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

Diogo A. Gomes
King Abdullah University of Science and Technology (KAUST)
CEMSE Division
AMCS Program
Thuwal, Saudi Arabia
diogo.gomes@kaust.edu.sa

Friedemann Krannich
King Abdullah University of Science and Technology (KAUST)
CEMSE Division
AMCS Program
Thuwal, Saudi Arabia
friedemann.krannich@kaust.edu.sa

Ricardo de Lima Ribeiro
King Abdullah University of Science and Technology (KAUST)
CEMSE Division
AMCS Program
Thuwal, Saudi Arabia
ricardo.ribeiro@kaust.edu.sa

ABSTRACT

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 under 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. In our algorithm, 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 by studying 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**.

KEYWORDS

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. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Contents

Conference'17, July 2017, Washington, DC, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

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.

While computations for determining conservation are often easy in simple cases, they get rather tedious for more complicated schemes or quantities. Hence, automating such computations in computer algebra systems is desirable.

An algorithm for finding conserved quantities for (continuous) differential equations was developed and implemented in Maple by Cheviakov (see [1] and [2]). Hereman et al. (see for example [10] and [9]), proposed an algorithm to compute conserved densities for semi-discretized schemes for PDEs with first-order time derivative. Their algorithm, which is implemented in a Mathematica package, uses the scaling symmetries of the scheme to construct conserved quantities and calculates their coefficients by using the discrete variational derivative. Gao et al. extended this algorithm to first-order time-explicit schemes [3] and to first-order time-implicit schemes [4]. They also implemented it in the computer algebra system Reduce.

In this paper, we propose an algorithm that checks if a quantity is conserved in time under a scheme. Conserved quantities usually involve integral expressions or their discrete analog, sums. Gomes et al. proposed algorithms for the simplification of sums in [8]. 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 in [5], that if the discrete time derivative of the quantity belongs to the difference ideal generated by the scheme, the quantity is conserved. 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 this 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 and 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 [13]. 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

Throughout this section, we follow the ideas and computations in [8]. Here, subscripts denote indices related to coordinates or tuples while superscripts denote indices related to sequences and families. To avoid dealing with boundary terms, we work with periodic functions in $\mathbb{Z}^d$.

Definition 2.1. *Let N, d and m be natural numbers. 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*

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

In the previous definition, the $e^1,\dots,e^k$ are not necessarily unit vectors and can vary between functionals. Here, we allow for smooth G. However, later we require G to be a polynomial function.

Example 2.3. Let $d = 1$, $m = 1$ and define

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

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

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

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

$$\begin{aligned}
D\mathcal{J}[u](v) &= \frac{d}{d\epsilon}\mathcal{J}[u+\epsilon v]\bigg|_{\epsilon=0} = \frac{d}{d\epsilon}\sum_{n\in\mathcal{I}} F_n[u+\epsilon v]\bigg|_{\epsilon=0} \\
&= \sum_{n\in\mathcal{I}} \frac{d}{d\epsilon} G(u(n+e^1)+\epsilon v(n+e^1),\dots,u(n+e^k)+\epsilon v(n+e^k))\bigg|_{\epsilon=0} \\
&= \sum_{n\in\mathcal{I}} D_1G(u(n+e^1),\dots,u(n+e^k))v(n+e^1) \\
&\quad + \dots + D_kG(u(n+e^1),\dots,u(n+e^k))v(n+e^k) \\
&= \sum_{n\in\mathcal{I}} D_1G(u(n),u(n+e^2-e^1),\dots,u(n+e^k-e^1))v(n) \qquad (2.1)\\
&\quad + D_2G(u(n+e^1-e^2),u(n),\dots,u(n+e^k-e^2))v(n) \\
&\quad + \dots + D_kG(u(n+e^1-e^k),u(n+e^2-e^k),\dots,u(n))v(n) \\
&= \sum_{n\in\mathcal{I}} \Big(D_1G(u(n),u(n+e^2-e^1)\dots,u(n+e^k-e^1)) \\
&\quad + D_2G(u(n+e^1-e^2),u(n),\dots,u(n+e^k-e^2)) \\
&\quad + \dots + D_kG(u(n+e^1-e^k),u(n+e^2-e^k),\dots,u(n))\Big)v(n) \\
&=: \sum_{n\in\mathcal{I}} \mathcal{V}(\mathcal{J})[u]v(n).
\end{aligned}$$

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

$$\begin{aligned}
D\mathcal{J}[u](v) &= \frac{d}{d\epsilon}\mathcal{J}[u+\epsilon v]\bigg|_{\epsilon=0} = \frac{d}{d\epsilon}\sum_{n\in\mathcal{I}} F_n[u+\epsilon v]\bigg|_{\epsilon=0} \\
&= \sum_{n\in\mathcal{I}} \frac{d}{d\epsilon} G[u(n+1)+\epsilon v(n+1), u(n)+\epsilon v(n)]\bigg|_{\epsilon=0} \\
&= \sum_{n\in\mathcal{I}} 2(u(n+1)-u(n))v(n+1) + (-2)(u(n+1)-u(n))v(n) \\
&= \sum_{n\in\mathcal{I}} 2(u(n)-u(n-1))v(n) + (-2)(u(n+1)-u(n))v(n) \\
&= \sum_{n\in\mathcal{I}} \Big(2(u(n)-u(n-1)) + (-2)(u(n+1)-u(n))\Big)v(n) \\
&= \sum_{n\in\mathcal{I}} \Big(-2u(n+1)+4u(n)-2u(n-1)\Big)v(n)
\end{aligned}$$

$$= \sum_{n\in\mathcal{I}} \mathcal{V}(\mathcal{J})[u]v(n)$$

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

Fundamental theorem

The algorithms for the simplification of sums presented in [8] 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

$$\mathcal{J}[u] := \sum_{n\in\mathcal{I}} u(n)u(n+1),$$

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

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:

$$D(\mathcal{J} - \tilde{\mathcal{J}})[u](v) =$$

$$= \sum_{n\in\mathcal{I}} (u(n+1) + u(n-1) - u(n+1) - u(n-1))v(n) = 0.$$

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 $d = 2$. It may not be obvious, that

$$\begin{gathered}\sum_{n_1,n_2} u(n_1,n_2-2)^4 - 3u(n_1,n_2-2)^3u(n_1+1,n_2-2) \\ +u(n_1,n_2)u(n_1+1,n_2)^3 + 3u(n_1,n_2-2)^2u(n_1+1,n_2-2)^2 \\ -u(n_1,n_2-2)u(n_1+1,n_2-2)^3 - u(n_1,n_2)^4 + \\ 3u(n_1,n_2)^3u(n_1+1,n_2) - 3u(n_1,n_2)^2u(n_1+1,n_2) \\ -3u(n_1,n_2-2)^2u(n_1+1,n_2-2)^2 + 3u(n_1,n_2)^2u(n_1+1,n_2) = 0.\end{gathered}$$

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*

$$\mathcal{F}(\mathcal{P}(\mathcal{I}\times\mathbb{N}^l,\mathbb{R}^m),\mathcal{D}(\mathbb{N}^l)) :=$$

$$\left\{\mathcal{J} : \mathcal{P}(\mathcal{I}\times\mathbb{N}^l) \to \mathcal{D}(\mathbb{N}^l) \,\middle|\, \mathcal{J}[u] = \sum_n F_{(n,t)}[u],\right.$$

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

Throughout this paper, all quantities, whose conservation in time we check, are functions $\mathcal{J} \in \mathcal{F}(\mathcal{P}(\mathcal{I}\times\mathbb{N}^l,\mathbb{R}^m),\mathcal{D}(\mathbb{N}^l))$. Define $\tilde{n} := (n_1,...,n_d,s_1,...,s_l) \in \mathcal{I}\times\mathbb{N}^l$. 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

$$\begin{gathered}D^l\mathcal{J}[u](v)(t_1,\dots,t_l) := \frac{d}{d\epsilon}\mathcal{J}[u+\epsilon v]\Big|_{\epsilon=0}(t_1,\dots,t_l) \\ = \sum_{\tilde{n}} D_1G(u(\tilde{n}+e^1),\dots,u(\tilde{n}+e^k))\delta_{(t_1,\dots,t_l)}(s_1,\dots,s_l)v(\tilde{n}+e^1) \\ +D_2G(u(\tilde{n}+e^1),\dots,u(\tilde{n}+e^k))\delta_{(t_1,\dots,t_l)}(s_1,\dots,s_l)v(\tilde{n}+e^2) \\ +\dots+D_kG(u(\tilde{n}+e^1),\dots,u(\tilde{n}+e^k))\delta_{(t_1,\dots,t_l)}(s_1,\dots,s_l)v(\tilde{n}+e^k) \\ = \sum_{\tilde{n}} D_1G(u(\tilde{n}),\dots,u(\tilde{n}+e^k-e^1))\delta_{(t_1,\dots,t_l)}(s_1-e^1_{d+1},\dots,s_l-e^1_{d+l})v(\tilde{n}) \\ +D_2G(u(\tilde{n}+e^1-e^2),\dots,u(\tilde{n}+e^k-e^2))\delta_{(t_1,\dots,t_l)}(s_1-e^2_{d+1},\dots,s_l-e^2_{d+l})v(\tilde{n}) \\ +\dots+D_kG(u(\tilde{n}+e^1-e^k),\dots,u(\tilde{n}))\delta_{(t_1,\dots,t_l)}(s_1-e^k_{d+1},\dots,s_l-e^k_{d+l})v(\tilde{n}) \\ = \sum_{\tilde{n}} \Big(D_1G(u(\tilde{n}),\dots,u(\tilde{n}+e^k-e^1))\delta_{(t_1+e^1_{d+1},\dots,t_l+e^1_{d+l})}(s_1,\dots,s_l) \\ +D_2G(u(\tilde{n}+e^1-e^2),\dots,u(\tilde{n}+e^k-e^2))\delta_{(t_1+e^2_{d+1},\dots,t_l+e^2_{d+l})}(s_1,\dots,s_l) \\ +\dots+D_kG(u(\tilde{n}+e^1-e^k),\dots,u(\tilde{n}))\delta_{(t_1+e^k_{d+1},\dots,t_l+e^k_{d+l})}(s_1,\dots,s_l)\Big)v(\tilde{n}) \\ =: \sum_{\tilde{n}} \mathcal{V}^l(\mathcal{J})[u](t_1,\dots,t_l)v(\tilde{n}).\end{gathered}$$

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

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

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

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,

$$\mathcal{V}(\mathcal{J} - \tilde{\mathcal{J}})[u] = 0$$

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*

$$\mathcal{S}(\mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}^m), \mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R})) :=$$
$$\Bigg\{ H : \mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}^m) \to \mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}) \ \Bigg| $$
$$H[u](n,t) = H(u(n+\tilde{e}^1, t+\tilde{l}^1), \dots, u(n+\tilde{e}^r, t+\tilde{l}^r)),$$
$$\text{where } \{\tilde{e}^1,\dots,\tilde{e}^r\} \subset \mathbb{Z}^d,\ \{\tilde{l}^1,\dots,\tilde{l}^r\} \subset \mathbb{Z} \Bigg\}.$$

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. In this work, we only consider finite-difference schemes with fixed step size $h = 1$, but one can generalize our results and algorithms to other step sizes.

Definition 4.2. *A scheme* $\{H^1,\dots,H^i\} \subset \mathcal{S}(\mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}^m), \mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}))$ *given by*

$$H^k(u(n+\tilde{e}_k^1, t+\tilde{l}_k^1),,\dots,u(n+\tilde{e}_k^r, t+\tilde{l}_k^r))$$

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

$$u_j(n,t+1) - \tilde{H}^k(u(n+\tilde{\tilde{e}}_k^1, t), u(n+\tilde{\tilde{e}}_k^2, t), \dots, u(n+\tilde{\tilde{e}}_k^r, t))$$

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

Example 4.3. Consider the forward-difference scheme for the heat equation [12]

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

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 [5], we now introduce difference ideals and how we use them in our algorithm.

Definition 4.4. *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*

$$\sigma_i^k \circ u_j(n+e, t+l) := u_j(n_1+e_1,\dots,n_i+e_i+k,\dots,t+l)$$

understanding that σ_{d+1}^k *shifts the time variable.*

Definition 4.5. *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 4.6. *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* $p_1 \cdot p_2 \in \tilde{I}$*,*
- $p_1 \in \tilde{I}, \theta \in \Theta$ *implies* $\theta \circ p_1 \in \tilde{I}$*.*

Definition 4.7. $\langle H^1,\dots,H^i\rangle$ *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. In particular, Algorithm **??** seeks to determine if the discrete time derivative $\mathcal{T} := \mathcal{J}[u](t+1) - \mathcal{J}[u](t)$ 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 ?? for checking the conservation of a quantity under the right-hand side of a scheme in time-explicit form. Our algorithm for general schemes can be found in the subsequent section.

Note, that the result of 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 from Example 4.3. The discrete time derivative (step 1) is

$$\sum_n u(n,t+1) - u(n,t).$$

Replacing $u(n,t+1)$ by the right-hand side of the scheme (step 2) gives

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

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 ??.

6 THE GENERAL CASE

In this section, we present Algorithm ?? that deals with general, not necessarily time-explicit, schemes. We explain its steps in detail below and demonstrate the algorithm in Example 6.7.

Step 1: Build the discrete time derivative

We build the discrete time derivative by subtracting $\mathcal{J}[u](t)$ from $\mathcal{J}[u](t+1)$. This task is done in our code by the TimeDifference.

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

Step 2: Translate the scheme

To make the step from difference to polynomial algebra to guarantee termination of the algorithm, we use a standard idea in symbolic computations and consider every instance of $u_j(n+e,t+l)$ as a variable of a multivariate polynomial. Hence, we compute all translations of the scheme, such that all variables in the polynomial associated with $\mathcal{T}$ appear in the polynomials associated with the translated scheme.

Example 6.2. The algorithm treats the discrete time derivative $u(n+1,t+1) - u(n+1,t)$ as the polynomial equation $x_1 - x_2$.

Definition 6.3. *Let $u = (u_1,\dots,u_m) \in \mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}^m)$ and*

$$F : \mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}^m) \to \mathcal{P}(\mathcal{I}\times\mathbb{N},\mathbb{R}),$$

$$\begin{aligned} F[u] = F(&u_1((n,t)+x^1),\dots,u_1((n,t)+x^{k_1}),\\ &u_2((n,t)+x^{k_1+1}),\dots,u_2((n,t)+x^{k_2}),\\ &\dots,u_m((n,t)+x^{k_{m-1}+1}),\dots,u_m((n,t)+x^{k_m})) \end{aligned}$$

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*

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

The range *of the stencil of F is*

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

*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. $[0,2] = \{0,1,2\}$.*

We calculate the minimal necessary translations of the scheme, such that all instances of $u_j(n+e,t+l)$, that appear in 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. We denote the resulting translated system by $\{H^1,\dots,H^r\}$. 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 6.4. 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

$$\Big(\{(2,1),(0,1),(2,0),(0,0)\},\{(0,1),(0,0)\}\Big)$$

with range

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

The stencil of the scheme equals the set of stencils of the equations in the scheme, i.e.

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

with range

$$\left\{\Big([0,1]\times[0,1],\emptyset\Big),\Big(\emptyset,[0,0]\times[0,1]\Big)\right\}.$$

Hence, we get the translations for the first equation of the scheme by

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

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 and the discrete time derivative) with a finite number of instances of $u_j(\tilde{n} + e, t + l)$ in the previous step. Now we reduce $\mathcal{T}$, using the translated scheme. Here, we adapt the definitions and theorems from [11] to our setting.

Definition 6.5. *Let $\mathcal{K}$ be as in Definition 4.6. Let $\mathcal{R}$ be the polynomial ring generated by $\mathcal{K}$ and all instances of $u_j(n + e, t + l)$ that occur in $\{H^1, \ldots, H^r, \mathcal{T}\}$. We call a set $I \subseteq \mathcal{R}$ a (polynomial) ideal if*

- *$p_1, p_2 \in I$ implies $p_1 + p_2 \in I$,*
- *$p_1 \in I, p_2 \in \mathcal{R}$ implies $p_1 p_2 \in I$.*

Definition 6.6. *We denote by $\langle H^1, \ldots, H^r \rangle$ the smallest (polynomial) ideal containing $\{H^1, \ldots, H^r\}$.*

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

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

Unfortunately, the resulting remainder p^{r+1} may not be unique [11, page 14, Example 1.2.3], i.e. non-zero, although $\mathcal{T}$ belongs to $\langle H^1, \ldots, H^r \rangle$. Replacing $\{H^1, \ldots, H^r\}$ by a Gröbner basis for the ideal $\langle H^1, \ldots, H^r \rangle$ guarantees the uniqueness of the remainder of the polynomial division. Contrary to the standard basis for the difference ideal that Gerdt's algorithm computes, the 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 **??**, we consider two monomial orders: (a) lexicographic and (b) explicit. To reduce $\mathcal{T}$, we compute the Gröbner basis of $\langle H^1, \ldots, H^r \rangle$ 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 explicit monomial order (b) results in the Gröbner basis that eliminates the instances of $u_j(n + e, t + l)$ at the latest time (with the largest l). 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 have been eliminated.
Out of the two remainders from (a) and (b), we choose the result with the least number of different instances of time. Our code also supports other elimination orders, including user-defined ones.

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 6.7. To demonstrate Algorithm **??**, we use the setting from Example 5.1. We compute the discrete time derivative (step 1)

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

For the translations (step 2), the stencil of the scheme is

$$\Big(\big\{(0,1), (0,0), (-1,0), (1,0)\big\}\Big)$$

with range $\Big([-1,1] \times [0,1]\Big)$. The discrete time derivative has stencil $\Big(\big\{(0,1), (0,0)\big\}\Big)$ 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 need to check for the number of instances of time (step 4). We calculate the discrete partial variational derivative (step 5) of

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

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

Example 6.8. We verify our result from Example 6.7, using DiscreteConservedQ, which detects automatically if we are in the time-explicit or the general setting.

Remark 6.9. 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 **??** finds, for a time-explicit scheme, a basis for conserved quantities that are generated by monomials up to a (total) degree.

We have implemented this algorithm in the 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.

8 EXAMPLES AND APPLICATIONS

We illustrate our code with the Burgers equation and a system of mean-field games.

Burgers equation

The Burgers equation is the PDE $u_t + uu_x = 0$ [12] in one space dimension. Any function of u is a conserved quantity. We are interested in discretizations that preserve mass $\int u\, dx$.

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

We want to find a scheme that preserves this quantity. Therefore we check for conservation under a class of schemes with a parameter, to find an appropriate choice for this parameter.

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

We have conservation if we choose $a = \frac{1}{2}$, corresponding to the standard central difference.

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.

Conserved quantities for a mean-field game

As a second application, we use our code to study conserved quantities admitted by the discretization of a system of PDEs.

Example 8.4. In [7], Gomes et al. derived the following system of PDEs

$$\begin{aligned} v_t + \left(\frac{v^2}{2} - \frac{m^2}{2}\right)_x &= 0 \\ m_t - (vm)_x &= 0. \end{aligned}$$

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:

We reproduce this result by noticing, that the scheme is in time-explicit form:

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

Example 8.5. The related backward-forward system reads

$$\begin{aligned} -v_t + \left(\frac{v^2}{2} - \frac{m^2}{2}\right)_x &= 0 \\ m_t - (vm)_x &= 0. \end{aligned}$$

This system admits the same conserved quantities as the forward-forward system, but we approximate it explicitly (forward) in time for m and implicitly (backward) 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:

9 POSSIBLE EXTENSIONS AND CONCLUDING REMARKS

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. For example, treat the expression $2\cos(u(n+e^1))e^{u(n+e^2)} + \frac{1}{u(n+e^3)}$ as the polynomial $2x_1x_2 + x_3$.

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.

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. ([6],[5]) define the notion of s-consistency, which guarantees that all discrete quantities are also preserved in the continuous setting. One possible extension of our code would be to check automatically if s-consistency holds for the translated scheme.

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] Alexei 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

[2] 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

[3] 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

[4] 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

[5] 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.

[6] 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

[7] Diogo 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.

[8] 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

[9] Willy Hereman, Paul Adams, Holly Eklund, Mark Hickman, and B. Herbst. 2008. Direct Methods and Symbolic Software for Conservation Laws of Nonlinear Equations. arXiv:0803.0083 [nlin.SI]

[10] 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.

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

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

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