Application of ENO technique to semi-Lagrangian interpolations

RC LACE stay report
Scientific supervisors: Petra Smolíková and Ján Mašek

Alexandra Crăciun
NMA, Romania

CHMI, Prague

18.05-12.06.2015

1 Introduction

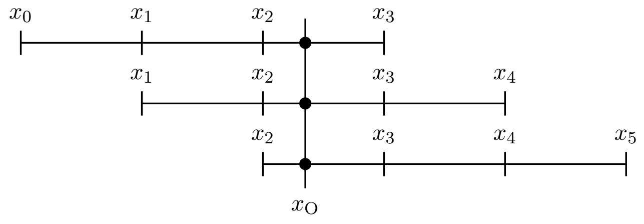
The purpose of this study was to continue the work started last year, which was to explore some alternative interpolators that are less overshooting than commonly used cubic Lagrange polynomial, but still accurate enough. One of these could be the ENO technique.

The evaluation of ENO interpolation in some 2D experiments (after some preliminary tests made by Ján Mašek) showed that the second order ENO interpolation (with quadratic Lagrange polynomials) produces a very smooth solution such that it could not be taken into consideration as a replacement for the interpolators used currently in the model. In that case, another idea was to analyze the behaviour of this scheme with increased order (using cubic Lagrange polynomials).

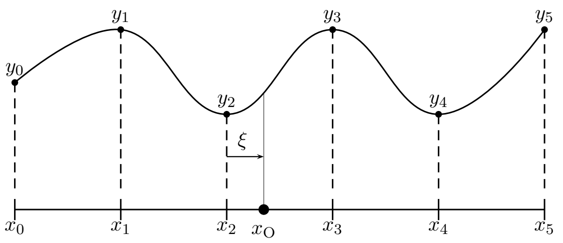
Even if third order ENO technique was not yet implemented in 2D model, results obtained with 1D linear advection equation proved to be encouraging. The task was to evaluate if this remains true when this interpolation scheme is implemented in the full model.

2 Implementation in the cycle 40t1

The third order ENO scheme refers at the possibility to choose between three stencils based on the smoothness of the solution. For a third order interpolation scheme, a 4 points stencil is needed to construct a 1D cubic interpolator.



Having six nodes and the interpolation point in the central interval, we can choose either the left, the central or the right 4 points stencil to get the interpolated value. The choice between the stencils is made after finding the smaller absolute value of the finite difference approximation for third derivate of the function for each of the three stencils.



For example: if y is the function to be interpolated, $y_i = y(x_i)$ and $|y_4 - 3y_3 + 3y_2 - y_1| < |y_3 - 3y_2 + 3y_1 - y_0|$ and $|y_4 - 3y_3 + 3y_2 - y_1| < |y_5 - 3y_4 + 3y_3 - y_2|$ (which means the interpolation point is situated in the central stencil), we will use the cubic interpolator on stencil $1 - 2 - 3 - 4$ and the interpolator will have the form:

$$y(x_O) = \sum_{i=\overline{1,4}} \left(y_i \prod_{\substack{j=\overline{1,4} \\ j \neq i}} \frac{x_O - x_j}{x_i - x_j} \right)$$

The variable NSTENCILWIDE (which is not fully implemented in the code) could be used to specify the number of points needed for interpolation (NSTENCILWIDE=2 $\rightarrow$ 4 points stencil, NSTENCILWIDE=3 $\rightarrow$ 6 points stencil). LENO is the name of the general switch for the ENO interpolation scheme, which was put in the new module YOMENO and declared in the namelist namdyn.h. Then in the setup (sudyn) NSLDIMK is set to 3 for LENO=T. (Moreover, variables LENO and L3DTURB should not be set true simultaneously, as well for LENO=T and NSPLTHOI=1, since these cases will not be coded at the moment.)

According to NSTENCILWIDE, in routine ELASCAW the array dimensions must be set considering KDIMK (=NSLDIMK), which gives the stencil (1,2,3). For vertical, PVINTW array (weights for vertical interpolation) must be enlarged about one index identifying the stencil used.

In the current code there is only 4 points stencil available. The first step in order to apply the third order scheme is to increase the number of points in the stencil. The declaration of SL-pointer IL0 was modified to IL0(;,:,0:5) in GP_MODEL.

Assuming regular mesh size Δx (for horizontal), $\xi = \dfrac{x_O - x_2}{\Delta x}$ and denoting $w_i^1(\xi)$, $i = \overline{1,4}$ the weights corresponding to the central stencil, we get:

$$w_1^1(\xi) = (-1/6)\xi(\xi-1)(\xi-2),\ w_2^1(\xi) = (1/2)(\xi+1)(\xi-1)(\xi-2)$$

$$w_3^1(\xi) = (-1/2)(\xi+1)\xi(\xi-2),\ w_4^1(\xi) = (1/6)(\xi+1)\xi(\xi-1),$$

so that:

$$y(x_O) = y_1(-1/6)\xi(\xi-1)(\xi-2) + y_2(1/2)(\xi+1)(\xi-1)(\xi-2)$$
$$+y_3(-1/2)(\xi+1)\xi(\xi-2) + y_4(1/6)(\xi+1)\xi(\xi-1).$$

Weights have also been calculated for the case when the interpolation point is not in the central stencil. The weights will have similar form: $w_i^0(\xi) = w_i^1(\xi+1)$, $w_i^2(\xi) = w_i^1(\xi-1)$, $i = \overline{1,4}$, for the left (w_i^0) and the right (w_i^2) stencil. These 3 sets of weights are calculated in ELASCAW through LASCAW_CLO.

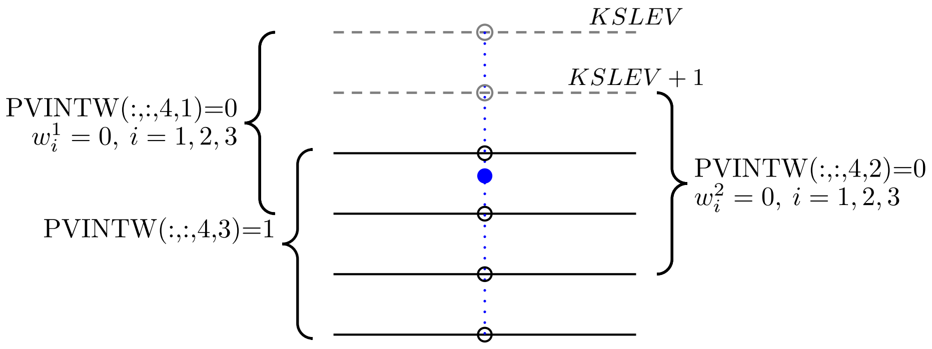
For horizontal interpolation (longitude and latitude) ξ is either PDLO or PDLAT, and the weights (PCLO and PCLA) are computed in subroutine LASCAW_CLO (where only three weights are calculated, since all four sum up to one), which is called from ELASCAW. For third order ENO scheme, this call must be made in a loop through KDIMK in order to obtain three sets for three stencils.

For vertical interpolation, vertical cubic interpolation weights are calculated in LASCAW_VINTW which is also called from ELASCAW. For the case of irregular grid in vertical direction, we use the same interpolation formulae as for horizontal, but the denominators in the weights formula may not be replaced with simple Δx and have to be calculated in subroutine SUVSLETA in array VCUICO for each of the three stencils. For central stencil, we have:

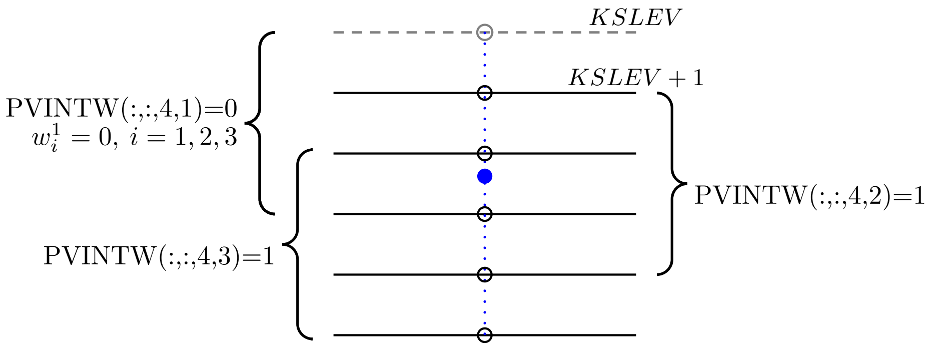
$$y(x_O) = y_1\frac{(x_O-x_2)(x_O-x_3)(x_O-x_4)}{(x_1-x_2)(x_1-x_3)(x_1-x_4)} + y_2\frac{(x_O-x_1)(x_O-x_3)(x_O-x_4)}{(x_2-x_1)(x_2-x_3)(x_2-x_4)}$$
$$+y_3\frac{(x_O-x_1)(x_O-x_2)(x_O-x_4)}{(x_3-x_1)(x_3-x_2)(x_3-x_4)} + y_4\frac{(x_O-x_1)(x_O-x_2)(x_O-x_3)}{(x_4-x_1)(x_4-x_2)(x_4-x_3)}$$

Close to the vertical boundaries, two cases are treated differently:

- the interpolation point (blue dot in figure below) is next to the top layer (KLEV=KSLEV) we consider the weights corresponding to the first two stencils (w_i^1 and w_i^2, $i = 1, 2, 3$) to be equal to 0 (in ELASCAW); the final interpolation (in vertical, in subroutine LAITRI) will be cubic, applied on the third stencil;



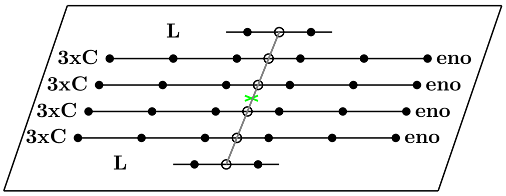
- the interpolation point is next to the second used layer (KLEV=KSLEV+1), then we assign the value 0 to the weights corresponding to the first stencil (w_i^1, $i = 1, 2, 3$); in that case, we will apply cubic ENO choice on the second and the third stencil.



As the forth value in the third dimension of array PVINTW, we can store the mask for stencils which may be used here, i.e. PVINTW(:,:,4,i) = 1 if the stencil i (1, 2 or 3) may be considered and PVINTW(:,:,4,i)=0 if not. Similar cases for bottom layer are treated analogously.

Summary of weights array dimensioning: KDIMK=3 (structure for KDIMK kept in ELASCAW, where it was, elsewhere NSLDIMENO is used not to mix too much with cases when $KDIMK > 1$), NSLDIMENO=3;
PCLO(:,:,3,2,KDIMK), PCLA(:,:,3,KDIMK), PVINTW(:,:,4,NSLDIMENO),
YRVSLETA%VCUICO(4,NSLDIMENO,0:NFLEVG-1),
YRVSLETA%VCUICOH(4,NSLDIMENO,0:NFLEVG) if needed.

The actual interpolations are made in subroutine LAITRI (as well as in LAIDDI for two dimensional fields). When using third order ENO, we need 6 vertical layers. For the first and the last layer we use linear interpolation. In each of the four intermediate layers we need to perform 2 linear interpolations on the boundaries and 4 ENO interpolations in the middle layers in the longitudinal direction, then 1 ENO interpolation in the latitudinal direction. Finally 1 ENO interpolation is made in the vertical direction. If we consider one ENO interpolation to be equivalent to three cubic interpolations, then it gives altogether (4 x 4 + 4 + 1) x 3 = 63 cubic interpolations instead of 7 cubic interpolations done originally in the code. The number of linear interpolations is not changed. Hence we see that the limits of the proposed technique are the computational demands.



A new subroutine LAIENO was created to calculate the interpolated value from 6 values of the interpolated field and 3 sets (for each stencil) of the 3 interpolation weights (forth value is not needed since weight sum up to 1) after the smoothest stencil is chosen. This routine is called from LAITRI and LAIDDI whenever needed.

3 Conclusion

The implementation in the code is not complete yet. Further steps will be made for debugging: firstly, to emulate the original code but with enlarged arrays under NSTENCILWIDE=3. A second part would be to check ENO separately in all three dimensions. For that purpose we define 2 logical arrays LENO(0:4) and LCUBIC(0:3). If LCUBIC(i)=TRUE then cubic Lagrange interpolation is applied (simple or ENO type) in vertical level corresponding to i. If LENO(i)=TRUE then this cubic interpolation uses ENO technique. LENO(4)=TRUE means eno technique applied in the vertical. Hence by setting LENO(0:4)=FALSE and LCUBIC(0)=LCUBIC(3)=FALSE, LCUBIC(1:2)=TRUE, we get the original code with 32 points interpolation. By changing logical keys we may check the functionality of the code separately in vertical and horizontal direction for different stencil levels. After that, specific tests must be made in order to evaluate the scheme.

Acknowledgements: I would like to thank Petra Smolíková and Ján Mašek for their help and support, as well to the whole team of CHMI for their hospitality.