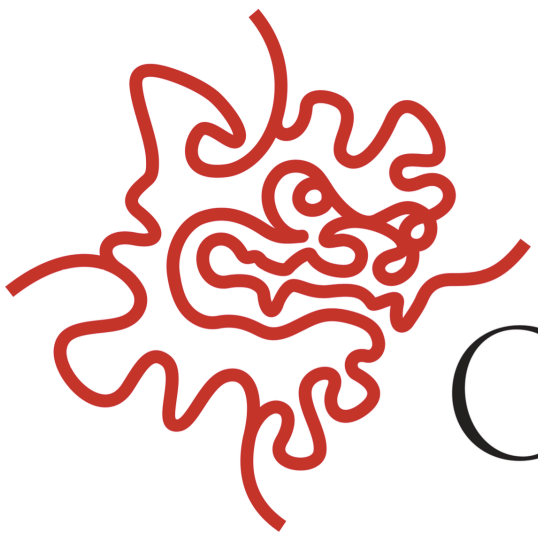
# Massively parallel split-step Fourier techniques for simulating quantum systems on graphics processing units

James Schloss

Advisor: **Thomas Busch**
Quantum Systems Unit

December 9, 2019



OIST

# Overview

**Physics: understanding superfluid vortices**

Bose–Einstein Condensate

↓

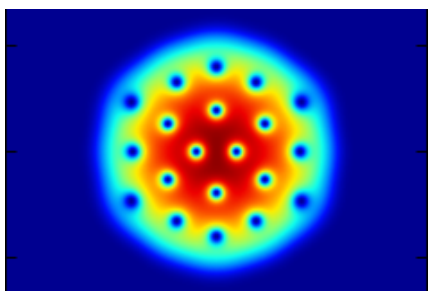
Vortex generation

↓

2D vortex chaos

↓

3D vortex structures



# Overview

**Physics: understanding superfluid vortices**

Bose–Einstein Condensate

↓

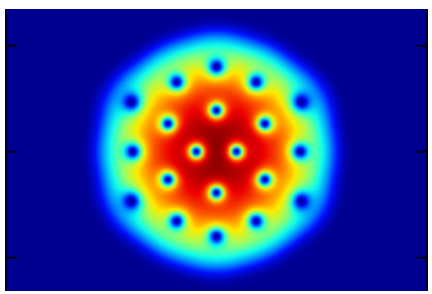
Vortex generation

↓

2D vortex chaos

↓

3D vortex structures



**Computer Science: SSFM on GPUs**

Split-step Fourier method

↓

GPU hardware

↓

GPUE codebase

↓

Optimizations

$$\mathbf{GP}\begin{pmatrix}\hat{U}\\ E\end{pmatrix}$$

# What we will miss...

**Physics: understanding superfluid vortices**

Quantum state engineering

(*NJP* 18 (3):035012, 2016)

**J Schloss**, A Benseny, J Gillet, J Swain, T Busch

**Computer Science: SSFM on GPUs**

Implementation details

(Tests, formats, etc.)

# The Fourier Transform

Fourier Transform:

$$F(\xi) = \int_{-\infty}^{\infty} f(t) e^{-2\pi i t \xi} dt$$

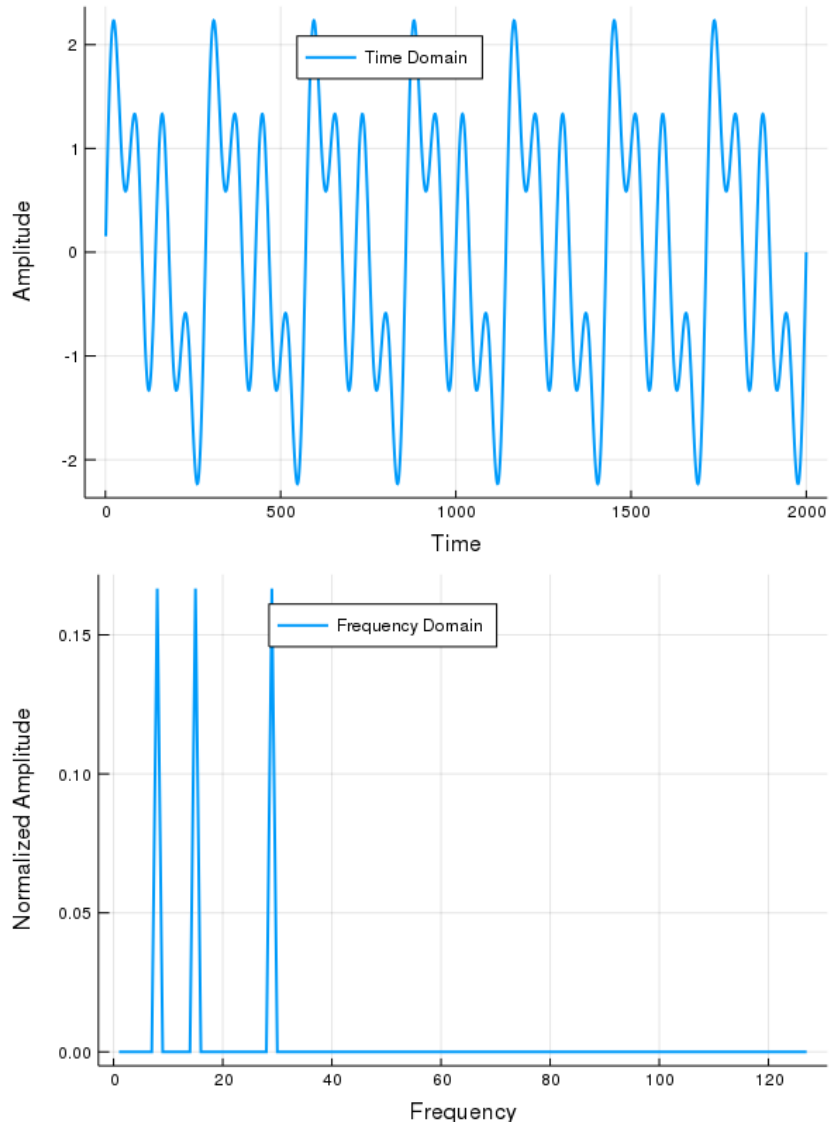
$$f(t) = \int_{-\infty}^{\infty} F(\xi) e^{2\pi i t \xi} d\xi$$

# The Fourier Transform

Fourier Transform:

$$F(\xi) = \int_{-\infty}^{\infty} f(t) e^{-2\pi i t \xi} dt$$

$$f(t) = \int_{-\infty}^{\infty} F(\xi) e^{2\pi i t \xi} d\xi$$



# The Fourier Transform

Fourier Transform:

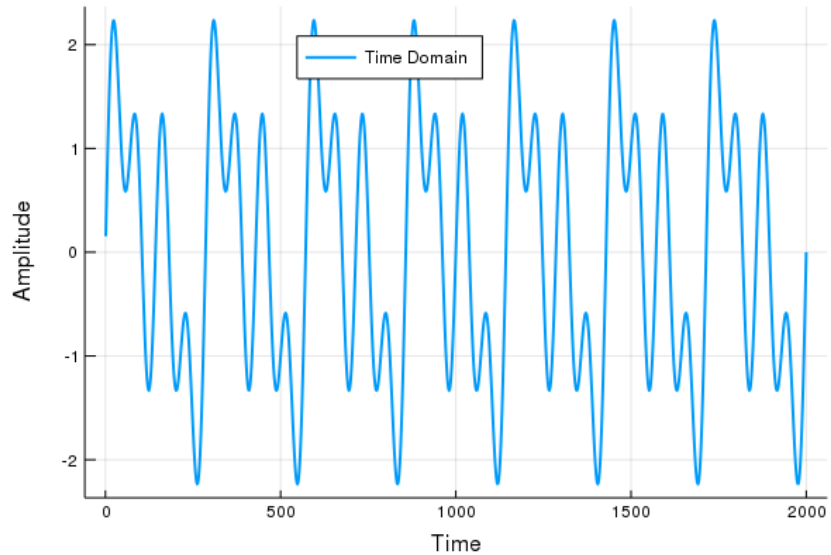
$$F(\xi) = \int_{-\infty}^{\infty} f(t) e^{-2\pi i t \xi} dt$$

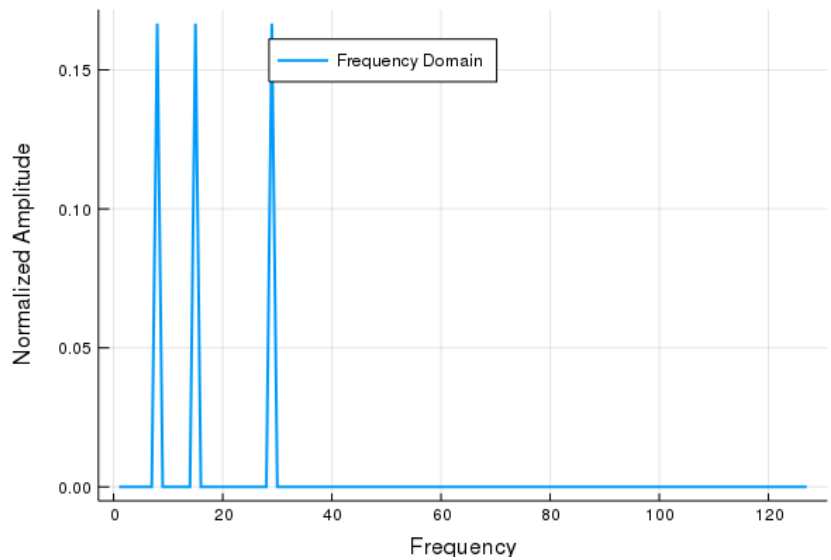
$$f(t) = \int_{-\infty}^{\infty} F(\xi) e^{2\pi i t \xi} d\xi$$

Discrete Fourier Transform:

$$X_k = \sum_{n=0}^{N-1} x_n e^{2\pi i k n / N}$$

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{-2\pi i k n / N}$$



# The Fourier Transform

Fourier Transform:

$$F(\xi) = \int_{-\infty}^{\infty} f(t) e^{-2\pi i t \xi} dt$$

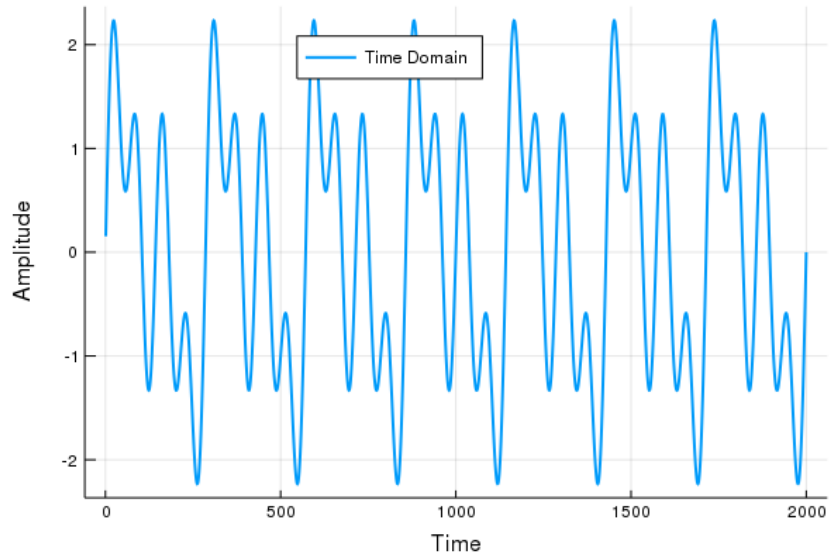
$$f(t) = \int_{-\infty}^{\infty} F(\xi) e^{2\pi i t \xi} d\xi$$

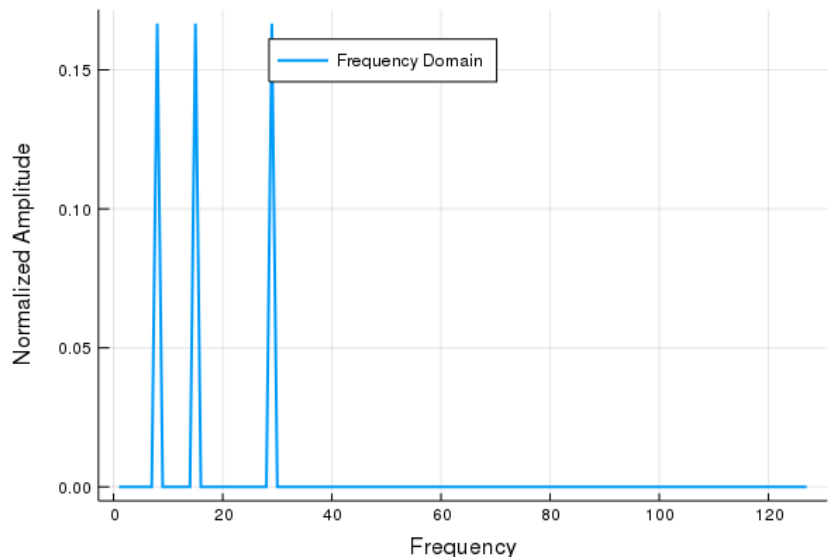
Discrete Fourier Transform:

$$X_k = \sum_{n=0}^{N-1} x_n e^{2\pi i k n / N}$$

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{-2\pi i k n / N}$$

The FFT is a *global* operation requiring iteration or recursion



# Hamiltonian

1D Schrödinger equation:

$$i\hbar\frac{\partial\Psi(x,t)}{\partial t} = \left(\frac{p^2}{2m} + \frac{1}{2}m\omega^2x^2\right)\Psi(x,t)$$

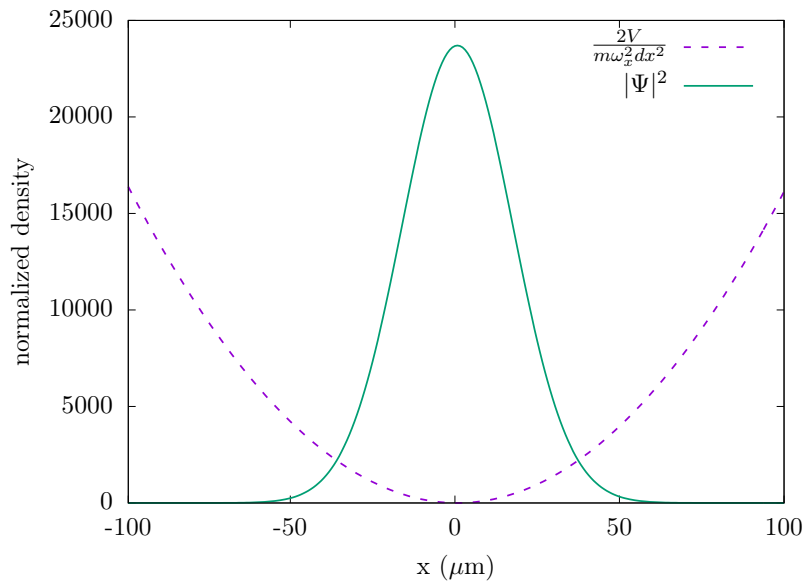
# Hamiltonian

1D Schrödinger equation:

$$i\hbar\frac{\partial\Psi(x,t)}{\partial t} = \left(\frac{p^2}{2m} + \frac{1}{2}m\omega^2x^2\right)\Psi(x,t) = \mathcal{H}\Psi(x,t)$$

Splits into:

$$\mathcal{H}_v = \frac{1}{2}m\omega^2x^2 \qquad \mathcal{H}_p = \frac{p^2}{2m} = \frac{1}{2m}\left(i\hbar\frac{\partial}{\partial x}\right)^2$$



# Split-Step Fourier Method (SSFM)

Differential Equations:

$$\Psi(x, t+dt) = \left[e^{-\frac{i\mathcal{H}dt}{\hbar}}\right]\Psi(x,t) = \left[e^{-\frac{i(\mathcal{H}_V+\mathcal{H}_P)dt}{\hbar}}\right]\Psi(x,t)$$

# Split-Step Fourier Method (SSFM)

Differential Equations:

$$\Psi(x, t+dt) = \left[e^{-\frac{i\mathcal{H}dt}{\hbar}}\right]\Psi(x,t) = \left[e^{-\frac{i(\mathcal{H}_V+\mathcal{H}_P)dt}{\hbar}}\right]\Psi(x,t)$$

Baker-Campbell-Hausdorff:

$$\Psi(x, t+dt) = \left[e^{-\frac{i\mathcal{H}_V dt}{\hbar}} e^{-\frac{i\mathcal{H}_P dt}{\hbar}} e^{-\frac{[i\mathcal{H}_V, i\mathcal{H}_P]dt^2}{2}}\right]\Psi(x,t)$$

# Split-Step Fourier Method (SSFM)

Differential Equations:

$$\Psi(x, t+dt) = \left[e^{-\frac{i\mathcal{H}dt}{\hbar}}\right]\Psi(x,t) = \left[e^{-\frac{i(\mathcal{H}_V+\mathcal{H}_P)dt}{\hbar}}\right]\Psi(x,t)$$

Baker-Campbell-Hausdorff:

$$\Psi(x, t+dt) = \left[e^{-\frac{i\mathcal{H}_V dt}{\hbar}} e^{-\frac{i\mathcal{H}_P dt}{\hbar}} e^{-\frac{[i\mathcal{H}_V, i\mathcal{H}_P]dt^2}{2}}\right]\Psi(x,t)$$

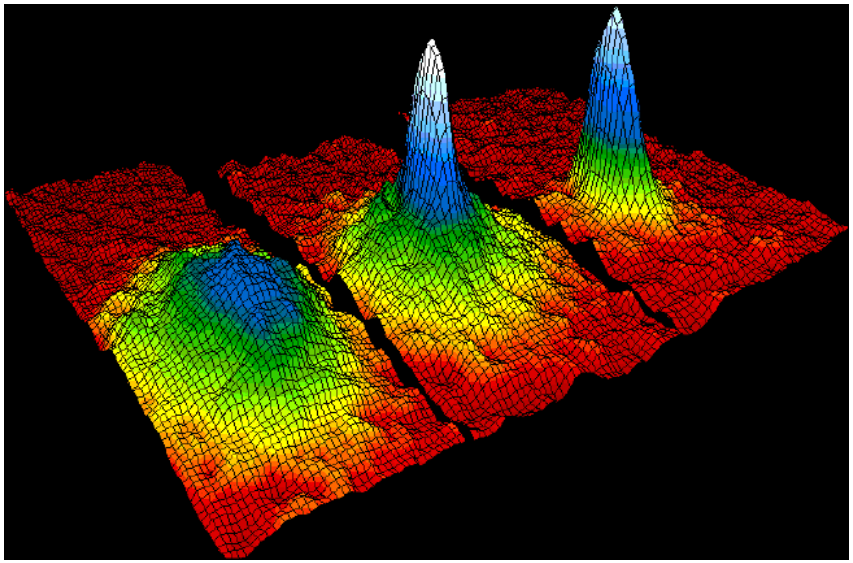
Matrices:

$$U_V = e^{-\frac{i\mathcal{H}_V dt}{\hbar}} \qquad U_P = e^{-\frac{i\mathcal{H}_P dt}{\hbar}}$$

$$\Psi(x, t+dt) = \mathcal{F}^{-1}\left[U_P(dt)\mathcal{F}\left[U_V(dt)\Psi(x,t)\right]\right] + \mathcal{O}(dt^2)$$

# Bose-Einstein condensation

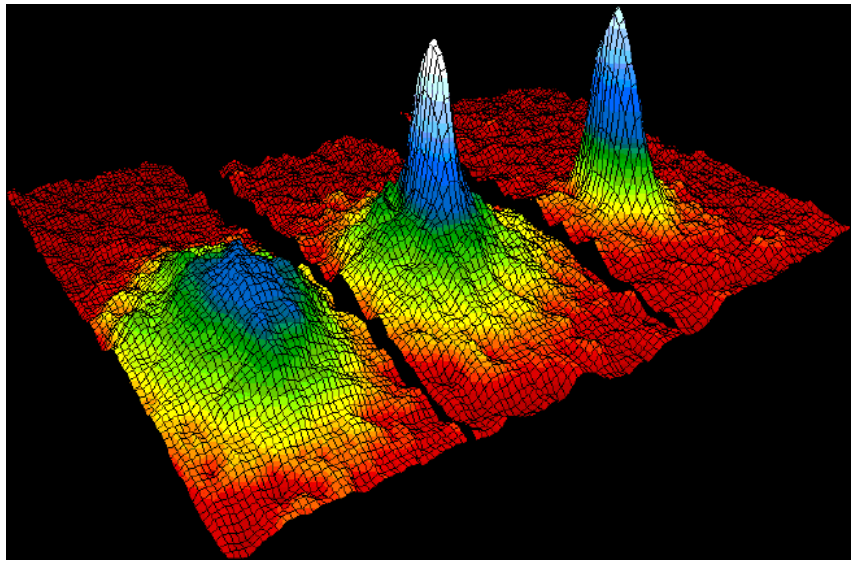
Bosons condense into a superfluid BEC



NIST/JILA/CU-Boulder

# Bose-Einstein condensation

Bosons condense into a superfluid BEC



NIST/JILA/CU-Boulder

Described by the mean field Gross–Pitaevskii equation:

$$i\hbar\frac{\partial}{\partial t}\Psi(x,t) = \left(-\frac{\hbar^2}{2m}\nabla^2 + \frac{1}{2}m\omega^2 x^2 + g|\Psi(x,t)|^2\right)\Psi(x,t)$$

# Superfluid rotation

Rotation leads to many vortices along the axis of rotation

Fluid

Credit: howstuffworks.com

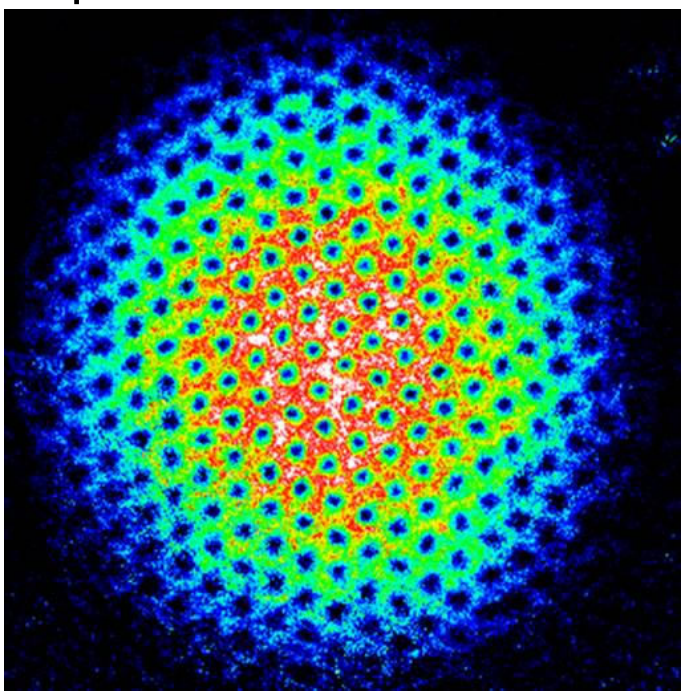
# Superfluid rotation

Rotation leads to many vortices along the axis of rotation

Fluid

Credit: howstuffworks.com

Superfluid



Credit: Peter Engels, JILA

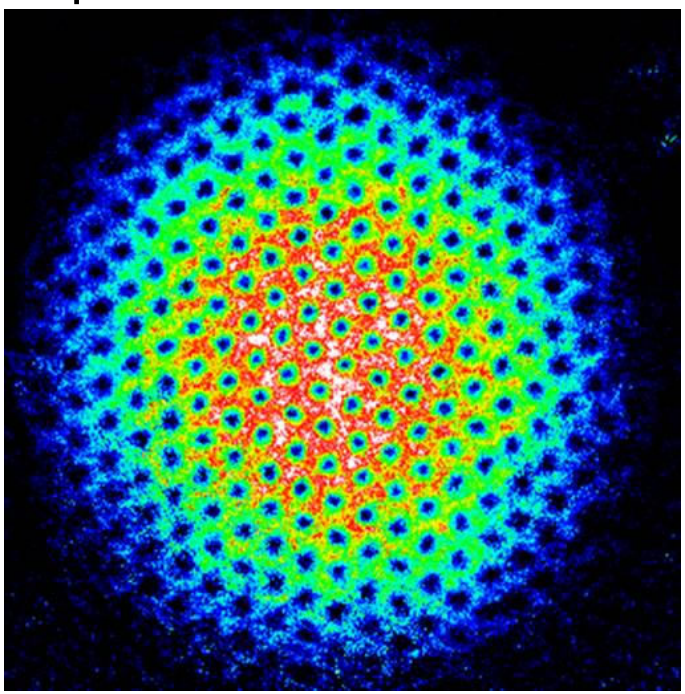
# Superfluid rotation

Rotation leads to many vortices along the axis of rotation

Fluid

Credit: howstuffworks.com

Superfluid



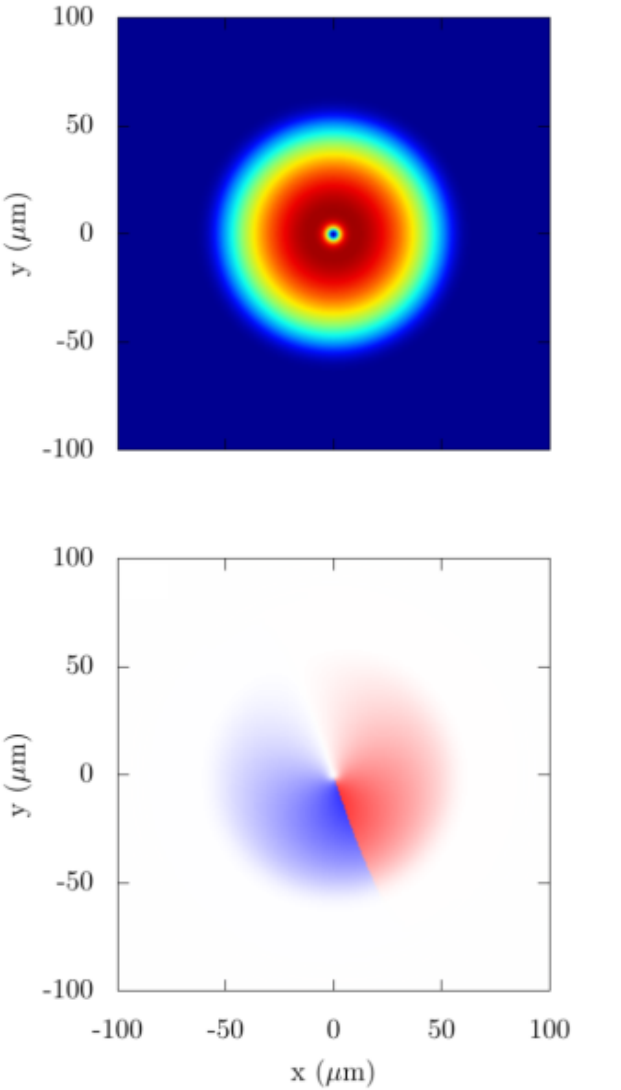
Credit: Peter Engels, JILA

$$i\hbar\frac{\partial}{\partial t}\Psi(x,t) = \left(-\frac{\hbar^2}{2m}\nabla^2 + V_0(x) + g|\Psi(x,t)|^2 - \Omega L_z\right)\Psi(x,t)$$

$$L_z = (xp_y - yp_x) = -i\hbar\left(x\frac{\partial}{\partial y} - y\frac{\partial}{\partial x}\right)$$

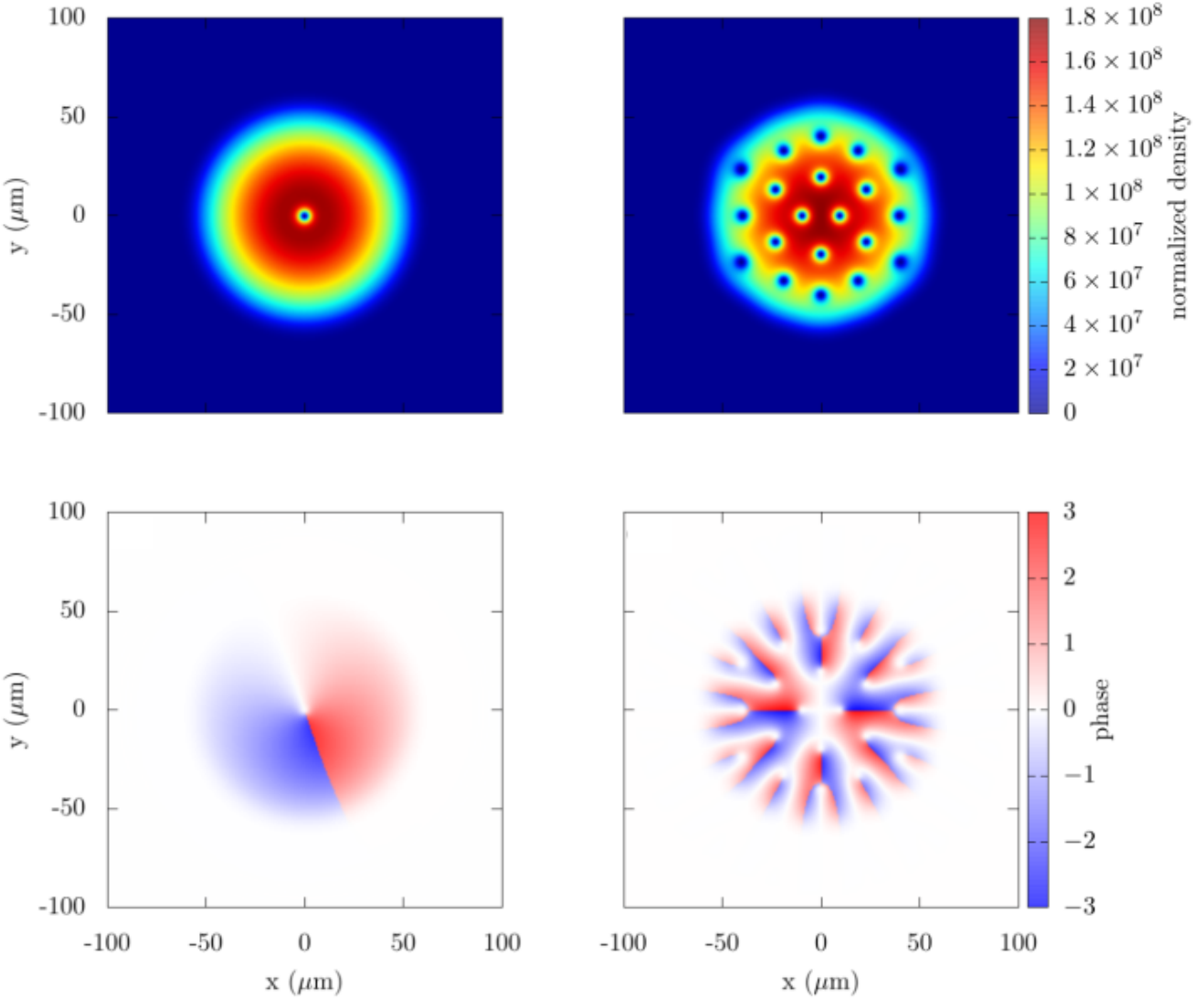
In the co-rotating frame

# Superfluid vortex phase




Each vortex has a $2\pi$ complex phase winding, $v \sim \nabla\phi$

# Superfluid vortex phase




Each vortex has a $2\pi$ complex phase winding, $v \sim \nabla\phi$

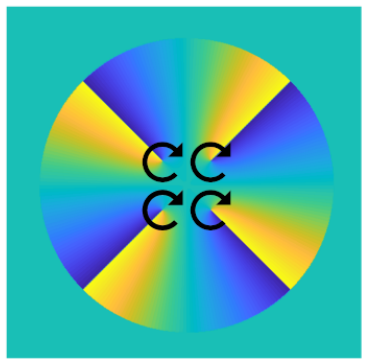
# What else can we do?

# Phase manipulation

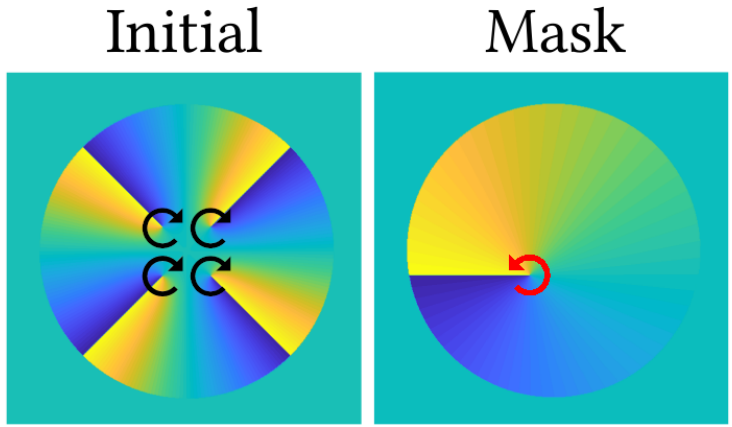
Initial

# Phase manipulation

Initial

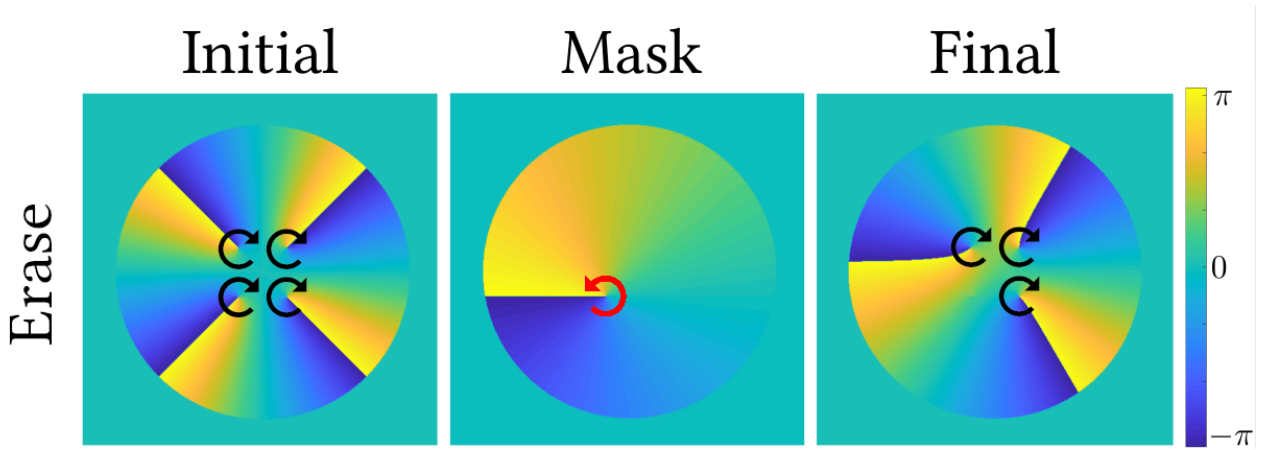


# Phase manipulation



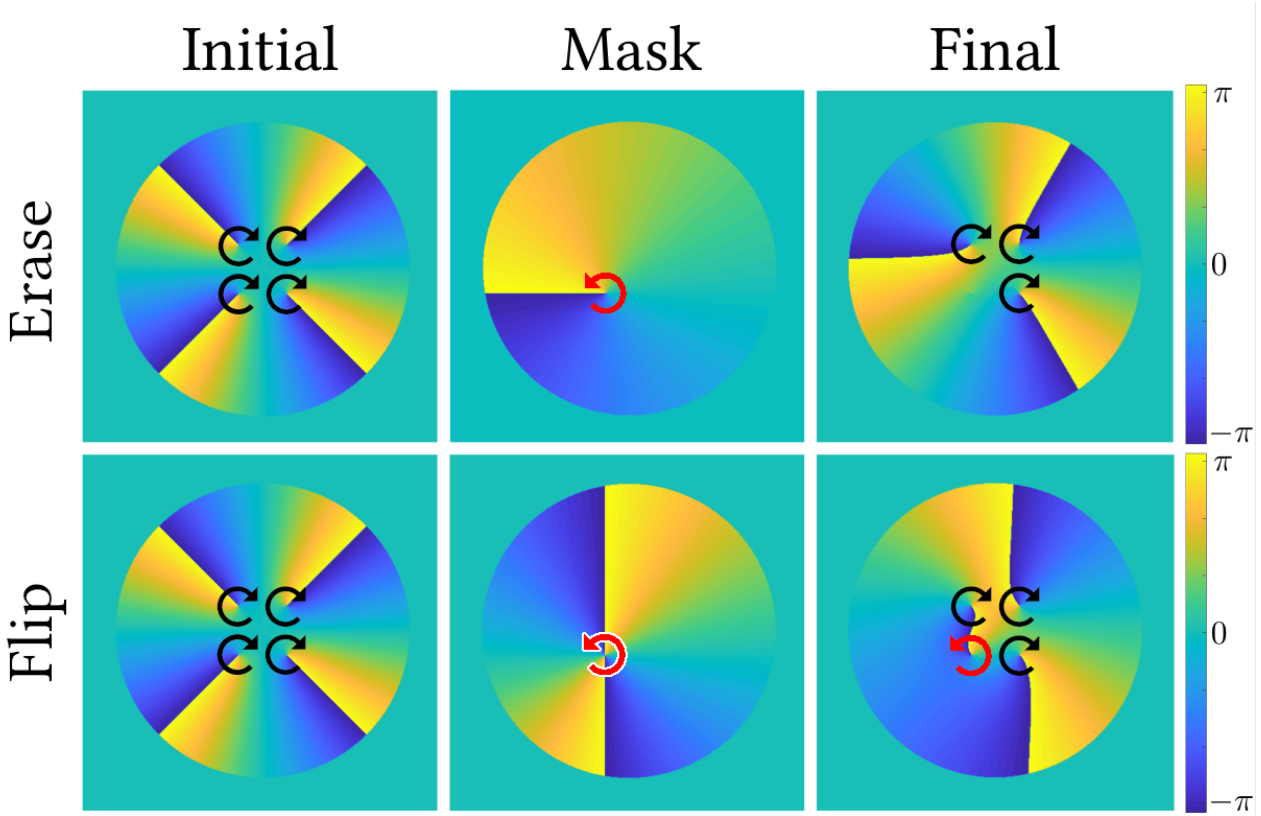

Phase masks can change the vortex distribution

# Phase manipulation



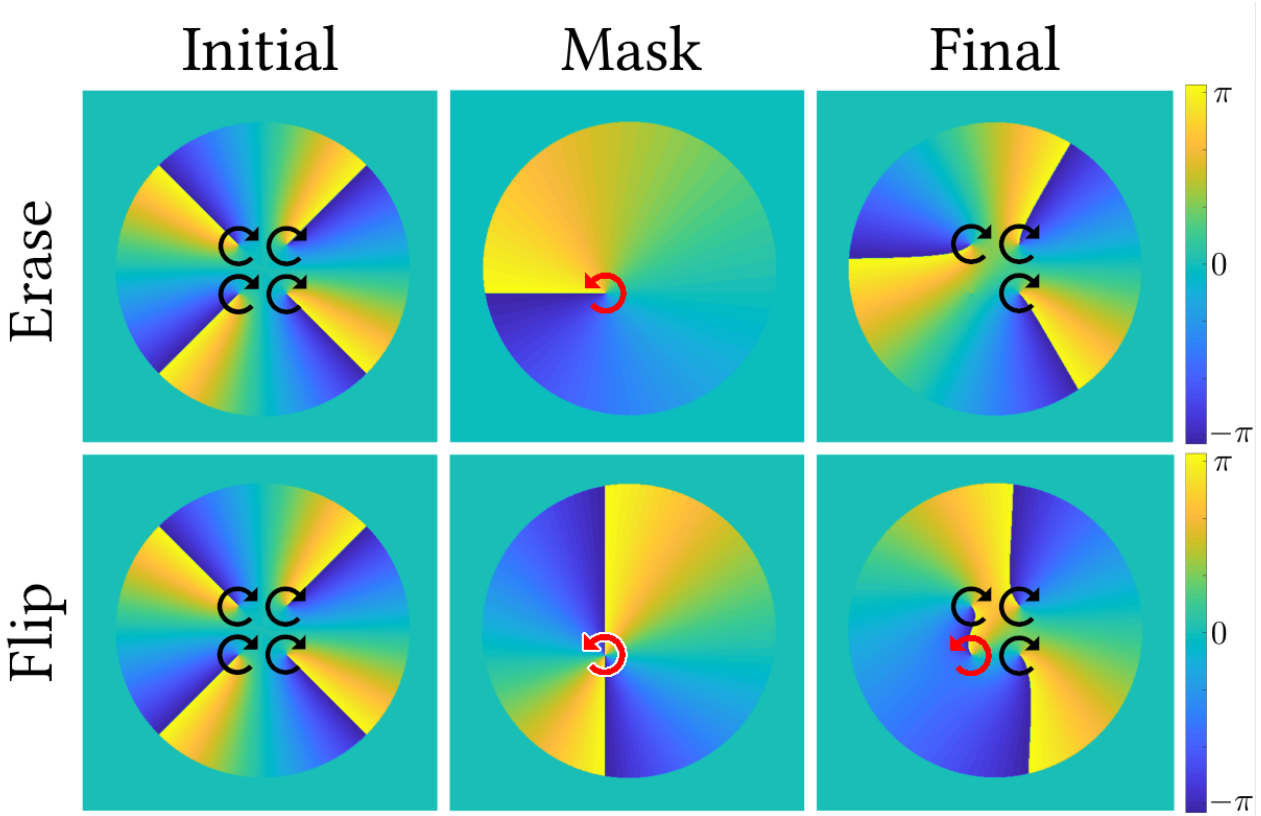

Phase masks can change the vortex distribution

# Phase manipulation




Phase masks can change the vortex distribution

# Phase manipulation

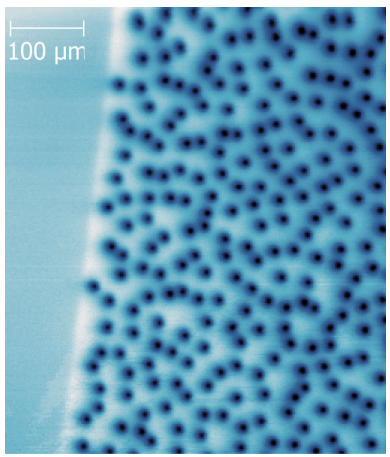



Phase masks can change the vortex distribution

... but will create energetic condensates

# Artificial magnetic fields

Magnetic fields cause rotation in *charged* particles
(Example: Type II superconductors)



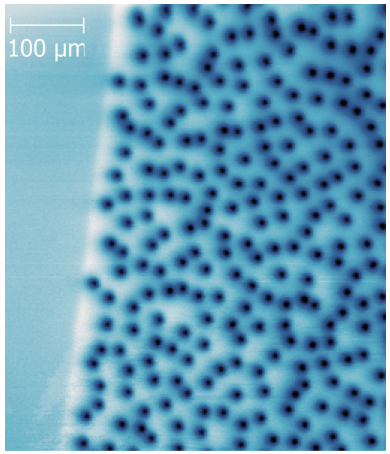

FS Wells *et al.*, *Scientific Reports* 5: 8677, 2015

# Artificial magnetic fields

Magnetic fields cause rotation in *charged* particles
(Example: Type II superconductors)

- The Hamiltonian of a particle with the Lorentz force law is:

$$\mathcal{H} = \frac{(p - q\mathbf{A}(x))^2}{2m}, \qquad \mathbf{B} = \nabla \times \mathbf{A}$$




FS Wells *et al.*, *Scientific Reports* 5: 8677, 2015

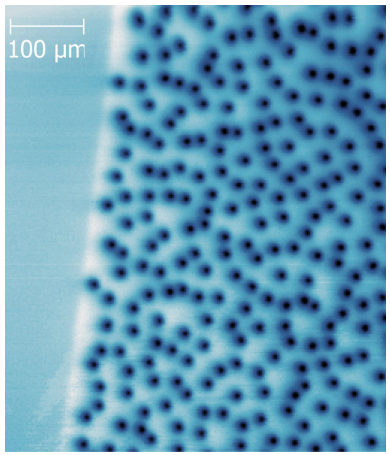
# Artificial magnetic fields

Magnetic fields cause rotation in *charged* particles
(Example: Type II superconductors)

- The Hamiltonian of a particle with the Lorentz force law is:

$$\mathcal{H} = \frac{(p - q\mathbf{A}(x))^2}{2m}, \qquad \mathbf{B} = \nabla \times \mathbf{A}$$

- Vortices follow the magnetic field lines




FS Wells *et al.*, *Scientific Reports* 5: 8677, 2015

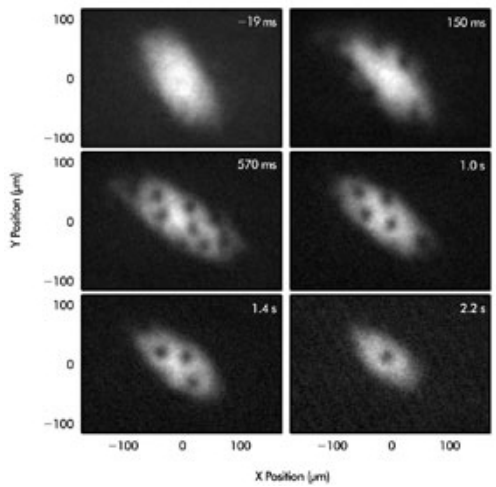
# Artificial magnetic fields

Magnetic fields cause rotation in *charged* particles
(Example: Type II superconductors)

- The Hamiltonian of a particle with the Lorentz force law is:

$$\mathcal{H} = \frac{(p - q\mathbf{A}(x))^2}{2m}, \qquad \mathbf{B} = \nabla \times \mathbf{A}$$

- Vortices follow the magnetic field lines




Ian Spielman, NIST

# GPE with gauge fields

With gauge fields, the GPE becomes

$$\mathcal{H} = \frac{(p - m\mathbf{A}(x))^2}{2m} + V_0 + g|\Psi(x,t)|^2$$

# GPE with gauge fields

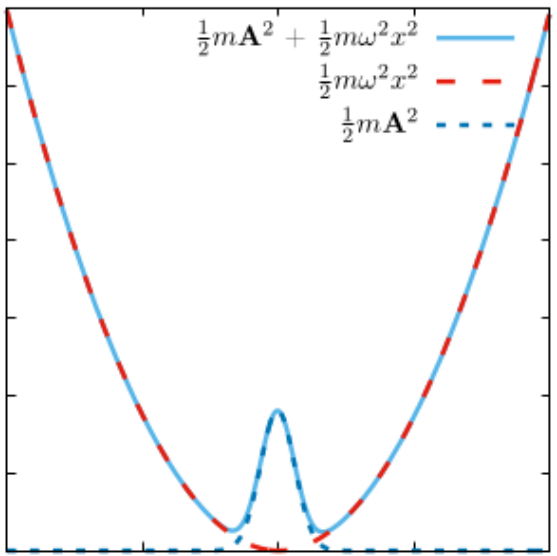
With gauge fields, the GPE becomes

$$\mathcal{H} = \frac{(p - m\mathbf{A}(x))^2}{2m} + V_0 + g|\Psi(x,t)|^2$$

Which creates

- A position-space component that couples with the trap

$$\frac{m\mathbf{A}(x)^2}{2}$$



# GPE with gauge fields

With gauge fields, the GPE becomes

$$\mathcal{H} = \frac{(p - m\mathbf{A}(x))^2}{2m} + V_0 + g|\Psi(x,t)|^2$$

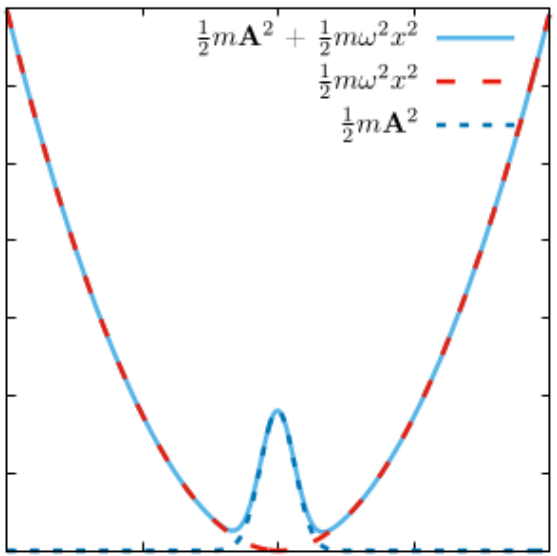
Which creates

- A position-space component that couples with the trap

$$\frac{m\mathbf{A}(x)^2}{2}$$

- Components in position and momentum-space, that **require 1D FFTs**

$$-\left(\frac{p\mathbf{A}(x) + \mathbf{A}(x)p}{2}\right)$$



# Intermittent summary

Physics

- Quantized vortices can be formed in BECs with rotation, phase imprinting, and artificial magnetic fields

# Intermittent summary

Physics

- Quantized vortices can be formed in BECs with rotation, phase imprinting, and artificial magnetic fields

Computer Science

- The SSFM requires a large number of FFTs
- This system is a great for testing spectral-like methods

# GPU computing and the GPUE codebase

**J Schloss**, LJ O'Riordan

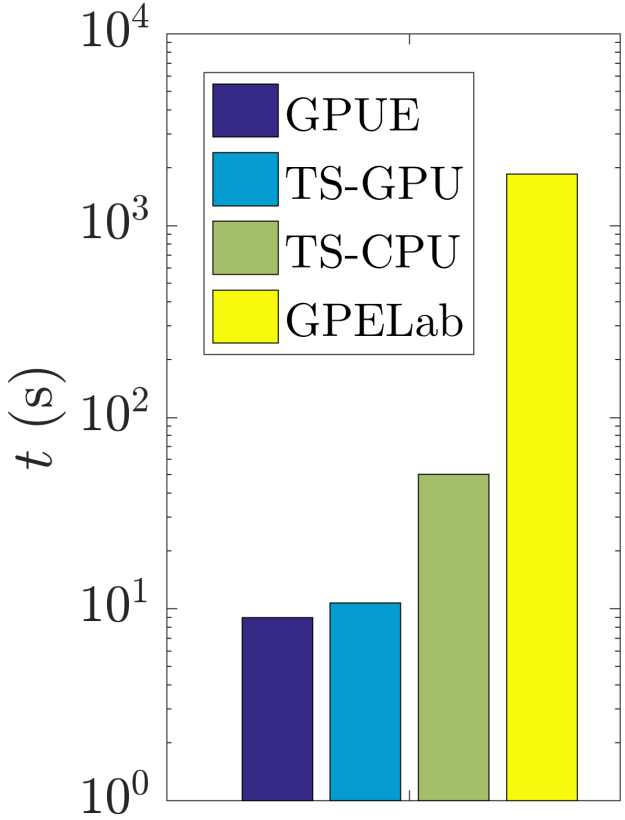
*Journal of Open Source Software* 3 (32):1037, 2018

# What is a GPU?

- Graphics processing units (GPUs) are massively parallel computing devices

# What is a GPU?

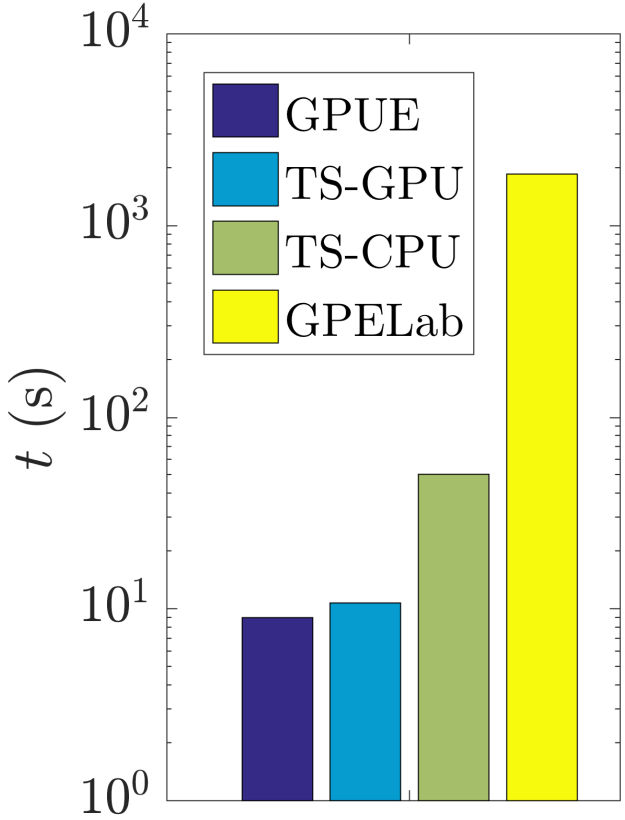
- Graphics processing units (GPUs) are massively parallel computing devices
- GPUs are fast for parallel tasks


Total runtime, smaller has higher performance!

# What is a GPU?

- Graphics processing units (GPUs) are massively parallel computing devices
- GPUs are fast for parallel tasks
- Summit uses GPUs

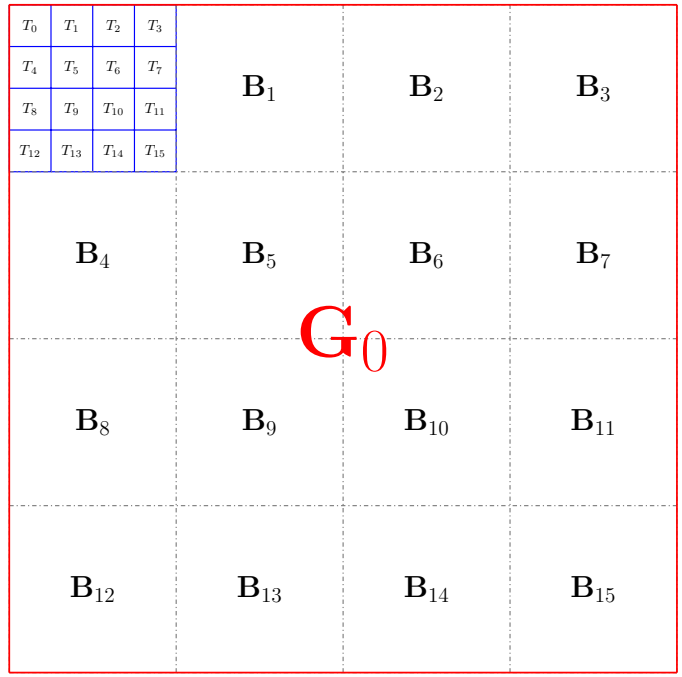




Total runtime, smaller has higher performance!

# GPU memory hierarchy

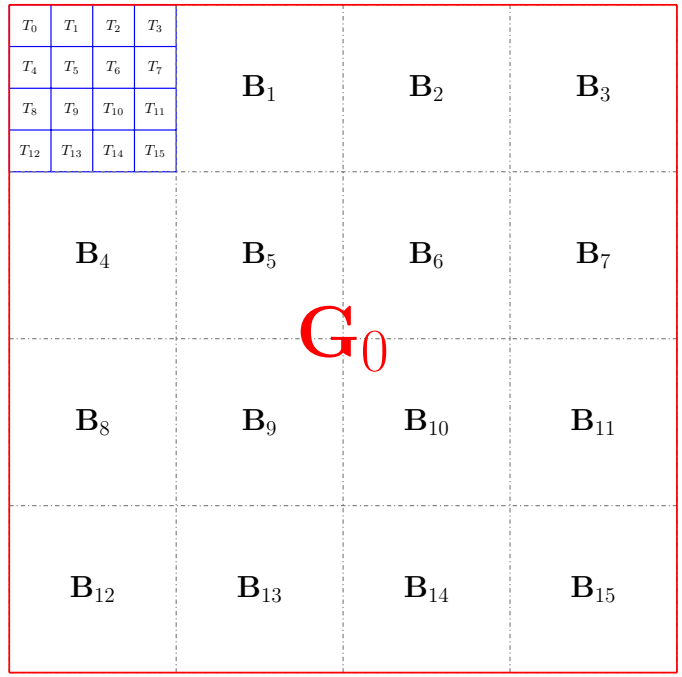
# GPU memory hierarchy

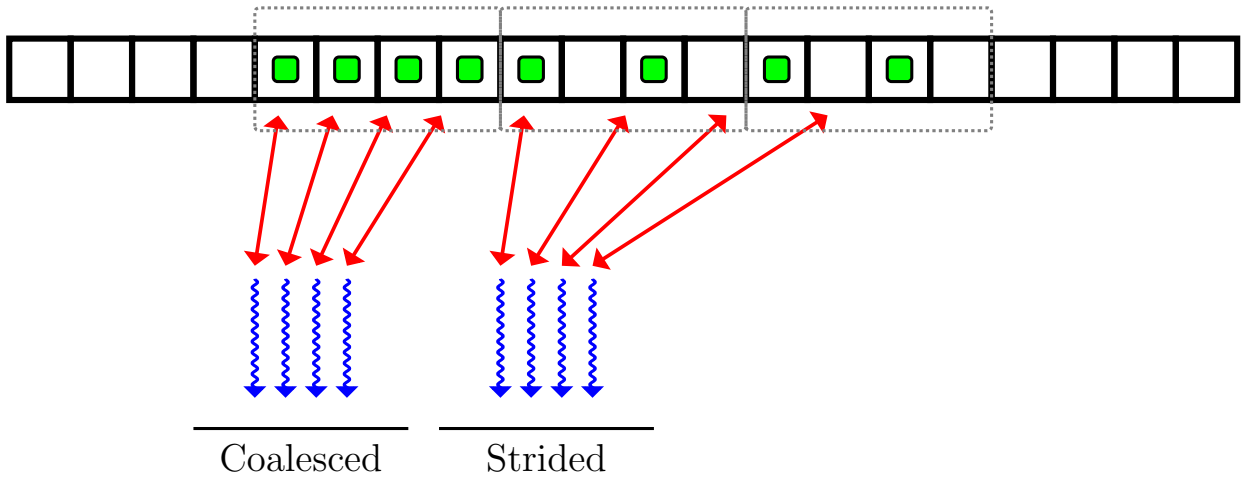
- Computing threads in blocks, blocks in grids



# GPU memory hierarchy

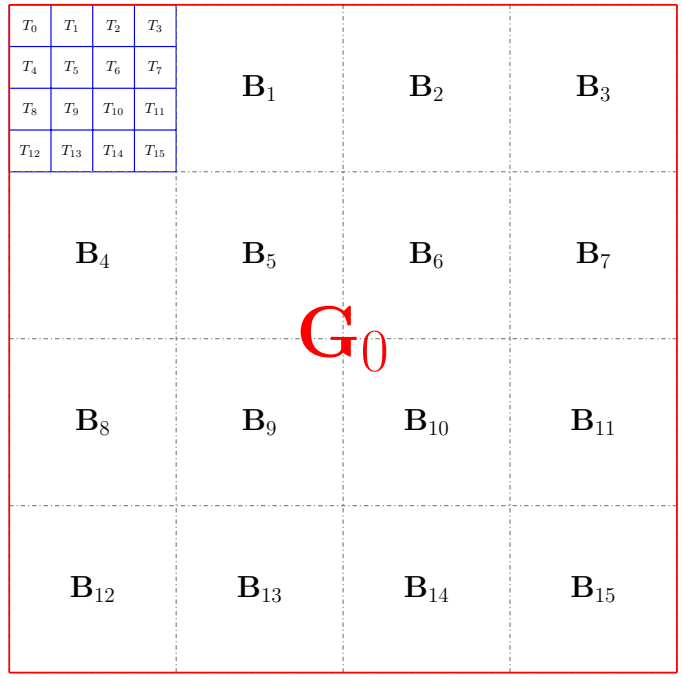
- Computing threads in blocks, blocks in grids
- Memory coalescence

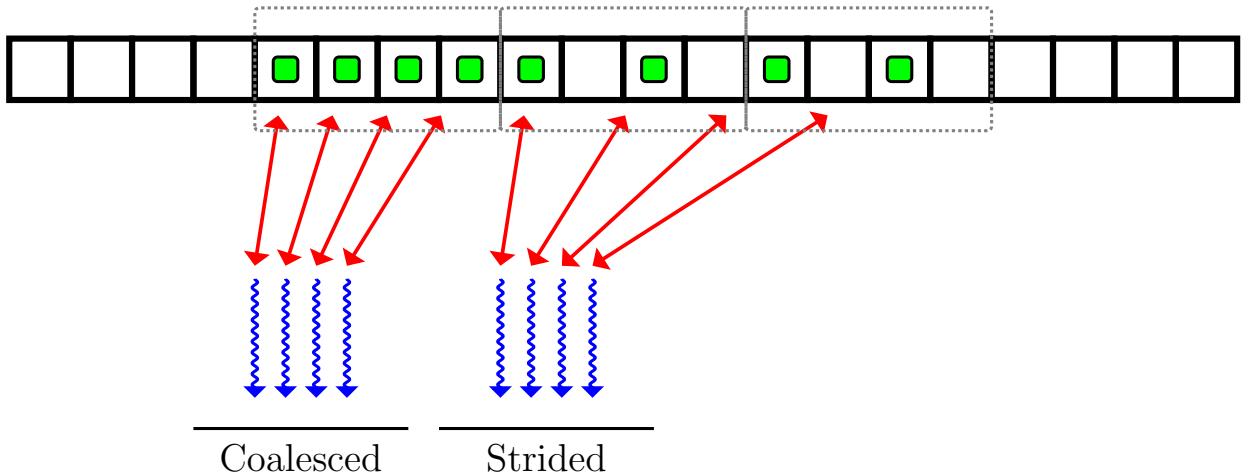


# GPU memory hierarchy

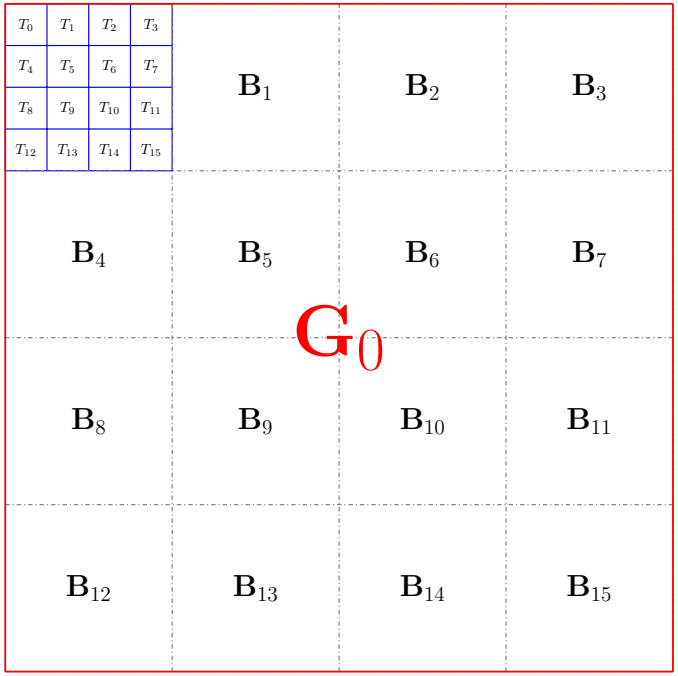
- Computing threads in blocks, blocks in grids
- Memory coalescence
- Data transfer is slow

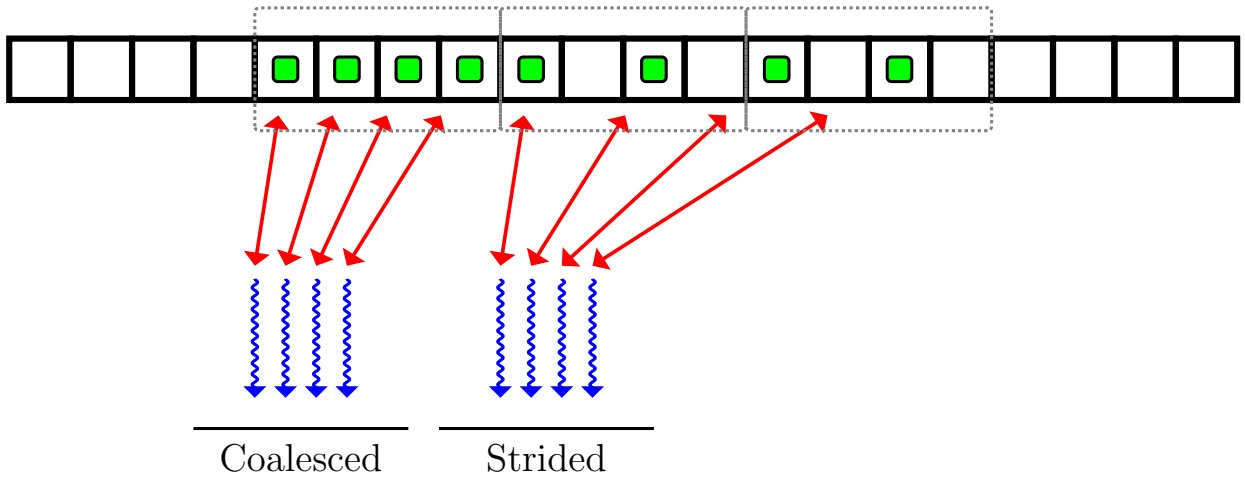


# GPU memory hierarchy

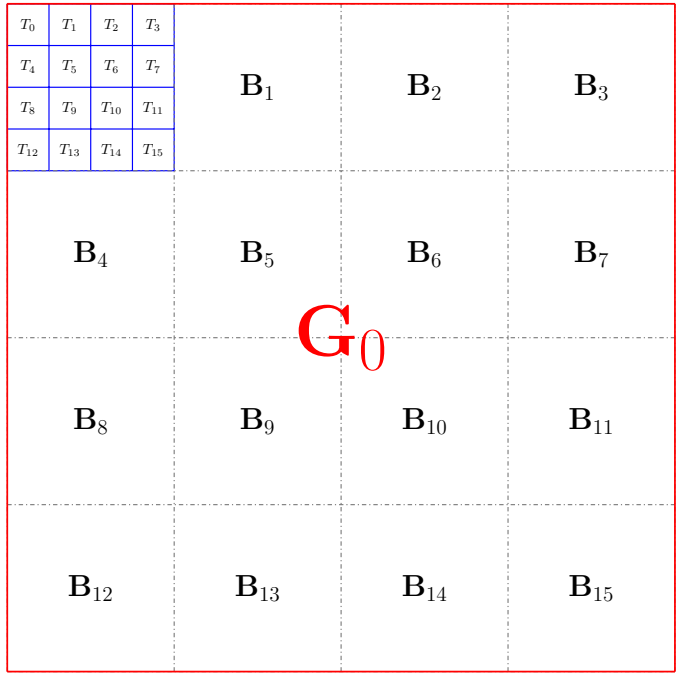
- Computing threads in blocks, blocks in grids
- Memory coalescence
- Data transfer is slow
- Recursion and iteration is slow

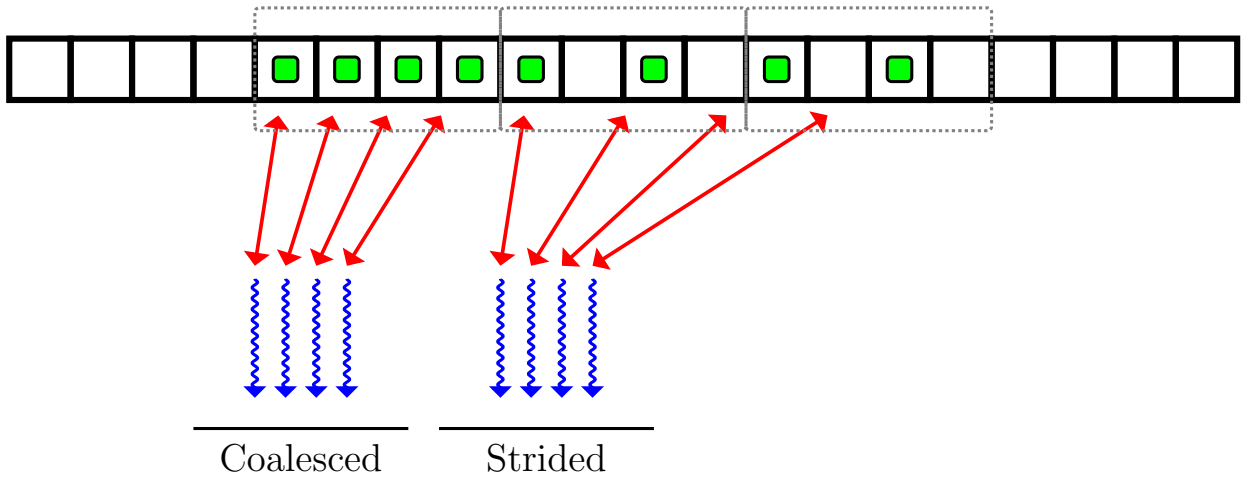


# GPU memory hierarchy

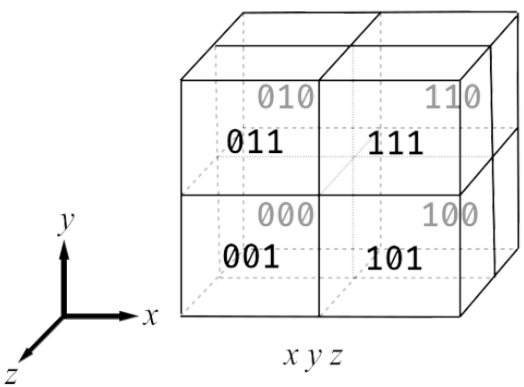
- Computing threads in blocks, blocks in grids
- Memory coalescence
- Data transfer is slow
- Recursion and iteration is slow
- Limited memory

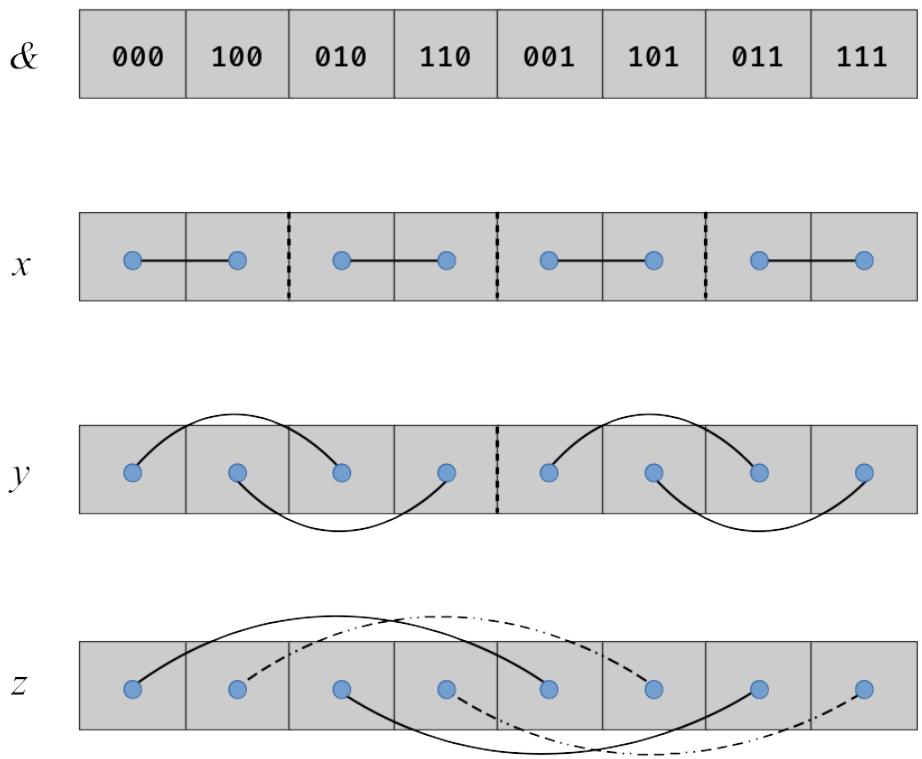


# FFT notes

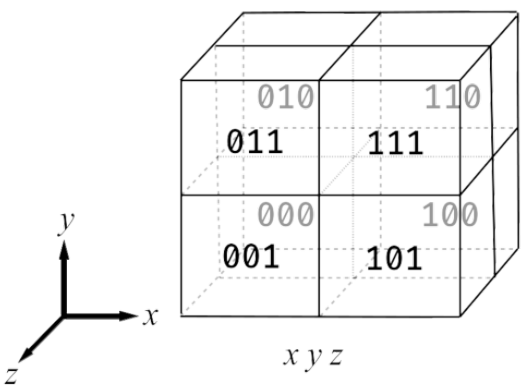
- Global operations
- 1D FFTs are not supported by CuFFT

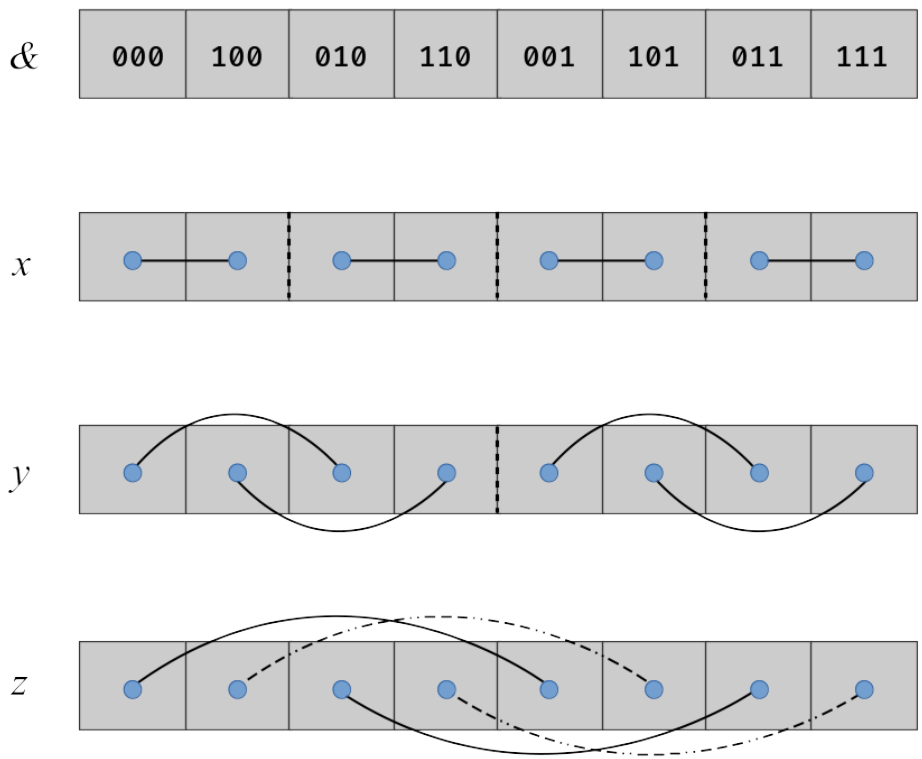


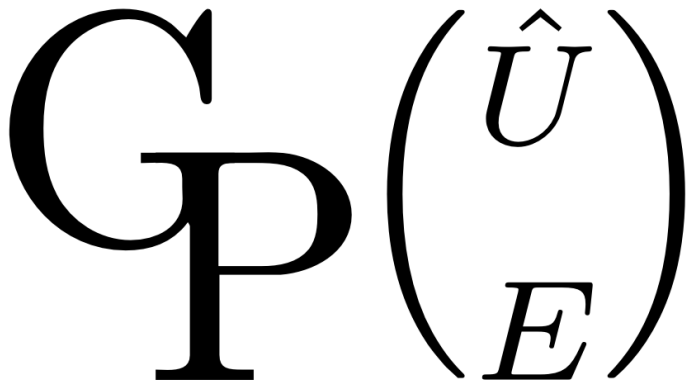
# FFT notes

- Global operations
- 1D FFTs are not supported by CuFFT
- Transposes are necessary



# Two important features



- Novel features useful for understanding examples
- No discussion of implementation details

# How do we scale?

Notes:

- GPUs have limited memory, so we should minimize the memory footprint with dynamic kernels

# How do we scale?

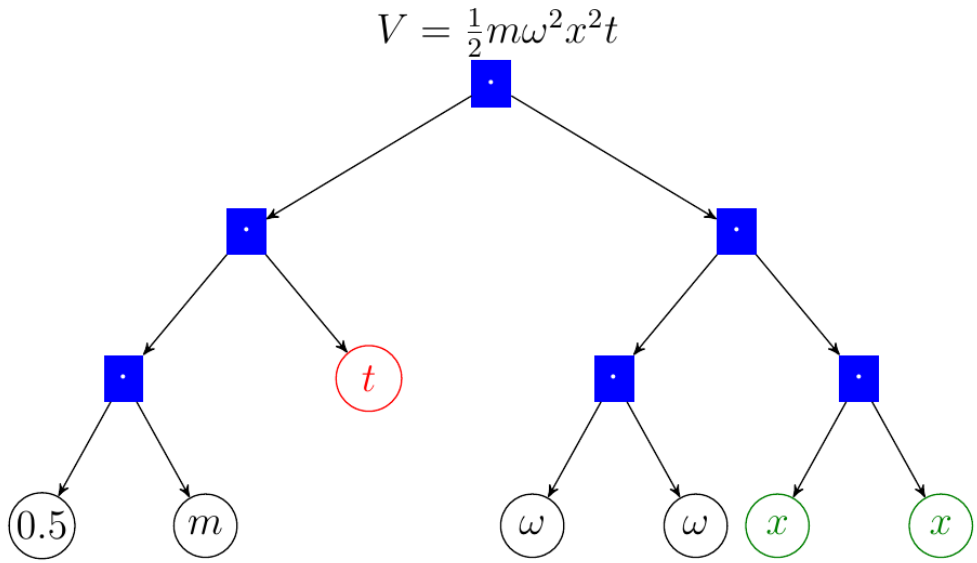
Notes:

- GPUs have limited memory, so we should minimize the memory footprint with dynamic kernels
- CUDA code is hard to write and users want time-dependent fields for state engineering

# How do we scale?

Notes:

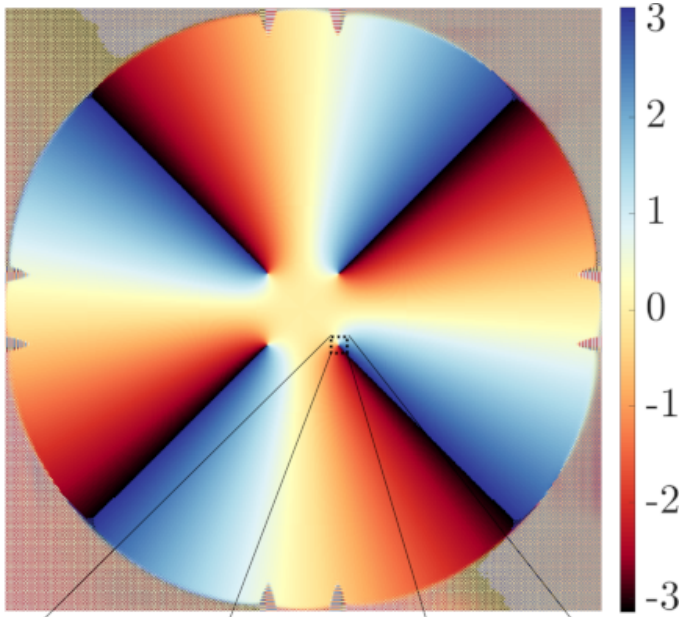
- GPUs have limited memory, so we should minimize the memory footprint with dynamic kernels
- CUDA code is hard to write and users want time-dependent fields for state engineering




Expression trees

# Vortex tracking and highlighting

- Phase plaquettes in 2D

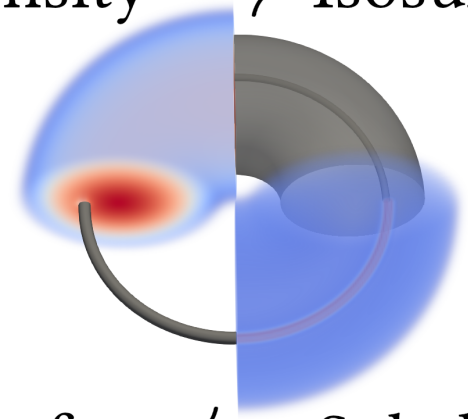



| | | | | | |
|---|---|---|---|---|---|
| -0.264 | 1.8e-07 | 0.63 | 1.37 | 1.71 | 1.86 |
| -0.657 | -0.63 | 1.6e-06 | 2.16 | 2.22 | 2.22 |
| -1.07 | -1.37 | -2.16 | 3.14 | 2.76 | 2.59 |
| -1.38 | -1.71 | -2.22 | -2.76 | 3.14 | 2.9 |
| -1.56 | -1.86 | -2.22 | -2.59 | -2.9 | 3.14 |
| -1.67 | -1.92 | -2.2 | -2.49 | -2.75 | -2.96 |

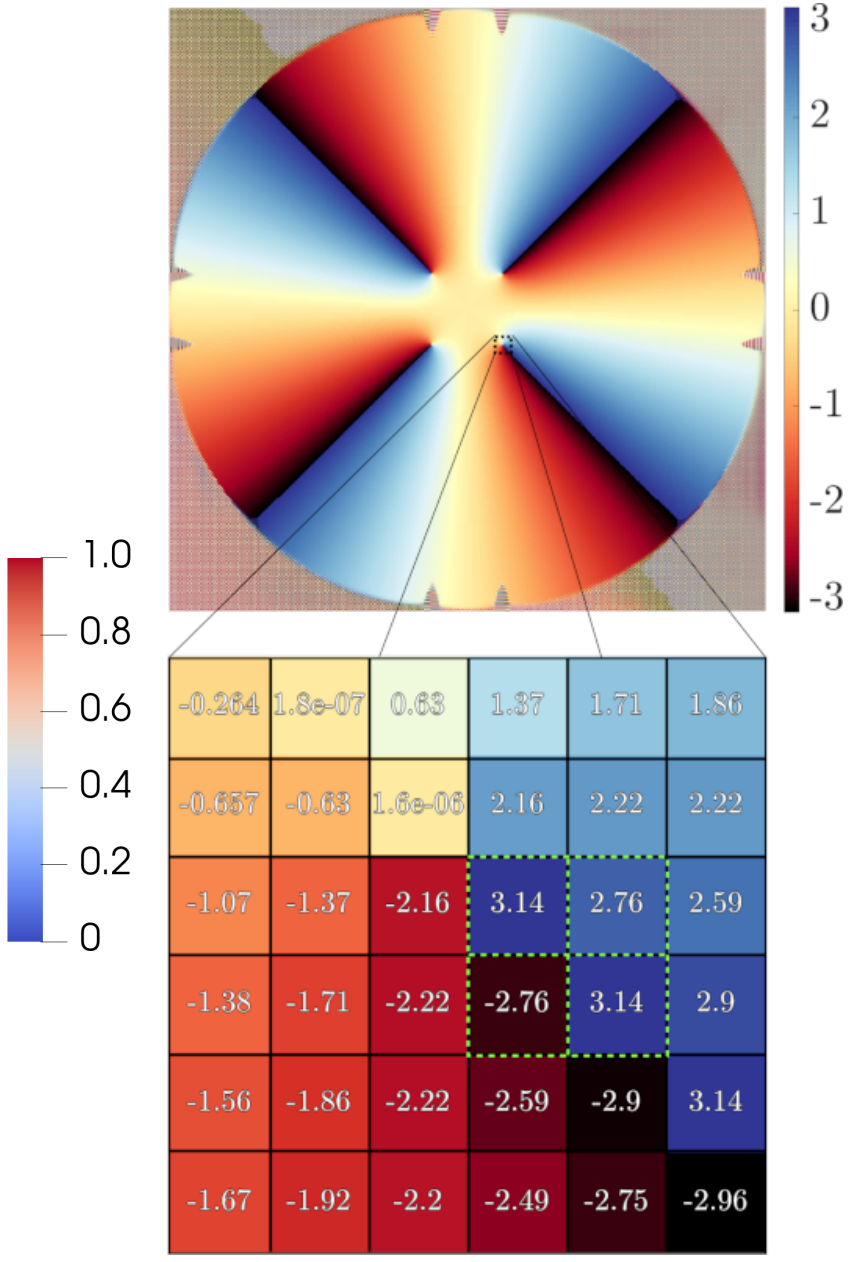
# Vortex tracking and highlighting

- Phase plaquettes in 2D
- Vortex highlighting in 3D

Density $\longrightarrow$ Isosurface



Isosurface $\longleftarrow$ Sobel



# Intermittent summary

Physics

- GPUE is fast, can do dynamic simulations, and vortex detection
- Multi-component simulations and HDF5 have also been implemented

# Intermittent summary

Physics

- GPUE is fast, can do dynamic simulations, and vortex detection
- Multi-component simulations and HDF5 have also been implemented

Computer Science

- Expression trees save a lot of GPU memory
- Distributed transpose methods are in development

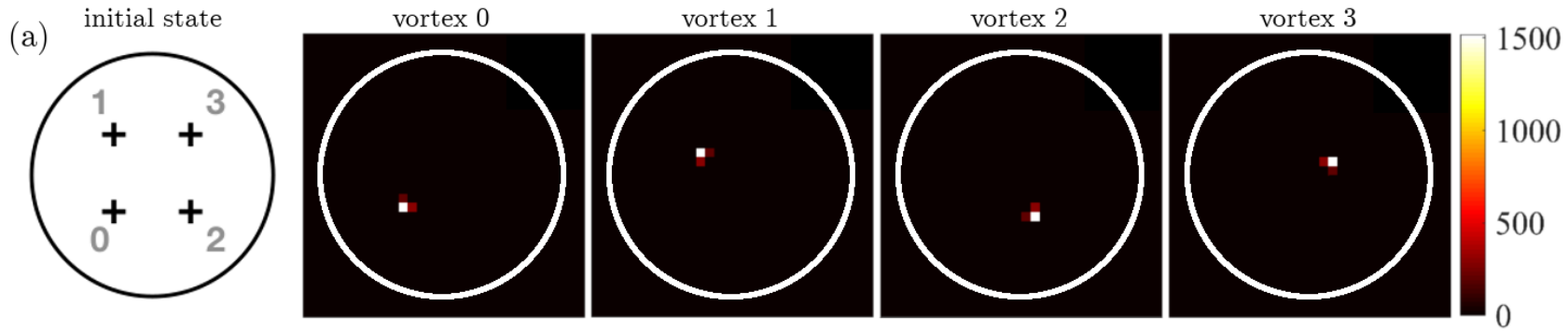
# Chaotic vortex dynamics in 2D BEC

T Zhang, **J Schloss**, A Thomasen, LJ O'Riordan, T Busch, A White

*Physical Review Fluids* 4 (5):054701, 2019

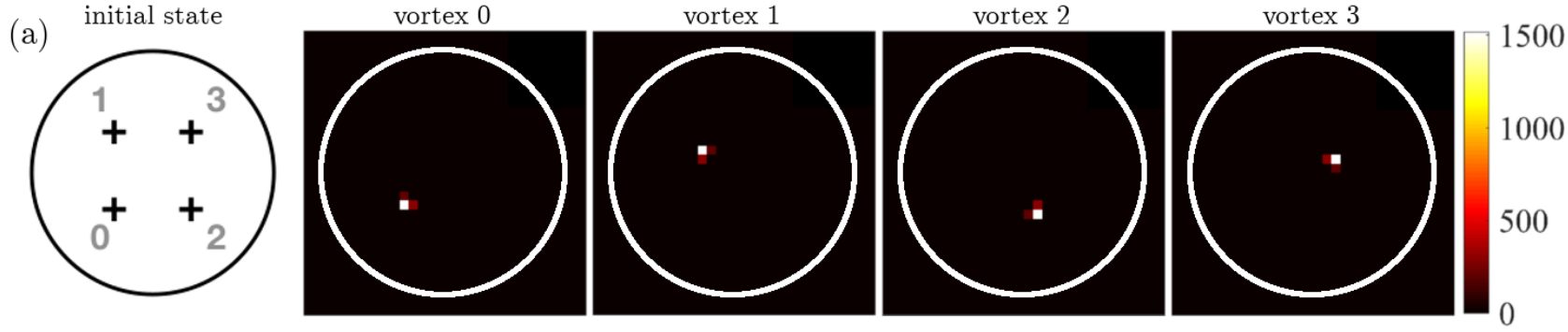
# Few vortex dynamics

4 vortices:

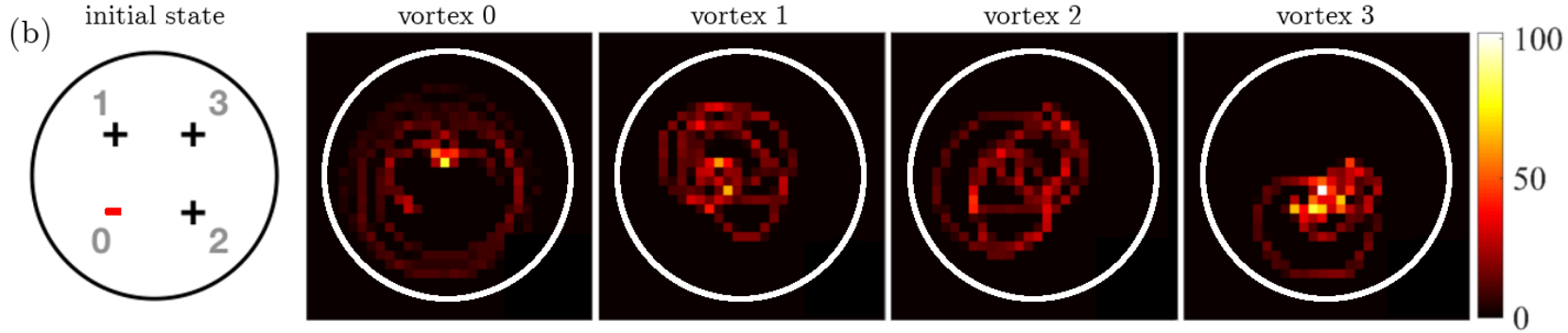


# Few vortex dynamics

4 vortices:

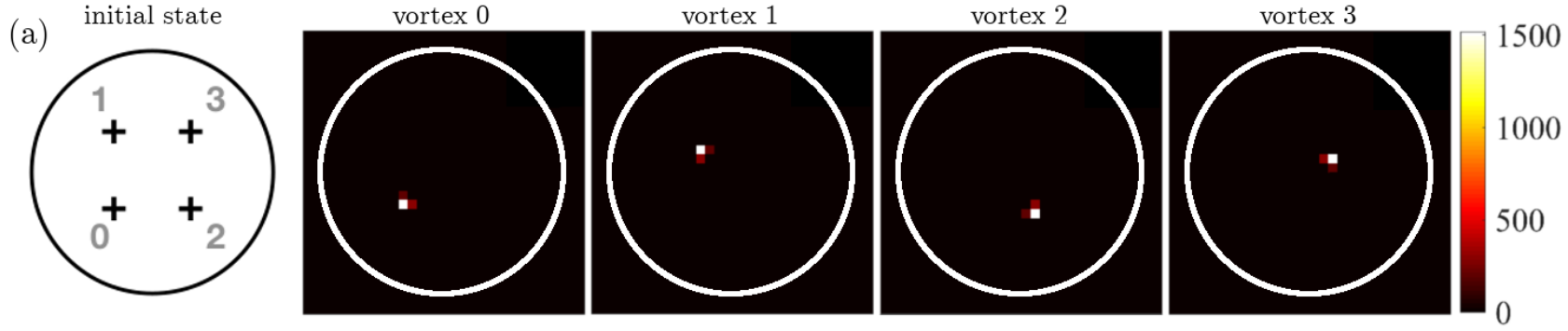



4 Vortex, 1 anti-vortex:

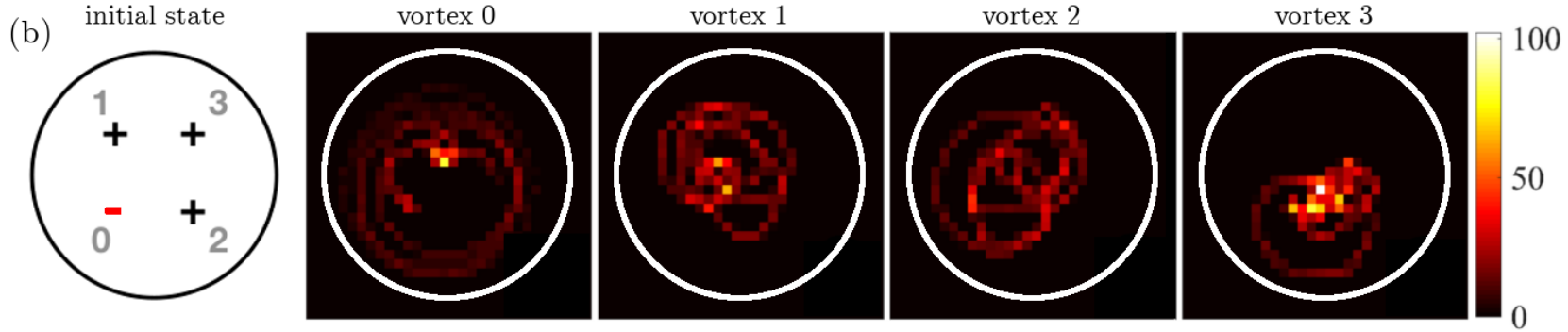


# Few vortex dynamics

4 vortices:




4 Vortex, 1 anti-vortex:

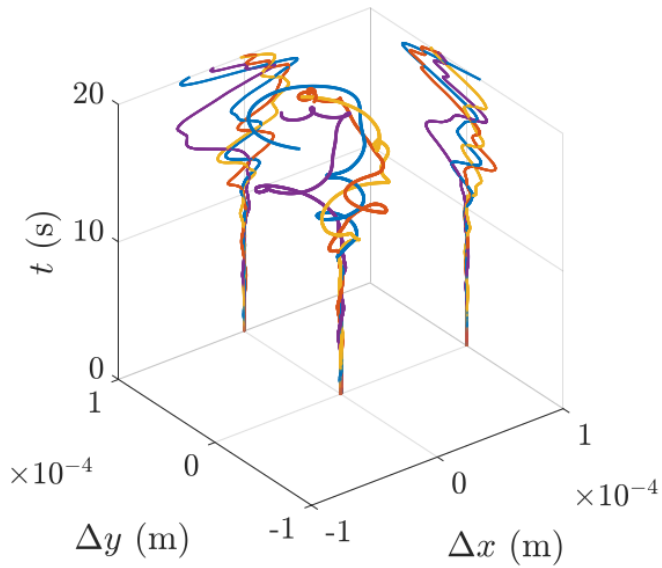



Is this chaotic?

# Divergence in trajectories

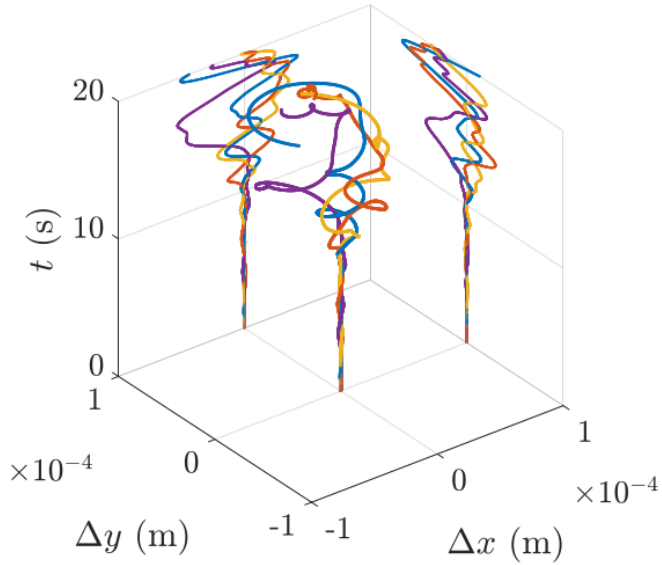
# Divergence in trajectories

Divergence in trajectories at $\sim$10s



# Divergence in trajectories

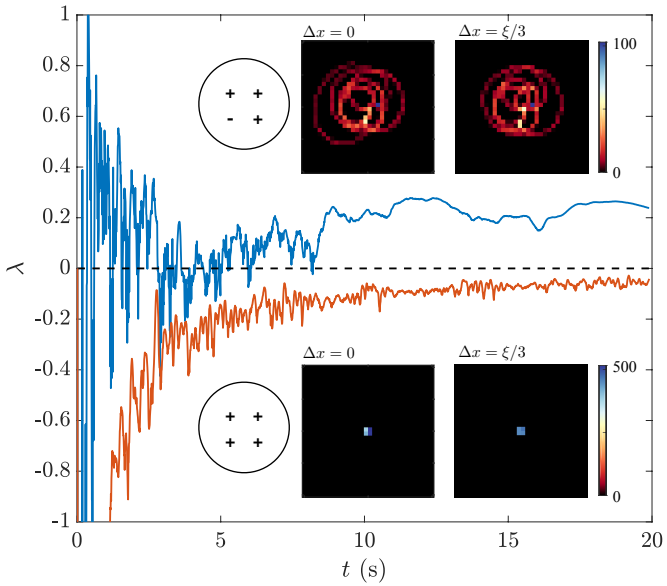
Divergence in trajectories at ∼10s




Divergence in trajectory when Lyapunov exponent ($\lambda$) becomes positive

$$\delta\mathbf{P}(\mathbf{t}) = (\delta x(t), \delta y(t), \delta v_x(t), \delta v_y(t))$$

$$|\delta\mathbf{P}(t)| \approx e^{\lambda t}|\delta\mathbf{P}_0|$$



# Divergence in trajectories

# Intermittent summary

Physics

- This is a chaotic system with a controlled initial state
- Chaotic dynamics of few-vortex systems is accelerated by collisional events

# Intermittent summary

Physics

- This is a chaotic system with a controlled initial state
- Chaotic dynamics of few-vortex systems is accelerated by collisional events

Computer Science

- 50 TB of data
- Reliant on post-processing metrics (Lyapunov exponent, Vortex tracking)
- 1 hour per run (on K80's), infeasible with other software

# 3D vortex ring generation in toroidal BEC systems

**J Schloss**, P Barnett, R Sachdeva, T Busch
arXiv:1910.02364

Submitted to *Physical Review Fluids*

# Artificial magnetic fields

Magnetic fields cause rotation in *charged* particles

# Artificial magnetic fields

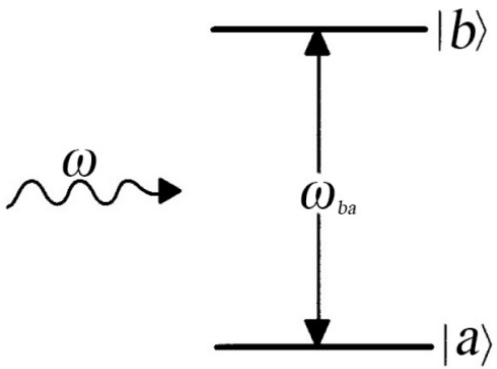
Magnetic fields cause rotation in *charged* particles

- If a two-level atom moves slowly in a tuned light field, *Berry's connection* is

$$\mathbf{A} = i\hbar \langle \psi_l | \nabla \psi_l \rangle$$

- The magnetic field is

$$\mathbf{B} = \nabla \times \mathbf{A}$$



# Artificial magnetic fields

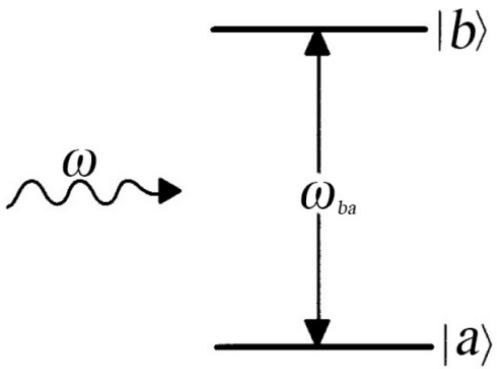
Magnetic fields cause rotation in *charged* particles

- If a two-level atom moves slowly in a tuned light field, *Berry's connection* is

$$\mathbf{A} = i\hbar \langle \psi_l | \nabla \psi_l \rangle$$

- The magnetic field is

$$\mathbf{B} = \nabla \times \mathbf{A}$$




- Artificial magnetic fields are created with a strongly varying electric field

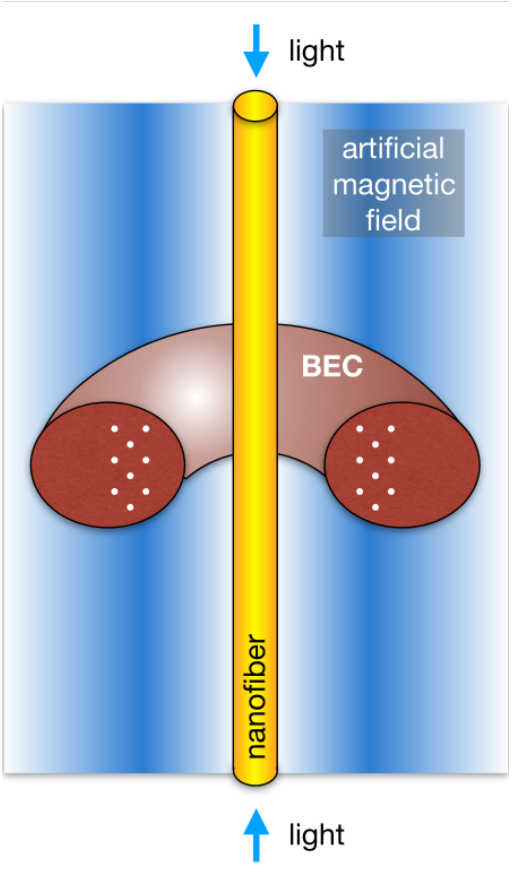
# Optical nanofibers and BECs

- Nanofiber has exponentially decaying evanescent field
- Nanofiber generates **A**

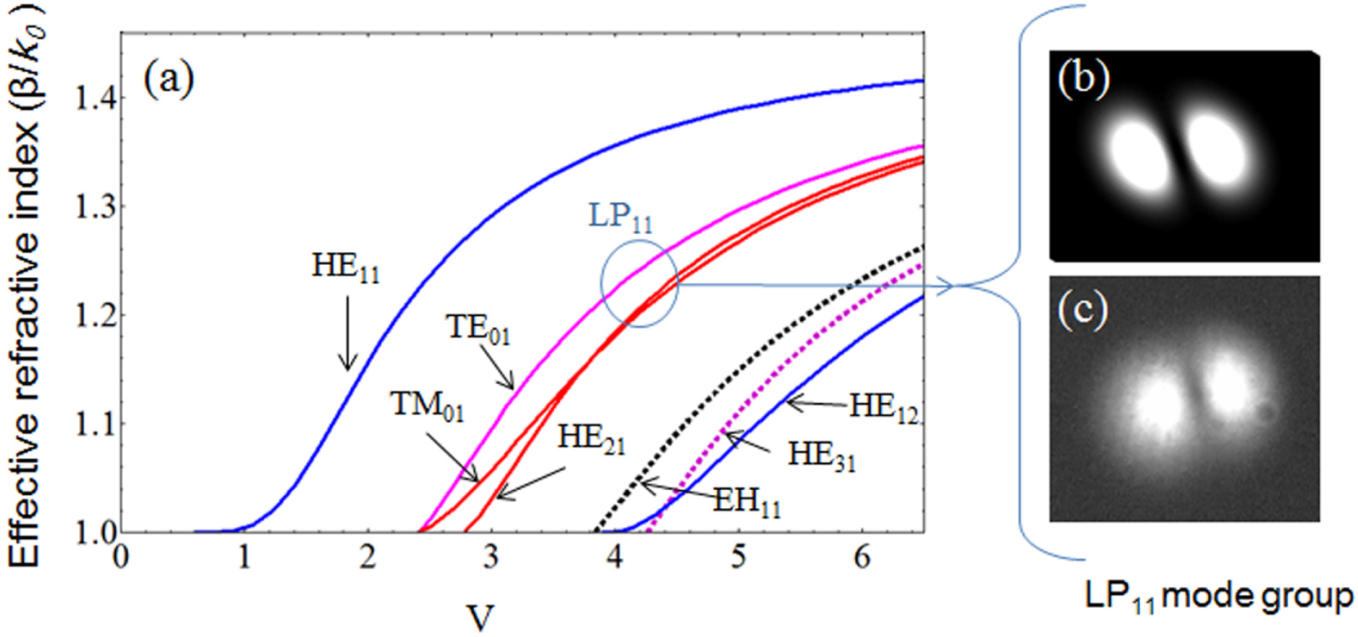

MJ Morrissey *et al. Sensors* 13(8):10449-81, 2013

# Optical nanofibers and BECs

- Nanofiber has exponentially decaying evanescent field
- Nanofiber generates **A**
- BEC toroidally trapped around nanofiber
- Vortices follow $\mathbf{B} = \nabla \times \mathbf{A}$



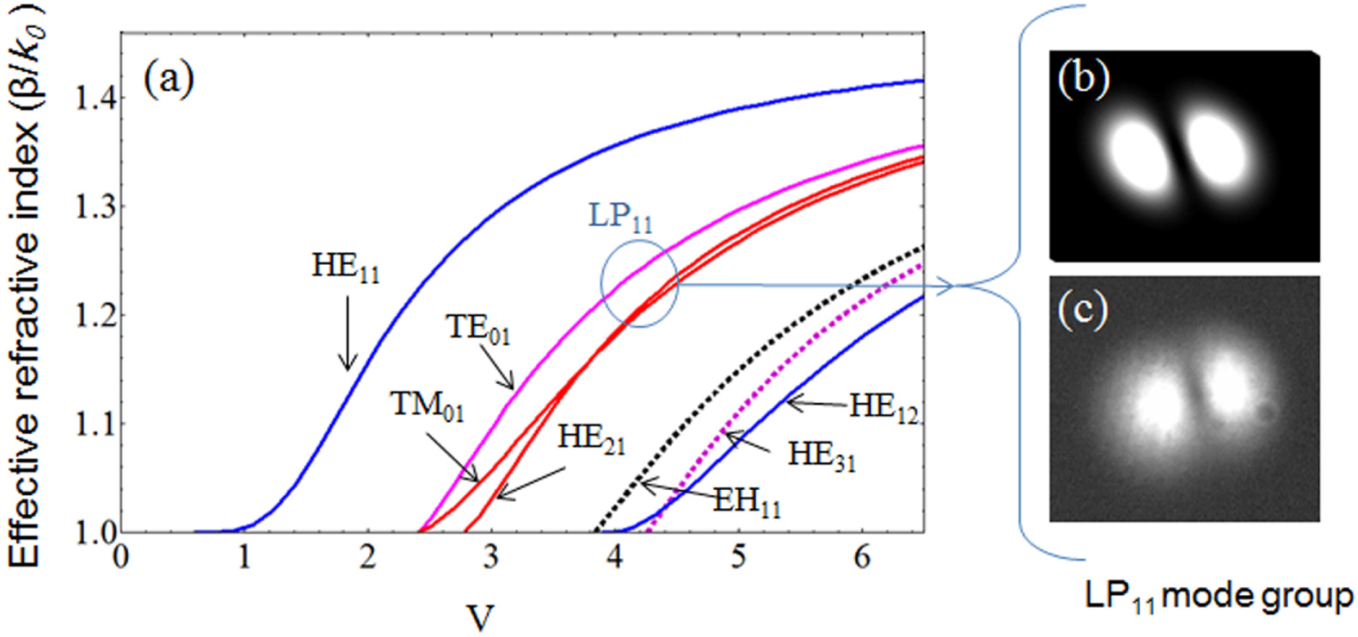
# Optical nanofiber modes




Kumar *et al.* *NJP* 17(1):013026, 2015

$$V = k_0 a \sqrt{n_1^2 - n_2^2}$$

# Optical nanofiber modes

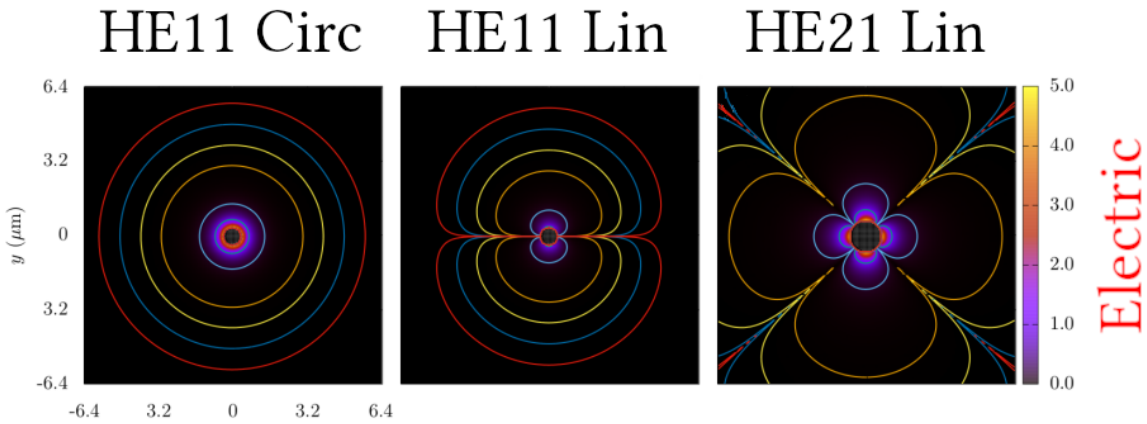



Kumar *et al.* *NJP* 17(1):013026, 2015

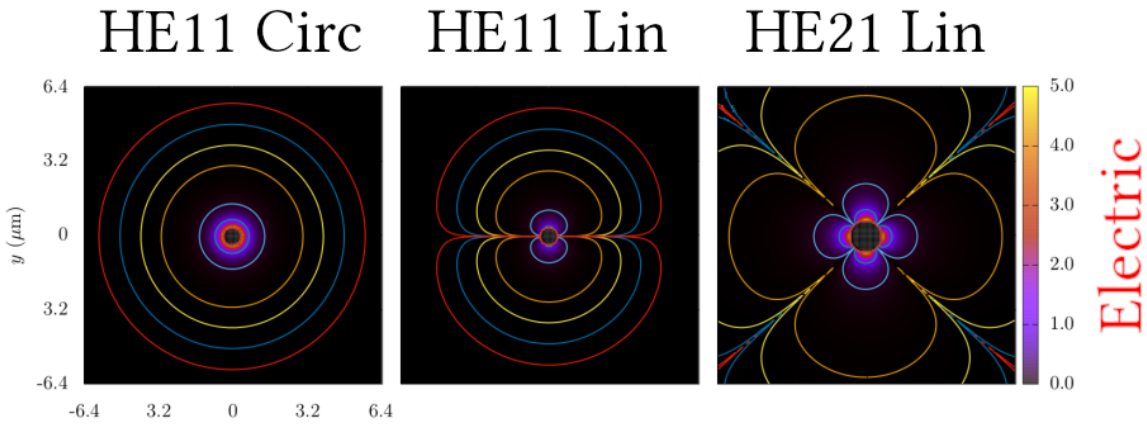
$$V = k_0 a \sqrt{n_1^2 - n_2^2}$$

- Different optical modes propagate at different fiber widths
- We will focus on HE11 and HE21 with linear and circular polarization

# Optical nanofiber fields

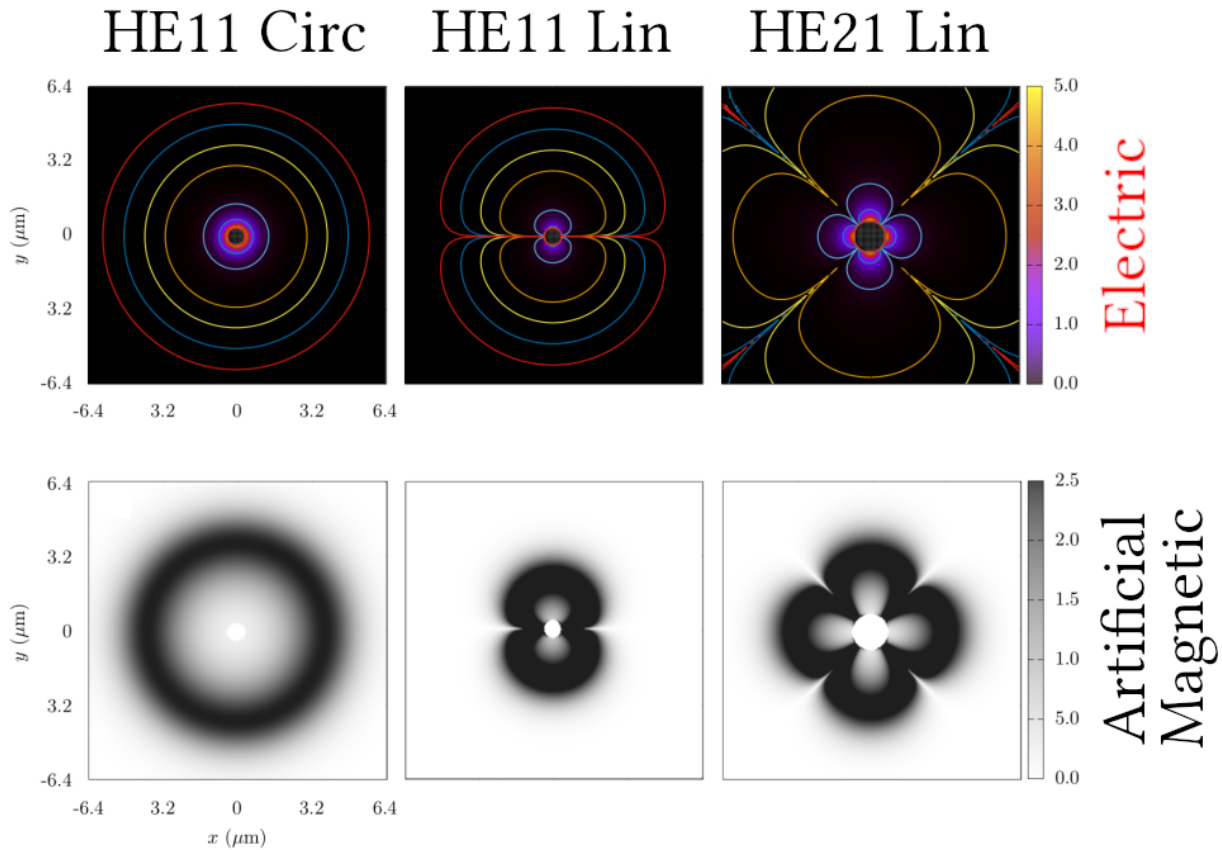


# Optical nanofiber fields




$$\mathbf{B} = \frac{\hbar\kappa_0 s^2 (n_1 + 1)}{(1 + \tilde{s}^2 |d_r E_r + d_\phi E_\phi + d_z E_z|^2)^2} \times \left[ \hat{\phi} \frac{\partial}{\partial r} |d_r E_r + d_\phi E_\phi + d_z E_z|^2 - \hat{r} \frac{1}{r} \frac{\partial}{\partial \phi} |d_r E_r + d_\phi E_\phi + d_z E_z|^2 \right]$$

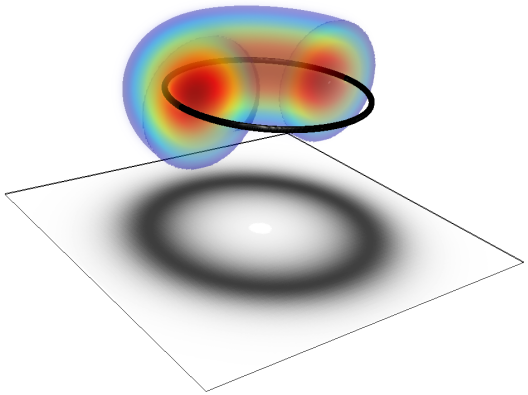
# Optical nanofiber fields



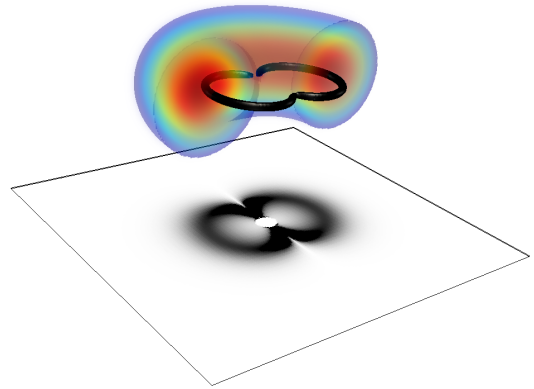

$$\mathbf{B} = \frac{\hbar\kappa_0 s^2 (n_1 + 1)}{(1 + \tilde{s}^2 |d_r E_r + d_\phi E_\phi + d_z E_z|^2)^2} \times \left[ \hat{\phi} \frac{\partial}{\partial r} |d_r E_r + d_\phi E_\phi + d_z E_z|^2 - \hat{r} \frac{1}{r} \frac{\partial}{\partial \phi} |d_r E_r + d_\phi E_\phi + d_z E_z|^2 \right]$$

# Vortex structures

HE11 modes:



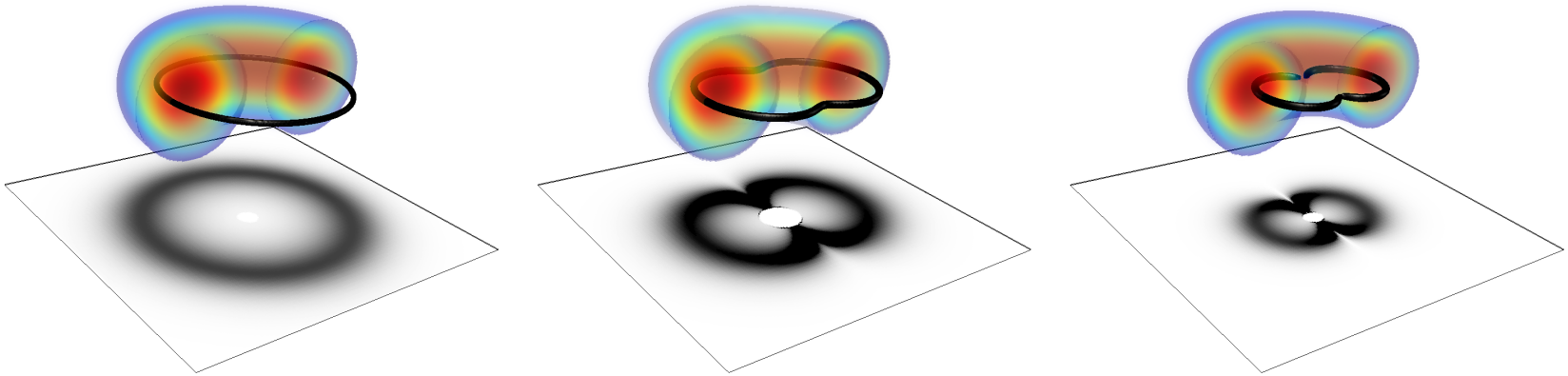
Circular



Linear

# Vortex structures

HE11 modes:

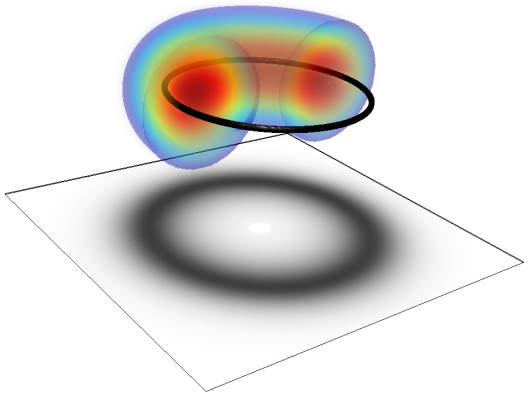


Circular Elliptic Linear

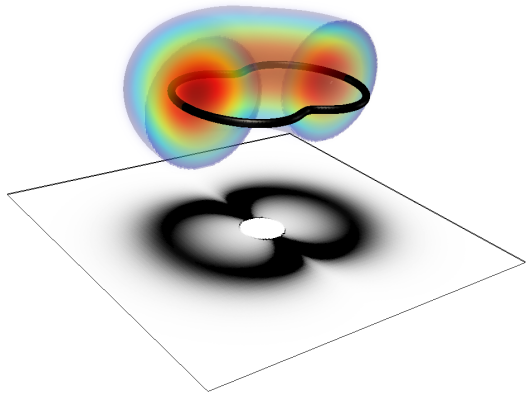
- Transition with linear polarization

# Vortex structures

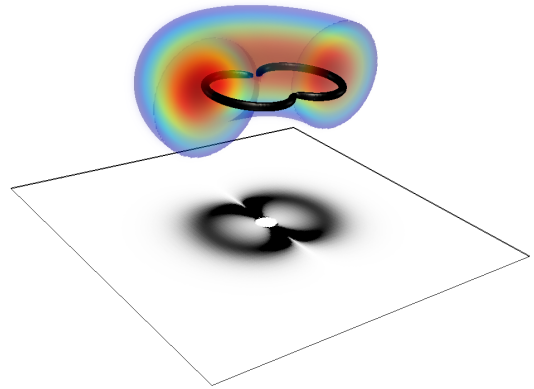
HE11 modes:



Circular

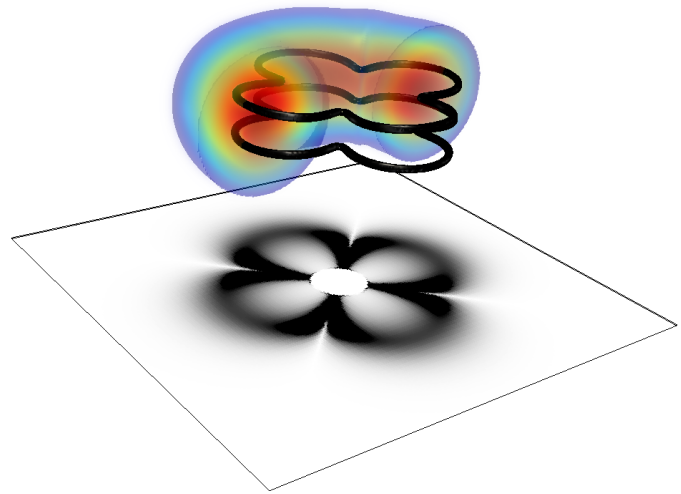


Elliptic



Linear

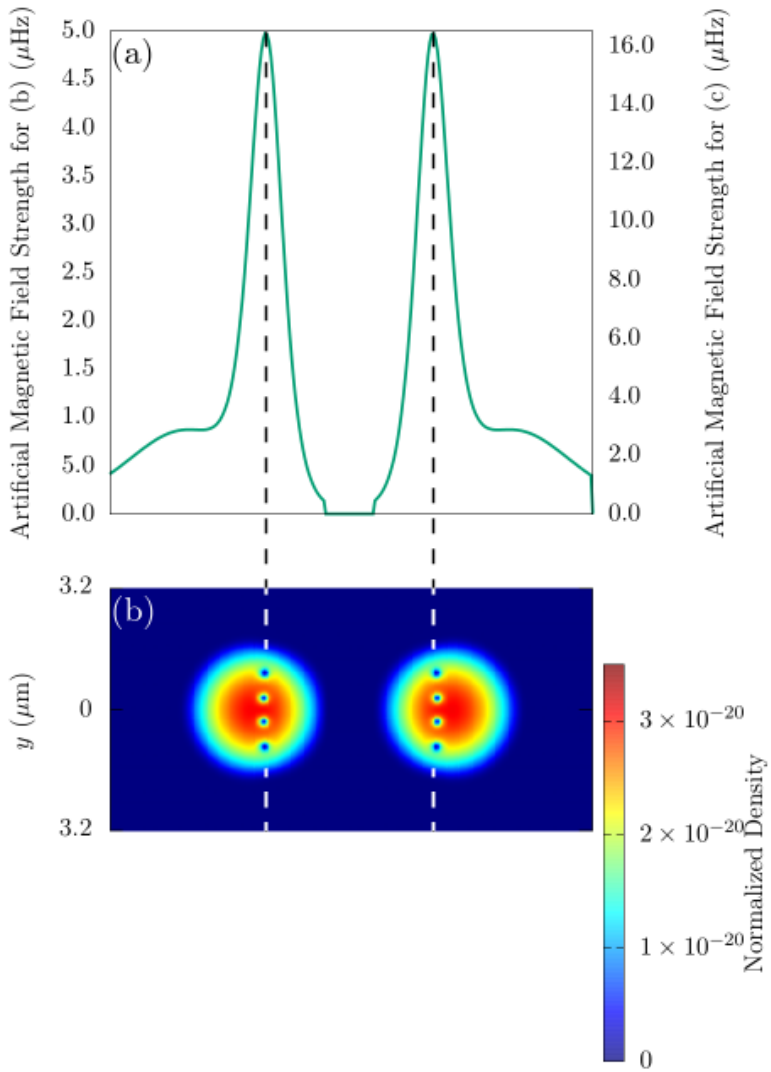
- Transition with linear polarization
- Highly controllable vortex structures that align to **B**



HE21 Linear

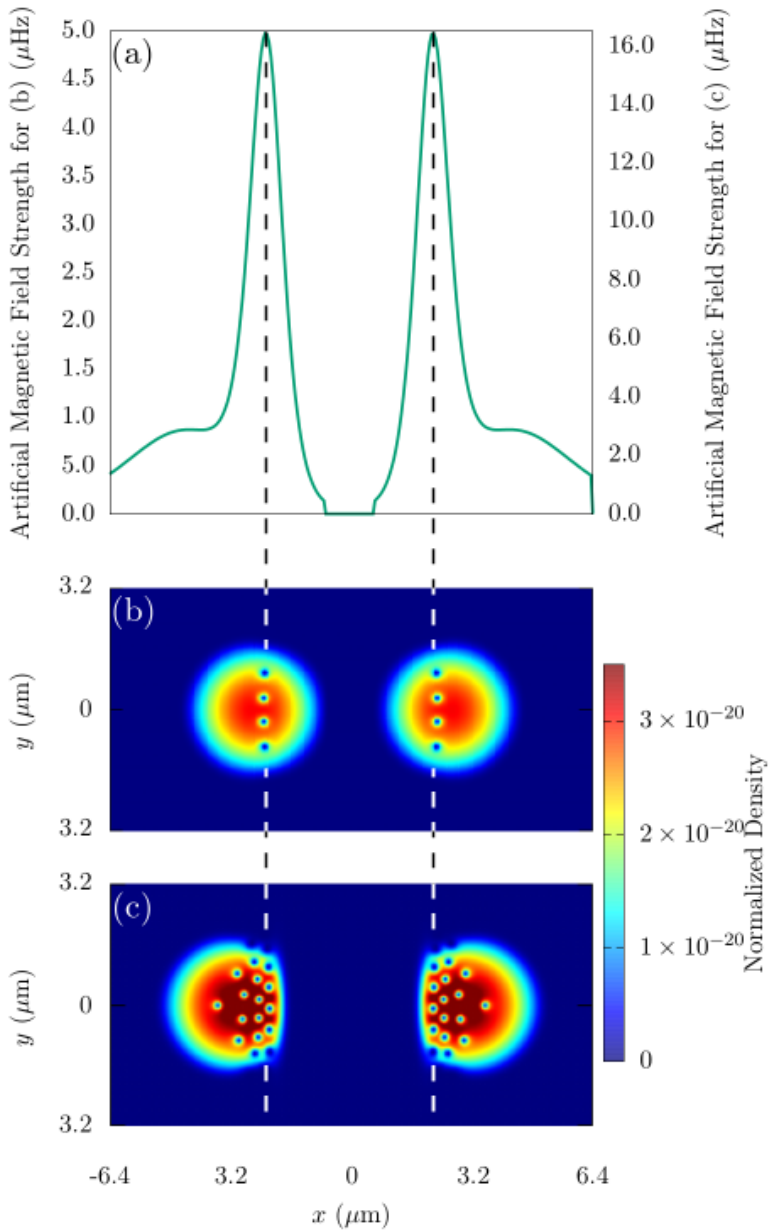
# Vortex ring lattices

- With low magnetic field, vortices follow magnetic field



# Vortex ring lattices

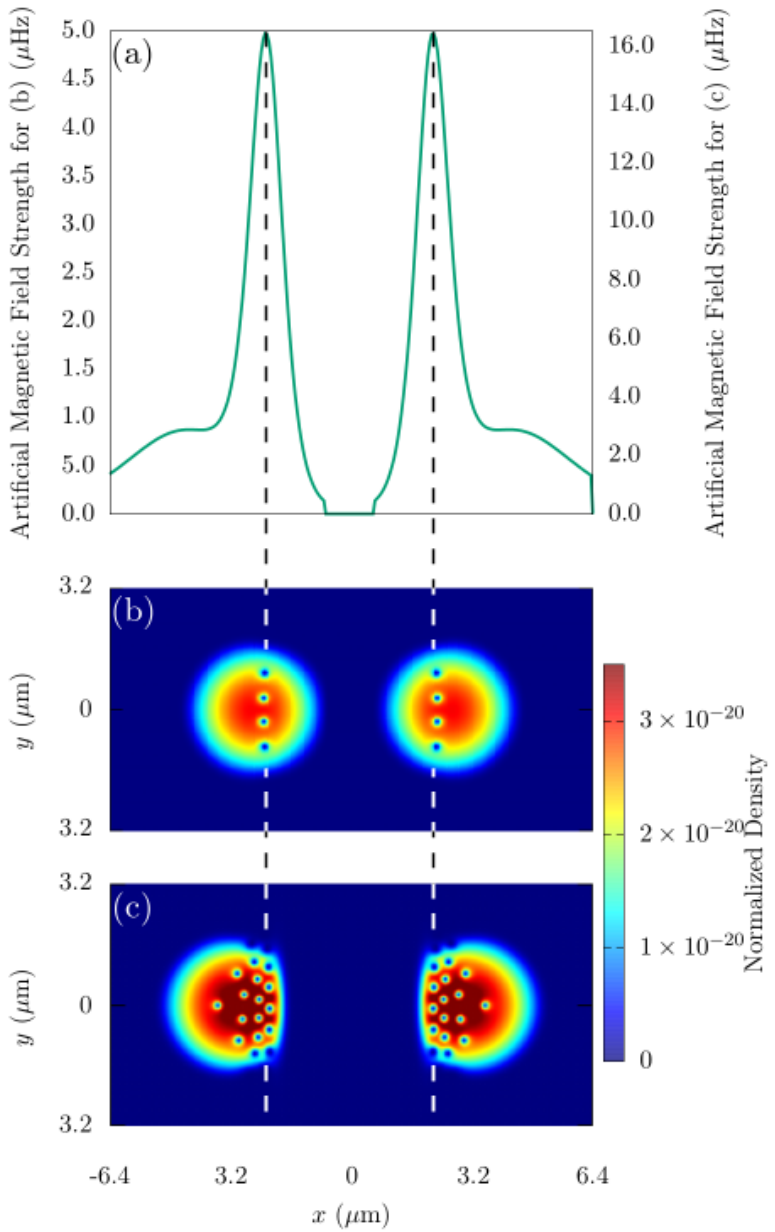
- With low magnetic field, vortices follow magnetic field
- With high magnetic field, they form a triangular lattice



# Vortex ring lattices

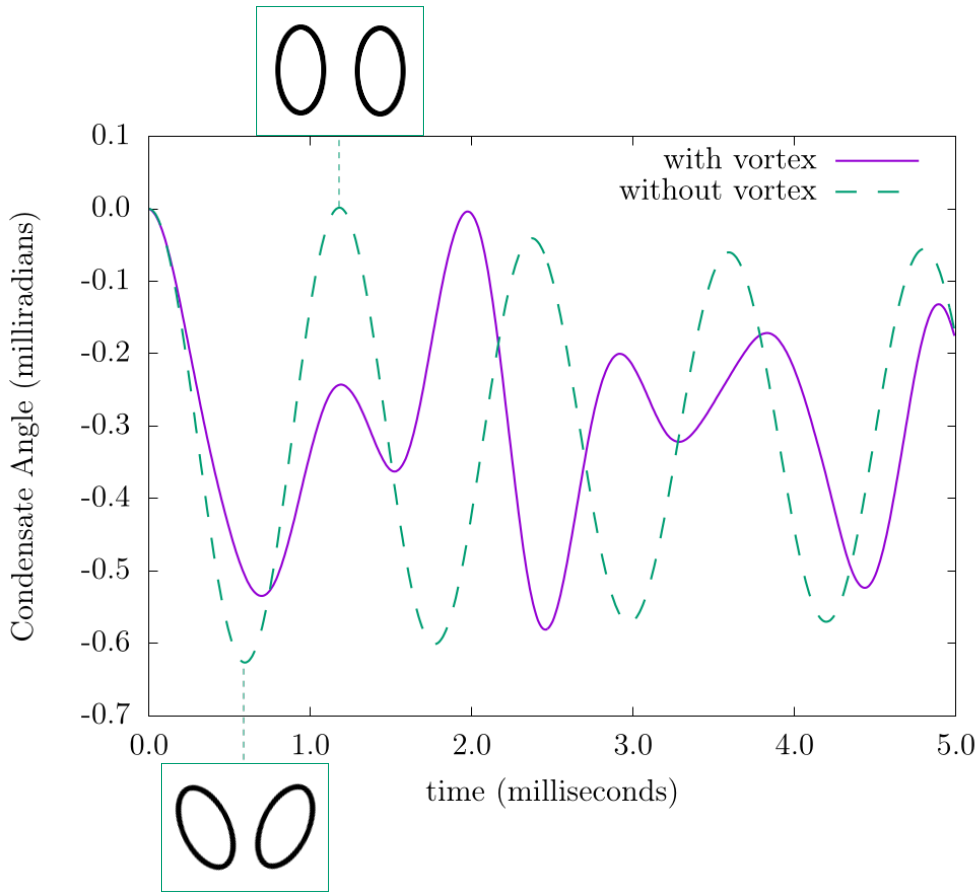
- With low magnetic field, vortices follow magnetic field
- With high magnetic field, they form a triangular lattice

## Vortex detection?



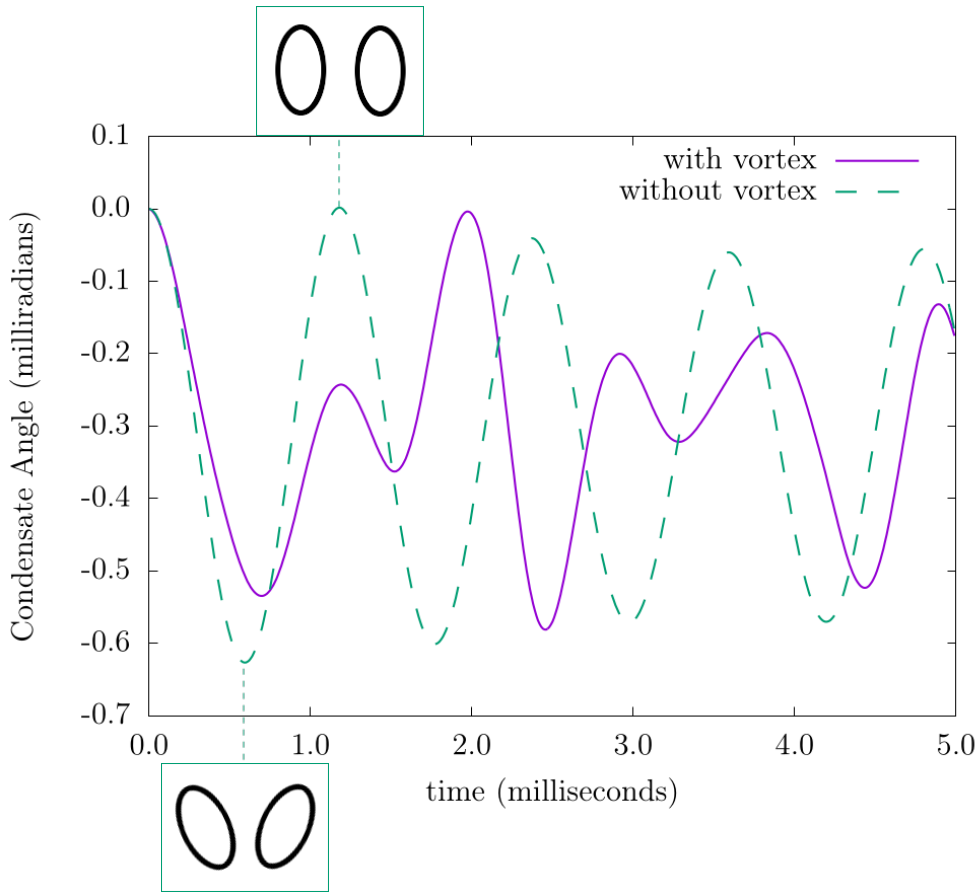
# Scissors modes

- Vortex detection in 3D is difficult
- Scissors modes provide different oscillations with vortices



# Scissors modes

- Vortex detection in 3D is difficult
- Scissors modes provide different oscillations with vortices
- Relies on elliptic-toroidal geometry



# Intermittent summary

Physics

- This system can generate, control, and detect vortex structures in a BEC around a nanofiber
- Dynamic simulations are underway!

# Intermittent summary

Physics

- This system can generate, control, and detect vortex structures in a BEC around a nanofiber
- Dynamic simulations are underway!

Computer Science

- Dynamic simulations require a large amount of fileIO
- Scaling to a larger grid requires multiple GPUs

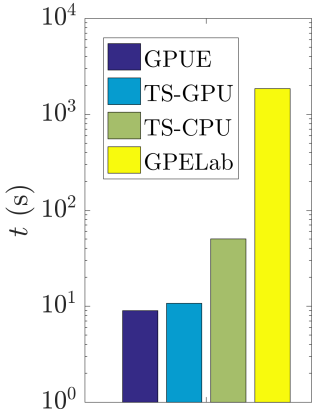
# Overall conclusions

# Overall conclusions

I developed GPUE and GPU methods for

- Expression trees
- Vortex tracking

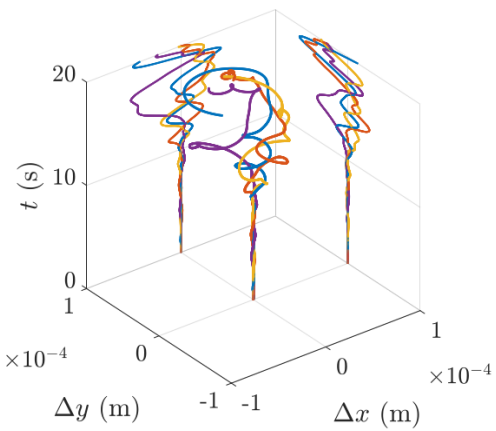
$$\mathrm{GP}\begin{pmatrix}\hat{U}\\ E\end{pmatrix}$$



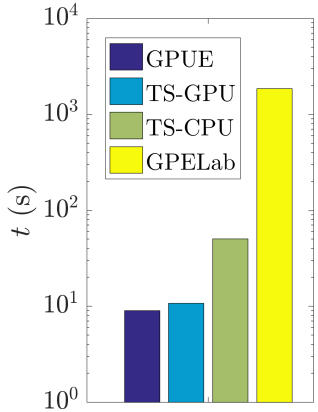
# Overall conclusions

I developed GPUE and GPU methods for

- Expression trees
- Vortex tracking




$$\mathrm{GP}\left(\hat{U}_E\right)$$




I have also shown...

Chaos can be found in few-vortex systems

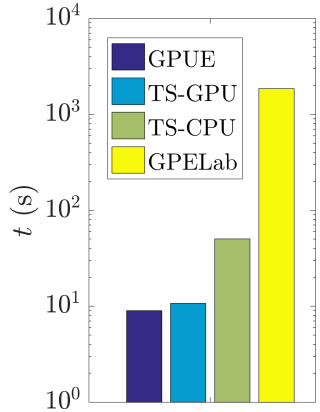
- Chaos is accelerated by collisions
- Highly controllable initial conditions

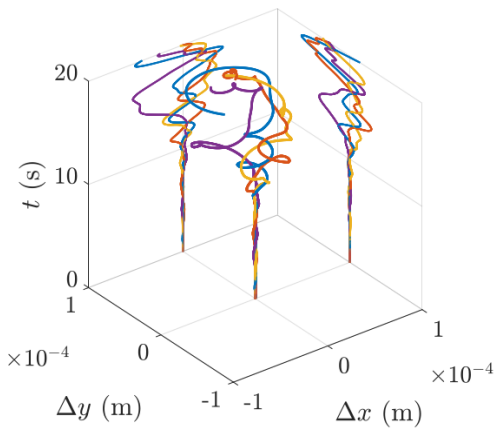
# Overall conclusions

I developed GPUE and GPU methods for

- Expression trees
- Vortex tracking

$$\mathrm{GP}\left(\begin{matrix}\hat{U}\\ E\end{matrix}\right)$$



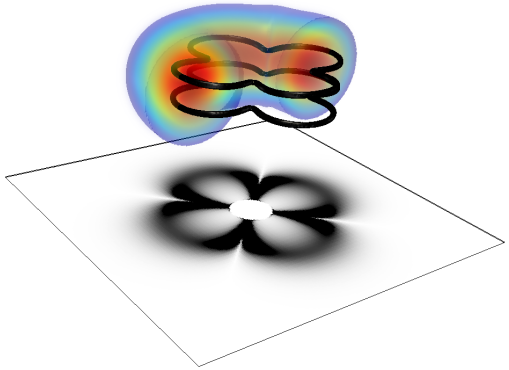

I have also shown...

Chaos can be found in few-vortex systems

- Chaos is accelerated by collisions
- Highly controllable initial conditions

3D vortex structures can be controlled, generated, and detected

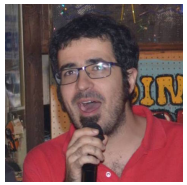
- Vortex ring lattice
- Scissors mode



# Future directions

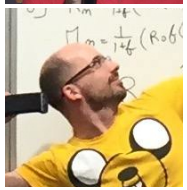
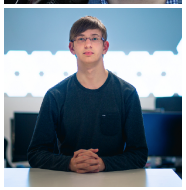
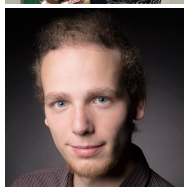
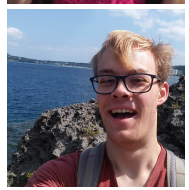
| **Physics: understanding superfluid vortices** | **Computer Science: SSFM on GPUs** |
| --- | --- |
| 3D vortex dynamics | Distributed Transpose |
| Vortex knots | GPUE.jl |
| Vortex turbulence | Expression tree extensions |
| Multicomponent simulations | Compression methods |

# People








Organizations:

OIST

JAPAN SOCIETY FOR THE PROMOTION OF SCIENCE · FOUNDED 1932 ·

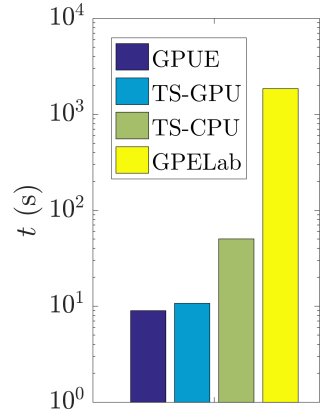
algorithm-archive.org

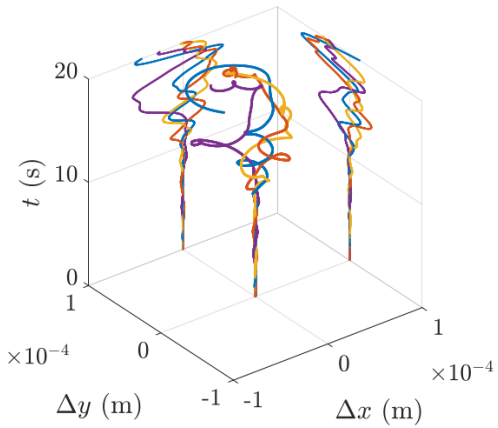
# Overall conclusions

GPU computing is fast

- Distributed transpose
- Expression trees

$$\mathrm{GP}\begin{pmatrix}\hat{U}\\ E\end{pmatrix}$$



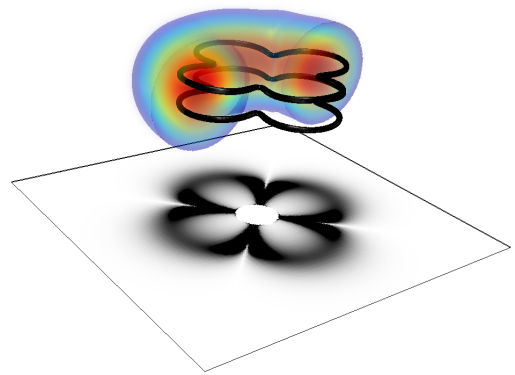




We can create chaotic systems

- Chaos is accelerated by collisions
- Highly controllable initial conditions

We can control vortex structures

- New vortex structures
- Vortex detection



# Quantum state engineering

# Quantum optimal control

Overall goal: optimize cost function by twiddling control parameters

# Quantum optimal control

Overall goal: optimize cost function by twiddling control parameters

- Many known methods, such as **gradient descent**, **genetic algorithms**, and **machine learning**

# Quantum optimal control

Overall goal: optimize cost function by twiddling control parameters

- Many known methods, such as **gradient descent**, **genetic algorithms**, and **machine learning**
- For quantum systems, cost function is often the fidelity:

$$\mathcal{F} = |\langle\psi|\phi\rangle|^2$$

which required re-simulation every time a control parameter is changed

# Quantum optimal control

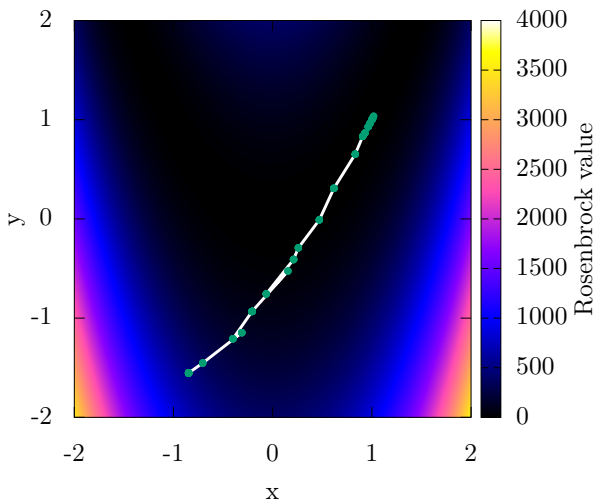
Overall goal: optimize cost function by twiddling control parameters

- Many known methods, such as **gradient descent**, **genetic algorithms**, and **machine learning**
- For quantum systems, cost function is often the fidelity:

$$\mathcal{F} = |\langle\psi|\phi\rangle|^2$$

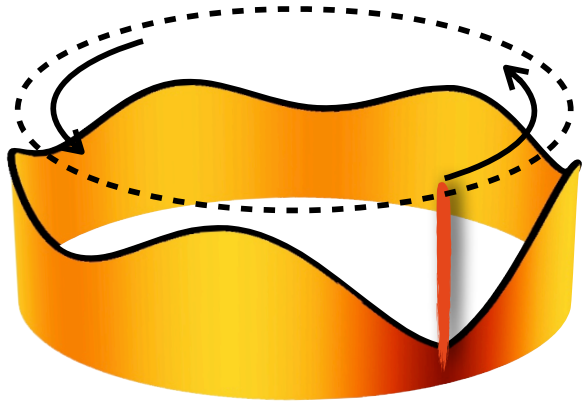
which required re-simulation every time a control parameter is changed

Nelder–Mead



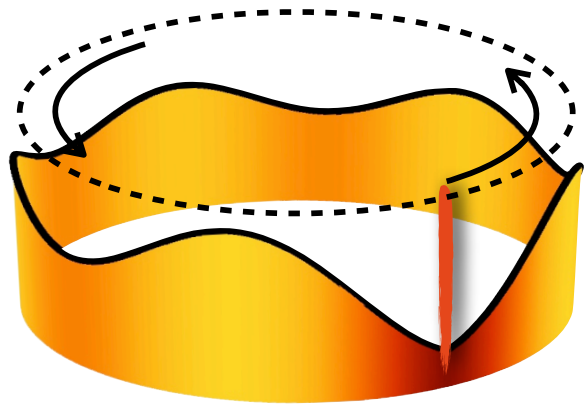
# Example Tonks–Girardeau gas system

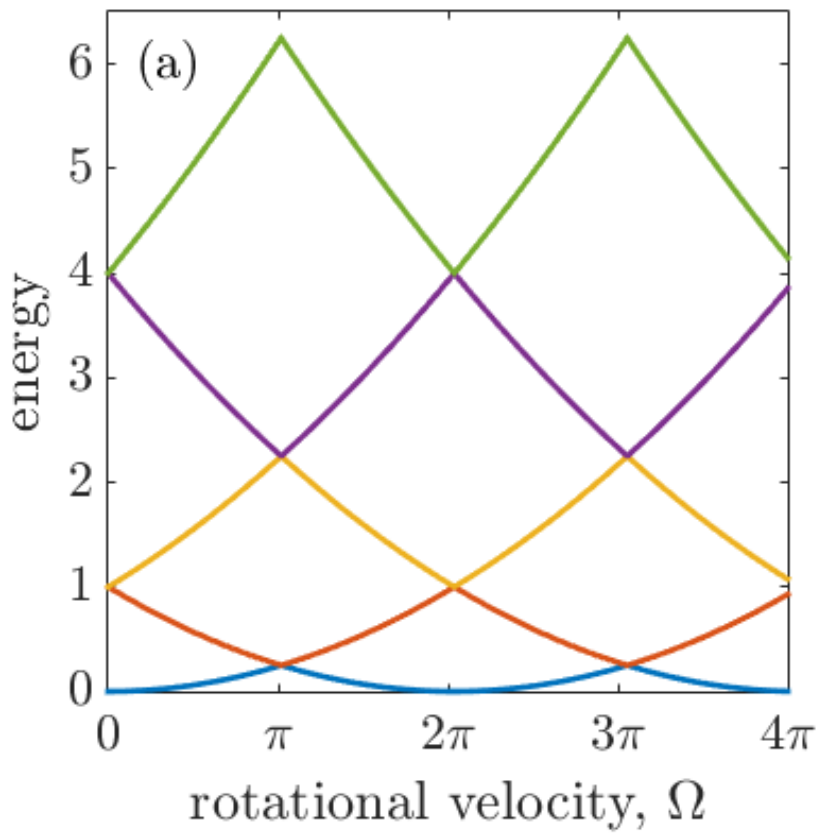
- NOON state: $|N, 0\rangle + |0, N\rangle$
- Tonks–Girardeau Gas:
  $g \to \infty$

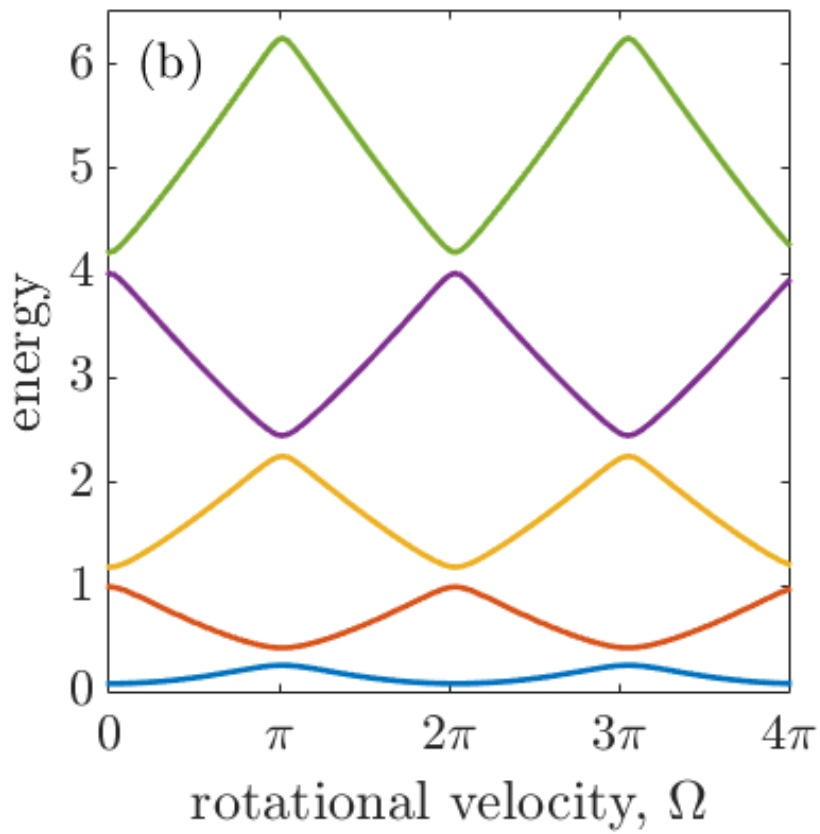


# Example Tonks–Girardeau gas system

- NOON state: $|N,0\rangle + |0,N\rangle$
- Tonks–Girardeau Gas: $g \to \infty$





# CRAB method

An example protocol is the Chopped RAndom Basis (CRAB) optimal control method where...

- A control parameter is modified with

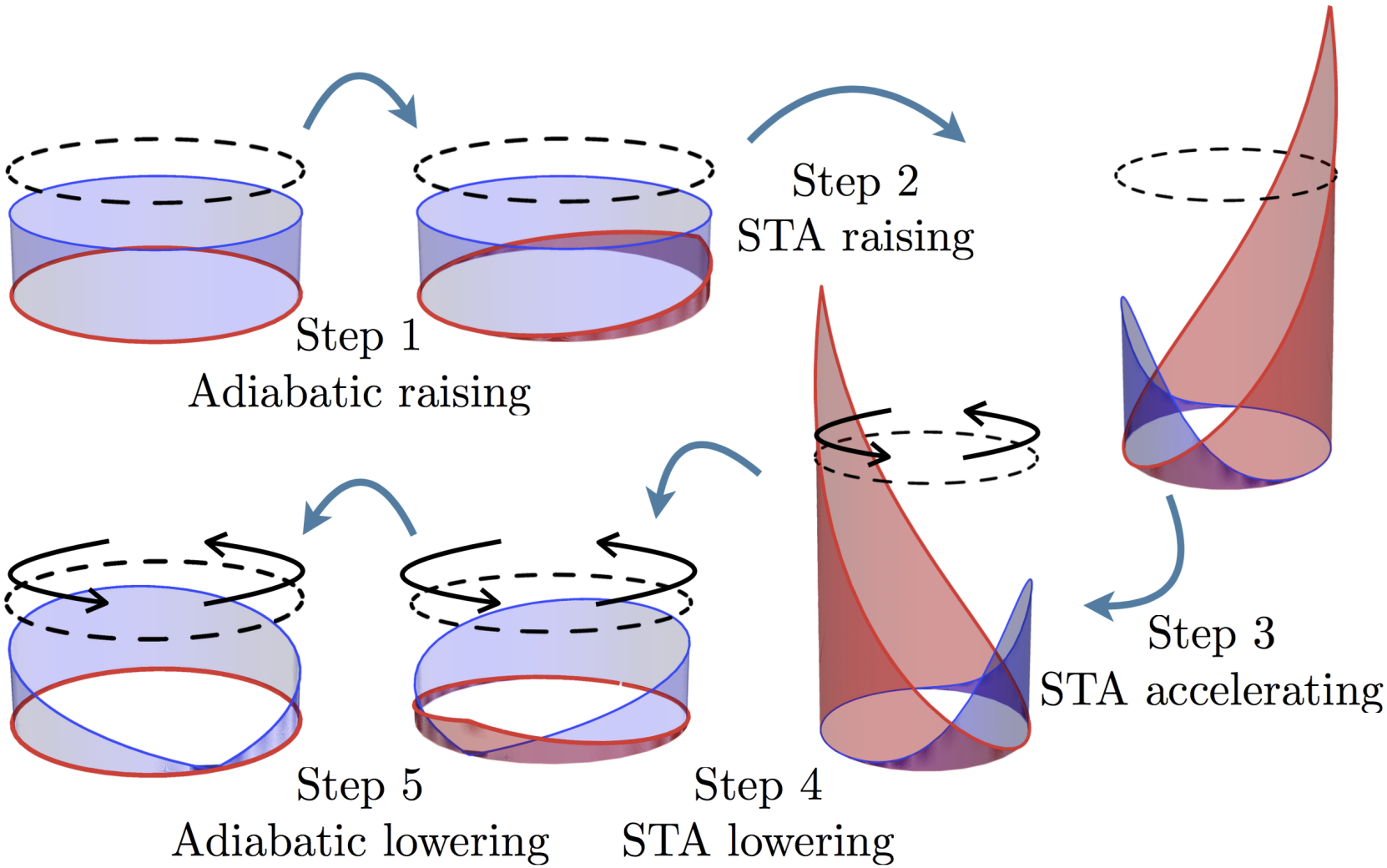
$$\Gamma^{\mathrm{CRAB}}(t) = \Gamma^{0}(t)\gamma(t)$$

  where

$$\gamma(t) = 1 + \frac{1}{\lambda(t)} \sum_{j=1}^{J} (A_j \sin(\nu_j t) + B_j \cos(\nu_j t))$$

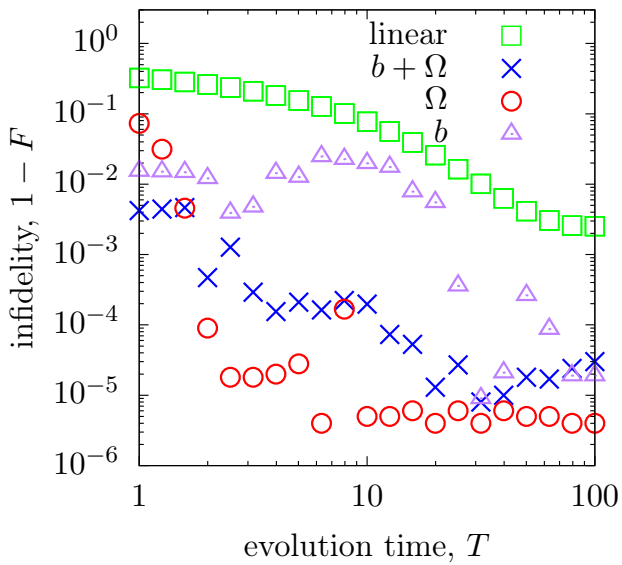
- Works if $\lim_{t\to 0} \lambda(t) = \lim_{t\to T} \lambda(t) = \infty$
- Creates a $3J$-dimensional space to optimize $(A, B, \nu)$

# STA protocol

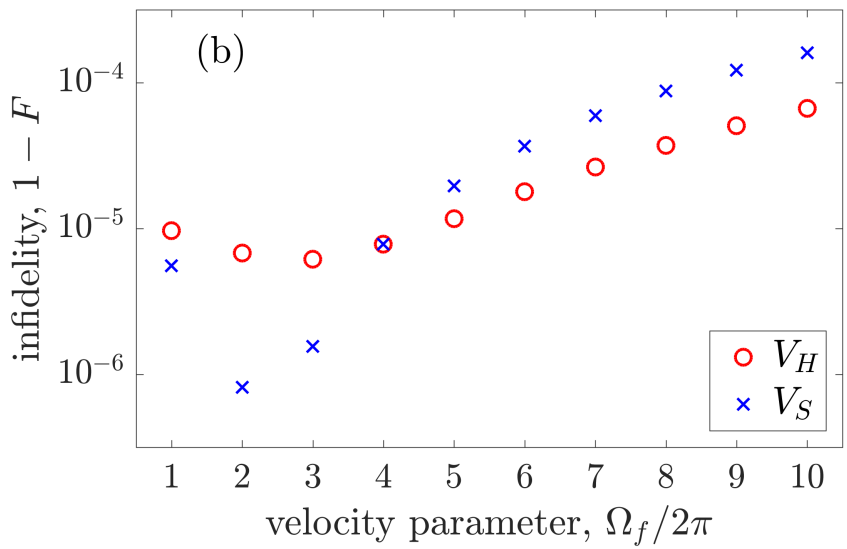


# NOON optimization

## Optimal control



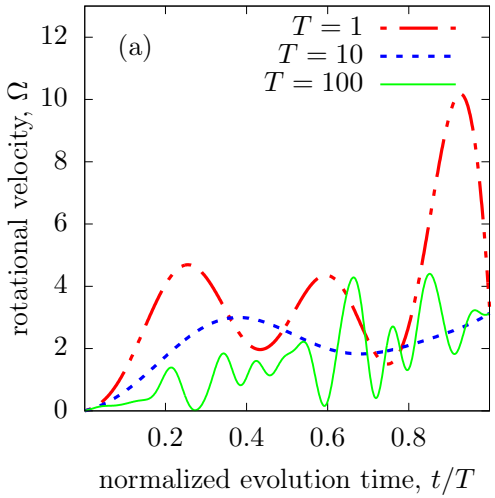

## STA




Please ask questions at the end!

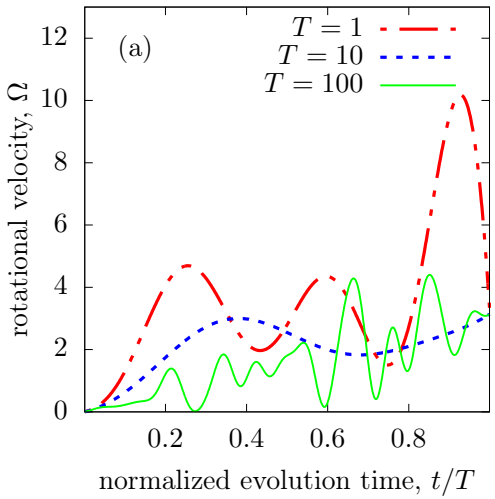
# Fidelities with optimal control

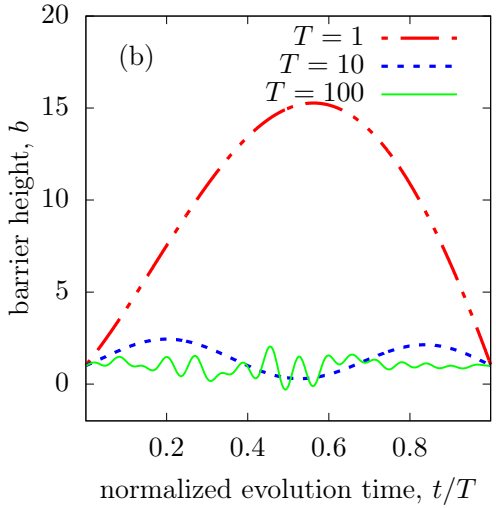
- Rotation ($\Omega(t)$)



# Fidelities with optimal control

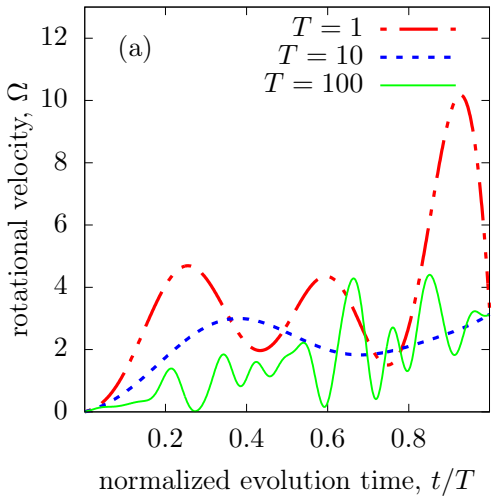
- Rotation ($\Omega(t)$)
- Barrier height ($b(t)$)

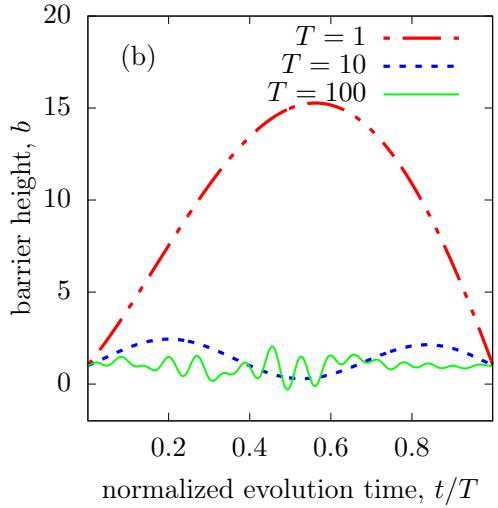


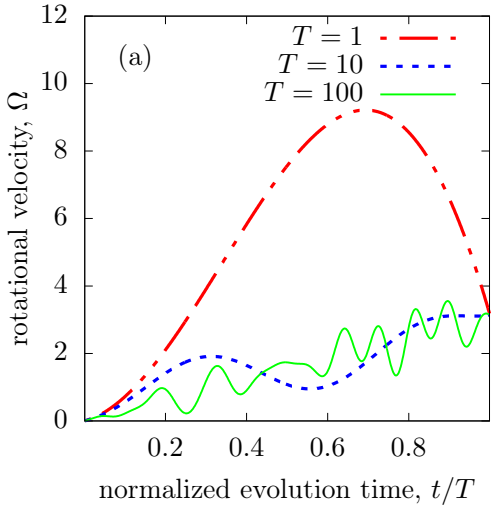



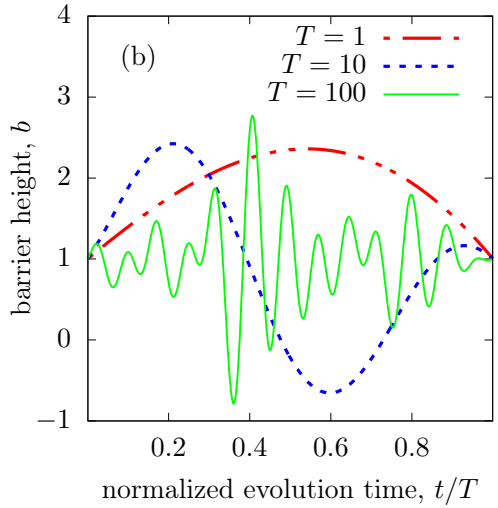
# Fidelities with optimal control

- Rotation ($\Omega(t)$)
- Barrier height ($b(t)$)
- Both



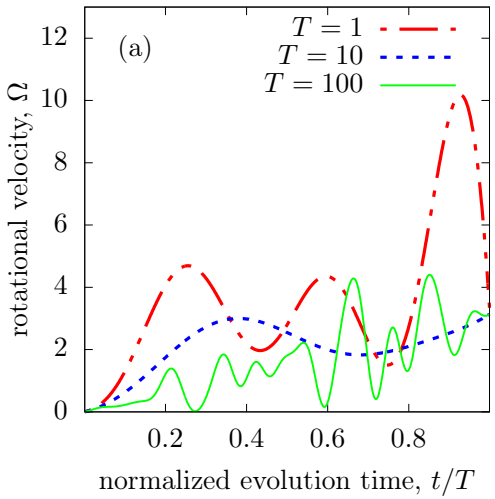



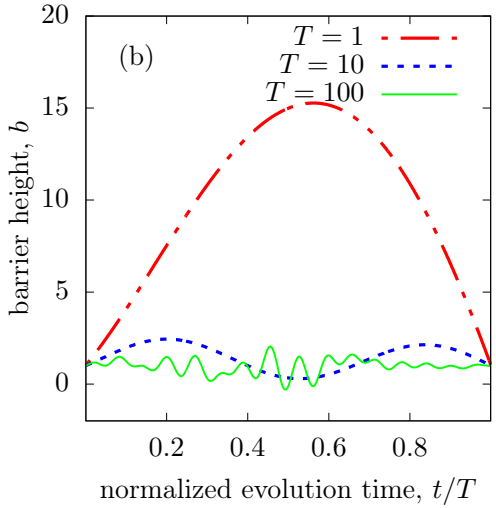



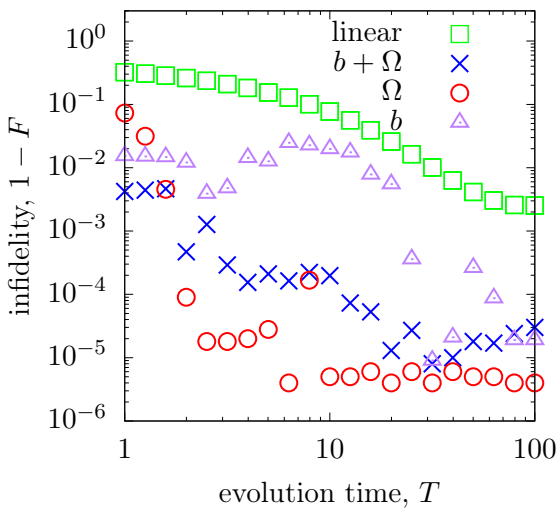



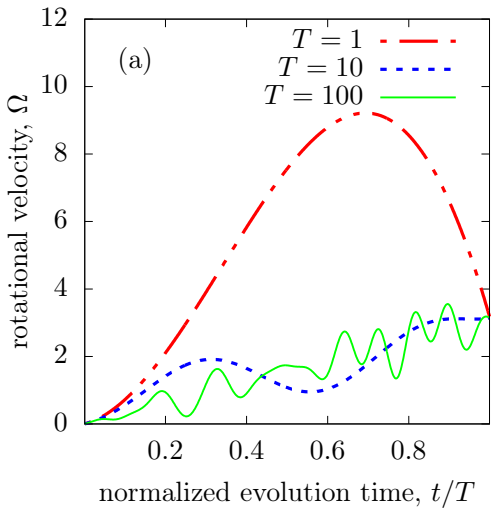
# Fidelities with optimal control

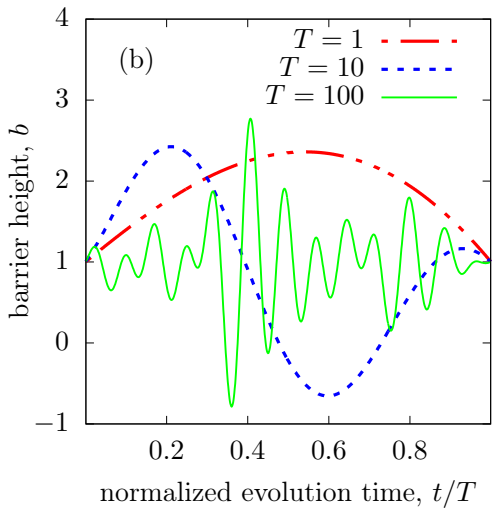
- Rotation ($\Omega(t)$)
- Barrier height ($b(t)$)
- Both



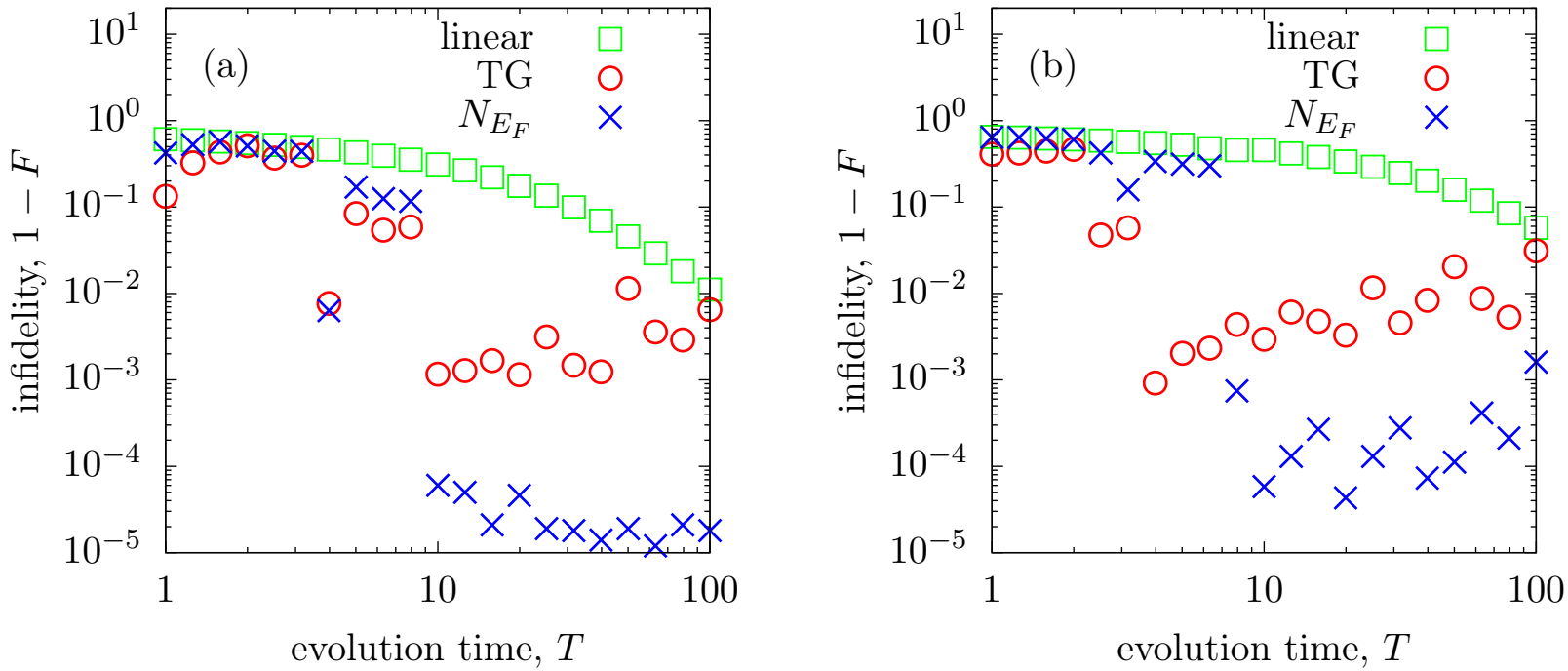



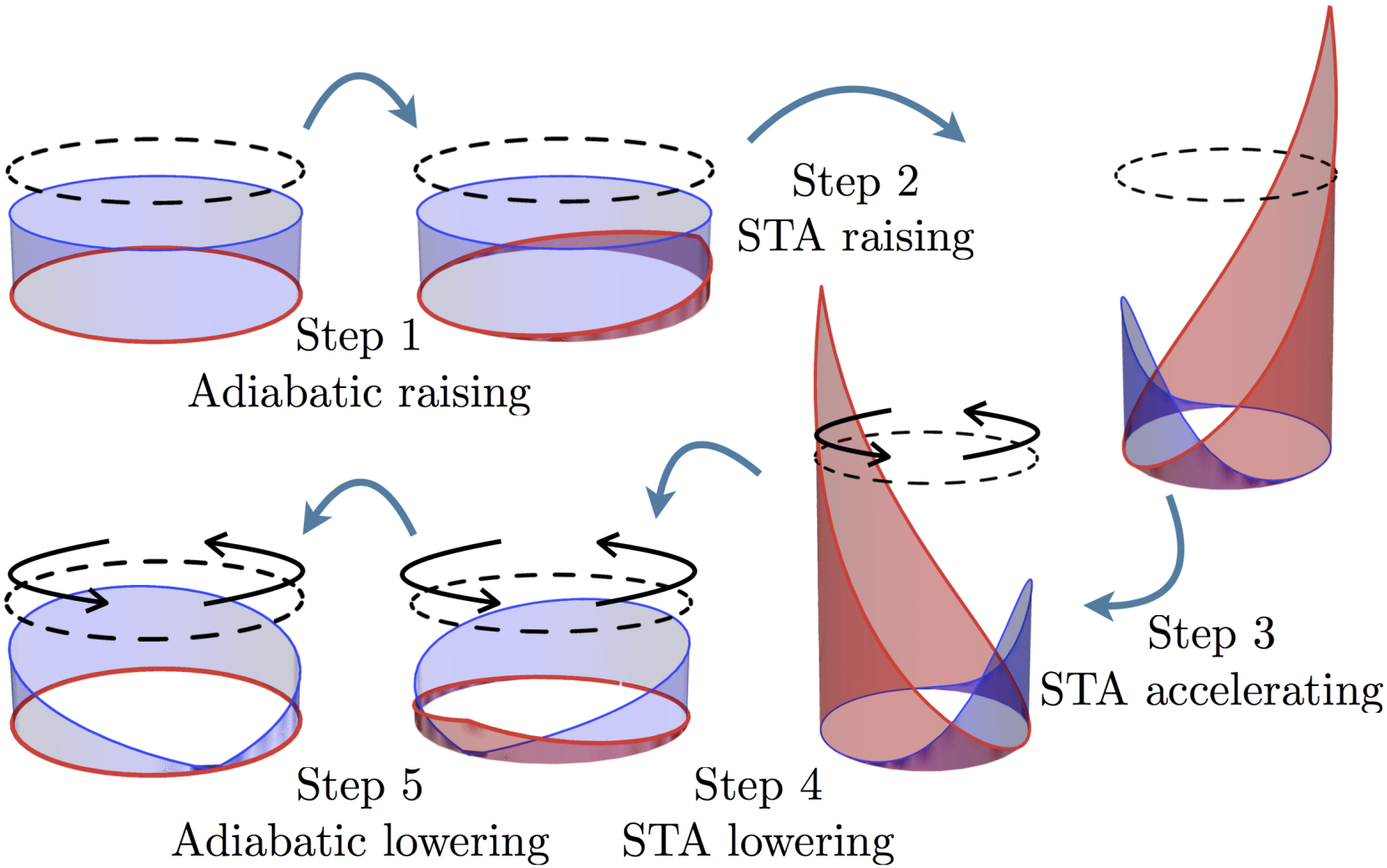



# NOON Optimization

Optimizations of NOON state generation with 3 and 5 particles

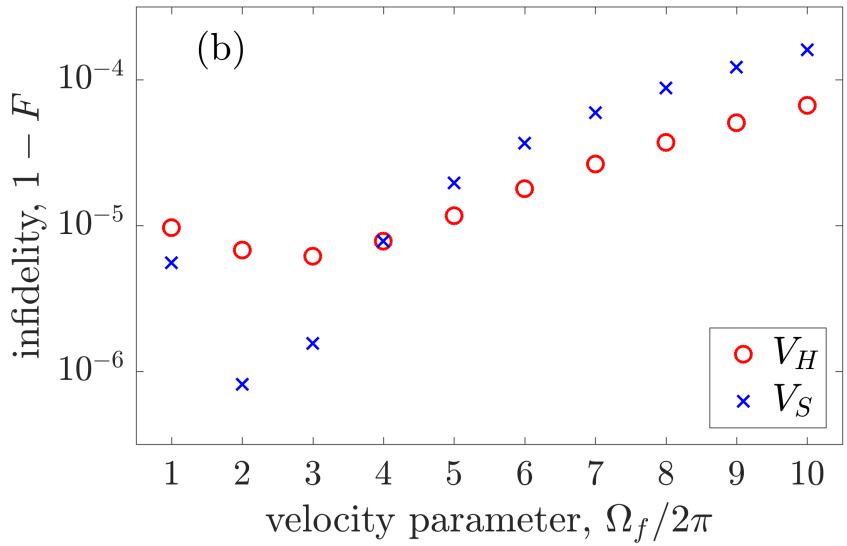


# STA protocol



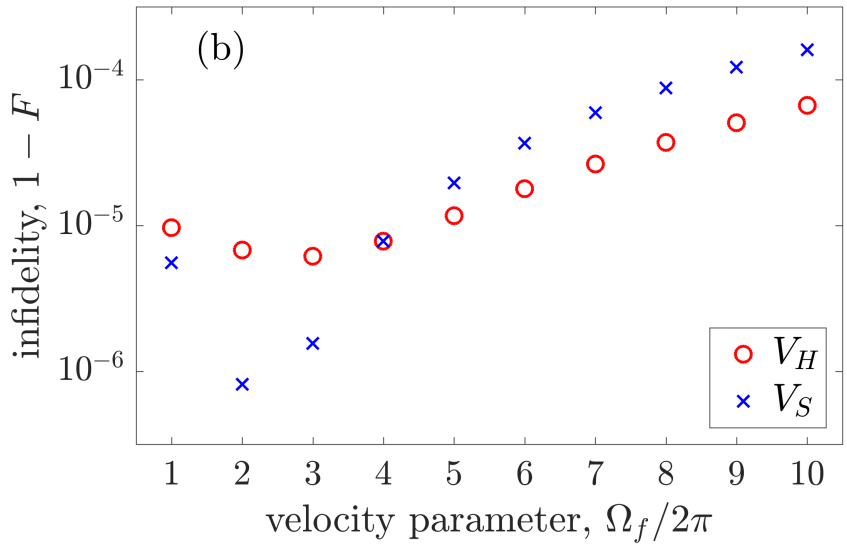
# STA fidelities

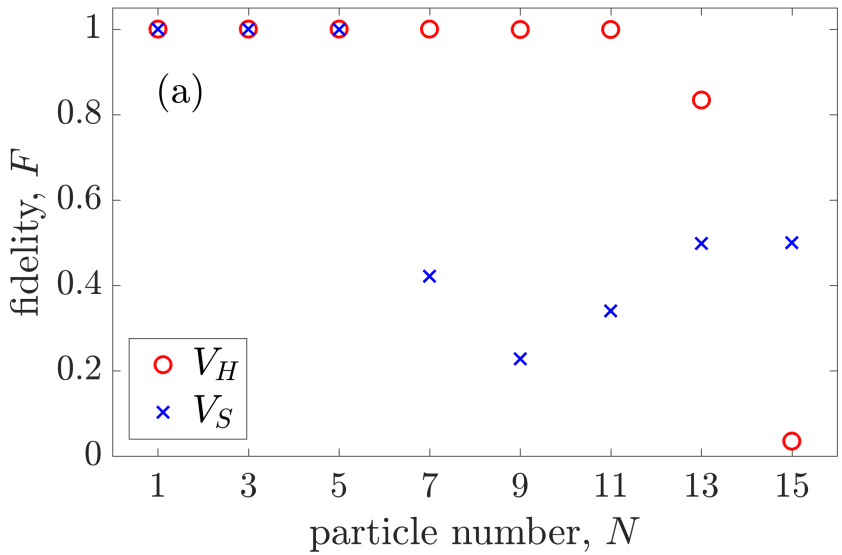
- Fidelities with rotation

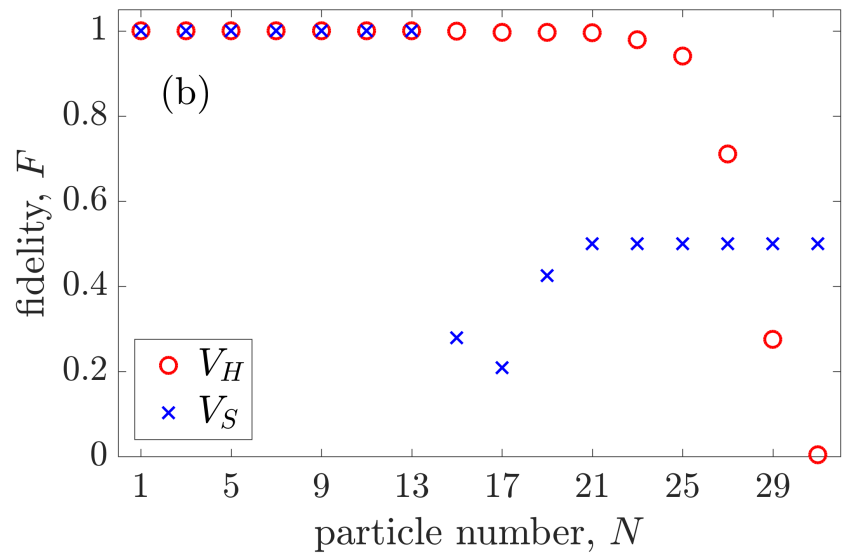


# STA fidelities

- Fidelities with rotation
- Fidelities with rotation of 100, 200 and higher particle number

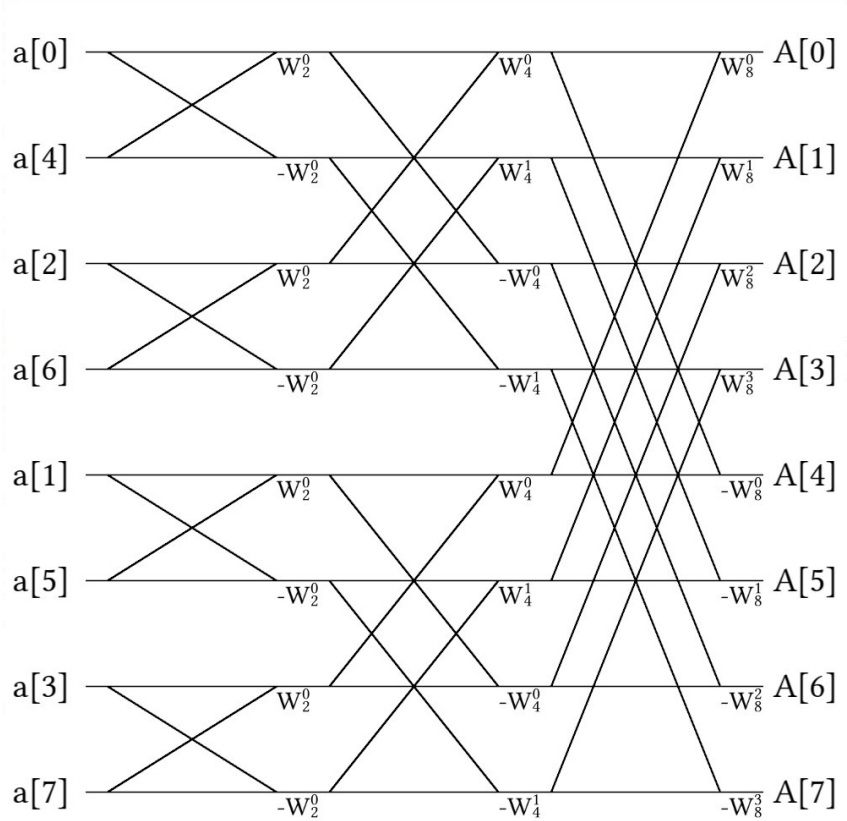


# Cooley-Tukey

- Recursively subdivides DFT into simple sums with twiddle factors
- Many known libraries, like FFTW, and CuFFT
- Hard to parallelize (note for later)



# Distributed transpose

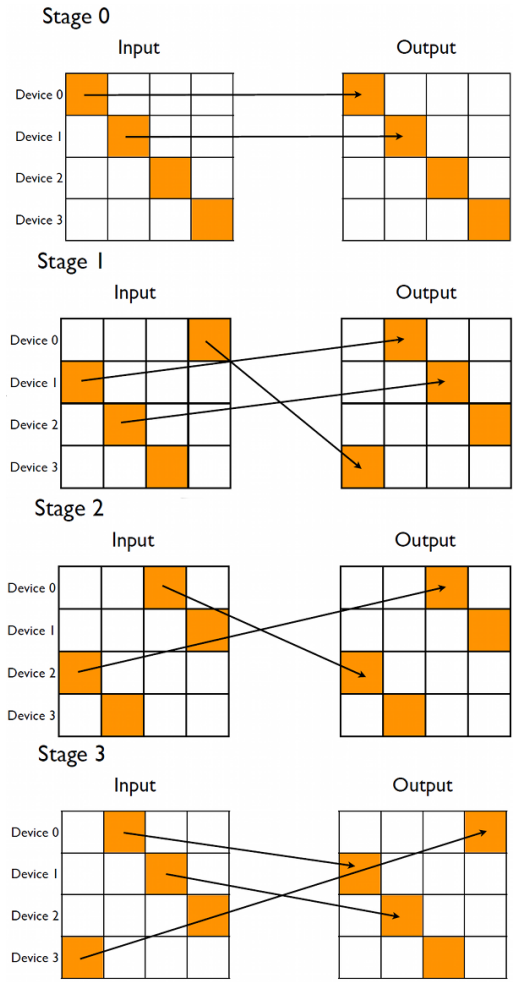
GPU transpose

- 2D out-of-place transpose $\approx$ copy
  (shared memory, coalescence, bank conflicts)
- In-place transposes are inefficient
- 3D permutations of arrays have 70% efficiency

# Distributed transpose

GPU transpose

- 2D out-of-place transpose ≈ copy
  (shared memory, coalescence, bank conflicts)
- In-place transposes are inefficient
- 3D permutations of arrays have 70% efficiency
- 2D distributed example




Ruetsch, G. and Fatica, M. 2013

# Strang splitting

$$\Psi(\mathbf{r}, t+dt) = \left[ e^{-\frac{i\hat{\mathcal{H}}_V dt}{2\hbar}} e^{-\frac{i\hat{\mathcal{H}}_p dt}{\hbar}} e^{-\frac{i\hat{\mathcal{H}}_V dt}{2\hbar}} \right] \Psi(\mathbf{r}, t) + \mathcal{O}(dt^3)$$

Taylor expansion of exponential:

$$e^{h(\mathbf{A}+\mathbf{B})} \approx \mathbf{I} + h(\mathbf{A}+\mathbf{B}) + frac12h^2(\mathbf{A}+\mathbf{B})^2$$

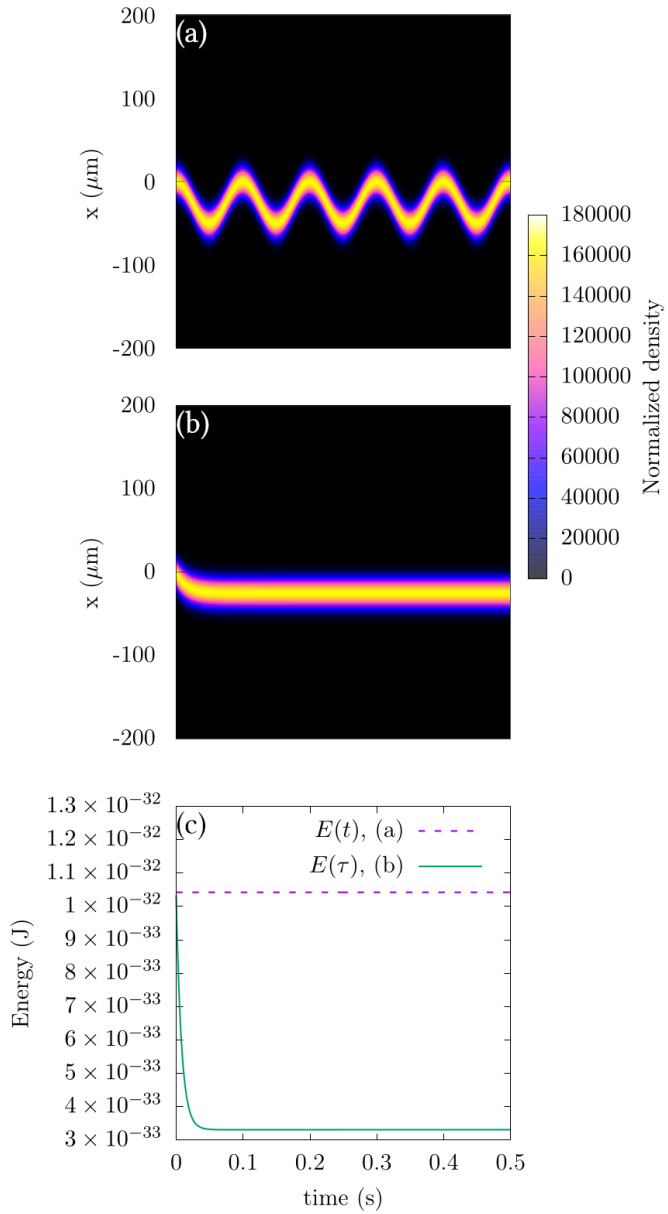
Last term:

$$\frac{1}{2}h^2(\mathbf{A}+\mathbf{B})^2 = \frac{1}{2}\left(\mathbf{A}^2 + \mathbf{AB} + \mathbf{BA} + \mathbf{B}^2\right)$$

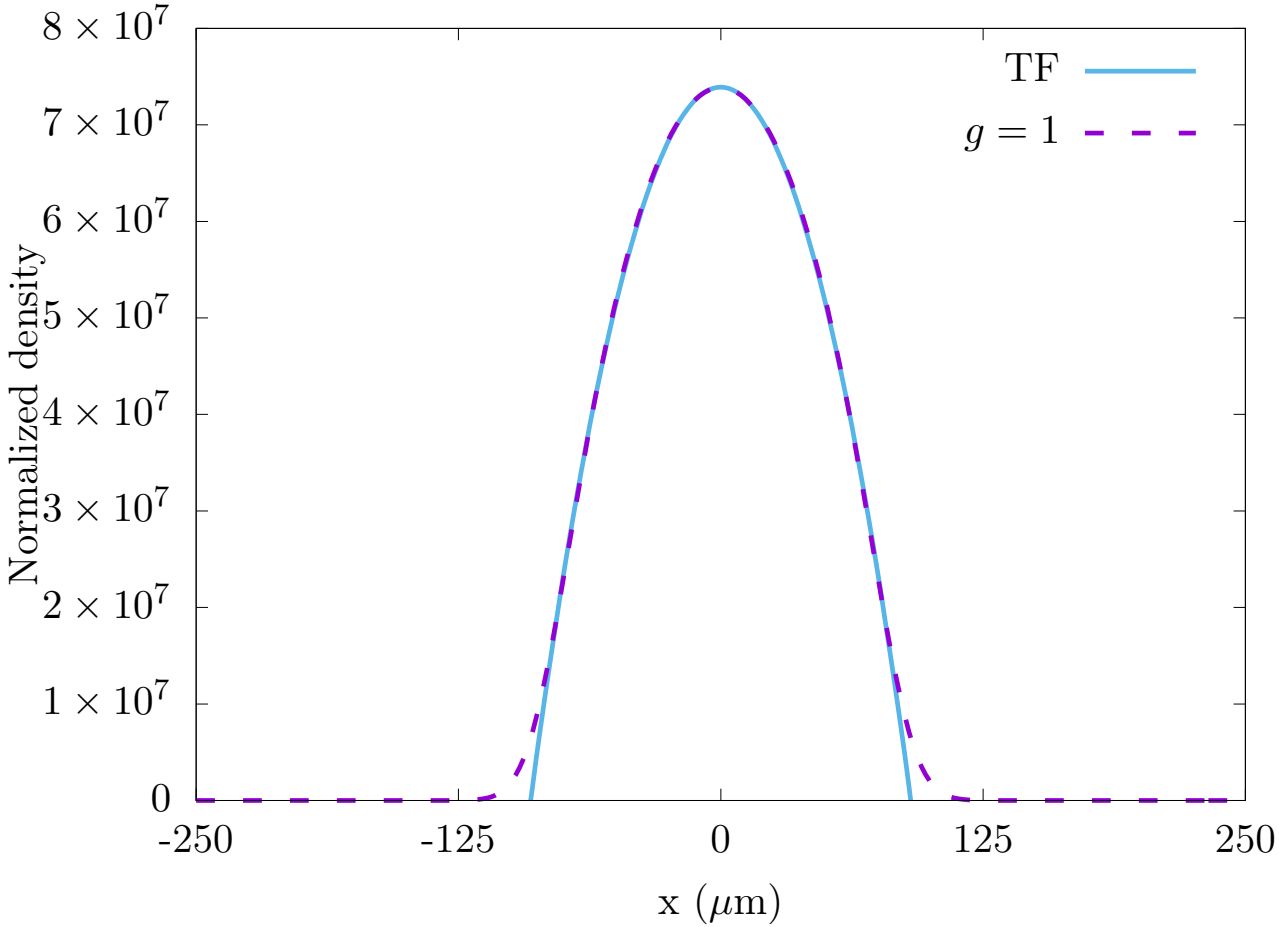
Symmetric splitting:

$$e^{h(\mathbf{A}+\mathbf{B})} \approx e^{h\mathbf{A}/2} e^{h\mathbf{B}} e^{h\mathbf{A}/2}$$

# Evolution of SHO



# Thomas–Fermi




$$R_{\mathsf{TF}} = \sqrt{\frac{2\mu}{m\omega_i^2}}$$