Cython for HPC

Niall Moran

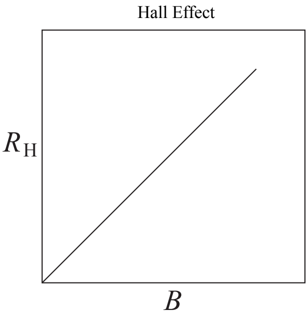
July, 2016

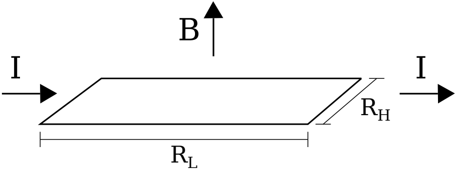
Overview

- Motivation
- Why (not) Python?
- Cython
- Examples

Hall Effect

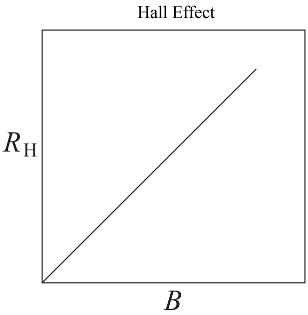
- Edwin Hall (1879)
- Magnetic field induces Hall current

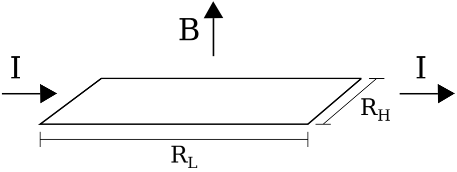




Hall Effect

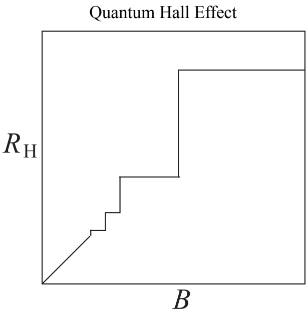
- Edwin Hall (1879)
- Magnetic field induces Hall current

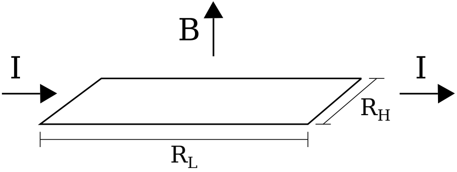




(Integer) Quantum Hall Effect

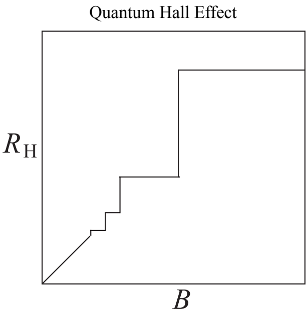
- von Klitzing (1980)
- Lower temperature, plateaus appear
- Quantization of conductance

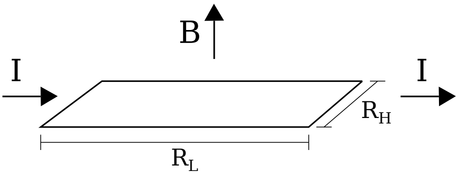




(Integer) Quantum Hall Effect

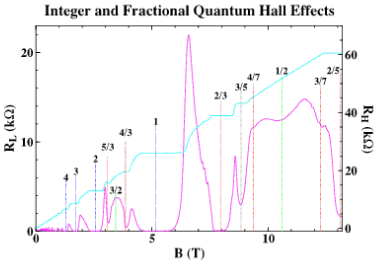
- von Klitzing (1980)
- Lower temperature, plateaus appear
- Quantization of conductance

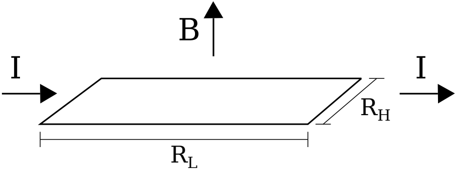




Fractional Quantum Hall Effect

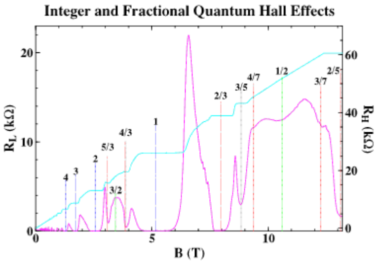
- Tsui, Stormer and Gossard (1982)
- Plateaus at fractional filling
- Anyonic excitations

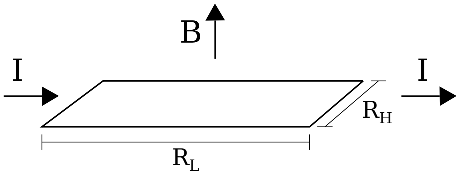




Fractional Quantum Hall Effect

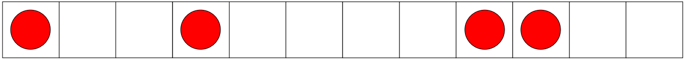
- Tsui, Stormer and Gossard (1982)
- Plateaus at fractional filling
- Anyonic excitations





Modelling

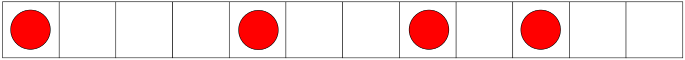
- Finite system N particles and N_ϕ flux with $\nu = \frac{N}{N_\phi}$
- Decompose into magnetic orbitals



Ways to fill available orbitals

Modelling

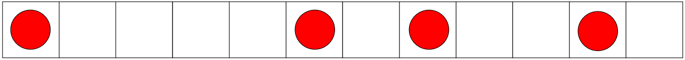
- Finite system N particles and N_ϕ flux with $\nu = \frac{N}{N_\phi}$
- Decompose into magnetic orbitals



Ways to fill available orbitals

Modelling

- Finite system N particles and N_ϕ flux with $\nu = \frac{N}{N_\phi}$
- Decompose into magnetic orbitals

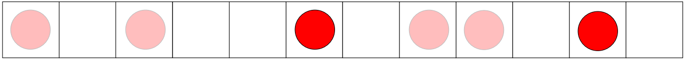


Ways to fill available orbitals

$$\binom{N_\phi}{N} \approx f(\nu)^N$$

Modelling

- Finite system N particles and N_ϕ flux with $\nu = \frac{N}{N_\phi}$
- Decompose into magnetic orbitals



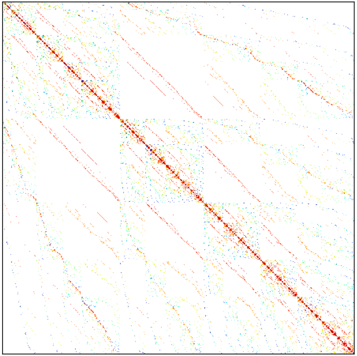
Ways to fill available orbitals

$$\binom{N_\phi}{N} \approx f(\nu)^N$$

Wave-function is complex vector of this dimension!

Computations

- Linear algebra
- Markov-Chain Monte Carlo
- Tensor network methods



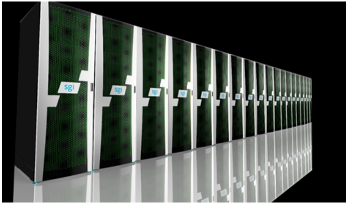
6 particles with 18 flux

Diminishing returns

10-12 electrons

Diminishing returns

10-12 electrons



16-20 electrons

Why (not) use Python

Cons

- Slow
- GIL
- Resource usage

Why (not) use Python

Cons

- Slow
- GIL
- Resource usage

Pros

- Fast development, debugging
- Amount of packages available
- Plotting
- Good glue

Cython

Cython C-Extensions for Python

- Superset of python
- Compiles to C code
- Best of both worlds
- Good for wrapping existing C code

Provides speed increase by

- Compiling
- Providing explicit types and functions
- Can release the GIL

Hello World

Create hello.pyx containing

```
print("Hello world!")
```

and setup.py containing

```
from distutils.core import setup
from Cython.Build import cythonize

setup(
    ext_modules = cythonize("hello.pyx")
)
```

then run on command line

```
$ python setup.py build_ext --inplace
```

Notebook

Load the cython extension

```
%load_ext Cython
```

Can now create cython blocks

```
%%cython
print("Hello world!")
```

Annotation

```
%%cython -a
print("Hello world!")
```

Demonstration

```
def mean_plain(A, N):
    sum = 0.0
    j = 0
    while j < N:
        sum += A[j]
        j+=1

    return sum / N
```

220ms for array of 1 000 000 elements

```
cimport numpy as np
cimport cython

@cython.boundscheck(False)
@cython.wraparound(False)
def mean_cython4(
        np.ndarray[np.float64_t, ndim=1, mode="c"] A,
        int N
    ):
    cdef double sum = 0.0
    cdef int j = 0

    while j < N:
        sum += A[j]
        j += 1

    return sum / N
```

1ms for array of 1 000 000 elements

Matrix multiplication benchmark

	80x80	1500x1500
	Units: MFLOPS	
Optimal layout		
Python	0.94	0.98
Cython	1.08	1.12
Added types	179	177
`boundscheck`/`wraparound`	770	692
`mode="c"`/`mode="fortran"`	981	787
BLAS `ddot` (ATLAS)	1282	911
Intel C	2560	1022
gfortran $A^T B$	1113	854
Intel Fortran $A^T B$	2833	1023
NumPy `dot`	3656	4757
Worst-case layout		
Python	0.94	0.97
`boundscheck`/`wraparound`	847	175
BLAS `ddot` (ATLAS)	910	183
gfortran AB^T	861	94
Intel Fortran AB^T	731	94

Conclusions and thank you

Summary

- Significant speedup
- Not always optimal
- Useful when cannot be vectorised (ODEs, MCMC)
- Can wrap standard C libraries
- Can disable the GIL

Further reading

- Cython website, (`http://cython.org/`)
- D. S. Seljebotn, *"Fast Numerical Computation with Cython"*, SciPy 2009. (`http://conference.scipy.org/proceedings/SciPy2009/paper_2/full_text.pdf`)
- S. Behnel et. al., *"Cython: The Best of Both Worlds"*, IEEE 2011. (`http://folk.uio.no/dagss/cython_cise.pdf`)