Met Office

A Hybrid Differential-Ensemble Linear Forecast Model for 4D-Var

or *How to fix probably the biggest problem with 4D-Var*

Tim Payne

Friday 20th October 2023

Why do we bother with 4D-Var, and what is the problem with it?

Since 2004 the Met Office has used 4D-Var (latterly hybrid 4D-Var) for its operational data assimilation. We are currently in the process of re-thinking and re-writing all our NWP systems, including DA (see Andrew Lorenc's talk).

The main alternative to 4D-Var for data assimilation in global NWP is 4DEnVar (*Liu, Xiao and Wang, MWR 2008*) with no TL Model.

Lorenc & Jardak (2018) found, in trials of hybrid 4D-Var with $N_{ens} = 44$, that

(i) Pure 4D-Var performed ∼1.8% better than pure 4DEnVar

(ii) Hybrid 4D-Var (with optimal weight on B_{ens}) performed ∼2.2% better than hybrid 4DEnVar (with optimal weight on B_{ens})

No amount of R&D has changed this, and the Met Office has decided to stick with hybrid 4D-Var for next generation global DA.

In our experience of 'first generation' 4D-Var, a major drawback, in terms of performance and the development and maintenance cost, was the linear model, and specifically the physics in the linear model.

4D-Var Notation

4D-Var analysis minimises misfit to observations and background, ie seek $\mathbf{x}$ minimising $J(\mathbf{x}) = J_b(\mathbf{x}) + J_o(\mathbf{x})$ where

$$J_o(\mathbf{x}) \quad = \quad \frac{1}{2}\sum_{t=1}^{n}\left(\mathbf{y}_t - \mathcal{H}_t\mathcal{M}_0^t(\mathbf{x})\right)^T R_t^{-1}\left(\mathbf{y}_t - \mathcal{H}_t\mathcal{M}_0^t(\mathbf{x})\right)$$

Linearising about $\mathbf{x}_b$ with $\mathbf{x} - \mathbf{x}_b = \boldsymbol{\delta}$

$$\mathcal{H}_t\mathcal{M}_0^t(\mathbf{x}) \approx \mathcal{H}_t\mathcal{M}_0^t(\mathbf{x}_b) + \mathbf{H}_t\boxed{\mathbf{M}_0^t}\boldsymbol{\delta}$$

Incremental 4D-Var analysis is $\mathbf{x}_b + \boldsymbol{\delta}$ where $\boldsymbol{\delta}$ minimises

$$J(\boldsymbol{\delta}) \quad = \quad J_b(\boldsymbol{\delta}) + J_o(\boldsymbol{\delta}) \;[+ \text{ other terms} \cdots]$$

where $$J_o(\boldsymbol{\delta}) \quad = \quad \frac{1}{2}\sum_{t=1}^{n}\left(\mathbf{y}_t - \mathcal{H}_t\mathcal{M}_0^t(\mathbf{x}_b) - \mathbf{H}_t\mathbf{M}_0^t\boldsymbol{\delta}\right)^T R_t^{-1}\left(\mathbf{y}_t - \mathcal{H}_t\mathcal{M}_0^t(\mathbf{x}_b) - \mathbf{H}_t\mathbf{M}_0^t\boldsymbol{\delta}\right)$$

- $\mathbf{x}$: State vector (size $\sim 10^8$), $\mathbf{x}_b$: Background state, $\mathbf{y}_t$: Obs at time t
- $\mathcal{H}_t$: Observation operator at time t, $\mathcal{M}_0^t$: Forecast model from 0 to t
- $\mathbf{H}_t$: Linear obs op at t, $\mathbf{M}_0^t$: Linear model (incorrectly 'TLM') from 0 to t.

Considerations for forming the linear model

To form $\mathbf{M}_{t-1}^{t}$ an obvious place to begin is with differentiation of $\mathcal{M}_{t-1}^{t}$ wrt state $\mathbf{x}$.

$$\mathcal{M}_{t-1}^{t}(\mathbf{x}+\delta\mathbf{x}) \approx \mathcal{M}_{t-1}^{t}(\mathbf{x}) + {\mathcal{M}_{t-1}^{t}}'(\mathbf{x})\delta\mathbf{x} \qquad (\star)$$

This is fine for the dynamics, but not really possible (or at least not useful) for the physics - even where $\mathcal{M}_{t-1}^{t}$ is differentiable the domain of validity of the TL approximation $(\star)$ is tiny.

For the physics in particular usual practice is to regularise and/or simplify $\mathcal{M}_{t-1}^{t}$ and (if simplification not already linear) differentiate that.

All operational centres currently do some version of this strategy. Inter alia it is

- Very labour intensive
- Only moderately successful in reducing linearisation error
- May need to be updated whenever the physics in $\mathcal{M}$ changes

Possible alternatives

To overcome these problems the following have been proposed:

(a) Machine Learning/Neural Nets (off-line)
[eg Payne (ASL 2009), Hatfield et al (2021)]

(b) Localised Ensemble Tangent Linear Model (LETLM) (on-line)
[eg Frolov and Bishop 2016; Allen et al. 2017; Bishop et al. 2017]

(c) Hybrid TLM [Payne (MWR 2021)] (on-line)

Important fundamental differences in how these are used.

Of these only (c) is currently viable for operational scale NWP.

LETLM - **L**ocalised **E**nsemble **T**angent **L**inear **M**odel

This is a finite difference method of reconstructing the linear model. At time step 1 choose linear operator L_1 so that, for an ensemble of increments $\boldsymbol{\delta}^j, j = 1, 2, \cdots, n_{ens}$

$$L_1(\boldsymbol{\delta}^j) \text{ best fits } \mathcal{M}_0^1(\mathbf{x}_b + \boldsymbol{\delta}^j) - \mathcal{M}_0^1(\mathbf{x}_b),\ j = 1, 2, \cdots, n_{ens} \qquad (1)$$

Construct L_1 a row at a time, eg row i of

$$d\mathbf{x}_1 = L_1 d\mathbf{x}_0$$

is

$$d\mathbf{x}_1(i) = L_1(i, :)d\mathbf{x}_0$$

where $d\mathbf{x}_1(i)$ is just one variable at one gridpoint at time 1. A variable at one grid point after one short time step δt will depend only on variables at nearby gridpoints at time 0. Hence there are only a few non-zero elements of $L_1(i, :)$ and these can be estimated from (1) using a small ensemble n_{ens}.

LETLM continued

Similarly at time step t choose L_t so that

$$L_t \cdots L_1(\boldsymbol{\delta}^j) \text{ best fits } \mathcal{M}_0^t(\mathbf{x}_b + \boldsymbol{\delta}^j) - \mathcal{M}_0^t(\mathbf{x}_b),\ j = 1, 2, \cdots, n_{ens}$$

Computationally, for **every** variable $(u, v, w, p, \rho, \theta, q_v, \cdots)$ at **every** grid point at **every** time step, we need to compute the pseudo-inverse of $n_{ens} \times s$ matrix where s is number of variables × number of grid points in influence region.

- We have run experiments comparing linearisation error using the LETLM with the Met Office's existing TLM, which contains some physics (simplified schemes for bounday layer drag/diffusion, cloud with latent heat release, convection, but no radiation, gravity wave drag,...)
- We found for realistic resolutions/time steps we needed ensembles in the high hundreds/low thousands for the LETLM to be competitive, and for pressure not even that was enough.

Discussion of LETLM

- One (possibly the main) difficulty with the LETLM approach is the large number of grid points in the influence domain needed to model the TLM. This implies a
  - Large ensemble required to sample the large influence region
  - Large computational burden to obtain pseudo-inverse of $n_{ens} \times s$ matrix every LETLM time step
- The tangent-linear of the dynamics part of the UM is more straightforward to code than the physics, in particular unlike the physics does not need regularising
- The physical parametrisations, which are the 'hard' part of the model to linearise, are largely on columns.

Hybrid TLM

This all suggests a better way forward is to

- Use traditional coded tangent-linear for the dynamics
- Use an LETLM on columns to adjust the linear model after each dynamics time step
- The adjustment process requires running an ensemble of TL Models as well as the ensemble of full models

This is the basis of the hybrid TLM.

We will use M_{t-1}^{t-} to denote the evolution from time step $t-1$ to t of an 'incomplete' linear model, probably dynamics only but might contain minimal physics.

We call this the 'simplified linear model' (SLM). The TL state resulting from this incomplete linear model, prior to the LETLM adjustment, will be denoted $d\boldsymbol{\phi}^{t-}$.

Hybrid TLM Method

Stage 1: Ensemble of forecast model differences

This part is common to many ensemble methods. Create an ensemble of perturbed analyses, and run the forecast model from these to form an ensemble of forecasts at times through the window. By subtracting off the corresponding background forecast (ie, the forecast from the corresponding member of the analysis ensemble in the previous cycle) we obtain an ensemble of forecast differences

$$\delta \mathbf{x}_m^t, \; m = 1, \cdots, N_{ens}, t = 0, \cdots, T$$

Hybrid TLM Method - Stage 2: Coefficient calculation

Written out in detail the coefficient calculation is quite involved, but in essence, and omitting some important details:

The ‘coefficients’ are parameters in operators $N_1, N_2, \cdots, N_{timesteps}$. At time step 1 choose N_1 so that, for an ensemble of increments $\boldsymbol{\delta}^j, j = 1, 2, \cdots, n_{ens}$

$$(I + N_1)M_0^{1-}(\boldsymbol{\delta}^j) \text{ best fits } \mathcal{M}_0^1(\mathbf{x}_b + \boldsymbol{\delta}^j) - \mathcal{M}_0^1(\mathbf{x}_b),\ j = 1, 2, \cdots, n_{ens}$$

and in general at time step i choose N_i so that

$$(I + N_i)M_{i-1}^{i-} \cdots (I + N_2)M_1^{2-}(I + N_1)M_0^{1-}(\boldsymbol{\delta}^j)$$
$$\text{best fits } \mathcal{M}_0^i(\mathbf{x}_b + \boldsymbol{\delta}^j) - \mathcal{M}_0^i(\mathbf{x}_b),\ j = 1, 2, \cdots, n_{ens}$$

Where $M_0^{1-}, M_1^{2-} \cdots, M_{i-1}^{i-}$ is the simplified linear model.
In this case $\mathbf{x}_b$ and the $\boldsymbol{\delta}^j$ are valid at $t = 0$.
Note that

$$\{(I + N_i)M_{i-1}^{i-} \cdots (I + N_2)M_1^{2-}(I + N_1)M_0^{1-}(\boldsymbol{\delta}^j),\ j = 1, 2, \cdots, n_{ens},$$
$$i = 1, \cdots, N_{timesteps}\} \text{ is the TLM ensemble}$$

LETLM v hybrid TLM - key differences

For hybrid TLM

1. We need a TL for dynamical core
2. The ‘LETLM’ part is only on columns, ie is now 1D instead of 3D, with a correspondingly dramatically smaller ensemble required
3. Instead of one very large ensemble of full model runs, we now need
   - a small ensemble of full model runs, and
   - a small ensemble of simplified linear models (SLMs).

   Every analysis cycle we run the full model ensemble once through the time window, and now also the SLM ensemble once through the time window, at every SLM time step computing adjustment coefficients and adjusting every SLM ensemble member by these coefficients.

NB. The simplified linear model might be a pure dynamical TLM, or could contain a minimal amount of physics.

Hybrid TLM practicalities

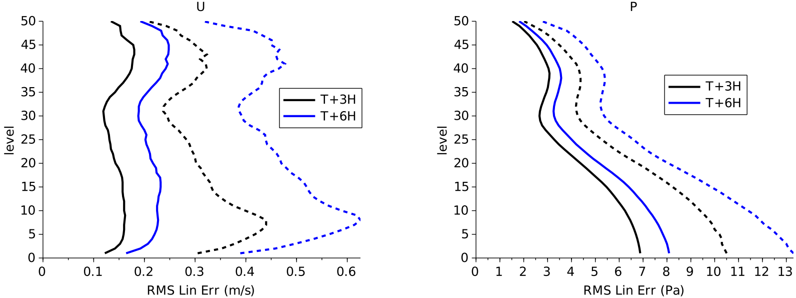
	Coded TLM	LETLM	H-TLM
Coded TL dynamics	Yes	No	Yes
Coded TL physics	Yes	No	No
Full Model ensemble	None	V large	Small
TL Model ensemble	None	None	Small
Cost of coeff comp'n	None	V high	Moderate
CPU cost at run time	High	Low	Intermediate
I/O cost at run time	Low	High	Intermediate

Impact of the hybrid TLM - Linearisation Error

The accuracy of a linear model $\mathbf{M}$, the so-called ‘linearisation error’, is

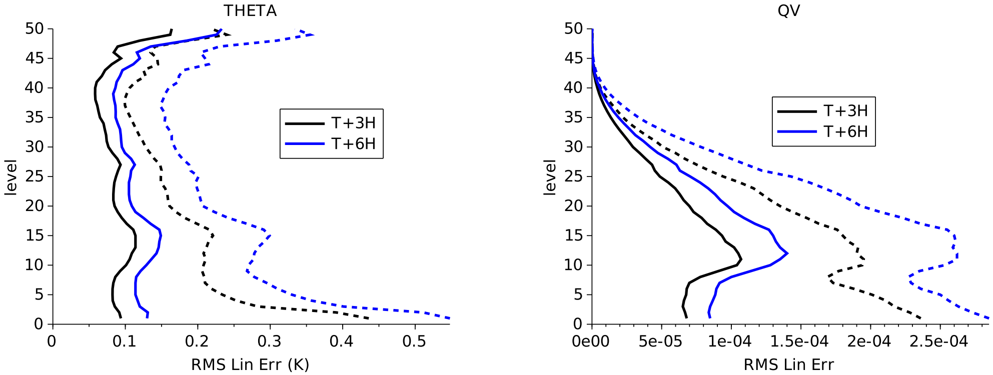
$$\mathcal{M}_0^t(\mathbf{x}_b+\boldsymbol{\delta}) - \mathcal{M}_0^t(\mathbf{x}_b) - \mathbf{M}_0^t\boldsymbol{\delta}$$

Compared with the Met Office’s current TLM, the hybrid TLM greatly reduces linearisation error:



Comparison of linearisation error using current TLM including all available coded physics (dashed) and hybrid TLM with $N_{ens} = 50$ (solid), for t=3,6 hours (black, blue).

Linearisation Error using hybrid TLM: θ' and q_v'



Dashed lines: linearisation error in current TLM (including all available physics)
Solid lines: linearisation error using hybrid TLM with influence region 5 points in vertical column centred on target point (so 40 predictors in total), and $N_{ens} = 50$.
Here TL/full model on $432 \times 325 \times 70$ and $640 \times 480 \times 70$ grids.

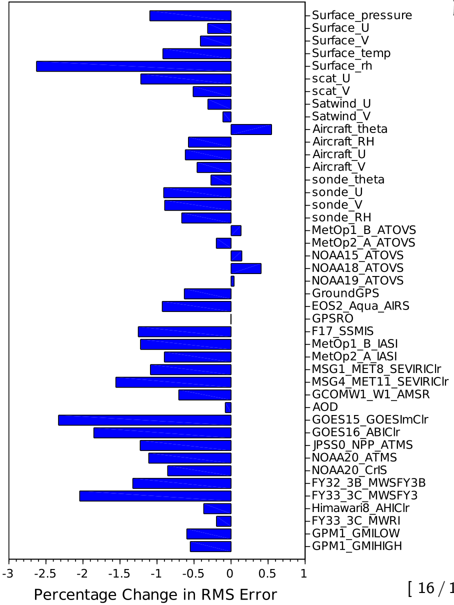
Impact of hybrid TLM on cycled trials: diagnostics at T+6 hours

The hybrid TLM and its adjoint have been inserted into 4D-Var.

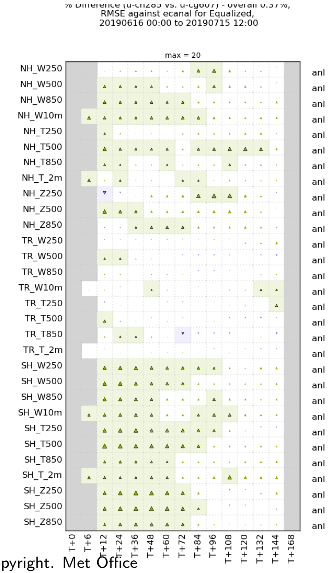
Trials of ∼7 weeks have been run, in which control uses standard Met Office linear model with all available physics, test uses hybrid TLM.

Figure on right shows percentage change in RMS Error (forecast - observations) at T+6 averaged over the first 124 analyses, ie

$$\frac{100[\text{RMSE(TEST) - RMSE(CNTL)}]}{\text{RMSE(CNTL)}}$$



Impact of hybrid TLM on cycled trials: longer forecasts



Impact of H-TLM on RMS fit of forecasts of length 1-6 days to independent ECMWF analyses, compared with standard operational system. Green up triangle signifies improvement, blue down triangle degradation.

Summary of hybrid TLM

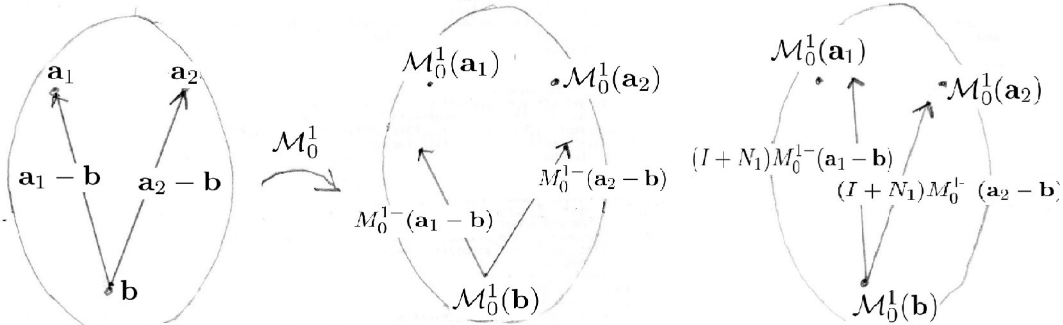
- Replace coded TL (of dynamics + physics) with coded TL for dynamics only, and on-the-fly ensemble method to adjust each time step for the physics
- Compared with current TLM:
  - Dramatic reductions in linearisation error
  - Large reductions in RMS T+6 forecast - obs
  - Improvements across the board in longer forecasts, and perhaps even more benefit with more work
- Much easier to develop and maintain than coded TLM
- The Met Office is currently engaged in re-writing all our NWP systems.
  The largest scientific change we will be making to 4D-Var is to introduce the hybrid TLM described in this talk.

Payne, Monthly Weather Review, Vol 149, pages 3-19, 2021
'A Hybrid Differential-Ensemble Linear Forecast Model for 4D-Var'

Additional Slides

Time step 1

Try to illustrate idea with ensemble of size two...

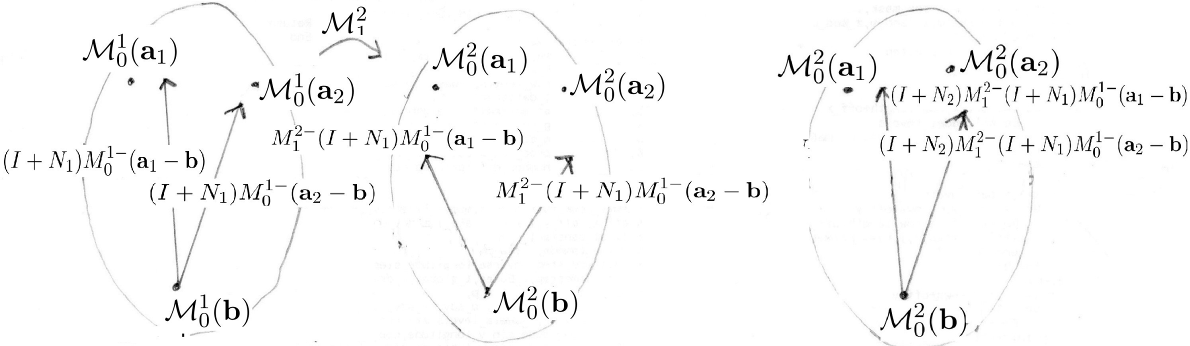


$\mathbf{a}_1$
$\mathbf{a}_2$
$\mathbf{a}_1 - \mathbf{b}$
$\mathbf{a}_2 - \mathbf{b}$
$\mathbf{b}$
$\mathcal{M}_0^1$
$\mathcal{M}_0^1(\mathbf{a}_1)$
$\mathcal{M}_0^1(\mathbf{a}_2)$
$M_0^{1-}(\mathbf{a}_1 - \mathbf{b})$
$M_0^{1-}(\mathbf{a}_2 - \mathbf{b})$
$\mathcal{M}_0^1(\mathbf{b})$
$\mathcal{M}_0^1(\mathbf{a}_1)$
$\mathcal{M}_0^1(\mathbf{a}_2)$
$(I + N_1)M_0^{1-}(\mathbf{a}_1 - \mathbf{b})$
$(I + N_1)M_0^{1-}(\mathbf{a}_2 - \mathbf{b})$
$\mathcal{M}_0^1(\mathbf{b})$

Time step 1 before adjustment

After adjustment

Time step 2



$\mathcal{M}_1^2$

$\mathcal{M}_0^1(\mathbf{a}_1)$

$\mathcal{M}_0^1(\mathbf{a}_2)$

$(I+N_1)M_0^{1-}(\mathbf{a}_1-\mathbf{b})$

$(I+N_1)M_0^{1-}(\mathbf{a}_2-\mathbf{b})$

$\mathcal{M}_0^1(\mathbf{b})$

$\mathcal{M}_0^2(\mathbf{a}_1)$

$\mathcal{M}_0^2(\mathbf{a}_2)$

$M_1^{2-}(I+N_1)M_0^{1-}(\mathbf{a}_1-\mathbf{b})$

$M_1^{2-}(I+N_1)M_0^{1-}(\mathbf{a}_2-\mathbf{b})$

$\mathcal{M}_0^2(\mathbf{b})$

$\mathcal{M}_0^2(\mathbf{a}_1)$

$\mathcal{M}_0^2(\mathbf{a}_2)$

$(I+N_2)M_1^{2-}(I+N_1)M_0^{1-}(\mathbf{a}_1-\mathbf{b})$

$(I+N_2)M_1^{2-}(I+N_1)M_0^{1-}(\mathbf{a}_2-\mathbf{b})$

$\mathcal{M}_0^2(\mathbf{b})$

Time step 2 before adjustment

After adjustment

Train H-TLM adjustment $(I+N_2)$ by choosing $(I+N_2)M_1^{2-}(I+N_1)M_0^{1-}(a_i-b)$ to best fit $\mathcal{M}_0^2(a_i)-\mathcal{M}_0^2(b), i=1,2,\cdots$.