Algorithmic detection of conserved quantities of finite-difference schemes for partial differential equations

DIOGO A. GOMES, King Abdullah University of Science and Technology (KAUST), Saudi Arabia
FRIEDEMANN KRANNICH, King Abdullah University of Science and Technology (KAUST), Saudi Arabia
RICARDO DE LIMA RIBEIRO, King Abdullah University of Science and Technology (KAUST), Saudi Arabia

Many partial differential equations (PDEs) admit conserved quantities like mass or energy. Those quantities are often essential to establish well-posed results. When approximating a PDE by a finite-difference scheme, it is natural to ask whether related discretized quantities remain conserved over time by the scheme. Such conservation may establish the stability of the numerical scheme. We present an algorithm for checking the preservation of a polynomial quantity under a polynomial finite-difference scheme. Our schemes can be explicit or implicit, have higher-order time and space derivatives, and an arbitrary number of variables. Additionally, we present an algorithm for, given a scheme, finding conserved quantities. We illustrate our algorithm with several finite-difference schemes.

CCS Concepts: • **Mathematics of computing → Partial differential equations**; **Discretization**; **Gröbner bases and other special bases**; • **Computing methodologies → Symbolic calculus algorithms**; **Discrete calculus algorithms**.

Additional Key Words and Phrases: Symbolic computations; Finite-difference schemes; Discrete variational derivative; Discrete partial variational derivative; Conserved quantities; Implicit schemes; Explicit schemes.

ACM Reference Format:
Diogo A. Gomes, Friedemann Krannich, and Ricardo de Lima Ribeiro. 2022. Algorithmic detection of conserved quantities of finite-difference schemes for partial differential equations. 1, 1 (August 2022), 18 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Contents

Authors' addresses: Diogo A. Gomes, King Abdullah University of Science and Technology (KAUST), AMCS/CEMSE Division, Thuwal, 23955-6900, Saudi Arabia, diogo.gomes@kaust.edu.sa; Friedemann Krannich, King Abdullah University of Science and Technology (KAUST), AMCS/CEMSE Division, Thuwal, 23955-6900, Saudi Arabia, friedemann.krannich@kaust.edu.sa; Ricardo de Lima Ribeiro, King Abdullah University of Science and Technology (KAUST), AMCS/CEMSE Division, Thuwal, 23955-6900, Saudi Arabia, ricardo.ribeiro@kaust.edu.sa.

ACKNOWLEDGMENTS

The authors were supported by King Abdullah University of Science and Technology (KAUST) baseline funds and KAUST OSR-CRG2021-4674.

1 INTRODUCTION

Many partial differential equations (PDEs) admit integral quantities, that are conserved in time. For example, the advection equation preserves energy and the heat equation conserves mass. When approximating PDEs by a finite-difference scheme, the question arises whether such quantities are conserved by the scheme. Such information can be crucial to estimate whether a scheme approximates a PDE accurately and to determine its stability.

Computations for determining conservation can get rather tedious. Hence, automating them in computer algebra systems is desirable.

Conserved quantities correspond to conservation laws, admitted by the PDE. A conservation law [4] is an equation of the form

$$D_t\Phi[u] + D_x\Psi[u] = 0$$

holding for all solutions u of the considered PDE, that induces the conserved quantity $\int \Phi[u]\,dx$ as

$$\frac{d}{dt}\int \Phi[u]\,dx = \int D_t\Phi[u]\,dx = -\int D_x\Psi[u]\,dx = 0$$

assuming periodic boundary conditions in x.

An analog formulation describes conservation laws for schemes of PDEs, where the derivatives are replaced by shifts [15]. The key for the construction of schemes, that admit certain conservation laws from the continuous PDE, is the use of the discrete Euler operator [4] (also called discrete variational derivative). Kupershmidt [16, II. Theorem 31] discovered, that an equation is a discrete conservation law if and only if it belongs to the kernel of the discrete Euler operator. Hence, standard approaches for constructing schemes with conservation are either discretizing the continuous conservation law or finding multipliers for the scheme [5], such that the result is in the kernel of the discrete Euler operator. This strategy was, for example, recently used by Dorodnitsyn et al. to develop a scheme for the shallow water equation, that preserves energy [5]. Cheviakov et al. used this idea to find schemes for the linear and nonlinear wave equation, that admit several discrete analogs of continous conservation laws [4]. The same strategies for finding conservation laws, that are described above, work for continuous PDEs or semi-discretized PDEs: Algorithms for finding conserved quantities, using multipliers and the Euler operator for PDEs were developed and implemented by Cheviakov [2, 3]. Hereman et al. [12, 13], proposed an algorithm to compute conserved densities for semi-discretized schemes for

PDEs with first-order time derivative. Their algorithm uses the scaling symmetries of the scheme to construct conserved quantities and calculates their coefficients using the discrete Euler operator. Gao et al. extended this algorithm to first-order time-explicit schemes [6] and first-order time-implicit schemes [7].
Bihlo et al. [1] used another approach and derived finite-difference schemes for the shallow water equations, that are symmetry-invariant on an adaptive mesh and then analyzed them regarding conserved quantities.
In this paper, we propose an algorithm that checks if a quantity is conserved in time under a scheme. Our approach differs from the ideas described above, as we do not try to construct discrete Φ and Ψ, using the scheme, but we check if a given discrete Φ is conserved in time under a given scheme.
Conserved quantities usually involve integral expressions or their discrete analog, sums. Gomes et al. proposed algorithms for the simplification of sums [11]. They also developed and implemented algorithms for detecting quantities conserved by PDEs and semi-discretized schemes. We revise these techniques in Section 2. To generalize those methods to situations where we do not sum over all arguments of the functions involved, we introduce the discrete partial variational derivative (Section 3). The main contribution of this paper is an algorithm for checking the conservation of a quantity under a numerical scheme. Gerdt showed [8], that the quantity is conserved, if its discrete time derivative belongs to the difference ideal generated by the scheme. However, some quantities may add to a constant (e.g. telescopic sums) and thus be trivially preserved without belonging to the difference ideal. Moreover, Gerdt's algorithm may not terminate, as the Gröbner basis for the difference ideal may not be finite. We overcome these issues by combining the discrete partial variational derivative with a polynomial ideal (instead of a difference ideal) with finite Gröbner basis (Section 4). Our algorithm works for schemes that are explicit and implicit in time and can treat schemes with several higher-order time and space derivatives, and with several space dimensions. Additionally, we can handle systems of equations and schemes with parameters. We have implemented this algorithm as part of a package in MATHEMATICA [18] [1]. In the examples, we show that our code finds conserved quantities and proper schemes for the time-implicit and time-explicit discretization of the Burgers equation and a system of PDEs arising in the study of mean-field games (Section 8).

2 PRELIMINARIES

Here, we follow the ideas in [11]. In this paper, subscripts denote indices related to coordinates or tuples and superscripts denote indices related to sequences. To avoid boundary terms, we work with periodic functions in $\mathbb{Z}^d$.

Definition 2.1. *Let N, d and m be positive integers. The* discrete torus *is $\mathcal{I} := \{0, \dots, N-1\}^d \subset \mathbb{Z}^d$. Define the space*

$$\mathcal{P}(\mathcal{I}, \mathbb{R}^m) := \{u \mid u : \mathcal{I} \to \mathbb{R}^m\},$$

extended to $\mathbb{Z}^d$ by periodicity.

Definition 2.2. *The space of functionals $\mathcal{F}(\mathcal{P}(\mathcal{I}, \mathbb{R}^m))$ (not necessarily linear) on $\mathcal{P}(\mathcal{I}, \mathbb{R}^m)$ is*

$$\mathcal{F}(\mathcal{P}(\mathcal{I}, \mathbb{R}^m)) := \left\{ \mathcal{J} : \mathcal{P}(\mathcal{I}, \mathbb{R}^m) \to \mathbb{R} \;\middle|\; \mathcal{J}[u] = \sum_{n \in \mathcal{I}} F_n[u], \right.$$
$$\left. \text{where } F_n[u] = G(u(n+e^1), u(n+e^2), \dots, u(n+e^k)) \right\}$$

[1] The code is available upon request. For access contact ricardo.ribeiro@kaust.edu.sa.

$$\text{and } \{e^1,\dots,e^k\} \subset \mathbb{Z}^d,\ G \text{ is smooth}\Big\}.$$

In the previous definition, the $e^1,\dots,e^k$ are not necessarily unit vectors and can vary between functionals.

Example 2.3. Let $d = m = 1$ and $\mathcal{J}[u] := \sum_n (u(n+1) - u(n))^2$.

To simplify sums, the discrete variational derivative is a useful tool. Let $u, v \in \mathcal{P}(\mathcal{I}, \mathbb{R}^m)$, $\epsilon \in \mathbb{R}$ and $\mathcal{J} \in \mathcal{F}(\mathcal{P}(\mathcal{I}, \mathbb{R}^m))$. Define

$$D\mathcal{J}[u](v) := \frac{d}{d\epsilon}\mathcal{J}[u+\epsilon v]\Big|_{\epsilon=0}.$$

Below in (2.1), we shift the indices due to periodicity of u and v,

$$\begin{aligned}
D\mathcal{J}[u](v) &= \frac{d}{d\epsilon}\mathcal{J}[u+\epsilon v]\Big|_{\epsilon=0} = \frac{d}{d\epsilon}\sum_{n\in\mathcal{I}} F_n[u+\epsilon v]\Big|_{\epsilon=0} \\
&= \sum_{n\in\mathcal{I}} \frac{d}{d\epsilon} G(u(n+e^1)+\epsilon v(n+e^1),\dots,u(n+e^k)+\epsilon v(n+e^k))\Big|_{\epsilon=0} \\
&= \sum_{n\in\mathcal{I}} D_1 G(u(n+e^1),\dots,u(n+e^k))v(n+e^1) \\
&\quad + \dots + D_k G(u(n+e^1),\dots,u(n+e^k))v(n+e^k) \\
&= \sum_{n\in\mathcal{I}} D_1 G(u(n), u(n+e^2-e^1),\dots,u(n+e^k-e^1))v(n) \\
&\quad + D_2 G(u(n+e^1-e^2), u(n),\dots,u(n+e^k-e^2))v(n) \\
&\quad + \dots + D_k G(u(n+e^1-e^k), u(n+e^2-e^k),\dots,u(n))v(n) \\
&= \sum_{n\in\mathcal{I}} \Big(D_1 G(u(n), u(n+e^2-e^1),\dots,u(n+e^k-e^1)) \\
&\quad + D_2 G(u(n+e^1-e^2), u(n),\dots,u(n+e^k-e^2)) \\
&\quad + \dots + D_k G(u(n+e^1-e^k), u(n+e^2-e^k),\dots,u(n))\Big)v(n) \\
&=: \sum_{n\in\mathcal{I}} \mathcal{V}(\mathcal{J})[u](n)v(n).
\end{aligned} \tag{2.1}$$

Definition 2.4. $\mathcal{V}(\mathcal{J})$ *is the* discrete variational derivative *of* $\mathcal{J}$.

Because we assume all related functions to be smooth, the discrete variational derivative always exists.

Example 2.5. Let us return to Example 2.3 and compute

$$\begin{aligned}
D\mathcal{J}[u](v) &= \frac{d}{d\epsilon}\mathcal{J}[u+\epsilon v]\Big|_{\epsilon=0} = \frac{d}{d\epsilon}\sum_{n\in\mathcal{I}} F_n[u+\epsilon v]\Big|_{\epsilon=0} \\
&= \sum_{n\in\mathcal{I}} \frac{d}{d\epsilon} G[u(n+1)+\epsilon v(n+1), u(n)+\epsilon v(n)]\Big|_{\epsilon=0}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{n\in\mathcal{I}} 2(u(n+1)-u(n))v(n+1) + (-2)(u(n+1)-u(n))v(n)\\
&= \sum_{n\in\mathcal{I}} 2(u(n)-u(n-1))v(n) + (-2)(u(n+1)-u(n))v(n)\\
&= \sum_{n\in\mathcal{I}} \Big(2(u(n)-u(n-1)) + (-2)(u(n+1)-u(n))\Big)v(n)\\
&= \sum_{n\in\mathcal{I}} \Big(-2u(n+1)+4u(n)-2u(n-1)\Big)v(n)\\
&= \sum_{n\in\mathcal{I}} \mathcal{V}(\mathcal{J})[u](n)v(n)
\end{aligned}
$$

and hence $\mathcal{V}(\mathcal{J})[u](n) = -2u(n+1)+4u(n)-2u(n-1)$.

The algorithms for the simplification of sums presented in [11] rely on the following result:

Theorem 2.6. Let $\mathcal{J},\tilde{\mathcal{J}} \in \mathcal{F}(\mathcal{P}(\mathcal{I},\mathbb{R}^m))$. If $\mathcal{V}(\mathcal{J}-\tilde{\mathcal{J}})[v]=0$ for all $v\in\mathcal{P}(\mathcal{I},\mathbb{R}^m)$ and if there exists $u_0\in\mathcal{P}(\mathcal{I},\mathbb{R}^m)$ such that $\mathcal{J}[u_0]=\tilde{\mathcal{J}}[u_0]$, then $\mathcal{J}[u]=\tilde{\mathcal{J}}[u]$ for all $u\in\mathcal{P}(\mathcal{I},\mathbb{R}^m)$. Conversely, $\mathcal{V}(\mathcal{J}-\tilde{\mathcal{J}})[u]=0$ for all $u\in\mathcal{P}(\mathcal{I},\mathbb{R}^m)$, if $\mathcal{J}[u]=\tilde{\mathcal{J}}[u]$.

Proof.

$$
\begin{aligned}
(\mathcal{J}-\tilde{\mathcal{J}})[u] &= (\mathcal{J}-\tilde{\mathcal{J}})[u] - (\mathcal{J}-\tilde{\mathcal{J}})[u_0]\\
&= \int_0^1 \frac{d}{d\lambda}(\mathcal{J}-\tilde{\mathcal{J}})[u_0+\lambda(u-u_0)]\,d\lambda\\
&= \int_0^1 \mathcal{V}(\mathcal{J}-\tilde{\mathcal{J}})[u_0+\lambda(u-u_0)](u-u_0)\,d\lambda = 0
\end{aligned}
$$

and hence $\mathcal{J}[u]=\tilde{\mathcal{J}}[u]$. □

We use Theorem 2.6 to examine if different sums represent the same quantity.

Example 2.7. Let $d=m=1$ and consider the functionals

$$
\mathcal{J}[u] := \sum_{n\in\mathcal{I}} u(n)u(n+1) \text{ and } \tilde{\mathcal{J}}[u] := \sum_{n\in\mathcal{I}} u(n-2)u(n-1).
$$

It is clear, that $\mathcal{J}$ and $\tilde{\mathcal{J}}$ represent the same quantity (by shifting n). We confirm this, using the discrete variational derivative:

$$
\begin{aligned}
&D(\mathcal{J}-\tilde{\mathcal{J}})[u](v) =\\
&= \sum_{n\in\mathcal{I}} (u(n+1)+u(n-1)-u(n+1)-u(n-1))v(n) = 0.
\end{aligned}
$$

Hence, $\mathcal{V}(\mathcal{J}-\tilde{\mathcal{J}})=0$ and $\mathcal{J}[0]=0=\tilde{\mathcal{J}}[0]$. Therefore, both functionals sum to the same quantity.

Example 2.8. Let $d=2$ and $m=1$. It may not be obvious, that

$$
\sum_{n_1,n_2} u(n_1,n_2-2)^4 - 3u(n_1,n_2-2)^3u(n_1+1,n_2-2)
$$

$$+u(n_1,n_2)u(n_1+1,n_2)^3+3u(n_1,n_2-2)^2u(n_1+1,n_2-2)^2$$
$$-u(n_1,n_2-2)u(n_1+1,n_2-2)^3-u(n_1,n_2)^4+$$
$$3u(n_1,n_2)^3u(n_1+1,n_2)-3u(n_1,n_2)^2u(n_1+1,n_2)$$
$$-3u(n_1,n_2-2)^2u(n_1+1,n_2-2)^2+3u(n_1,n_2)^2u(n_1+1,n_2)=0.$$

We confirm this, by computing the discrete variational derivative and noticing that the expression is 0 for $u_0 = 0$.

3 THE DISCRETE PARTIAL VARIATIONAL DERIVATIVE

Here, we generalize the discrete variational derivative for situations where we keep one (or several) of the arguments of u constant. Let $n = (n_1, \dots, n_d)$ be the variables for the functions in $\mathcal{P}(\mathcal{I}, \mathbb{R}^m)$. We call n the space variables. Let $l \in \mathbb{N}$ and call $t := (t_1, \dots, t_l) \in \mathbb{N}^l$ the time variables. Later in this paper, we only consider the case $l = 1$.

Definition 3.1. *Let $\mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m) := \{u \mid u(\cdot, t) \in \mathcal{P}(\mathcal{I}, \mathbb{R}^m)\ \forall t \in \mathbb{N}^l\}$, extended to $\mathbb{Z}^d \times \mathbb{N}^l$ by periodicity in the space variables and let $\mathcal{D}(\mathbb{N}^l) := \{u \mid u : \mathbb{N}^l \to \mathbb{R}\}$.*

When considering sums, we only sum over the space variables and not over the time variables.

Definition 3.2. *The space of functions $\mathcal{F}(\mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m), \mathcal{D}(\mathbb{N}^l))$ is*

$$\mathcal{F}(\mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m), \mathcal{D}(\mathbb{N}^l)) :=$$
$$\left\{ \mathcal{J} : \mathcal{P}(\mathcal{I} \times \mathbb{N}^l) \to \mathcal{D}(\mathbb{N}^l) \,\middle|\, \mathcal{J}[u](t) = \sum_{n \in \mathcal{I}} F_{(n,t)}[u], \right.$$
$$\text{where } F_{(n,t)}[u] = G(u((n,t)+e^1), u((n,t)+e^2), \dots, u((n,t)+e^k)),$$
$$\left. \text{and } \{e^1, \dots, e^k\} \subset \mathbb{Z}^{d+l},\ G \text{ is a polynomial in } k \text{ variables} \right\}.$$

Here, $(n,t) := (n_1, \dots, n_d, t_1, \dots, t_l) \in \mathcal{I} \times \mathbb{N}^l$.

Here, all quantities, whose conservation in time we check, are functions $\mathcal{J} \in \mathcal{F}(\mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m), \mathcal{D}(\mathbb{N}^l))$. Let $\tilde{n} := (n_1, \dots, n_d, s_1, \dots, s_l) \in \mathcal{I} \times \mathbb{N}^l$. Using the Kronecker delta δ, we rewrite

$$\mathcal{J}[u](t_1, \dots, t_l) = \sum_{\tilde{n} \in \mathcal{I} \times \mathbb{N}^l} F_{\tilde{n}}[u] \delta_{(t_1,\dots,t_l)}(s_1, \dots, s_l).$$

Let $u, v \in \mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m)$ and compute

$$D^l\mathcal{J}[u](v)(t_1, \dots, t_l) := \frac{d}{d\epsilon}\mathcal{J}[u+\epsilon v]\Big|_{\epsilon=0}(t_1, \dots, t_l)$$
$$= \sum_{\tilde{n}} D_1 G(u(\tilde{n}+e^1), \dots, u(\tilde{n}+e^k))\delta_{(t_1,\dots,t_l)}(s_1, \dots, s_l)v(\tilde{n}+e^1)$$
$$+D_2 G(u(\tilde{n}+e^1), \dots, u(\tilde{n}+e^k))\delta_{(t_1,\dots,t_l)}(s_1, \dots, s_l)v(\tilde{n}+e^2)$$
$$+\dots+D_k G(u(\tilde{n}+e^1), \dots, u(\tilde{n}+e^k))\delta_{(t_1,\dots,t_l)}(s_1, \dots, s_l)v(\tilde{n}+e^k)$$
$$= \sum_{\tilde{n}} D_1 G(u(\tilde{n}), \dots, u(\tilde{n}+e^k-e^1))\delta_{(t_1,\dots,t_l)}(s_1-e^1_{d+1}, \dots, s_l-e^1_{d+l})v(\tilde{n})$$
$$+D_2 G(u(\tilde{n}+e^1-e^2), \dots, u(\tilde{n}+e^k-e^2))\delta_{(t_1,\dots,t_l)}(s_1-e^2_{d+1}, \dots, s_l-e^2_{d+l})v(\tilde{n})$$
$$+\dots+D_k G(u(\tilde{n}+e^1-e^k), \dots, u(\tilde{n}))\delta_{(t_1,\dots,t_l)}(s_1-e^k_{d+1}, \dots, s_l-e^k_{d+l})v(\tilde{n})$$

$$
\begin{aligned}
&= \sum_{\tilde{n}} \Big(D_1 G(u(\tilde{n}), \dots, u(\tilde{n}+e^k-e^1)) \delta_{(t_1+e^1_{d+1},\dots,t_l+e^1_{d+l})}(s_1,\dots,s_l) \\
&+ D_2 G(u(\tilde{n}+e^1-e^2), \dots, u(\tilde{n}+e^k-e^2)) \delta_{(t_1+e^2_{d+1},\dots,t_l+e^2_{d+l})}(s_1,\dots,s_l) \\
&+ \dots + D_k G(u(\tilde{n}+e^1-e^k), \dots, u(\tilde{n})) \delta_{(t_1+e^k_{d+1},\dots,t_l+e^k_{d+l})}(s_1,\dots,s_l) \Big) v(\tilde{n}) \\
&=: \sum_{\tilde{n}} \mathcal{V}^l(\mathcal{J})[u](\tilde{n},t) v(\tilde{n}).
\end{aligned}
$$

Definition 3.3. *We define* $\mathcal{V}^l(\mathcal{J})$ *as the* discrete partial variational derivative *of* $\mathcal{J}$. *Note that* $\mathcal{V}^0 := \mathcal{V}$.

Example 3.4. The discrete partial variational derivative of

$$\mathcal{J}[u](t) = \sum_n u(n,t+1) - u(n,t) = \sum_{n,s} (u(n,s+1) - u(n,s)) \delta_t(s)$$

equals $\delta_{t+1}(s) - \delta_t(s)$, because

$$
\begin{aligned}
&D^1(\mathcal{J})[u](v)(t) \\
&= \frac{d}{d\epsilon} \sum_{n,s} ((u(n,s+1) + \epsilon v(n,s+1)) - (u(n,s) + \epsilon v(n,s))) \delta_t(s) \\
&= \sum_{n,s} v(n,s+1)\delta_t(s) - v(n,s)\delta_t(s) = \sum_{n,s} v(n,s)\delta_t(s-1) - v(n,s)\delta_t(s) \\
&= \sum_{n,s} (\delta_{t+1}(s) - \delta_t(s)) v(n,s) = \sum_{n,s} \mathcal{V}^1(\mathcal{J})(n,s,t) v(n,s).
\end{aligned}
$$

Example 3.5. In our MATHEMATICA implementation, Example 3.4 is computed by

```
In[1]:= variables=<|"indVars"→{n,t},"depVars"→{u},
        "timeVars"→{t}|>
        PartialDVarD[variables][u[n,t+1]-u[n,t]]
Out[1]= {-KroneckerDelta[t]+KroneckerDelta[1+t]}
```

We have a result similar to Theorem 2.6 for the discrete partial variational derivative:

Theorem 3.6. Let $\mathcal{J}, \tilde{\mathcal{J}} \in \mathcal{F}(\mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m), \mathcal{D}(\mathbb{N}^l))$. If $\mathcal{V}^l(\mathcal{J} - \tilde{\mathcal{J}})[v] = 0$ for all $v \in \mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m)$ and if there exists $u_0 \in \mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m)$ such that $\mathcal{J}[u_0] = \tilde{\mathcal{J}}[u_0]$, then $\mathcal{J}[u] = \tilde{\mathcal{J}}[u]$ for all $u \in \mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m)$. Conversely, $\mathcal{V}^l(\mathcal{J} - \tilde{\mathcal{J}})[u] = 0$ for all $u \in \mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m)$, if $\mathcal{J}[u] = \tilde{\mathcal{J}}[u]$.

The proof is similar to the proof of Theorem 2.6.

4 DIFFERENCE SCHEMES AND ALGEBRA

In this section, we define numerical schemes formally and introduce the tools from difference algebra, that we need for our algorithm. During the remainder of this paper, we assume that there is only one time variable t ($l = 1$).

Definition 4.1. *The* space of schemes *is*

$$
\begin{aligned}
&\mathcal{S}(\mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m), \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R})) := \\
&\left\{ H : \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m) \to \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}) \;\middle|\; \right.
\end{aligned}
$$

$$H[u](n,t) = H(u(n+\tilde{e}^1, t+\tilde{l}^1), \dots, u(n+\tilde{e}^r, t+\tilde{l}^r)),$$
$$\left.\begin{array}{c} \text{where } \{\tilde{e}^1,\dots,\tilde{e}^r\} \subset \mathbb{Z}^d,\ \{\tilde{l}^1,\dots,\tilde{l}^r\} \subset \mathbb{Z}, \\ H \text{ is a polynomial in } r \text{ variables} \end{array}\right\}.$$

A scheme *is a set of functions*

$$\{H^1,\dots,H^i\} \subset \mathcal{S}(\mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}^m),\mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}))$$

that represent the equations $\{H^1[u]=0,\dots,H^i[u]=0\}$ *for* $u \in \mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}^m)$*, holding pointwise for all points in* $\mathcal{I}\times\mathbb{N}$*.*

If a scheme contains a single function H, we call H the scheme. Sometimes, we also call the expression $H[u]$ the scheme.

Remark 4.2. In our examples, we only consider finite-difference schemes with fixed step sizes $\Delta x = \Delta t = 1$. However, our algorithms and our code handle parameters. Hence, one can generalize our results to schemes with other step sizes.

Definition 4.3. *A scheme* $\{H^1,\dots,H^m\} \subset \mathcal{S}(\mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}^m),\mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}))$ *given by*

$$H^j(u(n+\tilde{e}^1_j, t+\tilde{l}^1_j), \dots, u(n+\tilde{e}^r_j, t+\tilde{l}^r_j))$$

is in time-explicit form *if we can rewrite the previous equation (eventually translating the scheme) as*

$$u_j(n,t+1) - \tilde{H}^j(u(n+\tilde{e}^1_j,t), u(n+\tilde{e}^2_j,t), \dots, u(n+\tilde{e}^r_j,t))$$

for $1 \le j \le m$ *with* $\tilde{H}^j \in \mathcal{S}(\mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}^m),\mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}))$*. We call* $\{\tilde{H}^1,\dots,\tilde{H}^m\}$ *the right-hand side of* $\{H^1,\dots,H^m\}$*.*

Example 4.4. Consider the forward-difference scheme for the heat equation $u_t - u_{xx} = 0$ [17]

$$u(n,t+1) - u(n,t) - (u(n+1,t) - 2u(n,t) + u(n-1,t)).$$

This scheme is in time-explicit form with right-hand side

$$u(n,t) + (u(n+1,t) - 2u(n,t) + u(n-1,t)).$$

Difference ideals. Following Gerdt [8], we now introduce difference ideals and how we use them in our algorithm.

Definition 4.5. *Let* $u = (u_1,\dots,u_m) \in \mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}^m)$*. The* shift *in the* i*-th coordinate for* $1 \le i \le d+1$ *by* $k \in \mathbb{Z}$ *is*

$$\sigma_i^k \circ u_j(n,t) := u_j(n_1,\dots,n_{i-1},n_i+k,n_{i+1},\dots,t)$$

understanding that σ^k_{d+1} *shifts the time variable.*

Definition 4.6. *The* set of all possible shifts Θ *is*

$$\Theta := \{\sigma_1^{k_1} \circ \cdots \circ \sigma_{d+1}^{k_{d+1}} \mid k_1,\dots,k_{d+1} \in \mathbb{Z}\}.$$

Now we construct the difference ideal containing the scheme:

Definition 4.7. *Let $\mathcal{K}$ be the field generated by $\mathbb{R}$ and the variables $\{n_1, \dots, n_d, t\}$. Let $\tilde{\mathcal{R}}$ be the polynomial ring over the field $\mathcal{K}$ and the variables $\theta \circ u_j(n,t)$ for $\theta \in \Theta$ and $1 \le j \le m$. A set $\tilde{I} \subseteq \tilde{\mathcal{R}}$ is a* difference ideal *if $p_1, p_2 \in \tilde{I}$ implies $p_1 + p_2 \in \tilde{I}$; $p_1 \in \tilde{I}, p_2 \in \tilde{\mathcal{R}}$ implies $p_1 \cdot p_2 \in \tilde{I}$ and $p_1 \in \tilde{I}, \theta \in \Theta$ implies $\theta \circ p_1 \in \tilde{I}$.*

Definition 4.8. *$\langle H^1, \dots, H^i \rangle$ is the smallest difference ideal containing the scheme $\{H^1, \dots, H^i\}$.*

A solution of a numerical scheme is a function u, that makes all translations of the scheme vanish. Hence, every element of the difference ideal generated by the scheme vanishes under u. Algorithm 2 seeks to determine if the polynomial associated with the discrete time derivative $\mathcal{T} \coloneqq \mathcal{J}[u](t+1) - \mathcal{J}[u](t)$ belongs to the ideal $\langle H^1, \dots, H^i \rangle$. To examine, if $\mathcal{T} \in \langle H^1, \dots, H^i \rangle$, Gerdt proposes the notion of a standard (Gröbner) basis for the ideal $\langle H^1, \dots, H^i \rangle$. This standard basis may not be finite. We overcome this problem by considering polynomial instead of difference algebra and using a smaller polynomial ideal, that is contained in the difference ideal and admits a finite standard basis. However, there may be functions $\mathcal{J}$, whose sums add to zero, even though they may fail to belong to the difference ideal. Hence, we combine the discrete partial variational derivative with polynomial ideals.

5 THE TIME-EXPLICIT CASE

In this section, we present Algorithm 1 for checking the conservation of a quantity under the right-hand side of a scheme in time-explicit form. We discuss our algorithm for general schemes in the next section. Note, that the result of

Algorithm 1: time-explicit DiscreteConservedQ

Input: $\mathcal{J} \in \mathcal{F}(\mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m), \mathcal{D}(\mathbb{N}))$ and $\{H^1, \dots, H^m\} \subset \mathcal{S}(\mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m), \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}))$ right-hand side of a time-explicit scheme

Output: True, if $\mathcal{J}$ is conserved in time, False otherwise

1 Build the discrete time derivative $\mathcal{J}[u](t+1) - \mathcal{J}[u](t)$
2 Replace all instances of $u_j(n+e, t+1)$ (for $e \in \mathbb{Z}^d$) by an appropriate translation of $\{H^1, \dots, H^m\}$
3 Compute the discrete partial variational derivative
4 If the result is zero, conservation is True, else False

Step 2 exists only at time t because all instances of $t+1$ have been replaced. Thus, we do not need any computations of difference ideals or Gröbner basis.

Example 5.1. We check for conservation of $\sum_n u(n,t)$ under the scheme

$$u(n,t+1) - u(n,t) - (u(n+1,t) - 2u(n,t) + u(n-1,t)).$$

The discrete time derivative (Step 1) is

$$\sum_{n \in \mathcal{I}} u(n,t+1) - u(n,t).$$

Replace $u(n,t+1)$ by the right-hand side (Step 2) to get

$$\sum_{n \in \mathcal{I}} u(n,t) + (u(n+1,t) - 2u(n,t) + u(n-1,t)) - u(n,t).$$

Then, we compute the discrete partial variational derivative (Step 3), which equals zero. Hence (Step 4), we have conservation.

Example 5.2. We verify the result from Example 5.1, using our implementation in MATHEMATICA of Algorithm 1.

```
In[2]:= variables=<|"indVars"→{n},"depVars"→{u},
        "rhs"→{u[n]+(u[n+1]-2u[n]+u[n-1])}|>
        DiscreteConservedQ[variables][u[n]]
Out[2]= True
```

6 THE GENERAL CASE

In this section, we present Algorithm 2 that deals with general, not necessarily time-explicit, schemes. We explain its steps in detail below and demonstrate the algorithm in Example 6.8.

Step 1: Build the discrete time derivative. We build the discrete time derivative by subtracting $\mathcal{J}[u](t)$ from

Algorithm 2: GENERAL DISCRETECONSERVEDQ

Input: $\mathcal{J} \in \mathcal{F}(\mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m), \mathcal{D}(\mathbb{N}))$ and $\{H^1, \dots, H^i\} \subset \mathcal{S}(\mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m), \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}))$
Output: TRUE, if $\mathcal{J}$ is conserved in time, FALSE otherwise

1 Build the discrete time derivative $\mathcal{T} := \mathcal{J}[u](t+1) - \mathcal{J}[u](t)$
2 Translate $\{H^1, \dots, H^i\}$ by subtracting the range of the stencil of the scheme from the range of the stencil of $\mathcal{T}$
3 Reduce $\mathcal{T}$ by using the Gröbner basis generated by the translated scheme in two different ways (a) Using the lexicographic order (b) Using the explicit elimination order
4 Choose the result that admits the least different instances of time
5 Compute the discrete partial variational derivative
6 Apply Steps 2 to 4 to the result
7 If the result is zero, conservation is TRUE, else FALSE

$\mathcal{J}[u](t+1)$. This task is done in our code by the TIMEDIFFERENCE.

Example 6.1. The discrete time derivative for $\sum_n u(n,t)$ is

```
In[3]:= variables=<|"indVars"→{n,t},"depVars"→{u}|>
        TimeDifference[variables][u[n,t]]
Out[3]= <|exp→-u[n,t]+u[n,1+t]|>
```

Step 2: Translate the scheme. We now make the step from difference algebra, with an infinite number of independent variables, to polynomial algebra, where we only have a finite number of variables to guarantee finiteness of the Gröbner basis. Hence, we need to compute a finite number of translations of the scheme and then consider every instance $u_j(n+e, t+l)$, that appears either in the translated scheme or in the discrete time derivative as independent polynomial variable.

Example 6.2. The algorithm treats the discrete time derivative $\sum_{n\in\mathcal{I}} u(n+1,t+1) - u(n+1,t)$ as the polynomial $z_1 - z_2$.

There are several possiblities which translations of the scheme to take as basis for the polynomial ideal. However, an obvious idea is to compute the minimal number of translations of the scheme, such that all instances of $u_j(n+e, t+l)$, that appear in the range of the stencil of the discrete time derivative, also appear in the translated scheme. This is done by elementwise subtracting the range of the stencil of the scheme from the range of the stencil of the discrete

time derivative. Before we define the above translations formally, we give a simple example to illustrate the idea of computing the translations:

Example 6.3. Consider the expression

$$u(n+3) - u(n-2)$$

which we want to reduce by the scheme $u(n+1) - u(n) = 0$. We compute the translations as described above as

$$\{u(n+3) - u(n+2), \dots, u(n-1) - u(n-2)\}.$$

and see, that the expression indeed reduces to 0 under the scheme.

Definition 6.4. *Let* $u = (u_1, \dots, u_m) \in \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m)$ *and*

$$F : \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m) \to \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}),$$
$$F[u](n,t) = F(u_1((n,t) + x^1), \dots, u_1((n,t) + x^{k_1}),$$
$$u_2((n,t) + x^{k_1+1}), \dots, u_2((n,t) + x^{k_2}),$$
$$\dots, u_m((n,t) + x^{k_{m-1}+1}), \dots, u_m((n,t) + x^{k_m}))$$

with $\{x^1, \dots, x^{k_1}, \dots, x^{k_m}\} \subset \mathbb{Z}^{d+1}$. *The* stencil *of* F *is the* m*-tuple of sets of vectors*

$$\left(\left\{x^1, \dots, x^{k_1}\right\}, \left\{x^{k_1+1}, \dots, x^{k_2}\right\}, \dots, \left\{x^{k_{m-1}+1}, \dots, x^{k_m}\right\}\right).$$

The range *of the stencil of* F *is*

$$\Big([x_1^{k_1-}, x_1^{k_1+}] \times \dots \times [x_{d+1}^{k_1-}, x_{d+1}^{k_1+}], [x_1^{k_2-}, x_1^{k_2+}] \times \dots \times [x_{d+1}^{k_2-}, x_{d+1}^{k_2+}],$$
$$\dots, [x_1^{k_m-}, x_1^{k_m+}] \times \dots \times [x_{d+1}^{k_m-}, x_{d+1}^{k_m+}]\Big),$$

where $k_0 = 0$, $x_i^{k_l+} = \max_{k_{l-1}+1 \le k \le k_l}\{x_i^k\}$, *and*
$x_i^{k_l-} = \min_{k_{l-1}+1 \le k \le k_l}\{x_i^k\}$. *Here,* $[a,b]$ *denotes the discrete interval in* $\mathbb{Z}$, *i.e.* $[0,2] = \{0,1,2\}$.

If for any $[a,b]$ involved $b < a$, we write $[a,b] = \emptyset$ with the convention that $\emptyset - \emptyset = \emptyset$. Further, $\emptyset$ in the translations results in the use of the respective entry of the original scheme without translations.

Example 6.5. Consider the discrete time derivative

$$u_1(n+2,t+1) + u_1(n,t+1) + u_2(n,t+1) - u_1(n+2,t) - u_1(n,t) - u_2(n,t)$$

and the scheme $\{u_1(n+1,t+1) - u_1(n,t), u_2(n,t+1) - u_2(n,t)\}$. Then, the stencil of the discrete time derivative equals

$$\left(\{(2,1),(0,1),(2,0),(0,0)\}, \{(0,1),(0,0)\}\right)$$

with range $\left([0,2] \times [0,1], [0,0] \times [0,1]\right)$. The stencil of the scheme equals the set of stencils of the equations in the scheme, i.e.

$$\left\{\left(\{(1,1),(0,0)\}, \emptyset\right), \left(\emptyset, \{(0,1),(0,0)\}\right)\right\}$$

with range $\left\{\left([0,1] \times [0,1], \emptyset\right), \left(\emptyset, [0,0] \times [0,1]\right)\right\}$. Hence, we get the translations for the first equation of the scheme by

$$\left([0,2] \times [0,1], [0,0] \times [0,1]\right) - \left([0,1] \times [0,1], \emptyset\right) = \left([0,1] \times [0,0], \emptyset\right)$$

and for the second equation of the scheme by

$$\Big([0,2]\times[0,1],[0,0]\times[0,1]\Big)-\Big(\emptyset,[0,0]\times[0,1]\Big)=\Big(\emptyset,[0,0]\times[0,0]\Big).$$

Therefore, we translate the first equation of the scheme by $(0,0)$ and $(1,0)$ and the second one by $(0,0)$ to get the translated scheme

$$\{u_1(n+1,t+1)-u_1(n,t),u_1(n+2,t+1)-u_1(n+1,t),u_2(n,t+1)-u_2(n,t)\}.$$

Step 3 and 4: Compute the Gröbner basis and reduce the discrete time derivative.
Polynomial ideals and Gröbner bases. We found a finite set of polynomials (the translated scheme $\{H^1,\dots,H^r\}$ and the discrete time derivative) with a finite number of instances of $u_j(n+e,t+l)$ in the previous step. Now we reduce $\mathcal{T}$ as an element of a polynomial ring by multivariate polynomial division with respect to the translated scheme. Here, we adapt the definitions and theorems from [14] to our setting.

Definition 6.6. *Let $\mathcal{K}$ be as in Definition 4.7. Let $\mathcal{R}$ be the polynomial ring generated by $\mathcal{K}$ and all instances of $u_j(n+e,t+l)$ that occur in $\{H^1,\dots,H^r\}$. We call a set $I\subseteq\mathcal{R}$ a* (polynomial) ideal *if $p_1,p_2\in I$ implies $p_1+p_2\in I$ and $p_1\in I, p_2\in\mathcal{R}$ implies $p_1p_2\in I$.*

Definition 6.7. *We denote by $\langle H^1,\dots,H^r\rangle$ the smallest (polynomial) ideal containing $\{H^1,\dots,H^r\}$.*

Given the ideal $\langle H^1,\dots,H^r\rangle$ and the discrete time derivative $\mathcal{T}\in\mathcal{R}$, we want to determine if $\mathcal{T}\in\langle H^1,\dots,H^r\rangle$ or if we can write $\mathcal{T}$ in a simpler form, using $\{H^1,\dots,H^r\}$. Hence, using multivariate polynomial division, we search for $p^1,\dots,p^r,p^{r+1}\in\mathcal{R}$ such that

$$\mathcal{T}=p^1H^1+\ldots+p^rH^r+p^{r+1}.$$

Unfortunately, the resulting remainder p^{r+1} may not be unique [14, page 14, Example 1.2.3], i.e. non-zero, although $\mathcal{T}$ belongs to $\langle H^1,\dots,H^r\rangle$. Replacing $\{H^1,\dots,H^r\}$ by a Gröbner basis for the ideal $\langle H^1,\dots,H^r\rangle$ guarantees the uniqueness of the remainder of the polynomial division. Contrary to the standard basis for the difference ideal that Gerdt's algorithm computes, a Gröbner basis for the polynomial ideal always exists and is finite. A Gröbner basis is defined up to the order of the monomials involved, so, depending on the order, we get different remainders of the polynomial division.
Polynomial reduction. In Algorithm 2, we consider two monomial orders: the (a) lexicographic order and an experimental (b) elimination order. To reduce $\mathcal{T}$, we compute the Gröbner basis of $\langle H^1,\dots,H^r\rangle$ with respect to the monomial order and then calculate the remainder of the polynomial division of $\mathcal{T}$ with respect to this Gröbner basis. The (a) lexicographic order is induced by the time-explicit ordering of the instances of $u_j(n+e,t+l)$, which gives more weight to instances at later times than to instances at earlier times. For the (b) elimination order, we first order all instances of $u_j(n+e,t+l)$ according to a pre-defined ordering. Our code uses by default the time-explicit ordering, but other orderings, like an implicit ordering or an user-defined ordering, are also possible. Then, we try to successively eliminate the instance with the highest ordering from the discrete time derivative, using the elimination order induced by a weight-matrix. This elimination might not always be possible, hence the reduction may leave the discrete time derivative unchanged. Then we repeat this process for the instance at the second latest time and continue until all instances of $u_j(n+e,t+l)$ have been eliminated. However, the elimination order is rather experimental, as its weight-matrix does not have a full rank and hence does not induce a total order on the set of monomials.
Out of the two remainders from (a) and (b), we choose the result with the least number of different instances of time.
Step 5 to 7: Compute the discrete partial variational derivative and reduce again. In the next step, we take the

resulting expression and compute its discrete partial variational derivative. Then, we repeat Step 2 to Step 4 applied to the discrete partial variational derivative.

Example 6.8. To demonstrate Algorithm 2, we use the setting from Example 5.1. We compute the discrete time derivative (Step 1)

$$\mathcal{T} = \sum_{n \in \mathcal{I}} u(n, t+1) - u(n, t).$$

For the translations (Step 2), the stencil of the scheme

$$u(n, t+1) - u(n, t) - (u(n+1, t) - 2u(n, t) + u(n-1, t))$$

is $\Big(\{(0,1), (0,0), (1,0), (-1,0)\}\Big)$ with range $\Big([-1,1] \times [0,1]\Big)$. The discrete time derivative has stencil $\Big(\{(0,1), (0,0)\}\Big)$and range $\Big([0,0] \times [0,1]\Big)$. The translations are

$$\Big([0,0] \times [0,1]\Big) - \Big([-1,1] \times [0,1]\Big) = \Big([1,-1] \times [0,0]\Big) = \Big(\emptyset \times [0,0]\Big)$$

which results in no translations, as the range of the stencil of the scheme is greater than the range of the stencil of the discrete time derivative. Hence, the Gröbner basis (Step 3) coincides with the original scheme for both monomial orders. The reduction using the lexicographic monomial order yields as remainder

$$u(n-1, t) - 2u(n, t) + u(n+1, t).$$

For the reduction using the elimination order, we first eliminate $u(n, t+1)$ and then the remaining instances of u. The elimination order yields the same remainder as the lexicographic order. Hence, we do not check for the number of instances of time (Step 4). We calculate the discrete partial variational derivative (Step 5) of

$$\sum_{n \in \mathcal{I}} u(n-1, t) - 2u(n, t) + u(n+1, t)$$

that equals 0. Hence, repeating the above procedure (Step 6) becomes unnecessary. Therefore, (Step 7) we see that the scheme conserves the quantity.

Example 6.9. We verify our result from Example 6.8, using DiscreteConservedQ, which detects automatically if we are in the time-explicit or the general setting.

```
In[4]:= variables=<|"indVars"→{n,t},"depVars"→{u},
        "scheme"→{u[n,t+1]-u[n,t]
        -(u[n+1,t]-2u[n,t]+u[n-1,t])}|>
        DiscreteConservedQ[variables][u[n,t]]

Out[4]= True
```

Remark 6.10. The algorithm can only detect conservation, but can not check if something is not conserved for sure. So a resulting False should be understood as our algorithm not detecting conservation, not as there being no conservation at all.

7 A BASIS FOR CONSERVED QUANTITIES

So far, we have discussed how to check if a quantity is preserved in time under a scheme. But it is also desirable to have a systematic way to find conserved quantities, given a scheme. Algorithm 3 finds, for a time-explicit scheme, a basis for conserved quantities that are generated by monomials up to a degree. We have implemented this algorithm in the

Algorithm 3: FindDiscreteConservedQuantityBasis

Input: right-hand side $\{H^1, \dots, H^m\} \subset S(\mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m), \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}))$ of time-explicit scheme, list of generators, degree of polynomials
Output: basis of conserved quantities

1 Build all monomials combined from the generators up to the total degree
2 Linearly combine all monomials with undetermined coefficients
3 Compute the discrete time derivative and plug in the appropriate right-hand side(s) of the scheme
4 Compute the discrete partial variational derivative
5 Simplify and use the undetermined coefficients method to determine coefficients that make the result vanish

FindDiscreteConservedQuantityBasis.

Example 7.1. We find a basis for conserved quantities that are generated by $u(n,t)$ and n and that have at most degree 3, admitted by the scheme for the heat equation from Example 5.2.

```
In[5]:= variables=<|"indVars"→{n},"depVars"→{u},
        "rhs"→{u[n]+u[n+1]-2u[n]+u[n-1]}|>
        FindDiscreteConservedQuantityBasis[variables]
        [<|"degree"→3,"generators"→{u[n],n}|>]
Out[5]= {u[n],nu[n]}
```

8 EXAMPLES AND APPLICATIONS

We illustrate our code with the inviscid Burgers equation and a system of mean-field games. These only have first-order time-derivatives, but the algorithm handles PDEs with all orders of derivatives.

Burgers equation. The inviscid Burgers equation is the PDE $u_t + uu_x = 0$ [17] in one space dimension. Any function of u is a conserved quantity. We search for discretizations preserving $\int u\, dx$.

Example 8.1. We check for conservation of mass using a forward-difference discretization:

```
In[6]:= variables=<|"indVars"→{n,t},"depVars"→{u},
        "scheme"→{u[n,t+1]-u[n,t]
        -u[n,t](u[n+1,t]-u[n,t])}|>
        DiscreteConservedQ[variables][u[n,t]]
Out[6]= False
```

For finding a scheme that preserves mass, we check for conservation under a class of schemes with a parameter.

Example 8.2. We discretize u_x by a three-point stencil with a parameter a.

```
In[7]:= variables=<|"indVars"→{n,t},"depVars"→{u},
        "pars"→{a},"scheme"→{u[n,t+1]-u[n,t]
```

```
-u[n,t](a(u[n+1,t]-u[n,t])
+(1-a)(u[n,t]-u[n-1,t]))}|>
DiscreteConservedQ[variables][u[n,t]]
```

Out[7]=

False	a==0||a==1||-1+2a≠0
True	$a==\frac{1}{2}$

Conservation holds for $a = \frac{1}{2}$, the central difference case.

So far, we have only seen schemes that are in time-explicit form, but our algorithm allows also for general schemes:

Example 8.3. We consider the scheme from the previous example but in the time-implicit version.

In[8]:=
```
variables=<|"indVars"→{n,t},"depVars"→{u},
"pars"→{a},"elimOrder"→"implicit",
"scheme"→{u[n,t+1]-u[n,t]-u[n,t+1]
(a(u[n+1,t+1]-u[n,t+1])
+(1-a)(u[n,t+1]-u[n-1,t+1]))}|>
DiscreteConservedQ[variables][u[n,t]]
```

Out[8]=

False	a==0||a==1||-1+2 a≠0
True	$a==\frac{1}{2}$

Our results also hold for the viscous Burgers equation $u_t + uu_x - u_{xx} = 0$, if we use the second-order central difference for u_{xx}.

Conserved quantities for a mean-field game. We study conserved quantities admitted by the discretization of a PDE system.

Example 8.4. In [10], Gomes et al. derived the following system

$$v_t + \left(\frac{v^2}{2} - \frac{m^2}{2}\right)_x = 0, \quad m_t - (vm)_x = 0.$$

This system admits the conserved quantities $\int v\,dx$ and $\int m\,dx$. Because this system was derived from a forward-forward mean-field game, we discretize it forward in time. We check with our code if the scheme admits the same preserved quantities as the continuous system:

In[9]:=
```
variables=<|"indVars"→{n,t},"depVars"→{v,m},
"scheme"→{v[n,t+1]-v[n,t]
+(v[n+1,t]^2-m[n+1,t]^2)/2-(v[n,t]^2-m[n,t]^2)/2,
m[n,t+1]-m[n,t]-m[n+1,t]v[n+1,t]+m[n,t]v[n,t]}|>
DiscreteConservedQ[variables][{v[n,t],m[n,t]}]
```

Out[9]= True

We reproduce this result, noting that we have a time-explicit scheme:

In[10]:=
```
variables=<|"indVars"→{n},"depVars"→{v,m},
"rhs"→{v[n]-(v[n+1]^2-m[n+1]^2)/2
+(v[n]^2-m[n]^2)/2,m[n]+m[n+1]v[n+1]-m[n]v[n]}|>
DiscreteConservedQ[variables][{v[n],m[n]}]
```

Out[10]= True

Those are the only conserved quantities for this scheme, that are polynomials up to degree 4 in v and m.

```
In[11]:= variables=<|"indVars"→{n,t},"depVars"→{v,m},
         "eqRhs"→{0.5(v[n+1,t]^2-m[n+1,t]^2)
         -0.5(v[n,t]^2-m[n,t]^2),
         m[n+1,t]v[n+1,t]-m[n,t]v[n,t]}|>
         FindConservedQuantityBasis[variables]
         [<|"degree"→4,"generators"→{v[n,t],m[n,t]}|>]

Out[11]= {m[n,t],v[n,t]}
```

Example 8.5. The related backward-forward system reads

$$-v_t + \left(\frac{v^2}{2} - \frac{m^2}{2}\right)_x = 0, \quad m_t - (vm)_x = 0.$$

This system admits the same conserved quantities as the forward-forward system, but we approximate it explicitly in time for m and implicitly for v. Our code can handle this setting, if we specify an order for the elimination of the variables, telling the code to use an explicit monomial order for m and an implicit one for v:

```
In[12]:= variables=<|"indVars"→{n,t},"depVars"→{m,v},
         "elimOrder"→"explicitimplicit",
         "scheme"→{-(v[n,t+1]-v[n,t])
         +(v[n+1,t+1]^2-m[n+1,t+1]^2)/2
         -(v[n,t+1]^2-m[n,t+1]^2)/2,
         m[n,t+1]-m[n,t]-m[n+1,t]v[n+1,t]+m[n,t]v[n,t]}|>
         DiscreteConservedQ[variables][{v[n,t],m[n,t]}]

Out[12]= True
```

9 POSSIBLE EXTENSIONS AND CONCLUDING REMARKS

9.1 Polynomial treatment of non-polynomial expressions

It may be possible to extend our methods to non-polynomial PDEs and schemes. In this case, non-polynomial expressions can be handled by writing them in polynomial form.

Translate the scheme more accurately. In our elimination procedure, we work with a specific polynomial ideal that is a subset of the difference ideal generated by the scheme. However, it could be necessary to translate the scheme more than we did in our algorithm to get cancellation. Therefore, flexible methods to determine the translations of the scheme are desirable.

9.2 Experimental check of conservation

An experimental approach, that gives a hint, if a quantity is conserved, and that can refute conservation, is the following: We fix d and N. Then, we generate experimental initial data by randomly sampling $N \cdot d \cdot m$ values, with every value corresponding to an instance of $u_j(n,t)$ for $n \in \mathcal{I}$, $1 \le j \le m$. Afterward, we calculate $u_j(n,t+1)$ for all $n \in \mathcal{I}$, $1 \le j \le m$, using the scheme. With this values, we explicitly calculate the value of the discrete time derivative. If the discrete time derivative is zero for several different samples of initial data, we might have conservation on the discrete level.

Otherwise, if we find initial data such that the discrete time derivative is not zero, we can refute conservation.

9.3 Discrete conserved quantities that are not conserved by the PDE

It may arise the question if we can find conserved quantities, that are not preserved by the original PDE. Gerdt et al. [8, 9] define the notion of s-consistency, which guarantees that all discrete quantities are also preserved in the continuous setting. One possible extension of the code would be to check if s-consistency holds for the translated scheme.

9.4 Concluding remarks

We present an algorithm for checking the conservation of a quantity under a finite-difference scheme. Our algorithm allows for systems of equations, arbitrary time and space derivatives, and both explicit and implicit schemes. Also, we implemented a function for finding conserved quantities admitted by a scheme. We use our implementation of the algorithm to analyze the conservation properties of several schemes for PDEs that arise in applications.

REFERENCES

[1] Alexander Bihlo and Roman O. Popovych. 2012. Invariant Discretization Schemes for the Shallow-Water Equations. *SIAM Journal on Scientific Computing* 34, 6 (2012), B810–B839. https://doi.org/10.1137/120861187

[2] Alexei F. Cheviakov. 2007. GeM software package for computation of symmetries and conservation laws of differential equations. *Computer Physics Communications* 176, 1 (2007), 48–61. https://doi.org/10.1016/j.cpc.2006.08.001

[3] Alexei F. Cheviakov. 2010. Computation of fluxes of conservation laws. *Journal of Engineering Mathematics* 66, 1 (2010), 153–173. https://doi.org/10.1007/s10665-009-9307-x

[4] Alexei F. Cheviakov, Vladimir A. Dorodnitsyn, and Evgeniy I. Kaptsov. 2020. Invariant conservation law-preserving discretizations of linear and nonlinear wave equations. *J. Math. Phys.* 61, 8 (2020), 081504. https://doi.org/10.1063/5.0004372

[5] Vladimir A. Dorodnitsyn and Evgeniy I. Kaptsov. 2021. Discrete shallow water equations preserving symmetries and conservation laws. *J. Math. Phys.* 62, 8 (2021), 083508. https://doi.org/10.1063/5.0031936

[6] Min Gao, Yasuyuki Kato, and Masaaki Ito. 2002. A REDUCE package for finding conserved densities of systems of nonlinear difference–difference equations. *Computer Physics Communications* 148, 2 (2002), 242–255. https://doi.org/10.1016/S0010-4655(02)00556-8

[7] Min Gao, Yasuyuki Kato, and Masaaki Ito. 2004. A REDUCE package for finding conserved densities of systems of implicit difference–difference equations. *Computer Physics Communications* 160, 1 (2004), 69–89. https://doi.org/10.1016/j.cpc.2003.12.006

[8] Vladimir P. Gerdt. 2012. Consistency Analysis of Finite Difference Approximations to PDE Systems. In *Mathematical Modeling and Computational Science*, Gheorghe Adam, Ján Buša, and Michal Hnatič (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 28–42.

[9] Vladimir P. Gerdt and Daniel Robertz. 2010. Consistency of Finite Difference Approximations for Linear PDE Systems and Its Algorithmic Verification. In *Proceedings of the 2010 International Symposium on Symbolic and Algebraic Computation* (Munich) *(ISSAC '10)*. Association for Computing Machinery, New York, 53–59. https://doi.org/10.1145/1837934.1837950

[10] Diogo A. Gomes, Levon Nurbekyan, and Marc Sedjro. 2018. Conservation Laws Arising in the Study of Forward–Forward Mean-Field Games. In *Theory, Numerics and Applications of Hyperbolic Problems I*, Christian Klingenberg and Michael Westdickenberg (Eds.). Springer International Publishing, Cham, 643–649.

[11] Diogo A. Gomes, Mher Safaryan, Ricardo de Lima Ribeiro, and Mohammed Sayyari. 2020. A surprisingly effective algorithm for the simplification of integrals and sums arising in the partial differential equations and numerical methods. http://hdl.handle.net/10754/662309

[12] Willy Hereman, Paul J. Adams, Holly L. Eklund, Mark S. Hickman, and Barend M. Herbst. 2009. Direct methods and symbolic software for conservation laws of nonlinear equations. In *Advances in nonlinear waves and symbolic computation*, Zhenya Yan (Ed.). Nova Sci. Publ., New York, 19–78, loose errata.

[13] Willy Hereman, Jan Sanders, Jack Sayers, and Jing Ping Wang. 2004. Symbolic Computation of Polynomial Conserved Densities, Generalized Symmetries, and Recursion Operators for Nonlinear Differential-Difference Equations. *CRM Proceedings and Lecture Notes* 39 (2004), 133–148.

[14] Takayuki Hibi (Ed.). 2013. *Gröbner Bases*. Springer, Tokyo Heidelberg New York Dodrecht London. https://doi.org/10.1007/978-4-431-54574-3

[15] Peter E. Hydon. 2001. Conservation laws of partial difference equations with two independent variables. *Journal of Physics A: Mathematical and General* 34, 48 (nov 2001), 10347–10355. https://doi.org/10.1088/0305-4470/34/48/301

[16] Boris A. Kupershmidt. 1985. Discrete Lax equations and differential-difference calculus. *Astérisque* 123 (1985), 212.

[17] Joel Smoller. 1994. (2nd ed.). A Series of Comprehensive Studies in Mathematics, Vol. 258. Springer-Verlag, New York Berlin Heidelberg.

[18] Inc. Wolfram Research. 2021. Mathematica, Version 13.0.0. https://www.wolfram.com/mathematica Champaign, IL.