Comput Mech (2014) 53:91–99
DOI 10.1007/s00466-013-0895-y

ORIGINAL PAPER

Space–time computation techniques with continuous representation in time (ST-C)

Kenji Takizawa · Tayfun E. Tezduyar

Received: 1 June 2013 / Accepted: 27 June 2013 / Published online: 13 July 2013

Abstract We introduce space–time computation techniques with continuous representation in time (ST-C), using temporal NURBS basis functions. This gives us a temporally smooth, NURBS-based solution, which is desirable in some cases, and a more efficient way of dealing with the computed data. We propose two versions of ST-C. In the first version, the smooth solution is extracted by projection from a solution computed with a different temporal representation, typically a discontinuous one. We use a successive projection technique with a small number of temporal NURBS basis functions at each projection, and therefore the extraction can take place as the solution with discontinuous temporal representation is being computed, without storing a large amount of time-history data. This version is not limited to solutions computed with ST techniques. In the second version, the solution with continuous temporal representation is computed directly by using a small number of temporal NURBS basis functions in the variational formulation associated with each time step.

Keywords Space–time techniques · Continuous representation in time · Smooth solution in time · NURBS in time · Successive projection · Direct computation

K. Takizawa
Department of Modern Mechanical Engineering and Waseda Institute for Advanced Study, Waseda University, 1-6-1 Nishi-Waseda, Shinjuku-ku, Tokyo 169-8050, Japan

T. E. Tezduyar (✉)
Department of Mechanical Engineering, Rice University, MS 321, 6100 Main Street, Houston, TX 77005, USA
e-mail: tezduyar@rice.edu

1 Introduction

One of the earliest space–time (ST) computation techniques targeting fluid mechanics problems with moving interfaces is the deforming-spatial-domain/stabilized ST (DSD/SST) method [1–4]. It is a general-purpose moving-mesh technique that serves as core numerical technology in modeling fluid–structure interaction (FSI), fluid–object interaction, fluid–particle interaction, free-surface and multi-fluid flows, and flows with mechanical components in fast, linear or rotational relative motion. It is an alternative to the arbitrary Lagrangian–Eulerian (ALE) finite element formulation [5], which is the most widely used moving-mesh technique, with increased emphasis on FSI in recent years (see, for example, [6–39]). Though less widely used than the ALE formulation, over the past 20 years the DSD/SST method has been applied to some of the most challenging moving-interface problems, including FSI (see, for example, [34,35,40–59] and references therein). Prior to the inception of the DSD/SST formulation, the ST finite element formulations were introduced and tested by other researchers in the context of problems with fixed spatial domains (see [60]).

In the DSD/SST formulation, as it was originally envisioned, the ST computations are carried out for one ST "slab" at a time, where the "slab" is the slice of the ST domain between the time levels n and $n+1$. The basis functions are continuous within a ST slab, but discontinuous from one ST slab to another. The formulation is based on the streamline-upwind/Petrov–Galerkin (SUPG) [61] and pressure-stabilizing/Petrov–Galerkin (PSPG) [1,62] stabilizations. It also includes the "LSIC" (least-squares on incompressibility constraint) stabilization. New versions of the DSD/SST method have been introduced since its inception, including those in [46], which have been serving as the core numerical technology in the majority of the ST

FSI computations carried out in recent years. The most recent DSD/SST method is the ST version [51,63] of the residual-based variational multiscale (VMS) method [64–67]. It was named "DSD/SST-VMST" (i.e. the version with the VMS turbulence model) in [63], which was also called "ST-VMS" in [51]. The original DSD/SST method was named "DSD/SST-SUPS" in [63] (i.e. the version with the SUPG/PSPG stabilization), which was also called "ST-SUPS" in [34].

The ST techniques give us the option of using higher-order basis functions in time, including the NURBS basis functions, which have been used very effectively as spatial basis functions (see [8,12,68,69]). This has positive consequences beyond just increasing the order of accuracy in the computations [51,63,70]. It provides us better accuracy and efficiency in temporal representation of the motion and deformation of the moving interfaces and volume meshes, and better efficiency in remeshing. This has been demonstrated in a number of 3D computations, specifically, flapping-wing aerodynamics [48,52,53,55], separation aerodynamics of spacecraft [57], and wind-turbine aerodynamics [58].

There are some advantages in using a discontinuous temporal representation in ST computations. For a given order of temporal representation, we can reach a higher order accuracy than one would reach with a continuous representation of the same order. When we need to change the spatial discretization (i.e. remesh) between two ST slabs, the temporal discontinuity between the slabs provides a natural framework for that change. There are advantages also in continuous temporal representation. We obtain a smooth solution, NURBS-based when needed, and that is desirable in some cases. We also can deal with the computed data in a more efficient way, because we can represent the data with fewer temporal control points, and that reduces the computer storage cost. These advantages motivated the development of the ST computation techniques with continuous temporal representation (ST-C).

We propose two versions of the continuous temporal representation. In the first version, the continuous representation is extracted by projection from a solution computed with a different temporal representation, typically a discontinuous one. We use a successive-projection technique with a small number of temporal NURBS basis functions at each projection. Because of that, the extraction can take place as the solution with discontinuous temporal representation is being computed, without storing a large amount of time-history data. We note that this version is not limited to solutions computed with ST techniques. For example, they can also be applied to solutions computed with an ALE approach. In the second version, the solution with continuous temporal representation is computed directly from the ST variational formulation associated with each time step. Again, we use a small number of temporal NURBS basis functions.

The first version is described in Sect. 2, and the second version in Sect. 3. Test computations are presented in Sect. 4, and the concluding remarks are given in Sect. 5.

2 Extracting continuous temporal representation from computed data

This is essentially a post-processing method, and can also be seen as a data compression method.

2.1 Least-squares projection for full temporal domain

When we have the complete sequence of computed data, we can project that to a smooth representation, with basis functions that provide us that smooth representation, such as NURBS basis functions. As an example, Fig. 1 shows the goal continuous data ϕ_{C} and its basis functions, where ϑ denotes the parametric temporal coordinate. The projection for each spatial node can be done independently from the other nodes. Consider the time-dependent, typically discontinuous computed data ϕ_{D} for a node. We define the basis functions as T_{C}^{α}, where $\alpha = 0, 1, \ldots$, and the coefficients to be determined in the projection as ϕ^{α}. We use a standard least-squares projection: given ϕ_{D}, find the solution $\phi_{\mathrm{C}} \in \mathcal{S}_{\mathrm{C}}$, such that for all test functions $w_{\mathrm{C}} \in \mathcal{V}_{\mathrm{C}}$:

$$\int_0^T w_{\mathrm{C}} \left(\phi_{\mathrm{C}} - \phi_{\mathrm{D}}\right) \mathrm{d}t = 0, \tag{1}$$

where T represents time period of the computation, and $\mathcal{S}_{\mathrm{C}}$ and $\mathcal{V}_{\mathrm{C}}$ are the solution and test function spaces constructed from the basis functions. This approach requires that we store all the computed data before the projection, and that would

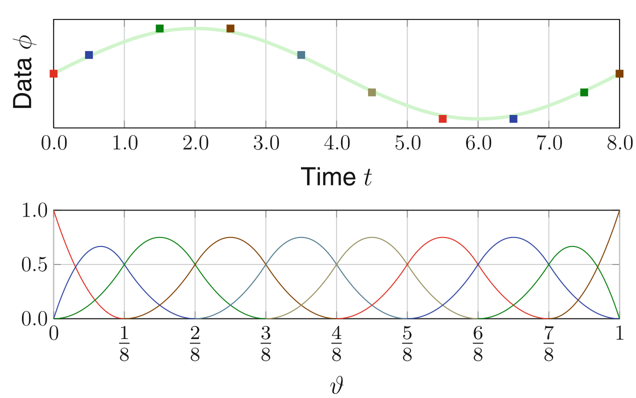

Fig. 1 Continuous solution (*top*) and its basis functions (*bottom*), where ϑ is the parametric coordinate

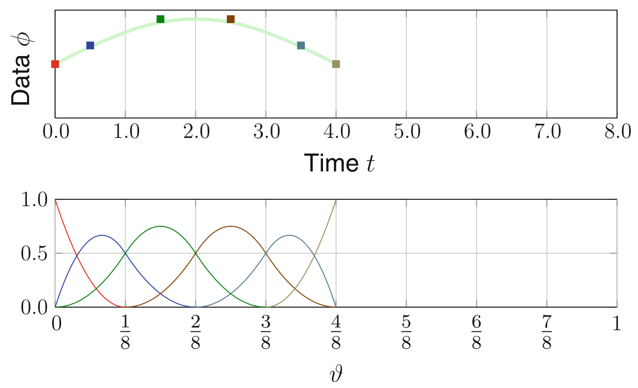

Fig. 2 Continuous solution up to $t_n = 4.0$ (*top*) and its basis functions (*bottom*)

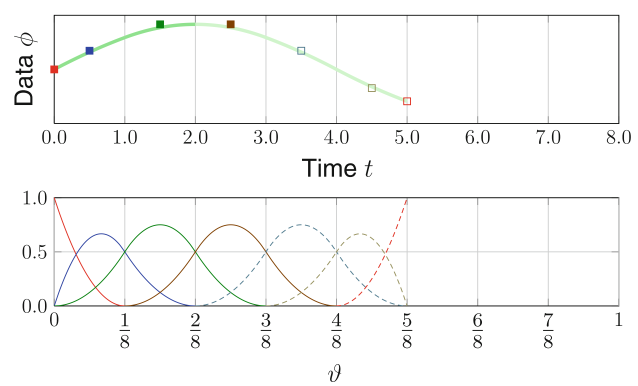

Fig. 3 Continuous solution up to $t_{n+1} = 5.0$ (*top*) and its basis functions (*bottom*). The *bold part* of the *top curve* indicates the part of the solution that does not change. The *empty squares* denote the temporal control values to be determined. The *dashed lines* denote the modified and new basis functions, which correspond to those control values

be a significant computer storage cost when the number of time steps is large.

2.2 Successive-projection technique

In ST-C with the successive-projection technique (ST-C-SPT), we extract the continuous solution shown in Fig. 1 without storing all the computed data.

2.2.1 Special case with quadratic B-splines

To explain the successive nature of the SPT, let us suppose that we have the continuous solution extracted up to $t_n = 4.0$, as shown in Fig. 2. We assume that this continuous solution, which we will call $\overline{\phi}_{\mathrm{C}}$, has already replaced ϕ_{D} up to $t_n = 4.0$. With that, we describe how we extract the continuous solution up to $t_{n+1} = 5.0$, as shown in Fig. 3. With the newly computed data ϕ_{D} between $t_n = 4.0$ and $t_{n+1} = 5.0$, we solve the following projection equation: given ϕ_{D} on $t \in (4.0, 5.0)$, $\overline{\phi}_{\mathrm{C}}$ on $t \in [2.0, 4.0]$, and $\phi_{\mathrm{C}}^{\alpha}$, $\alpha = 2, 3$, find $\phi_{\mathrm{C}} \in \mathcal{S}_{\mathrm{C}}$, such that $\forall w_{\mathrm{C}} \in \mathcal{V}_{\mathrm{C}}$:

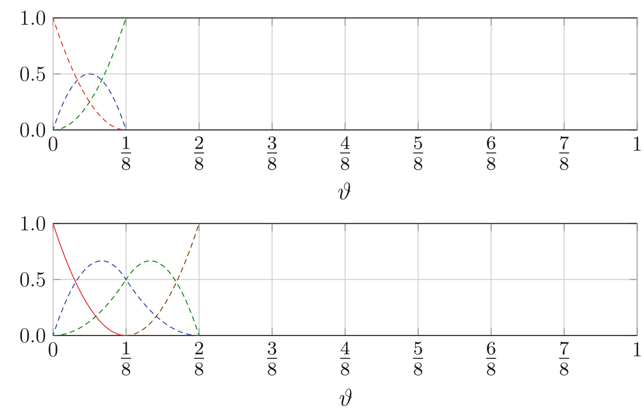

Fig. 4 Basis functions for the initial part (first two steps) of the extraction

$$\int_{2.0}^{4.0} w_{\mathrm{C}} \left(\phi_{\mathrm{C}} - \overline{\phi}_{\mathrm{C}}\right) \mathrm{d}t + \int_{4.0}^{5.0} w_{\mathrm{C}} \left(\phi_{\mathrm{C}} - \phi_{\mathrm{D}}\right) \mathrm{d}t = 0. \qquad (2)$$

We note that Eq. (2) is essentially used for defining the coefficients $\phi_{\mathrm{C}}^{\alpha}$, $\alpha = 4, 5, 6$, which correspond to the basis functions T_{C}^{α}.

We now explain the initial part of the extraction. Figure 4 shows basis functions for the first two steps. In the first step, we calculate the three coefficients to be determined by using the equation

$$\int_{0.0}^{1.0} w_{\mathrm{C}} \left(\phi_{\mathrm{C}} - \phi_{\mathrm{D}}\right) \mathrm{d}t = 0. \qquad (3)$$

In the second step, with the solution $\overline{\phi}_{\mathrm{C}}$ from the first step, and the newly computed data ϕ_{D}, which is defined on the parametric space $\frac{1}{8}$ to $\frac{2}{8}$, we calculate the three new coefficients to be determined by using the equation

$$\int_{0.0}^{1.0} w_{\mathrm{C}} \left(\phi_{\mathrm{C}} - \overline{\phi}_{\mathrm{C}}\right) \mathrm{d}t + \int_{1.0}^{2.0} w_{\mathrm{C}} \left(\phi_{\mathrm{C}} - \phi_{\mathrm{D}}\right) \mathrm{d}t = 0. \qquad (4)$$

For the steps after that, Eq. (2) is used.

2.2.2 General case

Let us suppose that we are using pth-order functions, as shown in Fig. 5. We have the solution $\overline{\phi}_{\mathrm{C}}$ up to t_n and the newly computed data ϕ_{D} between t_n and t_{n+1}. We solve the following projection equation written over $p + 1$ intervals for the $p + 1$ coefficients $\phi_{\mathrm{C}}^{\alpha}$ to be determined: given $\overline{\phi}_{\mathrm{C}}$ and ϕ_{D}, and with p coefficients $\phi_{\mathrm{C}}^{\alpha}$ specified, find $\phi_{\mathrm{C}} \in \mathcal{S}_{\mathrm{C}}$, such that $\forall w_{\mathrm{C}} \in \mathcal{V}_{\mathrm{C}}$:

$$\int_{t_{n-p}}^{t_n} w_{\mathrm{C}} \left(\phi_{\mathrm{C}} - \overline{\phi}_{\mathrm{C}}\right) \mathrm{d}t + \int_{t_n}^{t_{n+1}} w_{\mathrm{C}} \left(\phi_{\mathrm{C}} - \phi_{\mathrm{D}}\right) \mathrm{d}t = 0. \qquad (5)$$

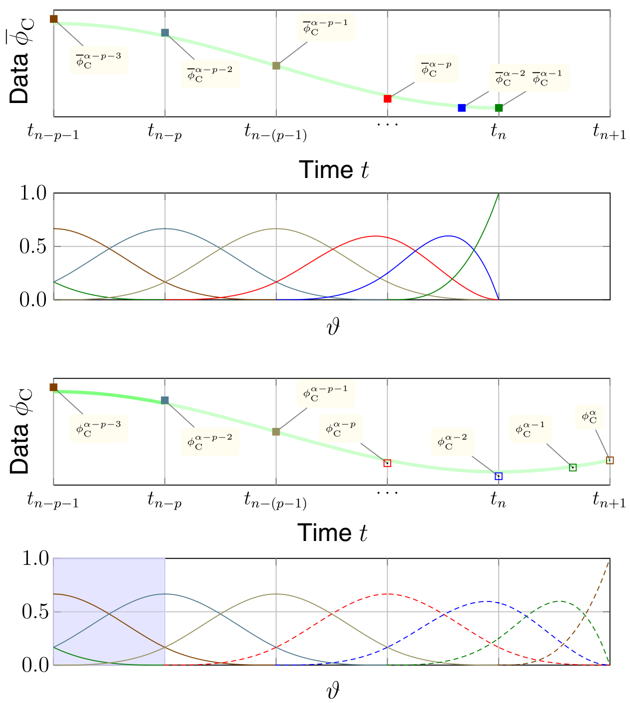

Fig. 5 Continuous solution and basis functions up to t_n (*top two*) and for extraction up to t_{n+1} (*bottom two*). The *bold part* of the *curve* in *third plot* indicates the part of the solution that does not change. The *empty squares* denote the control values to be determined. The *dashed lines* denote the modified and new basis functions, which correspond to those control values

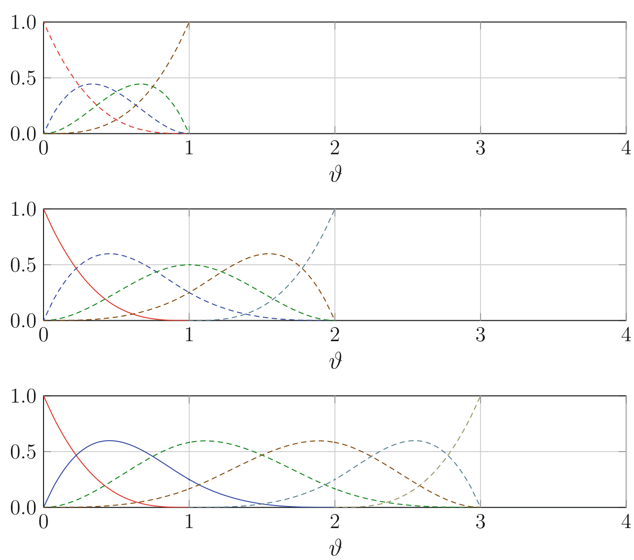

Fig. 6 Basis functions for the initial part (first p steps) of the computation

We again explain the initial part of the extraction. Figure 6 shows basis functions for the first p steps. In the first step there are $p+1$ coefficients to be determined. In the second step, we keep the first coefficient and calculate the remaining $p+1$ coefficients. In the third step we keep also the second coefficient and calculate the remaining $p+1$ coefficients. We keep going this way until we reach step $p+1$, and that is when we switch to Eq. (5). We generalize the extraction procedure as follows:

$$\int_{t_m}^{t_n} w_{\rm C}\left(\phi_{\rm C} - \overline{\phi}_{\rm C}\right) {\rm d}t + \int_{t_n}^{t_{n+1}} w_{\rm C}\left(\phi_{\rm C} - \phi_{\rm D}\right) {\rm d}t = 0, \tag{6}$$

where $m = \max(n-p, 0)$. In summary, the number of unknowns is always $p+1$, the number of specified coefficients is $\min(n, p)$, and the number of intervals of the projection equation is $\min(n+1, p+1)$.

Remark 1 Another way of looking at this, we determine the coefficients corresponding to all basis functions that are nonzero in the last interval. This also means that the basis functions with specified coefficients do not change between the previous and current steps.

2.3 Efficient implementation of the SPT

In general, $\phi_{\rm D}$ could be a solution computed over many time steps in the interval t_n to t_{n+1}; for example, there could be 1,000 steps. We do not need to store such a large amount of computed data to solve Eq. (5).The integration

$$\int_{t_n}^{t_{n+1}} w_{\rm C}\phi_{\rm D}{\rm d}t \tag{7}$$

would be performed at one of those 1,000 time steps at a time.

3 Direct computation of the solution with continuous temporal representation

In ST-C with the direct-computation technique (ST-C-DCT), instead of extracting $\phi_{\rm C}$ from $\phi_{\rm D}$, we compute it directly from the variational formulation. To explain this concept, let us consider an abstract differential equation, $\text{Ł}(\phi) = f$. Then, the counterpart of Eq. (1), before any integration by parts, would be

$$\int_{t_0}^{t_{n+1}} \int_{\Omega_t} w\text{Ł}(\phi){\rm d}\Omega{\rm d}t = \int_{t_0}^{t_{n+1}} \int_{\Omega_t} wf\,{\rm d}\Omega{\rm d}t. \tag{8}$$

Instead of using Eq. (8), we use the counterpart of Eq. (6):

$$\int_{t_m}^{t_{n+1}} \int_{\Omega_t} w\text{Ł}(\phi){\rm d}\Omega{\rm d}t = \int_{t_m}^{t_{n+1}} \int_{\Omega_t} wf\,{\rm d}\Omega{\rm d}t. \tag{9}$$

As before, the number of equations and unknown coefficients is $p+1$, the number of specified coefficients is $\min(n, p)$,

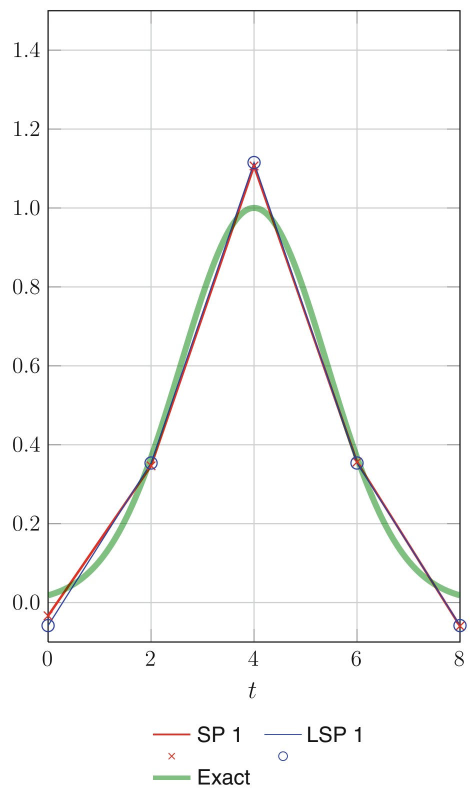

Fig. 7 SP with linear functions, compared with the exact solution and solution obtained with LSP. The *symbols* denote the control values

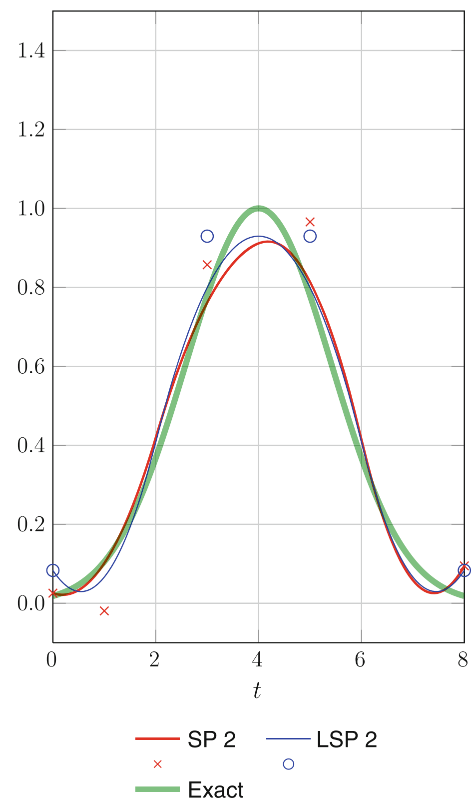

Fig. 8 SP with quadratic B-splines, compared with the exact solution and solution obtained with LSP. The *symbols* denote the control values

and the number of intervals is $\min(n+1, p+1)$. The initial guess for the unknown coefficients of the modified basis functions can be set in such a way that the resulting ϕ_{C} is unchanged until t_n (Bézier extraction technique [71] can be used for determining the new coefficients). The initial guess for the unknown coefficient of the new basis function would be set by taking into account the problem-dependent factors, such as using the same function value or its derivative.

Remark 2 We note that Eq. (9) does not involve the jump term seen in a typical ST formulation, and that is because here the functions are continuous in time. However, if the computation involves a temporal patch boundary because of remeshing (see [48,52]), the jump term would come back.

As an example, let us consider a first-order ordinary differential equation, $\text{Ł}(\phi) = \frac{\mathrm{d}\phi}{\mathrm{d}t}$. The counterpart of Eq. (9), after the integration by parts, is

$$w_{n+1}\phi_{n+1} - \int_{t_m}^{t_{n+1}} \frac{\mathrm{d}w}{\mathrm{d}t}\phi \mathrm{d}t = \int_{t_m}^{t_{n+1}} wf\mathrm{d}t. \tag{10}$$

When the jump term comes back, which would happen at $t = t_0$, we start with

$$w_0^+\left(\phi_0^+ - \phi_0^-\right) + \int_{t_0}^{t_1} w\frac{\mathrm{d}\phi}{\mathrm{d}t}\mathrm{d}t = \int_{t_0}^{t_1} wf\mathrm{d}t, \tag{11}$$

and after the integration by parts we obtain

$$w_1\phi_1 - w_0^+\phi_0^- - \int_{t_0}^{t_1} \frac{\mathrm{d}w}{\mathrm{d}t}\phi\mathrm{d}t = \int_{t_0}^{t_1} wf\mathrm{d}t. \tag{12}$$

Here ϕ_0^- is the value prior to remeshing.

An alternative to Eq. (9) would be

$$\int_{t_m}^{t_n}\int_{\Omega_t} w\text{Ł}(\phi)\mathrm{d}\Omega\mathrm{d}t - \int_{t_m}^{t_n}\int_{\Omega_t} w\text{Ł}(\overline{\phi})\mathrm{d}\Omega\mathrm{d}t + \int_{t_n}^{t_{n+1}}\int_{\Omega_t} w\text{Ł}(\phi)\mathrm{d}\Omega\mathrm{d}t = \int_{t_n}^{t_{n+1}}\int_{\Omega_t} wf\mathrm{d}\Omega\mathrm{d}t, \tag{13}$$

where $\overline{\phi}$, similar to $\overline{\phi}_{\mathrm{C}}$, is the solution up to t_n. While this alternative form is a closer extension of Eq. (6), we prefer Eq. (9) because we see it as a more direct source.

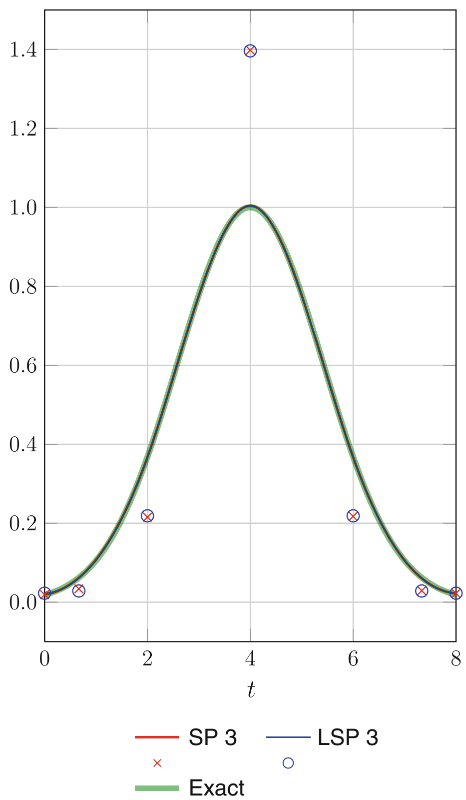

Fig. 9 SP with cubic B-splines, compared with the exact solution and solution obtained with LSP. The *symbols* denote the control values

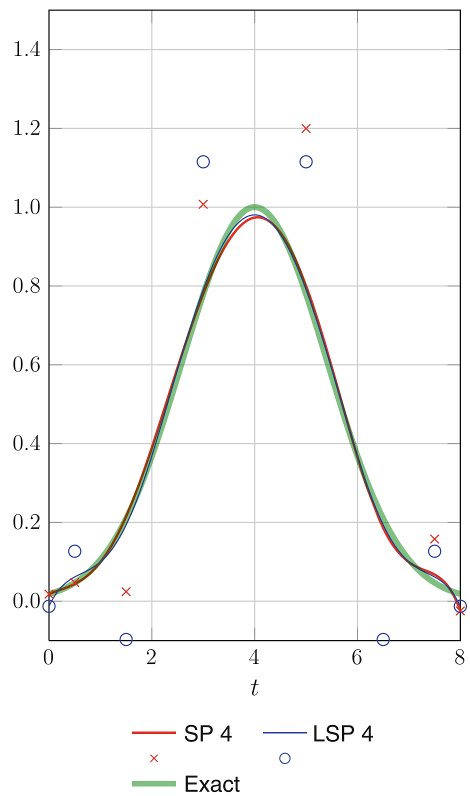

Fig. 10 SP with quartic B-splines, compared with the exact solution and solution obtained with LSP. The *symbols* denote the control values

4 Test calculations

We carry out test calculations to show how ST-C-SPT works. In place of ϕ_{D}, we use the following function:

$$\phi(t) = \exp\left(-\frac{(t-4)^2}{4}\right), \tag{14}$$

where $0 \leq t \leq 8$. We test SP with linear, quadratic, cubic, and quartic B-splines. Figures 7,8,9, and 10 show the test results for $\Delta t = 2.0$, together with the exact solution and solution obtained with least-squares projection (LSP).

Figure 11 shows the "L_2 Error," defined as

$$L_2 \text{ error} = \left(\frac{1}{8}\int_0^8 \left(\frac{\phi^h(t)-\phi(t)}{\phi(t)}\right)^2 \mathrm{d}t\right)^{\frac{1}{2}}. \tag{15}$$

Essentially both the SP and LSP have the same order of accuracy; nth-order functions result in $(n+1)$th-order accuracy. Figure 12 shows the "L_2 Symmetry Error," defined as

$$L_2 \text{ symmetry error} = \left(\frac{1}{8}\int_0^8 \left(\phi^h(t)-\phi^h(8-t)\right)^2 \mathrm{d}t\right)^{\frac{1}{2}}. \tag{16}$$

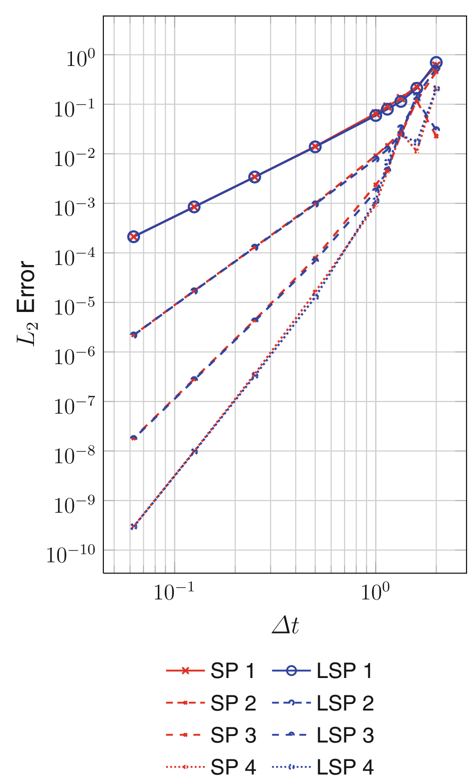

Fig. 11 L_2 error

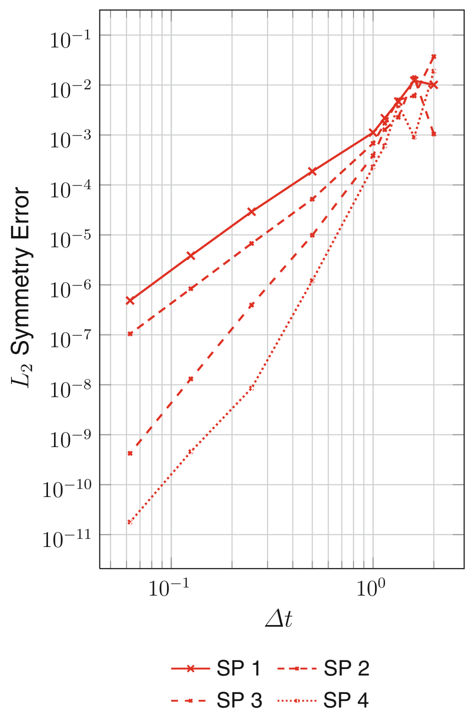

Fig. 12 L_2 symmetry error

We note that for LSP this error is zero. The figure shows that the asymmetry is decreasing quickly with increasing order of the basis functions.

5 Concluding remarks

We have introduced ST computation techniques with continuous representation in time (ST-C), using temporal NURBS basis functions. With ST-C, we can have a temporally smooth solution, which is sometimes desirable. We also deal with the computed data in a more efficient way, because we can represent the data with fewer temporal control points, resulting in reduced computer storage cost. We have introduced two versions of ST-C. In the first version, the continuous representation is extracted by projection from a solution already computed, typically a discontinuous one, but not necessarily limited to solutions computed with ST techniques. Because we use a SPT with a small number of temporal NURBS basis functions at each projection, the extraction can take place as the solution with discontinuous temporal representation is being computed, without storing a large amount of time-history data. We call the first version ST-C-SPT. In the second version, the solution with continuous temporal representation is obtained by a direct computation technique (DCT), from the ST variational formulation associated with each time step. Again, this can be done with a small number of temporal NURBS basis functions, resulting in efficient computation and storage. We call the second version ST-C-DCT. The test calculations with ST-C-SPT show that the technique works quite effectively.

Acknowledgments This work was supported in part by the Rice–Waseda research agreement (first author). It was also supported in part by ARO Grant W911NF-12-1-0162 (second author).

References

1. Tezduyar TE (1992) Stabilized finite element formulations for incompressible flow computations. Adv Appl Mech 28:1–44. doi:10.1016/S0065-2156(08)70153-4
2. Tezduyar TE, Behr M, Liou J (1992) A new strategy for finite element computations involving moving boundaries and interfaces—the deforming-spatial-domain/space–time procedure: I. The concept and the preliminary numerical tests. Comput Methods Appl Mech Eng 94:339–351. doi:10.1016/0045-7825(92)90059-S
3. Tezduyar TE, Behr M, Mittal S, Liou J (1992) A new strategy for finite element computations involving moving boundaries and interfaces—the deforming-spatial-domain/space–time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders. Comput Methods Appl Mech Eng 94:353–371. doi:10.1016/0045-7825(92)90060-W
4. Tezduyar TE (2003) Computation of moving boundaries and interfaces and stabilization parameters. Int J Numer Methods Fluids 43:555–575. doi:10.1002/fld.505
5. Hughes TJR, Liu WK, Zimmermann TK (1981) Lagrangian–Eulerian finite element formulation for incompressible viscous flows. Comput Methods Appl Mech Eng 29:329–349
6. Ohayon R (2001) Reduced symmetric models for modal analysis of internal structural-acoustic and hydroelastic-sloshing systems. Comput Methods Appl Mech Eng 190:3009–3019
7. van Brummelen EH, de Borst R (2005) On the nonnormality of subiteration for a fluid–structure interaction problem. SIAM J Sci Comput 27:599–621
8. Bazilevs Y, Calo VM, Zhang Y, Hughes TJR (2006) Isogeometric fluid–structure interaction analysis with applications to arterial blood flow. Comput Mech 38:310–322
9. Khurram RA, Masud A (2006) A multiscale/stabilized formulation of the incompressible Navier–Stokes equations for moving boundary flows and fluid–structure interaction. Comput Mech 38:403–416
10. Lohner R, Cebral JR, Yang C, Baum JD, Mestreau EL, Soto O (2006) Extending the range of applicability of the loose coupling approach for FSI simulations. In: Bungartz H-J, Schafer M (eds) Fluid–structure interaction, vol 53 of lecture notes in computational science and engineering. Springer, Heidelberg, pp 82–100
11. Bletzinger K-U, Wuchner R, Kupzok A (2006) Algorithmic treatment of shells and free form-membranes in FSI. In: Bungartz H-J, Schafer M (eds) Fluid–structure interaction, vol 53 of lecture notes in computational science and engineering. Springer, Heidelberg, pp 336–355
12. Bazilevs Y, Calo VM, Hughes TJR, Zhang Y (2008) Isogeometric fluid–structure interaction: theory, algorithms, and computations. Comput Mech 43:3–37
13. Dettmer WG, Peric D (2008) On the coupling between fluid flow and mesh motion in the modelling of fluid–structure interaction. Comput Mech 43:81–90
14. Bazilevs Y, Gohean JR, Hughes TJR, Moser RD, Zhang Y (2009) Patient-specific isogeometric fluid–structure interaction analysis of thoracic aortic blood flow due to implantation of the Jarvik (2000) left ventricular assist device. Comput Methods Appl Mech Eng 198:3534–3550

15. Bazilevs Y, Hsu M-C, Benson D, Sankaran S, Marsden A (2009) Computational fluid–structure interaction: methods and application to a total cavopulmonary connection. Comput Mech 45:77–89
16. Calderer R, Masud A (2010) A multiscale stabilized ALE formulation for incompressible flows with moving boundaries. Comput Mech 46:185–197
17. Bazilevs Y, Hsu M-C, Zhang Y, Wang W, Liang X, Kvamsdal T, Brekken R, Isaksen J (2010) A fully-coupled fluid–structure interaction simulation of cerebral aneurysms. Comput Mech 46:3–16
18. Bazilevs Y, Hsu M-C, Zhang Y, Wang W, Kvamsdal T, Hentschel S, Isaksen J (2010) Computational fluid–structure interaction: methods and application to cerebral aneurysms. Biomech Model Mechanobiol 9:481–498
19. Bazilevs Y, Hsu M-C, Akkerman I, Wright S, Takizawa K, Henicke B, Spielman T, Tezduyar TE (2011) 3D simulation of wind turbine rotors at full scale. Part I: geometry modeling and aerodynamics. Int J Numer Methods Fluids 65:207–235. doi:10.1002/fld.2400
20. Bazilevs Y, Hsu M-C, Kiendl J, Wüchner R, Bletzinger K-U (2011) 3D simulation of wind turbine rotors at full scale. Part II: fluid–structure interaction modeling with composite blades. Int J Numer Methods Fluids 65:236–253
21. Akkerman I, Bazilevs Y, Kees CE, Farthing MW (2011) Isogeometric analysis of free-surface flow. J Comput Phys 230:4137–4152
22. Hsu M-C, Bazilevs Y (2011) Blood vessel tissue prestress modeling for vascular fluid–structure interaction simulations. Finite Elements Anal Design 47:593–599
23. Nagaoka S, Nakabayashi Y, Yagawa G, Kim YJ (2011) Accurate fluid–structure interaction computations using elements without mid-side nodes. Comput Mech 48:269–276. doi:10.1007/s00466-011-0620-7
24. Bazilevs Y, Hsu M-C, Takizawa K, Tezduyar TE (2012) ALE-VMS and ST-VMS methods for computer modeling of wind–turbine rotor aerodynamics and fluid–structure interaction. Math Models Methods Appl Sci 22:1230002. doi:10.1142/S0218202512300025
25. Akkerman I, Bazilevs Y, Benson DJ, Farthing MW, Kees CE (2012) Free-surface flow and fluid–object interaction modeling with emphasis on ship hydrodynamics. J Appl Mech 79:010905
26. Hsu M-C, Akkerman I, Bazilevs Y (2012) Wind turbine aerodynamics using ALE-VMS: validation and role of weakly enforced boundary conditions. Comput Mech 50:499–511
27. Hsu M-C, Bazilevs Y (2012) Fluid–structure interaction modeling of wind turbines: simulating the full machine. Comput Mech 50:821–833
28. Akkerman I, Dunaway J, Kvandal J, Spinks J, Bazilevs Y (2012) Toward free-surface modeling of planing vessels: simulation of the Fridsma hull using ALE-VMS. Comput Mech 50:719–727
29. Minami S, Kawai H, Yoshimura S (2012) Parallel BDD-based monolithic approach for acoustic fluid–structure interaction. Comput Mech 50:707–718
30. Miras T, Schotte J-S, Ohayon R (2012) Energy approach for static and linearized dynamic studies of elastic structures containing incompressible liquids with capillarity: a theoretical formulation. Comput Mech 50:729–741
31. van Opstal TM, van Brummelen EH, de Borst R, Lewis MR (2012) A finite-element/boundary-element method for large-displacement fluid–structure interaction. Comput Mech 50:779–788
32. Yao JY, Liu GR, Narmoneva DA, Hinton RB, Zhang Z-Q (2012) Immersed smoothed finite element method for fluid–structure interaction simulation of aortic valves. Comput Mech 50:789–804
33. Larese A, Rossi R, Onate E, Idelsohn SR (2012) A coupled PFEM–Eulerian approach for the solution of porous FSI problems. Comput Mech 50:805–819
34. Bazilevs Y, Takizawa K, Tezduyar TE (2013) Computational fluid–structure interaction: methods and applications. Wiley, New York
35. Bazilevs Y, Takizawa K, Tezduyar TE (2013) Challenges and directions in computational fluid–structure interaction. Math Models Methods Appl Sci 23:215–221. doi:10.1142/S0218202513400010
36. Korobenko A, Hsu M-C, Akkerman I, Tippmann J, Bazilevs Y (2013) Structural mechanics modeling and FSI simulation of wind turbines. Math Models Methods Appl Sci 23:249–272
37. Yao JY, Liu GR, Qian D, Chen CL, Xu GX (2013) A moving-mesh gradient smoothing method for compressible CFD problems. Math Models Methods Appl Sci 23:273–305
38. Kamran K, Rossi R, Onate E, Idelsohn SR (2013) A compressible Lagrangian framework for modeling the fluid–structure interaction in the underwater implosion of an aluminum cylinder. Math Models Methods Appl Sci 23:339–367
39. Hsu M-C, Akkerman I, Bazilevs Y (2013) Finite element simulation of wind turbine aerodynamics: validation study using NREL phase VI experiment. Wind Energy. doi:10.1002/we.1599
40. Tezduyar T, Aliabadi S, Behr M, Johnson A, Mittal S (1993) Parallel finite-element computation of 3D flows. Computer 26:27–36. doi:10.1109/2.237441
41. Tezduyar TE, Aliabadi SK, Behr M, Mittal S (1994) Massively parallel finite element simulation of compressible and incompressible flows. Comput Methods Appl Mech Eng 119:157–177. doi:10.1016/0045-7825(94)00082-4
42. Tezduyar T, Aliabadi S, Behr M, Johnson A, Kalro V, Litke M (1996) Flow simulation and high performance computing. Comput Mech 18:397–412. doi:10.1007/BF00350249
43. Tezduyar TE (1999) CFD methods for three-dimensional computation of complex flow problems. J Wind Eng Ind Aerodyn 81:97–116. doi:10.1016/S0167-6105(99)00011-2
44. Tezduyar T, Osawa Y (1999) Methods for parallel computation of complex flow problems. Parallel Comput 25:2039–2066. doi:10.1016/S0167-8191(99)00080-0
45. Tezduyar TE (2001) Finite element methods for flow problems with moving boundaries and interfaces. Arch Comput Methods Eng 8:83–130. doi:10.1007/BF02897870
46. Tezduyar TE, Sathe S (2007) Modeling of fluid–structure interactions with the space–time finite elements: solution techniques. Int J Numer Methods Fluids 54:855–900. doi:10.1002/fld.1430
47. Tezduyar TE, Takizawa K, Brummer T, Chen PR (2011) Space–time fluid–structure interaction modeling of patient-specific cerebral aneurysms. Int J Numer Methods Biomed Eng 27:1665–1710. doi:10.1002/cnm.1433
48. Takizawa K, Henicke B, Puntel A, Spielman T, Tezduyar TE (2012) Space–time computational techniques for the aerodynamics of flapping wings. J Appl Mech 79:010903. doi:10.1115/1.4005073
49. Takizawa K, Tezduyar TE (2012) Computational methods for parachute fluid–structure interactions. Arch Comput Methods Eng 19:125–169. doi:10.1007/s11831-012-9070-4
50. Takizawa K, Bazilevs Y, Tezduyar TE (2012) Space–time and ALE-VMS techniques for patient-specific cardiovascular fluid–structure interaction modeling. Arch Comput Methods Eng 19:171–225. doi:10.1007/s11831-012-9071-3
51. Takizawa K, Tezduyar TE (2012) Space–time fluid–structure interaction methods. Math Models Methods Appl Sci 22:1230001. doi:10.1142/S0218202512300013
52. Takizawa K, Henicke B, Puntel A, Kostov N, Tezduyar TE (2012) Space–time techniques for computational aerodynamics modeling of flapping wings of an actual locust. Comput Mech 50:743–760. doi:10.1007/s00466-012-0759-x
53. Takizawa K, Kostov N, Puntel A, Henicke B, Tezduyar TE (2012) Space–time computational analysis of bio-inspired flapping-wing aerodynamics of a micro aerial vehicle. Comput Mech 50:761–778. doi:10.1007/s00466-012-0758-y
54. Takizawa K, Fritze M, Montes D, Spielman T, Tezduyar TE (2012) Fluid–structure interaction modeling of ringsail parachutes with

disreefing and modified geometric porosity. Comput Mech 50:835–854. doi:10.1007/s00466-012-0761-3
55. Takizawa K, Henicke B, Puntel A, Kostov N, Tezduyar TE (2012) Computer modeling techniques for flapping-wing aerodynamics of a locust. Comput Fluids. doi:10.1016/j.compfluid.2012.11.008
56. Takizawa K, Tezduyar TE (2012) Bringing them down safely. Mech Eng 134:34–37
57. Takizawa K, Montes D, Fritze M, McIntyre S, Boben J, Tezduyar TE (2013) Methods for FSI modeling of spacecraft parachute dynamics and cover separation. Math Models Methods Appl Sci 23:307–338. doi:10.1142/S0218202513400058
58. Takizawa K, Tezduyar TE, McIntyre S, Kostov N, Kolesar R, Habluetzel C (2013) Space–time VMS computation of wind-turbine rotor and tower aerodynamics. Comput Mech. doi:10.1007/s00466-013-0888-x
59. Takizawa K, Tezduyar TE, Boben J, Kostov N, Boswell C, Buscher A (2013) Fluid–structure interaction modeling of clusters of spacecraft parachutes with modified geometric porosity. Comput Mech. doi:10.1007/s00466-013-0880-5
60. Hughes TJR, Hulbert GM (1988) Space–time finite element methods for elastodynamics: formulations and error estimates. Comput Methods Appl Mech Eng 66:339–363
61. Brooks AN, Hughes TJR (1982) Streamline upwind/Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations. Comput Methods Appl Mech Eng 32:199–259
62. Tezduyar TE, Mittal S, Ray SE, Shih R (1992) Incompressible flow computations with stabilized bilinear and linear equal-order-interpolation velocity-pressure elements. Comput Methods Appl Mech Eng 95:221–242. doi:10.1016/0045-7825(92)90141-6
63. Takizawa K, Tezduyar TE (2011) Multiscale space–time fluid–structure interaction techniques. Comput Mech 48:247–267. doi:10.1007/s00466-011-0571-z
64. Hughes TJR (1995) Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles, and the origins of stabilized methods. Comput Methods Appl Mech Eng 127:387–401
65. Hughes TJR, Oberai AA, Mazzei L (2001) Large eddy simulation of turbulent channel flows by the variational multiscale method. Phys Fluids 13:1784–1799
66. Bazilevs Y, Calo VM, Cottrell JA, Hughes TJR, Reali A, Scovazzi G (2007) Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. Comput Methods Appl Mech Eng 197:173–201
67. Bazilevs Y, Akkerman I (2010) Large eddy simulation of turbulent Taylor–Couette flow using isogeometric analysis and the residual-based variational multiscale method. J Comput Phys 229:3402–3414
68. Hughes TJR, Cottrell JA, Bazilevs Y (2005) Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. Comput Methods Appl Mech Eng 194:4135–4195
69. Bazilevs Y, Hughes TJR (2008) NURBS-based isogeometric analysis for the computation of flows about rotating components. Comput Mech 43:143–150
70. Takizawa K, Wright S, Moorman C, Tezduyar TE (2011) Fluid–structure interaction modeling of parachute clusters. Int J Numer Methods Fluids 65:286–307. doi:10.1002/fld.2359
71. Borden MJ, Scott MA, Evans JA, Hughes TJR (2011) Isogeometric finite element data structures based on Bézier extraction of NURBS. Int J Numer Methods Eng 87:15–47