# Time integration of Tree Tensor Networks

Gianluca Ceruti, EPF Lausanne.
(based on joint work with Ch. Lubich and H. Walach)

Napoli, 15 Febbraio 2022.

# Problem

We consider

$$\dot{A}(t) = F(t, A(t)), \quad A(t_0) = A_0 \in \mathbb{C}^{n_1 \times \cdots \times n_d}$$

arising from, e.g.

- discrete Schrödinger equation ($d = 3N$particles).
- discrete kinetic equations ($d = 6$).

Direct computational treatment is infeasible for large $d$ and/or $n_i$.

We start from the real case $d = 2$.

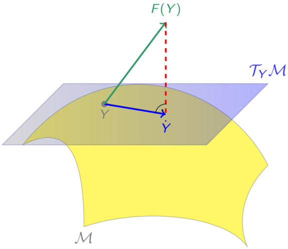
# Dynamical low-rank approximation

Given a matrix differential equation

$$\dot{A}(t) = F(t, A(t)), \quad A(t_0) = A_0 \in \mathbb{R}^{m \times n}$$

we aim to approximate $A(t) \approx Y(t) \in \mathcal{M}_r$ (rank $r$) by requiring

$$\dot{Y}(t) \in \mathcal{T}_{Y(t)}\mathcal{M}_r \quad \text{such that} \quad \|\dot{Y}(t) - F(t, Y(t))\| = \min!$$




---

*Koch, Lubich 2007*

# Tangent space projection

Recalling

$$\dot{Y}(t) \in \mathcal{T}_{Y(t)}\mathcal{M}_r \quad \text{such that} \quad \|\dot{Y}(t) - F(t, Y(t))\| = \min!$$

it is equivalent to

$$\dot{Y} = \mathrm{P}(Y)F(t, Y),$$

where $P(Y)$ is the orthogonal projection onto the tangent space at $Y = USV^\top$ given by

$$P(Y)Z = ZVV^T - UU^TZVV^T + UU^TZ.$$

# Matrix projector-splitting integrator

Idea: Split $P(Y)$ into its three parts, solve separately.

Efficiently implementable integrator such that

+ Reproduces rank-$r$ matrices exactly.
+ Robust error bound (independent of small singular values).
− Backward substep (problematic for dissipative problems).

---

*Lubich, Oseledets 2014*
*Kieri, Lubich, Walach 2016*

# From matrices to tree tensor networks

Low-rank matrices

$\downarrow$

Tucker tensors

$\downarrow$

Tree tensor networks

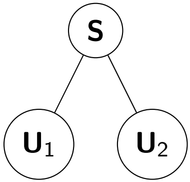
# Low-rank matrices

Let $\mathbf{Y} \in \mathbb{R}^{n_1 \times n_2}$ be a matrix of multilinear rank $(r_1, r_2)$

$$\mathbf{Y} = \mathbf{U}_1 \, \mathbf{S} \, \mathbf{U}_2^\top \; \in \mathbb{R}^{n_1 \times n_2}$$

where

$$\mathbf{U}_i \in \mathbb{R}^{n_i \times r_i} \qquad \forall i = 1, 2,$$
$$\mathbf{S} \in \mathbb{R}^{r_1 \times r_2} \; .$$



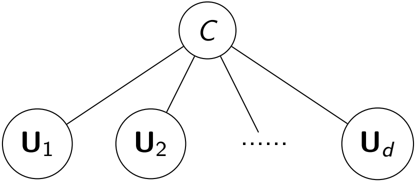
# Tucker tensors

Let $Y \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ be a tensor of multilinear rank $(r_1, \ldots, r_d)$

$$Y = C \bigtimes_{i=1}^{d} \mathbf{U}_i \ \in \mathbb{R}^{n_1 \times \cdots \times n_d}$$

where

$$\mathbf{U}_i \in \mathbb{R}^{n_i \times r_i} \qquad \forall i = 1, \ldots, d,$$
$$C \in \mathbb{R}^{r_1 \times \cdots \times r_d} \quad .$$



# Tucker Integrators

Starting from $Y^0 = C^0 \, \mathsf{X}_{i=1}^{d} \, \mathbf{U}_i^0$ :

Set $C_0^0 = C^0$.

For $i = 1, \ldots, d$, update $\mathbf{U}_i^0 \to \mathbf{U}_i^1$ and modify $C_{i-1}^0 \to C_i^0$.

Update $C_d^0 \to C^1$.

After one time step, this yields $Y^1 = C^1 \, \mathsf{X}_{i=1}^{d} \, \mathbf{U}_i^1$ .

---

*Lubich 2015*

*Lubich, Vandereycken, Walach 2018*

# Curse of the dimensionality of the rank

Let $C \in \mathbb{R}^{r_1 \times \cdots \times r_d}$ and $\mathbf{r} := \max(r_1, r_2, \ldots, r_d)$,

$$\text{size}\ (C) = \mathbf{r}^d\ .$$

The size of the core tensor grows *exponentially with* $d$.

# **Tree** tensor network - Preparation: Graphical representation

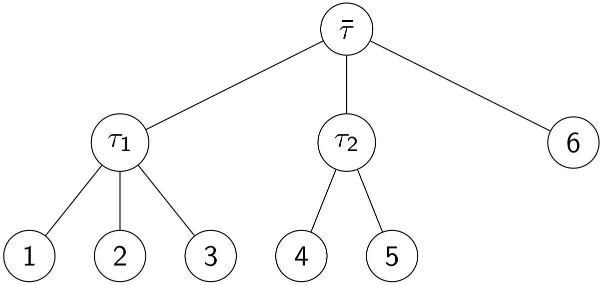



Figure: Graphical representation of a tree $\bar{\tau}$ with three subtrees and set of leaves $\mathcal{L} = \{1, 2, 3, 4, 5, 6\}$.

# Tree **tensor network** - Preparation: Graphical representation

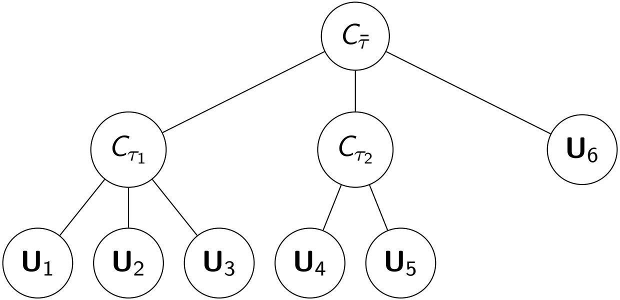
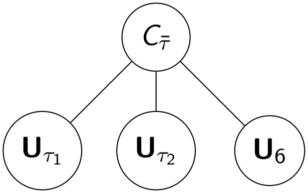
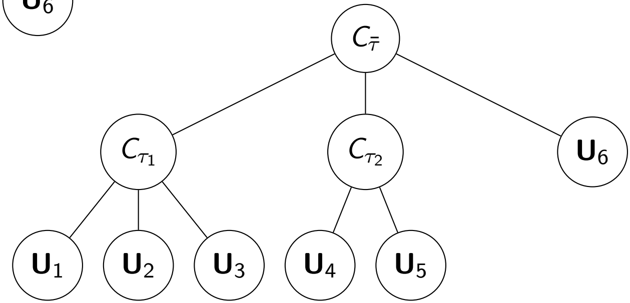



Figure: Graphical representation of a tree tensor network with the set of leaves $\mathcal{L} = \{1, 2, 3, 4, 5, 6\}$.

# Tree tensor network - Definition: Graphical representation



# Tree tensor network - Definition

## Definition (Tree tensor network)

For a given tree $\bar{\tau} \in \mathcal{T}$ and basis matrices $\mathbf{U}_\ell$ and connection tensors $C_\tau$ as described above, we recursively define a tensor $X_{\bar{\tau}}$ with a tree tensor network representation as follows:

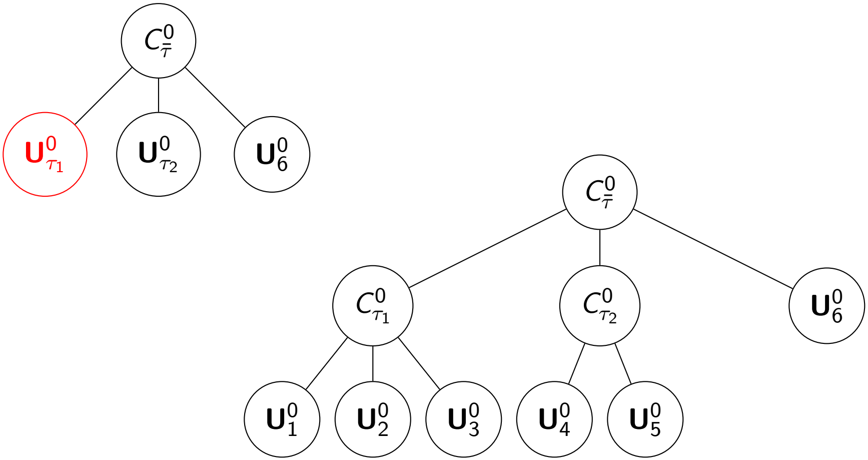
(i) For each leaf $\tau = \ell \in \mathcal{L}$, we set

$$X_\ell := \mathbf{U}_\ell^\top \in \mathbb{R}^{r_\ell \times n_\ell} \quad .$$

(ii) If, for some $m \geq 2$, the tree $\tau = (\tau_1, \ldots, \tau_m)$ is a subtree of $\bar{\tau}$, then we set $n_\tau = \prod_{i=1}^m n_{\tau_i}$ and $\mathbf{I}_\tau$ the identity matrix of dimension $r_\tau$, and

$$X_\tau := C_\tau \times_0 \mathbf{I}_\tau \mathsf{X}_{i=1}^m \mathbf{U}_{\tau_i} \in \mathbb{R}^{r_\tau \times n_{\tau_1} \times \cdots \times n_{\tau_m}},$$
$$\mathbf{U}_\tau := \mathbf{Mat}_0(X_\tau)^\top \in \mathbb{R}^{n_\tau \times r_\tau} \quad .$$

# Tree tensor network integrator - Tucker integrator first



# Tree tensor network integrator - Recursion process

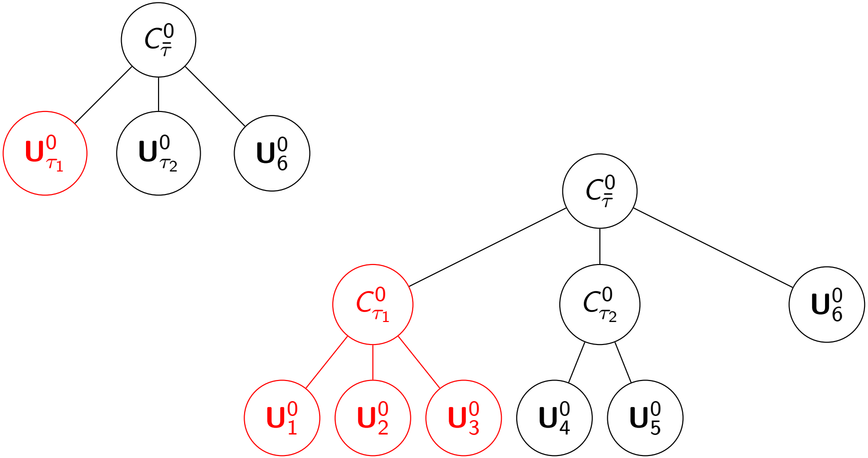
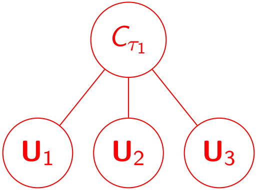



Figure: We apply the Tucker integrator on the smaller tree tensor network.

# Definition of $F_{\tau_i}$ and $Y^0_{\tau_i}$

Let $\tau = (\tau_1, \ldots, \tau_m)$ and $i = 1, \ldots, m$. We recursively define

$$F_{\tau_i} := \pi^\dagger_{\tau,i} \circ F_\tau \circ \pi_{\tau,i},$$
$$Y^0_{\tau_i} := \pi^\dagger_{\tau,i}(Y^0_\tau).$$

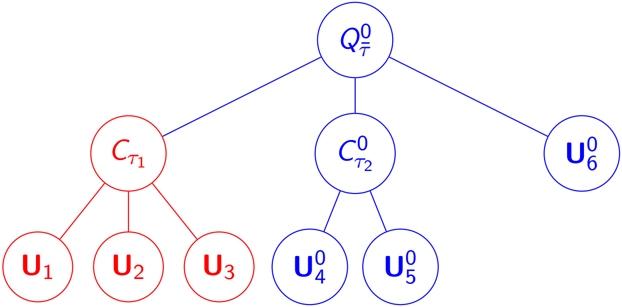


# Definition of $F_{\tau_i}$ and $Y^0_{\tau_i}$ - Prolongation

Let $\tau = (\tau_1, \ldots, \tau_m)$ and $i = 1, \ldots, m$. We recursively define

$$F_{\tau_i} := \pi^\dagger_{\tau,i} \circ F_\tau \circ \pi_{\tau,i},$$

$$Y^0_{\tau_i} := \pi^\dagger_{\tau,i}(Y^0_\tau).$$

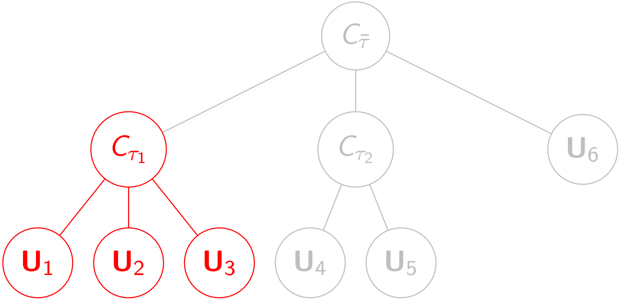


# Definition of $F_{\tau_i}$ and $Y^0_{\tau_i}$ - Restriction

Let $\tau = (\tau_1, \ldots, \tau_m)$ and $i = 1, \ldots, m$. We recursively define

$$F_{\tau_i} := \pi^\dagger_{\tau,i} \circ F_\tau \circ \pi_{\tau,i},$$

$$Y^0_{\tau_i} := \pi^\dagger_{\tau,i}(Y^0_\tau).$$



# Prolongation and restriction - Definition

Consider a tree $\tau = (\tau_1, \ldots, \tau_m)$. We define

$$\mathcal{V}_\tau := \mathbb{R}^{r_\tau \times n_{\tau_1} \times \cdots \times n_{\tau_m}} \quad .$$

We introduce the *prolongation*

$$\pi_{\tau,i}(Y_{\tau_i}) := \mathrm{Ten}_i\big((\mathbf{V}^0_{\tau_i}\, \mathbf{Mat}_0(Y_{\tau_i}))^\top\big) \in \mathcal{V}_\tau \quad \text{for} \quad Y_{\tau_i} \in \mathcal{V}_{\tau_i},$$

and the *restriction*

$$\pi^\dagger_{\tau,i}(Z_\tau) := \mathrm{Ten}_0\big((\mathbf{Mat}_i(Z_\tau)\mathbf{V}^0_{\tau_i})^\top\big) \in \mathcal{V}_{\tau_i} \quad \text{for} \quad Z_\tau \in \mathcal{V}_\tau.$$

# Prolongation and restriction - Properties

## Lemma

*Let* $\tau = (\tau_1, \ldots, \tau_m)$ *and* $i = 1, \ldots, m$. *The restriction* $\pi^\dagger_{\tau,i} : \mathcal{V}_\tau \to \mathcal{V}_{\tau_i}$ *is both a left inverse and the adjoint (with respect to the tensor Euclidean inner product) of the prolongation* $\pi_{\tau,i} : \mathcal{V}_{\tau_i} \to \mathcal{V}_\tau$, *that is,*

$$\pi^\dagger_{\tau,i}(\pi_{\tau,i}(Y_{\tau_i})) = Y_{\tau_i} \qquad \text{for all} \quad Y_{\tau_i} \in \mathcal{V}_{\tau_i}$$

$$\langle \pi_{\tau,i}(Y_{\tau_i}), Z_\tau \rangle_{\mathcal{V}_\tau} = \langle Y_{\tau_i}, \pi^\dagger_{\tau,i}(Z_\tau) \rangle_{\mathcal{V}_{\tau_i}} \qquad \text{for all} \quad Y_{\tau_i} \in \mathcal{V}_{\tau_i},\ Z_\tau \in \mathcal{V}_\tau.$$

*Moreover,* $\|\pi_{\tau,i}(Y_{\tau_i})\|_{\mathcal{V}_\tau} = \|Y_{\tau_i}\|_{\mathcal{V}_{\tau_i}}$ *and* $\|\pi^\dagger_{\tau,i}(Z_\tau)\|_{\mathcal{V}_{\tau_i}} \leq \|Z_\tau\|_{\mathcal{V}_\tau}$, *where the norms are the tensor Euclidean norms.*

# Recursive tree tensor network integrator

The recursive tree tensor network integrator is derived as a *recursive* application of the Tucker integrator.

Due to its recursive derivation, it preserves the **exactness** property and it remains **robust** with respect to the presence of small singular values in the matricizations of the connection tensors, as the matrix and the Tucker projector splitting integrator.

---

*C., Lubich, Walach 2021*

**Thanks for your attention!**

# Matrix projector-splitting integrator

1. **K-step** : Update $U_0 \to U_1, S_0 \to \hat{S}_1$
   Integrate to $t = t_1$ the $\mathbf{m} \times \mathbf{r}$ differential equation
   $$\dot{K}(t) = F(t, K(t)V_0^T)V_0, \quad K(t_0) = U_0 S_0$$
   and perform a **QR factorization** $K(t_1) = U_1 \hat{S}_1$.
2. **S-step** : Update $\hat{S}_1 \to \tilde{S}_0$
   Integrate to $t = t_1$ the $\mathbf{r} \times \mathbf{r}$ differential equation
   $$\dot{S}(t) = -U_1^T F(t, U_1 S(t) V_0^T)V_0, \quad S(t_0) = \hat{S}_1 \quad (!)$$
3. **L-step** : Update $V_0 \to V_1, \tilde{S}_0 \to S_1$
   Integrate to $t = t_1$ the $\mathbf{r} \times \mathbf{n}$ differential equation
   $$\dot{L}^T(t) = U_1^T F(t, U_1 L(t)^T), \quad L^T(t_0) = \tilde{S}_0 V_0^T$$
   and perform a **QR factorization** $L(t_1) = V_1 S_1^T$.

---

*Lubich, Oseledets 2014*

# Tensors in Tucker Format

A tensor $Y \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ has multilinear rank $(r_1, \ldots, r_d)$ if and only if it can be factorized as a *Tucker tensor*

$$Y = C \bigtimes_{i=1}^{d} \mathbf{U}_i, \quad \text{i.e.,} \quad y_{k_1,\ldots,k_d} = \sum_{l_1=1}^{r_1} \cdots \sum_{l_d=1}^{r_d} c_{l_1,\ldots,l_d} u_{k_1,l_1} \ldots u_{k_d,l_d},$$

where the *basis matrices* $\mathbf{U}_i \in \mathbb{R}^{n_i \times r_i}$ have orthonormal columns and the *core tensor* $C \in \mathbb{R}^{r_1 \times \cdots \times r_d}$ has full multilinear rank $(r_1, \ldots, r_d)$.

# Tucker Projector splitting integrator

Dynamical low-rank approximation of tensors in Tucker format is equivalent to

$$\dot{Y} = P(Y)F(t, Y), \quad Y(t_0) = Y_0 \in \mathcal{M}_{\mathbf{r}} \; .$$

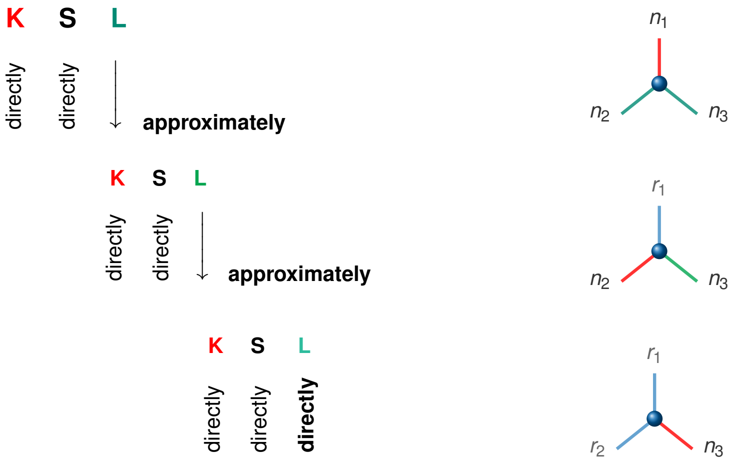
The orthogonal projection $P(Y)$ onto $\mathcal{T}_Y\mathcal{M}_{\mathbf{r}}$ is given by

$$P(Y) = \sum_{i=1}^{d} \Big( P_i^+(Y) - P_i^-(Y) \Big) + P_0(Y) \; .$$

---

*Lubich 2015*

# (d=3) Nested Tucker integrator




*Lubich, Vandereycken, Walach 2018*

# Nested Tucker integrator - Compact formulation

The result of the Tucker tensor integrator after one time step can be expressed in a compact way as

$$Y^1 = \Psi \circ \Phi^{(d)} \circ \cdots \circ \Phi^{(1)}(Y^0) \ .$$

---

*C., Lubich, Walach 2021*

**Algorithm 1:** Subflow $\Phi^{(i)}$

**Data:** $Y^0 = C^0 \bigtimes_{j=1}^{d} \mathbf{U}_j^0$ in factorized form, $F(t, Y)$, $t_0$, $t_1$

**Result:** $Y^1 = C^1 \bigtimes_{j=1}^{d} \mathbf{U}_j^1$ in factorized form

**begin**

set $\mathbf{U}_j^1 = \mathbf{U}_j^0 \quad \forall j \neq i$

compute the **QR decomposition** $\mathbf{Mat}_i(C^0)^\top = \mathbf{Q}_i^0\, \mathbf{S}_i^{0,\top} \in \mathbb{R}^{r_{\neg i} \times r_i}$

set $\mathbf{K}_i^0 = \mathbf{U}_i^0\, \mathbf{S}_i^0 \in \mathbb{R}^{n_i \times r_i}$

solve the $n_i \times r_i$ matrix differential equation

$\dot{\mathbf{K}}_i(t) = \mathbf{F}_i(t, \mathbf{K}_i(t))$ with initial value $\mathbf{K}_i(t_0) = \mathbf{K}_i^0$

and return $\mathbf{K}_i^1 = \mathbf{K}_i(t_1)$; here

$\mathbf{F}_i(t, \mathbf{K}_i) = \mathbf{Mat}_i(F(t, \mathrm{Ten}_i(\mathbf{K}_i(t)\mathbf{V}_i^{0,\top}))\mathbf{V}_i^0$ with

$\mathbf{V}_i^{0,\top} = \mathbf{Mat}_i(\mathrm{Ten}_i(\mathbf{Q}_i^{0,\top}) \bigtimes_{j \neq i} \mathbf{U}_j^0)$

compute the **QR decomposition** $\mathbf{K}_i^1 = \mathbf{U}_i^1\, \widehat{\mathbf{S}}_i^1$

solve the $r_i \times r_i$ matrix differential equation

$\dot{\mathbf{S}}_i(t) = -\widehat{\mathbf{F}}_i(t, \mathbf{S}_i(t))$ with initial value $\mathbf{S}_i(t_0) = \widehat{\mathbf{S}}_i^1$

and return $\widetilde{\mathbf{S}}_i^0 = \mathbf{S}_i(t_1)$; here

$\widehat{\mathbf{F}}_i(t, \mathbf{S}_i) = \mathbf{U}_i^{1,\top}\, \mathbf{F}_i(t, \mathbf{U}_i^1\, \mathbf{S}_i)$

set $C^1 = \mathrm{Ten}_i(\widetilde{\mathbf{S}}_i^0\, \mathbf{Q}_i^{0,\top})$

**end**

**Algorithm 2:** Subflow $\Psi$

**Data:** $Y^0 = C^0 \mathsf{X}_{j=1}^{d} \mathbf{U}_j^0$ in factorized form, $F(t, Y), t_0, t_1$

**Result:** $Y^1 = C^1 \mathsf{X}_{j=1}^{d} \mathbf{U}_j^1$ in factorized form

**begin**

set $\mathbf{U}_j^1 = \mathbf{U}_j^0 \quad \forall j = 1, \ldots, d.$

solve the $r_1 \times \cdots \times r_d$ tensor differential equation

$\dot{C}(t) = \widetilde{F}(t, C(t))$ with initial value $C(t_0) = C^0$

and return $C^1 = C(t_1)$; here

$$\widetilde{F}(t, C) = F(t, C \mathsf{X}_{j=1}^{d} \mathbf{U}_j^1) \mathsf{X}_{j=1}^{d} \mathbf{U}_j^{1,\top}$$

**end**

# Recursive TTN integrator

The recursive TTN integrator is derived as a recursive application of the Nested Tucker integrator

$$Y_\tau^1 = \Psi_\tau \circ \Phi_\tau^{(m)} \circ \cdots \circ \Phi_\tau^{(1)}(Y_\tau^0) \ .$$

*C., Lubich, Walach 2021*

**Algorithm 2:** Subflow $\Phi_\tau^{(i)}$

**Data:** tree $\tau = (\tau_1, \dots, \tau_m)$, TTN in factorized form
$Y_\tau^0 = C_\tau^0 \times_0 \mathbf{I}_\tau \mathsf{X}_{j=1}^m \mathbf{U}_{\tau_j}^0$ with $\mathbf{U}_{\tau_j}^0 = \mathbf{Mat}_0(X_{\tau_j}^0)^\top$,
function $F_\tau(t, Y_\tau), t_0, t_1$

**Result:** TTN $Y_\tau^1 = C_\tau^1 \times_0 \mathbf{I}_r \mathsf{X}_{j=1}^m \mathbf{U}_{\tau_j}^1$ with $\mathbf{U}_{\tau_j}^1 = \mathbf{Mat}_0(X_{\tau_j}^1)^\top$
in factorized form

**begin**

set $\mathbf{U}_{\tau_j}^1 = \mathbf{U}_{\tau_j}^0 \quad \forall j \neq i$

compute the **QR factorization** $\mathbf{Mat}_i(C_\tau^0)^\top = \mathbf{Q}_{\tau_i}^0 \mathbf{S}_{\tau_i}^{0,\top}$

set $Y_{\tau_i}^0 = X_{\tau_i}^0 \times_0 \mathbf{S}_{\tau_i}^{0,\top}$

**if** $\tau_i = \ell$ is a leaf, **then** solve the $n_\ell \times r_\ell$ matrix differential equation
$\dot{Y}_{\tau_i}(t) = F_{\tau_i}(t, Y_{\tau_i}(t))$ with initial value $Y_{\tau_i}(t_0) = Y_{\tau_i}^0$
and return $Y_{\tau_i}^1 = Y_{\tau_i}(t_1)$

**else**

compute $Y_{\tau_i}^1 = \textit{Recursive TTN Integrator}\ (\tau_i, Y_{\tau_i}^0, F_{\tau_i}, t_0, t_1)$

compute the **QR factorization** $\mathbf{Mat}_0(C_{\tau_i}^1)^\top = \widehat{\mathbf{Q}}_{\tau_i}^1 \widehat{\mathbf{S}}_{\tau_i}^1$, where
$C_{\tau_i}^1$ is the connecting tensor of $Y_{\tau_i}^1$

set $\mathbf{U}_{\tau_i}^1 = \mathbf{Mat}_0(X_{\tau_i}^1)^\top$, where the TTN $X_{\tau_i}^1$ is obtained from $Y_{\tau_i}^1$ by
replacing the connecting tensor with $\widehat{C}_{\tau_i}^1 = \mathrm{Ten}_0(\widehat{\mathbf{Q}}_{\tau_i}^{1,T})$

solve the $r_{\tau_i} \times r_{\tau_i}$ matrix differential equation
$\dot{\mathbf{S}}_{\tau_i}(t) = -\widehat{\mathbf{F}}_{\tau_i}(t, \mathbf{S}_{\tau_i}(t))$ with initial value $\mathbf{S}_{\tau_i}(t_0) = \widehat{\mathbf{S}}_{\tau_i}^1$
and return $\widetilde{\mathbf{S}}_{\tau_i}^0 = \mathbf{S}_{\tau_i}(t_1)$; here

$$\widehat{\mathbf{F}}_{\tau_i}(t, \mathbf{S}_{\tau_i}) = \mathbf{U}_{\tau_i}^{1,\top} \mathbf{Mat}_0\big(F_{\tau_i}(t, X_{\tau_i}^1 \times_0 \mathbf{S}_{\tau_i}^\top)\big)^\top$$

set $C_\tau^1 = \mathrm{Ten}_i(\widetilde{\mathbf{S}}_{\tau_i}^0 \mathbf{Q}_{\tau_i}^{0,\top})$

**end**

# Tensor Trains represented as TTNs

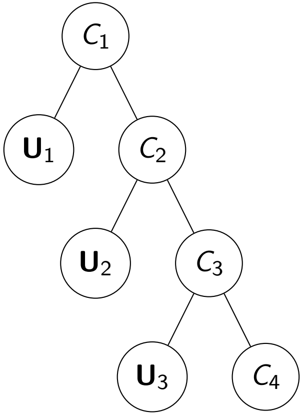



Figure: Tensor train represented in hierarchical Tucker (HT) format.