

2012 Basic Numerical Analysis

Further practical issues

Contents

- Post-processing and data visualization
- File input and output
- Final problem set: option pricing

Post-processing and data visualization

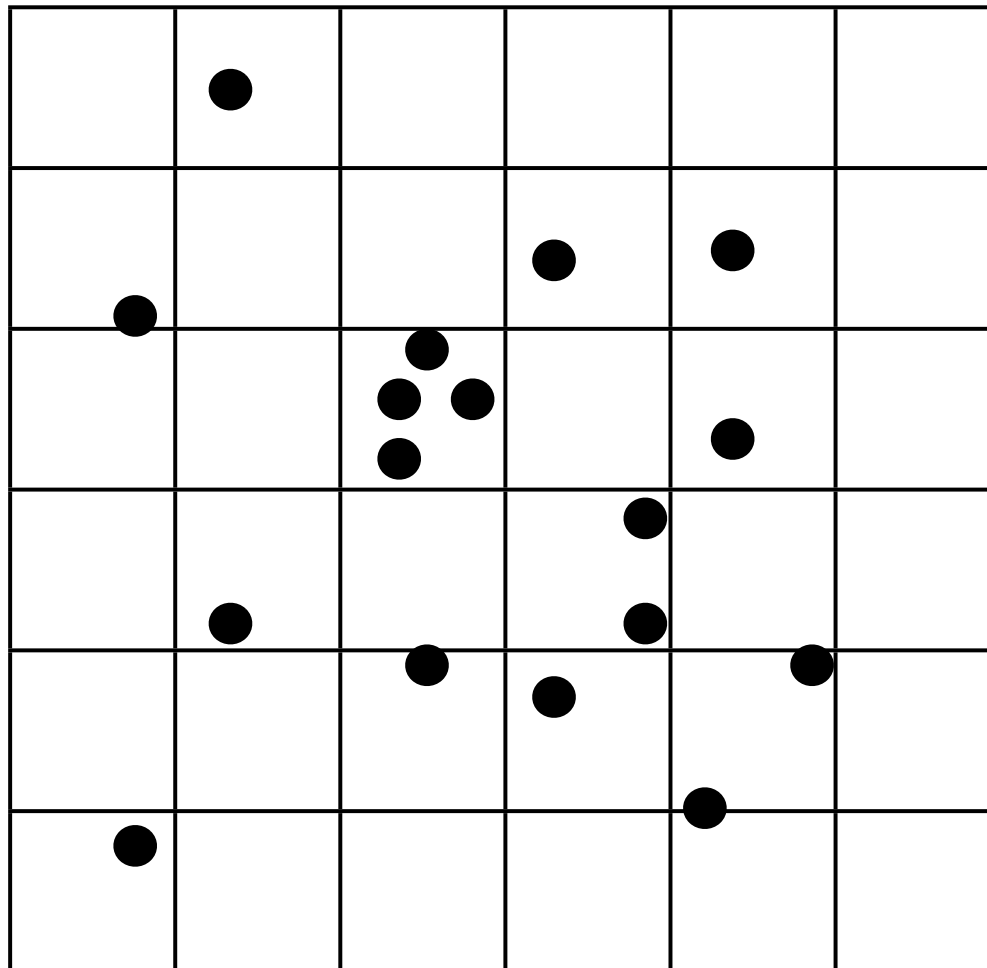
- Might not seem to be a main issue in high-performance computing....
- Data size can be very large:
10¹⁰ particles, grids, > 1TB.
Won't fit on your laptop.
- Data handling and visualization are big problems by themselves

Post-processing

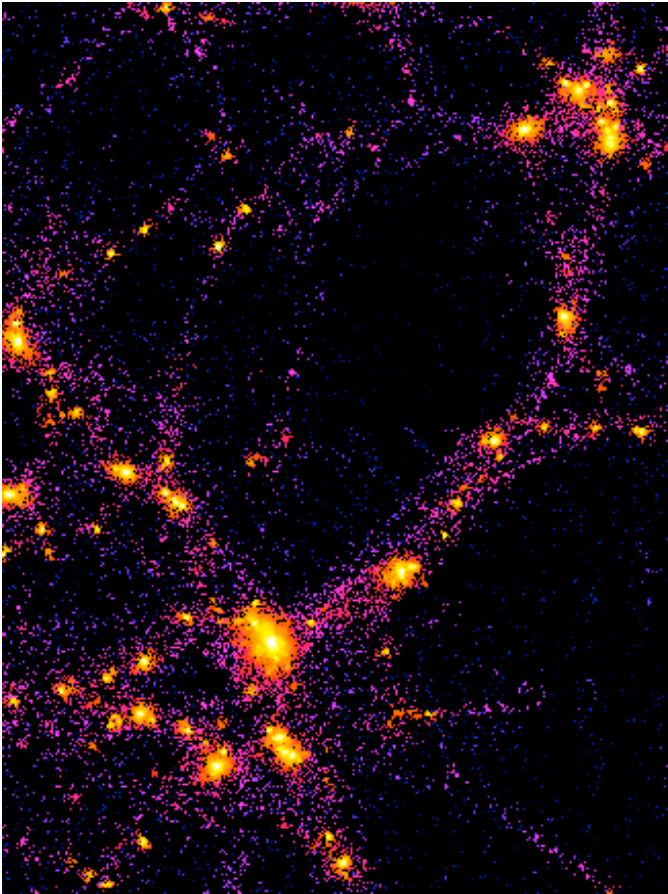
- Usually easy to parallelize
- Simple ways:
 - 1 Each node reads a different file and do the same work
 - 2 Multiple nodes read the same file but do different works

An example

Compute density from particle distribution



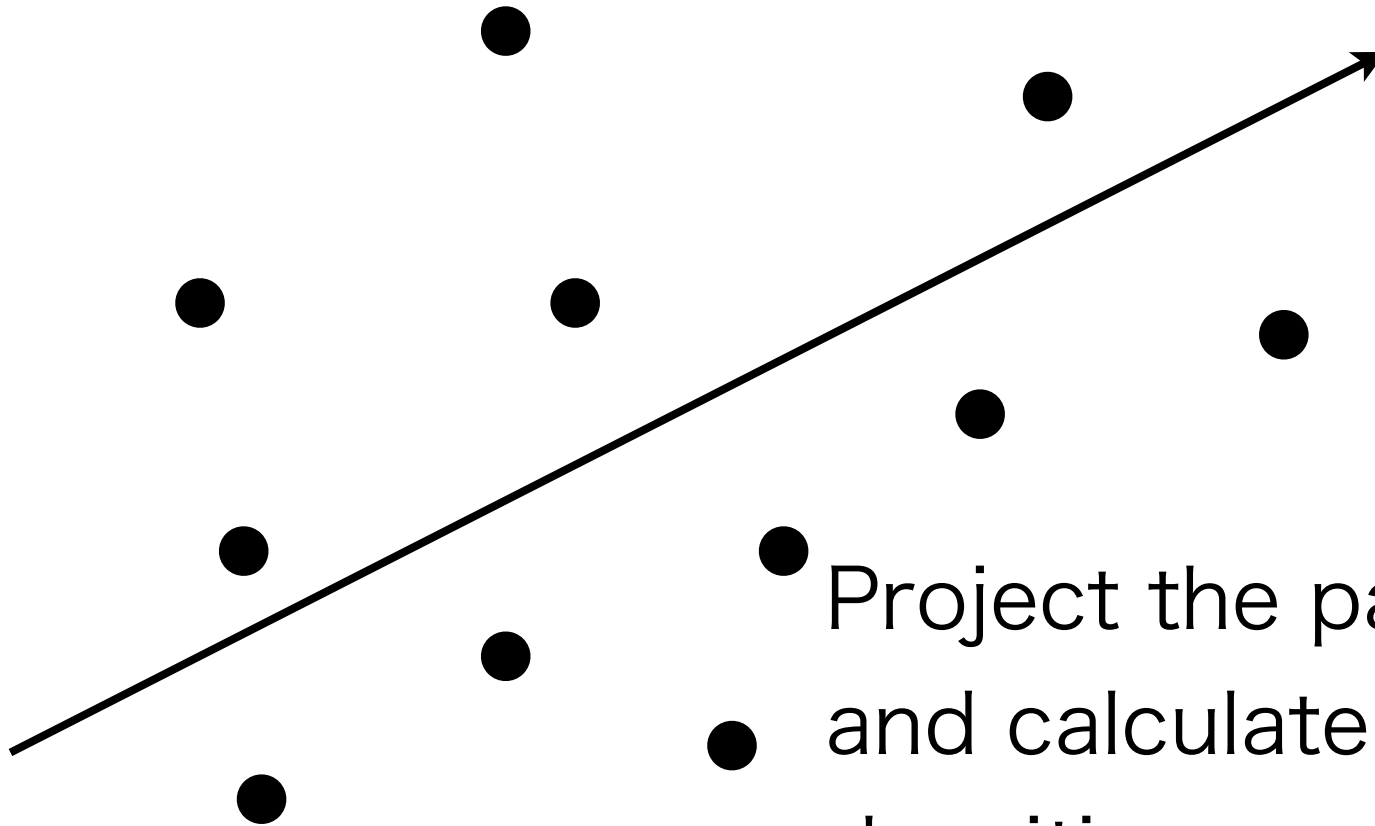
Problem!



With regular grids:

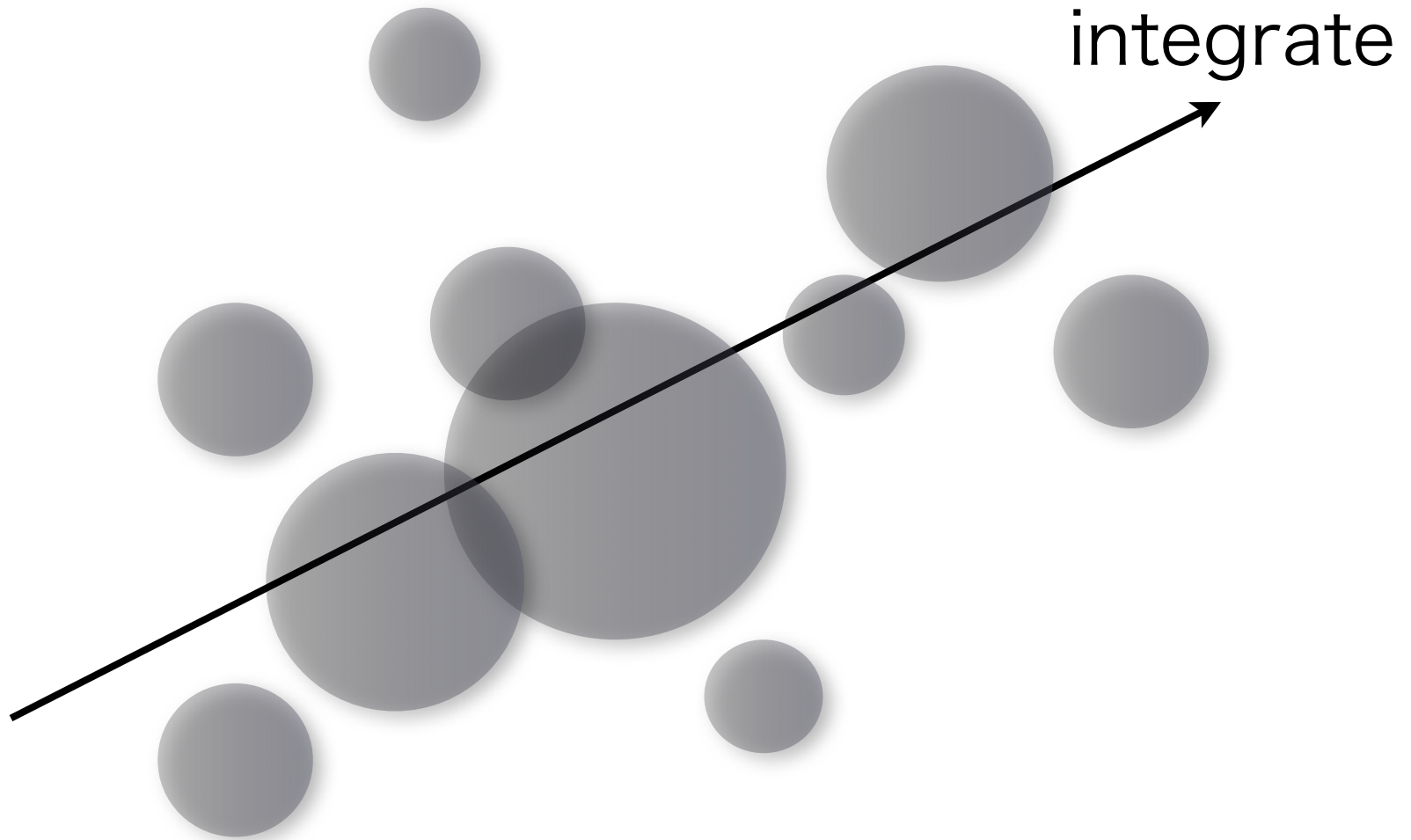
- “Densities” are discretized, which should be continuous quantities.
- “0 density” can be assigned.

Ray-tracing

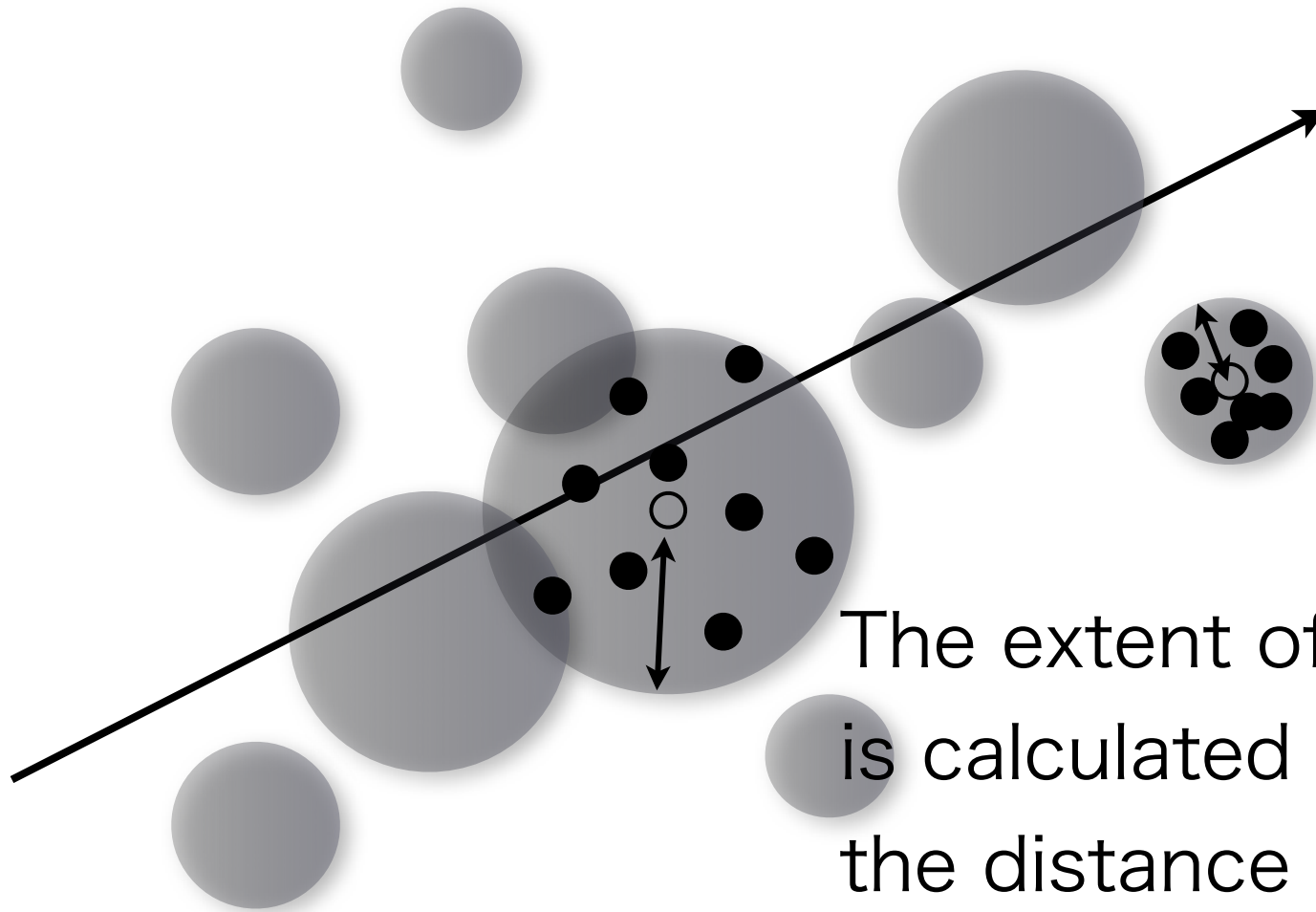


● Project the particles
● and calculate column
densities.

Particles as “clouds”



Particles as “clouds”

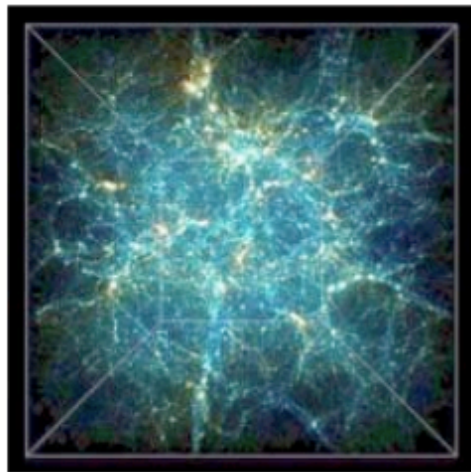
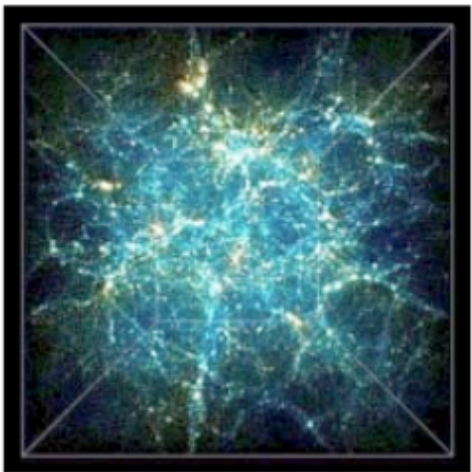
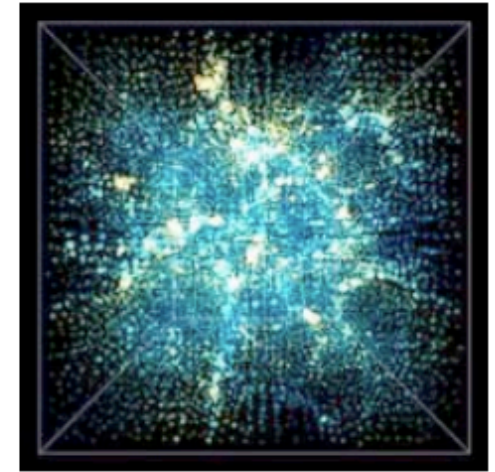
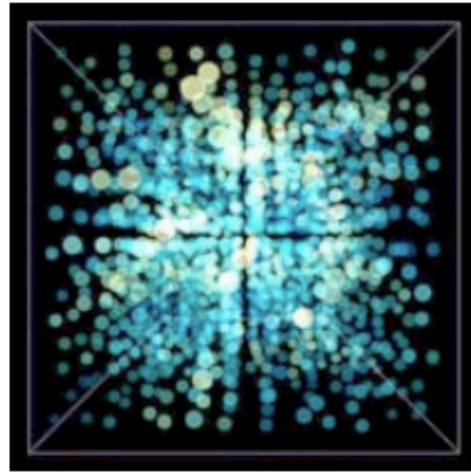
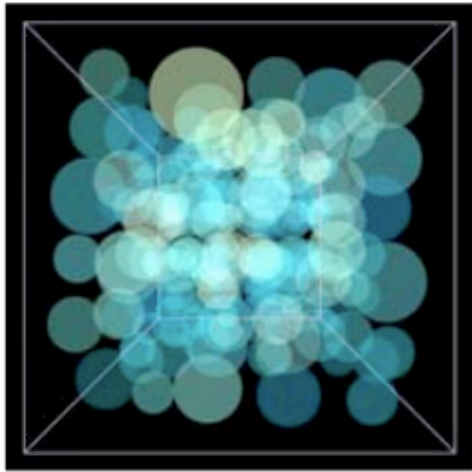


The extent of a cloud is calculated from, e.g., the distance to N-th neighbour.

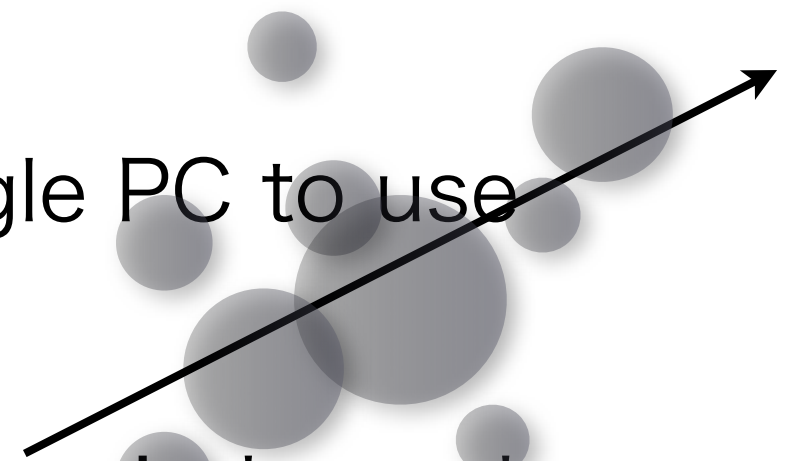
Smooth, and high-res.



Hierarchical projection

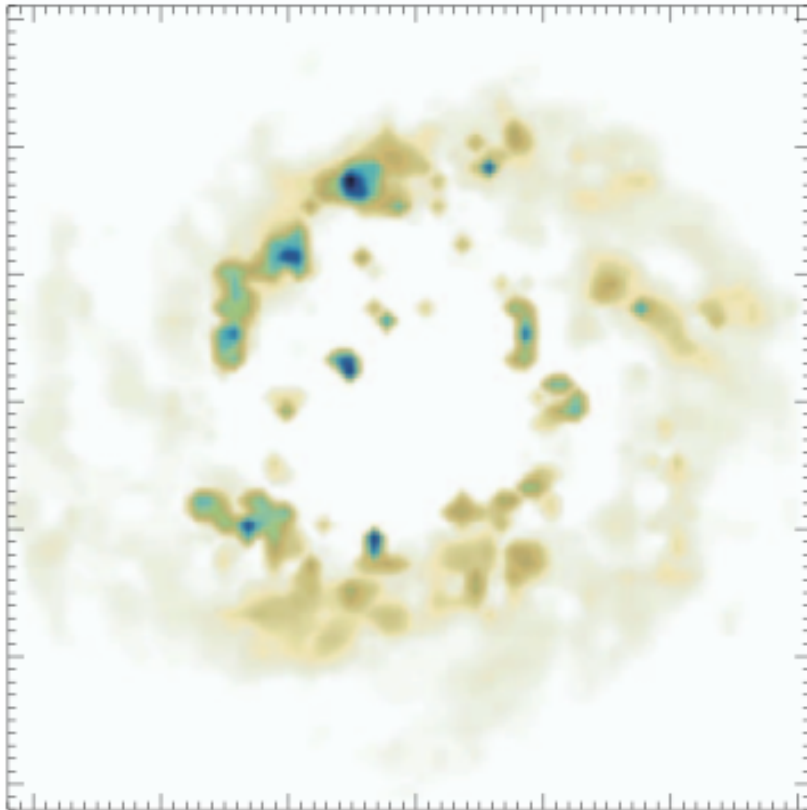


Parallelization

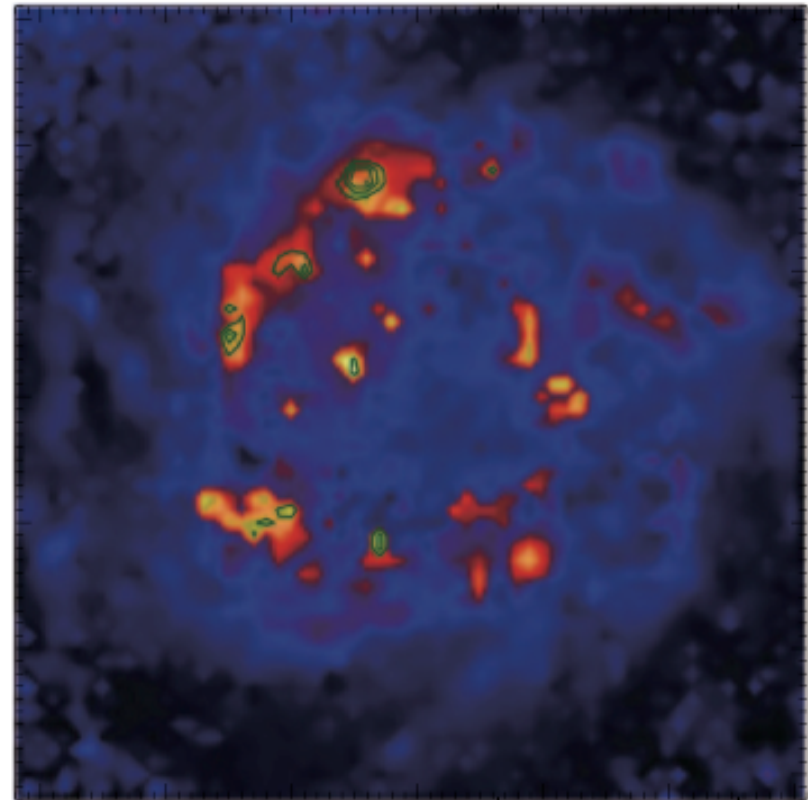
- Inefficient for a single PC to use billions of particles.
 - The outputs are often independent of each other, and are often additive (e.g., density)
 - Using M cores is $\sim M$ times faster
- 

Simulated observation

Flux



Flux ratio HCN/HCO^+



How does a blackhole look ?

Doppler effect
by fast spin



“ordinary” look



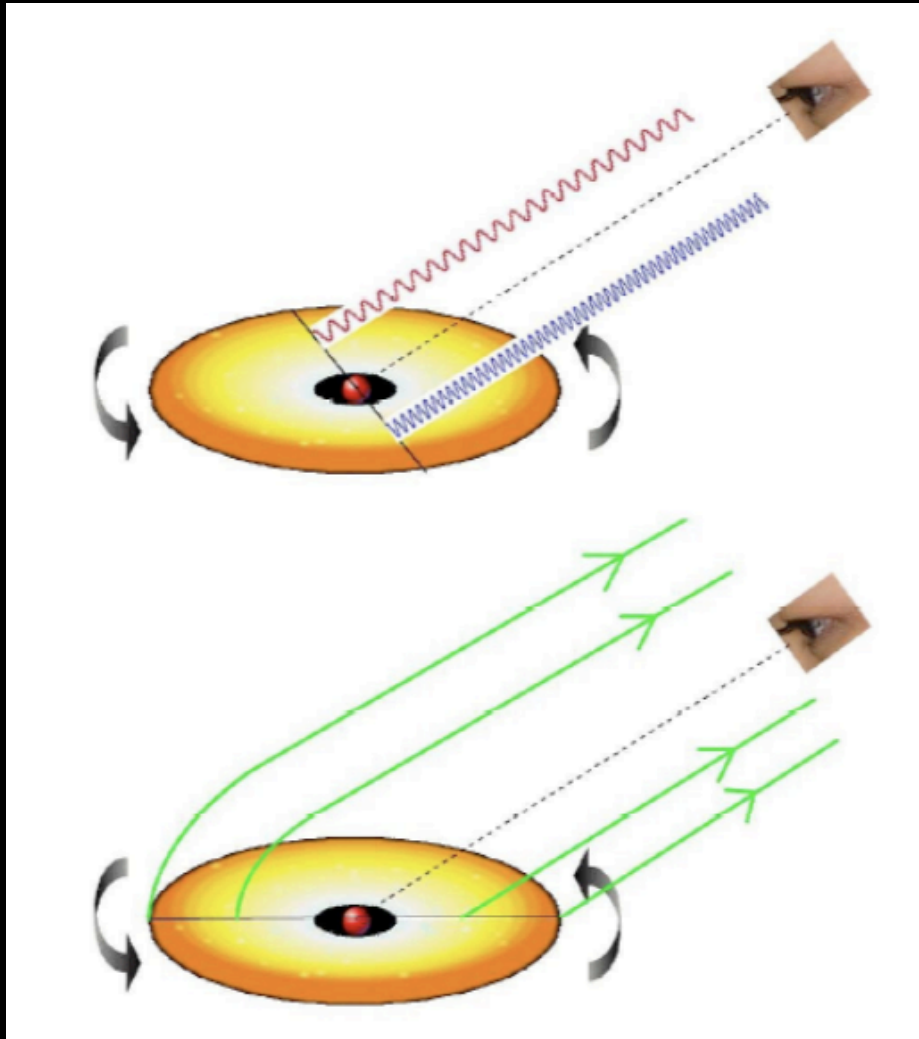
Gravitational effect



Gravity + spin



Seeing a BH



Spin :

Doppler effect.

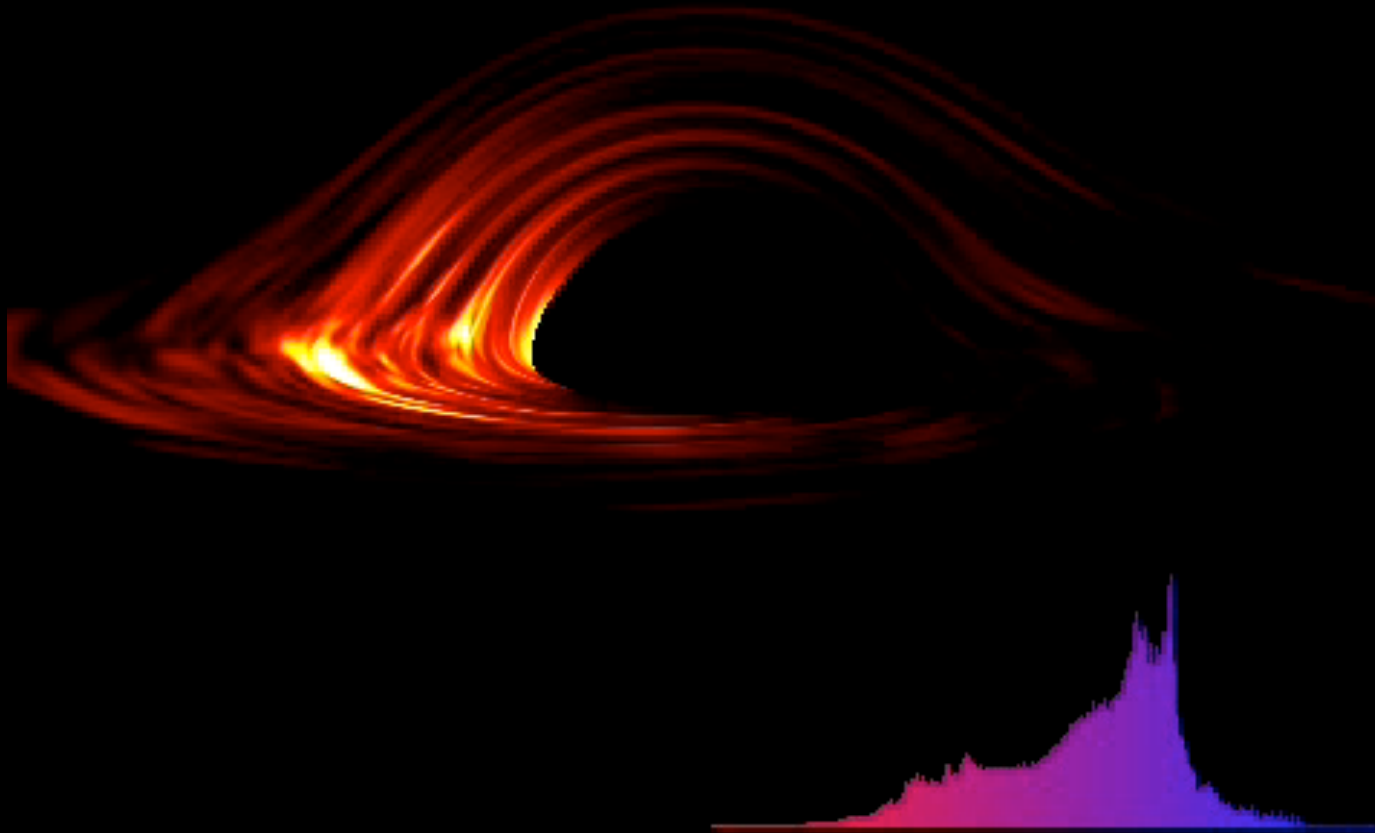
Relatively fast or slow

Strong gravity :

Light from behind can

reach us

More realistic case



Spectral energy distribution

File i/o

Opening a data file

```
int main()

{ FILE *stream;

  stream = fopen("yourfile.dat","r");

  fscanf("%d %lf %lf", &n, &x, &y);

  fclose(stream);

}
```

FILE is a special structure type that establishes a buffer area, and stream is the identifier of the created buffer area.

Using a binary data

```
int main()
{ FILE *stream;

  stream = fopen("yourfile.dat","r");

  fread(&pos[0].x, sizeof(struct grid), 200, stream);

  fclose(stream);
}
```

fread / fwrite command can be used with the data type and the size specified.

Compare the data *quality* in a binary/ascii data file.