Dynamical Two-way Nesting in HBM

Per Berg

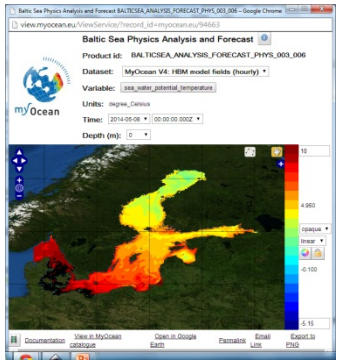
Danish Meteorological Institute (DMI)

Contents:

* Introduction to HBM and some applications involving nesting.
* Details on nesting, attempt to shed some light on what we do.

HBM: HIROMB-BOOS Model

myocean.eu

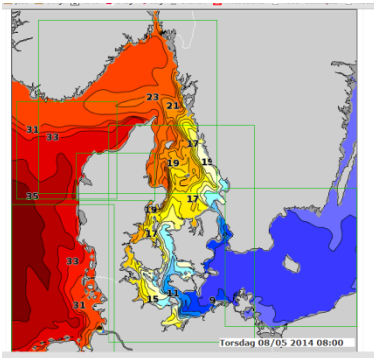


- An ocean circulation model code (i.e. solves IBVP)

- Different applications: Same code build, different input data.
  Regional operational forecast model at DMI:
  - DMI is responsable for storm surge warning in DK
  - MyOcean Baltic MFC --> Copernicus Baltic MFC

  Research: e.g. PanEU, Baffin Bay

dmi.dk



- Features two-way dynamical nesting
  (subject of this talk)

- Continuously being developed and validated by the
  HBM partners: DMI (repo), BSH, MSI, FMI, plus third-parties MoU

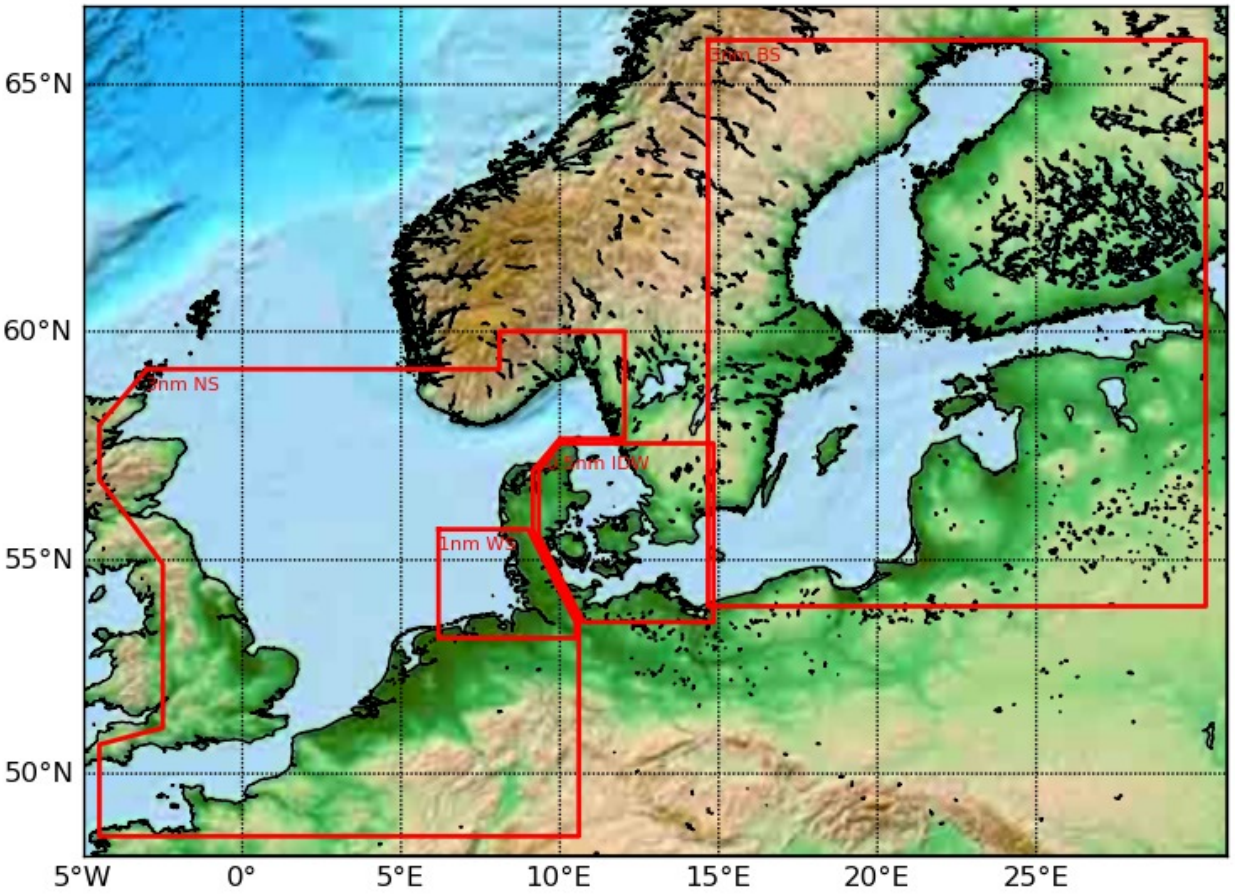
Examples of Applications:

Storm Surge Warning and MyOcean/Copernicus

Regional operational forecasting:

same code, different focus, different input

- Water levels predictions at Danish coasts, 4 times per day
- Ocean state of Baltic, physics and bio-geo-chemical, 2 times per day



Pan-European setup ... thanks to Lars Jonasson and Jens Murawski

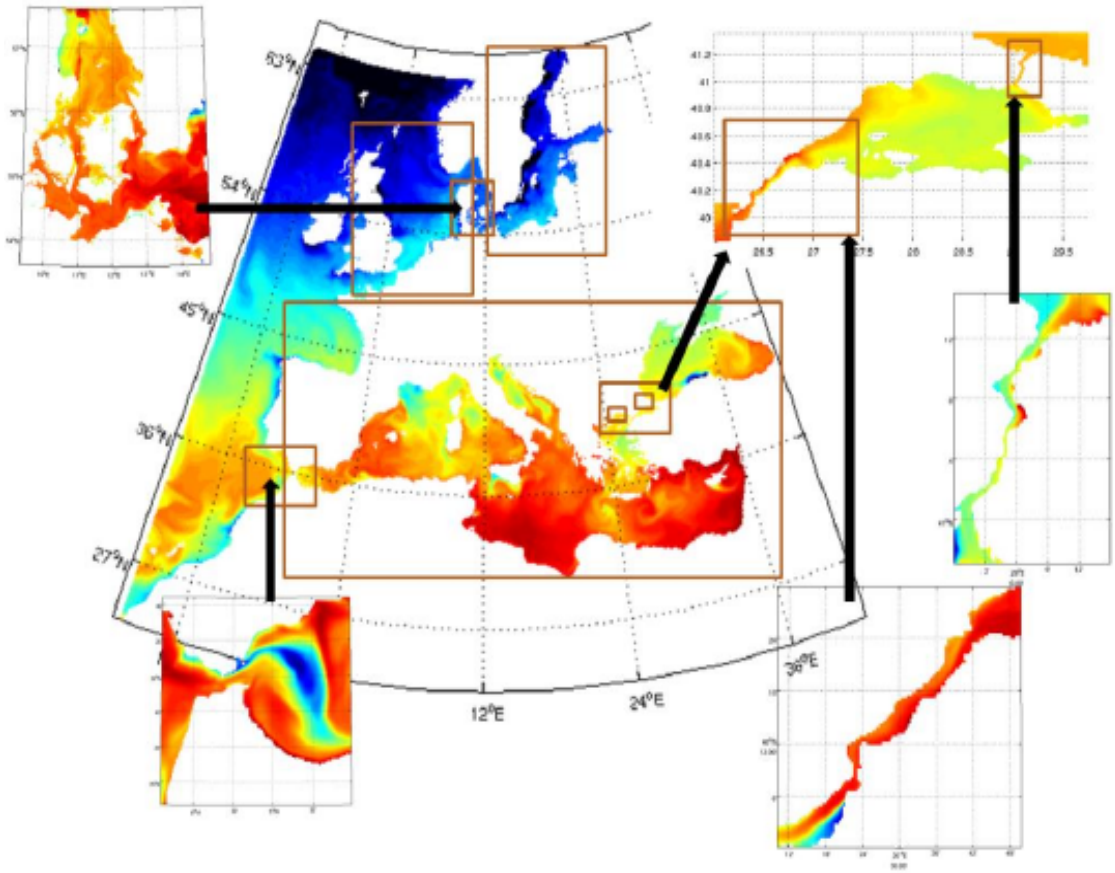
For research purposes (so far ...)

9 two-way nested domains:

Bosphorus / Dardanelles Straits:	~0.1 n.m.
Inner Danish waters:	~0.5 n.m.
Marmara Sea / Gibraltar / Baltic Sea:	~1 n.m.
North Sea / Shelf / Med.Sea / Black Sea:	~3 n.m.

Vertical resolution down to 1 meter, up to 122 layers

- For operational *forecasting* on a pan-European scale
- Address pan-European and regional aspects of *climate* change
  - basis for coupled model



Why nesting?

Nesting in HBM is

- a dynamical two-way exchange of mass and momentum at model time step level
- a practical means of setting up models with different local demands on "resolution properties" such as:

  - higher horizontal resolution,
  - higher vertical resolution,
  - larger toplayer size,
  - smaller time step size

  in different parts of the modelled region.

With nesting in HBM it becomes feasible to run models operationally that would otherwise be too computationally demanding, e.g. using the smallest grid spacing and/or the smallest time step size throughout the entire domain.

Alternatives to nesting (two-way nesting, dynamical nesting) could be

- curvi-linear grids,
- unstructured grids,
- or even regular structured grids which are computationally decoupled from each other with one-way transfer boundaries between.

Dynamical two-way nesting in HBM

Basic Fundamentals

"control volume formulation"

- Arakawa C-grid as basis
- scalar variables representing grid-cell averages and velocity components representing average values on the grid-cell faces
- formulate transport for mass and tracers as budget equations for each grid-cell with exchange to neighbor grid-cells across grid-cell faces

In principle, dynamical two-way nesting is no exception to this!

- we implement the nesting on the finite difference scheme level

For practical reasons, we use

- a number (up to 99) of larger areas of same “resolution properties” glued together by "**nesting**" along borders, plus some restrictions on allowed jumps in “resolution properties” (integer multiples)

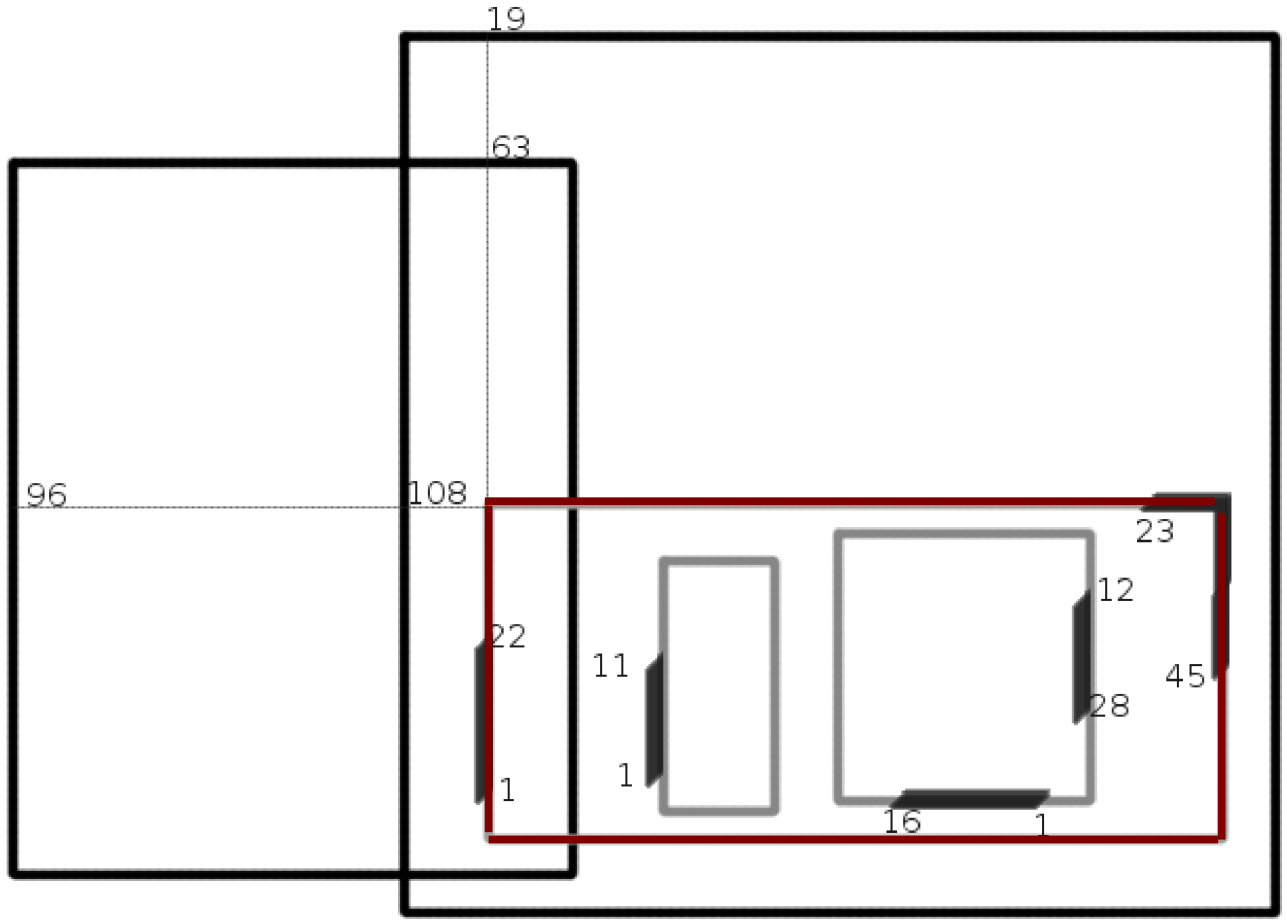
- description of nesting for a specific setup is defined by the user at runtime

Set up two-way nested models

Example with 5 areas, specify borders:

Red area is inclosed into two **black** areas and is enclosing two **grey** areas

--- Not like Chinese boxes!



Generalization of nesting implementation

Example, solve for momentum+mass, pseudo-code:

Inside time-loop of main-prg:

```
call SolveHydrodynamics ( timelevel(mainarea) )
```

where:

```
subroutine SolveHydrodynamics ( level )
  do itl=1,max_iterations_on_this_level
    do ia=1,narea
      if (onthislevel(ia,level)) call SolveMomEq (ia,itl)
    enddo

    call SolveNextLevel ( level+1 )

    do ia=1,narea
      if (onthislevel(ia,level)) call SolveMassEq (ia,itl)
    enddo
  enddo
end subroutine SolveHydrodynamics
```

using recursion of time levels:

```
subroutine SolveNextLevel ( level )
  if (level <= maxtimelevels) call SolveHydrodynamics(level)
end subroutine SolveNextLevel
```

Generalization of nesting implementation

momentum+mass, pseudo-code continued:

```
subroutine SolveMomEq ( ia, ... )
  call turbulence_model( ... )
  call momeqs( u(ia), v(ia), un(ia), vn(ia), ... )
    ...
  do iao=1, No_of_areas_nesting_from_ia
    !  ia: inclosing domain, iia: enclosing domain
    iia = nestingfrom(ia,iao)

    call mom_c_f( u(ia), v(ia), un(ia), vn(ia), ...  &
                  un(iia), vn(iia) )

  enddo
end subroutine SolveMomEq
```

Generalization of nesting implementation, pseudo-code

momentum+mass, continued:

```
subroutine SolveMassEq ( ia, ... )
  do ii=1,nestinglevels(ia)
    ! iia: enclosing domain, ia: inclosing domain
    iia = nestingto(ia,ii)

    call mom_f_c (un(ia), vn(ia), un(iia), vn(iia), ...)
  enddo

  !  solve mass equations for zn at interior points:
  call masseqs(z(ia), zn(ia), un(ia), vn(ia), ...)

  do iao=1, No_areas_nesting_from_ia
    !  ia:  inclosing,  iia: enclosing
    iia = nestingfrom(ia,iao)

    call z_c_f( zn(ia), z(ia), un(ia), vn(ia),      &
                un(iia), vn(iia), ... )
  enddo
end subroutine SolveMassEq
```

Bottlenecks related to nesting

With our generalized approach, we *cannot* assume that points in the vicinity of a nesting border in the enclosing and in the inclosing areas reside on the same MPI task, or on the same OpenMP thread, not even when they are located in geographically overlapping regions.

Extended explicit communication patterns needed:

- barriers
- gathers
- encoding/decoding

plus increased difficulties when

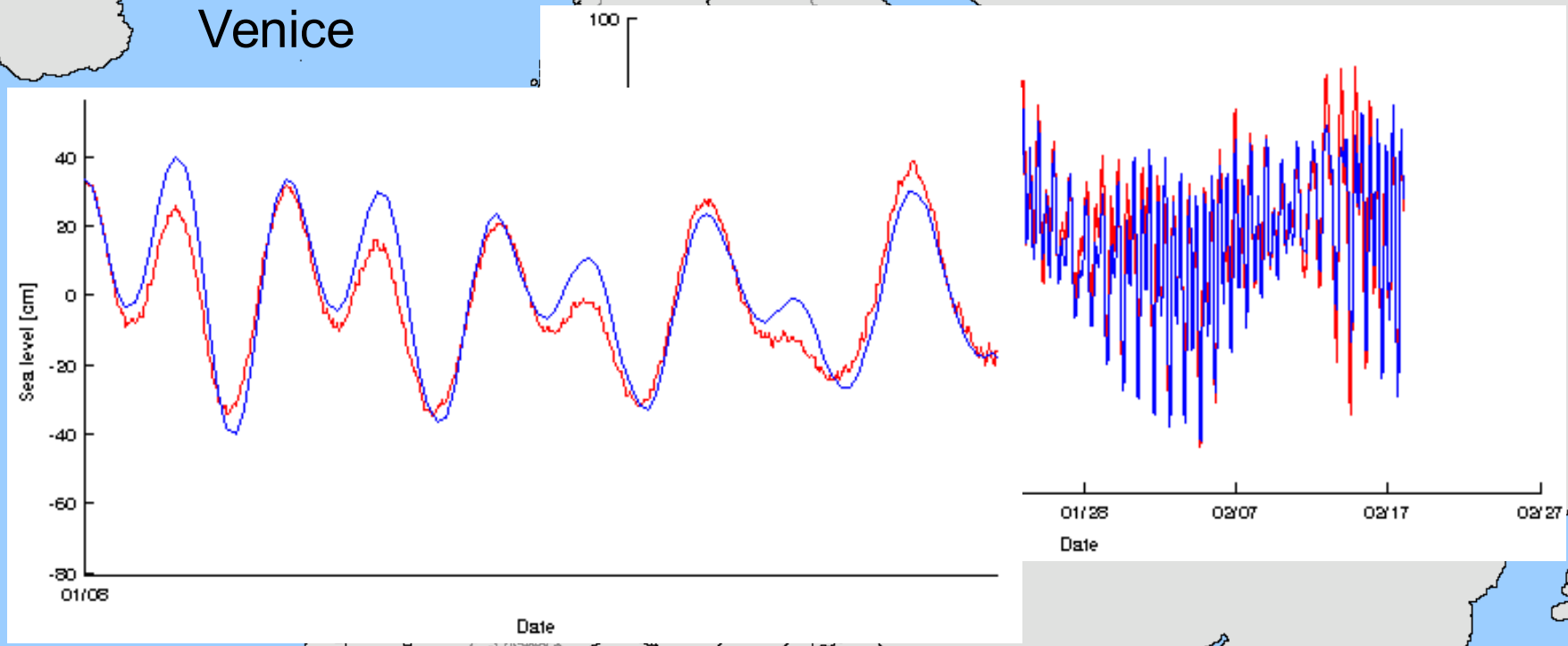
- keeping accessed data contiguous and unit stride
- maintaining proximity of accessed data
- doing efficient SIMD vectorization

Demo example:

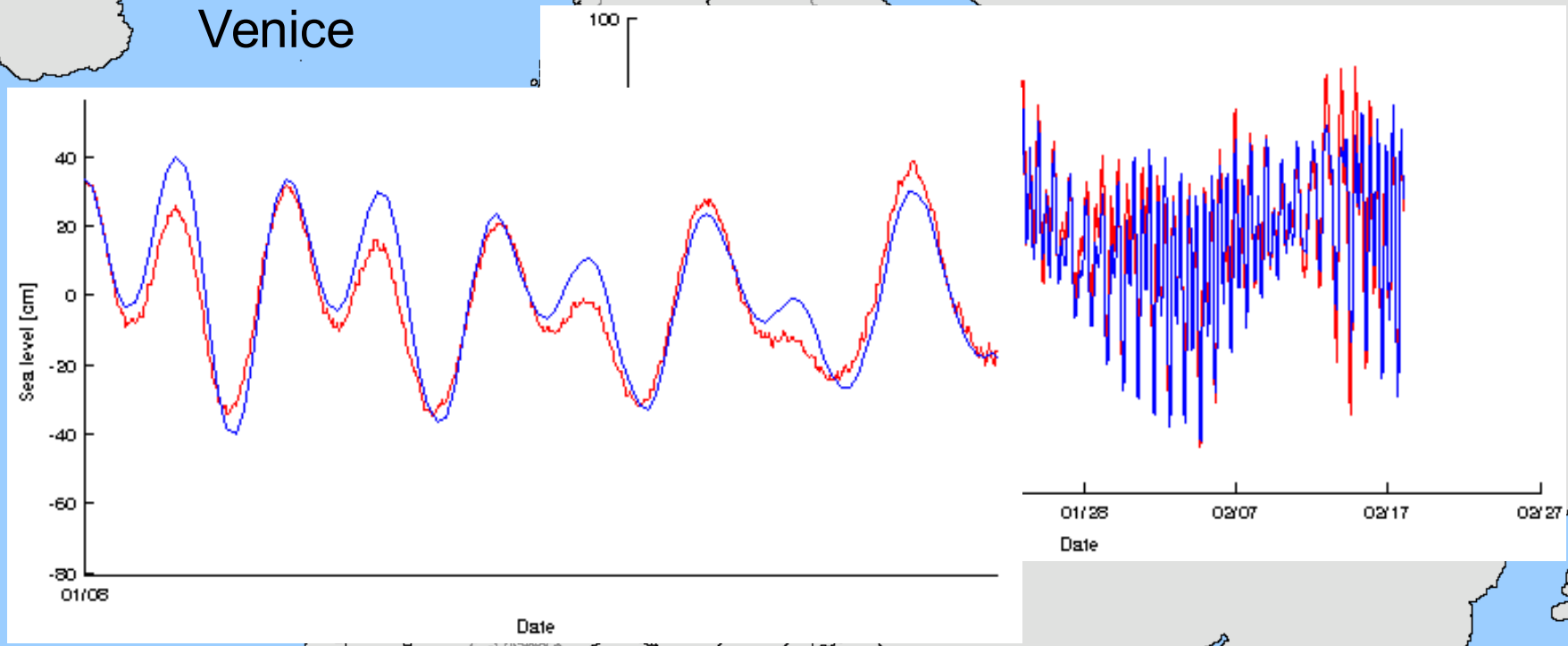
Regional forecasts on a
Pan-European scale
e.g. water levels

Interbasin connections
and
general circulation

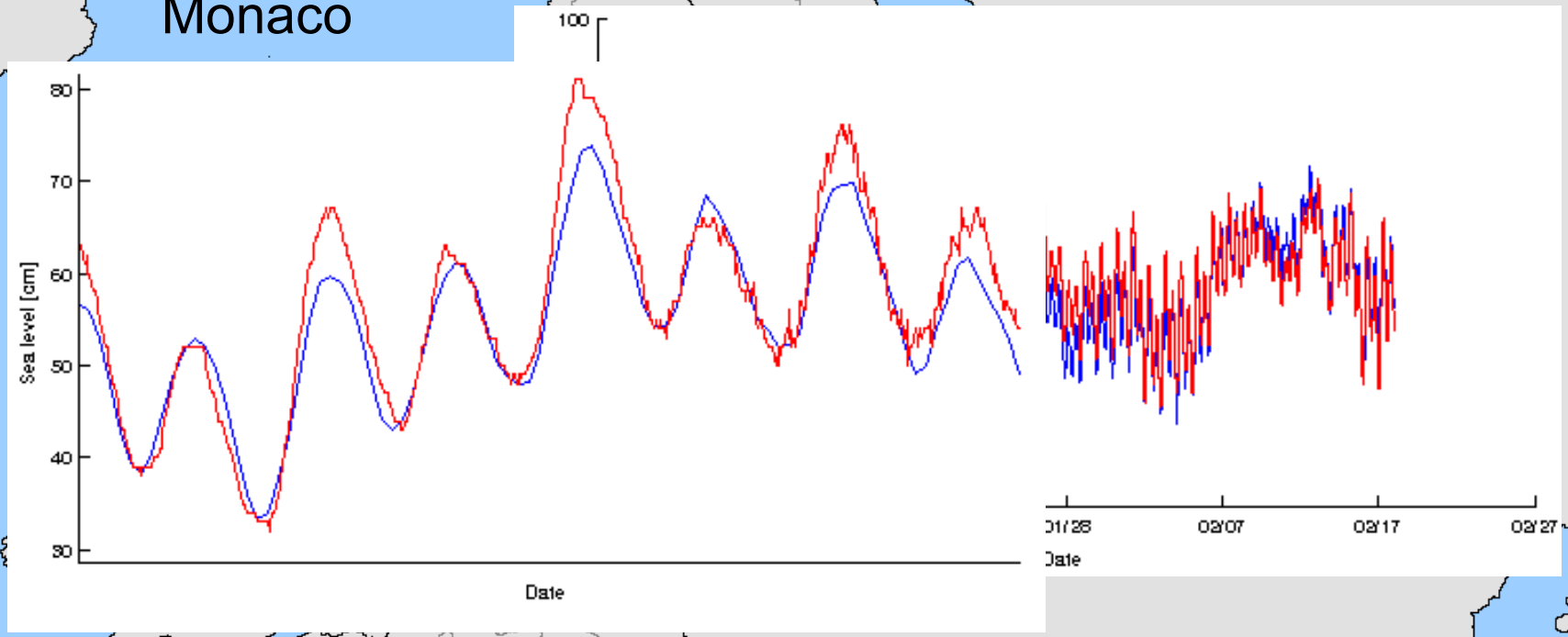
Venice

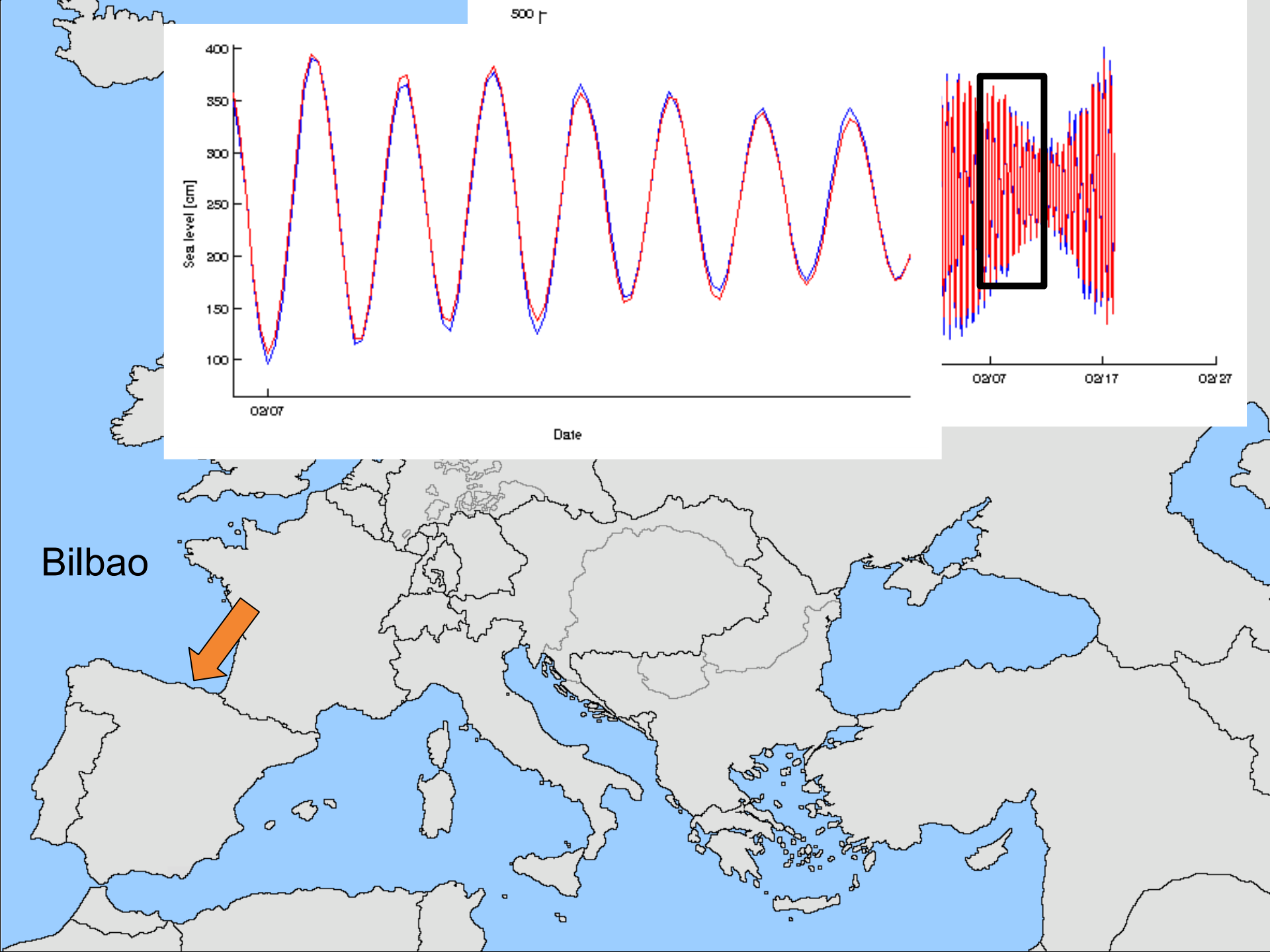

100
40
20
0
-20
-40
-60
-80
Sea level [cm]
01/06
Date
01/26
02/07
02/17
02/27
Date

Venice

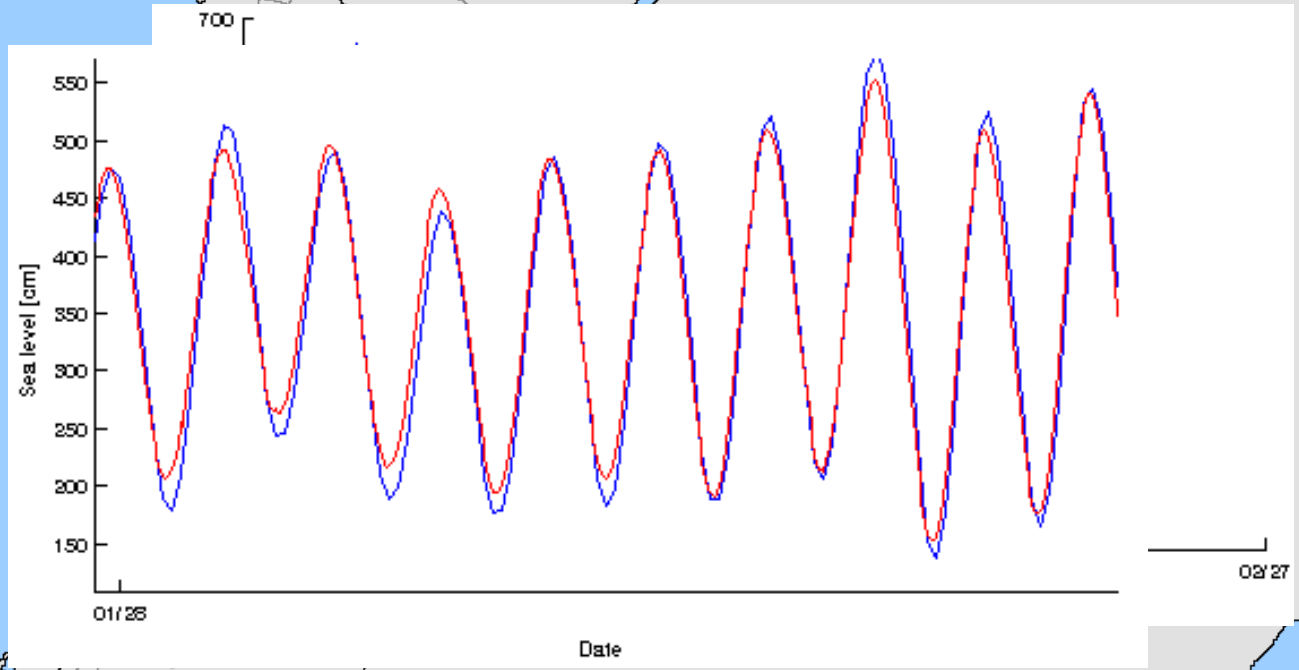

100
40
20
0
-20
-40
-60
-80
Sea level [cm]
01/06
Date
01/26
02/07
02/17
02/27
Date

Monaco

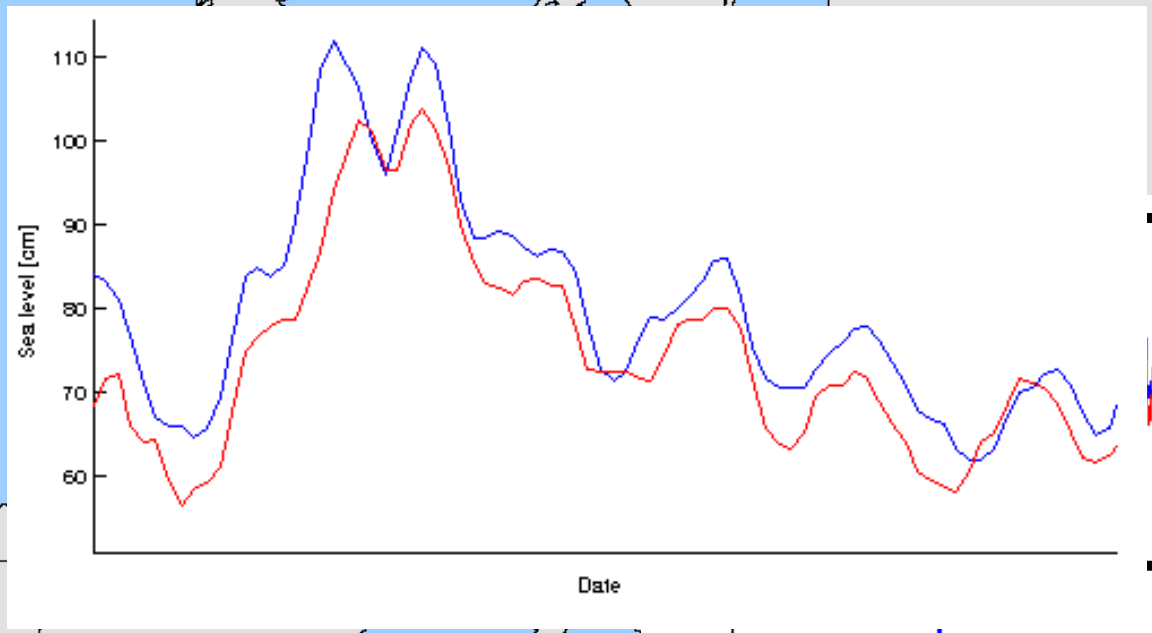


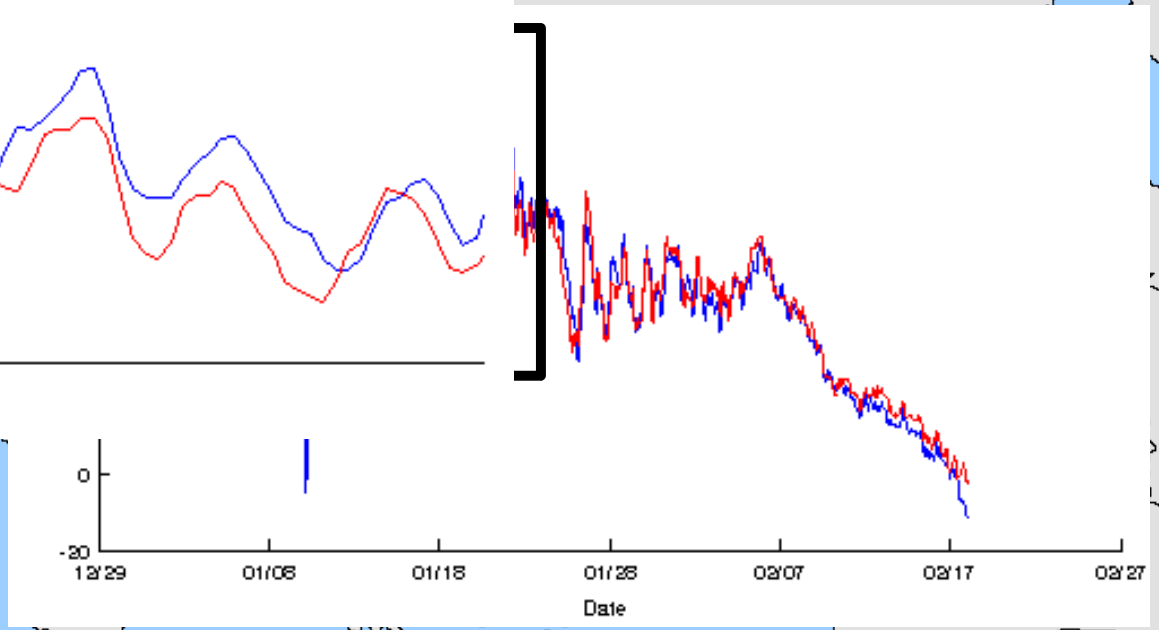

Bilbao
500
400
350
300
250
200
150
100
Sea level [cm]
02/07
Date
02/07
02/17
02/27

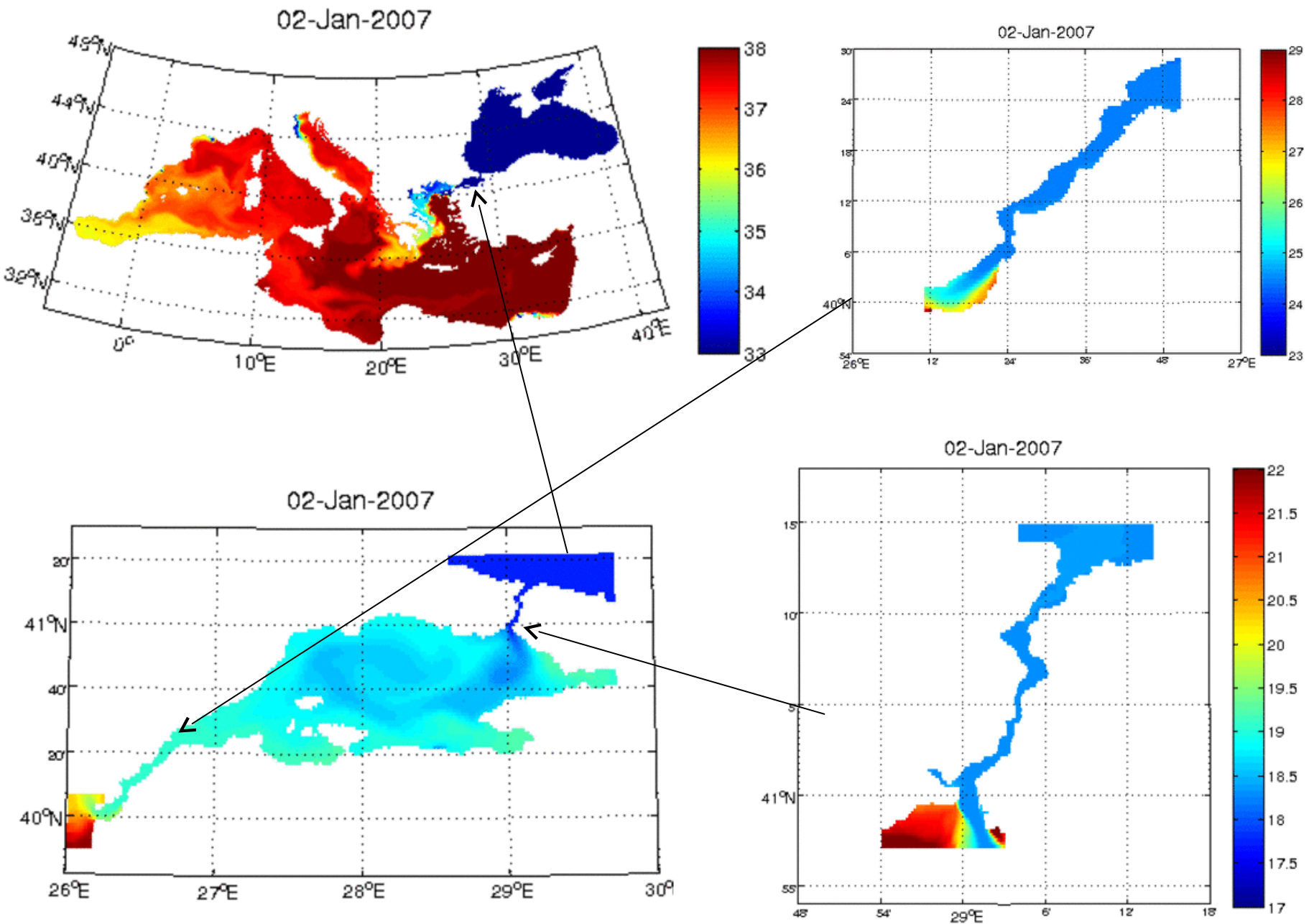
Whitby

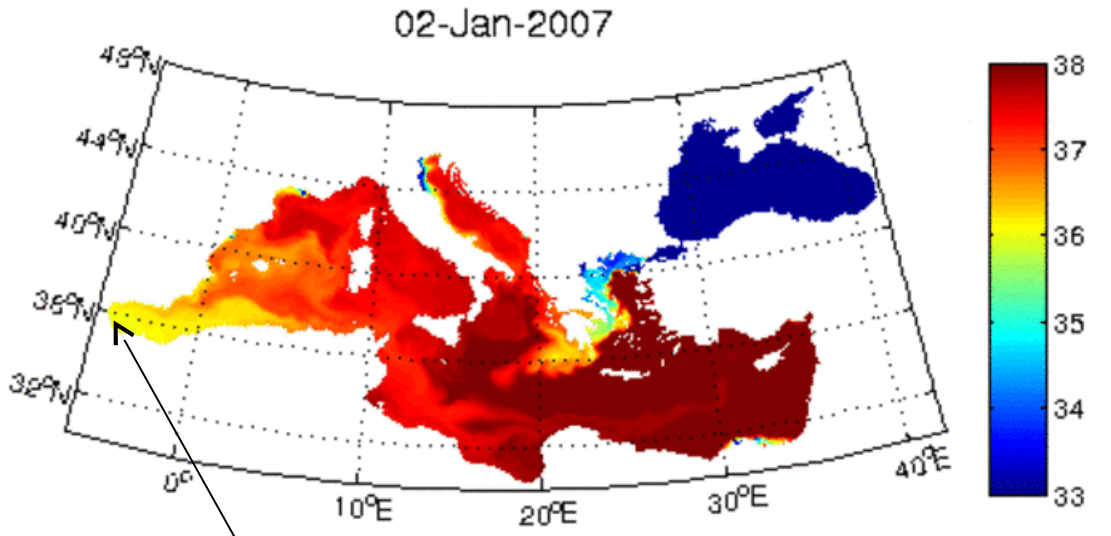

700
550
500
450
400
350
300
250
200
150
Sea level [cm]
01/26
02/27
Date

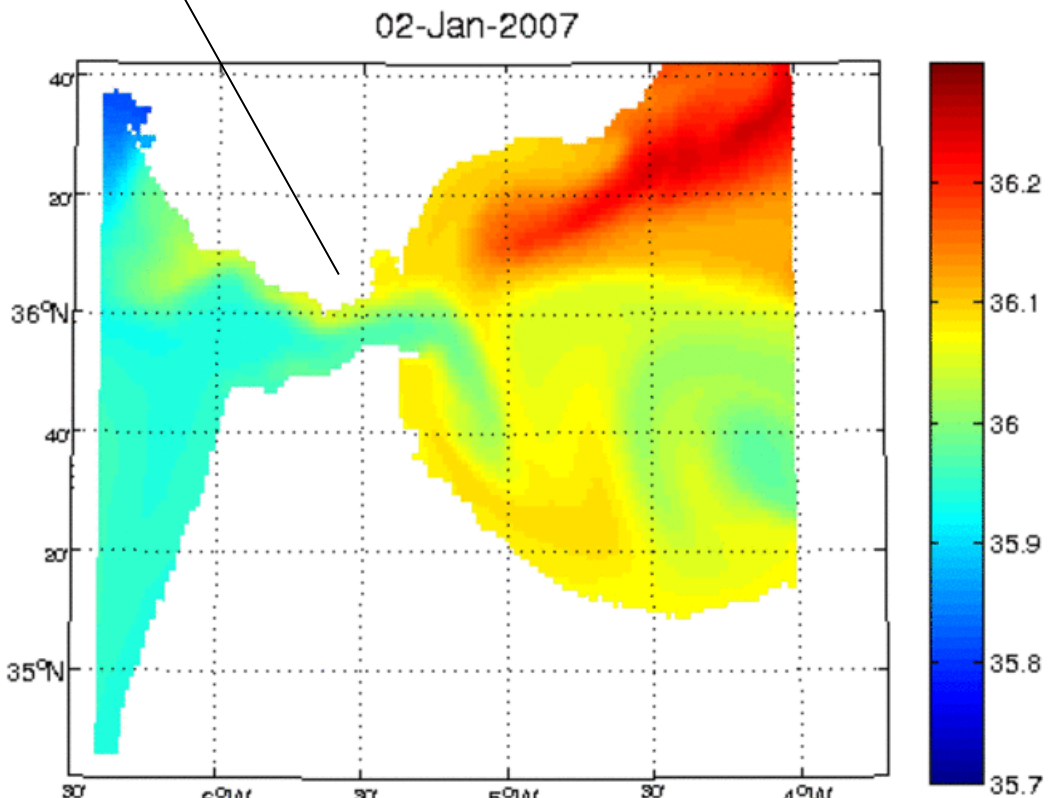
Stockholm


110
100
90
80
70
60
Sea level [cm]
Date


0
-20
12/29
01/08
01/18
01/28
02/07
02/17
02/27
Date


02-Jan-2007
02-Jan-2007
02-Jan-2007
02-Jan-2007


02-Jan-2007
45°N
40°N
35°N
30°N
0°
10°E
20°E
30°E
40°E
38
37
36
35
34
33


02-Jan-2007
40'
20'
36°N
40'
20'
35°N
36.2
36.1
36
35.9
35.8
35.7

Time consumption for Pan-EU setup

On DMI's ancient in-house computer: ~2.5h / 5 day forecast

More modern H/W (Ivy Bridge): <1.5h / 5 day forecast

- both using a modest 360 cores
- demonstrated scaling up to ~1600 cores

OK for operational forecasting

kind of hopeless for climate simulations ... as is

Further reading:

Per Berg and Jacob Weismann Poulsen: "Implementation details for HBM"
http://beta.dmi.dk/fileadmin/Rapporter/TR/tr12-11.pdf

Per Berg: "Mixing in HBM"
http://www.dmi.dk/fileadmin/Rapporter/SR/sr12-03.pdf

Jacob Weismann Poulsen and Per Berg: "More details HBM – general modelling theory and survey of recent studies"
http://beta.dmi.dk/fileadmin/Rapporter/TR/tr12-16.pdf

Jacob Weismann Poulsen and Per Berg: "Thread scaling with HBM"
http://www.dmi.dk/fileadmin/user_upload/Rapporter/tr12-20.pdf

Jacob Weismann Poulsen, Per Berg and Karthik Raman: "Better Concurrency and SIMD on HBM"
Chapter 3 in "High Performance Parallelism Pearls -- Multicore and Many-core Programming Approaches",
Jim Jeffers and James Reinders (eds.),
Morgan Kaufmann, 2014, ISBN 978-0-128-02118-7

PRACE Final Report:
"Next generation pan-European coupled Climate-Ocean Model - Phase 1 (ECOM-I)"