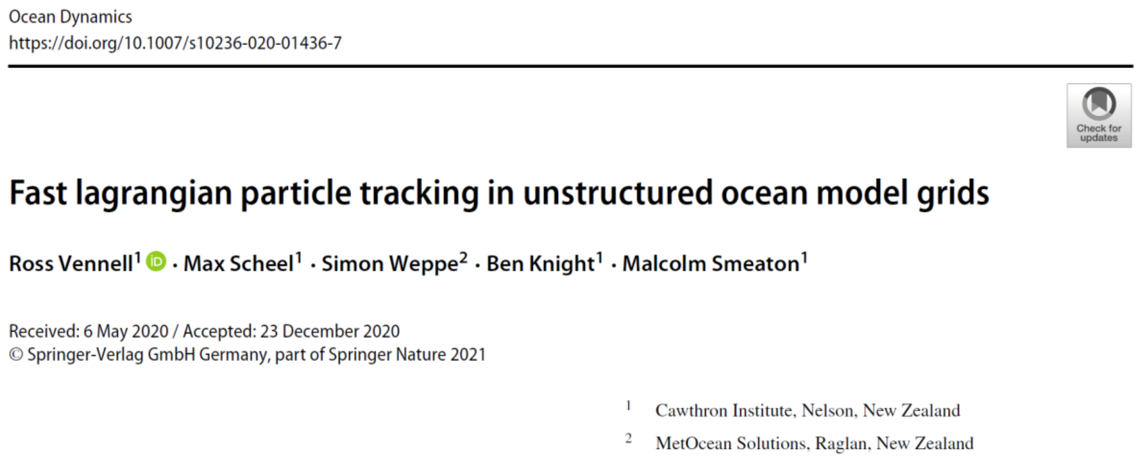
Fast Lagrangian Particle Tracking In Unstructured Ocean Model Grids

- Ross Vennell – Cawthron Institute, Nelson, New Zealand



Ocean Dynamics
https://doi.org/10.1007/s10236-020-01436-7

Fast lagrangian particle tracking in unstructured ocean model grids

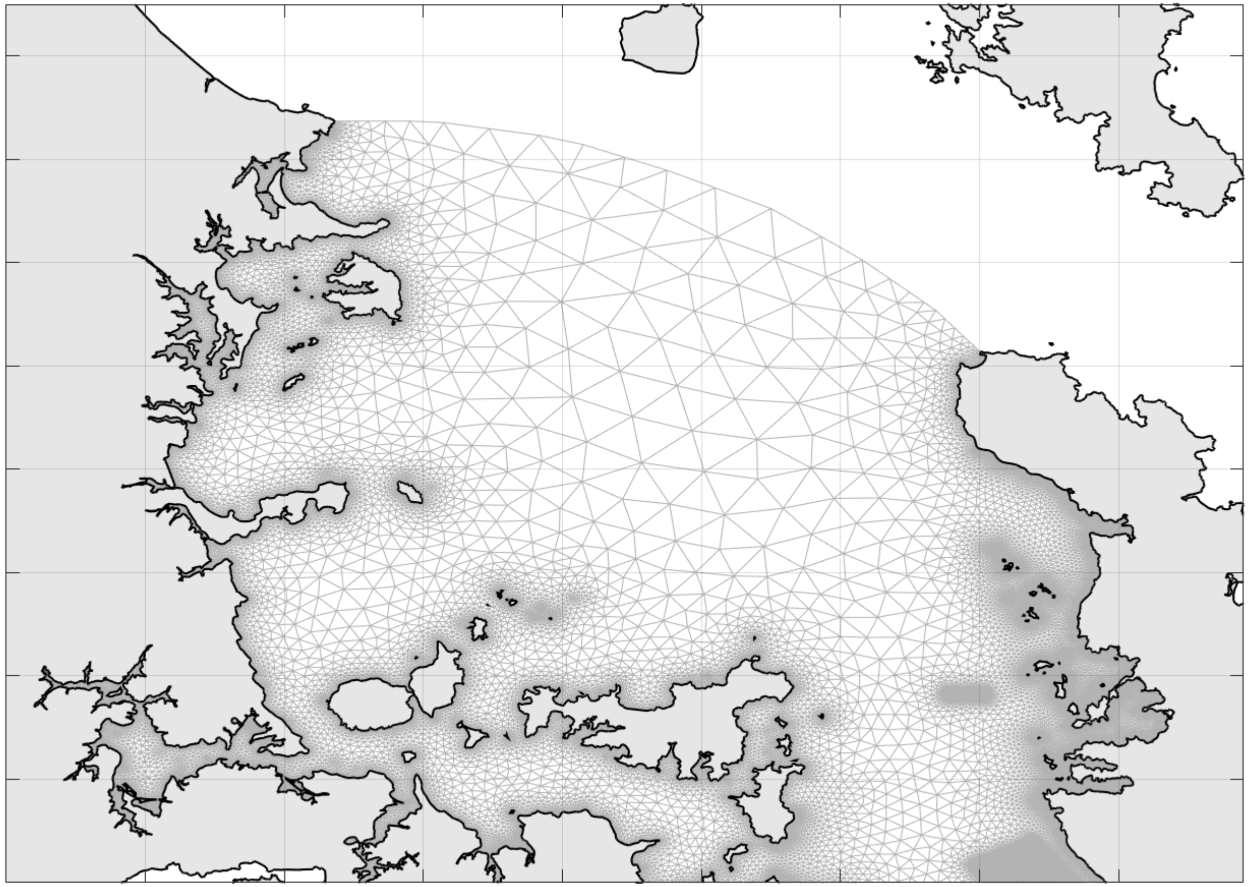
Ross Vennell[1] · Max Scheel[1] · Simon Weppe[2] · Ben Knight[1] · Malcolm Smeaton[1]

Received: 6 May 2020 / Accepted: 23 December 2020

[1] Cawthron Institute, Nelson, New Zealand
[2] MetOcean Solutions, Raglan, New Zealand

Unstructured grids are great!

Achieve resolutions not possible with regular grids



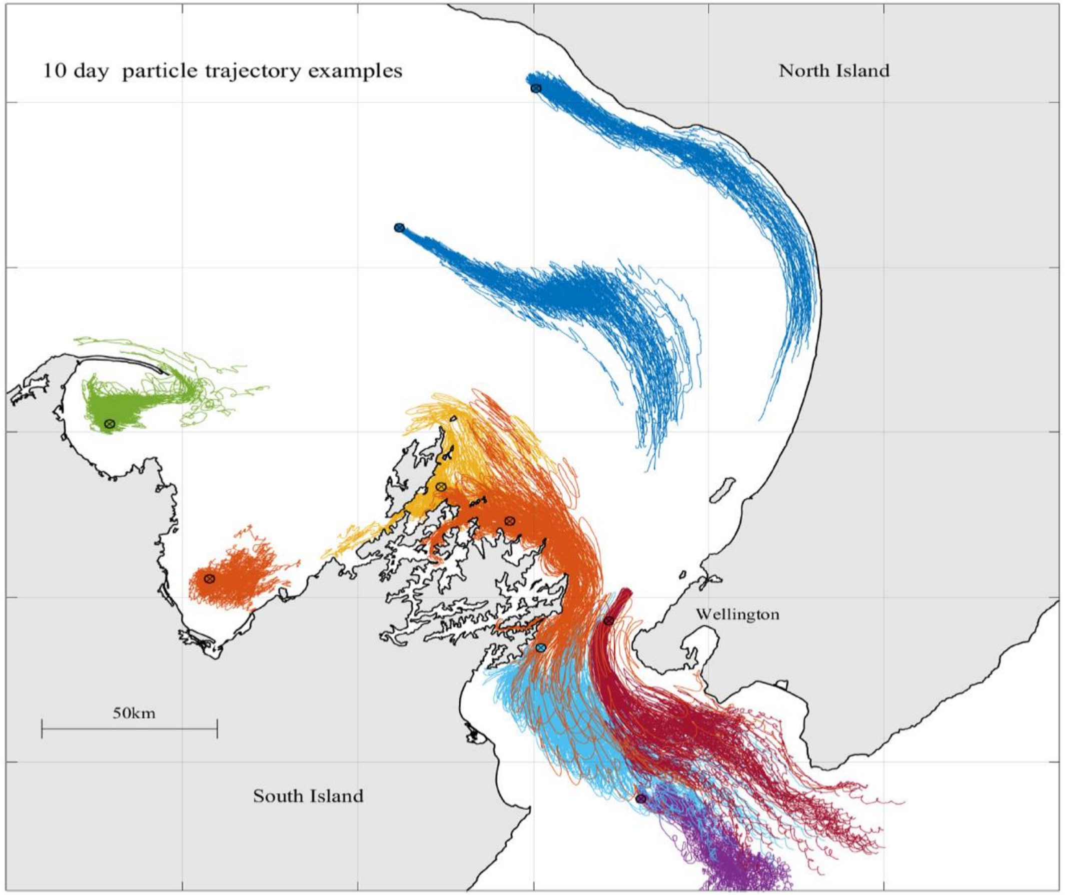
Post model run particle tracking?

- Regular grid – Fast
- Unstructured grids can be very slow!

A day to run 100,000 Particles!

CAWTHRON

Why speed matters in particle tracking!



Enables tracking millions of particles

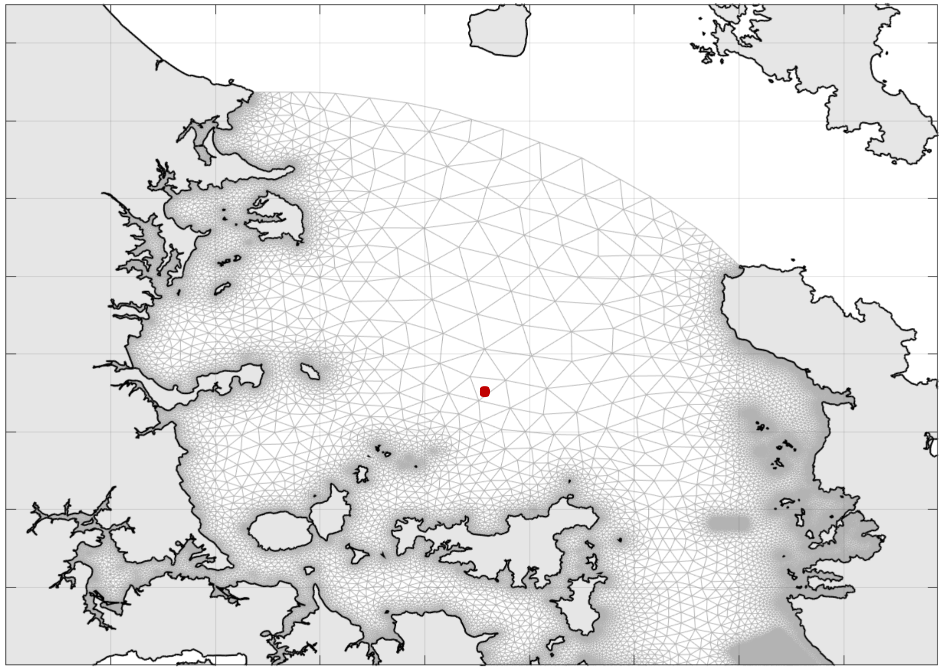
- Better statistics:
  - Many release locations and times
  - Heat maps
  - Low probabilities, high significance

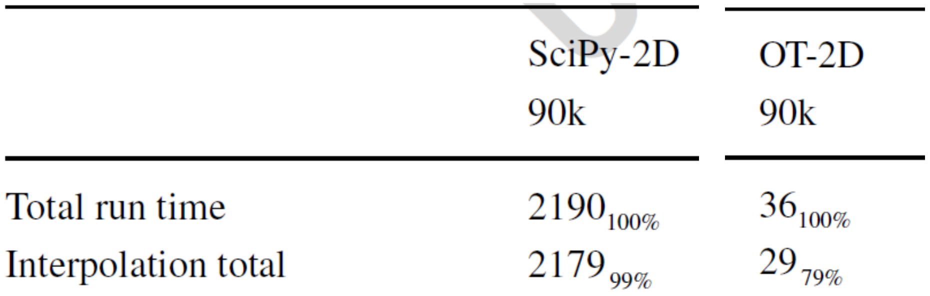
    (e.g. invasive species)
- Many individual behaviours

  (e.g. parameters drawn from distributions)
- My impatience!

CAWTHRON

Why is unstructured grid particle tracking so slow?



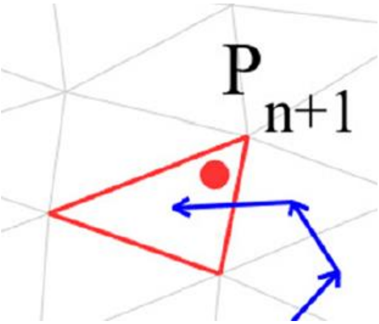


	SciPy-2D 90k	OT-2D 90k
Total run time	$2190_{100\%}$	$36_{100\%}$
Interpolation total	$2179_{99\%}$	$29_{79\%}$

Spatial interpolation velocity, depth, tide, temperature ….

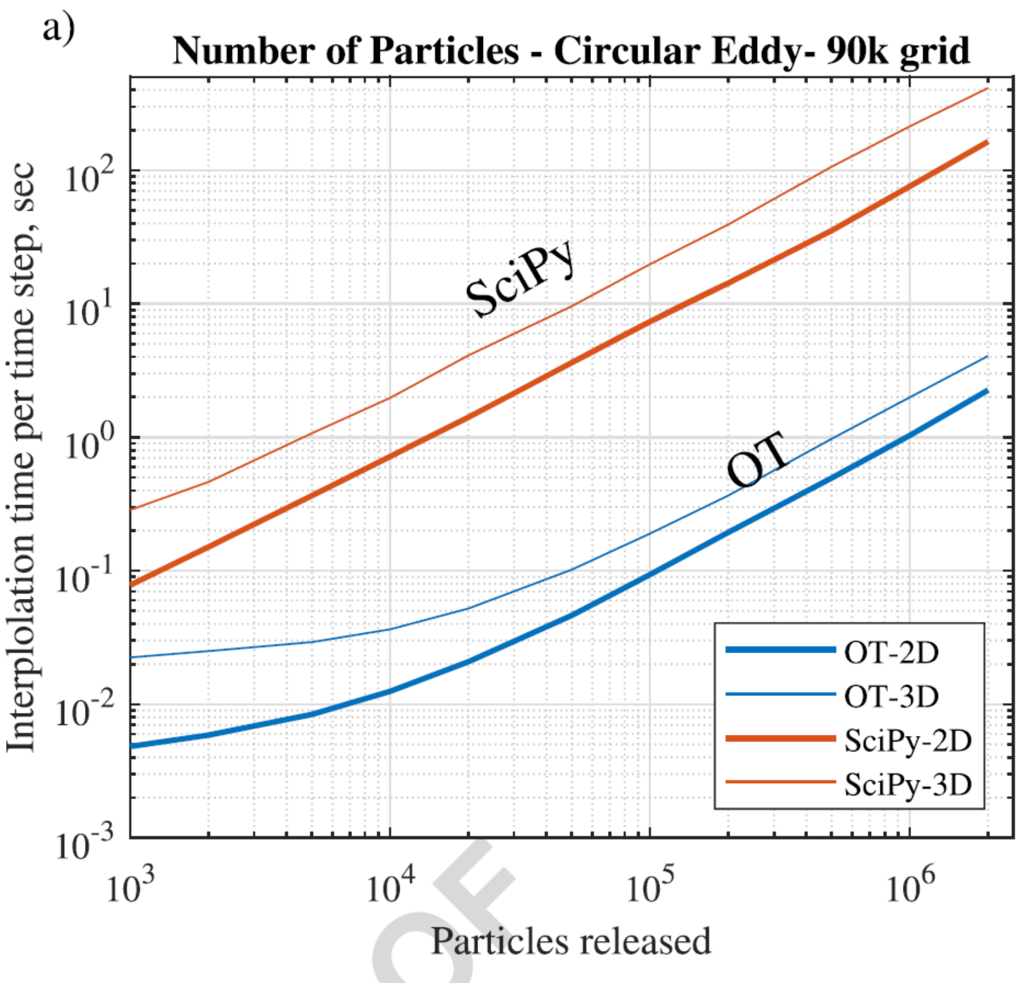
Three steps:

1. Find triangle containing particle
2. Calculate interpolation weights
3. Apply weights



CAWTHRON

How much faster than OpenDrift?

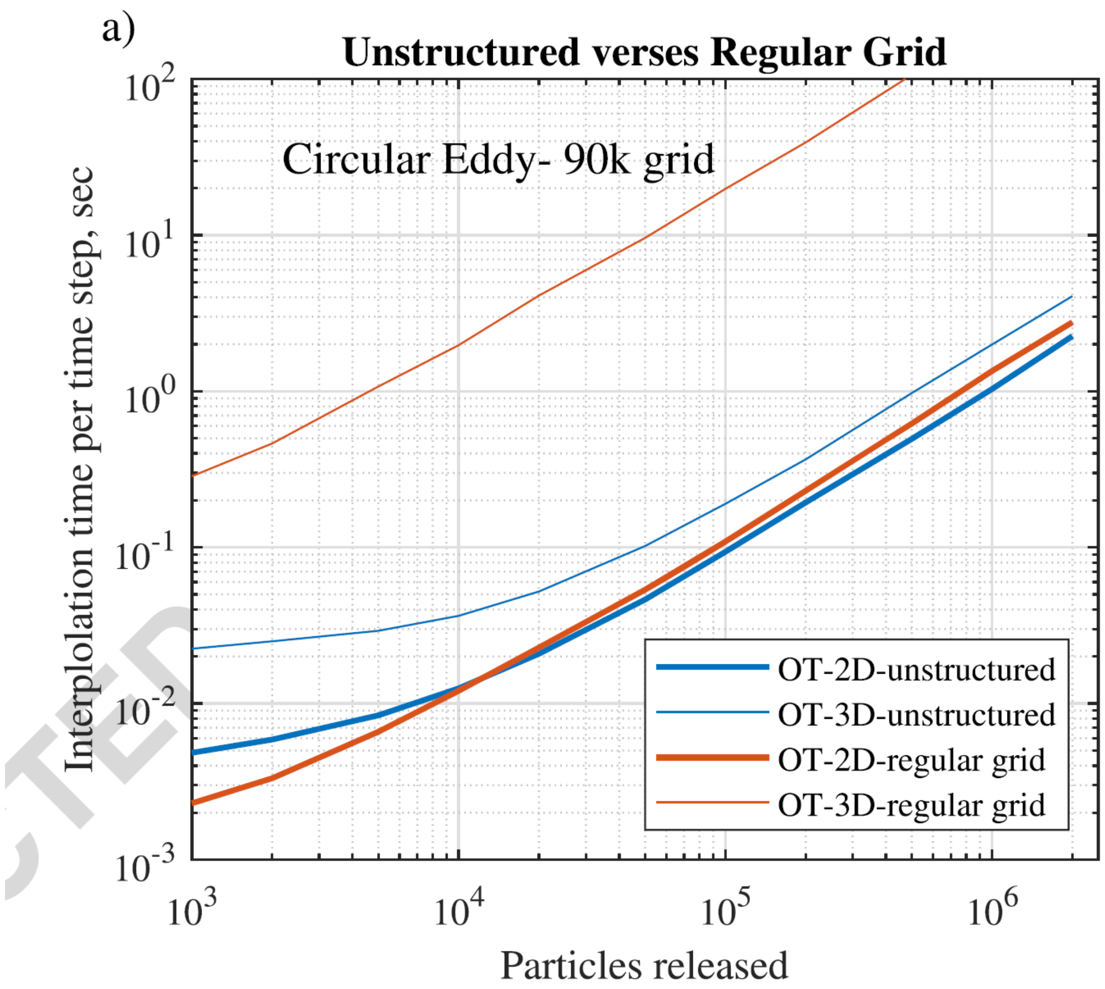


200-500 times faster

More importantly.... as fast as regular grid!

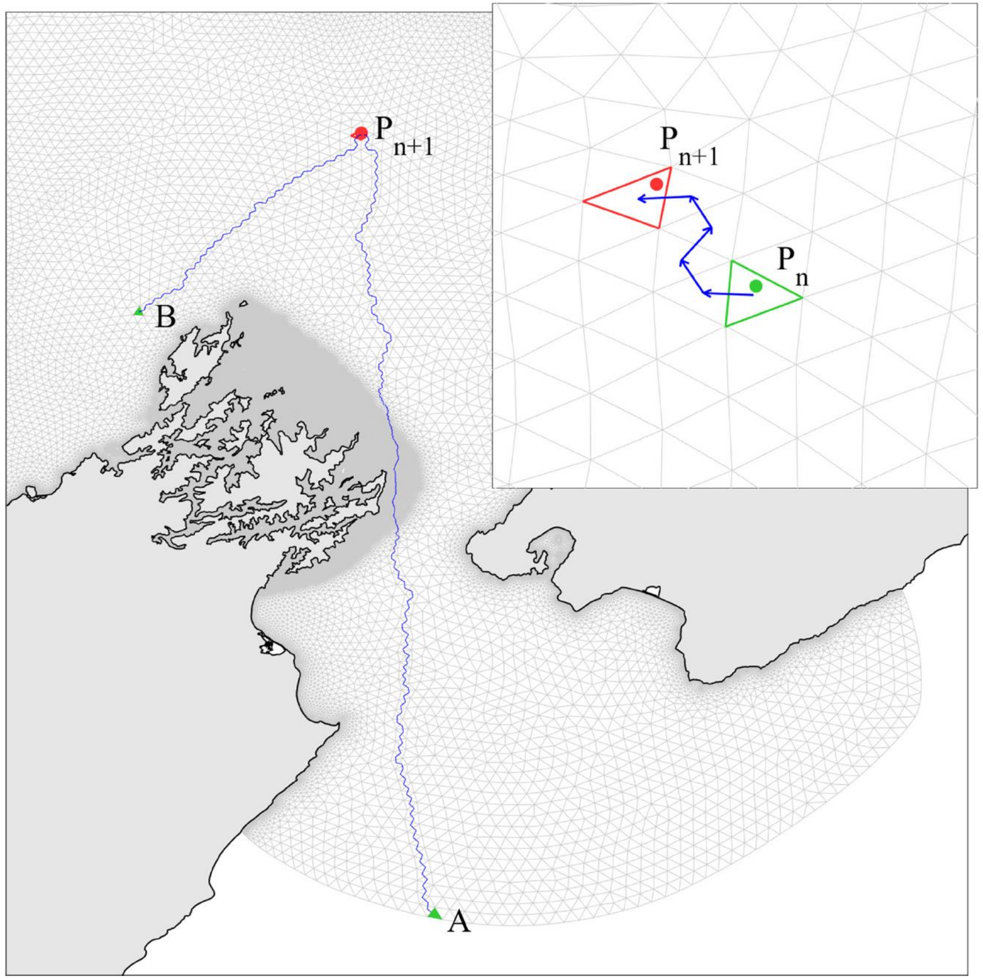
Finding cell containing particle

- Regular grid - fast by rounding
- Triangular grid was slow, now similar speed!



CAWTHRON

How? 1) Finding the triangle



The Triangle walk

OpenDrift (Python's SciPy) - poor initial guess

Obvious? - Start at triangle containing particle at last time step

Short Triangle Walk

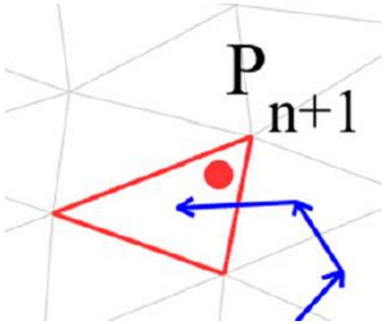
Average walk is typically 0.7 triangles!

In 3D Schism S-layer model

- *Short Vertical Walk* as well

CAWTHRON

How? 2) Store and reused interpolation weights

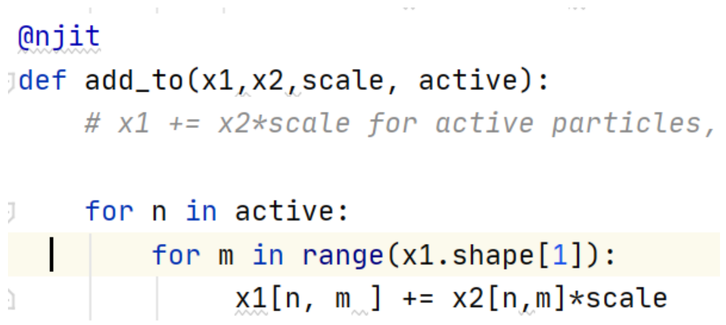


Calculate weights once- use many times!

e.g. time interpolation requires two spatial interpolations at same locations

How? 3) Python Fast?

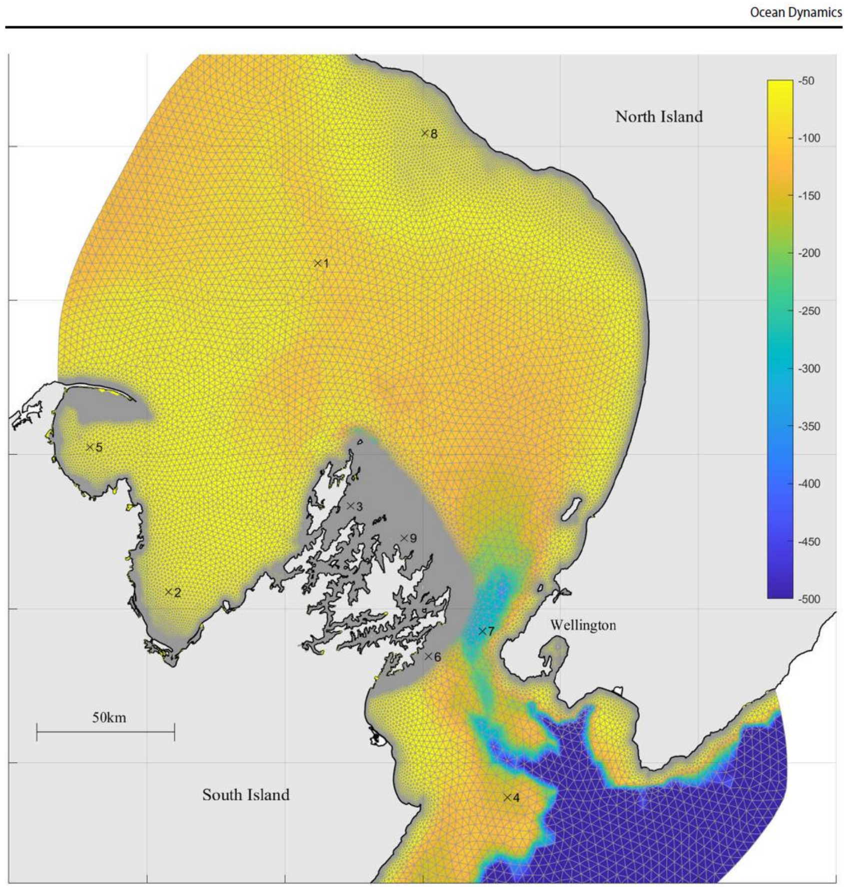
use Numba: Python loops at C speeds

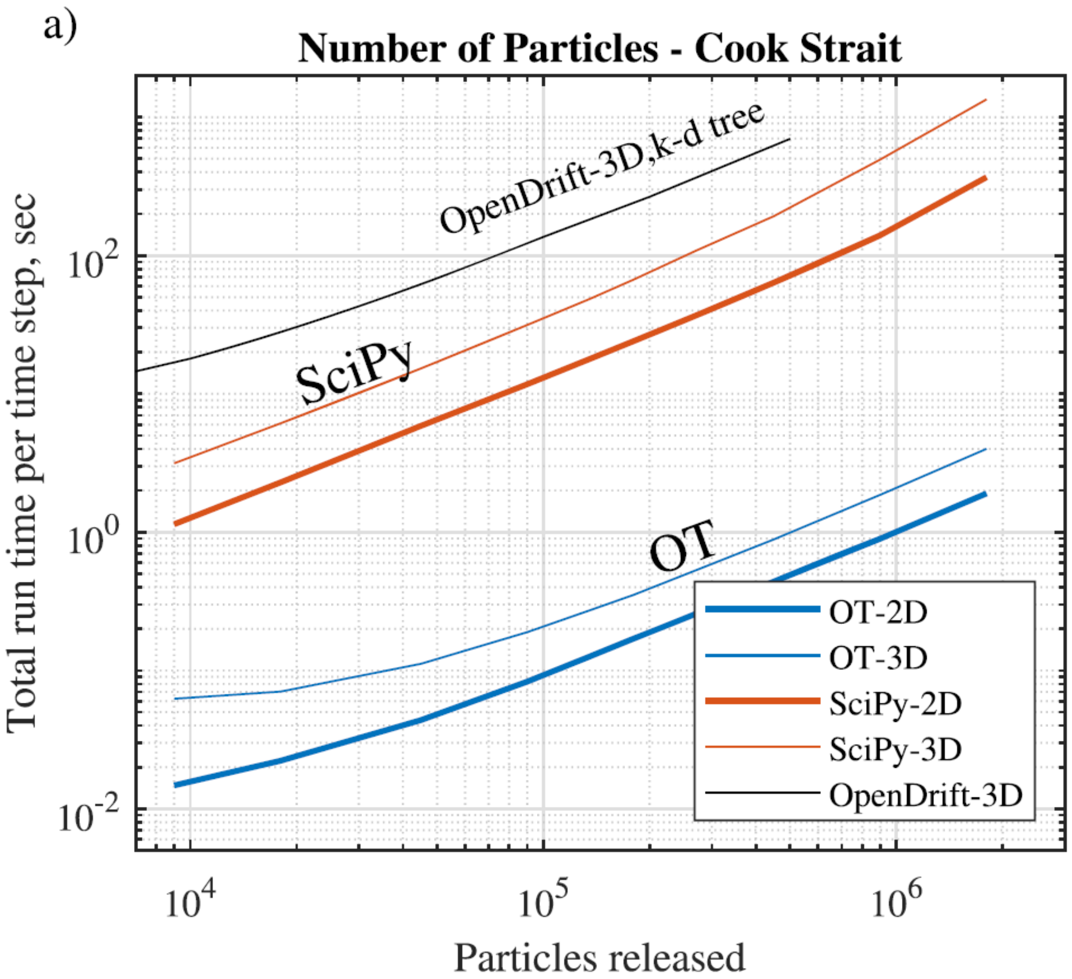


```
@njit
def add_to(x1,x2,scale, active):
    # x1 += x2*scale for active particles,

    for n in active:
        for m in range(x1.shape[1]):
            x1[n, m ] += x2[n,m]*scale
```

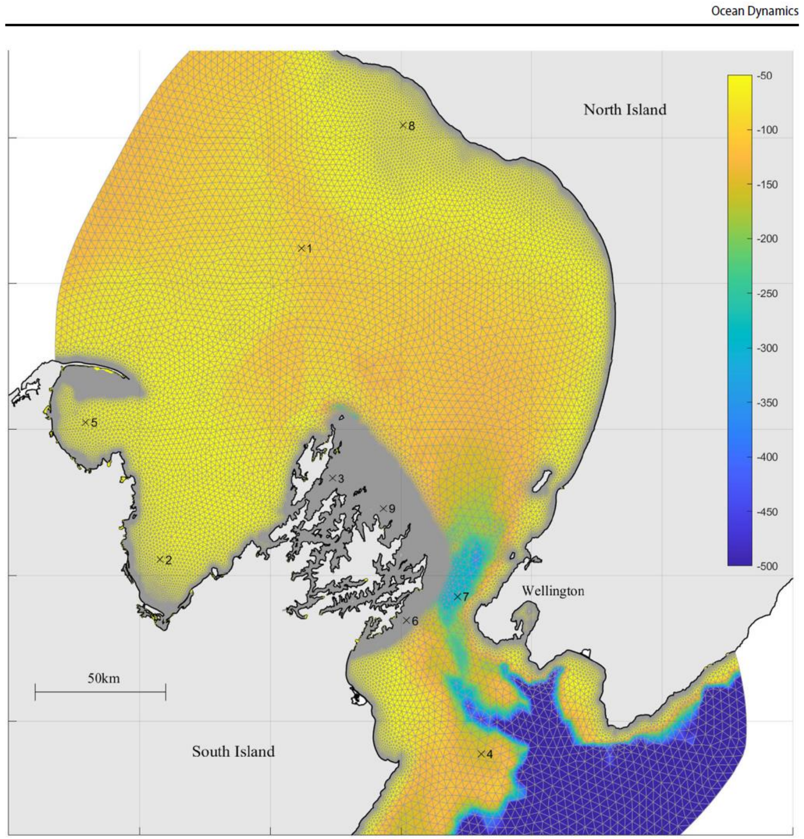
Cook Strait- OpenDrift Comparison

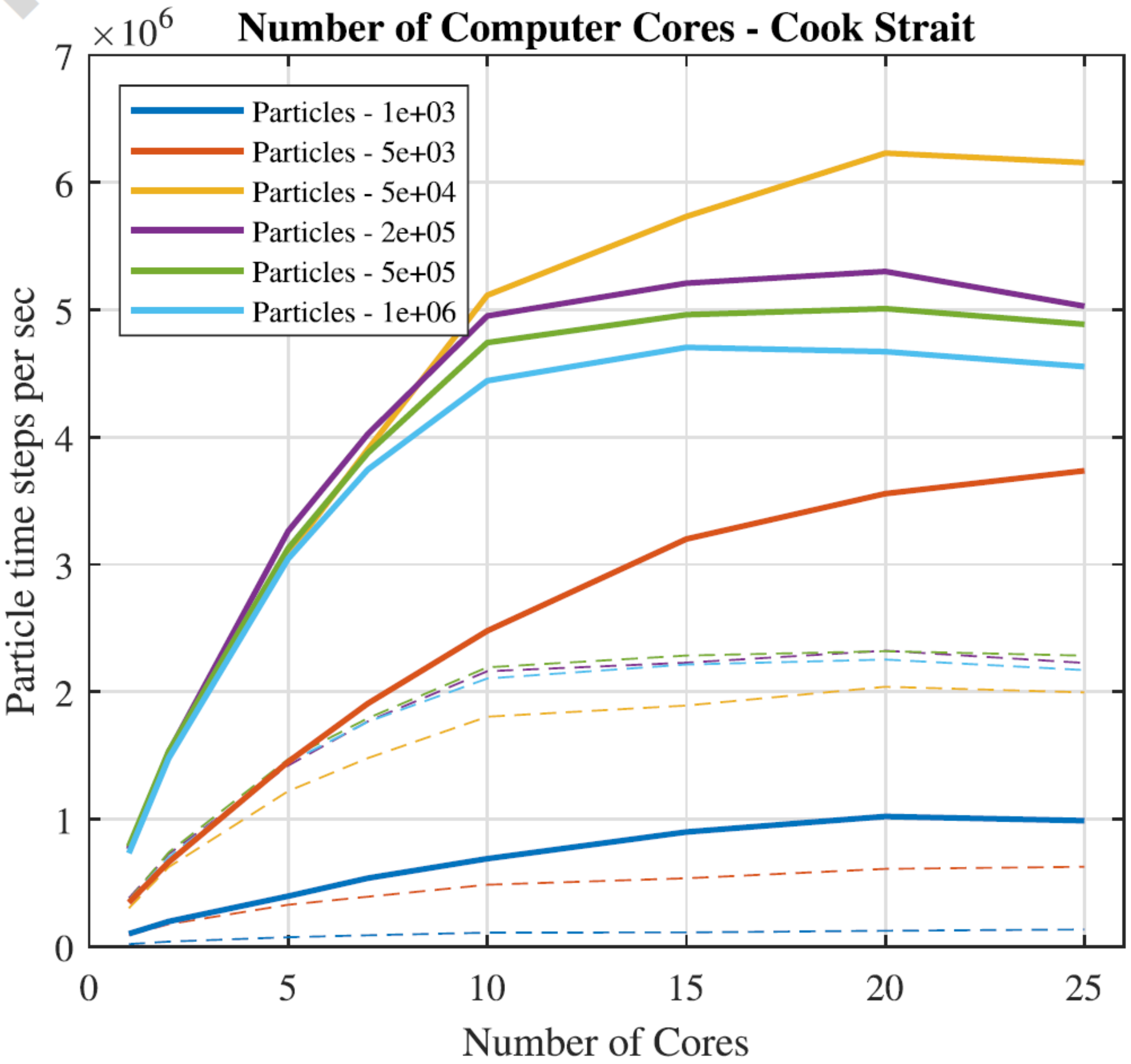




CAWTHRON

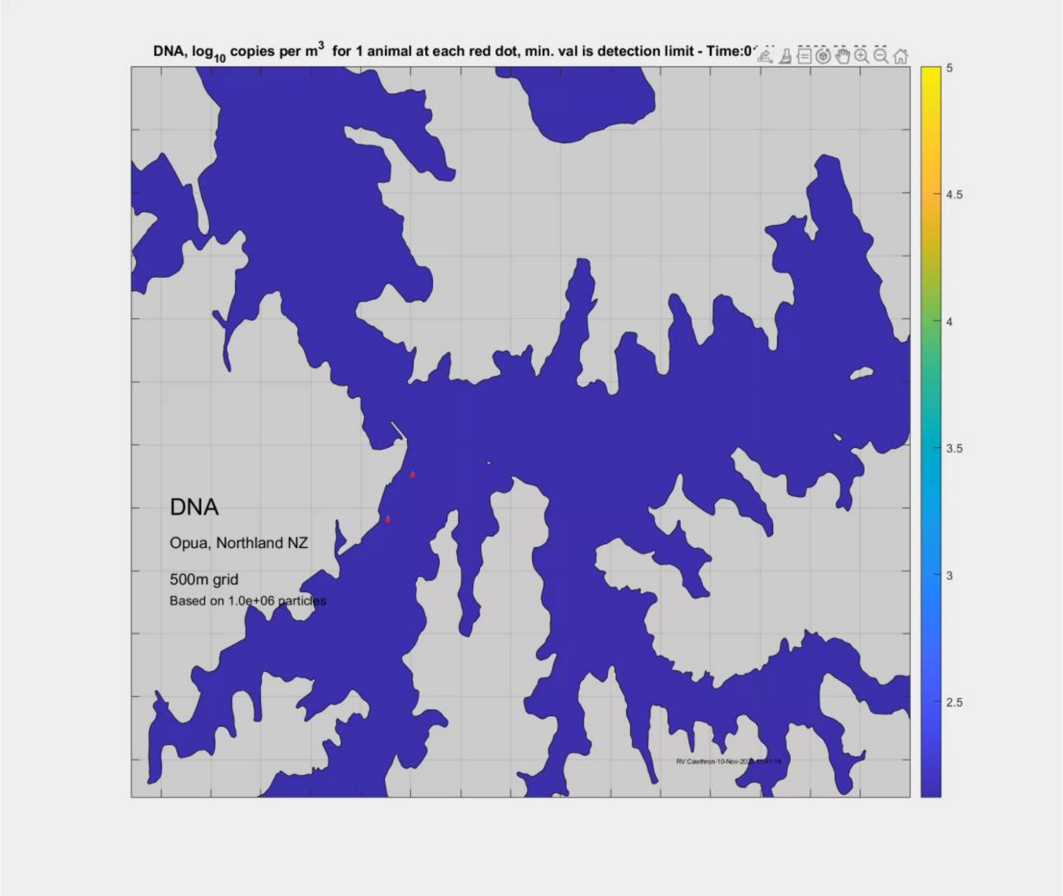
Multi Core performance:



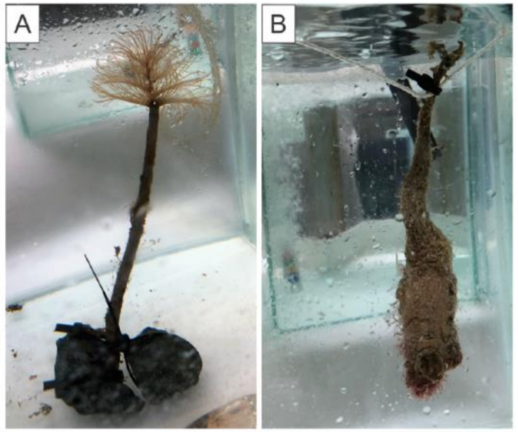


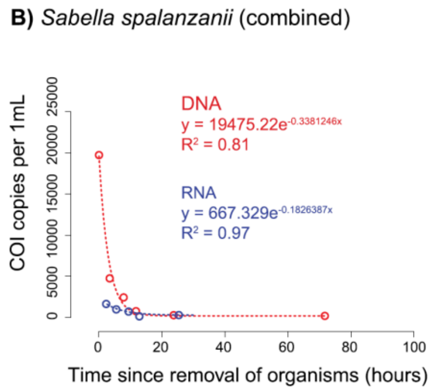
CAWTHRON

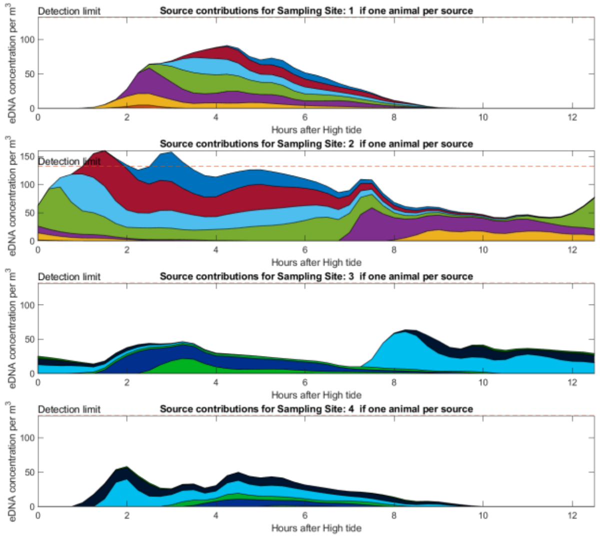
Example: A Biosecurity eDNA inverse problem



DNA, $\log_{10}$ copies per m^3 for 1 animal at each red dot, min. val is detection limit - Time:0
5
4.5
4
3.5
3
2.5
DNA
Opua, Northland NZ
500m grid
Based on 1.0e+06 particles



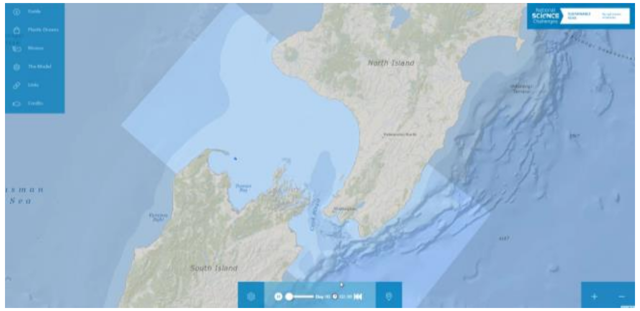




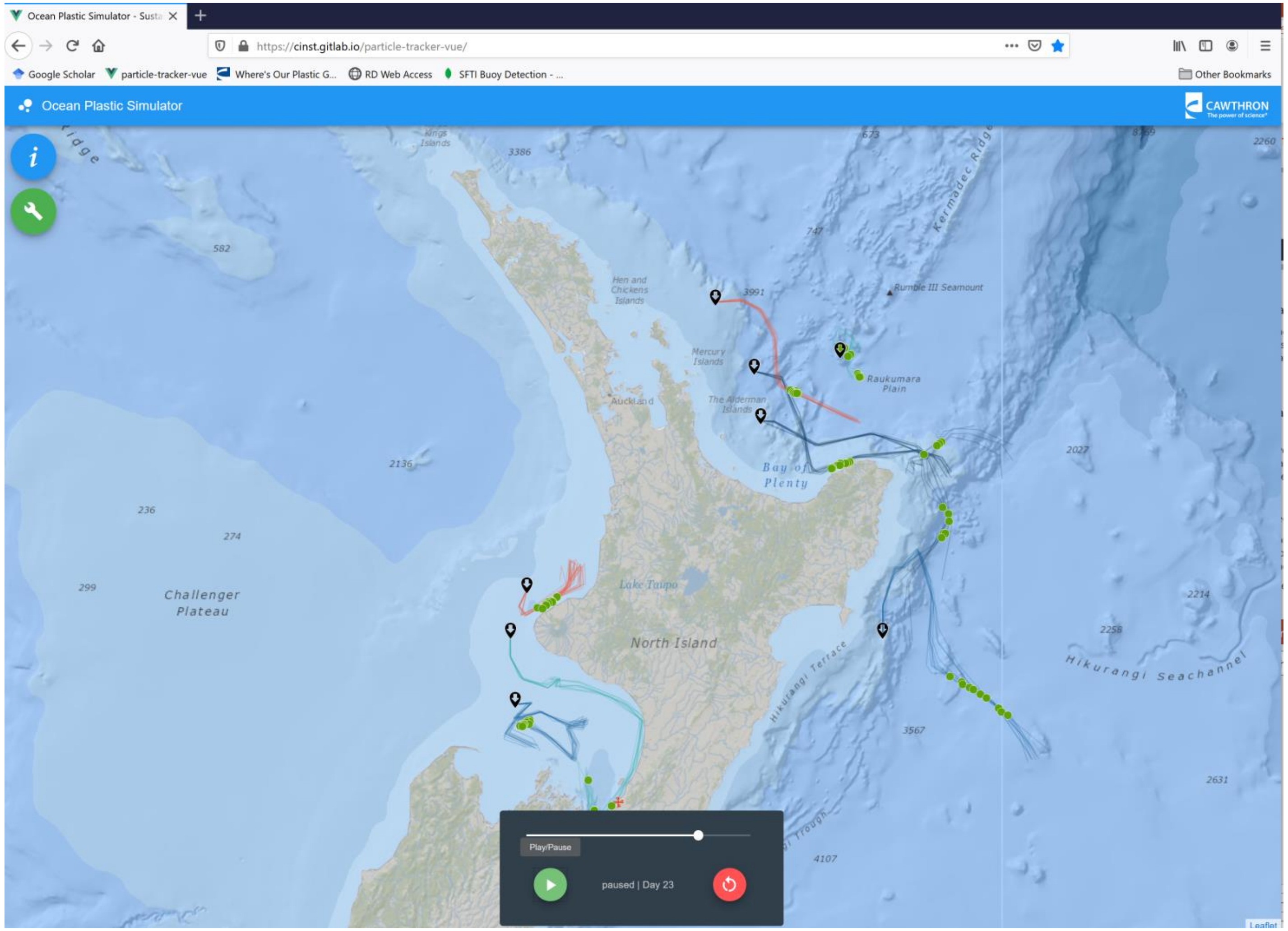
MBIE "Biosecurity toolbox program"

Why speed maters!

Ocean Plastics , public engagement tool

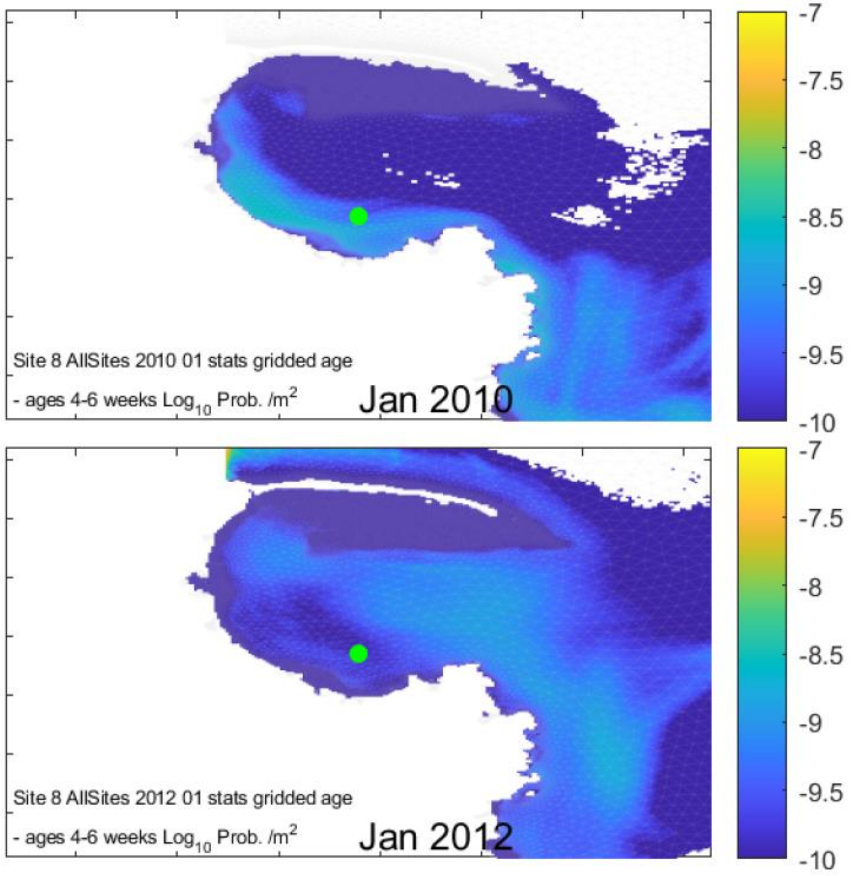


www.oceanplasticsimulator.nz



CAWTHRON

Statistics on the fly.


-7
-7.5
-8
-8.5
-9
-9.5
-10
Site 8 AllSites 2010 01 stats gridded age
- ages 4-6 weeks Log_{10} Prob. /m^2
Jan 2010
-7
-7.5
-8
-8.5
-9
-9.5
-10
Site 8 AllSites 2012 01 stats gridded age
- ages 4-6 weeks Log_{10} Prob. /m^2
Jan 2012

Sources of green lipped mussel spat larvae:

120 million particles released over 10 year Schism hindcast,
5 hour run time on 18 cores

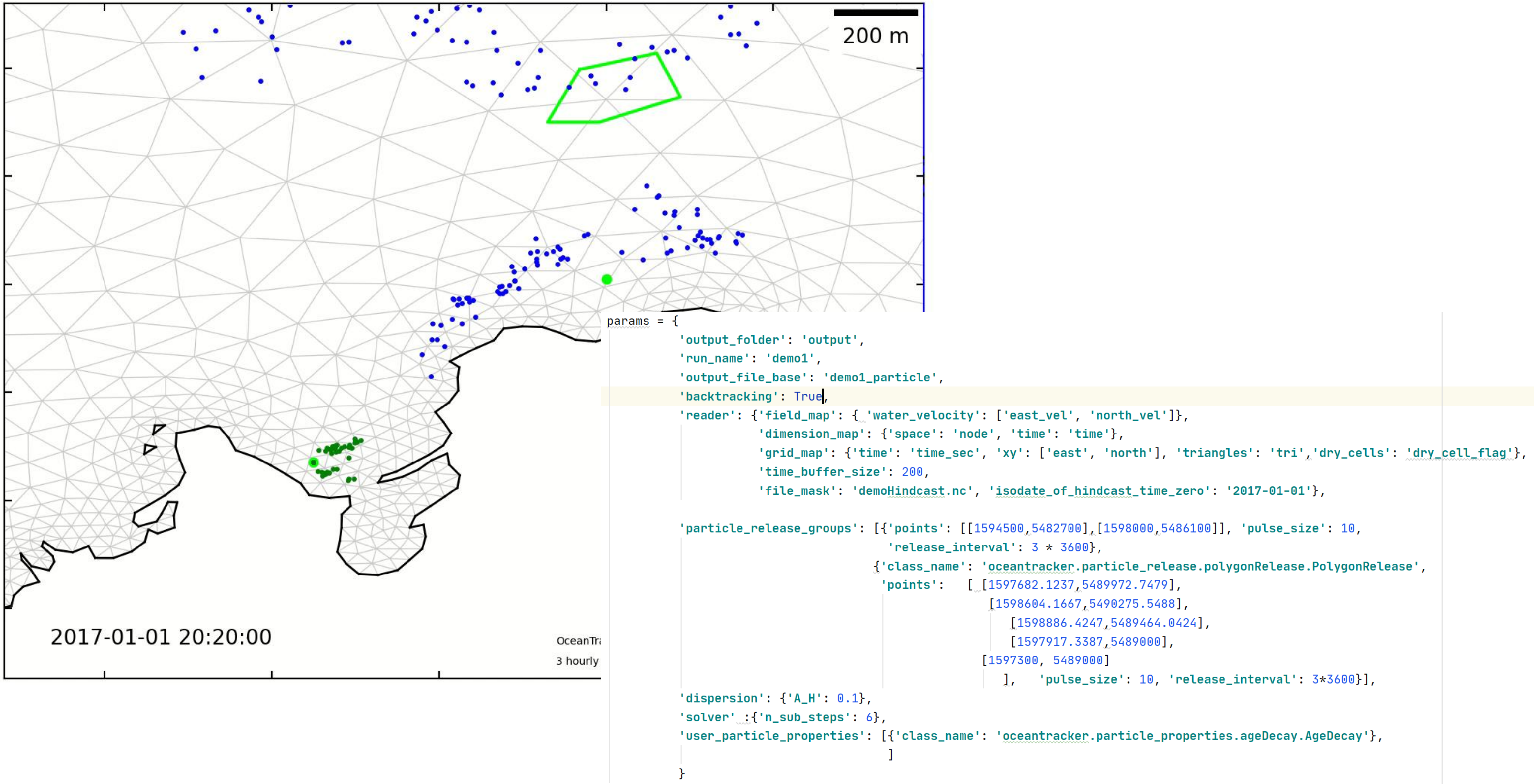
→ Tera Bytes of particle tracks to sort through!

Build heat maps as you go, i.e. don’t record tracks

only mega bytes of output !

CAWTHRON

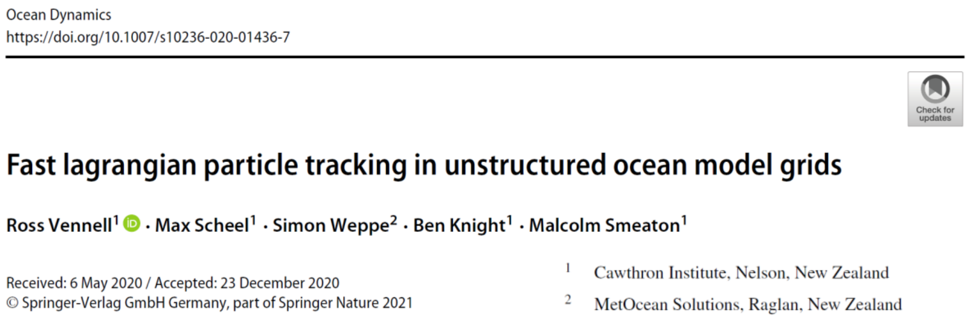
OceanTracker



```
params = {
            'output_folder': 'output',
            'run_name': 'demo1',
            'output_file_base': 'demo1_particle',
            'backtracking': True,
            'reader': {'field_map': { 'water_velocity': ['east_vel', 'north_vel']},
                         'dimension_map': {'space': 'node', 'time': 'time'},
                         'grid_map': {'time': 'time_sec', 'xy': ['east', 'north'], 'triangles': 'tri','dry_cells': 'dry_cell_flag'},
                         'time_buffer_size': 200,
                         'file_mask': 'demoHindcast.nc', 'isodate_of_hindcast_time_zero': '2017-01-01'},

            'particle_release_groups': [{'points': [[1594500,5482700],[1598000,5486100]], 'pulse_size': 10,
                                          'release_interval': 3 * 3600},
                                        {'class_name': 'oceantracker.particle_release.polygonRelease.PolygonRelease',
                                         'points':   [ [1597682.1237,5489972.7479],
                                                         [1598604.1667,5490275.5488],
                                                            [1598886.4247,5489464.0424],
                                                            [1597917.3387,5489000],
                                                       [1597300, 5489000]
                                                          ],   'pulse_size': 10, 'release_interval': 3*3600}],
            'dispersion': {'A_H': 0.1},
            'solver' :{'n_sub_steps': 6},
            'user_particle_properties': [{'class_name': 'oceantracker.particle_properties.ageDecay.AgeDecay'},
                                          ]
            }
```

Wrap up.


Ocean Dynamics
https://doi.org/10.1007/s10236-020-01436-7

Check for updates

Fast lagrangian particle tracking in unstructured ocean model grids

Ross Vennell[1] · Max Scheel[1] · Simon Weppe[2] · Ben Knight[1] · Malcolm Smeaton[1]

[1] Cawthron Institute, Nelson, New Zealand

[2] MetOcean Solutions, Raglan, New Zealand

Received: 6 May 2020 / Accepted: 23 December 2020

OceanTracker

- 200-500 times faster than some existing codes
- As fast as regular grid particle tracking
- Enables scaling to millions of particles in 2D and 3D
- Smart- extensible framework for particle tracking
- Inbuilt support for Schism output:
  - works with new LSC vertical grid
  - reads sets of NetCDF files as blocks in read buffer
  - dry cell stranding…

CAWTHRON