

# Divergence-free smoothing for volumetric PIV data

ChengYue Wang<sup>1</sup> · Qi Gao<sup>1</sup> · HongPing Wang<sup>1</sup> · RunJie Wei<sup>3</sup> ·  
Tian Li<sup>1,2</sup> · JinJun Wang<sup>1</sup>

Received: 22 April 2015 / Revised: 6 December 2015 / Accepted: 8 December 2015 / Published online: 14 January 2016  
© Springer-Verlag Berlin Heidelberg 2015

**Abstract** This paper proposes a divergence-free smoothing (DFS) method for the post-process of volumetric particle image velocimetry (PIV) data, which can smooth out noise and divergence error at the same time. The method is a combination of the penalized least squares regression and the divergence corrective scheme (DCS), employing the generalized cross-validation method to automatically determine the best smoothing parameter. By introducing a weight-changing algorithm similar to the all-in-one method, a robust version of DFS can simultaneously deal with vector validation, replacement of outliers and missing vectors, smoothing, and zero-divergence correction of the velocity field. Direct numerical simulation data of turbulent channel flow (Johns Hopkins Turbulence Databases) added with artificial noise, outliers and missing vectors are used to test the accuracy of DFS. The results show that DFS can smooth the velocity field to divergence-free and performs better than the all-in-one method, DCS and some other available conventional processing methods for post-process of velocity field, especially in dealing with clustered outliers and missing vectors. A block DFS is suggested to process large velocity field to save both time and memory. Tests on tomographic PIV data validate the effectiveness of DFS on improving both flow statistics and flow visualization.

## 1 Introduction

Particle image velocimetry (PIV) is one of the most efficient non-invasive techniques to capture velocity information from complex flows. The technique deduces velocity field by the cross-correlation analysis of particle images recorded by digital cameras (Adrian and Westerweel 2011). Since these images of tracer particles from experiments usually suffer from background noise, issue of out-of-pair particles and model shadows or reflecting light, the velocity fields from PIV measurements are inevitably subject to noise, outliers and missing vectors (Garcia 2011). These defects may not only affect the visualization of the flow, but also have a great influence on derivative quantities of velocity such as vorticity and quantities of vortex identification, as well as flow statistics. Therefore, post-processing on error correction is a necessary procedure for PIV data.

Typically, the post-processing of PIV contains the following three steps: data validation, replacement of spurious or missing vectors, and smoothing (Garcia 2011). In previous research, considerable attention was focused on outlier detection (Westerweel 1994; Nogueira et al. 1997; Song et al. 1999; Liang et al. 2003; Young et al. 2004). Most of those methods, such as the local-median test and cellular neural network (CNN) (Liang et al. 2003), need different values of threshold to regulate the number of detected outliers, which makes the threshold become a subjective setting. Shinnee et al. (2004) proposed an outlier detection method with variable threshold technique, which made the threshold parameter more objective. The normalized median test (NMT) method (Westerweel and Scarano 2005) utilizes a universal approach of threshold determination for a wide range of PIV measurements by normalizing the median residual using a robust estimation of the local variance of

---

✉ Qi Gao  
qigao@buaa.edu.cn

<sup>1</sup> Key Laboratory of Fluid Mechanics, Ministry of Education, Beijing University of Aeronautics and Astronautics, Beijing 100191, China

<sup>2</sup> Shenyang Aircraft Design and Research Institute, Shenyang 110035, China

<sup>3</sup> MicroVec., Inc, Beijing 100083, China

velocity. This method was demonstrated as an effective way of identifying outliers for different PIV experiments. After outliers have been detected, the corresponding velocities need to be replaced by interpolation methods. Kriging interpolation is an effective method in recovering the gappy velocity fields (Gunes et al. 2006; Gunes and Rist 2007). Proper orthogonal decomposition (POD) based methods such as gappy POD was also proposed to reconstruct missing velocities in PIV data, which outperformed Kriging interpolation for sufficient temporal resolution (Gunes et al. 2006). Wang et al. (2015b) proposed a POD-based outlier correction technique to deal with outliers and missing values by using reconstructed field of low-order POD modes as reference field. For noisy experimental data, smoothing is another necessary procedure to reduce noise error from following approaches. The most common smoothing method is to convolute the velocity field with a smoothing kernel. To avoid the additional error caused by the low-pass filtering on the velocity field, the kernel size should be smaller than the effective interrogation window size (Raffel et al. 2007).

Different from the conventional 3-step procedure, Garcia (2011) proposed a fast all-in-one technique to accomplish all the post-processing procedures in just one algorithm. The all-in-one method is based on the penalized least squares (PLS) regression (Wahba 1990), combining with the discrete cosine transform (DCT) and the generalized cross-validation (GCV) (Craven and Wahba 1978). PLS is a common method for data smoothing, which determines the smooth data  $\tilde{\mathbf{y}}$  from raw noisy data  $\mathbf{y}$  by minimizing a criterion function  $F(\tilde{\mathbf{y}})$ .  $F(\tilde{\mathbf{y}})$  represents a balance between the fidelity to the raw data  $\mathbf{y}$ , which is measured by the residual sum of  $\tilde{\mathbf{y}} - \mathbf{y}$ , and a penalty term  $R(\tilde{\mathbf{y}})$  that describes the roughness of  $\tilde{\mathbf{y}}$ , namely (Garcia 2010)

$$F(\tilde{\mathbf{y}}) = \|\tilde{\mathbf{y}} - \mathbf{y}\|^2 + sR(\tilde{\mathbf{y}}). \quad (1)$$

Here,  $\|\cdot\|$  denotes the 2-norm of vectors as a convention in this paper. The scalar parameter  $s$  controls the smoothing degree, which should be determined before applying PLS smoothing. In order to avoid under- or over-smoothing on raw data, Craven and Wahba (1978) suggested to determine  $s$  by minimize the generalized cross-validation score, who also validated the effectiveness of such GCV method. The PLS method could be carried out on a fast way of applying DCT and inverse discrete cosine transformation (IDCT) on the raw data (Buckley 1994), which was named as DCT-based penalized least squares method (DCT-PLS). The all-in-one method improved DCT-PLS to deal with outliers and missing values by employing weights on the raw data and changing the weights iteratively according to the residuals of smoothing results and raw data (Garcia 2010, 2011). The all-in-one method is an automatic adaptive process and does not require subjective adjustment on threshold

parameters or kernel size. Garcia (2011) also demonstrated that the all-in-one method is very well adapted to raw PIV data due to its efficiency, accuracy and rapidness. In this paper, we follow Garcia (2011) to use DCT-PLS to stand for the all-in-one method.

Most available methods for post-process resort to pure mathematical tools and ignore the physical mechanism behind flows, which would generate non-physical errors into the data. For a volumetric PIV data, one physical mechanism is very obvious that the flow should be governed by the Navier–Stokes equations. For incompressible flow, the 3D velocity field satisfies the divergence-free equation,  $\nabla \cdot \mathbf{u} = 0$ , where  $\mathbf{u}$  is the velocity vector. However, volumetric velocity measurements usually suffer a nonzero-divergence error due to experimental errors and uncertainties. The divergence corrective scheme (DCS) was designed by de Silva et al. (2013) to clear up the divergence error by solving an optimization problem, which seeks a divergence-free velocity field  $\mathbf{u}_c$  by making the smallest modification to the raw experimental velocity field  $\mathbf{u}_{\text{exp}}$ , namely

$$\begin{aligned} \min \|\mathbf{u}_c - \mathbf{u}_{\text{exp}}\| \\ \text{s.t. } \nabla \cdot \mathbf{u}_c = 0. \end{aligned} \quad (2)$$

The technique of DCS is verified helpful in reducing the experimental error and improving the flow statistics (de Silva et al. 2013). However, the improvement is limited and susceptible to noise and outliers since only the weakest correction is applied to the original data. In fact, DCS usually leads to a reduction of noise by  $1 - \sqrt{2/3} \approx 18\%$  because the number of constraints employed in the method is about one-third of the total number of variances (Schiavazzi et al. 2014). There are several alternative techniques (Yang et al. 1993; Sadati et al. 2011; Schiavazzi et al. 2014) to reconstruct divergence-free velocity field from noisy velocity data. Unfortunately, those methods lack enough smoothing manner for removing strong noise effectively.

This work proposes a divergence-free smoothing (DFS) method for the post-processing of volumetric PIV data. The method determines a well-smoothed and divergence-free velocity field from the original noisy field by solving an optimization problem. The optimization problem combines the PLS regression and DCS, which uses Eq. (1) as goal function and the divergence-free condition as constraint. To avoid over- or under-smoothing, DFS also employs GCV method to automatically determine the smoothing degree like DCT-PLS. Instead of using DCT in the DCT-PLS method, the raw velocity field is projected onto a set of divergence-free smoothing bases (DFSBS), which are constructed by the divergence-free constraint and roughness criterion. The DFSBS are a set of bases with zero divergence, which are ordered according to their roughness. By introducing a weight-changing iterative algorithm into

DFS, similar to the DCT-PLS method, a robust version of DFS could deal well with outliers and missing vectors.

This paper is arranged in the following fashion. Section 2 provides the general framework of DFS. Section 3 details the assessment of DFS based on synthetic noisy velocity fields from a direct numerical simulation (DNS) data. Section 4 discusses the computational efficiency of DFS and provides suggestion for improvement. Section 5 tests DFS on tomographic PIV measurements followed by conclusions in Sect. 6.

## 2 Divergence-free smoothing method

### 2.1 Theory of DFS

The original velocity field  $\mathbf{u}_{\text{exp}}$  from the experiment is usually neither divergence-free nor smooth because of the contamination of noise and outliers. We are going to determine a new velocity field  $\mathbf{u}_s$  from  $\mathbf{u}_{\text{exp}}$ , which meets the requirement of zero divergence and has been somehow smoothed as well. The experimental velocities are supposed to distribute on a uniform volumetric grid of  $n_x \times n_y \times n_z$  with spacings of  $\Delta x$ ,  $\Delta y$  and  $\Delta z$ , which are common in tomographic PIV (Elsinga et al. 2006).

The divergence-free of velocity field means that  $\mathbf{u}_s$  satisfies the continuity equation, namely  $\nabla \cdot \mathbf{u}_s = 0$  for incompressible flows. By using a second-order finite difference scheme, the equation is approximated as a set of linear equations:

$$\frac{u_s^{i+1,j,k} - u_s^{i-1,j,k}}{2\Delta x} + \frac{v_s^{i,j+1,k} - v_s^{i,j-1,k}}{2\Delta y} + \frac{w_s^{i,j,k+1} - w_s^{i,j,k-1}}{2\Delta z} = 0, \tag{3}$$

where  $i, j, k$  stand for the indices of three coordinates, and  $u_s^{i,j,k}$ ,  $v_s^{i,j,k}$ ,  $w_s^{i,j,k}$  denote the three components of  $\mathbf{u}_s$  at the position of  $(i, j, k)$ . Similarly, a first-order forward or backward difference scheme is employed at the boundary of the domain. For simplicity, the above equations are written in a matrix form,

$$\mathbf{A}\mathbf{U}_s = 0, \tag{4}$$

where  $\mathbf{A}$  is the divergence operator matrix, and  $\mathbf{U}_s$  is a column matrix (vector) reshaped from  $\mathbf{u}_s$ , which contains all three components of the entire velocity field. Thus, after the smoothing operation,  $\mathbf{U}_s$  needs to be restored back into the format of  $\mathbf{u}_s$  with volumetric grid of  $(i, j, k)$ . In this paper, without noticing, both  $\mathbf{U}_s$  and  $\mathbf{U}_{\text{exp}}$  denote such column matrices. Their subscripts of ‘s’ and ‘exp’ indicate the smooth velocity field and the original experimental field, respectively. It should also be noted that  $\mathbf{A}$  is a

matrix with dimensions of  $n$  rows by  $3n$  columns, where  $n = n_x \times n_y \times n_z$  denotes the total number of the grid points.

Evidently, the number of variations is much larger than the equations, which means that numerous solutions exist for these equations. Thus, optimal approach is needed for solving this kind of underdetermined problem. DCS chooses the most conservative modification on velocity field  $\mathbf{U}_{\text{exp}}$  to reach the critical value of its objective function of the optimal method. However, such minimal-change criterion ignores the characteristics of noise degree, which causes a failure when the noise level on the measurement data is high. Fortunately, noise is highly associated with smoothness, which could be used to estimate noise level. Therefore, differently from DCS, the goals of the new technique on  $\mathbf{U}_s$  is not only to achieve the divergence-free, but also to automatically estimate the noise level and possibly reduce the noise in a smoothing manner. Referring to the PLS method, we choose a minimized objective function as:

$$F(\mathbf{U}_s) = (\mathbf{U}_s - \mathbf{U}_{\text{exp}})^T(\mathbf{U}_s - \mathbf{U}_{\text{exp}}) + sR(\mathbf{U}_s). \tag{5}$$

The function  $R(\mathbf{U}_s)$  is a description of the roughness of the velocity field. Large  $R$  value associates with a field containing lots of small-scale flow structures or high-frequent noise, appearing to be not smooth. The function  $F(\mathbf{U}_s)$  represents a balance between the fidelity to the raw data and the roughness of the resulting field. The smoothing parameter  $s$  is a real positive scalar that controls the degree of smoothing. As the smoothing parameter increases, the smoothing degree of  $\mathbf{U}_s$  also increases.

A simple and straightforward approach to express the roughness is by using a second-order divided difference (Weinert 2007). For three-dimensional vector field,  $R$  should contain the full second-order derivatives of all three components, which yields

$$R(u) = \int \left( \frac{\partial^2 u}{\partial x^2} \right)^2 + \left( \frac{\partial^2 u}{\partial y^2} \right)^2 + \left( \frac{\partial^2 u}{\partial z^2} \right)^2 + 2 \left( \frac{\partial^2 u}{\partial x \partial y} \right)^2 + 2 \left( \frac{\partial^2 u}{\partial y \partial z} \right)^2 + 2 \left( \frac{\partial^2 u}{\partial z \partial x} \right)^2 dx dy dz, \tag{6}$$

$$R(v) = \int \left( \frac{\partial^2 v}{\partial x^2} \right)^2 + \left( \frac{\partial^2 v}{\partial y^2} \right)^2 + \left( \frac{\partial^2 v}{\partial z^2} \right)^2 + 2 \left( \frac{\partial^2 v}{\partial x \partial y} \right)^2 + 2 \left( \frac{\partial^2 v}{\partial y \partial z} \right)^2 + 2 \left( \frac{\partial^2 v}{\partial z \partial x} \right)^2 dx dy dz, \tag{7}$$

$$R(w) = \int \left( \frac{\partial^2 w}{\partial x^2} \right)^2 + \left( \frac{\partial^2 w}{\partial y^2} \right)^2 + \left( \frac{\partial^2 w}{\partial z^2} \right)^2 + 2 \left( \frac{\partial^2 w}{\partial x \partial y} \right)^2 + 2 \left( \frac{\partial^2 w}{\partial y \partial z} \right)^2 + 2 \left( \frac{\partial^2 w}{\partial z \partial x} \right)^2 dx dy dz, \tag{8}$$

$$R(\mathbf{u}) = R(u) + R(v) + R(w). \tag{9}$$

Taking  $u$ -component for instance, the second-order derivatives can be approximated by the finite difference scheme as

$$\frac{\partial^2 u}{\partial x^2} |_{ijk} = \frac{u_s^{i+1,j,k} + u_s^{i-1,j,k} - 2u_s^{i,j,k}}{\Delta x^2}, \tag{10}$$

$$\frac{\partial^2 u}{\partial x \partial y} |_{ijk} = \frac{u_s^{i+1,j+1,k} + u_s^{i,j,k} - u_s^{i+1,j,k} - u_s^{i,j+1,k}}{\Delta x \Delta y}. \tag{11}$$

At the boundaries of measurement domain, the velocity of outer layer is extrapolated by the neighboring inner grids like  $u^{0,j,k} = u^{1,j,k}$  and  $u^{n_x+1,j,k} = u^{n_x,j,k}$ . Thus, Eqs. (10) and (11) reduce to

$$\frac{\partial^2 u}{\partial x^2} |_{n_x j k} = \frac{u_s^{n_x-1,j,k} - u_s^{n_x,j,k}}{\Delta x^2}, \tag{12}$$

$$\frac{\partial^2 u}{\partial x \partial y} |_{n_x j k} = 0. \tag{13}$$

The integral operations in Eqs. (6)–(9) are substituted by calculating the sum of these differential terms from different spatial points. The resulting  $R(\mathbf{U}_s)$  has a quadratic form as

$$R(\mathbf{U}_s) = \mathbf{U}_s^T \mathbf{M} \mathbf{U}_s, \tag{14}$$

where  $\mathbf{M}$  is a constant and symmetrical matrix associated with the discrete coefficients of  $R(\mathbf{U}_s)$  and only depended on the spatial grids. The specific forms of the divergence operators  $\mathbf{A}$  and  $\mathbf{M}$  adopted in this paper are detailed in the ‘‘Appendix 1’’.

Therefore, the eventual description of the problem becomes the following optimization problem with linear constraint:

$$\begin{aligned} \min (\mathbf{U}_s - \mathbf{U}_{\text{exp}})^T (\mathbf{U}_s - \mathbf{U}_{\text{exp}}) + s \mathbf{U}_s^T \mathbf{M} \mathbf{U}_s, \\ \text{s.t. } \mathbf{A} \mathbf{U}_s = 0. \end{aligned} \tag{15}$$

where ‘min’ means the minimum of the objective function, and ‘s.t.’ is the abbreviation of ‘subject to’ indicating that the following equations would be the constraints. The optimization problem indicates that the smoothed velocity field  $\mathbf{U}_s$  that we are seeking should achieve the minimum of the objective function and be divergence-free at the same time. Such optimization problem could be solved directly by using well-developed optimization methods once the parameter  $s$  is given. Unfortunately, the best value of  $s$  is unknown here. Therefore, Eq. (15) actually has two unknowns, and we need one more objective function to make the optimal problem closed. As we will see in Sect. 2.3, another optimization problem is proposed to find the

optimal  $s$ , which need an explicit expression between  $\mathbf{U}_s$  and  $\mathbf{U}_{\text{exp}}$  for fast computation. At this stage, we can just assume that  $s$  is known that we could try to find the explicit solution of  $\mathbf{U}_s$  from Eq. (15).

According to the knowledge of linear algebra, there are a set of orthonormal bases existing in the solution space of Eq. (4), which could express all the solutions by linear combinations. Assuming that one set of such orthonormal bases  $\{\Phi_i\} (i = 1, 2, \dots, d)$  have been acquired,  $\mathbf{U}_s$  can be expressed in the following form

$$\mathbf{U}_s = \sum \Phi_i a_i = \Phi \mathbf{a}, \tag{16}$$

where  $\Phi = [\Phi_1, \Phi_2, \dots, \Phi_d]$ .  $d$  is the dimension of the solution space, which depends on the rank of  $\mathbf{A}$ . By replacing the  $\mathbf{U}_s$  in the Eq. (5), the problem becomes unconstrained optimization as

$$\min F(\mathbf{a}) = \min (\Phi \mathbf{a} - \mathbf{U}_{\text{exp}})^T (\Phi \mathbf{a} - \mathbf{U}_{\text{exp}}) + s \mathbf{a}^T \Phi^T \mathbf{M} \Phi \mathbf{a}. \tag{17}$$

To find out the minimal value of Eq. (17), one order derivative of  $\mathbf{a}$  is performed, which generates a set of linear algebra equations:

$$(\mathbf{I} + s \Phi^T \mathbf{M} \Phi) \mathbf{a} = \Phi^T \mathbf{U}_{\text{exp}}. \tag{18}$$

In order to avoid the complex inverse of matrix, we assume that  $\Phi^T \mathbf{M} \Phi$  is diagonal, which means that

$$\Phi^T \mathbf{M} \Phi = \Lambda, \tag{19}$$

where  $\Lambda$  is the related diagonal matrix. As we will show in the Sect. 2.2, such bases are not difficult to obtain, which can be named as divergence-free smoothing bases (DFSBS). Once DFSBS are obtained, the expressions for  $\mathbf{a}$  and  $\mathbf{U}_s$  could be rewritten as

$$\mathbf{a} = (\mathbf{I} + s \Lambda)^{-1} \Phi^T \mathbf{U}_{\text{exp}}, \tag{20}$$

$$\mathbf{U}_s = \Phi (\mathbf{I} + s \Lambda)^{-1} \Phi^T \mathbf{U}_{\text{exp}}. \tag{21}$$

Therefore, it suggests that  $\mathbf{U}_s$  can be obtained by three procedures. Firstly, project the original velocity field  $\mathbf{U}_{\text{exp}}$  onto the DFSBS to get the corresponding projection coefficients.  $\mathbf{U}_{\text{exp}}$  could be divided into two parts, divergence-free part and remaining part. Such projecting operation abandons the remaining part implicitly because DFSBS are specifically chosen in the divergence-free space. Secondly, weight the coefficients by diagonal matrix  $(\mathbf{I} + s \Lambda)^{-1}$ . The smoothing mechanism is introduced in this procedure by imposing larger weights on the smoother ones of DFSBS. Lastly, reconstruct a new velocity field with DFSBS and the weighted coefficients. The reconstruction process [Eq. (16)] is a reverse projecting operation. Such bases-projecting operation employed here could also be understood as coordinate transformation in a linear space, which is familiar in other smoothing method (Wang et al. 2015a).

It is also worthy to note that the Eq. (21) is similar to the DCT-PLS. The differences are that DFS replaces the DCT method with the projection of the new bases  $\Phi$ , and DCT-PLS deals with velocity components one by one independently, while DFS processes all three components at the same time under the divergence-free condition. DCS could be considered as a special case of DFS when the smoothing parameter  $s$  equals zero, which means that the smoothness of velocity  $U_s$  is ignored, and only minimal modification on the raw velocity field is considered in DCS. Thus, both DFS and DCS can be implemented by a quick matrix multiplication in Eq. (21) as we will further discuss in Sect. 4 regarding computational cost. The comparison of DCS, DCT-PLS and DFS is well summarized in Table 1.

### 2.2 Constructing of divergence-free smoothing bases

Starting from this section, we take  $\Phi$  as DFSB. According to Sect. 2.1, the base matrix  $\Phi$  should satisfy the following two conditions,

- (i)  $\Phi$  is a set of orthonormal bases from the solution space of the Eq. (4).
- (ii)  $\Phi$  could diagonalize  $M$  as Eq. (19).

Assume that  $\Psi$  is a set of orthonormal bases from the solution space of the Eq. (4). In fact, it is easy to find them from the results of the singular value decomposition of  $A$ .  $\Psi$  is usually not the DFSB which we are seeking, because it does not satisfy Eq. (19). However,  $\Phi$  could be easily acquired from  $\Psi$ . In fact, the following expression holds since both of them are sets of bases in the solution space of Eq. (4),

$$\Phi = \Psi Q, \tag{22}$$

where  $Q$  is a transforming matrix of the two sets of bases. Replacing  $\Phi$  in Eq. (19) with Eq. (22), we get

$$Q^T \Psi^T M \Psi Q = \Lambda. \tag{23}$$

Obviously,  $Q$  is the eigenvector matrix for  $\Psi^T M \Psi$ . Considering  $\Psi^T M \Psi$  is a real symmetric matrix as well as  $M$ , it yields  $Q^T Q = I$ . Therefore,  $\Phi$  is also a orthonormal matrix, because

$$\Phi^T \Phi = (\Psi Q)^T (\Psi Q) = (Q)^T (\Psi)^T \Psi Q = I. \tag{24}$$

Consequently,  $\Phi$  is the DFSB which we are seeking for.

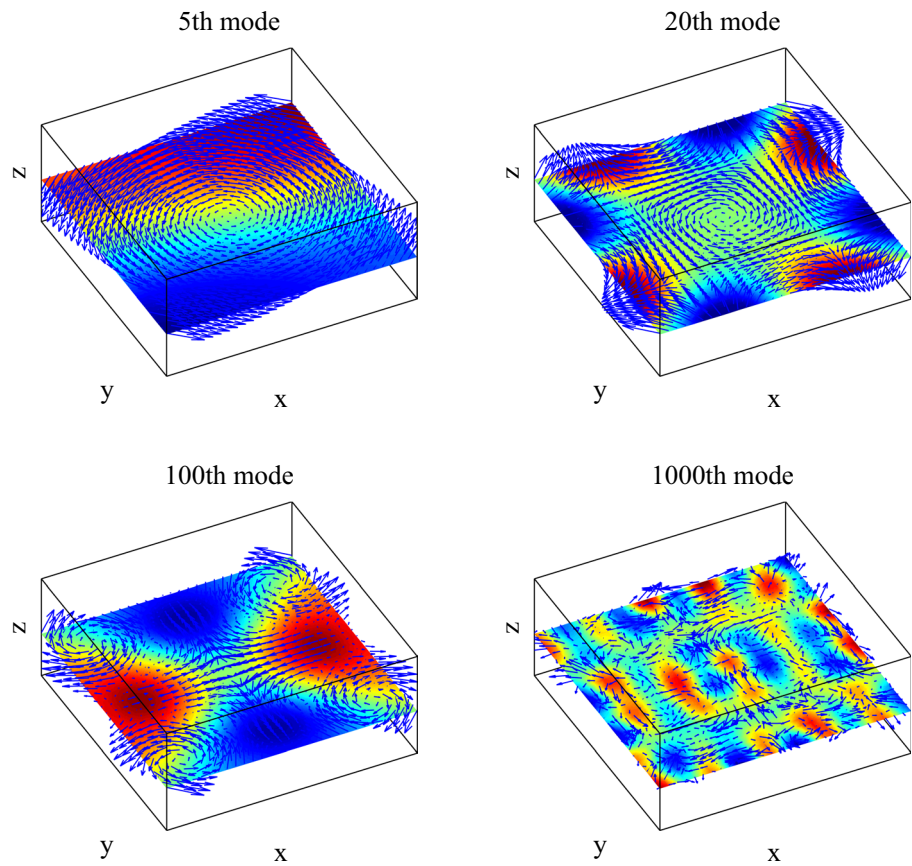
To sum up,  $\Phi$  could be obtained by the following steps:

- (i) Compute one specific set of orthonormal bases  $\Psi$  from the solution space of Eq. (4) by the singular value decomposition of  $A$ .

**Table 1** Comparison of DCS, DCT-PLS and DFS

	DCS	DCT-PLS	DFS
Processing object	The whole velocity $U_{exp}$ components	One component (for example $V_{exp}$ ) at a time	The whole velocity $U_{exp}$
Mathematical description	Find $U_c$ by solving $\min(\ U_c - U_{exp}\ )$ s.t. $AU_c = 0$	Find $V_s$ by the minimum of $F(V_s) = \ V_s - V_{exp}\ ^2 + sR(V_s)$ where $R(V_s)$ is defined by Eq. (7)	Find $U_s$ by solving $\min \ U_s - U_{exp}\ ^2 + sR(U_s)$ s.t. $AU_s = 0$
Formula for practical computation	Depend on concrete optimization method	$V_s = IDCCT3(\Gamma \circ DCT3(V_{exp}))$ with $\Gamma_{ijk} = (1 + s(6 - 2\cos(\frac{(j-1)\pi}{nk} - 2\cos(\frac{(j-1)\pi}{ny} - 2\cos(\frac{(k-1)\pi}{nz}))^2)^{-1})^{-1}$	$U_s = \Phi(I + s\Lambda)^{-1} \Phi^T U_{exp}$ where $\Lambda$ is determined by Eq. (19)
Result	Divergence-free	Smoothed	Smoothed and divergence-free

**Fig. 1** A collection of four DFSB modes with  $30 \times 30 \times 10$  uniform spaced grid. The modes are shown by section velocities. The vectors represent the in-plane velocity and the *color* denotes the velocity component vertical to the plane



- (ii) Perform eigenvalue decomposition on the matrix  $\Psi^T \mathbf{M} \Psi$  to get the eigenvector matrix  $\mathbf{Q}$  and the eigenvalue matrix  $\mathbf{\Lambda}$ .
- (iii) Obtain  $\Phi$  by Eq. (22).

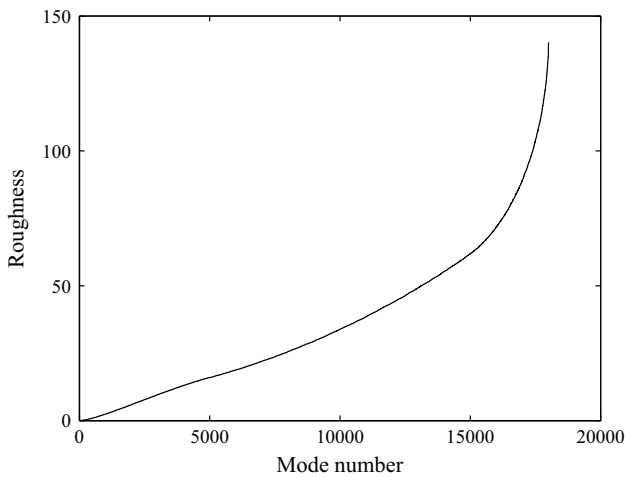
Since the important role of DFSBs in DFS method, further discussion on DFSBs is provided here. First, in this work, the ‘null’ function in MATLAB is employed to calculate a specific  $\Psi$  with  $\Psi = \text{null}(\mathbf{A})$ , and the ‘eig’ function in MATLAB is employed to solve this eigenvalue problem to get  $\mathbf{Q}$  and  $\mathbf{\Lambda}$ . Second, matrix  $\Phi$  is determined only by the matrix  $\mathbf{M}$  and  $\mathbf{A}$ . Therefore, it is independent to the experiment data, which means that DFS does not require different DFSBs for each instantaneous velocity field, and only needs one calculation of DFSBs for the entire data set with the same grid structure.

Denote  $\mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_d)$ . Equation (19) indicates that  $\lambda_i$  describes the roughness of  $\Phi_i$  because of  $R(\Phi_i) = \Phi_i^T \mathbf{M} \Phi_i = \lambda_i$ . Assume that  $\Phi_i$  is ordered as  $\lambda_1 < \lambda_2 < \dots < \lambda_d$ , which means that DFSBs are sorted in order of their roughness. Thus, it could be proved that  $\Phi_i$  is a solution of the following optimization problem:

$$\begin{aligned} & \min R(\Phi_i), \\ & \text{s.t. } R(\Phi_i) < R(\Phi_j), (i < j), \\ & \Phi_i^T \Phi_j = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases}, \\ & \mathbf{A} \Phi_i = 0. \end{aligned} \tag{25}$$

It suggests that the DFSBs could be acquired from the solution space of divergence equations by minimizing the roughness. In other words, the DFSBs are the most smooth orthonormal bases satisfying the divergence-free condition.

DFSBs could be reshaped into the velocity-field form according to the correspondence between the indices of base vectors and their spatial positions  $(i, j, k)$  as stated in Sect. 2.1. Such field-form bases are used to be called as modes. Four example modes have been displayed in Fig. 1, in which only the velocity vectors at the center cross-section of the fields along  $z$ -direction are presented. The data are from the numerical tests that will be further assessed in Sect. 3. From Fig. 1, it is clearly shown that each mode has a good centrosymmetric/anti-centrosymmetric manner since the modes are only dependent on the structures of grids and the difference scheme of Eqs. (4) and (9). The



**Fig. 2** Roughness corresponding to different bases (modes). The roughness is defined by Eq. (14)

roughness values of DFSBs are plotted in Fig. 2, which implies that the modes are arranged in order of increasing roughness. The modes with small roughness are relatively smooth and contain large-scale structures, while the ones with larger roughness are not smooth, containing lots of small-scale structures.

DFSBs is similar to POD bases, because both of them are sets of bases corresponding to different scales of flow structures. However, DFSBs are obtained from the solution space of continuity equation under the criterion of roughness, while POD bases are usually decomposed from the velocity fields according to the kinematic energy. If the constraint of continuity equation is removed, DFSBs reduce to the bases of the discrete cosine transformation, and only the roughness criterion is applied (Buckley 1994; Garcia 2010). The POD (Murray and Ukeiley 2007; Raben et al. 2012; Wang et al. 2015b), DCT and DFS methods obtain mode bases in different ways, but for the same smoothing purpose.

### 2.3 Automatic smoothing with DFS

Before employing DFS to the post-process of PIV data, the smoothing parameter  $s$  in Eq. (21) must be determined first. It is a challenge to estimate an optimal  $s$  for different noise situations. Failure on estimating  $s$  would lead to an over-smoothing or under-smoothing of velocity field. To avoid any subjectivity on the choice of smoothing level and optimal estimation of  $s$ , smoothing parameter  $s$  is determined by minimizing the so-called GCV score, which has been validated in theory by Craven and Wahba (1978). In this work, the GCV is defined as

$$GCV(s) = \frac{(\mathbf{U}_s - \mathbf{U}_{\text{exp}})^T (\mathbf{U}_s - \mathbf{U}_{\text{exp}}) / 3n}{(1 - \text{tr}((\mathbf{I} + s\mathbf{\Lambda})^{-1}) / 3n)^2}, \quad (26)$$

where  $\text{tr}(\cdot)$  is the matrix trace, and  $\mathbf{U}_s$  is determined by Eq. (21). This is an optimization problem with a single variable, which could be easily solved by available optimization methods. In current work, the function of ‘fminbnd’ in the MATLAB is used to find the minimum value of GCV function. A numerical test for the GCV method is provided in ‘‘Appendix 2’’.

### 2.4 Dealing with outliers and missing vectors: robust divergence-free smoothing (RDFS)

In addition to noise, the experimental data also suffers from outliers and missing vectors. Similar to the PLS regression, DFS is sensitive to significant errors. Furthermore, DFS is desired to be able to deal with missing values. Thus, a robust DFS algorithm is required. Considering the similarity of our technique to the all-in-one method, similar iteratively reweighted algorithm (Garcia 2010, 2011) is employed to improve the robustness of DFS and pad the missing vectors.

The robust algorithm reduces the influence of outliers by using smaller weights at the vectors with larger residuals. Therefore, an iterative algorithm is needed to dynamically update the weighs from the residuals that are calculated from the original velocities and the corresponding smoothed vectors. For each iterative step,  $\mathbf{U}_s$  is updated repeatedly until convergence according to the following weighted smoothing formula:

$$\mathbf{U}_s^{(r)} = \Phi(\mathbf{I} + s\mathbf{\Lambda})^{-1} \Phi^T (\mathbf{W}_s \circ (\mathbf{U}_{\text{exp}} - \mathbf{U}_s^{(r-1)}) + \mathbf{U}_s^{(r-1)}), \quad (27)$$

where the symbol  $\circ$  denotes the element-wise product between two vectors and the superscript  $(r)$  and  $(r - 1)$  denote results of  $r$ -th or  $(r - 1)$ -th updates. At the first iterative step,  $\mathbf{W}_s$  is initialized by all ones, while at the following iterative steps,  $\mathbf{W}_s$  is determined by a specified weighting function (Heiberger and Becker 1992), which varies from 0 to 1 and depends upon the residuals of the original field and the smoothed field from the latest iteration. In this work, the following bi-square weighting function is chosen,

$$W_s^i = \begin{cases} (1 - (u^i/c)^2)^2 & \text{if } |u^i/c| \leq 1 \\ 0 & \text{if } |u^i/c| > 1 \end{cases}, \quad (28)$$

where  $W_s^i$  is the  $i$ -th component of weight vector  $\mathbf{W}_s$  and  $u^i$  is the  $i$ -th component of the so-called studentized residual  $\mathbf{u}$ . The parameter  $c$  is a tuning constant chosen to adjust the asymptotic efficiency of the weighting function for data from a known distribution, usually the normal distribution (Heiberger and Becker 1992; Coleman et al. 1980). Here,  $c = 4.685$ , corresponding to a 95 % efficiency to process data of normal distribution (Holland and Welsch 1977). The studentized residual  $u^i$  is determined by

$$u^i = \frac{U_s^i - U_{\text{exp}}^i}{1.4826\text{MAD}(1 - h)^{1/2}}, \quad (29)$$

where  $U_s^i$  and  $U_{exp}^i$  are the  $i$ -th components of  $\mathbf{U}_s$  and  $\mathbf{U}_{exp}$ , respectively. MAD denotes the median absolute deviation, calculated by

$$MAD = \text{median}(|U_s^i - U_{exp}^i - \text{median}(U_s^i - U_{exp}^i)|), \quad (30)$$

$1.4826MAD$  is a robust estimation of the standard deviation of the residuals (Garcia 2010).  $h$  is the average value of the hat matrix diagonals. In this case, the hat matrix is  $\Phi(\mathbf{I} + s\Lambda)^{-1}\Phi^T$  and  $h = \text{tr}((\mathbf{I} + s\Lambda)^{-1})/3n$ .

A second weight vector  $\mathbf{W}_m$  is used to deal with the missing values. The weight vector is made up of zeros and ones, corresponding to the positions of missing values and normal values, respectively. Therefore, the final weight vector  $\mathbf{W} = \mathbf{W}_s \circ \mathbf{W}_m$  and the updating formula of  $\mathbf{U}_s$  eventually becomes

$$\mathbf{U}_s^{(r)} = \Phi(\mathbf{I} + s\Lambda)^{-1}\Phi^T(\mathbf{W} \circ (\mathbf{U}_{exp} - \mathbf{U}_s^{(r-1)}) + \mathbf{U}_s^{(r-1)}). \quad (31)$$

It is worthy to note that the smoothing parameter  $s$  in Eqs. (27) or (31) is also updated before calculating  $\mathbf{U}_s^{(r)}$  by minimizing the following weighted GCV function (Garcia 2010)

$$GCV(s) = \frac{\|\mathbf{W}_s^{\frac{1}{2}} \circ (\mathbf{U}_s^{(r)} - \mathbf{U}_s^{(r-1)})\|^2/3(n - n_{miss})}{(1 - \text{tr}((\mathbf{I} + s\Lambda)^{-1})/3n)^2}, \quad (32)$$

where  $n_{miss}$  is the number of missing vectors and  $\mathbf{U}_s^{(r)}$  is substituted by Eq. (31) before solving the GCV minimum.

The concrete procedures of dealing with outliers and missing data are detailed as follows:

(i) Initialize  $\mathbf{W} = \mathbf{W}_m, k = 1$ ,

$$\mathbf{U}_s = \begin{cases} 0 & \text{for missing values} \\ \mathbf{U}_{exp} & \text{for normal values} \end{cases}; \quad (33)$$

(ii) Update  $\mathbf{U}_s$  according to the weight  $\mathbf{W}$ ;

- (a)  $\mathbf{U}_s^{(0)} = \mathbf{U}_s, r = 1$ ;
- (b) Find the smoothing parameter  $s$  that minimizes the GCV score [Eq. (32)];
- (c) Update  $\mathbf{U}_s^{(r)}$  by Eq. (31);
- (d) if  $\|\mathbf{U}_s^{(r)} - \mathbf{U}_s^{(r-1)}\|/\|\mathbf{U}_s^{(r)}\| < Cr$ ,  $\mathbf{U}_s = \mathbf{U}_s^{(r)}$  and go to step (iii), else  $r = r + 1$  and go to step (b);

(iii) Calculate weight matrix  $\mathbf{W}_s$  according to Eq. (28), update  $\mathbf{W}$  by  $\mathbf{W} = \mathbf{W}_s \circ \mathbf{W}_m$ ;

(iv)  $k = k + 1$ ;

(v) If  $k \leq IterNum$ , go to step (ii); else, go to step (vi);

(vi) Return  $\mathbf{U}_s$ .

where  $k$  and  $r$  are iterative indices of the main loop and the sub-loop of step (ii), respectively. In step (ii),  $Cr$  is the exit condition for the convergence of  $\mathbf{U}_s$  in the sub-loop, which takes its default value of  $10^{-4}$  in this paper.  $IterNum$  is the iteration number of weight-

changing. Usually, 3–5 weight-changing iterations are enough for the processing of PIV data with outliers. In this paper, we perform three iterations as default. To process the field with only missing values and noise (no outliers), only one iteration is required ( $IterNum = 1$ ), which means that the weights are not changed. The initialization of the missing values is normally non-sensitive to the final result because of the robust iteration algorithm and the zero weight imposed at the missing region. Therefore, the initial values for the missing vectors are all zeros as default in this work. The flowchart of the new approach is provided in Fig. 3. Details of the weight function and the iterative algorithm were discussed by Garcia (2010).

### 3 Numerical assessment

In order to validate and quantify the accuracy of DFS, DNS dataset of turbulent channel flows (Li et al. 2008; Perlman

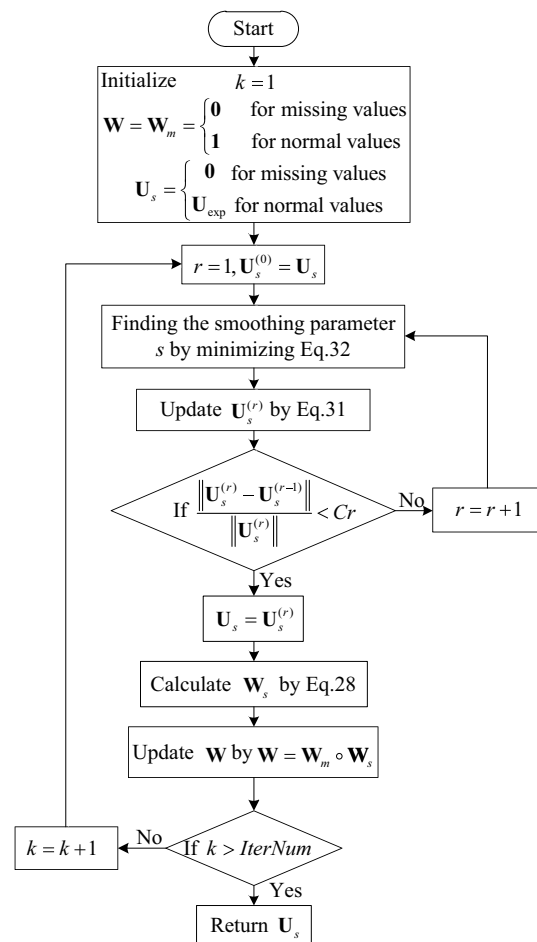


Fig. 3 Flowchart of robust DFS

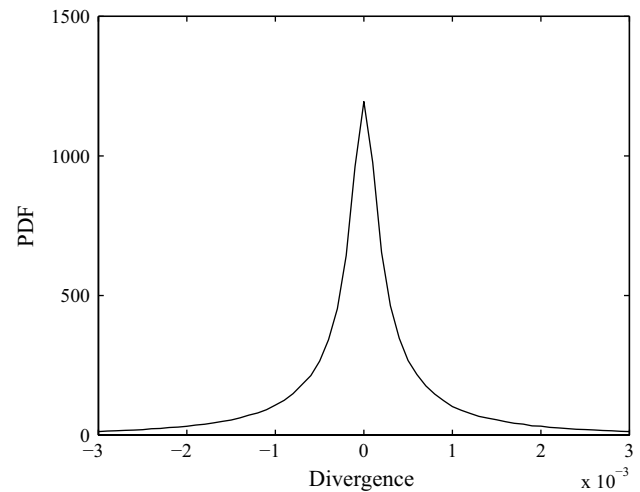


**Table 2** Summary of parameters for test subzones

Streamwise grid spacing $\Delta x^+$	6.13
Spanwise grid spacing $\Delta y^+$	6.13
Wall-normal grid spacing $\Delta z^+$	6.13
Wall-normal position of the subzone	438.5–555.1
Number of Vectors per subzone	$30 \times 30 \times 10$

et al. 2007; Graham et al. 2013) with synthetic errors is used for testing. The DNS velocity field distributes in a zone of  $8\pi \times 3\pi H \times 2H$ , where  $H$  is the half-channel height. This computation zone contains  $2048 \times 1536 \times 512$  spatial grid points in the streamwise, spanwise, and wall-normal directions, respectively. In our assessment, 100 independent small subzones of the DNS field are considered to obtain the averaged quantitative results. Detailed information is summarized in Table 2. A linear interpolation is employed in both streamwise and wall-normal direction to obtain a velocity field on a uniform grid along all three directions, making the DNS data more close to a practical PIV configuration, which is similar to what de Silva et al. (2013) did. In DNS simulation, the divergence-free constraint is enforced in the Fourier-space. Therefore, when evaluating the divergence by finite difference scheme, non-negligible error is obtained (Graham et al. 2013) as shown in Fig. 4. The maximum divergence error of the original DNS data is in the order of  $10^{-3}$ , which is much smaller than other quantities based on velocity gradient such as the vorticity. To make the DNS data more standard as a divergence-free velocity field under finite difference scheme to evaluate different algorithms, DCS is first performed to remove the remaining divergence errors in the original DNS data. In fact, the correction just causes a modification less than 1 % on the velocity fluctuation without changing fluid structures. This procedure can also correct the divergence error caused by the grid interpolation for getting a uniform grid. In the following discussion, the DNS data refer to the DNS data with divergence-free correction, and all the following tests are based on it.

The main sources of PIV error including noise, outliers and missing values are tested in Sect. 3.1–3.3, respectively. The PIV filtering effect (Schrijer and Scarano 2008), as a systematical error source, usually leads to two results: frequency truncation and correlated errors. In present work, the filtering effect is partially simulated by adding correlated noise to DNS field in Sect. 3.1. The frequency truncation, which could be simulated by moving average on the DNS data (de Kat and van Oudheusden 2012), is not involved in current work since it is impossible for DFS or other processing techniques to inversely restore the frequency-truncated velocity field to the original status.

**Fig. 4** PDF of divergence in the original DNS

### 3.1 Reducing noises

Reducing the random noise and correcting the velocity field to zero divergence are two basic tasks of DFS method. In this section, the accuracy and effectiveness of DFS in dealing with noise and divergence error are validated. Gaussian-distributed random noise with constant variance is added to the three components of the velocity field. The standard variance of noise normalized by the root mean square (RMS) of the original velocities is denoted as the noise level (NL). Similar to de Silva et al. (2013)'s work, two types of random white noise are added to the DNS velocity field to simulate the biased and unbiased noise in experimental measurements.

Case 1

$$\begin{aligned} U_n &= U_{\text{DNS}} + \Gamma \times \text{NL}, V_n = V_{\text{DNS}} + \Gamma \times \text{NL}, \\ W_n &= W_{\text{DNS}} + \Gamma \times \text{NL}; \end{aligned} \quad (34)$$

Case 2

$$\begin{aligned} U_n &= U_{\text{DNS}} + \Gamma \times \text{NL}, V_n = V_{\text{DNS}} + \Gamma \times \text{NL}, \\ W_n &= W_{\text{DNS}} + 2\Gamma \times \text{NL}; \end{aligned} \quad (35)$$

where  $U_{\text{DNS}}, V_{\text{DNS}}$  and  $W_{\text{DNS}}$  correspond to the DNS velocity fields, and  $U_n, V_n$  and  $W_n$  are the 'noisy' velocity fields.  $\Gamma$  denotes the random noise of Gaussian distribution in the velocity field, with zero-mean and a variance equal to the velocity magnitude. NL stands for the noise level that changes from 5 to 25 %, leading to six different noise levels in our tests.

In the second case, the noise level added to  $W_{\text{DNS}}$  is twice to the NL corresponding to the other two velocity components. The case is designed for simulating the

situation that in three-component PIV measurements, the velocity component along the direction of the laser sheet thickness is typically the least reliable one (Adrian and Westerweel 2011). The methods of DCS, DCT-PLS and DFS are employed to smooth or correct the velocity field in this work for comparison. DCT-PLS is carried out by the ‘smoothn’ function in Garcia (2011). DCS is implemented by DFS using a special smoothing parameter  $s = 0$ . Figure 5 displays their performances on reducing noise and removing divergence errors for a sample velocity field with noise of case 1 (NL = 5%). It shows that both DCT-PLS and DFS could reduce most of the noise resulting in a well-smoothed velocity field. It seems that they achieve a similar capability of reducing noise. However, DCT-PLS does not make the processed velocity field divergence-free. DCS just slightly reduces noise because the method is designed to correct the divergence error to zero in a minimal-change manner without consideration of noise level. The effect of divergence-correcting is also showed in the last column of Fig. 5, where the probability density functions (PDFs) of the corresponding divergence of the velocity fields are displayed. It shows that the DCS and DFS methods can eliminate the divergence error, while DCT-PLS can only reduce the divergence error to a relatively low level, but not low enough to close to zero. The comparison clearly reveals the capability of DFS on removing the noise and the divergence error. Another way to obtain a smoothed and divergence-free velocity field is to apply both the DCT-PLS and DCS treatments, which means that DCT-PLS is performed for smoothing followed by DCS for removing the divergence error. The combined method DCT-PLS+DCS leads to a very similar result with the result of DFS in Fig. 5. The performances of DCT-PLS+DCS and DFS will be quantitatively compared in the following tests.

To quantify the accuracy of three methods, we define a relative error as

$$\text{Error} = \frac{\|\mathbf{U}_s - \mathbf{U}_{\text{DNS}}\|}{\|\mathbf{U}_{\text{DNS}}\|} \times 100\%, \quad (36)$$

where  $\mathbf{U}_s$  denotes the DFS smoothing result and  $\mathbf{U}'_{\text{DNS}}$  is the fluctuation obtained by subtracting the freestream velocity from  $\mathbf{U}_{\text{DNS}}$ . Besides the above methods, a conventional Gaussian smoothing with filtering size of  $3 \times 3 \times 3$  and  $\sigma = 0.65$  is also applied for comparison. The errors from Gaussian smoothing, DCT-PLS, DCT-PLS+DCS and DFS under different noise levels are provided in Fig. 6. The figure shows that the errors corresponding to DCT-PLS, DCT-PLS+DCS and DFS methods increase more slowly with the noise level comparing to the Gaussian smoothing, which mainly benefits from the GCV method that is adaptive to different noise levels. The combined DCT-PLS+DCS method perform better than DCT-PLS, which is attributed to additional DCS correction. For the noise of

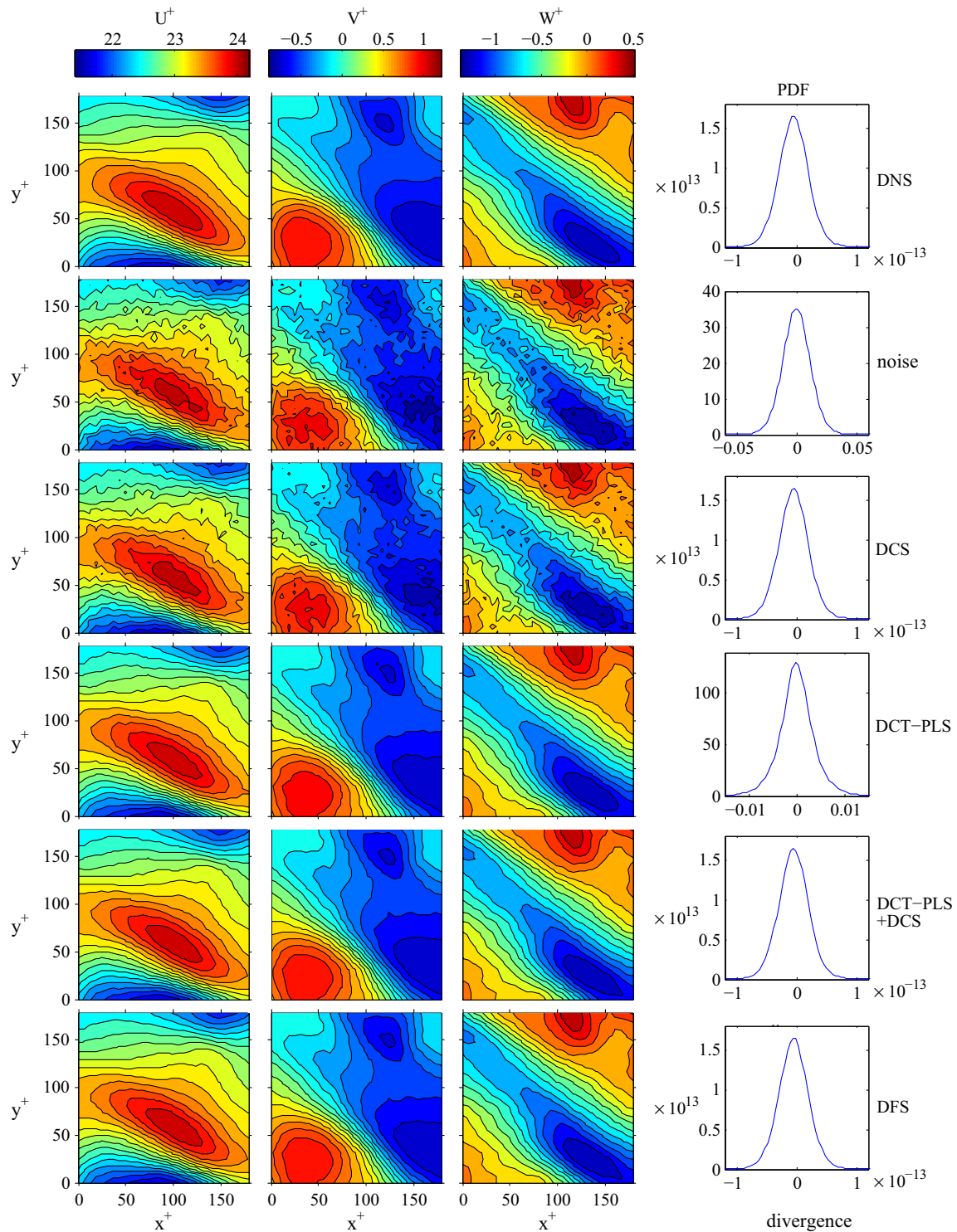
case 1, DFS achieves nearly equal noise-reducing effect with the combined DCT-PLS+DCS. However, It does not mean that DFS is simply equal to the combination of DCT-PLS and DCS. The outstanding performance and effectiveness of DFS will be shown in the following tests. For case 2, DFS brings slightly larger smoothing errors than the combined method. The reason is that DFS-PLS is performed on each velocity components separately. Therefore, it has three independent optimal smoothing parameters to adapt to different noise levels. DFS smooth the three velocity components simultaneously, which means that only one optimal smoothing parameter is employed for the full velocity components with divergence-free constraint and errors may be redistributed among different velocity components. This manner is more robust and does not make DFS lose effectiveness in reducing biased noise because it still outperforms Gaussian smoothing and DCT-PLS, and works very close to DCT-PLS+DCS in Fig. 6.

Besides the white noise in the tests of case 1 and 2, correlated noise is another important type of noise specifically in PIV data, when a large overlap of interrogation window is applied in correlation procedure (Foucaut et al. 2004). Therefore, an additional test is necessary to estimate DCT-PLS and DFS’s performances on reducing correlated noise. According to Foucaut et al. (2004), the power spectra of PIV noise along the  $i$ -th direction could be expressed as

$$E_{ii} = E_{\text{noise}} \left( \frac{\sin(kX/2)}{kX/2} \right)^2 \quad (37)$$

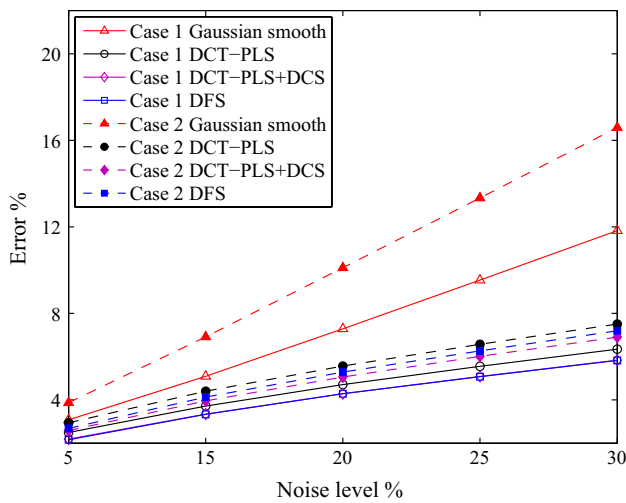
$E_{\text{noise}}$  is a parameter employed to fit noise level, and  $X$  is the window size.  $k$  is the wave number along the  $i$ -th direction. The correlated noise with such power spectra can be approximately simulated by applying an average smoothing on a white noise, which has been tested in current work (not shown) and indicated by Foucaut et al. (2004) as well. Thus, the correlated noise is generated by filtering Gaussian white noise with a  $3 \times 3 \times 3$  averaging window, which corresponds to an interrogation window of  $X^+ = 18.4$  and an overlap of  $2/3$ . Note that the standard variance of the filtered noise is reduced to  $1/\sqrt{27}$  of original white noise.

The results of this test are listed in Table 3. Firstly, all these methods lead to larger errors compared to the tests on the white noise, which suggests that correlated noise is more difficult to be reduced. Secondly, DCT-PLS makes very limited improvement on noise reduction due to the failure of its GCV method, while DCS, Gaussian smoothing and DFS achieve a certain level of noise reduction. In fact, for correlated noise, the GCV method employed in DCT-PLS (DCT-PLS-GCV) always determines a very small smoothing parameter ( $<10^{-3}$ ) resulting in almost no smoothing on the velocity field. The cause of failure on DCT-PLS-GCV is that it determines the smoothing



**Fig. 5** A slice from test DNS data at the position of  $z^+ = 524.4$ . The six rows, from *top to bottom*, displays the original DNS data, noisy velocity field and the velocity processed by DCS, DCT-PLS, DCT-PLS+DCS and DFS, respectively. The *first three columns* correspond

to the three velocity components  $U^+$ ,  $V^+$ , and  $W^+$ , respectively. The *last column* corresponds to the probability density function (PDF) of the divergence errors



**Fig. 6** The smoothing error for Gaussian smoothing, DCT-PLS, DCT-PLS+DCS and DFS in dealing with case 1 and case 2 noise. The red, black, purple and blue lines correspond to the errors of Gaussian smoothing, DCT-PLS, DCT-PLS+DCS and DFS, respectively. The solid and dashed lines indicate the case 1 and case 2

parameter according to the roughness of the velocity field, while correlated noise already smoothed due to the overlap manner. By employing DFSBs instead of DCT bases, the improved GCV method in DFS (DFS-GCV) determines the smoothing parameter by both smoothness and the divergence error, which is more suitable for the correlated noise than DCT-PLS. Therefore, DFS has good capability of reducing correlated noise, achieving the least smoothing error in this test. This is a significant advantage over DCT-PLS and DCT-PLS+DCS because the correlated noise is common in PIV configuration.

Following the strategy of de Silva et al. (2013), we also tested the performance of DFS on improving statistics of turbulent quantities, such as the mean velocity ( $\overline{U}^+$ ), the turbulence intensities ( $\overline{u}^{2+}$ ,  $\overline{v}^{2+}$ ,  $\overline{w}^{2+}$ ), the enstrophy ( $\overline{\omega}^{2+}$ ) and the dissipation ( $\overline{\varepsilon}^+$ ). Herein,  $\omega^{+2} = \omega_x^{+2} + \omega_y^{+2} + \omega_z^{+2}$  and  $\varepsilon^{+2} = 2s_{ij}s_{ij}$ , where  $s_{ij} = 1/2(\partial u_i^+/\partial x_j^+ + \partial u_j^+/\partial x_i^+)$ . The overline denotes the averaging of 100 testing velocity fields.

The so-called percentage differences (de Silva et al. 2013) between the statistics of the original DNS flow and the noisy and different processed fields under the noise level of 10% are calculated for quantitative comparison. For instance, the percentage difference of the streamwise turbulence intensity for the noisy velocity field is defined as

$$\Delta \overline{u}^{2+} = \frac{|\overline{u_n^{2+}} - \overline{u_{DNS}^{2+}}|}{|\overline{u_{DNS}^{2+}}|} \times 100\%, \tag{38}$$

where  $\overline{u_{DNS}^{2+}}$  and  $\overline{u_n^{2+}}$  denote the streamwise turbulence intensity of original DNS field and the noisy field, respectively.

Table 4 summaries the statistic quantities of all the tests. It provides the mean velocity, the turbulence intensities, the enstrophy and the dissipation of the original DNS data and the percentage differences of the noisy and different processed velocity fields for three test cases. Due to the format of the artificial noise, the mean velocity is almost unaffected. For Gaussian white noise, both DCT-PLS+DCS and DFS significantly reduces the noises, improving the accuracy of all the statistic quantities of fluctuations and high-order gradients. The combined method is slightly superior to DFS. For correlated noise, DFS has a distinct advantage over DCT-PLS+DCS, which is consistent with the results of Table 3, and once again validates the effectiveness of DFS in removing correlated noise.

### 3.2 Dealing with outliers

Another important task in the post-process of PIV data is to deal with outliers. Traditional methods need to identify outliers and replace them separately. Differently, DCT-PLS and DFS employ a robust iterative algorithm to fix outliers directly, which requires no identification of outliers. In this section, velocity field with outliers is designed to test the performance of DFS. Instead of  $\mathbf{U}_{DNS}$ , we perform this test on  $\mathbf{U}'_{DNS}$  (the fluctuation of  $\mathbf{U}_{DNS}$ ) for better visualization of flow structures and different outlier types. According to previous research works (Shinneeb et al. 2004; Garcia 2011; Wang et al. 2015b) on correlation calculation, three types of outliers are added to  $\mathbf{U}'_{DNS}$  (Fig. 7).

Type 1: random scattered outliers with a uniform distribution in the range of  $[-5\text{std}(|\mathbf{U}_{DNS}|), 5\text{std}(|\mathbf{U}_{DNS}|)]$  are added to the field at random positions.

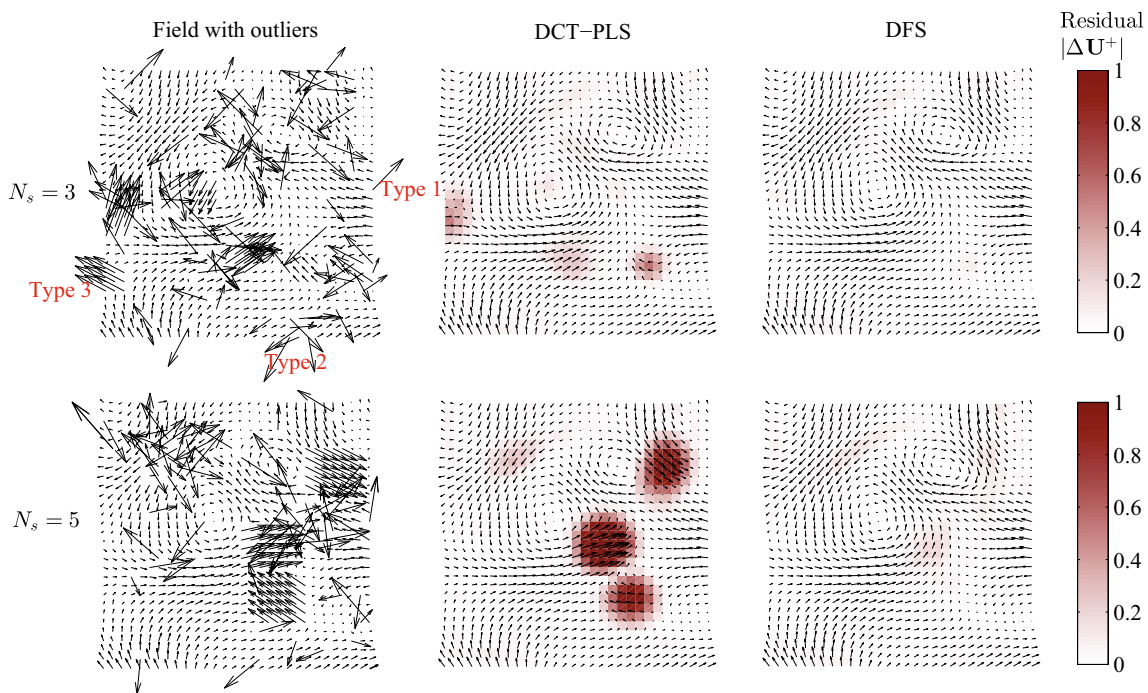
Type 2: clustered outliers with constant size of  $N_s \times N_s \times N_s$  points are added to the field at random positions. Each point in the block has its random value similar to type 1 independently.

**Table 3** Smoothing error for correlated noise

Noise level	Noisy (%)	DCT-PLS (%)	DCT-PLS +DCS (%)	Gaussian (%)	DFS (%)
NL = 5 %	7.71	7.68	6.76	6.05	5.47
NL = 10 %	15.45	15.40	13.55	11.61	10.60

**Table 4** Summary of percentage differences for different statistics with/without DCT-PLS+DCS and DFS

DNS statistics	Gaussian white noise						Correlated noise				
	Case 1			Case 2			Noisy (%)	DCT-PLS +DCS (%)	DFS (%)	DFS (%)	
	Noisy (%)	DCT-PLS +DCS (%)	DFS (%)	Noisy (%)	DCT-PLS +DCS (%)	DFS (%)					
$\bar{U}^+$	21.60	$\Delta\bar{U}^+$	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0010	0.0010	0.0010
$\bar{u}^2$	1.09	$\Delta\bar{u}^2$	0.96	0.24	0.24	0.96	0.25	0.36	1.13	0.85	0.33
$\bar{v}^2$	0.69	$\Delta\bar{v}^2$	1.57	0.32	0.34	1.57	0.37	0.52	2.07	1.61	0.79
$\bar{w}^2$	0.50	$\Delta\bar{w}^2$	2.09	0.43	0.43	8.42	0.80	0.52	2.48	1.84	0.78
$\bar{\omega}^2$	0.0018	$\Delta\bar{\omega}^2$	63.20	7.84	8.23	119.94	9.29	10.60	42.22	39.69	11.73
$\bar{\varepsilon}^+$	0.0018	$\Delta\bar{\varepsilon}^+$	123.88	5.65	6.16	253.96	6.89	8.05	82.89	55.42	20.04



**Fig. 7** A slice from test DNS data at the position of  $z^+ = 463.1$ . The subplots in two rows correspond to two outlier situations with block size of 3 and 5, respectively. The three columns, from left to right,

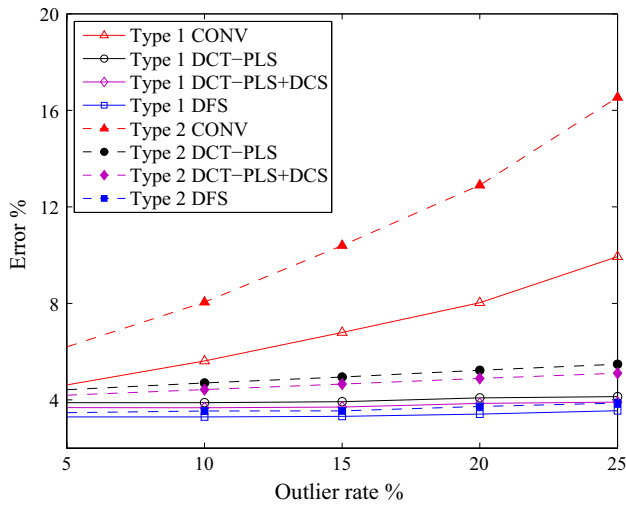
display the field deteriorated by three types of outliers, and the field smoothed by DCT-PLS and DFS. The background colors denote the residuals of processing results

Type 3: clustered outliers with size of  $N_s$  like type 2 are added to the original field at random positions. The block has the same random outlier for all points within the block.

Type 1 is the most common outlier, which is caused by the failure of the cross-correlation analysis. Type 2 occurs because of the poor seeding density or other local imperfection on imaging. When an iterative multi-pass correlation algorithm is employed, the failure in the first cross-correlation analysis of a large-size interrogation window would

affect the vectors of the small-size interrogation windows at the corresponding positions during latest passes of correlation analysis, leading to the occurrence of type 3 outliers. In previous research, DCT-PLS was validated on its good performance in coping with type 1 and type 2. However, Type 3 is more difficult to be fixed because of its larger size and great local influence.

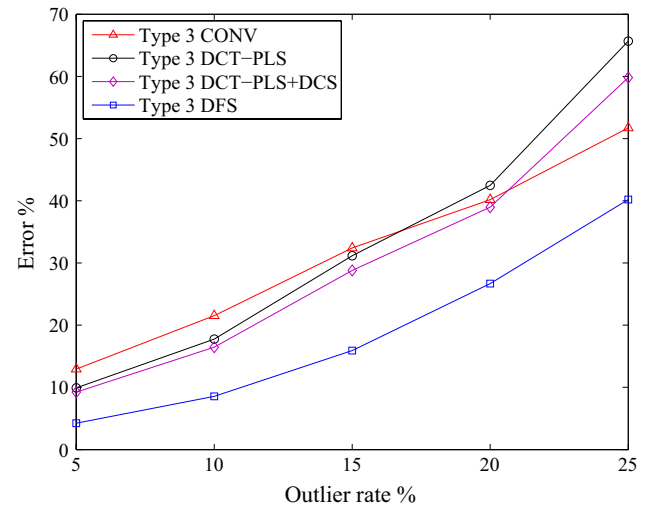
Figure 7 shows tests on all three types outliers with  $N_s = 3$  and 5 under a total outlier rate of 15 % at an example cross-section of the test volume. The background noise level is 5 %. Both robust DCT-PLS and DFS are employed



**Fig. 8** The smoothing error for conventional method, DCT-PLS, DCT-PLS+DCS, and DFS in dealing with type 1 and type 2 outliers. The red, black, purple and blue lines correspond to the error of conventional method, DCT-PLS, DCT-PLS+DCS and DFS, respectively. The solid and dashed lines indicate the type 1 and type 2

to smooth the contaminated velocity field. The corresponding results are shown in the second column and third column by in-plane vectors and background colors denoting the magnitude of velocity residual. For the case of  $N_s = 3$ , both DCT-PLS and DFS lead to a smoothed velocity field. However, the color contours highlight the areas with large residual where the type 3 outliers locate in DCT-PLS. It indicates that DCT-PLS could not inhibit the influence of type 3 outliers in an appropriate way. The failure of DCT-PLS in dealing with type 3 outliers is more obvious in the case of  $N_s = 5$ . It suggests that both methods can deal with type 1 and type 2 outliers successfully, while DCT-PLS fails to correct the outliers of type 3, and it becomes even worse when the size of outlier zone  $N_s$  increases. In fact, it is also a challenge to distinguish and eliminate type 3 outliers from real fluid velocity fields by traditional methods. In this work, by employing the divergence-free constraint, DFS can significantly reduce the influence of these outliers, which is valuable for the post-processing of volumetric PIV velocity.

Outliers with the block size of  $N_s = 3$  is utilized for quantitative comparisons among DCT-PLS, DFS and DCT-PLS+DCS as shown in Figs. 8 and 9. Another important reference provided is the result of a conventional processing method which combines the normalized median test, local-median interpolation and Gaussian smoothing with filtering size of  $3 \times 3 \times 3$  and  $\sigma = 0.65$ . It is found that all DCT-PLS, DCT-PLS+DCS and DFS outperform the conventional method on dealing with outliers. DCS improves the DCT-PLS slightly, reducing smoothing errors for all the three types outliers. DFS performs better than

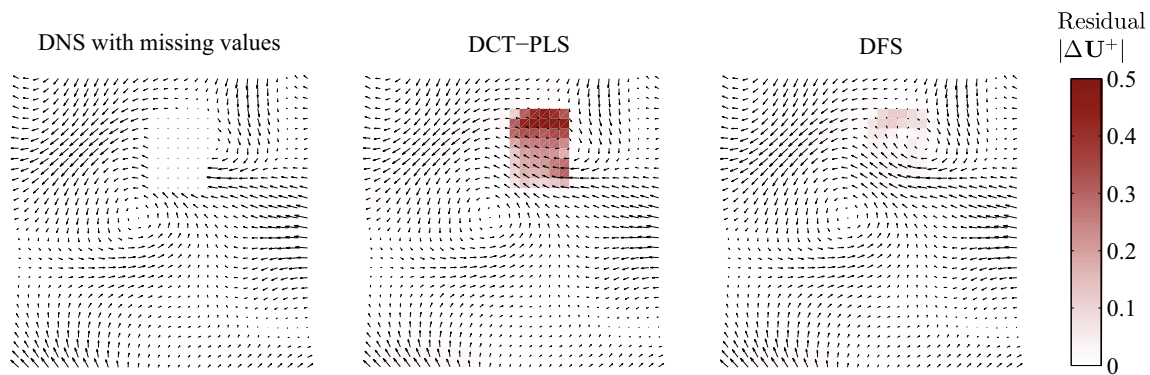


**Fig. 9** The smoothing error for conventional method, DCT-PLS, DCT-PLS+DCS and DFS in dealing with type 3 outliers. The red, black, purple and blue lines correspond to the error of conventional method, DCT-PLS, DCT-PLS+DCS and DFS, respectively

DCT-PLS+DCS especially for type 2 and type 3 outliers. It seems that the type 3 outlier is the most difficult type to be successfully corrected, and all four methods would fail when the outliers rate increase. However, among these methods, DFS is the most reliable approach. Thus, we believe that DFS is a well-functional post-process technique for extreme outliers under reasonable outlier rate in most volumetric PIV experiments.

### 3.3 Padding the missing values

Clustered missing values can sometimes occur in PIV measurements due to model shadows or light reflection resulting in gaps in the flow fields. Clustered missing values can be considered as a special type of block outlier with vector magnitude of zero. However, the positions of the missing values are fortunately known, which means that the post-process can only focus on the missing values without changing other good vectors. In this subsection, the ability of DFS in coping with missing values is tested. Conventional method for filling the gaps is interpolation, which determines the values by their neighbors. DCT-PLS pads the missing vectors by modifying the entire velocity field, but without any physical constraints. As detailed previously, DFS employs divergence-free constraint implicitly when dealing with the velocity field. Therefore, it is expected that DFS would perform better than conventional interpolation methods and DCT-PLS. An numerical example is displayed in Fig. 10, in which a block of vectors with size of  $6 \times 8 \times 4$  is removed and both DCT-PLS and DFS are employed to fix the field. The results clearly shows that



**Fig. 10** A slice from one example DNS velocity field at the position of  $z^+ = 463.1$ . The three columns, from left to right, display the field with missing values, and the field smoothed by DCT-PLS and DFS.

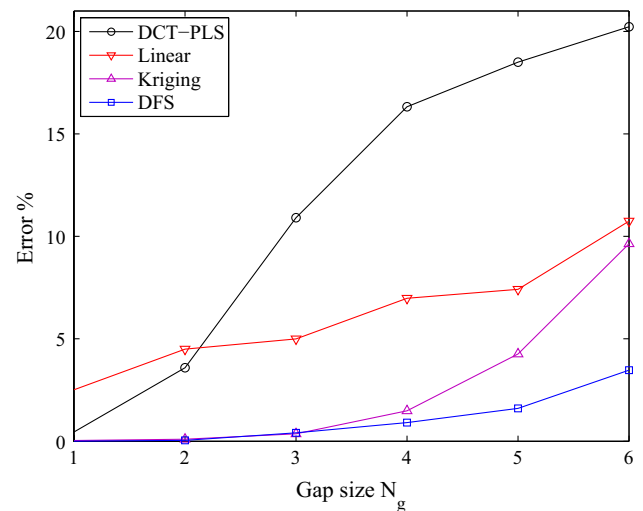
The background colors denote the residuals of processing results. The gap size is  $6 \times 8 \times 4$

DFS can recover the missing vortex structure better than DCT-PLS.

Quantitative comparison among four methods in padding missing values are also tested, including the linear and Kriging interpolations, DCT-PLS and DFS. In the test, vectors from volumes of  $N_g \times N_g \times N_g$  are removed resulting in an overall rate of missing vectors equal to 5%. The Kriging interpolation is implemented using the DACE toolbox in MATLAB with a second polynomial regression and a Gaussian correlation model (Lophaven et al. 2002). Results are displayed in Fig. 11. It shows that Kriging interpolation method and DFS are more powerful on recovering missing values than the other two methods, which both bring very little interpolation errors when  $N_g < 3$ . DFS has a significant advantage in dealing with large gappy zone of  $N_g > 3$ , which is mainly benefited from the divergence-free constraint. It is worthy to note that DCT-PLS leads to very large errors in interpolating larger missing zones of DNS data, which is consistent with the result of Wang et al. (2015b). It is believed that the failure of DCT-PLS is associated with its failure on smoothing correlated noise caused by initialization at the missing values. Moreover, experimental data usually suffer from noises. Kriging method would lose its effectiveness during interpolating noisy field, while DFS still performs well in that situation because of its capability of reducing noise.

#### 4 Computational efficiency and smoothing large volume

According to the discussions in Sect. 2, the most critical step of applying DFS is to calculate the large matrix of DFSBs. Usually, it takes about hour-order computational time and about 25 GB memory using a common computer to calculate a DFSBs matrix corresponding to  $30 \times 30 \times 10$



**Fig. 11** Errors for four methods (linear interpolation, Kriging interpolation, DCT-PLS and DFS) in fixing gappy vectors. The four lines with different colors and marks correspond to the error of linear interpolation, Kriging interpolation, DCT-PLS, and DFS, respectively

or  $21 \times 21 \times 21$  spatial points in MATLAB. However, once such matrix is obtained, the DFS operation on a large set of velocity fields is fast. The Table 5 provides the time cost of DCS, DCT-PLS and DFS in dealing with a  $30 \times 30 \times 10$  velocity field with combined outlier rate of 5% and background noise level of 5%. The test is performed on a computer with quad-core i7 CPU of 3.7GHz and 64 GB RAM. 100 Monte-Carlo simulations are employed to reach the statistical convergence. The time cost of calculating DFSBs is also involved when averaging the computational time of the DFS processing on a single field. It shows that DCT-PLS is the fastest method, which is mainly benefited from the fast DCT and IDCT operation. DCS is proposed to solve the optimization problem using the 'fmincon' function of

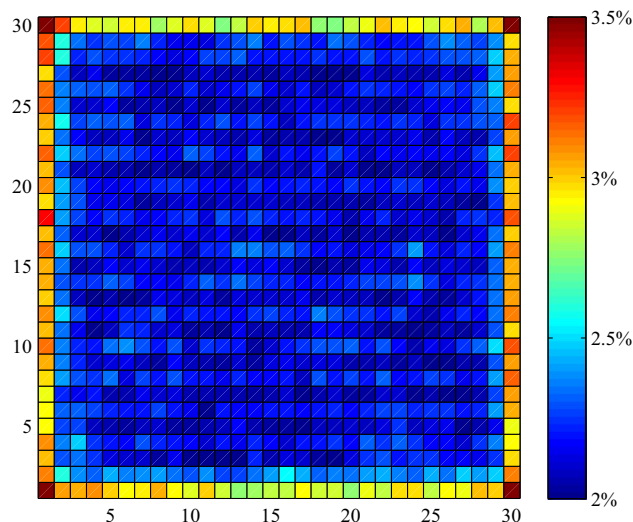
**Table 5** Time cost of different methods

	DCS		DCT-PLS		DFS	
	By fmincon	By $s = 0$	Not robust	Robust	Not robust	Robust
Time cost (s)	>1000	58.3	<0.1	0.3	63.1	135.3

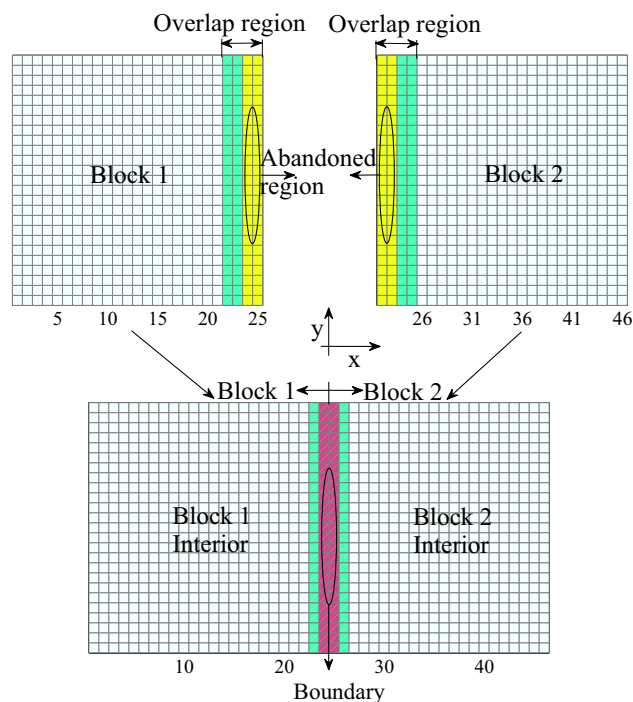
MATLAB (de Silva et al. 2013), which takes much more time and memory than DFS method. In this paper, DCS is also achieved very fast by DFS technique under smoothing parameter of  $s = 0$ , as we can see in Table 5. DFS takes longer time than DCT-PLS but still acceptable comparing to the time-consuming particle reconstruction and 3D cross-correlation in volumetric PIV measurement. Furthermore, the average processing time would be less if more fields are being processed, because the DFBSs only need to be calculated once.

To reduce time cost and memory usage when dealing a large volumetric velocity field, it is smart to divide the entire flow field into several smaller blocks of same size with certain overlap. Thus, only one small DFBSs matrix is necessary for applying DFS on all the blocks, saving considerable memory and time. After processing the sub-fields with DFS, the finally processed flow field is assembled from all the smoothed blocks. The vectors at the overlap can be easily generated by averaging or combining the vectors from the corresponding blocks. A numerical test on the smoothing error of DFS is performed on velocity fields of  $30 \times 30 \times 10$  with  $NL = 5\%$ . The contours of the relative error averaged from 256 testing DNS velocity fields (detailed in Sect. 3) are shown in Fig. 12. The relative error is defined as the magnitude of the difference between the DFS-processed velocity and DNS velocity, normalized by the DNS velocity fluctuation. The figure indicates that DFS method usually leads to larger errors at the borders of smoothing block, while the inner region has fairly low relative errors around 2%. Such border-effect is caused by the first-order difference scheme at the border, which only affects about two layers of grid in the inner region. Therefore, when block DFS is used, the smoothing blocks should share an overlap of at least four layers of grid, and two layers of border need to be removed when generating the entire processed field to avoid the large errors at the borders.

To validate the effectiveness of block DFS further, two tests on a large DNS velocity field (turbulent channel flow as introduced in Sect. 3) with  $256 \times 256 \times 16$  spatial points are performed. In case 1, only Gaussian white noise with  $NL = 5\%$  is added to the DNS field. In case 2, besides background Gaussian noise, all the three types of outliers each with outlier rate of 5% and  $N_s = 3$  are added, similar to Fig. 7. The whole velocity field is divided into  $12 \times 12$  sub-volumes with  $25 \times 25 \times 16$  spatial grid points, sharing an overlap of  $4 \times 4 \times 16$ . After block DFS operation, these sub-volumes abandon two layers of border grid



**Fig. 12** Contours of the relative error magnitude of DFS-smoothed velocity fields ( $NL = 0.05$ )



**Fig. 13** A sketch of the block DFS scheme

at the overlap and combine into a complete velocity field. A sketch of block DFS is shown in Fig. 13 to explain how to merge two sub-volumes.



**Table 6** Smoothing errors of DCT-PLS and block DFS (%)

	DCT-PLS	DFS		
	Entire field	Entire field	At boundaries	In inner
Noisy field				
Velocity	2.44	1.98	2.05	1.97
Vorticity	13.95	12.66	12.46	12.71
Outlier field				
Velocity	7.00	3.22	4.19	2.99
Vorticity	27.75	16.78	19.26	16.19

**Table 7** RMSs of divergence residuals  $\nabla \cdot \mathbf{u}^+$  for DCT-PLS and block DFS

	DCT-PLS	DFS		
	Entire field	Entire field	At boundaries	In inner
Noisy field	0.0032	0.0002	0.0006	$<10^{-13}$
Outlier field	0.0066	0.0010	0.0025	$<10^{-13}$

In the tests, the smoothing errors of velocity are defined as Eq. (36) in Sect. 3.1. Errors calculated from three difference regions are compared for showing the influence of border-effect. They are the average error in the entire field, the error at the boundaries and the error at the inner region excluding the boundaries (as shown in Fig. 13). Considering that the calculation of vorticity at boundaries is based on the velocities from two neighboring smoothed blocks, the corresponding errors at the boundaries are the biggest concern for block DFS. The errors of vorticity are defined by a similar equation with the errors of velocities. Table 6 provides all three different errors of velocity and vorticity for the tests. Errors of DCT-PLS are also provided in Table 6 as important references. It shows that errors at boundaries and the interior appear similarly for both velocity and vorticity in the noise test (case 1), which indicates that the block scheme does not cause biased errors at boundaries. For another test with 15 % of three types of outliers (case 2), the smoothing errors at the boundaries are obviously larger than the ones at the interior. However, even at the boundaries, the smoothing errors of DFS are smaller than the ones for DCT-PLS, indicating that the boundary effect is not a significant issue for block DFS. The root mean squares (RMSs) of divergence residuals for DCT-PLS and DFS are presented in Table 7. It shows that the block DFS cannot completely remove the divergence errors at the boundaries, because the calculation of divergence at the boundaries involves velocities from two separated blocks. However, the remaining divergence errors at boundaries are much smaller than the corresponding errors caused by DCT-PLS, which means that block DFS still retains good divergence-reducing capability at the

boundaries. On the other hand, it is worthy to note that the boundary effect will be further reduced if larger overlap region is applied in the block DFS.

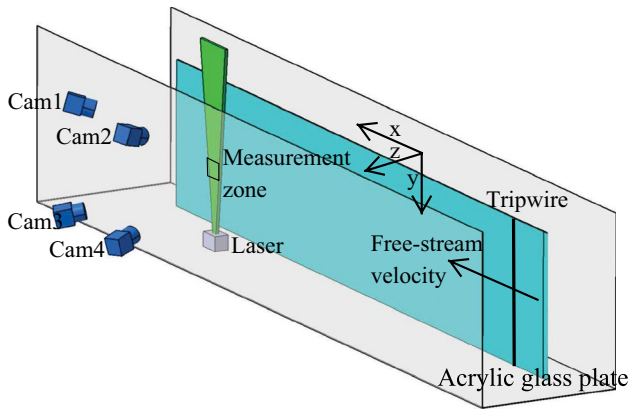
## 5 DFS on tomographic PIV data

In this section, DFS is applied to process real TPIV data. Two sets of tomographic PIV (TPIV) data of different flows are employed for evaluating DFS. One is a turbulent boundary layer (TBL) flow, which is a similar flow with the former DNS data in Sect. 3. Another is about a vortex breakdown over a delta wing, which is a complex flow phenomenon. In the test of TBL, statistics are assessed to validate the effectiveness of DFS quantitatively. In the test of vortex breakdown, the ability to improving instantaneous flow visualization and vortex identification is addressed.

### 5.1 Turbulent boundary layer

The experiment of turbulent boundary layer was conducted in a large water tunnel in the Beijing University of Aeronautics and Astronautics in China. The tunnel has a working section of  $16 \text{ m} \times 1 \text{ m} \times 1.2 \text{ m}$  in the streamwise, spanwise and vertical directions, respectively. A  $1 \text{ m} \times 7 \text{ m}$  acrylic glass plate with a tripwire near the elliptic leading edge was vertically put in the tunnel to generate the turbulent boundary layer. A schematic of the channel flow facility and the experimental setup is shown in Fig. 14. The TPIV system includes four high-resolution ( $2058 \times 2456$  pixels) and dual-exposure CCD cameras with 45 mm Nikon lenses, and a 500 mJ Nd:Yag laser device. The measurement was focused on a region of  $80 \text{ mm} \times 45 \text{ mm} \times 16 \text{ mm}$  at 6.7 m downstream the leading edge of the plate. The  $x$ ,  $y$  and  $z$  axes were set as the streamwise, spanwise and wall-normal directions of the plate, respectively. The laser sheet was parallel to the plate, illuminating a volume with vertical range from  $z = 0.6 \text{ mm}$  to  $z = 2.1 \text{ mm}$  along the thickness direction, which is in the logarithmic law region of the velocity profile. The free-stream velocity was adjusted to 412.8 mm/s, and the corresponding turbulence level was  $<1 \%$ .

Laser Doppler velocimetry (LDV) was employed to measure the velocity profile and obtain the flow parameters. The wall viscous unit and skin friction velocity were obtained by fitting the LDV data with the Musker profile method (Kendall and Koochesfahani 2008). This resulted in a wall unit of 0.068 mm and a skin friction velocity of 15.6 mm/s, corresponding to a friction Reynolds number  $Re_\tau$  of 1769. The experimental TPIV data were processed by an in-house Tomo-PIV program. The multiplicative algebraic reconstruction technique (MART, Elsinga et al. 2006) with pixel-to-voxel ratio of one was employed to reconstruct the 3D particle intensity field, giving a volume size of



**Fig. 14** Experimental setup for TBL measurement

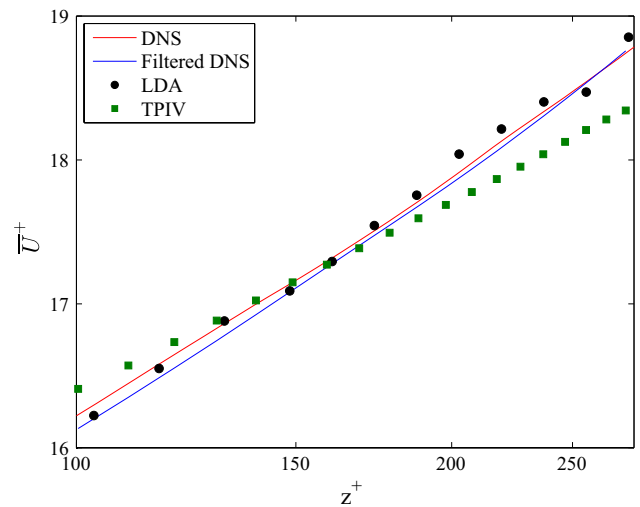
**Table 8** Parameters for TPIV velocity field

Spacing $\Delta^+$	9.75
Correlation window $W^+$	39.0
Wall-normal range $z^+$	100.3–275.8
Number of vectors per volume	$118 \times 65 \times 19$
Size of velocity field (wall unit)	$1135.6 \times 621.2 \times 175.5$
Number of velocity fields	150
Repetition frequency	4 Hz

$1454 \times 818 \times 291$  voxels with magnification of 0.055 mm per voxel. The cross-correlation was performed on window of  $48 \times 48 \times 48$  voxels with 75 % overlap. The parameters of the TPIV velocity field are listed in Table 8.

The TPIV velocity fields were further processed by DCT-PLS, DCS, DCT-PLS+DCS and DFS, respectively. Statistics of the mean velocity, turbulent intensity, enstrophy and dissipation, as defined in Sect. 3.1, were calculated based on the processed velocity fields. To save memory and time, the original velocity fields were divided into several blocks with size of  $25 \times 25 \times 19$  spatial grid points. Therefore, all the processing methods could be employed to deal with the same small blocks. More than 1200 such blocks were calculated to obtain converged statistic results.

To provide a standard reference, another set of DNS data was employed to get statistics. The DNS data came from a direct numerical simulation of a zero pressure gradient turbulent boundary layer over a flat plate (Simens et al. 2009; Sillero et al. 2013; Borrell et al. 2013). The total DNS data contains  $15,361 \times 4096 \times 535$  spatial nodes for streamwise, spanwise and wall-normal directions. The developing TBL has a range of friction Reynold number from 976 to 2040. The data selected here were extracted from the entire data at the local with the local  $Re_\tau$  from 1749 to 1793 and wall-normal position ranging from 100 to 276 (wall units), which are very close to our experimental configuration.



**Fig. 15** Mean velocity profile calculated from DNS, filtered DNS, LDV, and TPIV data

The testing DNS data have  $651 \times 4096 \times 45$  nodes for streamwise, spanwise and wall-normal directions, with corresponding average spacings  $\Delta^+$  of 6.42, 3.71 and 3.32. Linear interpolation was employed to make the selected DNS data have equal spacings as the experimental data. To simulate the PIV filter effect, the DNS data were further filtered by a moving average block with equal size of the spatial resolution of the TPIV data (39.0, wall unit).

The velocity profile is shown in Fig. 15. It shows that the filter effect brings very small change on the velocity profile. The LDV data match quite well with the DNS data as well as the TPIV data. The TPIV profile here is obtained from the raw velocity field with only post-processing of outliers correction, but not other smoothing or correcting processes. It is worthy to note that the slope of the TPIV velocity is slightly smaller than DNS and LDV data. The bias error is believed to be caused by modulation effect of ghost particles, which was also reported by Atkinson et al. (2011) and Gao (2011).

Table 9 provides all the statistical results for DNS, filtered DNS, TPIV and processed TPIV data at  $z^+ = 197.8$ . It shows that the filtering effect brings very limited variation on the mean velocity of the original DNS data and slightly reduce the turbulent intensities. However, the enstrophy and dissipation are significantly reduced by the filtering. Considering that such filtering effect is inherently introduced in the correlation analysis, the filtered DNS data provide a better reference to estimate the accuracy of the testing processing methods. It shows that the raw TPIV data cause a significant overestimate on the third turbulent intensity  $\overline{w^{2+}}$  (even larger than  $\overline{v^{2+}}$ ), which is caused by the high uncertainty along the thickness direction of measurement volume. During the particle reconstruction,

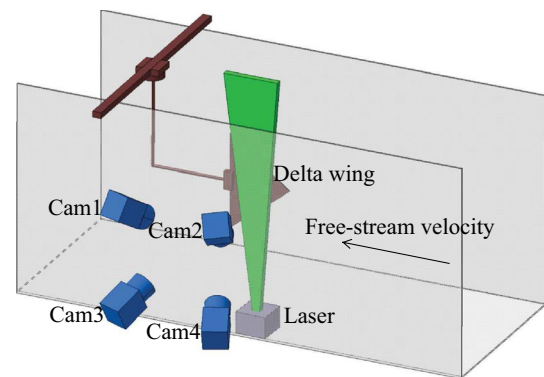
**Table 9** The statistical results for DNS, filtered DNS, TPIV and processed TPIV data

	DNS		TPIV				
	Original	Filtered	No process	DCS	DCT-PLS	DCT-PLS+DCS	DFS
$\overline{U}^+$	17.81	17.81	17.66	17.66	17.66	17.66	17.66
$\overline{u^2}^+$	1.95	1.86	1.89	1.88	1.85	1.85	1.85
$\overline{v^2}^+$	1.37	1.26	1.17	1.17	1.12	1.12	1.12
$\overline{w^2}^+$	1.11	0.99	1.37	1.19	1.15	1.05	1.04
$\overline{\omega^2}^+$	0.0147	0.0040	0.0119	0.012	0.0055	0.0054	0.0052
$\overline{\varepsilon}^+$	0.0153	0.0041	0.0213	0.0138	0.0086	0.0064	0.0060

the reconstructed particles are normally elongated along the thickness direction due to the small viewing angles of imaging, which causes a high uncertainty of determining the particle displacements along that direction. All the testing methods reduce the turbulent intensities at different degrees. A highly effective correction on the overestimated turbulent intensity  $\overline{w^2}^+$  has been observed. Among all the testing methods, DCT-PLS+DCS and DFS perform the best. The enstrophy and dissipation for raw TPIV data are much larger than the filtered DNS because of their sensitivity to the noise error. In the comparison, DCT-PLS+DCS and DFS provide the most accurate corrections on the enstrophy and dissipation as well. It seems that DFS and DCT-PLS+DCS share similar performance on improving the flow statistics. Results in Table 9 show a little different trend to the results of Table 4. In fact, the real experimental contains a comprehensive error of all different types of errors, while those errors are individually tested in Table 4. Furthermore, other experimental errors uncovered in Sect. 3 might cause more complicated situation in the real experimental data, such as the errors due to the ghost particles (Elsinga et al. 2011).

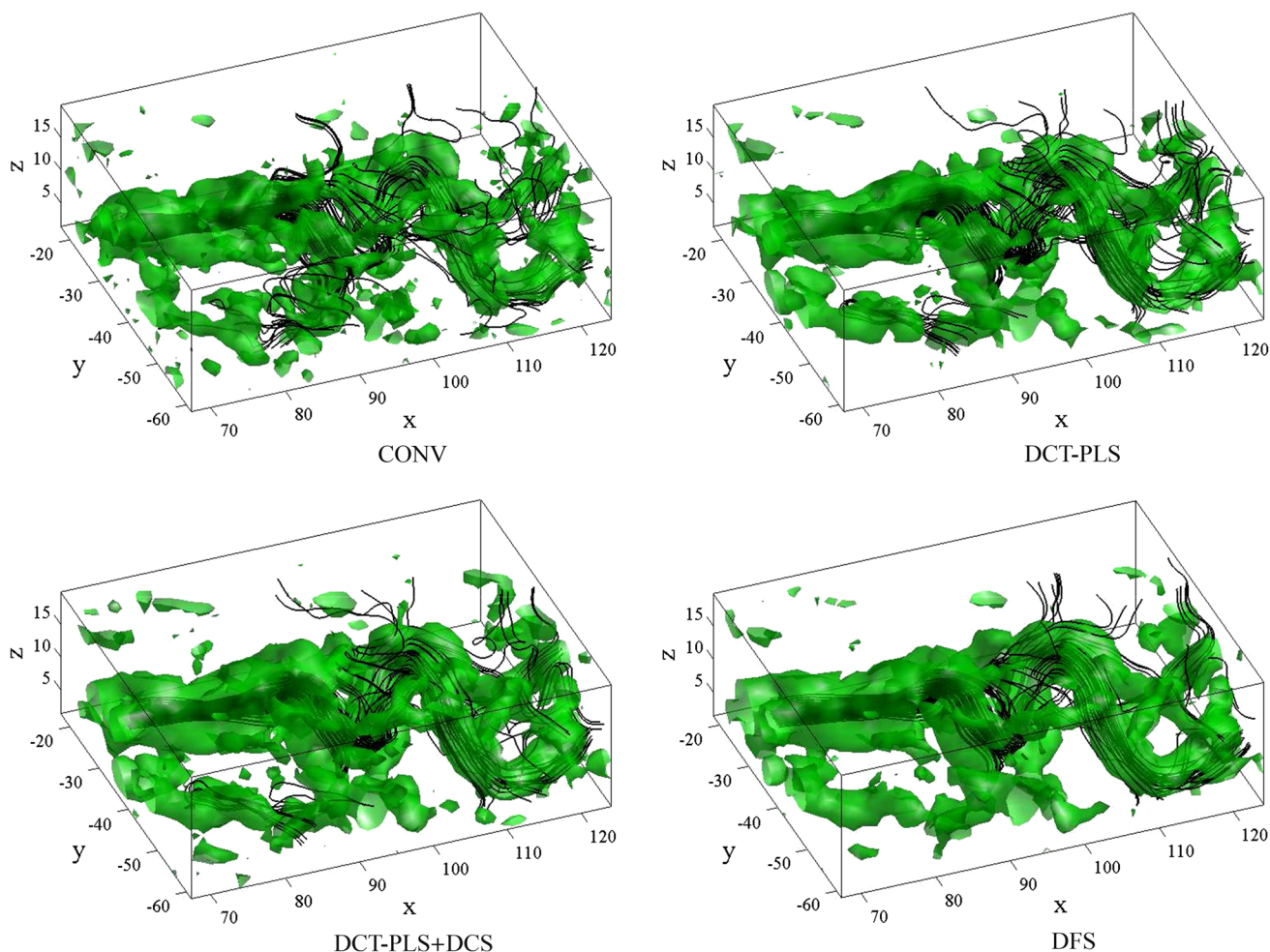
## 5.2 Vortex breakdown

Besides the testing on the statistics, a further test of DFS for its performance on instantaneous field is presented in this subsection based on a complex flow of vortex breakdown. The TPIV measurement for vortex breakdown over a delta wing was conducted in a small water channel at the Beijing University of Aeronautics and Astronautics in China. The water channel has a working section of  $3\text{ m} \times 0.6\text{ m} \times 0.7\text{ m}$  in the streamwise, spanwise and vertical directions, respectively. The channel facility and experimental setup are illustrated in Fig. 16. The free stream velocity  $U_\infty$  was  $60\text{ mm/s}$ . The delta wing with  $52^\circ$  sweep angle generated leading vortices when the angle of attack was  $20^\circ$  under Reynolds number of  $1.2 \times 10^4$  with chord length  $c$  of  $200\text{ mm}$ . The thickness of the delta wing is  $3\text{ mm}$ , leading to a thickness-to-chord ratio

**Fig. 16** Experimental setup for vortex breakdown measurement

of  $1.5\%$ . A laser sheet with thickness of  $20\text{ mm}$  emitted by a dual-exposure Nd:YAG laser was aligned parallel to the delta wing. Four high-resolution ( $2058 \times 2456$  pixels) and dual-exposure CCD cameras were employed to record the particle images at  $5\text{ Hz}$ . The measurement zone was focused on the middle of the chord length, where vortex breakdown occurred. An in-house module of Tomo-PIV was used to process the experimental data. MART was used to reconstruct the particle field in the measurement volume, and cross-correlation analysis with window size of  $48 \times 48 \times 48$  and  $50\%$  overlap was applied to obtain the 3D velocity field. The resulting velocity field has vectors of  $44 \times 35 \times 17$  corresponding to a physical domain of  $56.76\text{ mm} \times 44.88\text{ mm} \times 21.12\text{ mm}$ . The origin of the coordinate is set at the apex of the delta wing. Three coordinates correspond to the chordwise, spanwise and wall-normal directions  $(x, y, z)$ , respectively.

We post-process the data by four methods for comparison: conventional smoothing, DCT-PLS, DCT-PLS+DCS and block DFS. The conventional method (CONV) combines the determination of outliers with the normalized media test, replacement of outliers with the local-median interpolation, and Gaussian smoothing of  $3 \times 3 \times 3$  kernel size with a default standard variance of  $0.65$ . The block DFS divided the entire domain into two subzones of



**Fig. 17** Isosurfaces of  $Qc/U_\infty = 100$  and vorticity lines from the results of the four method

$24 \times 35 \times 17$  grids with a overlap of  $4 \times 35 \times 17$ . The final processed field is composed of two  $22 \times 35 \times 17$  subzones coming from the two smoothed subzones, which means that each subzone provides a  $2 \times 35 \times 17$  domain to recover the overlapped region.

An example snapshot of vortex breakdown phenomenon is shown in Fig. 17. After smoothing with three techniques, the velocity field is improved on noise reduction. The leading vortex is identified using the  $Q$ -criterion (Hunt et al. 1988). The vorticity lines starting from  $x = 72.57$  mm near the leading vortex core are displayed as well on the same figures. All post-processed results show a clear curled vortex core associated with the spiral vortex breakdown. It shows that compared to conventional processing method, DCT-PLS, DCT-PLS+DCS and DFS improve the visualization by reducing noisy isosurfaces of  $Q$  and make the vortex core more smooth. In the result of DCT-PLS, the vortex core is still not smoothed enough, and the vorticity lines are messy at the downstream of the breakdown spot. DCS improves the result of DCT-PLS by making

the vortex core more smooth and well-identified. DFS achieves best smoothing results among all these methods, which gives a clear cluster of spiral vorticity lines, agreeing well with the well-smoothed leading vortex core, which provides an experimental evidence of the advantage of DFS.

## 6 Conclusions

The divergence-free smoothing method discussed in this article is a combination of penalized least squares regression and divergence corrective scheme, which smooths the vector field of volumetric experiment measurement while removing the divergence errors at the same time. DFS employs GCV method to find the optimal smoothing parameter  $s$ , which is adaptive to different noise situations. Tests suggest that DFS has an outstanding advantage in smoothing correlated errors over DCT-PLS, DCS and DCT-PLS+DCS.

By employing an iterative weight-changing algorithm, DFS achieves validation of raw data, replacement of spurious and missing vectors, smoothing, and zero-divergence correction for the velocity field simultaneously. Numerical assessments demonstrate that DFS always outperform DCT-PLS, DCT-PLS+DCS and some traditional processing methods in dealing with different types of outliers and missing values. For padding missing values, DFS could recover the velocity field under reasonable rate of missing vectors, while DCT-PLS normally does not work very well. On fixing velocity fields with large gaps ( $N_g > 3$ ), DFS is superior to all other methods including the well-known Kriging interpolation.

Calculating the large matrix of DFSBs is usually time-consuming and memory-hogging. Block DFS is suggested to process large velocity field to save time and memory usage. Numerical tests validate the applicability of the block DFS. Block DFS is also applied to process two sets of TPIV data, resulting in improved statistical results and better flow visualization compared to conventional method, DCT-PLS and even the combined method DCT-PLS+DCS.

**Acknowledgments** This work is supported by the National Natural Science Foundation of China (11472030, 11327202, 11490552). We would like to thank Professor Charles Meneveau for providing the DNS data of turbulent channel flow. We would also like to thank Professor Sillero for providing the DNS data of turbulent boundary layer flow.

### Appendix 1: Specific forms of A and M

Assume that the  $l$ -th ( $l = 1, 2, 3$ ) component of velocity at the position  $(i, j, k)$  is arranged as the  $(i + (j - 1)n_x + (k - 1)n_x n_y + (l - 1)n)$ -th element of  $\mathbf{U}_{\text{exp}}$  or  $\mathbf{U}_s$ . Then corresponding divergence operator  $\mathbf{A}$  and  $\mathbf{M}$  are calculated by the Eqs. (39)–(42), noting that the symbol  $\otimes$  denotes the Kronecker product between two matrices.

$$\mathbf{A} = \left[ \begin{array}{c} \frac{1}{\Delta x} \mathbf{I}_{n_z} \otimes \mathbf{I}_{n_y} \otimes \mathbf{D}_{n_x}, \\ \frac{1}{\Delta y} \mathbf{I}_{n_z} \otimes \mathbf{D}_{n_y} \otimes \mathbf{I}_{n_x}, \frac{1}{\Delta z} \mathbf{D}_{n_z} \otimes \mathbf{I}_{n_y} \otimes \mathbf{I}_{n_x} \end{array} \right], \tag{39}$$

where  $\mathbf{I}_{n_x}$ ,  $\mathbf{I}_{n_x}$  and  $\mathbf{I}_{n_z}$  are identical matrices and  $\mathbf{D}_{n_x}$ ,  $\mathbf{D}_{n_x}$  and  $\mathbf{D}_{n_z}$  take the following form

$$\mathbf{D}_m = \left[ \begin{array}{ccc} -1 & 1 & \\ -1/2 & 0 & 1/2 \\ & -1/2 & 0 & 1/2 \\ & & \dots & \\ & & & -1 & 1 \end{array} \right]_{m \times m} \quad (m = n_x, n_y \text{ or } n_z). \tag{40}$$

$$\begin{aligned} \mathbf{M} = & \frac{1}{\Delta x^2} \mathbf{I}_3 \otimes \mathbf{I}_{n_z} \otimes \mathbf{I}_{n_y} \otimes \mathbf{N}_{n_x}^2 + \frac{1}{\Delta y^2} \mathbf{I}_3 \otimes \mathbf{I}_{n_z} \otimes \mathbf{N}_{n_y}^2 \otimes \mathbf{I}_{n_x} \\ & + \frac{1}{\Delta z^2} \mathbf{I}_3 \otimes \mathbf{N}_{n_z}^2 \otimes \mathbf{I}_{n_y} \otimes \mathbf{I}_{n_x} + \frac{2}{\Delta x \Delta y} \mathbf{I}_3 \otimes \mathbf{I}_{n_z} \\ & \otimes \mathbf{N}_{n_y} \otimes \mathbf{N}_{n_x} + \frac{2}{\Delta y \Delta z} \mathbf{I}_3 \otimes \mathbf{N}_{n_z} \\ & \otimes \mathbf{N}_{n_y} \otimes \mathbf{I}_{n_x} + \frac{2}{\Delta x \Delta z} \mathbf{I}_3 \otimes \mathbf{N}_{n_z} \otimes \mathbf{I}_{n_y} \otimes \mathbf{N}_{n_x}, \end{aligned} \tag{41}$$

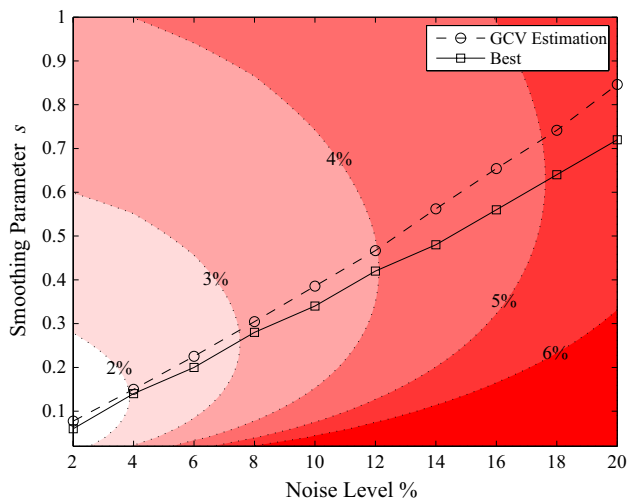
where  $\mathbf{N}_{n_x}$ ,  $\mathbf{N}_{n_x}$  and  $\mathbf{N}_{n_z}$  take the following form.

$$\mathbf{N}_m = \left[ \begin{array}{ccc} -1 & 1 & \\ 1 & -2 & 1 \\ & 1 & -2 & 1 \\ & & \dots & 1 \\ & & & 1 & -1 \end{array} \right]_{m \times m} \quad (m = n_x, n_y \text{ or } n_z). \tag{42}$$

The Eq. (41) is a generalization of the work of Buckley (1994), who detailed the form of  $\mathbf{M}$  for the smoothing of two-dimensional scalar data.

### Appendix 2: Numerical test for GCV

To validate the performance of the GCV method on estimating smoothing parameter, a test based on 100 DNS velocity fields with different noise levels is performed to get the average results. Details of the testing DNS data and the added artificial noise are introduced in Sect. 3, where more tests of DFS are performed on the same DNS data. To find the best smoothing parameter  $s$  directly in this test, DFS using different value of  $s$  (from 0.02 to 1) is scanned to smooth the noisy velocity field, with which the estimation of  $s$  by GCV method is compared. The relative smoothing error is defined as the norm of the difference between the DFS-processed field and the original DNS field, normalized by the norm of the original DNS field with the free-stream velocity subtracted [Eq. (36) in Sect. 3.1]. Errors of DFS using different  $s$  to smooth velocity field with different noise levels are contoured in Fig. 18. The best  $s$  marked in Fig. 18 is acquired by finding the result with the smallest smoothing error under the corresponding noise level. The estimations of  $s$  by GCV are also marked in the figure for comparison. It shows that GCV method always choose a decent smoothing parameter, which is very close to the best DFS results at different noise levels. In fact, the differences between the smoothing errors resulting from GCV estimation and the ones using the best  $s$  are always less than 0.1 %. It suggests that the GCV method is a good adaptive method for different noise levels. Another interesting result is that the optimal smoothing parameter  $s$  is proportional to the noise level.



**Fig. 18** Test of GCV method. The contours indicate the relative smoothing error. Two curves with different markers stand for the best parameter  $s$  and the GCV-estimated  $s$

Therefore, it seems possible to estimate the noise level of the experimental data according to the GCV estimation of smoothing parameter  $s$ .

Besides the theoretical proof (Craven and Wahba 1978) and the above numerical test, the GCV method, as a general method, has been successfully applied in the DCT-PLS method on dealing with various types of experimental errors (Garcia 2010). In current work, we also perform a systematical test of DFS on all possible errors of PIV data in the following Sects. 3–5. The good performances of both DCT-PLS and DFS in practical application strongly evidence that the GCV method is an effective and robust algorithm in choosing the smoothing parameter.

At last, it is worth to note that any smoothing operation has certain low-pass filtering effect for the flow field. For DFS, larger smoothing parameter means stronger noise-reduce effect and, therefore, larger range of frequency truncation. DFS seeks a good balance between the two effects of noise reduction and spatial attenuation by using the GCV method.

## References

- Adrian RJ, Westerweel J (2011) Particle image velocimetry, vol 30. Cambridge University Press, Cambridge
- Atkinson C, Coudert S, Foucaut JM, Stanislas M, Soria J (2011) The accuracy of tomographic particle image velocimetry for measurements of a turbulent boundary layer. *Exp Fluids* 50(4):1031–1056
- Borrell G, Sillero JA, Jiménez J (2013) A code for direct numerical simulation of turbulent boundary layers at high reynolds numbers in BG/P supercomputers. *Comput Fluids* 80(7):3743
- Buckley M (1994) Fast computation of a discretized thin-plate smoothing spline for image data. *Biometrika* 81(2):247–258
- Coleman D, Holland P, Kaden N, Klema V, Peters SC (1980) A system of subroutines for iteratively reweighted least squares computations. *ACM Trans Math Softw (TOMS)* 6(3):327–336
- Craven P, Wahba G (1978) Smoothing noisy data with spline functions. *Numer Math* 31(4):377–403
- de Kat R, van Oudheusden BW (2012) Instantaneous planar pressure determination from PIV in turbulent flow. *Exp Fluids* 52(5):1089–1106
- de Silva CM, Philip J, Marusic I (2013) Minimization of divergence error in volumetric velocity measurements and implications for turbulence statistics. *Exp fluids* 54(7):1–17
- Elsinga GE, Scarano F, Wieneke B, van Oudheusden BW (2006) Tomographic particle image velocimetry. *Exp Fluids* 41(6):933–947
- Elsinga GE, Westerweel J, Scarano F, Novara M (2011) On the velocity of ghost particles and the bias errors in tomographic-PIV. *Exp Fluids* 50(4):825–838
- Foucaut JM, Carlier J, Stanislas M (2004) PIV optimization for the study of turbulent flow using spectral analysis. *Meas Sci Technol* 15(6):1046–1058
- Gao Q (2011) Evolution of eddies and packets in turbulent boundary layers. PhD thesis, University of Minnesota
- Garcia D (2010) Robust smoothing of gridded data in one and higher dimensions with missing values. *Comput Stat Data Anal* 54(4):1167–1178
- Garcia D (2011) A fast all-in-one method for automated post-processing of PIV data. *Exp Fluids* 50(5):1247–1259
- Graham J, Lee M, Malaya N, Moser R, Eyink G, Meneveau C, Kanov K, Burns R, Szalay A (2013) Turbulent channel flow data set. <http://turbulence.pha.jhu.edu/docs/README-CHANNEL.pdf>
- Gunes H, Rist U (2007) Spatial resolution enhancement/smoothing of stereo-particle-image-velocimetry data using proper-orthogonal-decomposition-based and kriging interpolation methods. *Phys Fluids (1994-present)* 19(6):064,101
- Gunes H, Sirisup S, Karniadakis GE (2006) Gappy data: To krig or not to krig? *J Comput Phys* 212(1):358–382
- Heiberger RM, Becker RA (1992) Design of an  $s$  function for robust regression using iteratively reweighted least squares. *J Comput Graph Stat* 1(3):181–196
- Holland PW, Welsch RE (1977) Robust regression using iteratively reweighted least-squares. *Commun Stat Theory Methods* 6(9):813–827
- Hunt JCR, Wray AA, Moin P (1988) Eddies, streams, and convergence zones in turbulent flows. *Stud Turbul Using Numer Simul Databases* 2:193–208
- Kendall A, Koochesfahani M (2008) A method for estimating wall friction in turbulent wall-bounded flows. *Exp Fluids* 44(5):773–780
- Li Y, Perlman E, Wan M, Yang Y, Meneveau C, Burns R, Chen S, Szalay A, Eyink G (2008) A public turbulence database cluster and applications to study lagrangian evolution of velocity increments in turbulence. *J Turbul* 9:N31
- Liang D, Jiang C, Li Y (2003) Cellular neural network to detect spurious vectors in PIV data. *Exp Fluids* 34(1):52–62
- Lophaven SN, Nielsen HB, Søndergaard J (2002) Dace—a matlab kriging toolbox, version 2.0. Technical report
- Murray NE, Ukeiley LS (2007) An application of gappy pod. *Exp Fluids* 42(1):79–91
- Nogueira J, Lecuona A, Rodriguez P (1997) Data validation, false vectors correction and derived magnitudes calculation on PIV data. *Meas Sci Technol* 8(12):1493
- Perlman E, Burns R, Li Y, Meneveau C (2007) Data exploration of turbulence simulations using a database cluster. In: Proceedings of the 2007 ACM/IEEE conference on supercomputing, ACM

- Raben SG, Charonko JJ, Vlachos PP (2012) Adaptive gappy proper orthogonal decomposition for particle image velocimetry data reconstruction. *Meas Sci Technol* 23(2):025,303
- Raffel M, Willert CE, Wereley S, Kompenhans J (2007) *Particle image velocimetry: a practical guide*. Springer, Berlin, Heidelberg
- Sadati M, Luap C, Kröger M, Öttinger HC (2011) Hard vs soft constraints in the full field reconstruction of incompressible flow kinematics from noisy scattered velocimetry data. *J Rheol* (1978-present) 55(6):1187–1203
- Schiavazzi D, Coletti F, Iaccarino G, Eaton JK (2014) A matching pursuit approach to solenoidal filtering of three-dimensional velocity measurements. *J Comput Phys* 263:206–221
- Schrijer FFJ, Scarano F (2008) Effect of predictor–corrector filtering on the stability and spatial resolution of iterative PIV interrogation. *Exp Fluids* 45(5):927–941 (15)
- Shinneeb A, Bugg J, Balachandar R (2004) Variable threshold outlier identification in PIV data. *Meas Sci Technol* 15(9):1722
- Sillero JA, Jiménez J, Moser RD (2013) One-point statistics for turbulent wall-bounded flows at reynolds numbers up to  $\delta^+ \approx 2000$ . *Phys Fluids* (1994–present) 25(10):105102
- Simens MP, Jiménez J, Hoyas S, Mizuno Y (2009) A high-resolution code for turbulent boundary layers. *J Comput Phys* 228:42184231
- Song X, Yamamoto F, Iguchi M, Murai Y (1999) A new tracking algorithm of PIV and removal of spurious vectors using delaunay tessellation. *Exp Fluids* 26(4):371–380
- Wahba G (1990) *Spline models for observational data*, vol 59. SIAM, Philadelphia
- Wang C, Gao ZH, Huang JT, Zhao K, Jing LI (2015a) Smoothing methods based on coordinate transformation in a linear space and application in airfoil aerodynamic design optimization. *Sci China Technol Sci* 58(2):297–306
- Wang H, Gao Q, Feng L, Wei R, Wang J (2015b) Proper orthogonal decomposition based outlier correction for PIV data. *Exp Fluids* 56(2):1–15
- Weinert HL (2007) Efficient computation for Whittaker–Henderson smoothing. *Comput Stat Data Anal* 52(2):959–974
- Westerweel J (1994) Efficient detection of spurious vectors in particle image velocimetry data. *Exp Fluids* 16(3–4):236–247
- Westerweel J, Scarano F (2005) Universal outlier detection for PIV data. *Exp Fluids* 39(6):1096–1100
- Yang G, Kilner P, Firmin D, Underwood S, Burger P, Longmore D (1993) 3D cine velocity reconstruction using the method of convex projections. In: *Proceedings IEEE conference on computers and cardiology*, pp 361–364
- Young C, Johnson D, Weckman E (2004) A model-based validation framework for PIV and PTV. *Exp Fluids* 36(1):23–35