Advances in Time-Parallel Four Dimensional Data Assimilation in a Modular Software Framework

Brian Etherton, with Christopher W. Harrop, Lidia Trailovic, and Mark W. Govett

NOAA/ESRL/GSD

28 October 2016

The HPC group at NOAA/ESRL/GSD

- Strong track record in high performance computing
  - Massively Parallel Fine Grain (MPFG) Computing
  - Graphics Processing Units (GPUs)
  - Many Integrated Core (MIC)

- Wishes to advance the state of the art in data assimilation, in particular, via improved performance and design
  - NOAA/NCEP GSI has a core limit in the hundreds
  - 4D-Var approaches are time consuming
  - 4D-Ensemble approaches are memory & I/O intensive
  - Wish to use a ‘great’ solver with any model (atmos, ocean…)

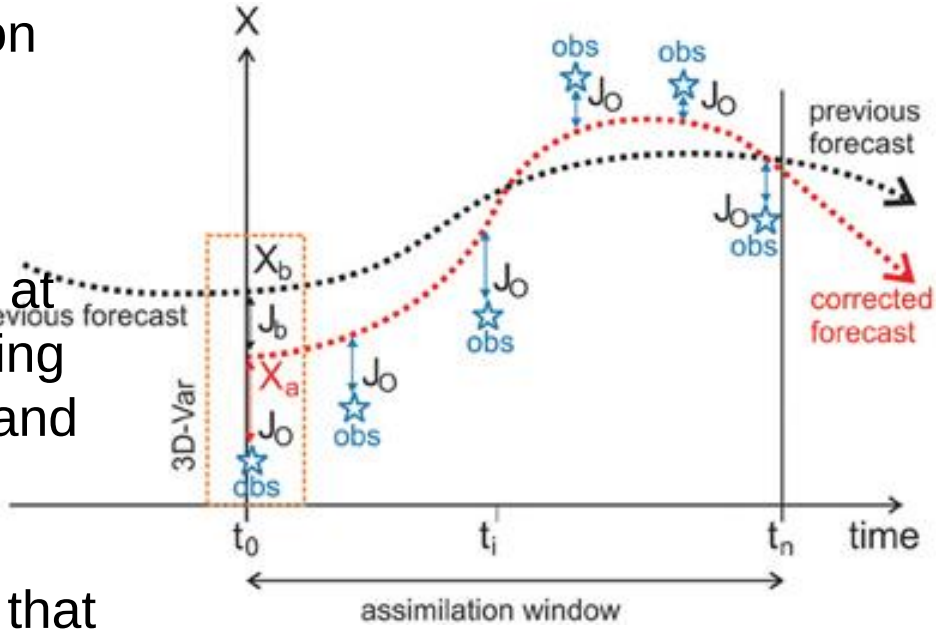
- First steps into data assimilation (started this year)

Time Parallel 4DDA - Motivation

- Four Dimensional Data Assimilation (4D DA) requires having information on the state of the system at many different times

- In some approaches, information at different times is achieved by running a model forward (Tangent-Linear) and backward (Adjoint) in time

- Optimal results with a TL and AD that mimic the true model



Lahoz and Schneider
Front. Environ. Sci., 28 May 2014
http://dx.doi.org/10.3389/fenvs.2014.00016

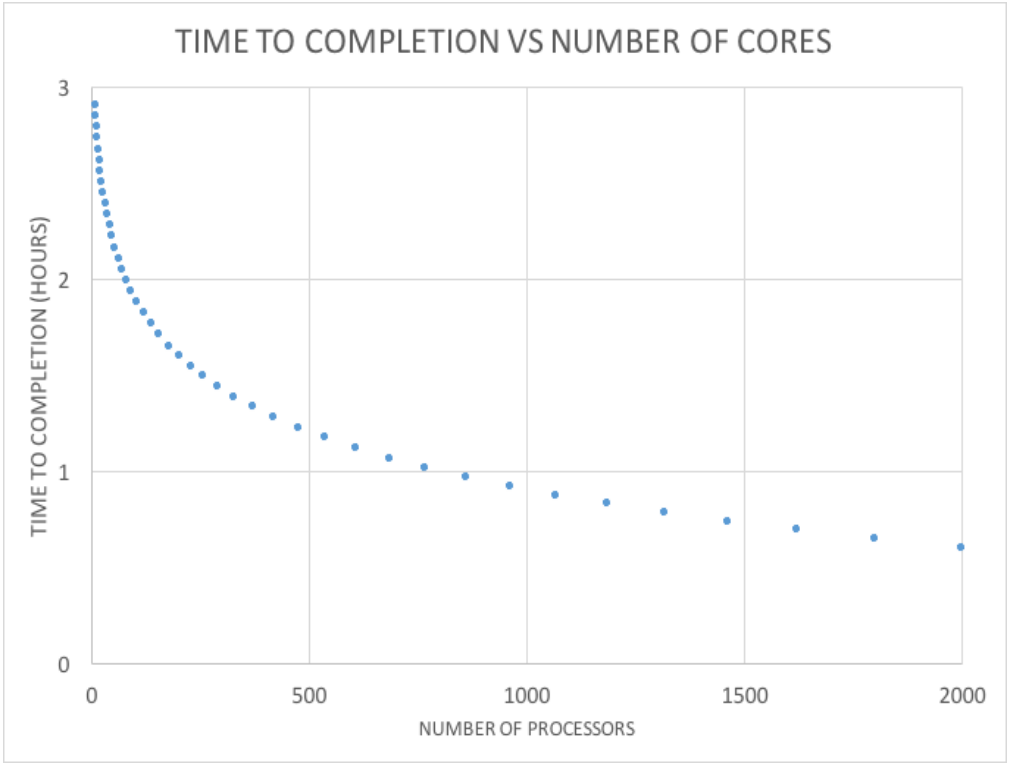
Time Parallel 4DDA - Motivation

- The time spent running the TL and AD is, roughly:

```
  LENGTH OF ASSIM WINDOW
* 2 (TL & AD)
* 1.5 (TL TAKES LONGER)
* 1.5 (AD TAKES LONGER)
* NUMBER OF ITERATIONS
```

- For a 12 hour window, 40 iterations, this value is 54*40=2160 hours, or 90 days

- This is, perhaps, 6x longer than the forecast itself - this must be improved



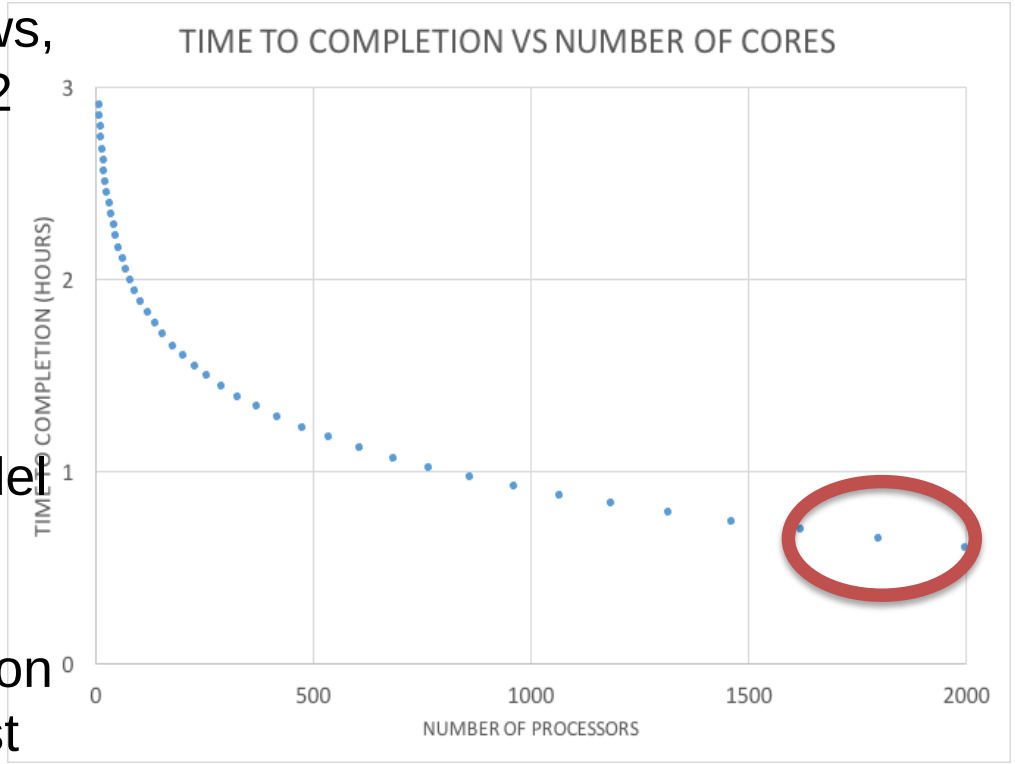
Time Parallel 4DDA - Motivation

- If the assimilation window could be broken into 48 ¼-hour windows, then run time *could* be closer to 2 model days rather than 90

- Would take ~27-minutes to compute for 1% real-time model

- Achieve scaling when your model is no longer scaling

- If scaling achieved, is the solution from this time-parallel version just as good?



4DVAR Data Assimilation

4DVAR traditionally involves taking one state (bucket), moving it all the way from the start to finish to start

Time parallel 4DVAR sends a number of states (buckets) from one time to the adjacent time

4DVAR Data Assimilation

We did not invent time-parallel 4DVAR – the ECMWF has done this sort of work (Saddle Point), as have others (e.g. Virginia Tech)

Our goal is not to develop a brand new DA system, but to explore promising existing approaches

Software Design - The Current State

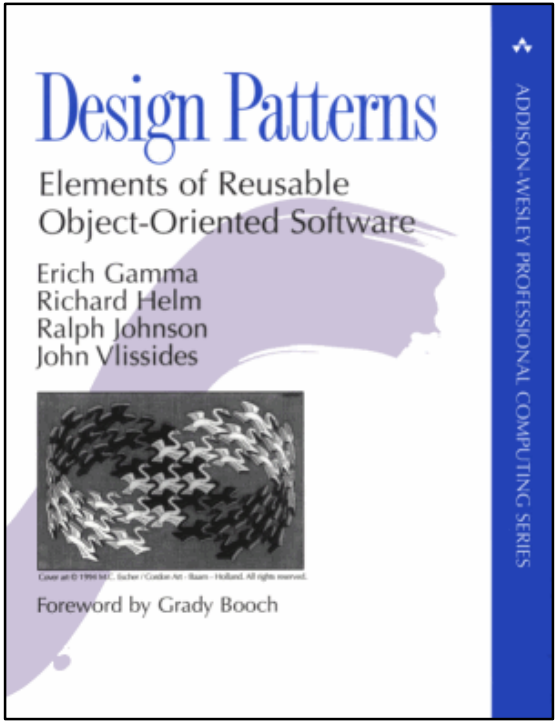
- Legacy Fortran 77/90 codes 100K+ lines
- Developers lack training & expertise in modern software engineering practices
- Poor software designs
  - Difficult to extend with new capabilities
  - Difficult to debug
  - Lack interoperability
  - Hinder R2O
  - Slow scientific progress

Addressing Design Deficiencies

- Maximize cohesion of software components
  - Components cannot be subdivided without exposing internal state.
- Minimize coupling of software components
  - Changes to A should not require changes to B
- Object-Oriented Design
  - Facilitates high cohesion and low coupling
  - Adopted by Software Industry 20+ years ago!

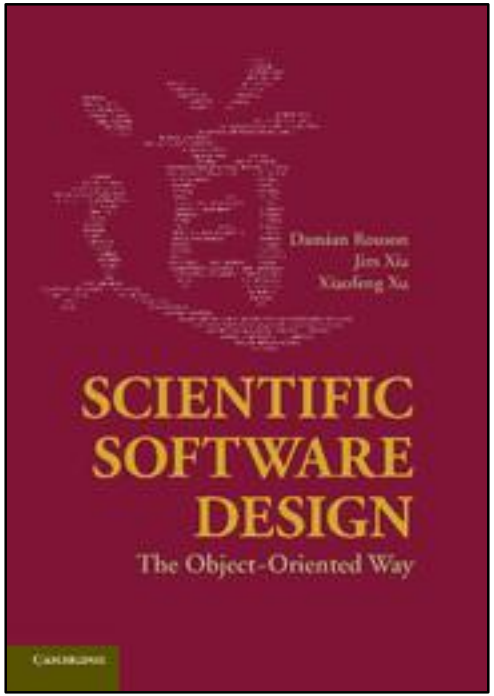
Addressing Design Deficiencies

- Design Patterns (1994)
- Arsenal of well-tested recipes for solving recurring design problems
- Arguably revolutionized object-oriented software development



Addressing Design Deficiencies

- Scientific Software Design (2011)
- Patterns for scientific applications (Fortran/C++)
- Design is as important as performance



Is Object Oriented Design The Solution?

- Why aren't we using modern software design principles?
  - Historical lack of language support (pre-Fortran 2003)
  - "Make it work, then make it fast, then forget it" culture
  - Design to accommodate future requirements not a priority?
  - Bad Performance?

- Will object-oriented design make it too slow?

- Performance is key to meet Operations deliver deadlines but…
  - Inflexible designs hurt R2O - A less tangible cost
  - Total cost of ownership includes cost of code maintenance

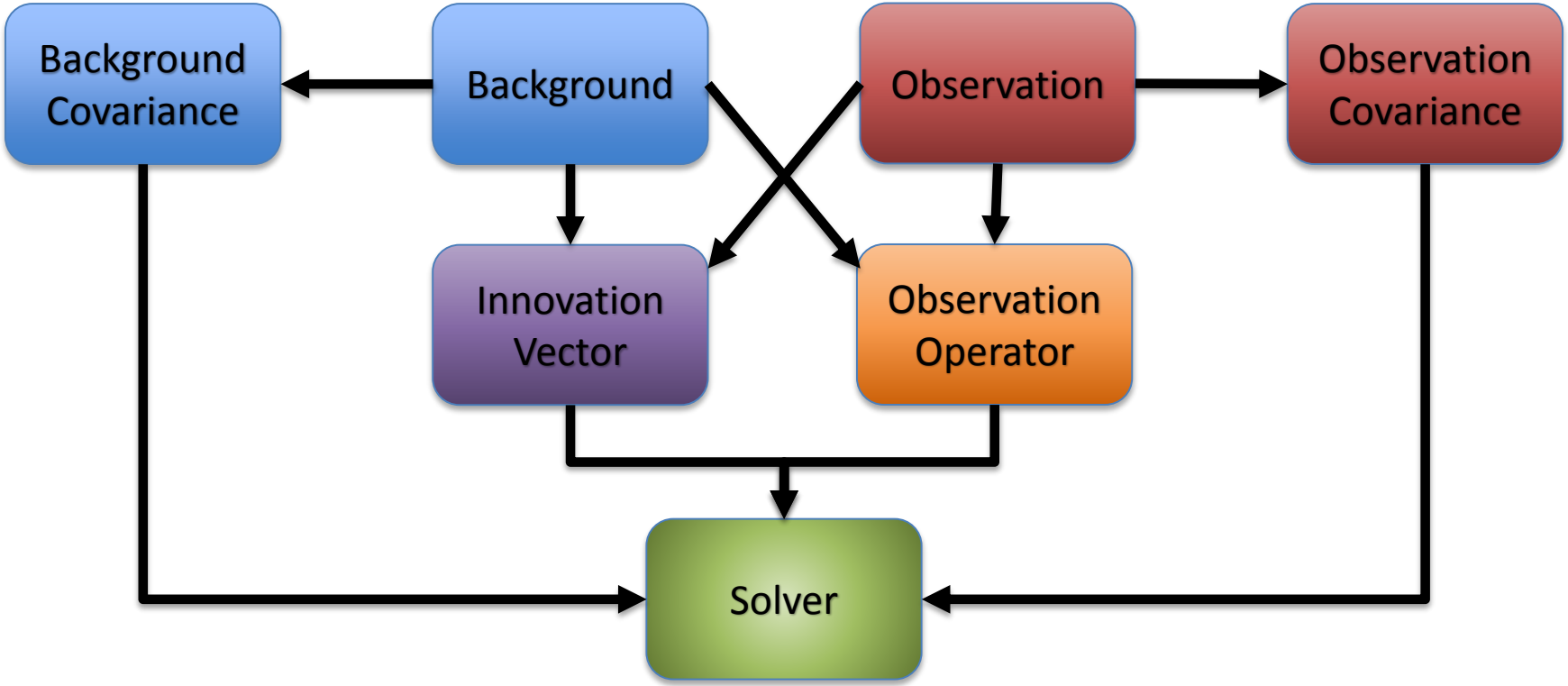
We Are Investigating

- What is the performance impact of object oriented design?
  - Let's measure it!
- Construct a data assimilation prototype
  - Design from scratch using "old school" methods
  - Provides baseline verification of correctness
  - Provides baseline performance measurements
- Construct a copy of the prototype
  - Progressively refactor and apply object oriented techniques
  - Measure correctness and performance against baseline

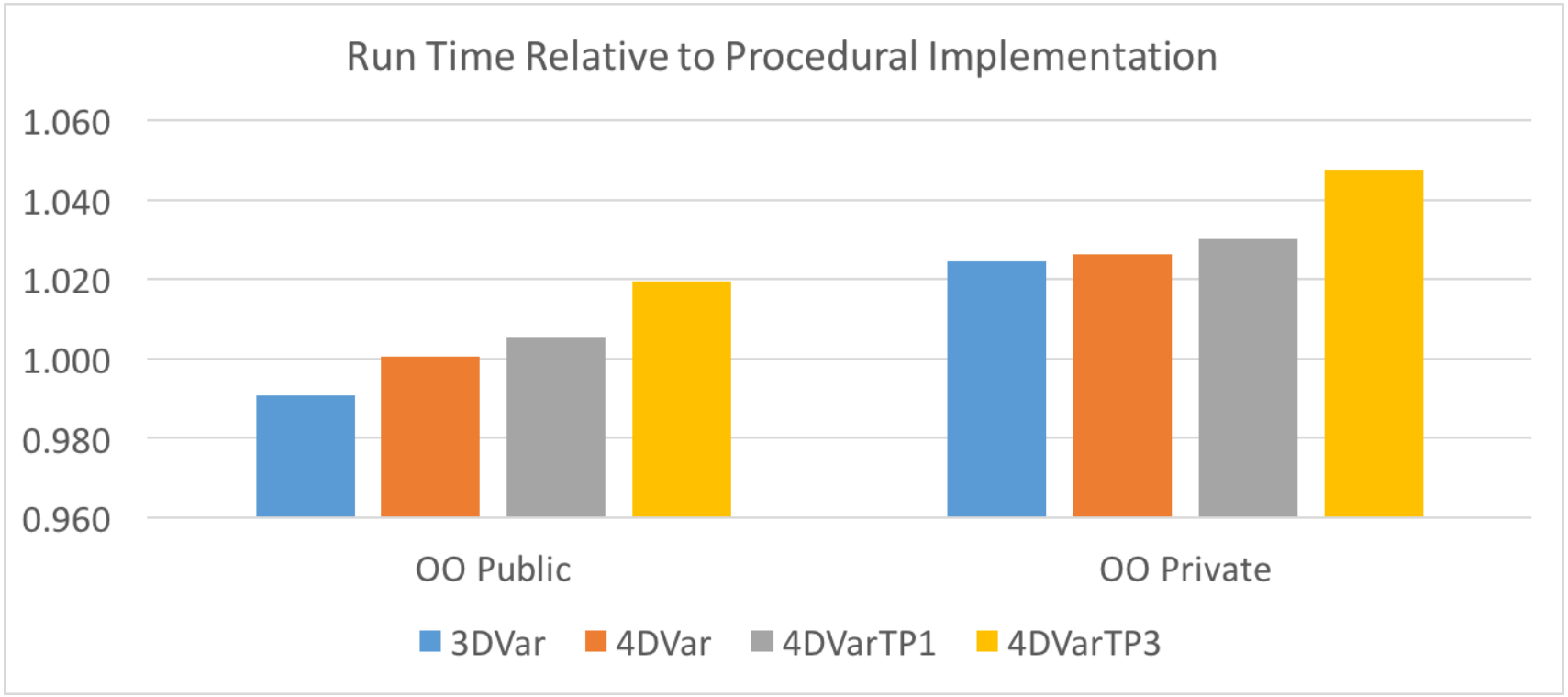
We Are Investigating

- Incremental approach
  - Facilitates identification of designs that cause performance penalties
  - Facilitates accumulation of patterns for techniques to avoid performance penalties
- Explore Fortran 2003/2008 features
  - Limitations of Fortran 2003/2008 object oriented features
  - Limitations of compiler support for 2003/2008 features
  - Interactions with OpenMP and OpenACC
- May consider other languages in the future

What We Have So Far

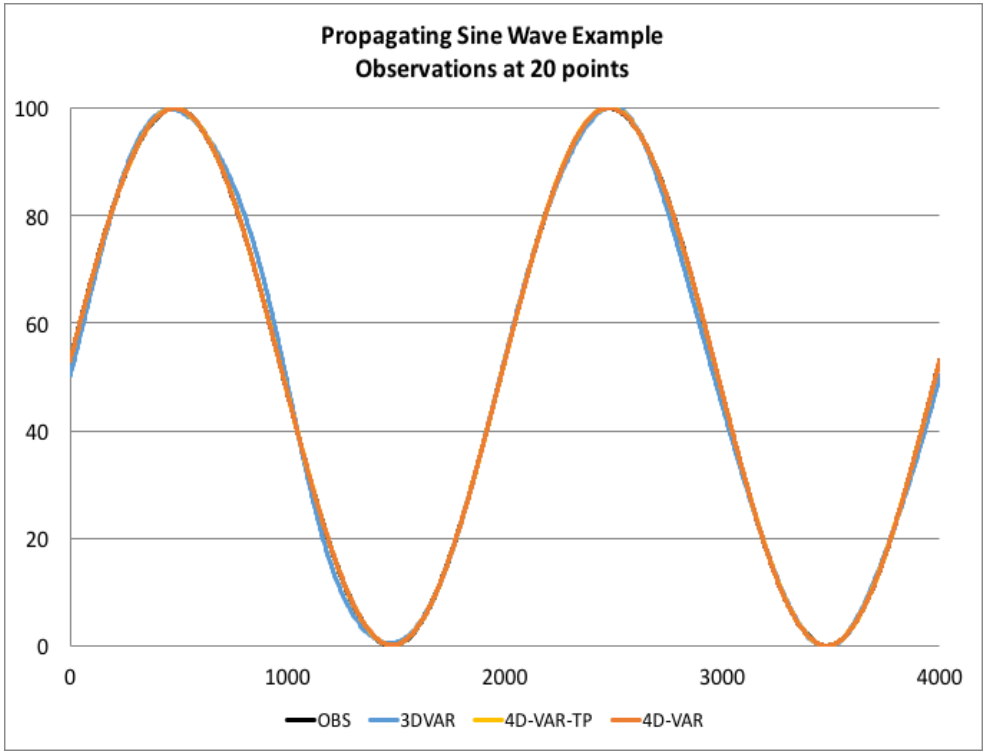


Performance of Object Oriented Implementation



Test 1: The eastward propagation of a 1D sine wave

- Tested on a single case
- Assimilation over a 3 “hour” assimilation window
- Time parallel – 3 ‘windows’, each its own OMP thread
- 4000 gridpoints, 20 observations
- TL and AD simple: derivative of sine is cosine

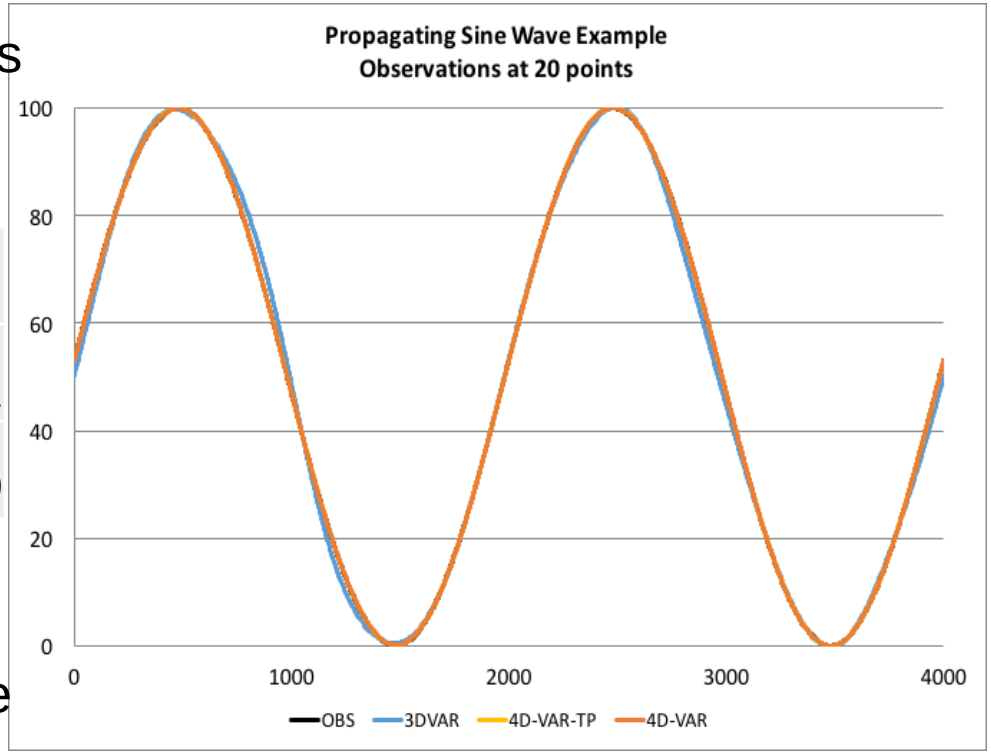


Test 1: The eastward propagation of a 1D sine wave

- Results show that all the methods produce an improved analysis of the sine wave

	3D-VAR	4D-VAR	4D-VAR-TP
BIAS	0.2805	-0.0326	-0.0302
RMSE	2.4913	0.0073	0.0049

- The question then is: does the time-parallel 4DVAR take less time to complete?

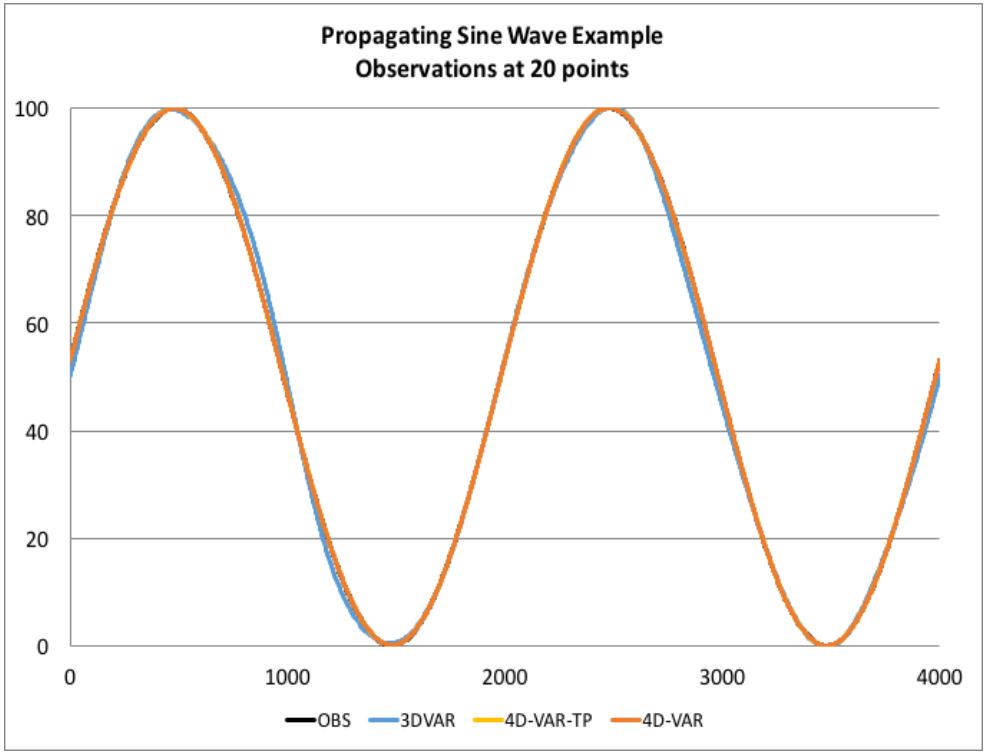


Test 1: The eastward propagation of a 1D sine wave

- Timing results from this case (in seconds):

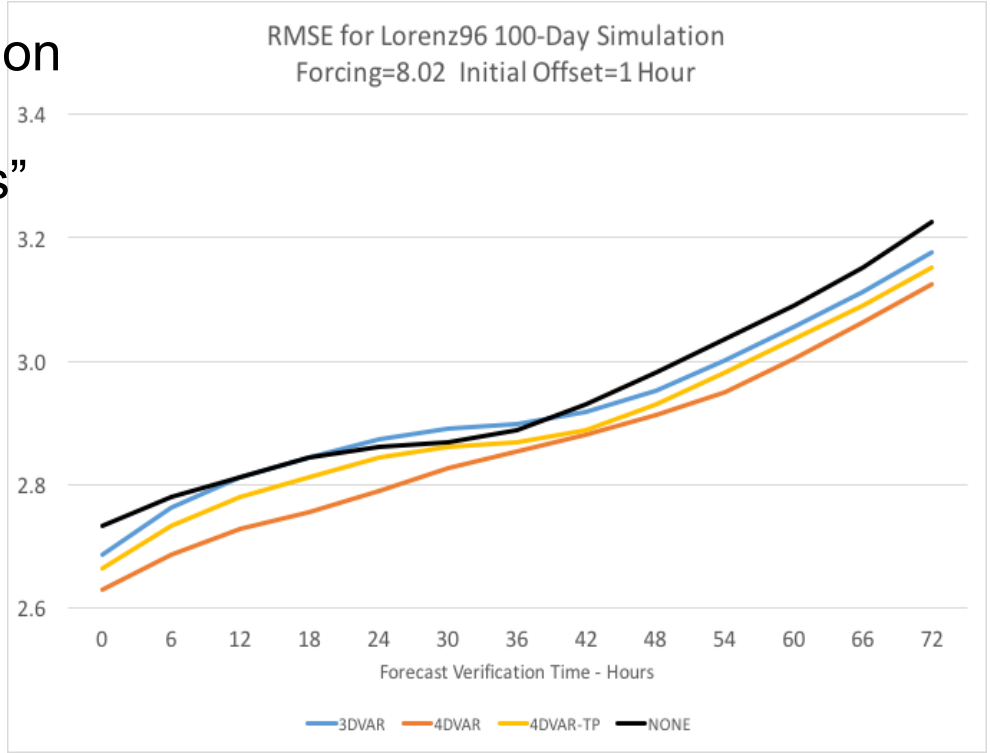
3DVAR	35.0
4DVAR	108.8
4DVAR-TP-1	108.6
4DVAR-TP-3	44.2

- Results show that using the 3 OMP-THREAD Time Parallel 4DVAR results in a substantial reduction in run-time length



Test 2: The Lorenz96 Model

- Tested over a 100 “day” simulation
- Data assimilation every 6 ”hours”
- Assimilation over a 3 “hour” assimilation window
- 40 gridpoints / 20 observations
  (every other gridpoint)
- F=8.02, Initial Offset=1 hour

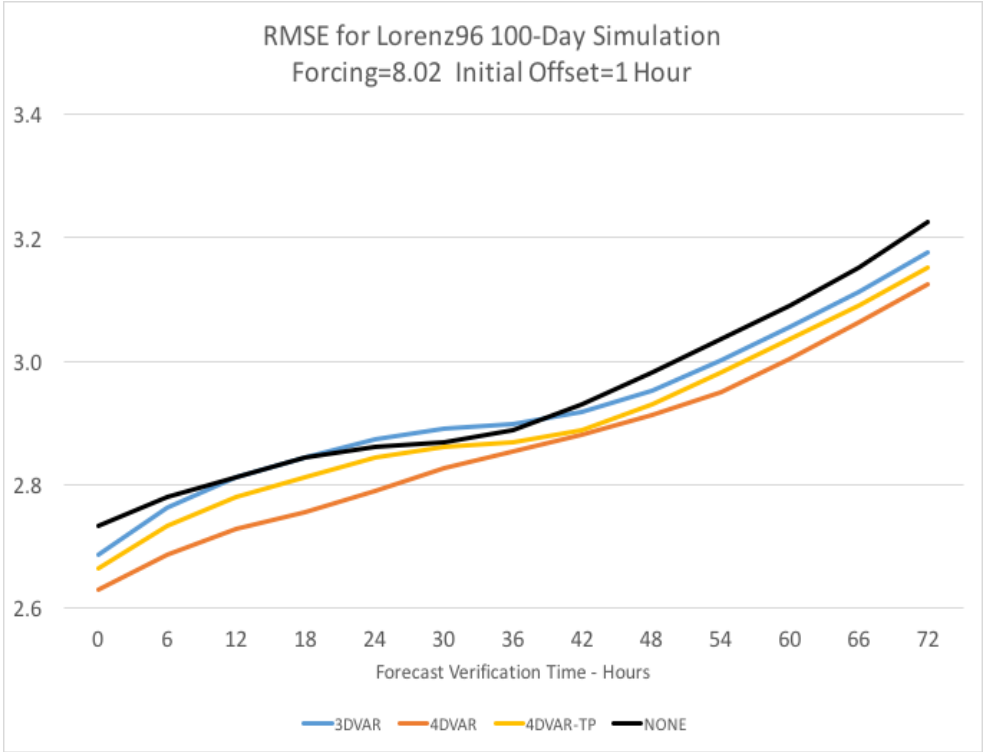


Test 2: The Lorenz96 Model

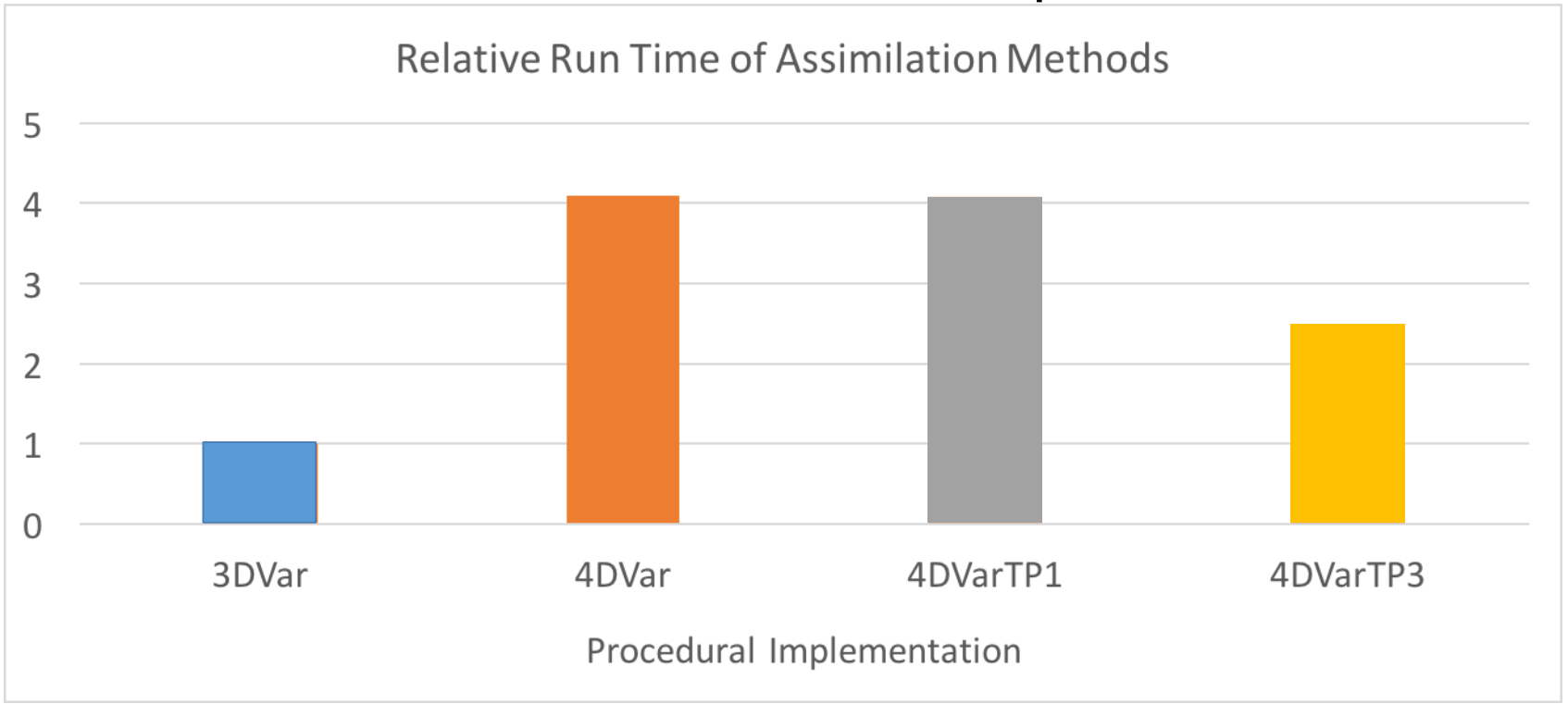
- The time parallel 4DVAR (yellow line) performed better than 3DVAR, but not quite as well as 4DVAR

- No great performance statistics here – the 40-point problem was not taxing

- Nonetheless – the Time Parallel 4DVAR results encourage us to continue on



Performance of Procedural Implementation
Lorenz Model with 4000 points



Conclusions

- Our simple experiments have shown
  - Time-Parallel 4D DA can result in reduced time-to-completion
  - Object oriented design resulted in minimal increases in time-to-completion
  - Accuracy of analyses and forecasts, with this method, are not unreasonable

- We expect that the benefits of Time-Parallel 4D DA will be greatest when the forecast model (and the Tangent-Linear and Adjoint thereof) have past the point of scaling well
  - Time-Parallel provides a new avenue to achieve scaling

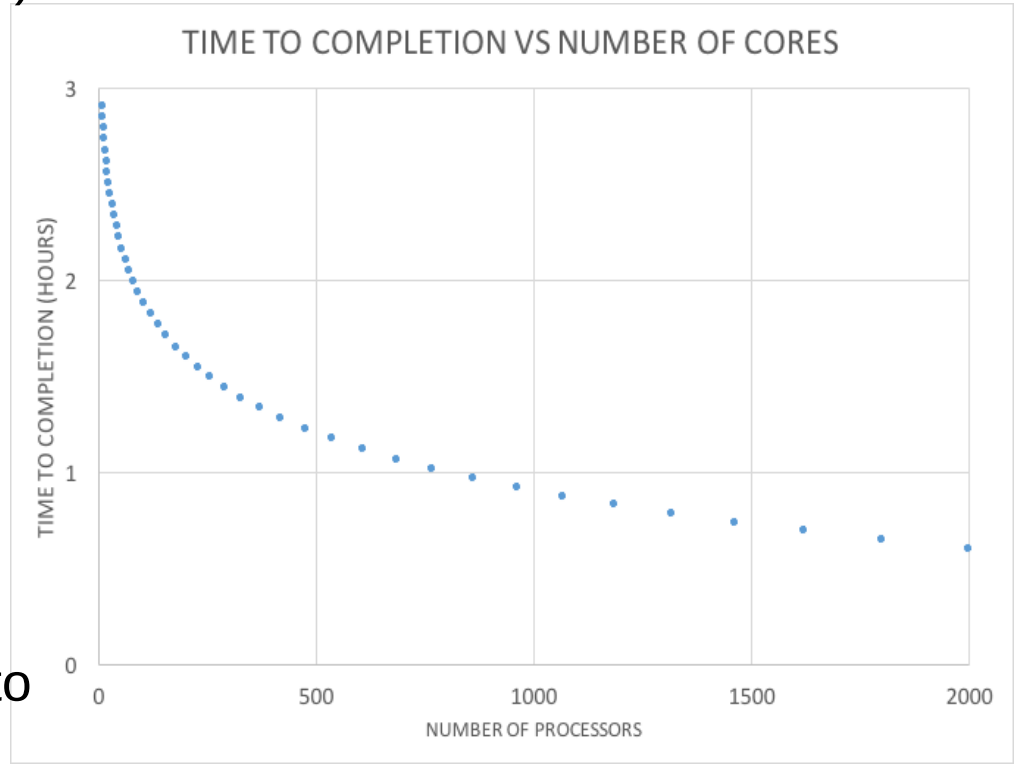
Time Parallel 4DDA - Motivation

- From Govett et al (BAMS paper)

G11 NIM (3.75KM resolution) using 64*20=1280 K80 GPUs runs in 1.6% of forecast time

- 12-hour forecast takes 12-minutes (could do only ONE iteration of 4DVAR)

- Time-parallel could do 40 iterations in ~30 minutes if the iterations could be subdivided into ~48 sections (60,000 K80s, 30,000 Pascals)



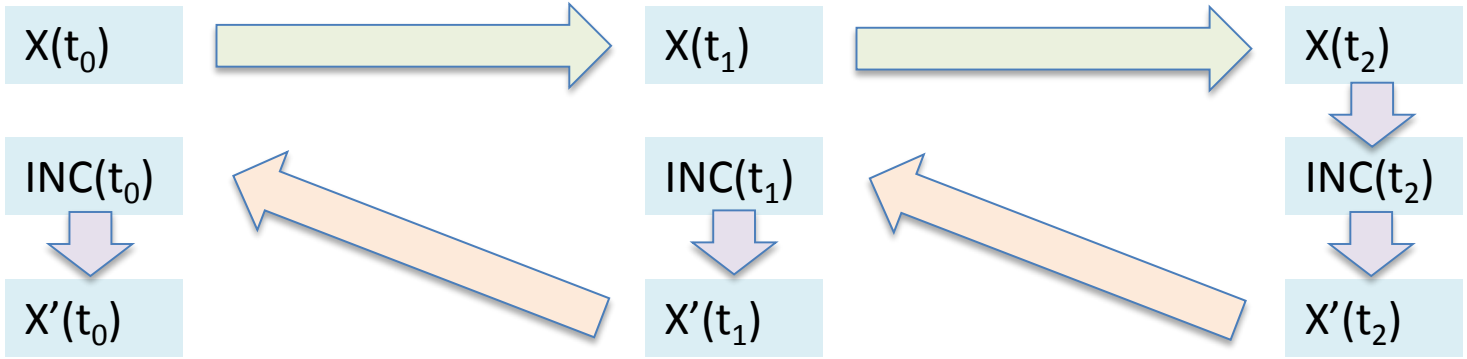
Thoughts for the future

- In the near term, we shall:
  - Make use of symmetry of matrices
  - Work to reduce amount of memory use by each thread
  - Test on an atmospheric model, likely QG T103L3
- In the longer term, we must:
  - Think of how this would work for a global model running at non-hydrostatic scales
  - Would like to keep all the states in memory, but that might be too much memory use. But if we write to disk, will that really slow things down?
  - Work to incorporate ensembles, making for a hybrid framework
    (48 windows, 51 ensemble members, shared approaches?)
    (BLAS/LAPACK – opportunities for better performance?)
  - Apply object oriented design patterns and measure impact

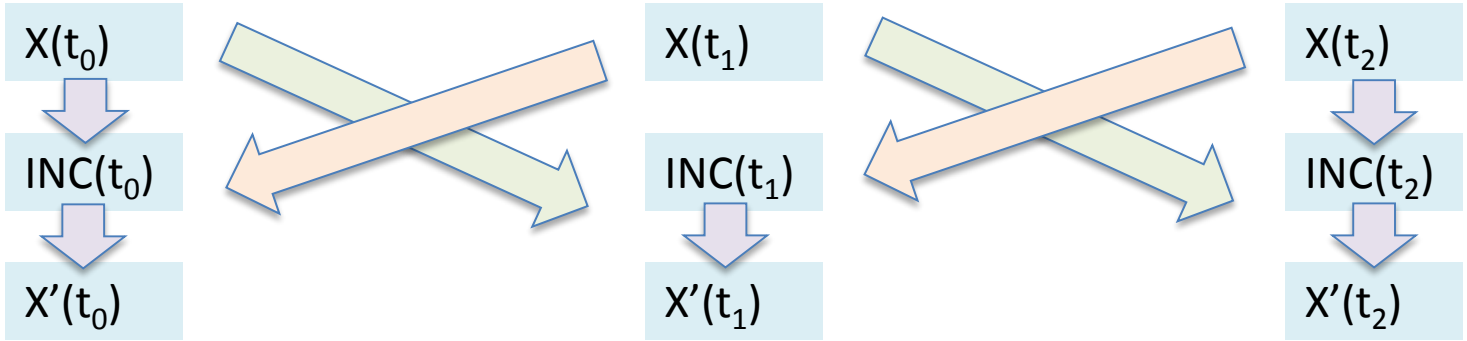
Slides that are backup

4D Data Assimilation – Time Parallel

4DVAR traditionally involves:



Time-Parallel Approach does:



4DVAR traditionally involves:

- Calculating a cost function

$$
\begin{aligned}
2J(\mathbf{x}) = {} & (\mathbf{x}-\mathbf{x}_b)^T\mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + (\mathbf{x}-\mathbf{x}_b)^T\mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}(\mathbf{x}-\mathbf{x}_b) \\
& - \left[\mathbf{y}_0 - H(\mathbf{x}_b)\right]^T\mathbf{R}^{-1}\mathbf{H}(\mathbf{x}-\mathbf{x}_b) \\
& - (\mathbf{x}-\mathbf{x}_b)^T\mathbf{H}^T\mathbf{R}^{-1}\left[\mathbf{y}_0 - H(\mathbf{x}_b)\right] \\
& + \left[\mathbf{y}_0 - H(\mathbf{x}_b)\right]^T\mathbf{R}^{-1}\left[\mathbf{y}_0 - H(\mathbf{x}_b)\right]
\end{aligned}
$$

- Calculating an increment to the present guess, ‘x

$$\nabla J(\mathbf{x}) = \mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}_b) + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}(\mathbf{x}-\mathbf{x}_b) - \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H}\left[\mathbf{y}_0 - H(\mathbf{x}_b)\right]$$

- The question then is: what is “x”?

$$\mathbf{x} = \frac{1}{2}\left[\,\mathrm{TL}(\mathbf{x}_{prior}) + \mathrm{AD}(\mathbf{x}_{prior})\,\right] \qquad t = 0,\ \ \mathbf{x} = \frac{1}{2}\left[\,\mathbf{x} + \mathrm{AD}(\mathbf{x})\,\right] \qquad t = T,\ \ \mathbf{x} = \frac{1}{2}\left[\,\mathbf{x} + \mathrm{TL}(\mathbf{x})\,\right]$$