Towards operational implementation of the Object Oriented Prediction System at ECMWF

Marcin Chrust, Mats Hamrud, Olivier Marsden, Deborah Salmond, Sebastien Massart, Patrick Laloyaux, Peter Lean, Lars Isaksen, Elias Holm, Alan Geer, Stephen English

Aveiro, 2018

Motivation

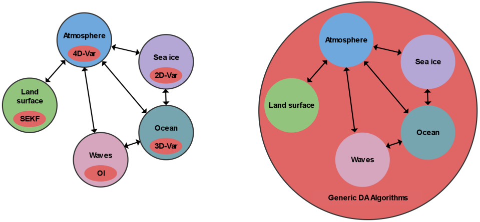


Figure: Common Framework for Coupled DA. Courtesy Y. Trémolet.

Figure: Foster and scale collaborations.

Facilitate research:

- Saddle point formulation
- Preconditioning
- EnVar
- ..

Object Oriented Prediction System

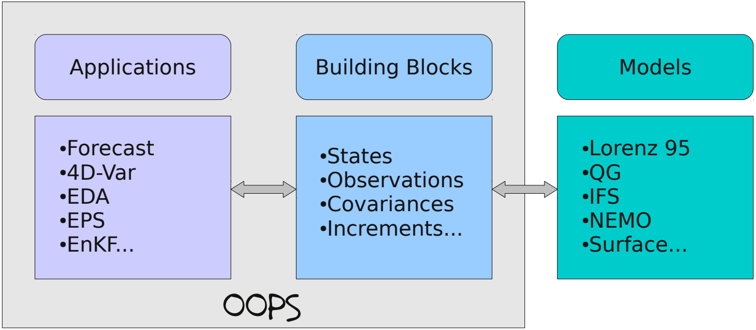
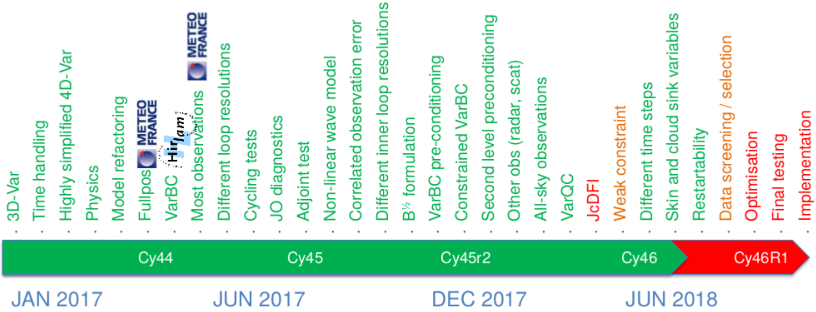


Figure: Object Oriented Prediction System. Courtesy Y. Trémolet.

- The high levels Applications use abstract building blocks;
- The Models implement the building blocks;
- OOPS is independent of the Model being driven.

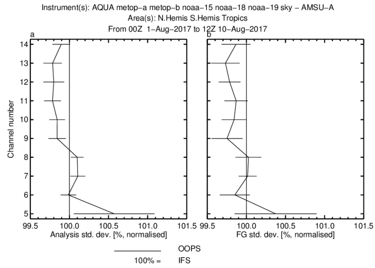
OOPS-IFS status

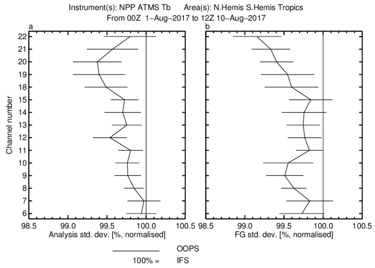


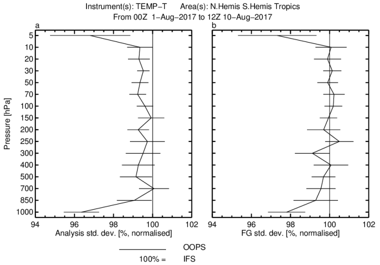
Path to operations:

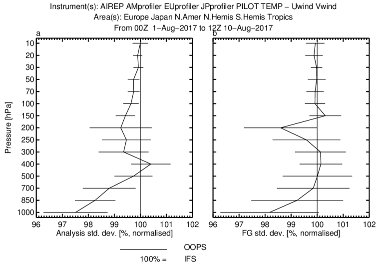
- Ring fenced OOPS Cy46R2 later this year - default DA system in RD onwards;
- Operational implementation - 2020-21 - due to Bologna Data Centre project.

OOPS-IFS validation: Tco399-T95-T159

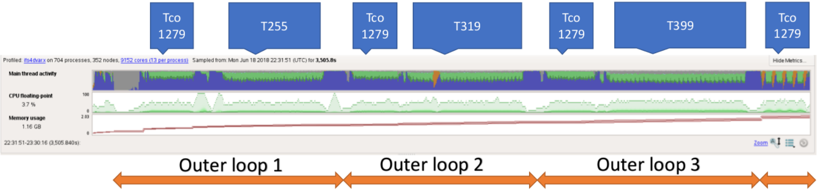








OOPS-IFS runtime: Tco1279-T255-T319-T399



Name of the executable : /fws4/lb/work/rd/das/gz91/bin/ifs4dvar.x
Number of MPI-tasks : 704
Number of OpenMP-threads : 18
Wall-times over all MPI-tasks (secs) : Min=3499.250, Max=3679.340, Avg=3631.339, StDev=20.393
Routines whose total time (i.e. sum) > 0.000 secs will be included in the listing

Avg-%	Avg.time	Min.time	Max.time	Incl.min	Incl.max	St.dev	Imbal-%	# of calls	: Name of the routine
11.12%	403.624	393.617	459.839	393.619	459.844	17.559	14.40%	4628096	: GATH_GRID
7.90%	287.026	201.524	501.521	201.657	501.649	38.060	59.82%	43201664	: TRGTOL
7.83%	284.423	186.775	354.594	186.816	354.634	25.673	47.33%	16130752	: TRLTOM
4.30%	156.270	132.555	182.268	132.589	182.303	9.581	27.27%	16349696	: TRMTOL
4.83%	175.522	127.428	205.792	127.500	205.865	15.534	38.08%	16349696	: TRLTOG
0.00%	0.109	76.804	76.804	76.805	76.805	2.895	0.00%	1911	: GRIB_API:IGRIB_WRITE_BYTES_INT
2.55%	92.463	71.524	114.983	71.528	114.990	8.005	37.80%	10115776	: SLCOMM:SLCOMM_INT
0.96%	34.835	33.826	35.011	33.826	35.011	0.060	3.38%	6336	: VARBC_SETUP:LOAD_TABLE
2.39%	86.751	31.997	96.840	31.999	96.842	19.339	66.96%	3271488	: DIST_GRID
0.96%	34.918	30.409	46.371	30.648	46.622	2.150	34.42%	3244032	: BRPTOB
0.92%	33.470	28.520	42.293	28.523	42.296	2.017	32.56%	46573696	: LWVDRAD
0.88%	32.055	27.139	37.250	27.142	37.255	1.651	27.14%	46573696	: LWVDR
0.86%	31.182	25.736	41.865	25.739	41.870	3.288	38.53%	10110144	: SLCOMM2A:SLCOMM2A_INT
0.73%	26.340	23.764	27.213	23.775	27.224	0.481	12.67%	4124736	: JB_CONTROL_VECTORS_PARA_MOD:DOT_PRODUCT_JBCV_JBCV_WEIGHT
0.74%	26.750	22.890	31.104	22.895	31.108	1.723	26.41%	213265536	: CLOUDSC
0.87%	31.430	22.469	43.908	22.702	44.349	4.270	48.83%	37599577664	: CUADJTQSAD
1.84%	66.991	22.350	82.269	22.532	82.445	7.262	72.83%	4625280	: ORDER_INDEPENDENT_SUMMATION_MOD:ORDER_INDEP_GLOBAL_SUM
0.66%	23.923	20.900	28.185	35.249	47.598	1.620	25.85%	58392576	: SWNIAD
0.72%	26.296	20.603	40.768	23.258	43.257	2.971	49.46%	13592832	: TRMTOS
0.00%	0.029	20.064	20.064	20.065	20.065	0.756	0.00%	7816	: GRIB_API:IGRIB_READ_FROM_FILE
0.53%	19.124	18.193	22.242	299.274	300.916	1.267	18.20%	223168	: FIELD_CONTAINER_OPER_MOD:FIELD_CONTAINER_INTERP
0.93%	33.745	17.589	43.732	17.591	43.734	4.555	59.78%	46573696	: LWCAD

Notes:
- Runtime ~**1.1x** IFS reference;
- This was collected with adjoint tests;
- Debugging output was on;
- GATH_GRID can be optimized away almost completely;
- Restart mechanism was deactivated;
- Communication bound;

Lessons learned: we can't abandon the square root formulation

- Right B-preconditioned formulation

  change of variables: $dx = \mathbf{B}dx'$

  $$(\mathbf{I} + \mathbf{G}^\mathbf{T}\mathbf{R}^{-1}\mathbf{G}\mathbf{B})dx' = -\sum dx'_i + \mathbf{G}^\mathbf{T}\mathbf{R}^{-1}\mathbf{d}$$

  We solve the above system using a symmetric solver with a modified inner product $dx'^T\mathbf{B}dx'$

- Square root $B^{1/2}$ formulation

  change of variables: $dx = \mathbf{B}^{1/2}v = \mathbf{U}v$

  $$(\mathbf{I} + \mathbf{U}^\mathbf{T}\mathbf{G}^\mathbf{T}\mathbf{R}^{-1}\mathbf{G}\mathbf{U})v = -\sum v_i + \mathbf{U}^\mathbf{T}\mathbf{G}^\mathbf{T}\mathbf{R}^{-1}\mathbf{d}$$

  We solve the above system using a symmetric solver with canonical inner product

Why do we need the $B^{1/2}$ formulation in OOPS

Multi-resolution test case: T255/T95/T159

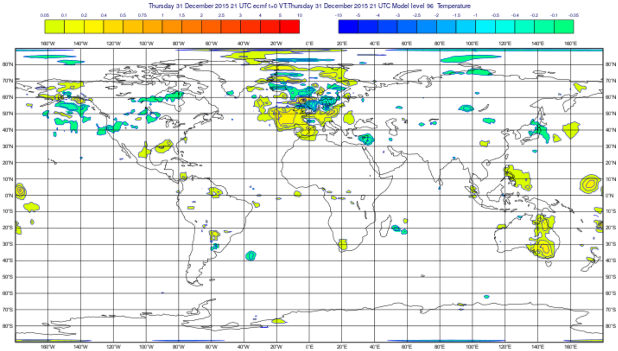


Figure: OOPS; T increments; 500mb; $dx_{95}^{159} - dx_{159}^{*}/2.5$, where $dx_{159}^{*}/2.5 = \mathbf{B}^{159} dx_{95}^{\prime 159}/2.5$.

Why do we need the $B^{1/2}$ formulation in OOPS

Multi-resolution test case: T255/T95/T159

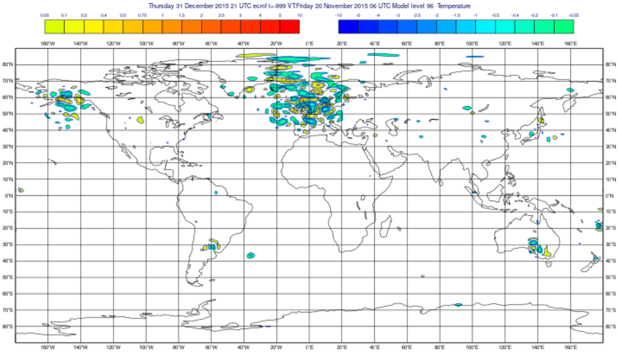


Figure: IFS; T increments difference; 500mb; $dx_{95}^{159} - dx_{159}^{*}$, where $dx_{159}^{*} = \mathbf{U}^{159} dv_{95}^{159}$.

Lessons learned 2 - IFS is overwhelmingly complex and difficult to understand

IFS operators

- $\hat{G}dx = y_0 + Gdx - (y_0 - \mathcal{G}x_{HR}) - y_0 = Gdx - d$
- $\hat{G}^T dy = G^T dy$
- $\hat{U}dv = Udv - dx_{fg}$
- $\hat{U}^T dx = U^T dx$

Evaluation of a gradient of the cost function (SIM4D):

$$\begin{aligned} g &= dv + \hat{U}^T \hat{G}^T R^{-1} \hat{G} \hat{U} dv = \\ &= dv + U^T G^T R^{-1} [G(Udv - dx_{fg}) + d] \end{aligned}$$

In particular the evaluation of the initial gradient:

$$\begin{aligned} g_0 &= dv_{fg} + \hat{U}^T \hat{G}^T R^{-1} \hat{G} \hat{U} dv_{fg} = \\ &= dv_{fg} + U^T G^T R^{-1} [G(Udv_{fg} - dx_{fg}) + d] = \\ &= dv_{fg} + U^T G^T R^{-1} d \end{aligned}$$

IFS: initial gradient calculation

$$
\begin{aligned}
g_0 &= \begin{bmatrix} dv_{fg} \\ 0 \end{bmatrix} + \begin{bmatrix} U^T & \\ & \widetilde{U}^T \end{bmatrix} \begin{bmatrix} G^T \\ P^T \end{bmatrix} R^{-1} \begin{bmatrix} G & P \end{bmatrix} \begin{bmatrix} U & \\ & \widetilde{U} \end{bmatrix} \begin{bmatrix} dv_{fg} \\ 0 \end{bmatrix} - \\
&- \begin{bmatrix} U^T & \\ & \widetilde{U}^T \end{bmatrix} \begin{bmatrix} G^T \\ P^T \end{bmatrix} R^{-1} \begin{bmatrix} G & P \end{bmatrix} \begin{bmatrix} dx_{fg} \\ d\beta_{fg} \end{bmatrix} + \begin{bmatrix} U^T & \\ & \widetilde{U}^T \end{bmatrix} \begin{bmatrix} G^T \\ P^T \end{bmatrix} R^{-1} d = \\
&= \begin{bmatrix} dv_{fg} \\ 0 \end{bmatrix} + \begin{bmatrix} U^T G^T R^{-1} G U dv_{fg} \\ \widetilde{U}^T P^T R^{-1} G U dv_{fg} \end{bmatrix} - \begin{bmatrix} U^T G^T R^{-1} G dx_{fg} + U^T G^T R^{-1} P d\beta_{fg} \\ \widetilde{U}^T P^T R^{-1} G dx_{fg} + \widetilde{U}^T P^T R^{-1} P d\beta_{fg} \end{bmatrix} + \begin{bmatrix} U^T G^T R^{-1} d \\ \widetilde{U}^T P^T R^{-1} d \end{bmatrix} = \\
&\approx \begin{bmatrix} dv_{fg} \\ 0 \end{bmatrix} - \begin{bmatrix} U^T G^T R^{-1} P d\beta_{fg} \\ \widetilde{U}^T P^T R^{-1} P d\beta_{fg} \end{bmatrix} + \begin{bmatrix} U^T G^T R^{-1} d \\ \widetilde{U}^T P^T R^{-1} d \end{bmatrix} = \\
&= \begin{bmatrix} dv_{fg} \\ 0 \end{bmatrix} + \begin{bmatrix} U^T G^T R^{-1} (d - P d\beta_{fg}) \\ \widetilde{U}^T P^T R^{-1} (d - P d\beta_{fg}) \end{bmatrix} \\
&= \begin{bmatrix} dv_{fg} \\ 0 \end{bmatrix} + \begin{bmatrix} U^T G^T \\ \widetilde{U}^T P^T \end{bmatrix} R^{-1} (d - P d\beta_{fg}) = \\
&= \begin{bmatrix} dv_{fg} \\ 0 \end{bmatrix} + \begin{bmatrix} U^T & \\ & \widetilde{U}^T \end{bmatrix} \begin{bmatrix} G^T \\ P^T \end{bmatrix} R^{-1} (d - P d\beta_{fg})
\end{aligned}
$$

Variational Bias Correction implementation

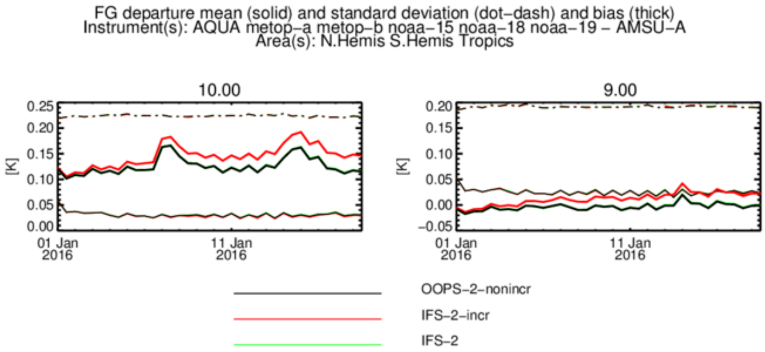


Figure: Non-incremental general OOPS VarBC implementation vs IFS. T255/T95/T159 experiment.

Lessons learned 3 - Proper implementation rather than a hack can sometimes be more expensive and more time consuming: Constrained VarBC

Non-linear cost function $\mathcal{J}_o^c(\beta) = \frac{1}{2}\frac{(\mathcal{P}(\beta)-b_o)^2}{\sigma_c^2}$

- $\mathcal{P}(\beta)$ is the non-linear bias correction operator,
- β is the bias correction parameter vector,
- b_o is the bias anchoring state vector and,
- σ_c is a weighting factor/

Quadratic cost function

$$\begin{aligned} J_o^c(d\beta) &= \frac{1}{2}\frac{(\mathcal{P}(\beta)+P(d\beta)-b_o)^2}{\sigma_c^2} = \frac{1}{2}\frac{(b+P(d\beta)-b_o)^2}{\sigma_c^2} = \\ &= \frac{1}{2}\frac{(P(d\beta)-(b_o-b))^2}{\sigma_c^2} = \frac{1}{2}\frac{(P(d\beta)-d_c)^2}{\sigma_c^2} \end{aligned}$$

Gradient of the quadratic cost function

$$\frac{\partial J_o^c(d\beta)}{\partial(d\beta)} = P^T\frac{1}{\sigma_c^2}Pd\beta - P^T\frac{1}{\sigma_c^2}d_c$$

Constrained VarBC

OOPS implementation Let's introduce the constrained VarBC term into the gradient of the quadratic cost function

$$\left(\begin{bmatrix} I & \\ & \widetilde{U}^T_{\beta,k} B_\beta^{-1} \widetilde{U}_{\beta,k}\end{bmatrix} + \begin{bmatrix} U^T G^T R^{-1} G U & U^T G^T R^{-1} P \widetilde{U}_{\beta,k} \\ \widetilde{U}^T_{\beta,k} P^T R^{-1} G U & \widetilde{U}^T_{\beta,k} P^T (R^{-1} + \frac{1}{\sigma_c^2}) P \widetilde{U}_{\beta,k}\end{bmatrix}\right) \begin{bmatrix} dv_k \\ dv_{\beta,k}\end{bmatrix} =$$

$$\begin{bmatrix} -\sum_{j=0}^{k-1} dv_j \\ -\sum_{j=0}^{k-1} \widetilde{U}^T_{\beta,k} B_\beta^{-1} d\beta_j \end{bmatrix} + \begin{bmatrix} U^T G^T R^{-1} d_{k-1} \\ \widetilde{U}^T_{\beta,k} P^T (R^{-1} d_{k-1} + \frac{1}{\sigma_c^2} d_{c,k-1}) \end{bmatrix}$$

Which can be written as:

$$\left(\begin{bmatrix} I & \\ & \widetilde{U}^T_{\beta,k} B_\beta^{-1} \widetilde{U}_{\beta,k}\end{bmatrix} + \begin{bmatrix} U^T & \\ & \widetilde{U}^T_{\beta,k}\end{bmatrix} \begin{bmatrix} G^T & \\ P^T & P^T\end{bmatrix} \begin{bmatrix} R^{-1} & \\ & \frac{1}{\sigma_c^2}\end{bmatrix} \begin{bmatrix} G & P \\ & P\end{bmatrix} \begin{bmatrix} U & \\ & \widetilde{U}_{\beta,k}\end{bmatrix}\right) \begin{bmatrix} dv_k \\ dv_{\beta,k}\end{bmatrix} =$$

$$\begin{bmatrix} -\sum_{j=0}^{k-1} dv_j \\ -\sum_{j=0}^{k-1} \widetilde{U}^T_{\beta,k} B_\beta^{-1} d\beta_j \end{bmatrix} + \begin{bmatrix} U^T & \\ & \widetilde{U}^T_{\beta,k}\end{bmatrix} \begin{bmatrix} G^T & \\ P^T & P^T\end{bmatrix} \begin{bmatrix} R^{-1} d_{k-1} & \\ & \frac{1}{\sigma_c^2} d_{c,k-1}\end{bmatrix}$$

Lessons learned 4 - Devil is in the detail: Second-level preconditioning

Limited memory preconditioners - general formulation

Let $\mathbf{A}$ be an $n \times n$ symmetric positive definite matrix, and let $\mathbf{S_l}$ be a $n \times l$ matrix, with $l << n$, whose column $\mathbf{s_1}, .., \mathbf{s_l}$ are assumed to be A-conjugate, i.e.

$$\mathbf{s_i}^{\mathbf{T}} \mathbf{A} \mathbf{s_j} \begin{Bmatrix} > 0 & if & j = i \\ = 0 & if & j \neq i \end{Bmatrix}$$

The limited memory preconditioner is defined as:

$$\mathbf{K_l} = \left(\mathbf{I_n} - \sum_{\mathbf{i=1}}^{\mathbf{l}} \frac{\mathbf{s_i s_i^T}}{\mathbf{s_i^T A s_i}} \mathbf{A} \right) \left(\mathbf{I_n} - \sum_{\mathbf{i=1}} \mathbf{lA} \frac{\mathbf{s_i s_i^T}}{\mathbf{s_i^T A s_i}} \right) + \sum_{\mathbf{i=1}}^{\mathbf{l}} \frac{\mathbf{s_i s_i^T}}{\mathbf{s_i^T A s_i}}$$

Second-level preconditioning

Spectral LMP

Normalized eigenpair $(\lambda_i, \mathbf{v_i})$ of an $n \times n$ symmetric positive definite matrix $\mathbf{A}$ satisfy:

$$\mathbf{v_i}^{\mathbf{T}}\mathbf{A}\mathbf{v_j} \begin{Bmatrix} \lambda_i > 0 & \textit{if} & j = i \\ = 0 & \textit{if} & j \neq i \end{Bmatrix}$$

and

$$\mathbf{v_i}^{\mathbf{T}}\mathbf{v_j} \begin{Bmatrix} = 1 & \textit{if} & j = i \\ = 0 & \textit{if} & j \neq i \end{Bmatrix}$$

Using $\mathbf{A}\mathbf{u_i} = \lambda\mathbf{u_i}$, we get the following expression for the Spectral LMP:

$$\mathbf{K_l^{spectral}} = \mathbf{I_n} + \sum_{i=1}^{\ell} (\lambda_i - 1)\, \mathbf{v_i}\mathbf{v_i}^{\mathbf{T}} \approx J''$$

In practice we use Ritz pairs $(\tilde{\lambda}_i, \tilde{\mathbf{v}}_i)$, which shall be orthonormal and $\mathbf{A}$ conjugate.

Second-level preconditioning

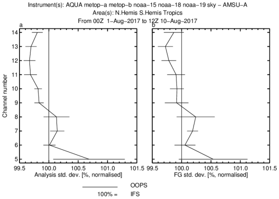


Figure: Tco399-T95-T159; no preconditioning

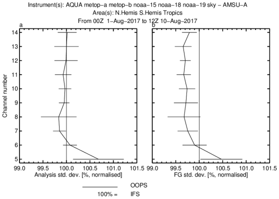


Figure: Tco399-T95-T159-T255; no preconditioning

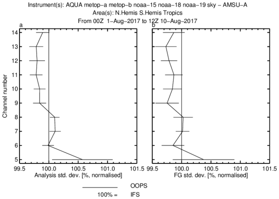


Figure: Tco399-T95-T159; with preconditioning

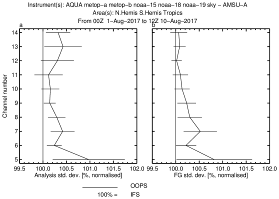


Figure: Tco399-T95-T159-T255; with preconditioning

Second-level preconditioning

If preconditioning has been employed, the Ritz vectors and values provide approximation to preconditioned Hessian, $\mathbf{M}^{-\frac{1}{2}} J'' \mathbf{M}^{-\frac{1}{2}}$, of the form

$$\mathbf{M}^{-\frac{1}{2}} J'' \mathbf{M}^{-\frac{1}{2}} \approx \mathbf{I} + \sum_{i=1}^{K} (\lambda_i - 1) \mathbf{v_i} \mathbf{v_i}^{\mathbf{T}}$$

Multiplying to the left and right by $\mathbf{M}^{\frac{1}{2}}$, gives

$$J'' \approx \mathbf{M} + \sum_{i=1}^{K} (\lambda_i - 1)(\mathbf{M}^{\frac{1}{2}} \mathbf{v_i})(\mathbf{M}^{\frac{1}{2}} \mathbf{v_i})^{\mathbf{T}}$$

$$J'' \approx \mathbf{I} + \sum_{i=1}^{L+K} \mathbf{s_i} \mathbf{s_i}^{\mathbf{T}}$$

where:

$$\mathbf{s_i} = \left\{ \begin{array}{ccc} (\mu_i - 1)^{\frac{1}{2}} \mathbf{w_i} & \text{for} & i = 1..L \\ (\lambda_{i-L} - 1)^{\frac{1}{2}} \mathbf{M}^{\frac{1}{2}} \mathbf{v_{i-L}} & \text{for} & i = L+1..L+K \end{array} \right\}$$

Second-level preconditioning

When combining Ritz vectors from multiple minimizations, in general:

$$\tilde{\mathbf{s}}_\mathbf{i}^\mathbf{T}\tilde{\mathbf{s}}_\mathbf{j} \begin{Bmatrix} \neq 1 & \textit{if} & j = i \\ \neq 0 & \textit{if} & j \neq i \end{Bmatrix}$$

In this case the approximation to the inverse of the Hessian $(J'')^{-1}$ is not readily available.

We need to resort to the Shermann-Morrison-Woodbury formula:

$$(\mathbf{A}+\mathbf{UCV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}\left(\mathbf{C}^{-1}+\mathbf{VA}^{-1}\mathbf{U}\right)^{-1}\mathbf{VA}^{-1}$$

Which then reads:

$$\begin{aligned} \left(\mathbf{I_n}+\mathbf{SI}_\ell\mathbf{S}^\mathbf{T}\right)^{-1} &= \mathbf{I_n} - \mathbf{S}\left(\mathbf{I}_\ell+\mathbf{S}^\mathbf{T}\mathbf{S}\right)^{-1}\mathbf{S}^\mathbf{T} \\ &= \mathbf{I_n} - \mathbf{S}(\mathbf{L}^{-1})^\mathbf{T}\mathbf{L}^{-1}\mathbf{S}^\mathbf{T} \\ &= \mathbf{I_n} - \bar{\mathbf{S}}\bar{\mathbf{S}}^\mathbf{T} \approx (\mathbf{J}'')^{-1} \end{aligned}$$

Second-level preconditioning

Recall $\mathbf{\bar{S}}$ is a matrix such that $\mathbf{I_n} - \mathbf{\bar{S}\bar{S}^T} = (\mathbf{J}'')^{-1}$, we can perform QR decomposition:

$$\mathbf{\bar{S}\bar{S}^T} = \mathbf{Q^T}(\mathbf{Q\bar{S}})(\mathbf{Q\bar{S}})^\mathbf{T}\mathbf{Q}$$

where:

- $\mathbf{Q}$ is an orthogonal matrix: $\mathbf{QQ}^T = \mathbf{I}$
- $\mathbf{Q\bar{S}}$ is an upper triangular matrix
- $(\mathbf{Q\bar{S}})(\mathbf{Q\bar{S}})^T$ has ℓ non-zero eigenvalues ρ_i with corresponding eigenvectors $\mathbf{p}_i$

The required orthonormal preconditioning vectors are given by $\mathbf{w_i} = \mathbf{Q^T p_i}$.
To cast in standard form denote $\mu_i = 1 - \frac{1}{\rho_i}$:

$$\mathbf{K_l^{spectral}} = \mathbf{I_n} + \sum_{i=1}^{\ell} (\mu_i - 1)\,\mathbf{w_i w_i^T} \approx \mathbf{J}''$$

$$\left(\mathbf{K_l^{spectral}}\right)^{-1} = \mathbf{I_n} + \sum_{i=1}^{\ell} \left(\frac{1}{\mu_i} - 1\right) \mathbf{w_i w_i^T} \approx (\mathbf{J}'')^{-1}$$

NOTE: we need to form $\ell \times \ell$ matrix formed by non-zero elements of $(\mathbf{QU})(\mathbf{QU})^T$; to do that we need to move sections of preconditioning vectors through the C++/Fortan interface.

Second-level preconditioning

Control vector is in the wavelet space; A non-orthogonal transform on the sphere is defined by a set of functions of great-circle distance:

$$\{\psi_j(|\mathbf{r}|)\,; j = 1 \ldots K\}$$

with the property

$$\sum_j \hat{\psi}_j^2(n) = 1$$

the "transform" pair is then defined:

$$f_j = \psi_j \otimes f, \quad f = \sum_j \psi_j \otimes f_j$$

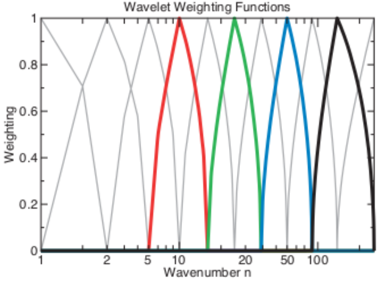


Figure: Weighting functions for the different wavenumber bands in "Wavelet" J_b. Courtesy M. Fisher.

Summary

OOPS project board announced the project has achieved targets and will be closed. We will hold an ECMWF wide celebration on the 30th of August.
Summary:

- It took several years and a number of dedicated people to refactor IFS and interface it to OOPS; this work is <u>not complete</u>;
- OOPS system is much more resilient and robust;
- It may be more difficult to implement certain new ideas properly in OOPS rather that hack them in as before, but it is the only sustainable path;
- C++/Fortran mixed code can be a challenge; initial learning curve is steep;
- IFS required tailored solutions, but object orientation makes the developments straight forward;


Annual
Seminar
2018
Earth System Assimilation
10–13 September
http://www.ecmwf.int/en/learning/workshops-and-seminars/en/annual-seminar-2018
ECMWF