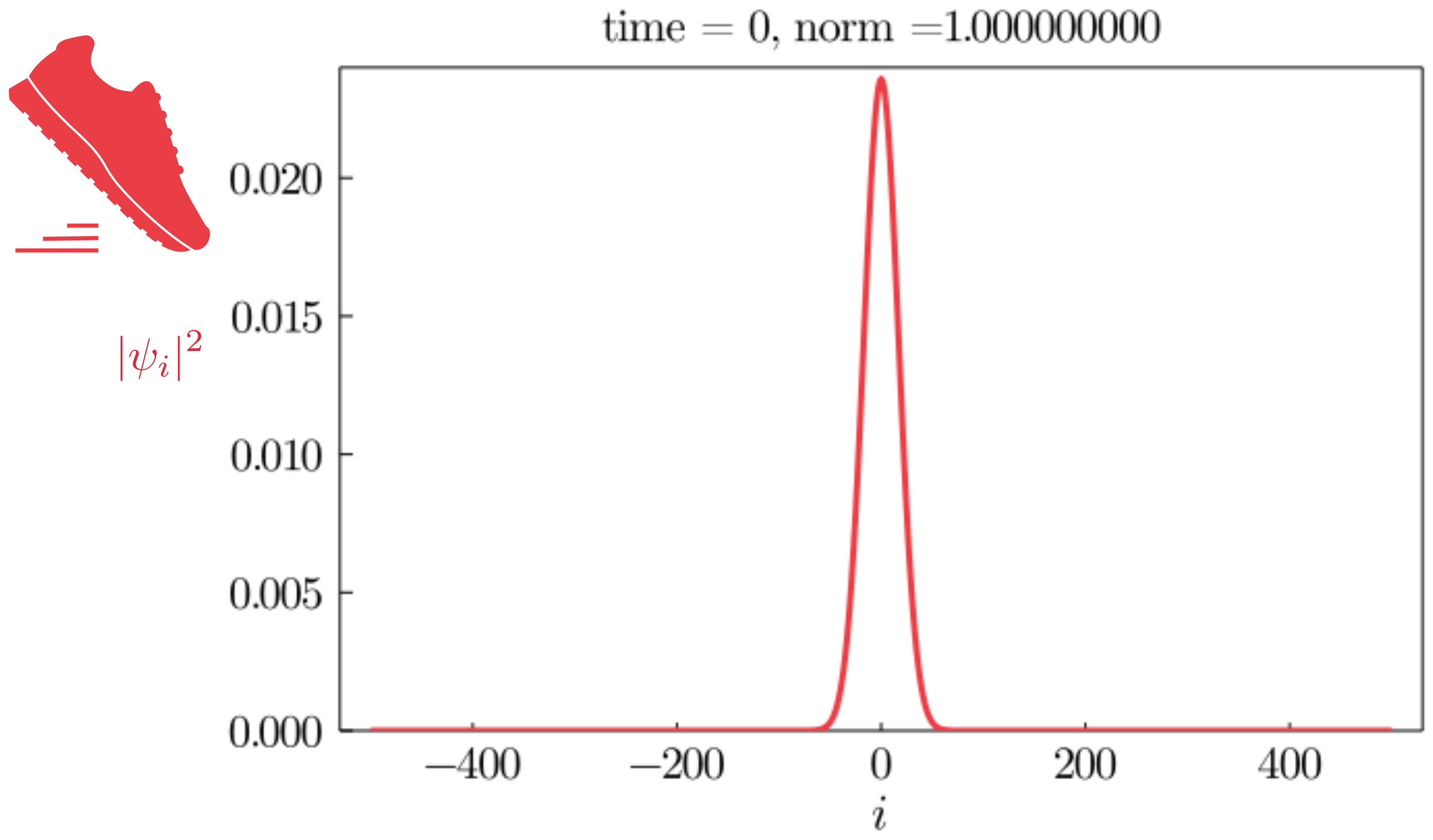
Solving it with RK4

$hJ = 0.02$

1001 grid points

periodic boundaries

- We kick it stronger
- … and kick the system! This can be done by applying a phase-gradient $\psi_i \to \psi_i e^{\mathrm{i}(ka)i}$



$g = 0$

$ka = 0.4\pi$

Stronger kick! Faster standard evolution of quantum wave-packet with diffusion.

Lecture 1 - GP simulations with Runge-Kutta

$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

$$J = \frac{1}{2ma^2} \equiv 1$$

Solving it with RK4

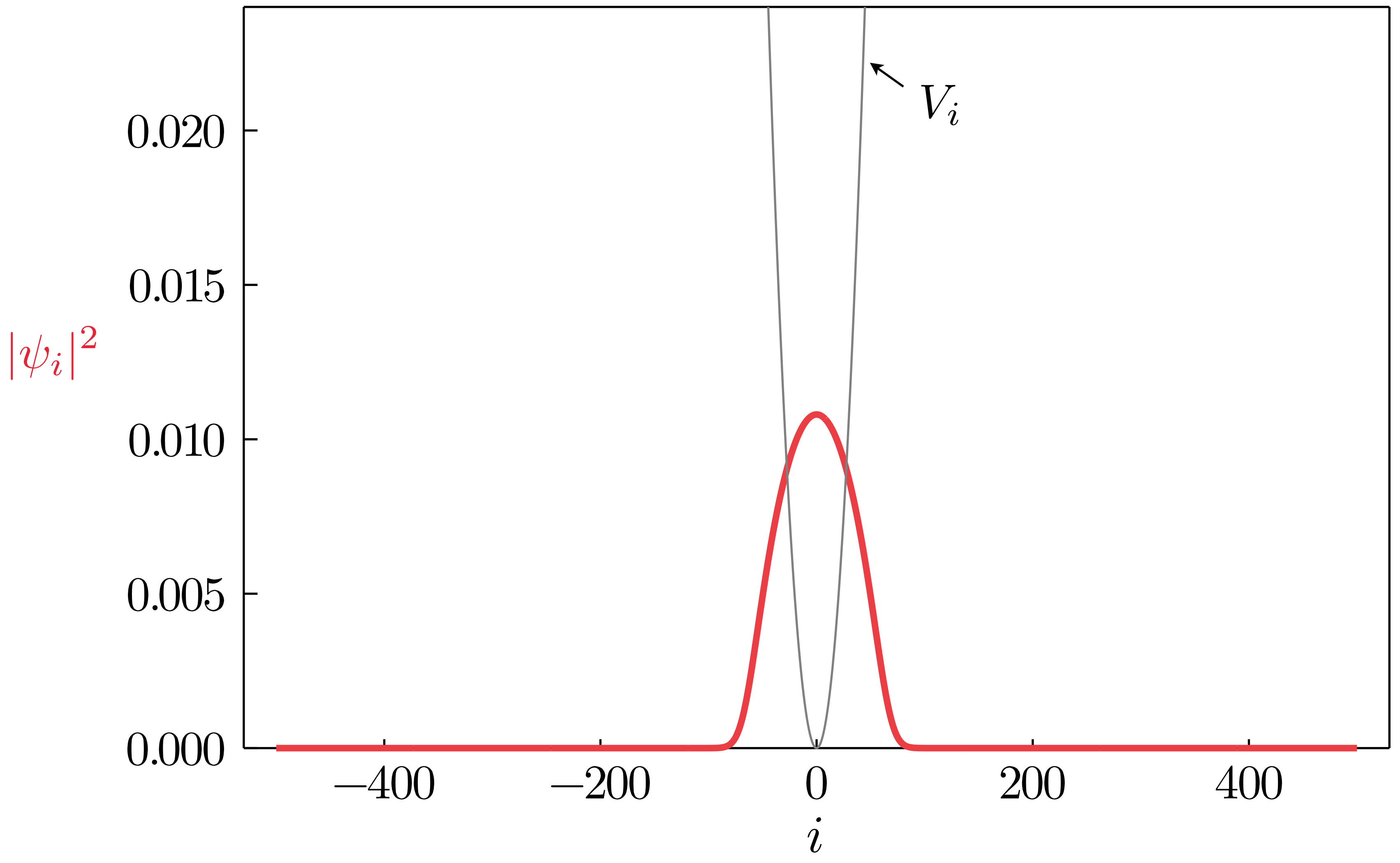
$hJ = 0.02$

1001 grid points

periodic boundaries

- Initial trap: $V_i = 1.2 \times 10^{-5} \times i^2$
- And first we compute the ground state.

…we do this by evolving in imaginary time (will see how that works later)



$g = 5J$

Interactions on!
Much broader classical field for the BEC

Potential takes form of inverted trap -> Result in Thomas-Fermi approximation!

Lecture 1 - GP simulations with Runge-Kutta

$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

$$J = \frac{1}{2ma^2} \equiv 1$$

Solving it with RK4

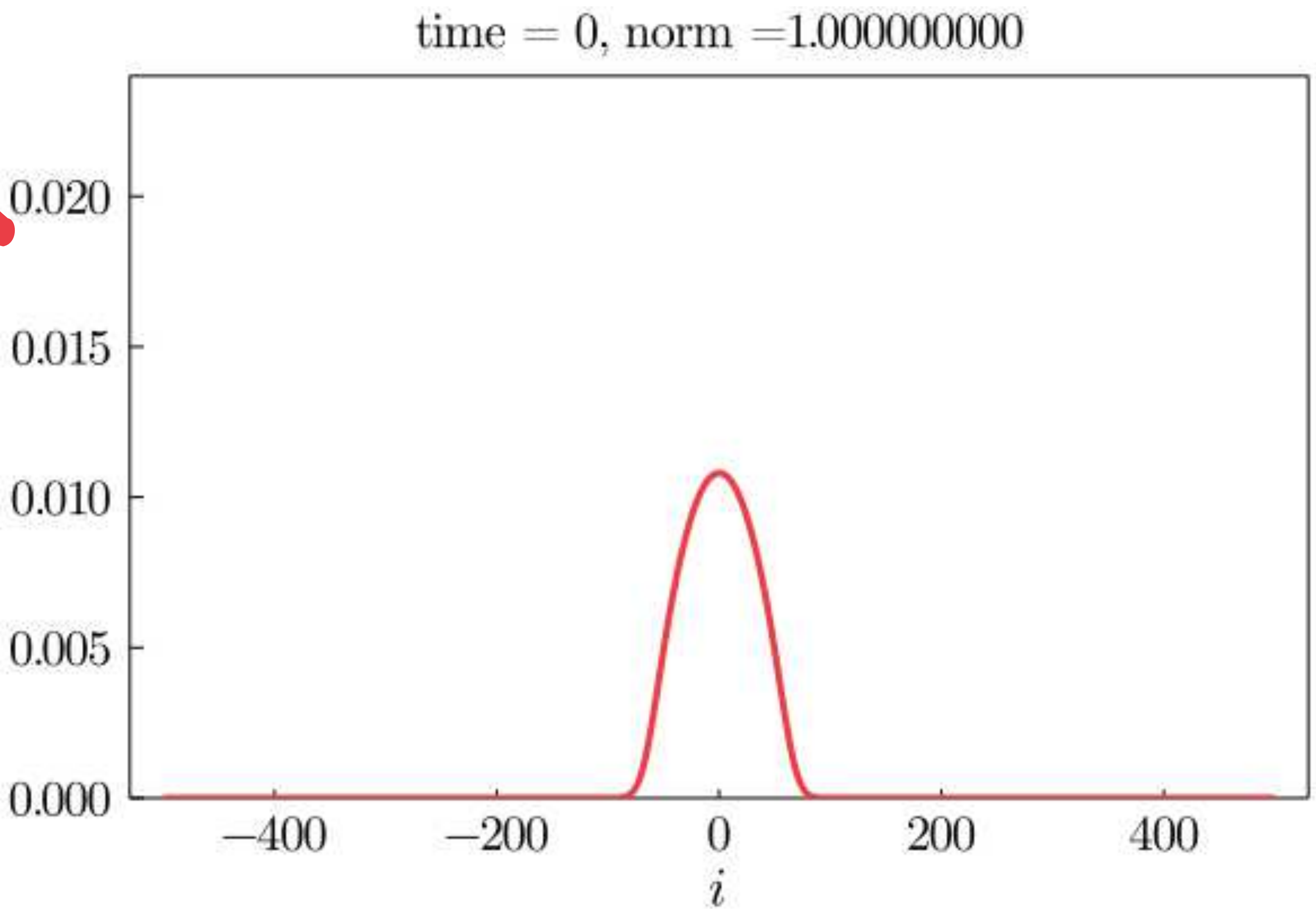
$hJ = 0.02$

1001 grid points

periodic boundaries

- We remove the trap $V_i = 0$
- … and kick the system! This can be done by applying a phase-gradient $\psi_i \to \psi_i e^{\mathrm{i}(ka)i}$

$|\psi_i|^2$



$g = 5J$

$ka = 0.2\pi$

Similar evolution as non-interacting case. Just diffusion. But condensate field keeps the Thomas-Fermi shape (BEC is stable)

Lecture 1 - GP simulations with Runge-Kutta

$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

$$J = \frac{1}{2ma^2} \equiv 1$$

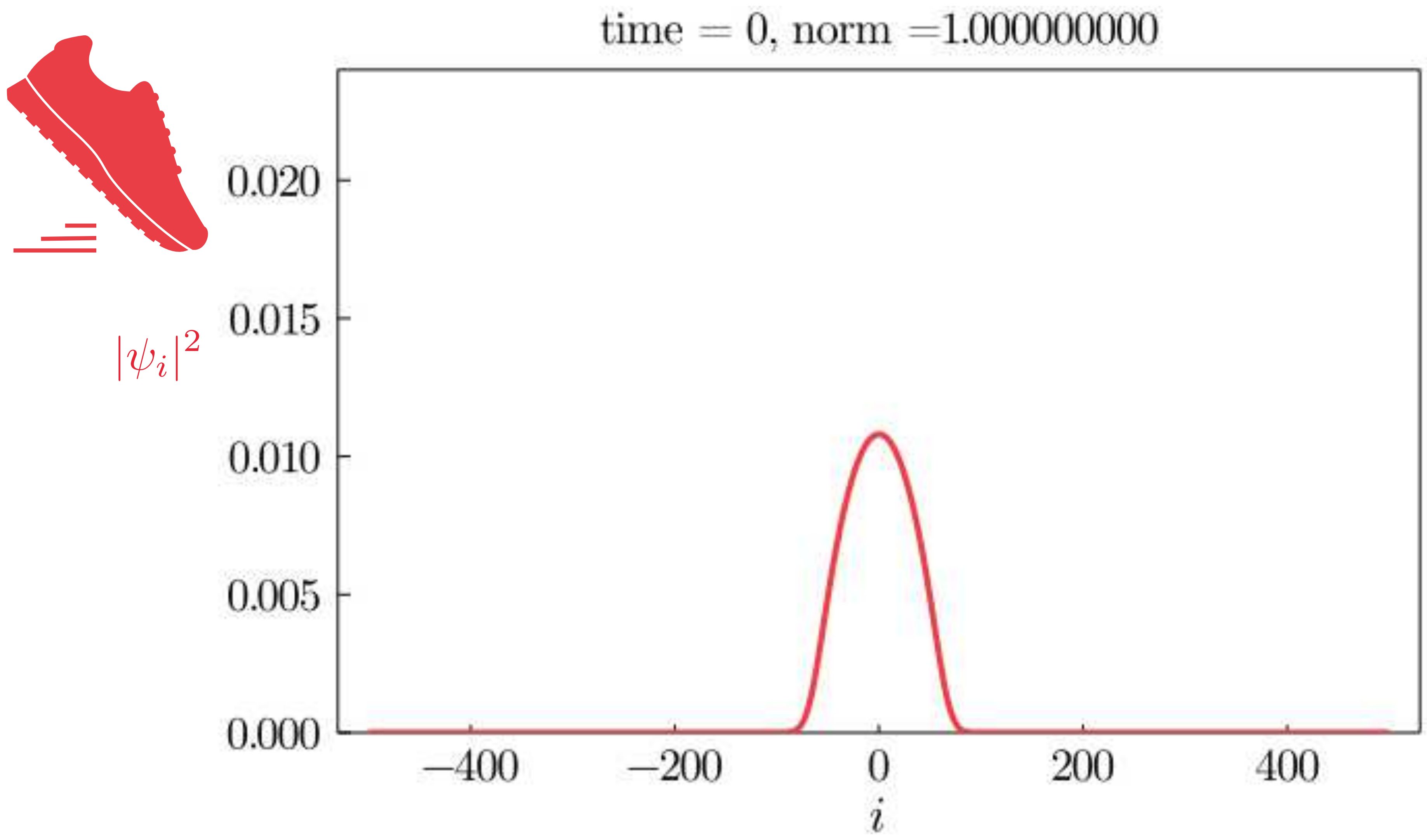
Solving it with RK4

$hJ = 0.02$

1001 grid points

periodic boundaries

- We kick it stronger
- … and kick the system! This can be done by applying a phase-gradient $\psi_i \to \psi_i e^{\mathrm{i}(ka)i}$



$g = 5J$

$ka = 0.4\pi$

Our BEC get's destroyed!

This is known as dynamical instability!

Lecture 1 - GP simulations with Runge-Kutta

$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

$$J = \frac{1}{2ma^2} \equiv 1$$

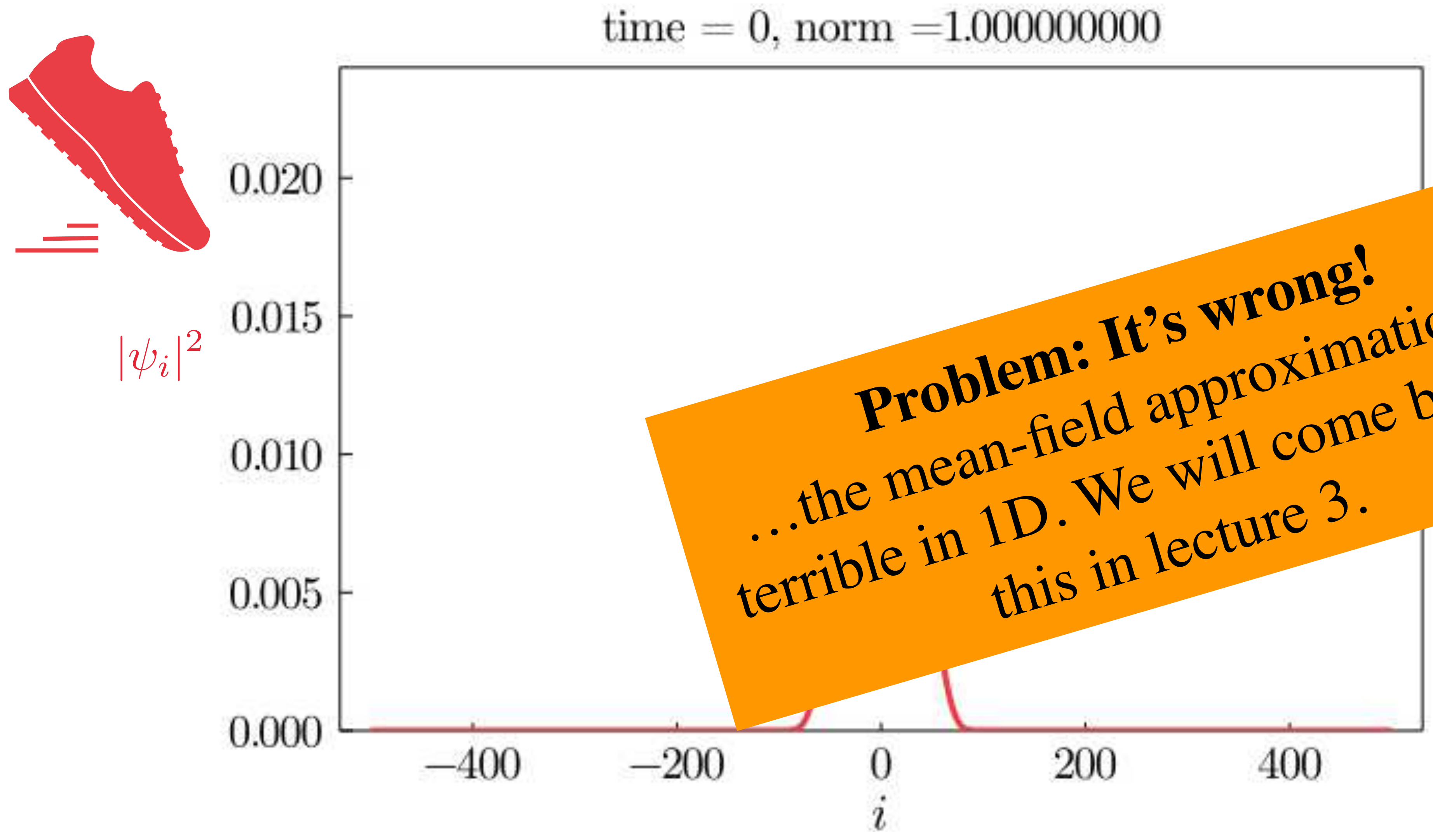
Solving it with RK4

$hJ = 0.02$

1001 grid points

periodic boundaries

- We kick it stronger
- … and kick the system! This can be done by applying a phase-gradient $\psi_i \to \psi_i \mathrm{e}^{\mathrm{i}(ka)i}$



Problem: It's wrong!

…the mean-field approximation is terrible in 1D. We will come back to this in lecture 3.

$g = 5J$

$ka = 0.4\pi$

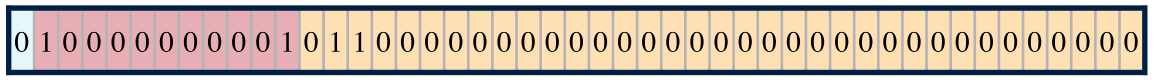
… BEC get's destroyed!

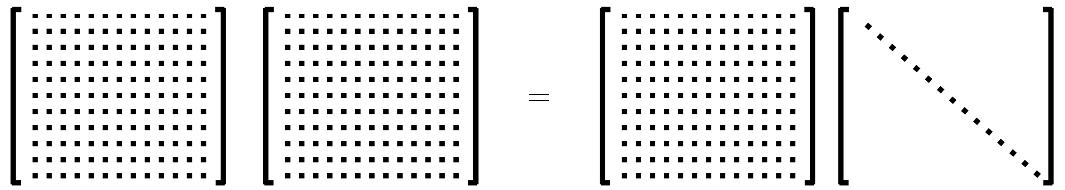
This is known as dynamical instability!

See e.g.
New J. Phys. 12, 025014 (2010)

Lecture 1 - Recap

- We discussed how to represent numbers as bits. We describe systems with state-vectors (not only in QM). If the problem is linear (QM), an exact diagonalization of the Hamiltonian solves everything, and is easily doable numerically for small systems.



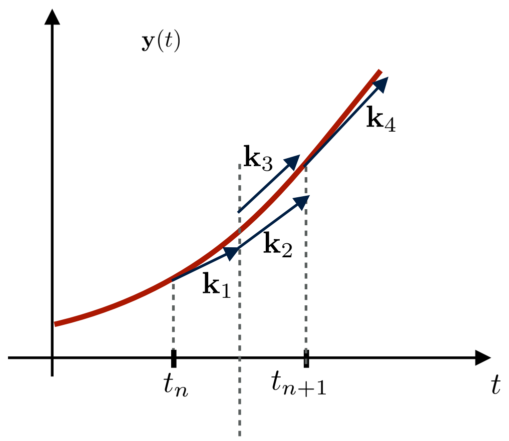


- Not all problems are linear. Also in quantum mechanics, when making a mean-field approximation (e.g. Gross-Pitaevskii equations for ultra-cold bosonic gases) the problem becomes non-linear. However the state-space drastically decreases in this case (from exponential to linear growth with system size).

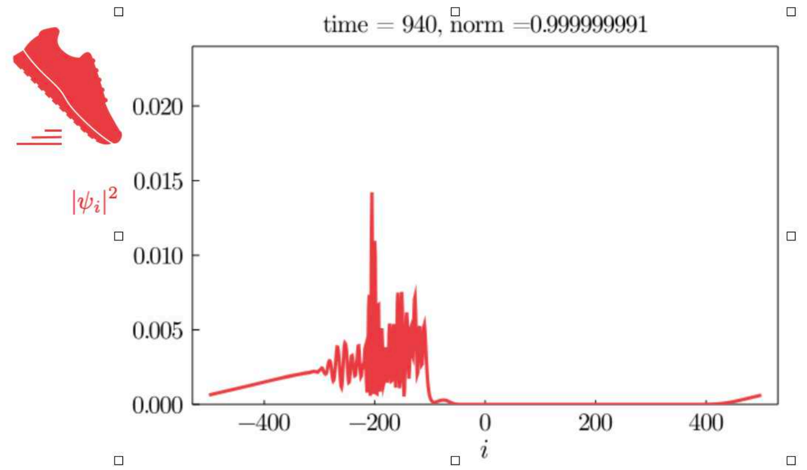
$$\frac{d}{dt}\psi(x,t) = -\mathrm{i}\left(\frac{\hat{p}^2}{2m} + V(x) + g|\psi(x,t)|^2\right)\psi(x,t)$$

- Runge-Kutta methods are a general tool to simulate dynamics of linear and non-linear problems. Formally derived from Taylor expansions using multiple steps. In particular the 4-th order method is a good compromise (stable, small error for reasonable time-step).

$$\mathbf{k}_1 = f(t_n, \mathbf{y}_n)$$
$$\mathbf{k}_2 = f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_1\right)$$
$$\mathbf{k}_3 = f\left(t_n + \frac{h}{2}, \mathbf{y}_n + \frac{h}{2}\mathbf{k}_2\right)$$
$$\mathbf{k}_4 = f\left(t_n + h, \mathbf{y}_n + h\mathbf{k}_3\right)$$
$$\mathbf{y}_{n+1} \approx \mathbf{y}_n + \frac{h}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)$$



- We looked at a kicked bosonic condensate as example application, using the GP approximation, and re-produced the effect of an dynamics instability with a few lines of codes. However, this physics is actually wrong in 1D as we will see later. Mean-field approximations can drastically increase the treatable system size but are also strong approximations neglecting any entanglement.


time = 940, norm =0.999999991
$|\psi_i|^2$
0.020
0.015
0.010
0.005
0.000
−400
−200
0
200
400
i