EGU General Assembly 2019

Short Course SC1.1

Data assimilation in the geosciences –

Practical data assimilation with the Parallel Data Assimilation Framework

Lars Nerger[1], Maria Broadbridge[2],
Gernot Geppert[3], Peter Jan van Leeuwen[2,4]

1: Alfred Wegener Institute, Bremerhaven, Germany

2: University of Reading, UK

3: Deutscher Wetterdienst (DWD), Offenbach, Germany

4: Colorado State University, Fort Collins, CO, USA

PDAF Parallel Data Assimilation Framework

3

Hands-on Example: Build an Assimilation System with PDAF

Get the tutorial code

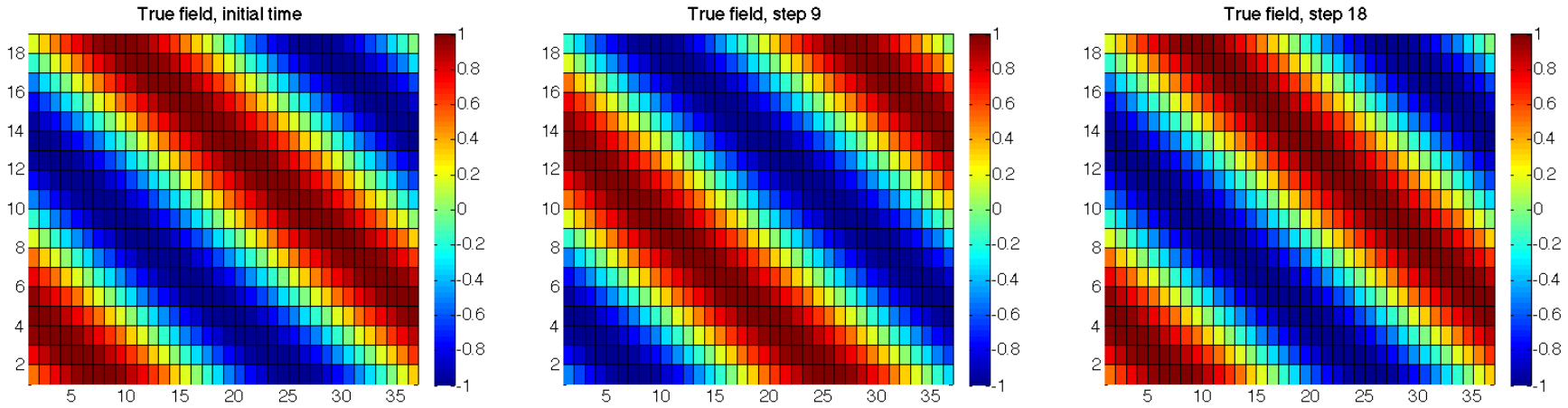
Download the tutorial

Directory layout:

`make.arch`	- build configurations
`src`	- source files
`tutorial`	
`online_2D_serial`	
`model`	- serial model code
`model_coupled_to_pdaf`	- final assimilation code
`pdaf`	- code to be added to the model
`online_2D_serial.noMPI`	- alternative code without MPI

2D „Model“

- Simple 2-dimensional grid domain
- 36 x 18 grid points (longitude x latitude)
- True state: sine wave in diagonal direction
  (periodic for consistent time stepping)
- Simple time stepping:
  Shift field in vertical direction one grid point per time step
- Output to text files (18 rows) – `true_step*.txt`



PDAF Parallel Data Assimilation Framework

General program structure: `model/main.f90`

```
program main
   initialize
   integrate
end program
```

initialize model information:
- set dimensions
- allocate model field array
- read initial field

perform time stepping
- shift model field
- write new model field

No parallelization!

Files in the tutorial directories

The model source code consists of the following files (`model/`):

- `main.F90`
- `mod_model.F90`
- `initialize.F90`
- `integrate.F90`
- `Makefile`

Files in the tutorial directories

The PDAF coupling code consists of (`pdaf/`)

- interface subroutines (called from the model code)
  - `init_parallel_pdaf.F90`
  - `init_pdaf.F90`
  - `assimilate_pdaf.F90`
  - `finalize_pdaf.F90`
- user subroutines (called from the PDAF library), eg.
  - `collect_state_pdaf.F90`
- “supporting” modules and subroutines (used in the interface and user subroutines), eg.
  - `mod_assimilation.F90`
  - `init_pdaf_parse.F90`

Running the tutorial model

- cd to `tutorial/online_2D_serialmodel/model`
- Set environment variable `PDAF_ARCH`

```
export PDAF_ARCH=linux_gfortran_openmpi
```

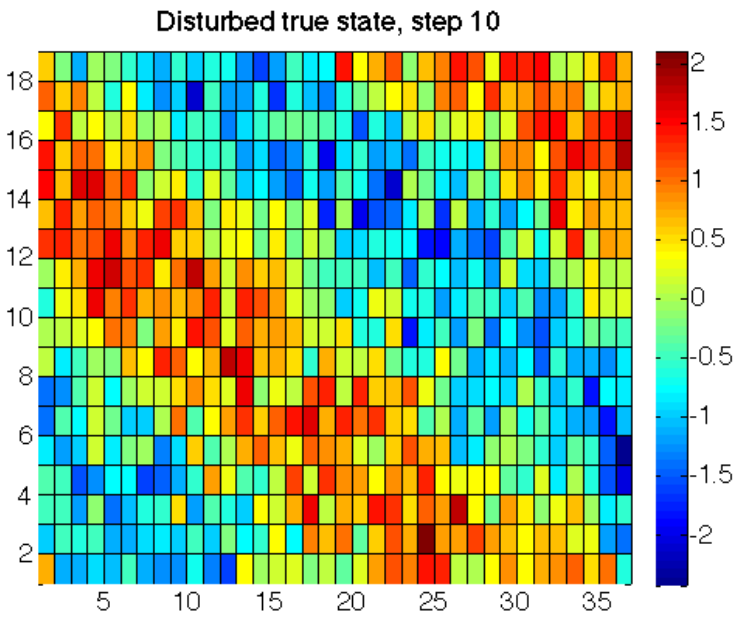
- Run `make`
- Run the model with `./model`

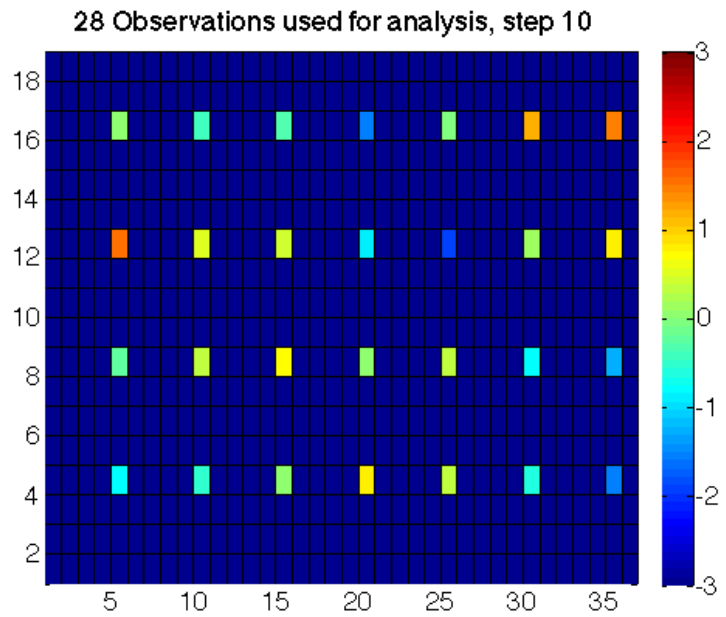
- Inputs are read in from `tutorial/inputs_online`
- Outputs are written in `tutorial/online_2D_serialmodel/model`

  eg. `true_step10.txt`

Observations

- Add random error to true state (standard deviation 0.5)
- Select a set of observations at 28 grid points
- File storage (in `inputs_online`):
  text file, full 2D field, -999 marks 'no data' – `obs_step*.txt`
  one file for each time step





PDAF Parallel Data Assimilation Framework

Coupling the model to PDAF: Online mode

- Combine model with PDAF into single program
  - modify `Makefile` to build `model_pdaf`

- Add 4 subroutine calls:

  `init_parallel_pdaf`- add parallelization

  `init_pdaf` - initialize assimilation

  `assimilate_pdaf` - perform assimilation

  `finalize_pdaf` - clean up

- Implement user subroutines, e.g. for
  - observation operator
  - initialization of observation vector
  - transfer between state vector and model fields

http://pdaf.awi.de/trac/wiki/OverviewOfUserRoutinesWithDefaultNames

Online coupling: Parallelization

- Online coupling avoids writing to disk to exchange state vectors between the model and PDAF
- Add MPI to the model to run several model instances in parallel
- Run the parallel version with

```
mpirun -np <n> ./model_pdaf ...
```

- *Alternative*: PDAF's "flexible" approach: http://pdaf.awi.de/ModifyModelForEnsembleIntegration
  - cd to `tutorial/online_2D_serialmodel.noMPI/model`

Files to copy from pdaf to model

```
init_parallel_pdaf.F90
  mod_parallel_pdaf.F90
  parser_mpi.F90

finalize_pdaf.F90

init_pdaf.F90
  mod_assimilation.F90
  init_pdaf_info.F90
  init_pdaf_parse.F90
  init_ens.F90
  next_observation_pdaf.F90
  distribute_state_pdaf.F90
  prepoststep_ens_pdaf.F90
```

parallelization

clean up

initialization

ensemble forecast

post step

PDAF interface subroutine - called from the model

helper module/subroutine for the interface

PDAF user subroutine - called from PDAF library

... (continued on next slide)

Files to copy from `pdaf` to `model`

... (continued from previous slide)

```
assimilate_pdaf.F90
   collect_state_pdaf.F90
   init_dim_obs_pdaf.F90
   obs_op_pdaf.F90
   init_obs_pdaf.F90
   prodrinva_pdaf.F90
```

- Each file contains a short summary what the subroutine does

Files to be adapted in `model`

`main.F90`	-	add calls to PDAF interface
`integrate.F90`	-	add calls to PDAF interface
`Makefile`	-	add linking to PDAF library, PDAF interface and user subroutines

- *Reference solutions for the modified files are in `model_coupled_to_pdaf`*
- When complete, run `make` again
- Then run

```
mpirun -np 9 ./model_pdaf -dim_ens 9
```

- Outputs are written to

```
ens_<i>_step<j>_for.txt
ens_<i>_step<j>_ana.txt
```

This runs a filter without localization with ensemble size 9

Plotting

- When your coupling is working, lookt at the results
- With Matlab/Octave you can use

```
load ens_01_step02_for.txt
pcolor(ens_01_step02_for)
```

- Or use the Python scripts

```
./plot_file.py ens_<i>_step<j>_for.txt
./plot_ens.py <i> <j>
```

More PDAF experiments

- Find PDAF command line parameters in

  `./pdaf/init_pdaf_parse.F90`

- Try for example

```
mpirun -np 4 ./model_pdaf -dim_ens 4
```

  (this runs a filter (ESTKF) without localization with ensemble size 4; it gives a worse result than ensemble size 9)

```
mpirun -np 9 ./model_pdaf -dim_ens 9 -filtertype 7
```

  (this runs a filter (LESTKF) with localization and localization radius 0, i.e. correcting only at observed grid points)

```
mpirun -np 9 ./model_pdaf -dim_ens 9 -filtertype 7 -
  local_range 5
```

  (this runs a filter (LESTKF) with localization and localization radius of 5 grid points)

Feedback, Questions, more code, …

Full PDAF package contains

- more tutorial code, more filters, and the fully implemented Lorenz-96 model and MITgcm model binding

Web site provides an extensive tutorial for self-study

For further questions

- Contact us at pdaf@awi.de
- Poster A.14, Friday 14:00–15:45 (L. Nerger)

pdaf@awi.de

Slides are available online:

http://pdaf.awi.de

PDAF Parallel Data Assimilation Framework