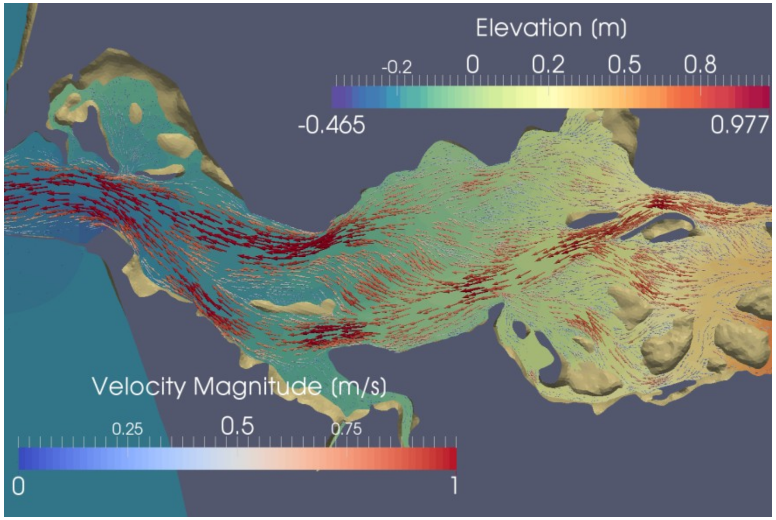
Wanted: Modern unstructured-grid regional circulation model



Tuomas Kärnä,

António Baptista

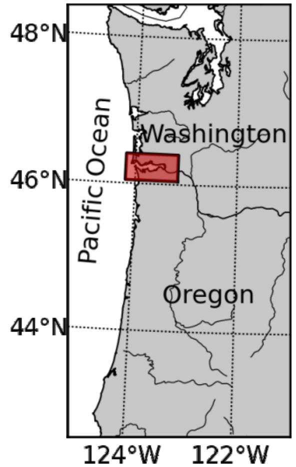
Center for Coastal Margin Observation & Prediction,

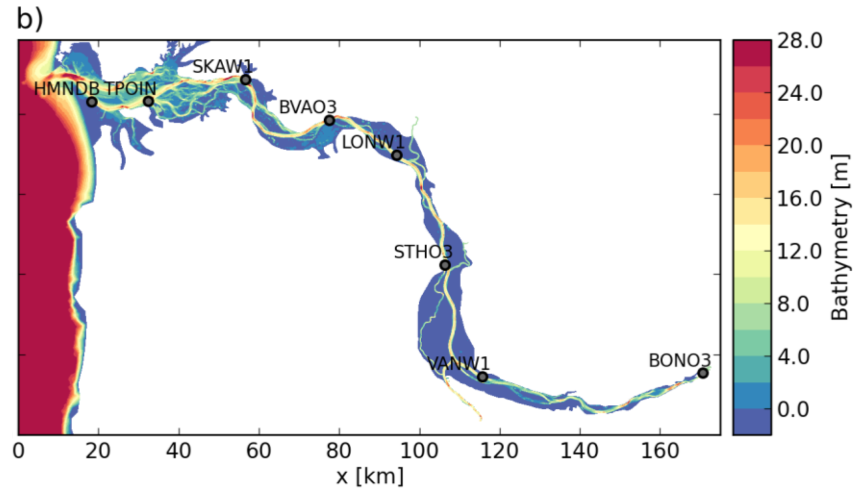
Oregon Health & Science University,

Portland, Oregon, USA

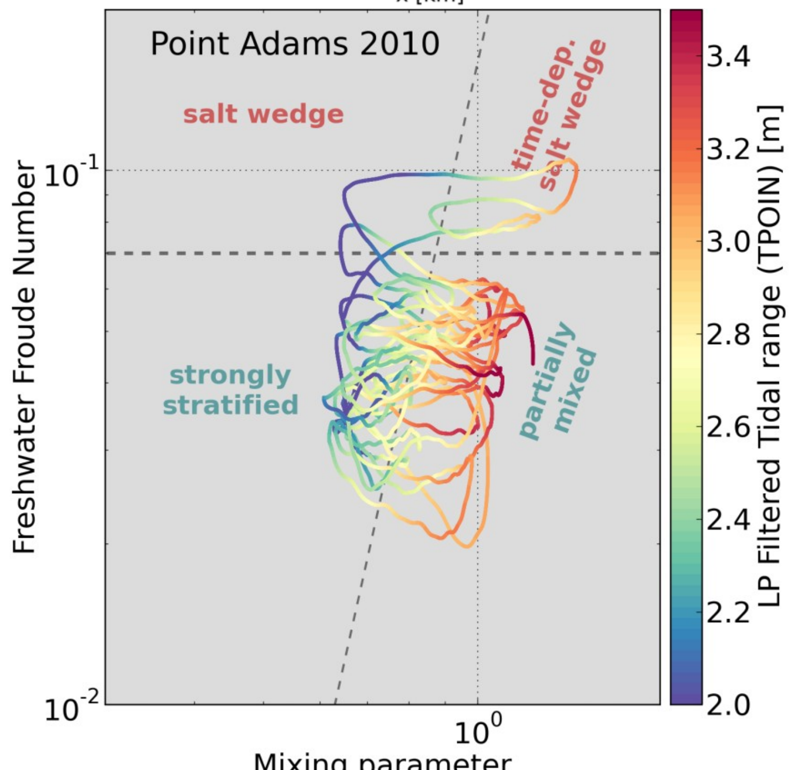
Columbia River Estuary

Unique estuary
- High river flow
- Strong tides
- Eastern boundary current





Challenging to model
- Strong currents (>3 m/s)
- Sharp density gradients
- Complex bathymetry
  - Wetting-drying
- Narrow continental shelf
  - Depth up to 2.8 km

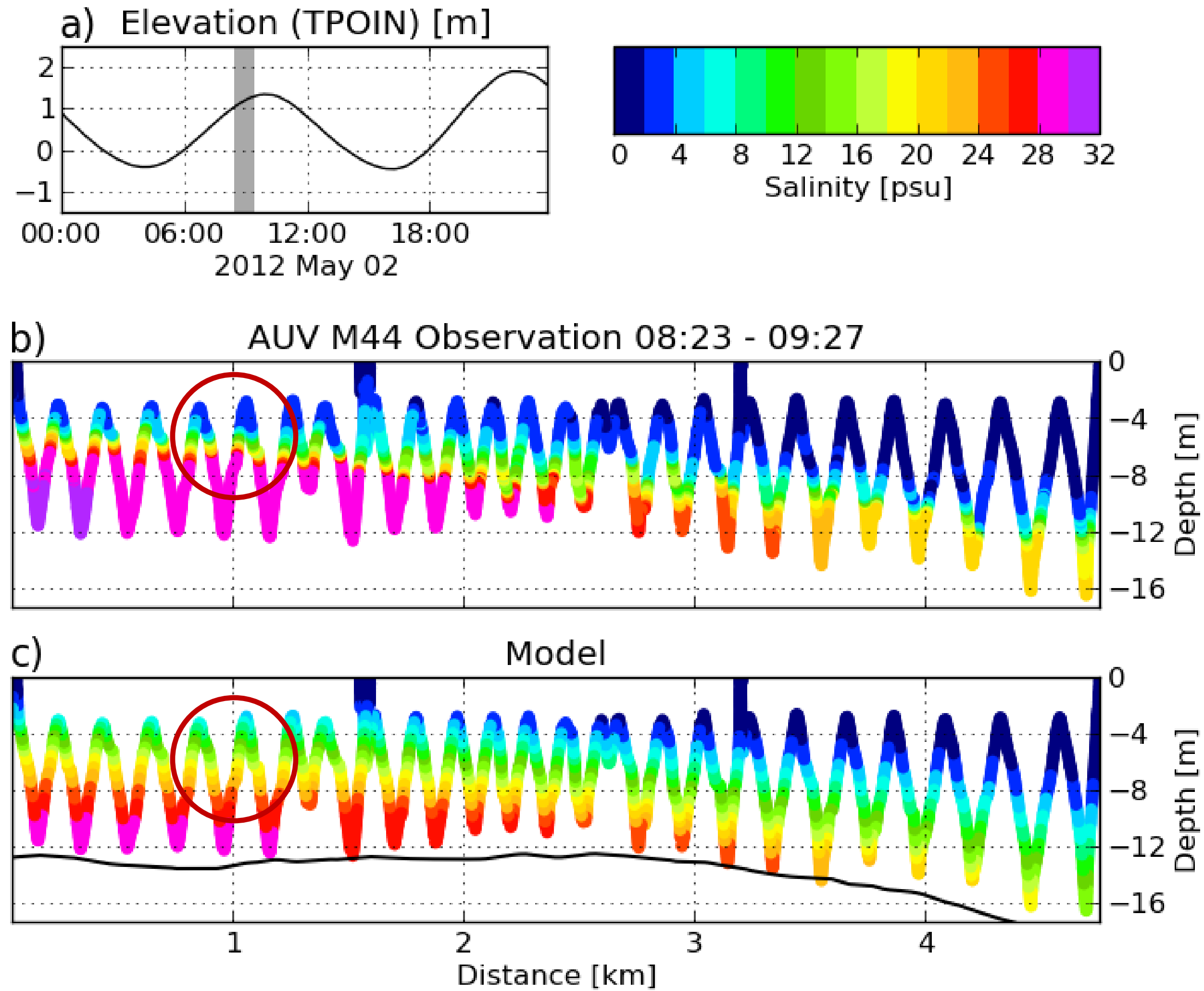


SELFE – AUV data comparison

High flow, neap tides

Salinity comparison

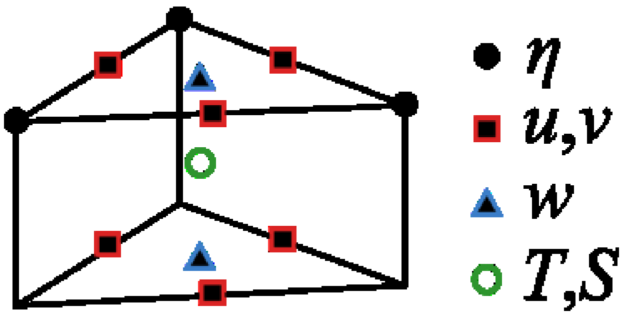
- Model diffuses sharp density gradients
- This changes the physics of the flow



Current circulation model: SELFE

Key limitations:

- SELFE is low order
  - Velocity p1nc, Elevation p1, Tracers p0
  - Low order quadrature rules
  - Low order time integration
  - Numerical dissipation



- SELFE is a low-level Fortran code
  - Fixing discretization implies rewriting most of the code
  - Laborious, error prone

-> Develop an alternative model using most prominent recent technology

Model development is risky business

- Initial choices of methods affect implementation
  - May render code base difficult to change later on
  - Especially for highly optimized code
- Solution: software platform that offers
  - **Flexibility**
    - Allow massive changes in model formulation
    - Without excessive amount of work
  - No compromises in **computational efficiency**
    - Targeting current and emerging supercomputer technology

High-level abstractions and Automatic Programming

- High-level abstractions
  - Intuitive and flexible interface
  - Rapid model development
  - Typically result in **slow code**
- Automatic Programming
  - Generates efficient low level (C) code for executing critical routines
  - Optimized, application specific code
  - Amortizes computational cost of high-level interface

Firedrake FE modeling framework

- Universal Form Language UFL (FENiCS)
  - Symbolic language for defining/manipulating weak forms
- FENiCS form compiler FFC (FENiCS)
  - Compiles forms to efficient C code
- PyOP2
  - Evaluates forms on unstr. mesh and assembles systems
  - Hardware agnostic (MPI, OpenMP, OpenCL, CUDA)

Automatic Programming

- Problem solved with PETSc
  - Extremely flexible solver library
- Firedrake (firedrakeproject.org)
  - Generic FE solver framework

Firedrake

Advantages

- User can change
  - Equations (add/remove/change terms)
  - Spatial discretization (elements, order, quadratures)
  - Time integration (implicit/explicit, RK/IMEX)
  - Solver options (lin/nonlin solvers, preconditioners)
  - Target hardware (MPI/OpenMP/GPU)
- **Takes literally minutes to change formulation**
- **Generates efficient code**

Example: Linear shallow water equations


```
mesh = Mesh('stommel_square.msh')
U = VectorFunctionSpace(mesh, 'CG', 2) # for uv
H = FunctionSpace(mesh, 'CG', 1)       # for eta
W = MixedFunctionSpace([U, H])
w, v = TestFunctions(W)
uv_tri, eta_tri = TrialFunctions(W)
sol_old = Function(W)
sol_new = Function(W)
uv, eta = split(sol_old)

g = Constant(9.81); h = Constant(1000.0); dt = 3.5
a = (1.0/dt)*(inner(w, uv_tri)+inner(v, eta_tri))*dx
L = g*inner(w, grad(eta))*dx -h*inner(uv, grad(v))*dx
solve(a == L, sol_new,
      solver_parameters={'ksp_type':'fgmres'})
```

Example: Switch to Discontinuous elements


```
H = FunctionSpace(mesh, 'CG', 1)
pres_grad = g*inner(w, grad(eta))*dx
```

- Change elements
- Integrate terms by parts
- Add interface terms with stabilization

```
H = FunctionSpace(mesh, ‘DG', 1)
pres_grad = -g*inner(div(w), eta)* dx
pres_grad += g*inner(jump(w,n), avg(eta))* dS
```

Example: use openMP instead of MPI


```
mpiexec ... python script.py
```

- Set environment variables

```
export PYOP2_BACKEND=openmp
export OMP_NUM_THREADS=8
mpiexec ... python script.py
```

- Will re-compile the code for openMP

Example: Adjoint model

- Assume you have defined a system

```
F = 0, F=F(...,p,...)
```

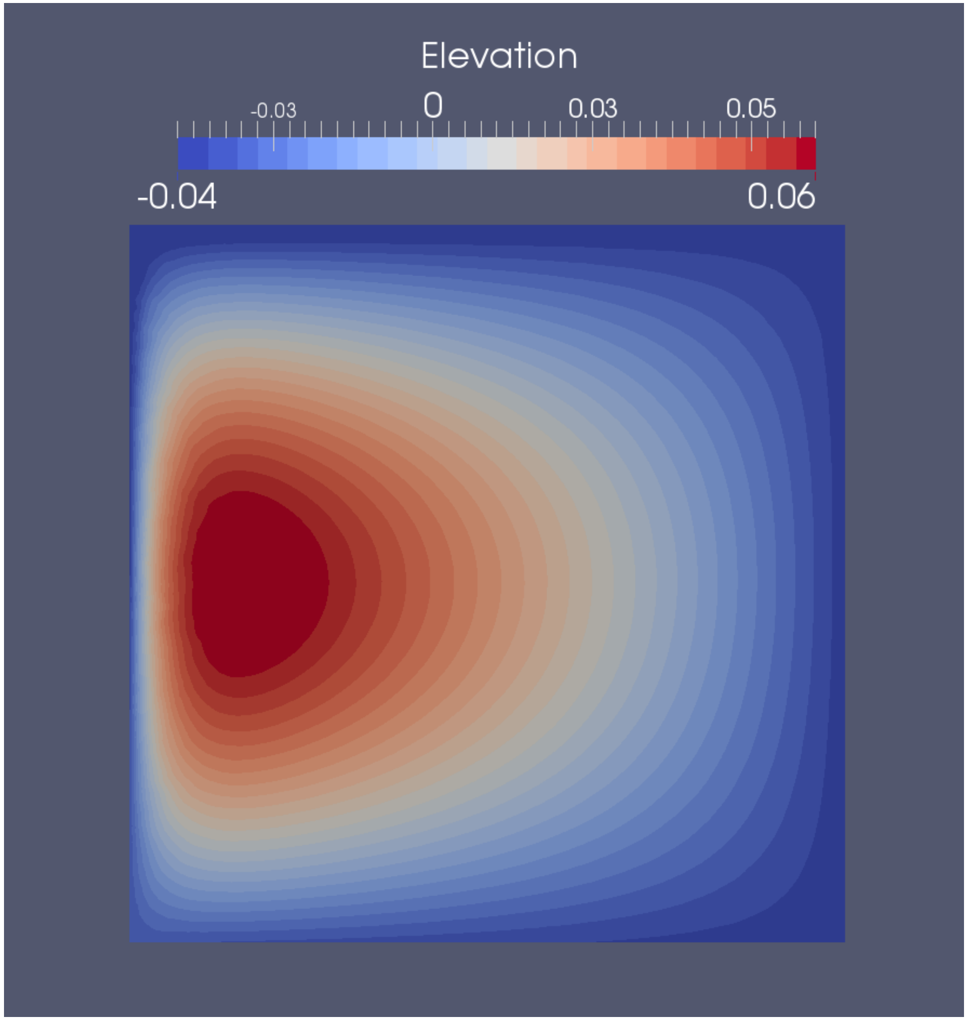
- F can be differentiated versus any arguments

```
dFdp = derivative(F,p)
```

- dFdp can be evaluated as any other form
- Leads to code *optimized to solve the gradient*

2D shallow water : Stommel test case

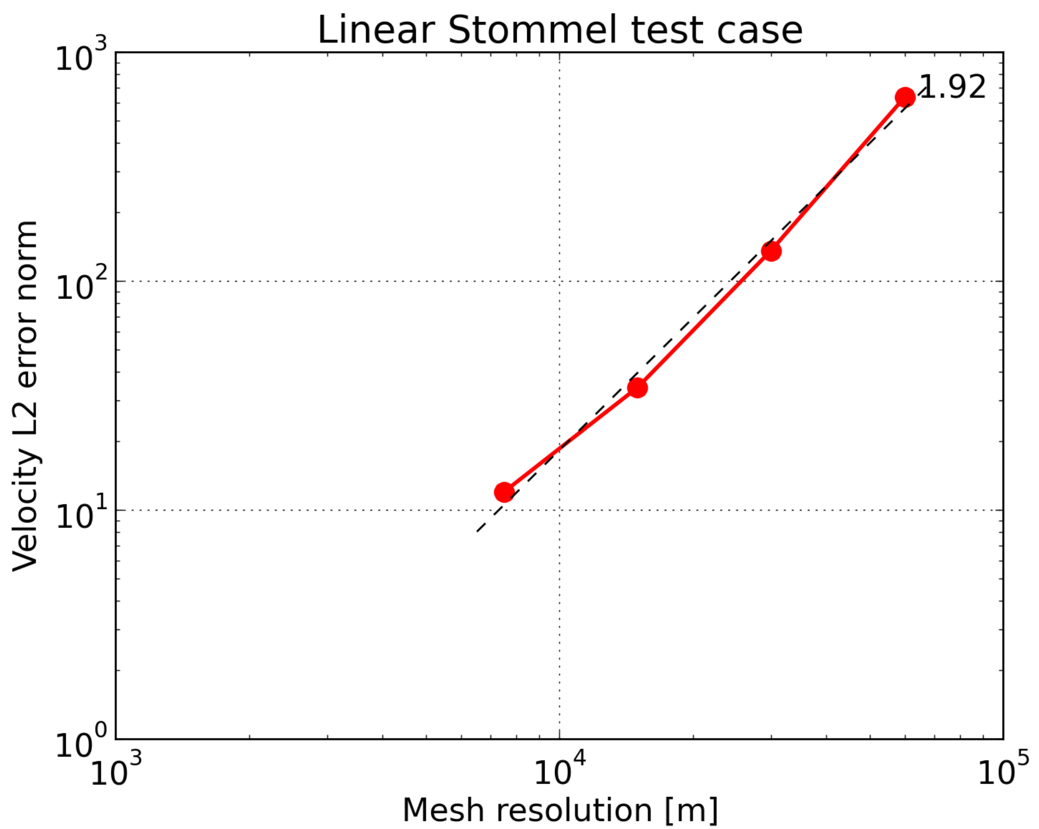
- 1000 km x 1000 km basin
- 1 km deep
- Impermeable boundaries
- Beta-plane Coriolis approx.
- Zonal wind stress =>
  Geostrophic gyre
- Intensifies on western
  boundary
- Shallow water equations
- Linear / Non-linear



2D shallow water : Stommel test case

Spatial convergence test

- Linear equations
- p1dg-p2 elements
- Crank-Nicholson time integration
- Element sizes: 60km to 5km
- Second order convergence



Simple run-time comparison

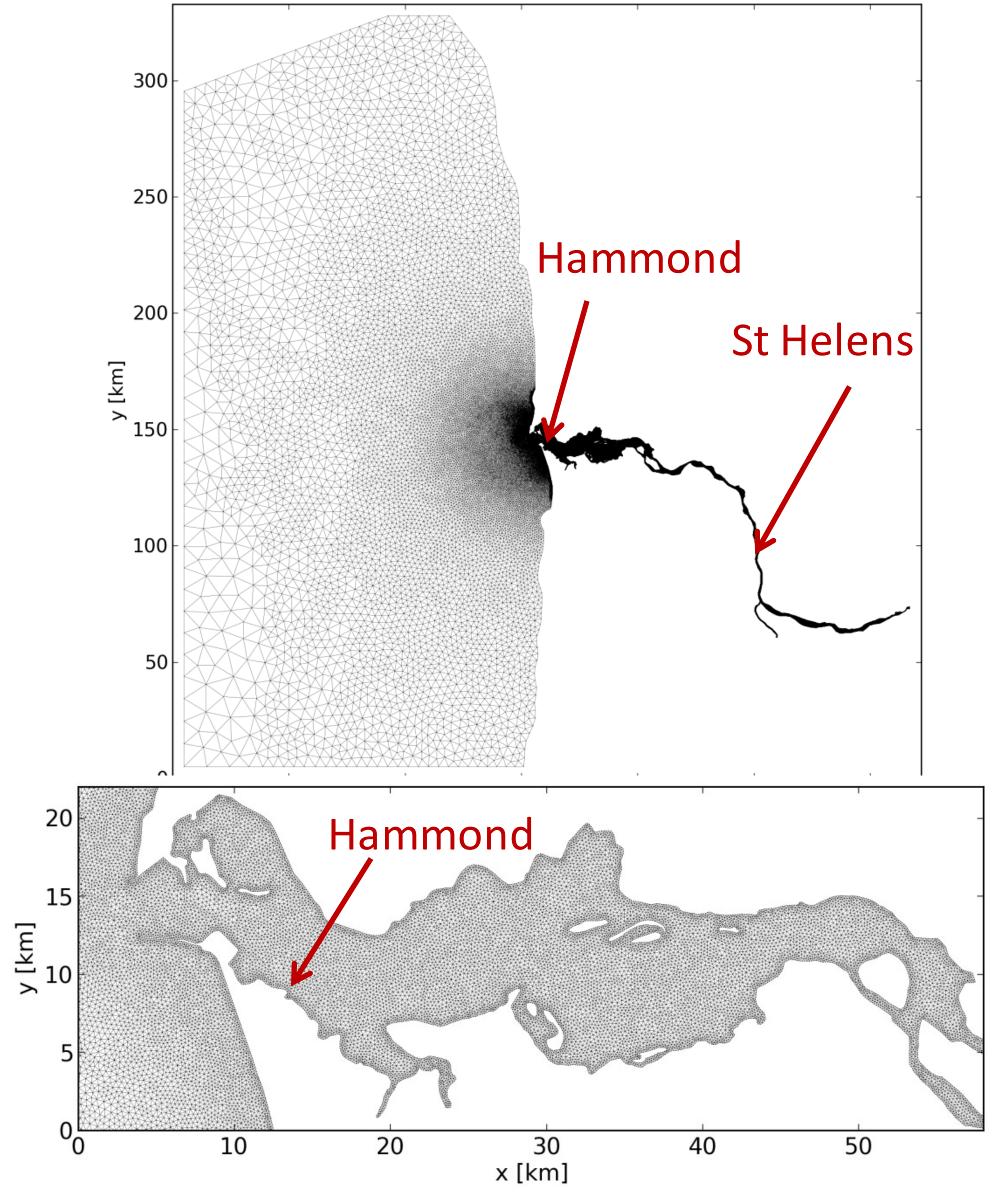
- Linear Stommel test case
  - Discontinuous elements, p1dg-p1dg
  - 26k element mesh
  - Forward Euler time integration, dt = 3.5 s
  - Time 1000 iterations on 1 CPU
- SLIM : **89.8 s**
  - No linear system to solve (precomputed inv. mass matrix)
- Firedrake implementation: **84.5 s**
  - Solver: PETSc with block Jacobi preconditioner

Application to Columbia River

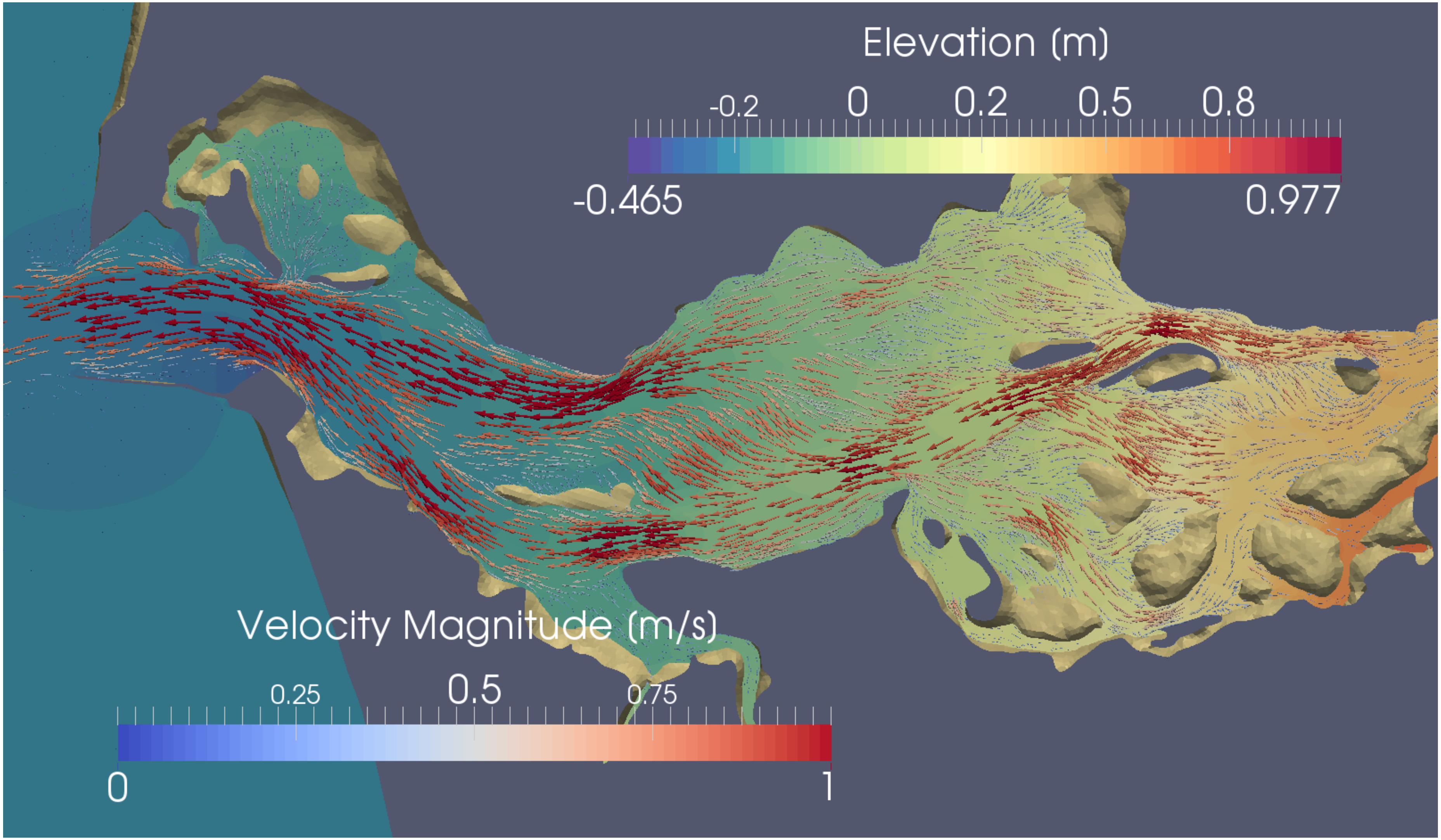
- Shallow water equations
- Coriolis forcing
- Quadratic bottom friction
- Wetting-drying
- p2-p1 elements
- 3rd order DIRK time integration
- 76k triangles, 40k nodes

Forcings

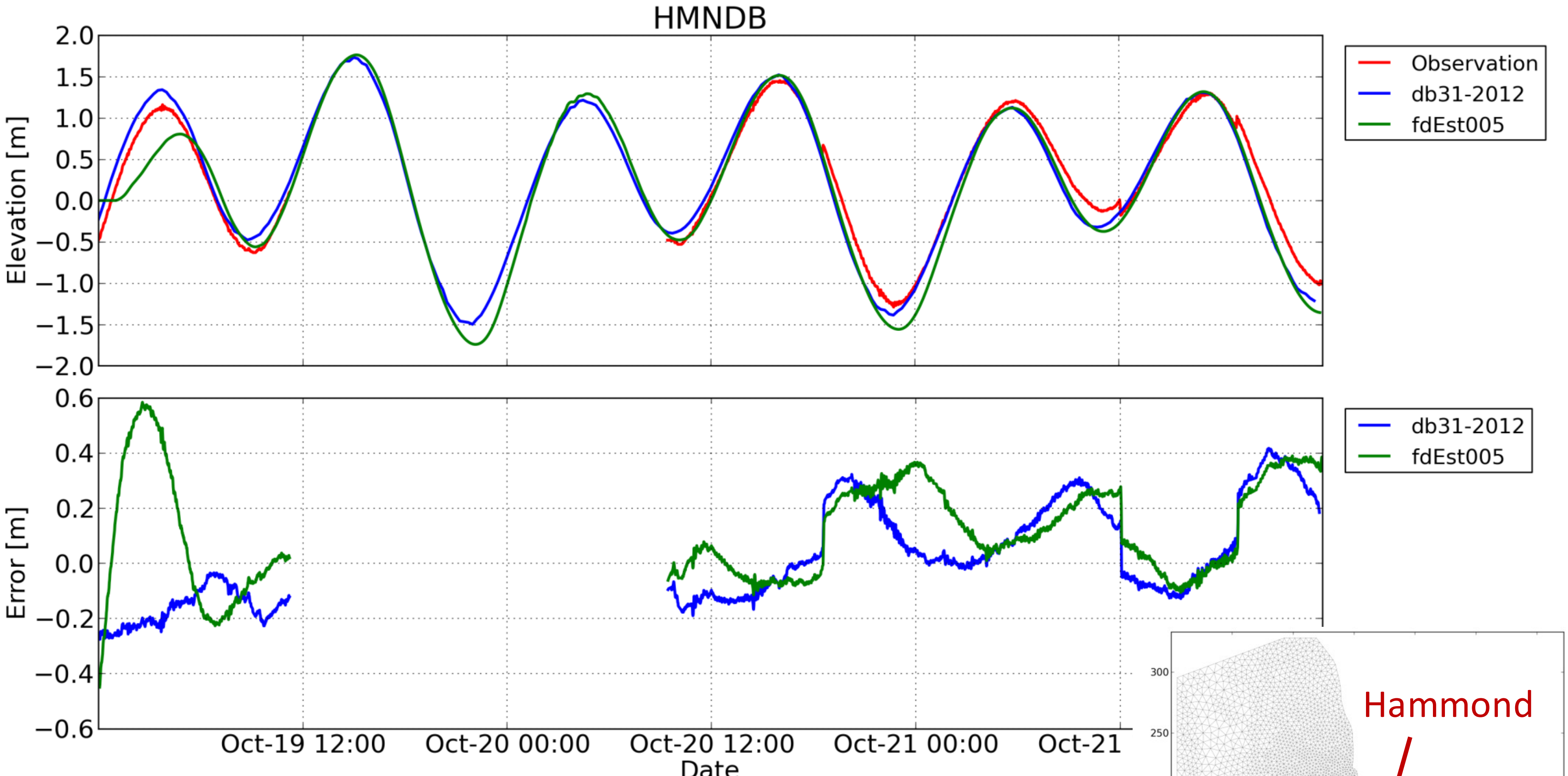
- River flux from data
  - Columbia, Willamette
- Ocean boundary
  - SELFE hindcast

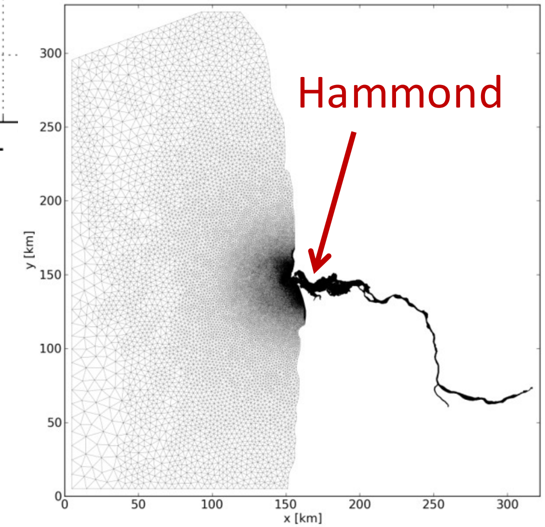


Application to Columbia River

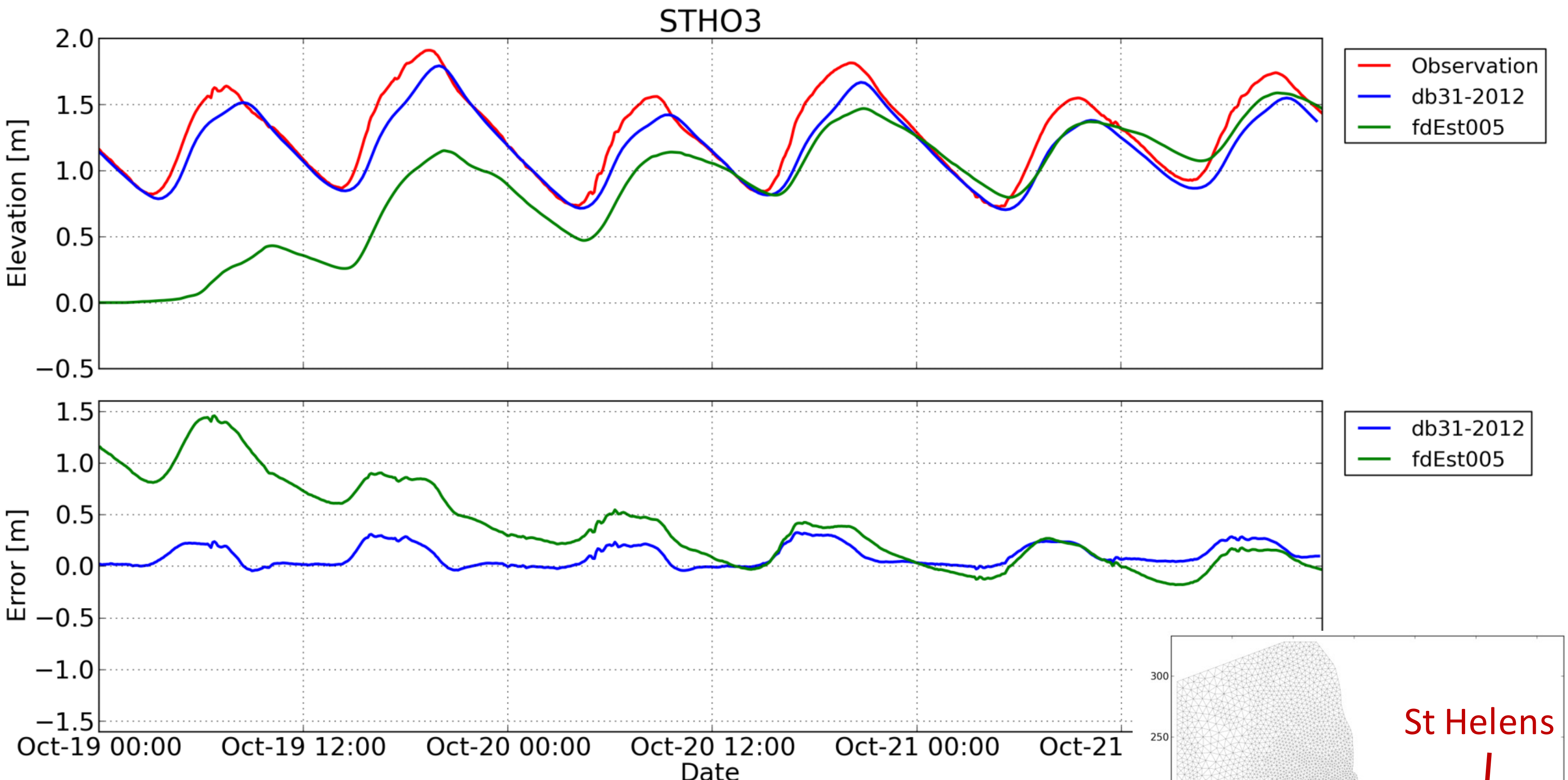


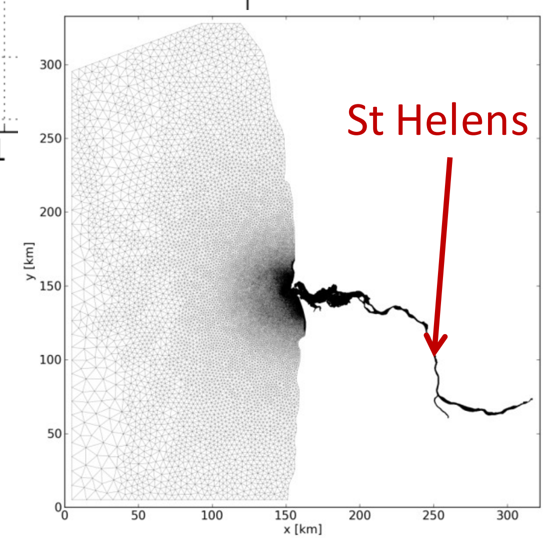
Elevations at Hammond (lower estuary)





Comparison of elevations at St Helens (river)





Goal: Three dimensional coastal model

- 3D prismatic mesh
  - Vertically moving mesh to track free surface
  - Arbitrary Lagrangian Eulerian formulation
- Wetting-drying
- Strict volume and tracer mass conservation
- Monotonic tracer advection scheme
- Generic length scale turbulence model
- Explicit/Semi-Implicit time integration

Conclusions

- Columbia River estuary application is demanding
  - We are seeking a new regional circulation model
  - Ability to capture sharp gradients is critical
    (see Baptista talk tomorrow)
- Flexibility and computational efficiency
  - High-level abstractions + automatic programming
- Firedrake project looks promising
- 3D tests to be done / in progress

Acknowledgements

Firedrake

http://firedrakeproject.org

PyOP2

http://op2.github.com/PyOP2

XSEDE

Extreme Science and Engineering Discovery Environment

FEniCS project

http://fenicsproject.org

TACC

PETSc

http://www.mcs.anl.gov/petsc/

Thanks to:
David Ham, Collin Cotter, Lawrence Mitchell, Florian Rathgeber, Imperial College London