On a new WENO algorithm of order $2r$ with improved accuracy close to discontinuities

Sergio Amat[a], Juan Ruiz*[a], Chi-Wang Shu[b]

[a] *Departamento de Matemática Aplicada y Estadística. Universidad Politécnica de Cartagena. Cartagena, Spain*
[b] *Division of Applied Mathematics. Brown University. Providence, Rhode Island, USA.*

Abstract

In this article we present a generalization of the WENO algorithm of order six for data discretized in the point values introduced in [S. Amat, J. Ruiz, C.-W. Shu, On new strategies to control the accuracy of WENO algorithms close to discontinuities, SIAM J. Numer. Anal. 57 (3) (2019) 1205 - 1237]. This generalization requires a slight modification of the mentioned algorithm. The objective is to obtain a new WENO algorithm of order $2r$ that is capable of rising the accuracy of the interpolation step by step close to the discontinuities. In order to keep the computational cost controlled, we will propose simple smoothness indicators of high order as a function of those used by the classical implementation of WENO. We will discuss about the accuracy of the new algorithm and present numerical experiments that support our claims.

Keywords: WENO-2r, high accuracy interpolation, improved adaption to discontinuities, generalization, 41A05, 41A10, 65D05, 65M06, 65N06

1. Introduction and review

In [1, 2], with the aim of improving the results obtained by ENO (essentially non oscillatory) algorithm, it was introduced in [3] the WENO (weighted essentially non oscillatory) algorithm. The technique was designed as a nonlinear convex combination of interpolants with adjacent sub-stencils. The weights of the combination were calculated estimating the smoothness of each sub-stencil. In [4], the authors introduced new smoothness indicators that were computationally more efficient than those proposed in [3]. In [5] we proposed a new technique of order of accuracy six aimed to increase the accuracy of the WENO-6 algorithm close to jump and kinks for the interpolation of functions discretized in the point values. This new technique consists in a redesign of the optimal weights of the original WENO algorithm with the objective of obtaining the maximum possible theoretical accuracy close to singularities while maintaining maximum accuracy at smooth regions of the data. It is a known fact that the classical implementation of WENO algorithm does not reach the maximum theoretical accuracy when more than one substencil is smooth. The objective of this paper is to generalize the interpolation technique presented in [5] for $2r$ point values, as this technique actually manages to reach the maximum theoretical accuracy when more than one substencil is smooth. In [5] we consider the space of finite sequences V and a uniform partition X of the interval $[a,b]$ in J subintervals,

$$X=\{x_i\}_{i=0}^{J},\quad x_0=a,\quad h=x_i-x_{i-1},\quad x_J=b.$$

For a piecewise smooth function f we consider a point value discretization,

$$f_i=f(x_i),\quad f=\{f_i\}_{i=0}^{J}. \tag{1}$$

The previous discretization is local and conserves the information only at the positions x_i. We consider that discontinuities are positioned far enough from each other.

The WENO-2r scheme uses the big stencil of $2r$ nodes $\{x_{j-r},\cdots,x_{j+r-1}\}$ when interpolating in the interval (x_{j-1},x_j) and it reaches order of accuracy $2r$ in smooth regions of f. By $S_k^r(j),\ k=0,\cdots,r-1$ we will denote the r^{th} sub-stencil of length $r+1$ that contains the interval (x_{j-1},x_j), where we want to interpolate by $S_k^r(j)=\{x_{j-r+k},\cdots,x_{j+k}\}$, for $k=0,\cdots,r-1$. WENO-2r algorithm is based on the construction of the convex combination,

$$I\left(x;f\right)=\sum_{k=0}^{r-1}\omega_k^r(j)p_{j-r+k}^r(x), \tag{2}$$

☆Research supported in part by Programa de Apoyo a la investigación de la fundación Séneca-Agencia de Ciencia y Tecnología de la Región de Murcia 20928/PI/18, by MTM2015-64382-P (MINECO/FEDER), by Spanish MCINN MTM 2014-54388P and by NSF grant DMS-1719410.

Email addresses: sergio.amat@upct.es (Sergio Amat), juan.ruiz@upct.es (Juan Ruiz*), chi-wang_shu@brown.edu (Chi-Wang Shu)
*Corresponding author.

where $\omega_k^r(j) \geq 0$, $k = 0, \cdots, r-1$ and $\sum_{k=0}^{r-1} \omega_k^r(j) = 1$. In (2), $p_{j-r+k}^r(x)$ denotes the r^{th} degree interpolation polynomial with stencil $S_k^r(j)$. We will consider the interpolation at the mid point of the interval (x_{j-1}, x_j), that will be denoted as $x_{j-\frac{1}{2}}$,

$$I(x_{j-\frac{1}{2}}; f) = \sum_{k=0}^{r-1} \omega_k^r(j) p_{j-r+k}^r(x_{j-\frac{1}{2}}). \tag{3}$$

The values of the $\omega_k^r(j)$ are forced to be those that allow to obtain order of accuracy $2r$ at $x_{j-\frac{1}{2}}$ when the stencil is smooth. When interpolating the discretized function $f(x_j)$, the objective is to obtain an interpolation polynomial that satisfies, $p_{j-r}^{2r-1}(x_{j-\frac{1}{2}}) = f(x_{j-\frac{1}{2}}) + O(h^{2r})$, based on the big stencil $\{x_{j-r}, \cdots, x_{j+r-1}\}$, through the convex combination of r interpolation polynomials of lower order, $p_{j-r+k}^r(x_{j-\frac{1}{2}}) = f(x_{j-\frac{1}{2}}) + O(h^{r+1})$, that use the smaller stencils $S_k^r(j)$. The classical WENO-2r algorithm imposes that the optimal weights are $\bar{C}_k^r(j) \geq 0, \forall k$ and $\sum_{k=0}^{r-1} \bar{C}_k^r(j) = 1$, such that,

$$p_{j-r}^{2r-1}\left(x_{j-\frac{1}{2}}\right) = \sum_{k=0}^{r-1} \bar{C}_k^r(j) p_{j-r+k}^r\left(x_{j-\frac{1}{2}}\right). \tag{4}$$

A formula for the optimal weights is obtained in [6],

$$\bar{C}_k^r(j) = \frac{1}{2^{2r-1}} \binom{2r}{2k+1}, \quad k = 0, \cdots, r-1. \tag{5}$$

For $r = 3$ the optimal weights are equal to,

$$\bar{C}_0^3(j) = \frac{3}{16}, \quad \bar{C}_1^3(j) = \frac{10}{16}, \quad \bar{C}_2^3(j) = \frac{3}{16}. \tag{6}$$

In [3] the authors design the weights $\omega_k^r(j)$ such that at smooth zones they satisfy,

$$\omega_k^r(j) = \bar{C}_k^r(j) + O(h^m), \quad k = 0, \cdots, r-1,$$

with $m \leq r-1$, in a manner that the interpolation in (3) reaches order of accuracy $2r$ when $m = r-1$,

$$f(x_{j-\frac{1}{2}}) - I\left(x_{j-\frac{1}{2}}; f\right) = O(h^{r+m+1}),$$

that is the same as the one obtained by the interpolation polynomial $p_{j-r}^{2r-1}(x)$ built using the big stencil. The expression for the weights in [3, 4] is,

$$\omega_k^r(j) = \frac{\alpha_k^r(j)}{\sum_{i=0}^{r-1} \alpha_i^r(j)}, \quad k = 0, \cdots, r-1 \text{ where } \alpha_k^r(j) = \frac{\bar{C}_k^r(j)}{(\epsilon + I_k^r(j))^t}, \tag{7}$$

that satisfies $\sum_k \omega_k^r(j) = 1$. The $I_k^r(j)$ are called *smoothness indicators* for $f(x)$ on the stencils $S_k^r(j)$. The parameter t is an integer used to assure the maximum order of accuracy close to the discontinuities. Depending on the implementation, the value of t might vary. We will choose $t = 2$. The parameter $\epsilon > 0$ avoids divisions by zero and takes the size of the smoothness indicators at smooth zones. We will set it to $\epsilon = 10^{-6}$. In order to ease the notation, in the rest of the article we will drop (j) in $S_k^r(j), \omega_k^r(j), \bar{C}_k^r(j), \alpha_k^r(j)$ and use $S_k^r, \omega_k^r, \bar{C}_k^r, \alpha_k^r$. Talking about the smoothness indicators, we will use those introduced in [5], that work well for detecting kinks and jumps in the function if the data is discretized in the point values (1),

$$I_k^n(j) = \sum_{l=2}^{n} h^{2l-1} \int_{x_{j-1}}^{x_j} \left(\frac{d^l}{dx^l} p_{j-r+k}^n(x)\right)^2 dx. \tag{8}$$

Let us denote the differences as $\delta_i^2 = f_i - 2\, f_{i+1} + f_{i+2}$, $\delta_i^3 = \delta_{i+1}^2 - \delta_i^2$ and $\delta_i^4 = \delta_{i+1}^3 - \delta_i^3$. For $r = 3$ the smoothness indicators in (8) can be expressed in terms of finite differences as,

$$I_0^3 = \frac{10}{3}(\delta_{j-3}^3)^2 + 3\delta_{j-3}^3\delta_{j-3}^2 + (\delta_{j-3}^2)^2, \quad I_1^3 = \frac{4}{3}(\delta_{j-2}^3)^2 + \delta_{j-2}^3\delta_{j-2}^2 + (\delta_{j-2}^2)^2, \quad I_2^3 = \frac{4}{3}(\delta_{j-1}^3)^2 - \delta_{j-1}^3\delta_{j-1}^2 + (\delta_{j-1}^2)^2. \tag{9}$$

For $r = 4$ they have the expression,

$$\begin{aligned}
I_0^4 &= \frac{27}{2}\,\delta_{j-4}^4\,\delta_{j-4}^3 + \frac{11}{3}\,\delta_{j-4}^4\,\delta_{j-4}^2 + 5\,\delta_{j-4}^3\,\delta_{j-4}^2 + \frac{2107}{240}\,(\delta_{j-4}^4)^2 + \frac{22}{3}\,(\delta_{j-4}^3)^2 + (\delta_{j-4}^2)^2,\\
I_1^4 &= \frac{547}{240}(\delta_{j-3}^4)^2 + \frac{10}{3}\,(\delta_{j-3}^3)^2 + (\delta_{j-3}^2)^2 + \frac{19}{6}\,\delta_{j-3}^4\,\delta_{j-3}^3 + \frac{2}{3}\,\delta_{j-3}^4\,\delta_{j-3}^2 + 3\,\delta_{j-3}^3\,\delta_{j-3}^2,\\
I_2^4 &= \frac{89}{80}\,(\delta_{j-2}^4)^2 + \frac{4}{3}\,(\delta_{j-2}^3)^2 + (\delta_{j-2}^2)^2 - \frac{1}{6}\,\delta_{j-2}^4\,\delta_{j-2}^3 - \frac{1}{3}\,\delta_{j-2}^4\,\delta_{j-2}^2 + \delta_{j-2}^3\,\delta_{j-2}^2,\\
I_3^4 &= \frac{547}{240}\,(\delta_{j-1}^4)^2 + \frac{4}{3}\,(\delta_{j-1}^3)^2 + (\delta_{j-1}^2)^2 - \frac{5}{2}\,\delta_{j-1}^4\,\delta_{j-1}^3 + \frac{2}{3}\,\delta_{j-1}^4\,\delta_{j-1}^2 - \delta_{j-1}^3\,\delta_{j-1}^2.
\end{aligned} \tag{10}$$

Theorem 3.1 of [5] should be proved for polynomials of any order $n \geq 3$ to serve our purposes.

Theorem 1.1. *At smooth zones, the smoothness indicators obtained through (8) satisfy,*

$$I_k^n = \left(h^2 p''(x_{j-1/2})\right)^2 \cdot (1 + O(h^2)).$$

Proof. Let p be a polynomial of degree $n \geq 3$, that is an n times continuously differentiable function in a neighborhood of the point $x_{j-1/2} = x_j - h/2$. Then, we can write $\left(p^{(l)}(x)\right)^2$ using the Taylor expansion of $p(x)$ as,

$$\begin{aligned}\left(p^{(l)}(x)\right)^2 &= \left(p^{(l)}(x_{j-1/2}) + p^{(l+1)}(x_{j-1/2})(x - x_{j-1/2}) + O((x - x_{j-1/2})^2)\right)^2 \\ &= \left(p^{(l)}(x_{j-1/2})\right)^2 + \left(p^{(l+1)}(x_{j-1/2})(x - x_{j-1/2})\right)^2 + 2p^{(l)}(x_{j-1/2})p^{(l+1)}(x_{j-1/2})(x - x_{j-1/2}) \\ &+ 2p^{(l)}(x_{j-1/2})p^{(l+2)}(x_{j-1/2})(x - x_{j-1/2})^2 + O((x - x_{j-1/2})^3).\end{aligned}$$

Integrating the previous expression between x_{j-1} and x_j, replacing in (8) and simplifying, we obtain the result, as for $l > 2$ all the terms of the summation have a size smaller or equal than $O(h^6)$,

$$I^n(j) = \sum_{l=2}^{n} h^{2l-1} \int_{x_{j-1}}^{x_j} \left(p^{(l)}(x)\right)^2 dx = \sum_{l=2}^{n} \left(h^l p^{(l)}(x_{j-1/2})\right)^2 \cdot (1 + O(h^2)) = \left(h^2 p''(x_{j-1/2})\right)^2 \cdot (1 + O(h^2)). \quad \square$$

As a particularization of (8) for $n = 3, 4$, (9) and (10) satisfy this property. Through this theorem, it is clear that the smoothness indicators are $O(h^4)$ at smooth zones and looking at their expression, it is clear that they are $O(1)$ when they touch a jump discontinuity. This theorem is one of the keys needed to prove the accuracy of the new algorithm introduced in [5]. In this article, instead of using high order smoothness indicators obtained directly through (8), we will use a function of the indicators obtained through the polynomials of lowest order involved in the algorithm with the aim of reducing the computational effort. We want Theorem 1.1 to hold while keeping $O(1)$ at discontinuities, so a good option is to define the smoothness indicators of high order as sums of the smoothness indicators used by the classical WENO of order $2r$. For $r = 3$ (in this case we have a stencil of 6 points) we can use as smoothness indicators of four points I_0^3, I_1^3 and I_2^3 defined through (8) and then, as smoothness indicators of five points, I_0^4 and I_1^4 defined as,

$$I_0^4 = I_0^3 + I_1^3, \quad I_1^4 = I_1^3 + I_2^3. \tag{11}$$

For $r = 4$ (in this case we have a stencil of 8 points) we will use as smoothness indicators of five points I_0^4, I_1^4, I_2^4 and I_3^4 defined through (8) and then as smoothness indicators of six and seven points $I_0^5, I_1^5, I_2^5, I_0^6, I_1^6$ defined as,

$$I_0^5 = I_0^4 + I_1^4, \quad I_1^5 = I_1^4 + I_2^4, \quad I_2^5 = I_2^4 + I_3^4, \quad I_0^6 = I_0^4 + I_1^4 + I_2^4, \quad I_1^6 = I_1^4 + I_2^4 + I_3^4. \tag{12}$$

For higher orders, we follow a similar strategy. This design of the smoothness indicators significantly reduce the computational cost obtained when using the expression in (8) in the whole algorithm, that was the option chosen in [5].

2. The new algorithm for $r = 3$

In [5] we analyze how to construct the new algorithm for $r = 3$, i.e. using $2r = 6$ points. We use the polynomials of degree 3, $p_0^3(x), p_1^3(x)$ and $p_2^3(x)$ in order to write the convex combination in (4). In this case we have two polynomials of degree 4 (stencil of 5 points), denoted as $p_0^4(x), p_1^4(x)$ and one polynomial of degree 5 (big stencil of 6 points), denoted as $p_0^5(x)$. We can build the two polynomials of degree 4 as a convex combination of the polynomials of degree 3,

$$p_0^4(x_{j-1/2}) = \frac{3}{8}p_0^3(x_{j-1/2}) + \frac{5}{8}p_1^3(x_{j-1/2}), \quad p_1^4(x_{j-1/2}) = \frac{5}{8}p_1^3(x_{j-1/2}) + \frac{3}{8}p_2^3(x_{j-1/2}). \tag{13}$$

And we can write the polynomial of degree 5 as a convex combination of the polynomials of degree 4,

$$p_0^5(x_{j-1/2}) = \frac{1}{2}p_0^4(x_{j-1/2}) + \frac{1}{2}p_1^4(x_{j-1/2}). \tag{14}$$

Remark 2.1. *In [5] we use a slightly different construction that provides the same order of accuracy as (14). The generalization requires the process described here.*

Now we can design two vectors of optimal weights $\mathbf{C_0^4}, \mathbf{C_1^4}$, each of which is suitable for a particular position of the discontinuity. The vectors will be composed of the weights in (13) and have the following expression,

$$\mathbf{C_0^4} = \left(\frac{3}{8}, \frac{5}{8}, 0\right), \quad \mathbf{C_1^4} = \left(0, \frac{5}{8}, \frac{3}{8}\right). \tag{15}$$

If the big stencil is $\{x_{j-3}, x_{j-2}, x_{j-1}, x_j, x_{j+1}, x_{j+2}\}$, then it is convenient to use $\mathbf{C_0^4}$ if the discontinuity is in the interval (x_{j+1}, x_{j+2}) and $\mathbf{C_1^4}$ if the discontinuity is in the interval (x_{j-3}, x_{j-2}). It is desirable that, if there is no discontinuity affecting

the stencil, a combination of the vectors in (15) would return the weights in (6), so that we can achieve maximum accuracy. This combination is,

$$\bar{\mathbf{C}}^{\mathbf{3}} = \frac{1}{2}\mathbf{C_0^4} + \frac{1}{2}\mathbf{C_1^4} = \frac{1}{2}\left(\frac{3}{8},\frac{5}{8},0\right) + \frac{1}{2}\left(0,\frac{5}{8},\frac{3}{8}\right) = \left(\frac{3}{16},\frac{10}{16},\frac{3}{16}\right) = \left(\bar{C}_0^3,\bar{C}_1^3,\bar{C}_2^3\right). \tag{16}$$

We can see how this process resembles a WENO algorithm performed step by step in order to grow the accuracy one step at a time. Now, instead of using fixed weights multiplying the vectors in (16), we can build nonlinear weights as in the original WENO algorithm with the aim of create a convex combination of the vectors in (15). We will replace the original WENO optimal weights in (7) by the resulting convex combination. Thus, in order to assure the adaption of the resulting weights, we will use the possible smoothness indicators of 4 and 5 points that arise from the 6 points stencil. We can denote by $\tilde{\omega}_k^n$ the quotients,

$$\tilde{\omega}_0^4 = \frac{\tilde{\alpha}_0^4}{\tilde{\alpha}_0^4+\tilde{\alpha}_1^4}, \quad \tilde{\omega}_1^4 = \frac{\tilde{\alpha}_1^4}{\tilde{\alpha}_0^4+\tilde{\alpha}_1^4}, \quad \text{with} \quad \tilde{\alpha}_0^4 = \frac{1/2}{(\epsilon+I_0^4)^t}, \quad \tilde{\alpha}_1^4 = \frac{1/2}{(\epsilon+I_1^4)^t}. \tag{17}$$

Now we can just define the adapted optimal weights as,

$$(\tilde{C}_0^3,\tilde{C}_1^3,\tilde{C}_2^3) = \tilde{\omega}_0^4\mathbf{C_0^4} + \tilde{\omega}_1^4\mathbf{C_1^4}. \tag{18}$$

These nonlinear optimal weights $\tilde{C}_k^r$ are used in place of the optimal weights $\bar{C}_k^r$ in the expression (7). The smoothness indicators in (17) are those presented in (11) and the smoothness indicators that appear in (7) are the ones shown in (9).

Now we can reproduce some results from [5]. Theorem 3.2 of [5] is equivalent to the next theorem, and the proof is analogous:

Theorem 2.1. *For $r = 3$ the accuracy of the interpolation at adjacent intervals to an isolated discontinuity is $\cdots, O(h^6), O(h^5), O(h^4), O(1), O(h^4), O(h^5), O(h^6), \cdots$.*

Corollary 3.3 and Theorem 3.4 from [5] exactly hold for this case:

Corollary 2.2. *For $r = 3$, $t \geq 1$, and $\epsilon \leq h^4$ the new WENO interpolant is at least as good as WENO interpolant close to discontinuities.*

Theorem 2.3. *The new algorithm satisfies the ENO property for $t \geq 2$, satisfying at the same time Theorem 2.1.*

The proof for the previous theorems and corollary is a reproduction of the steps followed in [5].

3. The new algorithm for $r = 4$

We can proceed in the same manner for higher order schemes. For example, for WENO-8 we start from the stencil of 8 points $\{x_{j-4}, x_{j-3}, x_{j-2}, x_{j-1}, x_j, x_{j+1}, x_{j+2}, x_{j+3}\}$. We will have four polynomials of degree 4 (stencil of 5 points), denoted as $p_0^4(x), p_1^4(x), p_2^4(x), p_3^4(x)$, three polynomials of degree 5 (stencil of 6 points), denoted as $p_0^5(x), p_1^5(x), p_2^5(x)$, two polynomials of degree 6 (stencil of 7 points), denoted as $p_0^6(x), p_1^6(x)$ and one polynomial of degree 7 (big stencil of 8 points), denoted as $p_0^7(x)$. As before, we want to obtain vectors of weights that progressively attain the maximum possible theoretical accuracy. In a similar manner as for $r = 3$, we can write the polynomials of degree 5 using the polynomials of degree 4,

$$\begin{aligned} p_0^5(x_{j-1/2}) &= \frac{3}{10}p_0^4(x_{j-1/2}) + \frac{7}{10}p_1^4(x_{j-1/2}), \quad p_1^5(x_{j-1/2}) = \frac{1}{2}p_1^4(x_{j-1/2}) + \frac{1}{2}p_2^4(x_{j-1/2}), \\ p_2^5(x_{j-1/2}) &= \frac{7}{10}p_2^4(x_{j-1/2}) + \frac{3}{10}p_3^4(x_{j-1/2}). \end{aligned} \tag{19}$$

Then we can write the polynomials of degree 6 using the polynomials of degree 5,

$$p_0^6(x_{j-1/2}) = \frac{5}{12}p_0^5(x_{j-1/2}) + \frac{7}{12}p_1^5(x_{j-1/2}), \quad p_1^6(x) = \frac{7}{12}p_1^5(x_{j-1/2}) + \frac{5}{12}p_2^5(x_{j-1/2}). \tag{20}$$

Finally, the two polynomials of degree 6 can be used to obtain the polynomial of degree 7,

$$p_0^7(x_{j-1/2}) = \frac{1}{2}p_0^6(x_{j-1/2}) + \frac{1}{2}p_1^6(x_{j-1/2}). \tag{21}$$

Now, similarly as before, we can design three vectors of optimal weights $\mathbf{C_0^5}, \mathbf{C_1^5}, \mathbf{C_2^5}$, each of which is suitable for a particular position of the discontinuity. The vectors will be composed of the weights in (19),

$$\mathbf{C_0^5} = \left(\frac{3}{10},\frac{7}{10},0,0\right), \quad \mathbf{C_1^5} = \left(0,\frac{1}{2},\frac{1}{2},0\right), \quad \mathbf{C_2^5} = \left(0,0,\frac{7}{10},\frac{3}{10}\right). \tag{22}$$

These vectors can be multiplied by the weights calculated in (20) and (21) to obtain the desired weights at smooth zones,

$$\bar{\mathbf{C}}^{\mathbf{4}} = \frac{1}{2}\left(\frac{5}{12}\mathbf{C_0^5} + \frac{7}{12}\mathbf{C_1^5}\right) + \frac{1}{2}\left(\frac{7}{12}\mathbf{C_1^5} + \frac{5}{12}\mathbf{C_2^5}\right) = \left(\frac{1}{16},\frac{7}{16},\frac{7}{16},\frac{1}{16}\right) = \left(\bar{C}_0^4,\bar{C}_1^4,\bar{C}_2^4,\bar{C}_3^4\right), \tag{23}$$

that are precisely the weights in (5) for $r = 4$. Now we can proceed as in the previous section and build a non linear convex combination of the vectors in (22) that provides the optimal nonlinear weights to be used in (7). In this case, we can denote by $\tilde{\omega}_k^n$ the quotients,

$$\tilde{\omega}_0^5 = \frac{\tilde{\alpha}_0^5}{\tilde{\alpha}_0^5 + \tilde{\alpha}_1^5}, \quad \tilde{\omega}_1^5 = \frac{\tilde{\alpha}_1^5}{\tilde{\alpha}_0^5 + \tilde{\alpha}_1^5}, \quad \tilde{\omega}_2^5 = \frac{\tilde{\alpha}_2^5}{\tilde{\alpha}_2^5 + \tilde{\alpha}_3^5}, \quad \tilde{\omega}_3^5 = \frac{\tilde{\alpha}_3^5}{\tilde{\alpha}_2^5 + \tilde{\alpha}_3^5}, \quad \tilde{\omega}_0^6 = \frac{\tilde{\alpha}_0^6}{\tilde{\alpha}_0^6 + \tilde{\alpha}_1^6}, \quad \tilde{\omega}_1^6 = \frac{\tilde{\alpha}_1^6}{\tilde{\alpha}_0^6 + \tilde{\alpha}_1^6}, \tag{24}$$

with,

$$\tilde{\alpha}_0^5 = \frac{5/12}{(\epsilon + I_0^5)^t}, \quad \tilde{\alpha}_1^5 = \frac{7/12}{(\epsilon + I_1^5)^t}, \quad \tilde{\alpha}_2^5 = \frac{7/12}{(\epsilon + I_1^5)^t}, \quad \tilde{\alpha}_3^5 = \frac{5/12}{(\epsilon + I_2^5)^t}, \quad \tilde{\alpha}_0^6 = \frac{1/2}{(\epsilon + I_0^6)^t}, \quad \tilde{\alpha}_1^6 = \frac{1/2}{(\epsilon + I_1^6)^t}. \tag{25}$$

Now we can just define the adapted optimal weights replacing the fixed weights in (23) by the nonlinear weights in (24),

$$(\tilde{C}_0^4, \tilde{C}_1^4, \tilde{C}_2^4, \tilde{C}_3^4) = \tilde{\omega}_0^6 \left(\tilde{\omega}_0^5 \mathbf{C_0^5} + \tilde{\omega}_1^5 \mathbf{C_1^5}\right) + \tilde{\omega}_1^6 \left(\tilde{\omega}_2^5 \mathbf{C_1^5} + \tilde{\omega}_3^5 \mathbf{C_2^5}\right). \tag{26}$$

As before, these nonlinear optimal weights $\tilde{C}_k^r$ are used in place of the optimal weights $\bar{C}_k^r$ in the expression (7). The smoothness indicators in (25) are those presented in (12) and the smoothness indicators that appear in (7) are obtained using five points, and have the expression shown in (10). Following [5], theorems equivalent to Theorem 2.1, Corollary 2.2 and Theorem 2.3 can be given for this case. The proofs would be very similar to the ones presented in [5] and straightforward if one follows the steps presented in this reference.

4. A generalization to order $2r$

Following what has been done in previous section, we can build the optimal weights for WENO-2r algorithm executing the steps shown in Algorithm 1. The final optimal weights would be stored in the vector $\bar{\mathbf{C}}_0^{2r-1}$ and the components of this vector should replace the $\bar{C}_k^r$ in (7). It can be easily checked that the algorithm reproduces the expressions in (18) and (26). It is important to remark that line 4 of the algorithm implies the solution of a system of two equations with two unknowns every time it is executed. It is also important to have in mind that Algorithm 1 is only executed once in order to obtain the expression for the nonlinear optimal weights.

Again, Following [5], theorems equivalent to Theorem 2.1, Corollary 2.2 and Theorem 2.3 can be given for any value of r. It can be proved that any particular WENO-2r scheme built using Algorithm 1 with $t \geq 2$ and $\epsilon \leq h^4$ satisfy the design requirement of reaching $O(h^{2r})$ accuracy at smooth zones and reduce its accuracy close to the discontinuity following the pattern $O(h^{2r}), O(h^{2r-1}), \cdots, O(h^{r+2}), O(h^{r+1}), O(1), O(h^{r+1}), O(h^{r+2}), \cdots, O(h^{2r-1}), O(h^{2r})$. We can mention here that the additional computational cost of the new algorithm is the calculation of the weights $\tilde{\omega}_0^4$ and $\tilde{\omega}_1^4$ in (18) for $r = 3$, the calculation of the weights $\tilde{\omega}_0^5, \tilde{\omega}_1^5, \tilde{\omega}_2^5, \tilde{\omega}_3^5, \tilde{\omega}_0^6$ and $\tilde{\omega}_1^6$ in (26) for $r = 4$ and so on for greater values of r. Observing (18) and (26), it is easy to reach the conclusion that the number of extra coefficients that we need to calculate for each interpolation point is given by the difference equation $u_{r+1} = 2u_r + 2$ for $r = 3, 4, \cdots$, with $u_3 = 2$. It is true that for a very high r, the computational cost will be affected, but for low values of r that are the ones used in practice, the computational cost is not very much affected, as it can be observed in the numerical experiments.

Algorithm 1 Algorithmic sequence to obtain the nonlinear optimal weights for order 2r

1: **for** $m = r-1, \ldots, 1$ **do**
2: $k = 0$
3: **for** $n = 0, \ldots, m-1$ **do**
4: Solve for the unknown weights $C_n^{2r-(m+1)}, C_{n+1}^{2r-(m+1)}$: $P_n^{2r-m} = C_n^{2r-(m+1)} P_n^{2r-(m+1)} + C_{n+1}^{2r-(m+1)} P_{n+1}^{2r-(m+1)}$
5: **if** $m \neq r-1$ **then**
6: $\tilde{\alpha}_{n+k}^{2r-(m+1)} = \frac{C_{n+k}^{2r-(m+1)}}{(\epsilon + I_{n+k}^{2r-(m+1)})^t}, \quad \tilde{\alpha}_{n+k+1}^{2r-(m+1)} = \frac{C_{n+k+1}^{2r-(m+1)}}{(\epsilon + I_{n+k+1}^{2r-(m+1)})^t}$
7: $\tilde{\omega}_{n+k}^{2r-(m+1)} = \frac{\tilde{\alpha}_{n+k}^{2r-(m+1)}}{\tilde{\alpha}_{n+k}^{2r-(m+1)} + \tilde{\alpha}_{n+k+1}^{2r-(m+1)}}, \quad \tilde{\omega}_{n+k+1}^{2r-(m+1)} = \frac{\tilde{\alpha}_{n+k+1}^{2r-(m+1)}}{\tilde{\alpha}_{n+k}^{2r-(m+1)} + \tilde{\alpha}_{n+k+1}^{2r-(m+1)}}$
8: $\bar{\mathbf{C}}_n^{2r-(m+1)+1} = \tilde{\omega}_{n+k}^{2r-(m+1)} \bar{\mathbf{C}}_n^{2r-(m+1)} + \tilde{\omega}_{n+k+1}^{2r-(m+1)} \bar{\mathbf{C}}_{n+1}^{2r-(m+1)}$
9: $k = k+1$
10: **end if**
11: **end for**
12: **if** $m = r-1$ **then**
13: Define the $r-1$ vectors of r weights:

$$\bar{\mathbf{C}}_0^{r+1} = (C_0^r, C_1^r, 0, 0, \cdots, 0), \quad \bar{\mathbf{C}}_1^{r+1} = (0, C_2^r, C_3^r, 0, \cdots, 0), \quad \cdots, \quad \bar{\mathbf{C}}_{r-2}^{r+1} = \left(0, \cdots, 0, C_{2r-2}^r, C_{2r-1}^r\right). \tag{27}$$

14: **end if**
15: **end for**; **return** $\bar{\mathbf{C}}_0^{2r-1}$

5. Numerical experiments

In this section we present some numerical examples that support the theoretical results presented in previous sections for WENO-6 and WENO-8 algorithms. In order to check the accuracy of both algorithms we consider the piecewise continuous function,

$$f(x) = \begin{cases} -x^9 + x^8 - 4x^7 + x^4 + 5x^2 + 3x, & 0.5 \leq x < 0, \\ -x^9 + x^8 - 8x^7 + x^4 + 5x^2 + 3x + 1, & 0 \leq x < 0.5. \end{cases} \tag{28}$$

Tables 1 and 3 present a grid refinement analysis for the new WENO-6 and the new WENO-8 algorithms. Tables 2 and 4 present the results for the classical WENO-6 and the classical WENO-8 algorithms. In these tables we use 2^i initial points. We present the results in an interval around the discontinuity. We can see how the accuracy is reduced step by step towards the discontinuity in the results of the new algorithms while this is not true for the classical implementation of WENO. Tables 3 and 4 only present the results of order of accuracy for all the stencils affected to the left of the discontinuity. To the right we only present some of the results, as they are symmetric. About the computational time, we perform 500 executions of each subroutine and obtain the mean of the computational time. We can see that the computational time is similar for both algorithms.

	$\cdots x_{2j-6}$		x_{2j-4}		x_{2j-2}		x_{2j}		x_{2j+2}		x_{2j+4}		$x_{2j+6}\cdots$		Comp. t.
i	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	
5	9.737e-09	-	1.584e-08	-	9.052e-07	-	4.976e-01	-	7.931e-07	-	5.781e-08	-	1.868e-08	-	1.340e-03
6	8.000e-11	6.927	1.289e-10	6.941	5.597e-08	4.016	4.994e-01	-0.005	5.508e-08	3.848	4.437e-10	7.026	1.342e-10	7.122	2.056e-03
7	5.511e-13	7.181	9.357e-13	7.107	3.493e-09	4.002	4.999e-01	-0.001	3.486e-09	3.982	3.541e-12	6.969	1.106e-12	6.923	3.601e-03
8	3.587e-15	**7.263**	6.592e-15	**7.149**	2.182e-10	**4.001**	5.000e-01	**0**	2.183e-10	**3.997**	2.842e-14	**6.961**	9.104e-15	**6.924**	6.749e-03

Table 1: Grid refinement analysis for the new WENO-6 algorithm for the function in (28).

	$\cdots x_{2j-6}$		x_{2j-4}		x_{2j-2}		x_{2j}		x_{2j+2}		x_{2j+4}		$x_{2j+6}\cdots$		Comp. t.
i	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	
5	9.735e-09	-	2.000e-07	-	9.047e-07	-	4.983e-01	-	7.936e-07	-	2.126e-07	-	1.871e-08	-	1.281e-03
6	8.000e-11	6.927	1.284e-08	3.961	5.596e-08	4.015	4.996e-01	-0.004	5.509e-08	3.849	1.294e-08	4.039	1.342e-10	7.124	2.753e-03
7	5.511e-13	7.181	8.056e-10	3.995	3.493e-09	4.002	4.999e-01	-0.001	3.486e-09	3.982	8.063e-10	4.004	1.105e-12	6.923	3.723e-03
8	3.587e-15	**7.263**	5.036e-11	**4.000**	2.182e-10	**4.001**	5.000e-01	**0**	2.184e-10	**3.997**	5.038e-11	**4.000**	9.104e-15	**6.924**	7.596e-03

Table 2: Grid refinement analysis for the classical WENO-6 algorithm for the function in (28).

	$\cdots x_{2j-8}$		x_{2j-6}		x_{2j-4}		x_{2j-2}		x_{2j}		x_{2j+2}		$x_{2j+4}\cdots$		Comp. t.
i	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	
5	6.023e-11	-	1.799e-09	-	1.035e-08	-	2.833e-08	-	4.990e-01	-	1.230e-07	-	2.596e-08	-	7.619e-04
6	2.250e-13	8.065	1.244e-11	7.176	7.697e-11	7.071	2.154e-10	7.039	4.997e-01	-0.002	9.707e-10	6.985	2.052e-10	6.983	9.928e-04
7	7.355e-16	**8.257**	9.188e-14	7.081	5.880e-13	7.032	1.692e-12	6.992	4.999e-01	-0.001	7.662e-12	6.985	1.619e-12	6.986	1.324e-03
8	-	-	6.974e-16	**7.042**	4.535e-15	**7.019**	3.260e-14	**5.698**	5.000e-01	**0**	7.927e-14	**6.595**	1.288e-14	**6.974**	1.544e-03

Table 3: Grid refinement analysis for the new WENO-8 algorithm for the function in (28).

	$\cdots x_{2j-6}$		x_{2j-4}		x_{2j-2}		x_{2j}		x_{2j+2}		x_{2j+4}		$x_{2j+6}\cdots$		Comp. t.
i	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	e_i	$\log_2\left(\frac{e_i}{e_{i+1}}\right)$	
5	6.175e-11	-	1.094e-09	-	4.101e-09	-	2.871e-08	-	4.994e-01	-	1.234e-07	-	2.353e-08	-	4.550e-04
6	2.253e-13	8.099	8.091e-12	7.080	3.154e-11	7.023	2.174e-10	7.045	4.999e-01	-0.001	9.727e-10	6.987	1.854e-10	6.988	8.220e-04
7	7.494e-16	**8.232**	6.244e-14	7.018	2.709e-13	6.863	1.768e-12	6.942	5.000e-01	-0.000	7.738e-12	6.974	1.481e-12	6.968	1.074e-03
8	0	-	9.957e-16	**5.971**	1.925e-14	**3.815**	7.510e-14	**4.557**	5.000e-01	**0**	1.219e-13	**5.988**	2.864e-14	**5.692**	1.437e-03

Table 4: Grid refinement analysis for the classical WENO-8 algorithm for the function in (28).

6. Conclusions

In this paper we have presented a generalization to order of accuracy $2r$ of the algorithm introduced in [5]. We have provided an algorithm to construct the nonlinear optimal weights of any WENO-2r scheme for interpolation of data discretized in the point values. We have proposed smoothness indicators of high order calculated from those used by the classical implementation of WENO that are computationally more efficient than those used in [5]. We have discussed about the accuracy and presented numerical experiments that support the claims made for WENO-6 and WENO-8. In future works we plan to extend this generalization to cell averages and conservation laws.

[1] A. Harten and S. Osher. Uniformly high-order accurate nonoscillatory schemes. I. *SIAM J. Numer. Anal.*, 24(2):279–309, 1987.

[2] A. Harten, B. Engquist, S. Osher, and S.R. Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes, III. *J. Comput. Phys.*, 71(2):231 – 303, 1987.

[3] X.-D. Liu, S. Osher, and T. Chan. Weighted essentially non-oscillatory schemes. *J. Comput. Phys.*, 115(1):200 – 212, 1994.

[4] G. Jiang and C.W. Shu. Efficient implementation of weighted ENO schemes. *J. Comput. Phys.*, 126(1):202 – 228, 1996.

[5] S. Amat, J. Ruiz, and C.-W. Shu. On new strategies to control the accuracy of WENO algorithms close to discontinuities. *SIAM J. Numer. Anal.*, 57(3):1205 – 1237, 2019.

[6] F. Aràndiga, A.M. Belda, and P. Mulet. Point-value WENO multiresolution applications to stable image compression. *J. Sci. Comput.*, 43(2):158–182, 2010.