

TTSTOOLS

MATLAB[®] FUNCTIONS FOR TENSOR-VALUED TIME-SERIES PROCESSING

João P. Hespanha

November 11, 2014

Abstract

This toolbox provides functions to process tensor-valued time series, including differentiations, integration, resampling, solution of ODEs, etc. These functions can be applied to numerical values or to TensCalcTools Symbolic Tensor-Valued Expressions (STVEs).

Contents

1	Tensor-Valued Time Series	1
2	Functions provided by TTSTools	3
2.1	Signal processing functions	3
tsDerivative		3
tsDerivative2		4
tsDot		4
tsCross		4
tsQdot		4
tsQdotStar		4
tsRotation		5
tsRotationT		5
2.2	Criteria	5
tsIntegral		5
2.3	Dynamical systems functions	6
tsODEzho		6
tsCLTIzho		6
tsDLTI		6

1 Tensor-Valued Time Series

Tensors are essentially multi-dimensional arrays. Specifically, an α -index *tensor* is an array in $\mathbb{R}^{n_1 \times n_2 \times \dots \times n_\alpha}$ where α is an integer in $\mathbb{Z}_{\geq 0}$. By convention, the case $\alpha = 0$ corresponds to a *scalar* in \mathbb{R} . We use the terminology *vector* and *matrix* for the cases $\alpha = 1$ and $\alpha = 2$, respectively. The integer α is called the *index*

of the tensor and the vector of integers $[n_1, n_2, \dots, n_\alpha]$ (possibly empty for $\alpha = 0$) is called the *dimension* of the tensor.

A *tensor-valued time series* (TVTS) is a sampled-based representation of a time-varying tensor, i.e., a function $F : \mathbb{R} \rightarrow \mathbb{R}^{n_1 \times n_2 \times \dots \times n_\alpha}$. A TVTS represents F through a pair (X, T) where $T \in \mathbb{R}^{n_t}$ is a vector of sample times and $X \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_\alpha \times n_t}$ is an $\alpha + 1$ -*index tensor* with the understanding that

$$F(T_i) = [X_{i_1, i_2, \dots, i_\alpha, i}]_{i_1=1, i_2=1, \dots, i_\alpha=1}^{i_1=n_1, i_2=n_2, \dots, i_\alpha=n_\alpha}, \quad \forall i \in \{1, 2, \dots, n_t\},$$

i.e., the first α indices of X represent the value of F and the last index represents time.

2 Functions provided by TTSTools

2.1 Signal processing functions

These functions take one or two TVTS signals as input and operate on them to produce a TVTS output.

tsDerivative

```
1 [dx,ts]=tsDerivative(x,ts)
2 [dx,ts]=tsDerivative(x,ts,invDts,invD2ts)
```

This function returns a TVTS (dx,ts) that represents the time derivative of the input TVTS (x,ts). The time derivative is computed assuming that the input time-series is piecewise quadratic.

The input parameters are:

1. x : matrix representing the values of the TVTS at the sampling times. Can be
 - (a) $n \times N$ double matrix, or
 - (b) $n \times N$ Tcalculus matrix.
2. ts : vector/scalar representing the sampling times of the x TVTS. Can be
 - (a) $N \times 1$ double or vector of samplig time;
 - (b) 1×1 double scalar with the sampling interval;
 - (c) N Tcalculus vector of samplig time; or
 - (d) Tcalculus scalar with the (fixed) sampling interval.
3. $invDts$: (optional) vector/scalar representing the inverse of the sampling intervals. Can be
 - (a) $N - 1$ Tcalculus vector of inverses of samplig intervals, i.e.,

```
3 invDts = 1./(ts(2:end)-ts(1:end-1));
```
 - (b) Tcalculus scalar with the inverse of the (fixed) sampling interval, i.e., $invDts=1/ts$.

This parameter is only needed when ts is of type Tcalculus. By including this variable, the output does not include divisions and is therefore more “friendly” for optimizations.

4. $invD2ts$: $(N - 2)$ Tcalculus vector with the inverses of the sampling 2-intervals, i.e.,

```
4 invD2ts = 1./(ts(3:end)-ts(1:end-2))
```

This parameter is only needed when ts is of type Tcalculus. By including this variable, the output does not include divisions and is therefore more “friendly” for optimizations.

The output parameters are:

1. dx : matrix representing the values of the derivative of x the sampling times. Will be
 - (a) $n \times N$ double matrix, or

(b) $n \times N$ Tcalculus matrix.

2. **ts**: vector/scalar representing the sampling times of the **dx** TVTS and replicates exactly the input **ts**.

1

tsDerivative2

```
5 [ddx,ts]=tsDerivative2(x,ts)
```

This function returns a TVTS (**ddx,ts**) that represents the second time derivative of the input TVTS (**x,ts**). The output sampling times **ts** are equal to the input sampling times **ts** and therefore the size of derivative **dx** is equal to the size of the input **x**. The time derivatives are computed assuming that the input time-series is piecewise quadratic.

tsDot

```
6 [y,ts]=tsDot(x1,x2,ts)
```

This function returns a scalar-valued time-series (**y,ts**) that represents the dot product of two n-vector time-series (**x1,ts**) and (**x2,ts**):

$$y = x1' \cdot x2$$

The size of the output **y** is equal to the size of **ts**.

tsCross

```
7 [y,ts]=tsCross(x1,x2,ts)
```

This function returns a 3-vector time-series (**y,ts**) that represents the cross product of two 3-vector time-series (**x1,ts**) and (**x2,ts**):

$$y = \text{cross}(x1, x2)$$

The size of the output **y** is equal to the size of the inputs **x1** and **x2**.

tsQdot

```
8 [y,ts]=tsQdot(q1,q2,ts)
```

This function returns a 4-vector time-series (**y,ts**) that represents the product of two quaternions (**q1,ts**), (**q2,ts**):

$$y = q1 \cdot q2$$

The size of the output **y** is equal to the size of the inputs **x1** and **x2**. However, if either (**q1,ts**) or (**q2,ts**) is a 3-vector time series, then the corresponding input quaternion is assumed pure, but the output quaternion is always a 4-vector time series.

¹João: All remining functions need to be updated.

tsQdotStar

```
9 [y,ts]=tsQdotStar(q1,q2,ts)
```

This function returns a 4-vector time-series (y,ts) that represents the product of two quaternions $(q1,ts)*, (q2,ts)$:

$$y = q1^* \cdot q2$$

The size of the output y is equal to the size of the inputs $x1$ and $x2$. However, if either $(q1,ts)$ or $(q2,ts)$ is a 3-vector time series, then the corresponding input quaternion is assumed pure, but the output quaternion is always a 4-vector time series.

tsRotation

```
10 [y,ts]=tsRotation(q,x,ts)
```

This function returns a 3-vector time-series (y,ts) that represents the rotation of a 3-vector time-series (x,ts) by a 4-vector time-series (q,ts) representing a quaternion:

$$y = q \cdot x \cdot q^*$$

The size of the output y is equal to the size of the input x .

tsRotationT

```
11 [y,ts]=tsRotationT(q,x,ts)
```

This function returns a 3-vector time-series (y,ts) that represents the rotation of a 3-vector time-series (x,ts) by a 4-vector time-series $(q,ts)*$ representing a quaternion:

$$y = q^* \cdot x \cdot q$$

The size of the output y is equal to the size of the input x .

2.2 Criteria

The following functions take a one or two TVTS signals as input and produce a scalar that determines a particular property of the TVTS. Typically, they are used to define optimization criteria.

tsIntegral

```
12 y=tsIntegral(x,ts)
```

This function returns a tensor y that represents the time integral of the input TVTS (x,ts) . The size of the integral y is equal to the size of the input x with the last (time) dimension removed. The integral is computed assuming that the input time-series is piecewise quadratic.

2.3 Dynamical systems functions

These functions produce TensCalcTools constraints that encode the solution dynamical systems modeled by differential and difference equations.

tsODEzho

13	<code>constraint=tsODE(x,u,d,ts,fun)</code>
----	---

ZHO on input u but no ZHO on input d

$$\dot{x} = f(x, u, d, t)$$

`fun(x,u,d,t)`

tsCLTIzho

14	<code>constraint=tsCLTI(x,u,ts,A,B)</code>
----	--

tsDLTI

15	<code>constraint=tsCLTI(x,u,ts,A,B)</code>
----	--
