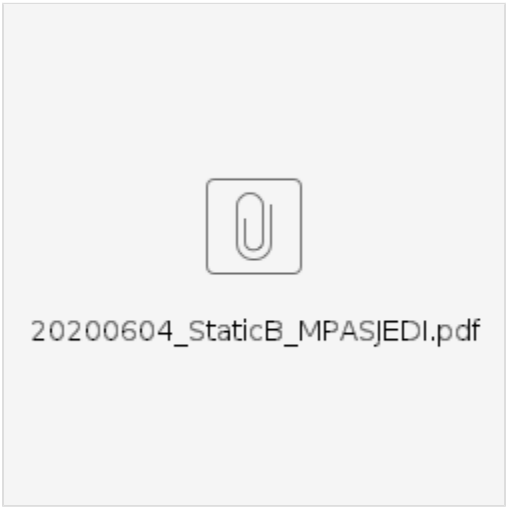
2020-06-04: Static B Modeling

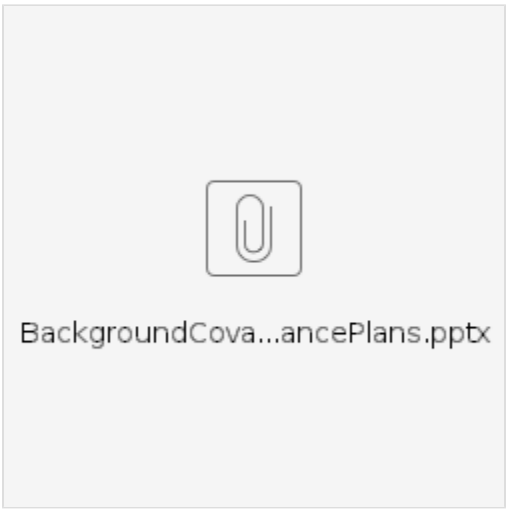
Yannick opened the meeting saying that there were no specific announcements and introduced the topic of the day, which is to discuss recent progress and plans for modeling the static B matrix. This includes the static B Matrix developed for specific models as well as any opportunities we can identify to generalize these models and make them more generic for wider use.

BJ started by presenting the following slides describing recent work by the MPAS group.



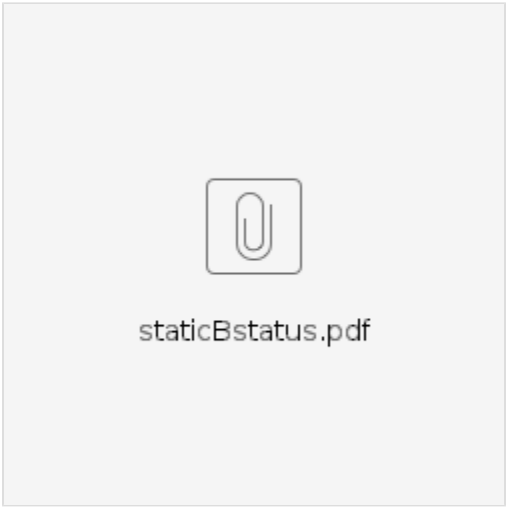
BJ began with an overview of the approach followed by MPAS, which is based on a univariate covariance matrix computed with BUMP, a linear variable change and vertical balance (also computed with BUMP), and a variable change from u, v to psi, chi. The latter variable change currently makes use of spherical harmonics and an interpolation onto a Gaussian grid. When computing the matrix inverse for the vertical regression component, they first implemented a Cholesky decomposition method but found that it was too noisy. So, they moved instead to a pseudo inverse based on eigen value decomposition. The resulting increments look reasonable from a dynamics standpoint and they have been able to run 1-month cycling experiments with a 110 km mesh. However, the results from the cycling runs were inferior to the 3DEnVar cycling experiment. Areas where the static B modeling can potentially be improved include adding a moisture control variable and using a more physical variable for the vertical coordinate (height or pressure as opposed to the vertical level they are using now). See slides for further details.

Marek then presented the following slides that summarize the status of B matrix modeling at UKMO:



See slides for details. One highlight was the use of atlas in the UMJEDI interface, where states and increments are represented as atlas fields. Yannick agreed that this is a promising approach and may help to make some aspects of the modeling more generic for wider use, in particular the variable transforms.

Dan then summarized the status of static B modeling with FV3 with the following slide:



Yannick then asked if anyone else had anything to add. Guillaume commented that they have had a static B matrix for MOM6 implemented in SOCA for some time. This was also constructed using BUMP. Yannick asked if this could be generalized for wider use. Guillaume responded that it is based mainly on the physics of the ocean so it would be of limited use for atmospheric B matrix modeling. But, it many be of some use for other ocean models.

Marek then asked for Dan to elaborate about an item on his slide about plans to use atlas for the Poisson solver. Dan said that the Poisson solver that is currently implemented uses a multi-grid method parallelized over levels. The current algorithm has some limitations. For example, the multigrid currently coarsens the grid by halving the number of points on each face of the cubed sphere. If you're running a standard, intermediate-resolution GEOS c96 application, then halving the grid would give you 43 points. Coarsening this further would require a different approach. Using atlas fields and potentially other atlas tools such as the mesh generation and interpolation could eliminate this problem, making the multigrid more flexible. Marek then asked about how one might construct the adjoint for this Poisson solver and Dan thought that it should be straightforward.

Sarah then gave an update from NRL. She said that they have made progress in extracting the operational Static B from NAVGEM and are now implementing it in Fortran in a way that can be used by JEDI. She also added that they may be able to make this generic enough that it could be implemented in SABER for general use.

Yannick encouraged this and encouraged others to do the same; anything that is generic enough to be useful for others should go into SABER. Just as ufo is a collection of obs operators, the intention for saber is that it should be a collection of B matrices and related tools. Guillaume asked if this is where we should put variable transforms. Benjamin responded yes, if they can be of use for other models. Guillaume thinks that the balance operators they use for the ocean could be generic and made a plan to add this to the "to do" list for the SOCA team.

Marek then asked for further elaboration on the plans for SABER. He described four general approaches to B matrix modeling and asked if there were plans to include all of them in SABER, namely:

1. BUMP
2. Diffusion
3. Recursive filter
4. Spectral

Benjamin responded that we should at least implement a spectral modeling approach. This would be the next priority on this list, providing a useful alternative for and consistency check for BUMP. We can make it generic if we include the capability to interpolate onto Gaussian grids. Regarding the other two items on the list, we might possibly want to include the recursive filter from GSI at some point. And, with regard to a diffusion approach, Benjamin said he has already worked with a colleague (Anthony Weaver) on comparing this approach to BUMP; the results show reasonable agreement.

Then there was some discussion about whether or not SABER required the use of unstructured grids. Benjamin responded that BUMP uses unstructured grids internally but includes interfaces that allow you to pass it structured grids, as represented for example by the atlas functionspace::NodeColumns class. It may be possible in the future to tune the modeling approach to the structure of the input data. For example, if the variables are already passed to SABER on a Gaussian grid, then it may be most efficient to use a spectral approach by default. Dan asked if we would need the grid in atlas format in order to take advantage of interpolation from one structured grid to another. This may facilitate the creation of a generic API and the use of atlas functionality (mesh generation, interpolation), but it should also be possible for models to implement their own customized interpolation procedures if they wish, to fully take advantage of their own grid structure and domain decomposition.

Chris S then asked about the time line for the adoption of atlas. In particular, it is used now in oops and saber but the models are not currently required to pass fields to saber in atlas format. Will this change? The short answer is yes - modelers are encouraged to begin using the atlas interfaces as soon as is convenient. Even now, they might offer some performance improvements for bump. This is because the deprecated BUMP interfaces that use the old oops unstructured grid class now make an extra copy, representing fields as atlas data structures that are then passed to the new interface. Benjamin added that atlas is now optional only for compatibility reasons and that he will help people switch over to the use of atlas. Chris also asked if there are other pieces of atlas that need adjoints. The answer there too is yes - for example, the interpolations.

Chris S then asked if we need to define an atlas mesh. Benjamin responded that, in order to use BUMP, this is not the case. You can pass BUMP fields that are represented as FunctionSpace::PointCloud objects that do not have connectivity information. Then bump will set up the necessary communication pathways. However, if you want to use other atlas tools, such as the native atlas interpolation classes, then it would be of benefit to generate an atlas mesh that includes the proper data decomposition and communication pathways. Atlas uses an external library called CGAL to generate a communication mesh through Delauney triangulation. Though BUMP does not currently use atlas for mesh generation, this may change in the future - there are plans to use more atlas functionality deeper in bump.

Guillaume asked how much work would it take to change to atlas data structures when interfacing with SABER. Benjamin responded that it depends on how sophisticated you want to get. If you treat your grid as an unstructured grid with no connectivity information, then it is easy to represent it as an atlas PointCloud and pass it to SABER - this could be accomplished perhaps in a day. However, if you want to exploit a particular grid structure for efficiency, then this could take a bit longer to implement. For example, Dan is currently working on adding cubed sphere support to atlas. It was mentioned that the atlas team is currently working on implementing a NEMO grid; the resulting data structures might be generic enough to use with MOM6.

Chris S then asked Marek for elaboration on the UM interface for atlas. Marek said atlas classes are used for the state, increment, and other objects and will help to make the variable transform generic enough to move into SABER. Chris S also wanted to talk with Dan offline about the process of adding a grid and mesh into atlas.

Yannick added that there had been plans for a code sprint focused on atlas integration into JEDI. But, these plans were disrupted by the pandemic. We would still like to possibly arrange a virtual sprint to work on this.

Since our hour was nearly up, Yannick brought the meeting to a close. But, he encouraged all participants to let us know if you would like to discuss any aspect of these issues more (or other issues). There are already plans to have one of our Thursday morning focused discussions on atlas (time TBD). As always, if you have other ideas or request, either contact a member of the JEDI core team or enter it directly onto our ZenHub board.