Algorithmic detection of conserved quantities for finite-difference schemes*

Diogo A. Gomes[1][0000−0002−3129−3956], Friedemann Krannich[1][0000−0002−8642−4319], Bashayer Majrashi[1][0009−0008−3677−8064], and Ricardo de Lima Ribeiro[1][0000−0002−2707−2210]

King Abdullah University of Science and Technology (KAUST), AMCS/CEMSE Division, Thuwal 23955-6900, Saudi Arabia
diogo.gomes,friedemann.krannich,bashayer.majrashi,
ricardo.ribeiro@kaust.edu.sa

Abstract. Many partial differential equations (PDEs) admit conserved quantities like mass or energy. These are often essential for establishing well-posedness results. When approximating a PDE with a finite-difference scheme, it is crucial to determine whether related discretized quantities remain conserved by the scheme. Such conservation may ensure the stability of the numerical scheme. We present an algorithm for verifying 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 arbitrary number of variables. Additionally, we introduce an algorithm for finding conserved quantities. We illustrate our algorithm in several finite-difference schemes. Our approach incorporates parametric Göbner bases to handle parameters, ensuring accurate computation of conserved quantities.

Keywords: Finite-difference schemes · Discrete variational derivative · Discrete partial variational derivative · Conserved quantities · Implicit schemes · Explicit schemes · Parametric Groebner systems.

1 Introduction

add Comprehensive Gröbner Systems 2222 In many partial differential equations (PDEs), conserved integral quantities play a crucial role, as they often represent properties of physical significance, such as energy or mass. For example, the advection equation preserves energy and the heat equation conserves mass. Since closed-form solutions for these PDEs are often unavailable, finite-difference schemes serve as a practical numerical alternative. Hence, it becomes imperative to determine whether the chosen scheme preserves these conserved quantities. This information is not only valuable for gauging the accuracy of the approximation, but also vital for evaluating its stability. Automating conservation computations in computer algebra systems streamlines the process, saves time and enhances accuracy.

* Supported by King Abdullah University of Science and Technology (KAUST) baseline funds and KAUST OSR-CRG2021-4674.

Φ and Ψ need domains and ranges, also explain that divergence is only in x

Conserved quantities correspond to conservation laws, admitted by the PDE. A conservation law [4] is an equation of the form $\frac{\partial}{\partial t}\Phi[u] + \operatorname{div}\Psi[u] = 0$ 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.

There is a similar formulation for for conservation laws for finite difference schemes of a PDE where the derivatives in the preceding identity are replaced by shifts [15]. . As a result, standard approaches for constructing schemes with conservation involve either discretizing the continuous conservation law or finding multipliers for the scheme [6]. Alternatively, to verify that a particular scheme admits conservation laws, we can use the discrete Euler operator [4]. Kupershmidt [17, II. Theorem 31] discovered that an equation represents a discrete conservation law if and only if it is in the kernel of the discrete Euler operator This strategy has been employed in recent studies, such as by Dorodnitsyn et al. for developing a shallow water equation scheme that preserves energy [6], and by Cheviakov et al. for creating linear and nonlinear wave equation schemes that admit discrete analogs of continuous conservation laws [4]. These strategies also apply to continuous or semi-discretized PDEs. Algorithms using multipliers and the Euler operator for PDEs were developed and implemented by Cheviakov [2,3]. Hereman et al. [13,12] proposed an algorithm for computing conserved densities for semi-discretized PDE schemes with first-order time derivatives, using scaling symmetries and the discrete Euler operator. Gao et al. extended this algorithm to first-order time-explicit [7] and time-implicit schemes [8]. 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 present an algorithm that verifies if a quantity is conserved over time under a given scheme. Our approach diverges from previous methods, as we do not attempt to construct discrete Φ and Ψ using the scheme, but rather check if a given discrete Φ is conserved over time under the scheme. Typically, conserved quantities involve integral expressions or their discrete analogs, such as sums. Gomes et al. proposed algorithms for simplifying sums [11] and developed algorithms for detecting conserved quantities in PDEs and semi-discretized schemes [11]. We review these techniques in Section 2. To generalize these methods for cases without summing over all arguments of the involved functions, we introduce the discrete partial variational derivative (Section 3). Our main contribution is an algorithm for verifying the conservation of a quantity under a numerical scheme. Gerdt [9] demonstrated that the quantity is conserved if its discrete time derivative belongs to the difference ideal generated by the scheme. However, some quantities may add only to a constant and be trivially preserved without being part of the difference ideal. Furthermore, Gerdt's algorithm may not always terminate, as the Gröbner basis for the difference ideal might be infinite. We address these issues by combining the discrete partial variational derivative with a polynomial ideal (instead of a difference ideal) having a finite Gröbner basis (Section 4). Our algorithm is applicable to schemes that are explicit and implicit in time, can handle schemes with multiple higher-order time and space derivatives, multiple space dimensions, systems of equations, and schemes with parameters. As the process of finding conserved quantities involves polynomial division, special care must be taken to ensure that the division is performed correctly with respect to the parameters. We employ parametric Gröbner bases, which are crucial for the proper treatment of parameters in our algorithm, see Section 5. We have implemented this algorithm as a part of a Mathematica package [20][1]. Our examples include Burgers equation and a PDE system related to mean-field games (Section 9). In these examples, we show that our code detects conserved quantities and can be used to select suitable schemes for time-implicit and time-explicit discretizations.

2 Preliminaries

In this section, we introduce the concept of discrete torus and discrete variational derivatives, and discuss their properties and applications in simplifying sums. These definitions are essential for the analysis of methods presented in the rest of the paper. Here, we discuss the connection between discrete functionals and discrete variational derivatives and present examples that illustrate how to simplify sums using this approach. We follow the ideas in [11]. In this paper, subscripts denote coordinate or tuple indices, and superscripts denote sequence indices. To avoid boundary terms, we work with periodic functions in $\mathbb{Z}^d$.

When discretizing PDEs using finite difference schemes, we need to work with functions defined on discrete sets. For instance, periodic functions in $\mathbb{R}^d$ can be replaced by functions defined on the discrete torus, which will be introduced in the following discussion.

Definition 1. *Let N, d and m be positive integers. The* discrete torus *is the set $\mathcal{I} := \{0, \dots, N-1\}^d \subset \mathbb{Z}^d$. Let $\mathcal{P}(\mathcal{I}, \mathbb{R}^m) := \{u \mid u : \mathcal{I} \to \mathbb{R}^m\}$. Functions in $\mathcal{P}(\mathcal{I}, \mathbb{R}^m)$ extend periodically to $\mathbb{Z}^d$.*

We introduce discrete functionals in the previous function space. These functionals are the analog of integral functionals for discrete approximations of functions.

Definition 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*

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

where $e^1, \dots, e^k$ are may not be unit vectors and may vary between functionals.

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

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

The discrete variational derivative is a useful tool for simplifying sums. 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 $D\mathcal{J}[u](v) := \frac{d}{d\epsilon}\mathcal{J}[u+\epsilon v]\Big|_{\epsilon=0}$. In (2), we shift the indices because of the periodicity of u and v,

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

The operator $\mathcal{V}(\mathcal{J})$ represents the derivative $D\mathcal{J}u$ with respect to the L^2 inner product $\langle w, v\rangle = \sum_{n\in\mathcal{I}} w_n v_n$, leading to the following definition.

Definition 3. *$\mathcal{V}(\mathcal{J})$ is the* discrete variational derivative *of $\mathcal{J}$.*

Because here all functions are smooth, the discrete variational derivative exists.

Example 2. Now, we return to Example 1 and compute

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

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

The algorithms for simplification of sums in [11] rely on the following result, which gives a criterion to decide if two functionals represent the same quantity.

Theorem 1. *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. We have

$$(\mathcal{J} - \tilde{\mathcal{J}})[u] = (\mathcal{J} - \tilde{\mathcal{J}})[u] - (\mathcal{J} - \tilde{\mathcal{J}})[u_0] = \int_0^1 \frac{d}{d\lambda}(\mathcal{J} - \tilde{\mathcal{J}})[u_0 + \lambda(u - u_0)]\; d\lambda$$

$$= \int_0^1 \mathcal{V}(\mathcal{J} - \tilde{\mathcal{J}})[u_0 + \lambda(u - u_0)](u - u_0)\ d\lambda = 0$$

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

We use Theorem 1 to determine whether different sums represent the same quantity.

Example 3. Let $d = m = 1$ and consider the functionals $\mathcal{J}[u] := \sum_{n\in\mathcal{I}} u(n)u(n+1)$ and $\tilde{\mathcal{J}}[u] := \sum_{n\in\mathcal{I}} u(n-2)u(n-1)$. By shifting n, we see that $\mathcal{J}$ and $\tilde{\mathcal{J}}$ represent the same quantity . Accordingly, $\mathcal{J}[0] = 0 = \tilde{\mathcal{J}}[0]$, and the discrete variational derivative must vanish

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

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

Example 4. Let $d = 2$ and $m = 1$. It may not be obvious, that

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

$$+ u(n_1, n_2)u(n_1 + 1, n_2)^3 + 3u(n_1, n_2 - 2)^2 u(n_1 + 1, n_2 - 2)^2$$

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

$$3u(n_1, n_2)^3 u(n_1 + 1, n_2) - 3u(n_1, n_2)^2 u(n_1 + 1, n_2)$$

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

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

3 The discrete partial variational derivative

In this section, we introduce the concept of the discrete partial variational derivative, a generalization of the discrete variational derivative for cases where one or several arguments of the indices are kept constant. This generalization provides a more flexible approach to handling various problems in conservation laws for finite-difference schemes, addressing challenges and limitations of the traditional discrete variational derivative. The main result of this section is Theorem 2, which gives a criterion for equality of two functionals based on their discrete partial variational derivatives."

Let $n = (n_1, \dots, n_d)$, the space variables, be the variables for the functions in $\mathcal{P}(\mathcal{I}, \mathbb{R}^m)$. Let $l \in \mathbb{N}$ and call $t := (t_1, \dots, t_l) \in \mathbb{N}^l$ the time variables. For generality, we consider the vectorial time case, while later in this paper, we focus on the case when $l = 1$. The analog to Definition 1 is the following.

Definition 4. *Let $\mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m) := \{u \mid u(\cdot, t) \in \mathcal{P}(\mathcal{I}, \mathbb{R}^m)\ \forall t \in \mathbb{N}^l\}$. Functions in $\mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m)$ extend to $\mathbb{Z}^d \times \mathbb{N}^l$ through periodicity in the space variables. Let $\mathcal{D}(\mathbb{N}^l) := \{u \mid u : \mathbb{N}^l \to \mathbb{R}\}$.*

We can regard $\mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m)$ as the set of functions depending in space and time and $\mathcal{D}(\mathbb{N}^l)$ as the functions depending only on time.

Next, we examine functions that involve summation only over spatial variables, extending the concept presented in Definition 2.

Definition 5. *The space of functions* $\mathcal{F}(\mathcal{P}(\mathcal{I} \times \mathbb{N}^l, \mathbb{R}^m), \mathcal{D}(\mathbb{N}^l))$ *is the set*

$$\Bigg\{ \mathcal{J} : \mathcal{P}(\mathcal{I} \times \mathbb{N}^l) \to \mathcal{D}(\mathbb{N}^l) \Bigg| \; \mathcal{J}[u](t) = \sum_{n \in \mathcal{I}} F_{(n,t)}[u],$$

$$\textit{where } F_{(n,t)}[u] = G(u((n,t) + e^1), u((n,t) + e^2), \dots, u((n,t) + e^k)),$$

$$\textit{and } \{e^1, \dots, e^k\} \subset \mathbb{Z}^{d+l}, G \textit{ is a polynomial in } k \textit{ variables} \Bigg\}.$$

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

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

Definition 6. $\mathcal{V}^l(\mathcal{J})$ *is the* discrete partial variational derivative *of* $\mathcal{J}$. *Note that* $\mathcal{V}^0 = \mathcal{V}$.

Example 5. The discrete partial variational derivative of

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

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

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

$$\sum_{n,s} v(n, s+1) \delta_t(s) - v(n, s) \delta_t(s) = \sum_{n,s} (\delta_{t+1}(s) - \delta_t(s)) v(n, s).$$

Example 6. In our Mathematica implementation, Example 5 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]}
```

The analog to Theorem 1 for the discrete partial variational derivative is:

Theorem 2. *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 1.

4 Difference schemes and algebra

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

Definition 7. *The* space of schemes, $\mathcal{S}(\mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m), \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}))$ *is the set*

$$\Bigg\{ H : \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m) \to \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}) \;\Bigg|$$

$$H[u](n,t) = H(u(n + \tilde{e}^1, t + \tilde{l}^1), \ldots, u(n + \tilde{e}^r, t + \tilde{l}^r)),$$

$$\textit{where } \{\tilde{e}^1, \ldots, \tilde{e}^r\} \subset \mathbb{Z}^d,\; \{\tilde{l}^1, \ldots, \tilde{l}^r\} \subset \mathbb{Z}, H \textit{ is polynomial} \Bigg\}.$$

A scheme *is a set of functions $\{H^1, \ldots, H^i\} \subset \mathcal{S}(\mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m), \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}))$ that represent the equations $\{H^1[u] = 0, \ldots, H^i[u] = 0\}$ 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 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, our results can be generalized to schemes with other step sizes.

Example 7. Consider the heat equation $u_t - u_{xx} = 0$ [18]. A possible scheme is $u(n, t+1) - u(n, t) - (u(n+1, t) - 2u(n, t) + u(n-1, t))$. A solution of the corresponding numerical approximation is a function $u \in \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R})$, that satisfies $u(n, t+1) - u(n, t) - (u(n+1, t) - 2u(n, t) + u(n-1, t)) = 0$ for all $n \in \mathcal{I}$ and $t \in \mathbb{N}$.

Definition 8. *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* $u_j(n,t+1)-\tilde{H}^j(u(n+\tilde{\tilde{e}}^1_j,t),u(n+\tilde{\tilde{e}}^2_j,t),\dots,u(n+\tilde{\tilde{e}}^r_j,t))$ *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 8. Consider the forward-difference scheme for the heat equation in Example 7. 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 [9], we now introduce difference ideals and how we use them in our algorithm.

Definition 9. *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* $\sigma^k_i\circ u_j(n,t):=u_j(n_1,\dots,n_{i-1},n_i+k,n_{i+1},\dots,t)$ *understanding that* σ^k_{d+1} *shifts the time variable.*

Definition 10. *The* set of all possible shifts Θ *is* $\Theta:=\{\sigma^{k_1}_1\circ\cdots\circ\sigma^{k_{d+1}}_{d+1}\mid k_1,\dots,k_{d+1}\in\mathbb{Z}\}$.

We now construct the difference ideal containing the scheme.

Definition 11. *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 12. $\langle H^1,\dots,H^i\rangle$ *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. As a result, every element of the difference ideal generated by the scheme vanishes under u. If the polynomial associated with the discrete time derivative $\mathcal{T}:=\mathcal{J}[u](t+1)-\mathcal{J}[u](t)$ belongs to the ideal $\langle H^1,\dots,H^i\rangle$, the corresponding functional vanishes. Gerdt introduced the concept of a standard Gröbner basis for the ideal $\langle H^1,\dots,H^i\rangle$, which might not be finite. To address this issue, we use polynomial algebra instead of difference algebra and a smaller polynomial ideal contained within the difference ideal that has a finite standard basis.

However, ideal membership is not a necessary condition for conservation. Some functions, $\mathcal{J}$, may have sums that add to zero even if they do not belong to the difference ideal. Thus, we combine the discrete partial variational derivative with polynomial ideals. This idea is implemented in Algorithm 2 in Section 7.

5 Parametric Groebner Systems

Here, we review essential concepts for parametric polynomial ideals. These include parametric rings, specialized ideals, and comprehensive Gröbner systems, which are built upon Gröbner basis theory. We discuss the advantages of comprehensive Gröbner systems compared with standard Gröbner basis. Furthermore, we expand on the ideal membership problem to encompass parametric ideals.

Definition 13. *The ring of parametric polynomials $\mathcal{K}[a_1, .., a_s][x_1, ..., x_n]$ in a field $\mathcal{K}$ is a ring of polynomials in the variables $x_1, ..., x_n$ with coefficients that are polynomials in the ring $\mathcal{K}[a_1, .., a_s]$. To simplify, we denote this ring by $\mathcal{K}[A][X]$.*

Definition 14. *Let L be an algebraically closed field extension of $\mathcal{K}$ and let $\sigma : \mathcal{K}[A][X] \to L[X]$ be a ring homomorphism. σ is a specialization given by a polynomial evaluation on the parameters and an identity on the variables, i.e., for $P \in \mathcal{K}[A][X]$ σ is given by $P \to P(a_{1_0}, .., a_{s_0})$, for some $(a_{1_0}, .., a_{s_0}) \in L^s$.*

It is easier to think about specializations as a choice of parameters. Comprehensive Gröbner systems extend Gröbner basis to parametric polynomials.

Definition 15. *Let $\mathcal{I}$ be a finitely generated, non-zero ideal in the ring $\mathcal{K}[A][X]$ with a fixed monomial order $\prec$ on the variables. Let $\{A_1, \ldots, A_k\}$ be a family of locally closed sets that forms a partition of L^m , such that the disjoint union $\bigcup_{i=1}^{k} A_i = L^m$, and $\{G_1, \ldots, G_k\}$ be a family of subsets in $\mathcal{K}[A][X]$. The system,*

$$\begin{cases} G_1 & A_1 \\ \vdots & \vdots \\ G_k & A_k \end{cases} \tag{1}$$

is a comprehensive Gröbner system (CGS) of I if for all specializations $\sigma_i \in A_i, \sigma_i(G_i)$ is a Gröbner basis of the specialized ideal $\sigma_i(I) \subset \mathcal{K}[X]$. Where each pair $\{G_i, A_i\}$ is called a branch. [16]

Here, the notation $\sigma_i(\mathcal{F})$, with $\mathcal{F} = \{f_1, f_2, \ldots, f_k\}$, is the specialization σ_i of each f_j such that $\sigma_i(\mathcal{F}) = \{\sigma_i(f_j) \mid f_j \in \mathcal{F}\}$ and "$\sigma_i \in A_i$" stands for a specialization given by $P \to P(a_{1_0}, \ldots, a_{m_0})$ for $(a_{1_0}, \ldots, a_{m_0}) \in A_i$. In this paper, A_i is of the form $V(p_1, \ldots, p_s)\backslash V(q_1, \ldots, q_t) \subset L^m$. Where $p_1, \ldots, p_s, q_1, \ldots, q_t \in \mathcal{K}[A]$ and $V(p_1, \ldots, p_s)$ is a variety.

A proof of existence and an algorithm to compute CGS appeared first in [19].

Theorem 3 ([19]). *Let $\mathcal{I} \subseteq \mathcal{K}[A][X]$. Then, there exists a CGS .*

Here, to compute CGS, we use a parametric implementation of Buchberger's algorithm which is illustrated in the following examples.

I think we need to explain briefly the parametric implementation. Basically we use Buchberger algorithm but branch every time we divide by a coefficient of a polynomial, considering the cases where this coefficient vanishes and when it does not.

Example 9. Let $\mathcal{F} = \left\{ax^2 + \left(a^2 - 2ab\right)xy, b^2y^2 - x\right\}$, and $\mathcal{I} \subset \mathbb{R}[a,b][x,y]$ be the ideal generated by $\mathcal{F}$. The comprehensive Gröbner system of $\mathcal{I}$ with a Lexicographic order $y \prec x$ is: we use our code here

I had issues running the code, will check again once online

The idea of a comprehensive Gröbner system is to provide a "parameter sensitive Gröbner basis". It branches the cases of the parameters that affect leading terms. For instance, in the previous example, if $a = 0$, then the leading term ax^2 vanishes which leads to a different Gröbner basis. Therefore, the (CGS) considers cases of the leading coefficients parameters vanishing.

Example 10. Let $\mathcal{F} = \left\{a^2x - y, -a + b^2xy - y^2\right\}$, and $\mathcal{I} \subset \mathbb{R}[a,b][x,y]$ be the ideal generated by $\mathcal{F}$. The comprehensive Gröbner system of $\mathcal{I}$ with a Lexicographic order $y \prec x$ is:

we use our code here

Remark 1. If a polynomial ideal contains a non-zero constant polynomial, then it is called a unit ideal $\langle 1 \rangle$. Unit ideals can generate all polynomials in the ring, so $\langle 1 \rangle = \mathcal{K}[A][X]$. As a convention, the Gröbner basis of such ideals is $\{1\}$.

Issues arise when computing the Gröbner basis of $\mathcal{F}$ without considering the parameters; that is, computing Gröbner basis assuming the non-zero coefficients. This assumption guarantees a Gröbner basis only for one set of specializations $\mathbb{R}^2 \backslash V(a,b)$, i.e., $\sigma(\mathcal{G})$ is a Gröbner basis of the specialized ideal $\sigma(\mathcal{I})$ only for $\sigma \in \mathbb{R}^2 \backslash V(a,b)$. In general $\sigma(\mathcal{G})$ is not necessarily a Gröbner basis.

Example 11. Let $\mathcal{I} = \langle ax^2 - y, -2 + bxy \rangle$. The corresponding Gröbner basis is $\mathcal{G} = \left\{b^2y^3 - 4a, 2ax - by^2, bxy - 2\right\}$. Consider the specialization σ given by $P \to P(0,1)$. The set $\sigma(\mathcal{G}) = \left\{y^3, -y^2, xy - 2\right\}$ is not a Gröbner basis. This can be shown by the Buchberger Criterion (see Chapter 2 in[5]).

Consequently, the algorithm utilized to resolve the ideal membership problem in Chapter 2 of [5] can not be used for all specializations.

Example 12. Consider the ideal $\mathcal{I}$ in the above example. The corresponding Gröbner basis computed under the assumption of non-zero coefficients is

$$\sigma_0(\mathcal{G}) = \left\{b^2y^3 - 4a, 2ax - by^2, bxy - 2\right\}$$

where $\sigma_0 \in \mathbb{R}^2 \backslash V(a,b)$. Now, it is possible to check if a polynomial $f \in \mathbb{R}[x,y]$ belongs to $\sigma(\mathcal{I})$ only for σ_0 because we know that $\sigma_0(\mathcal{G})$ is a Gröbner basis. For other specializations such as the specialization σ given by $P \to P(0,1)$, $\sigma(\mathcal{G})$ is not a Gröbner basis as shown in the above example. Therefore, we can not proceed with the division algorithm as we can not guarantee a unique remainder.

Using CGS can resolve these problems. Problem 1 is fixed by the definition of (CGS), as we have $\sigma_i(G_i)$ a Gröbner basis for any specialization σ_i. Moreover, to discuss problem 2, we need to look at the parametric ideal membership first.

Is this a definition? DG: I think now it is better Let $f \in \mathcal{K}[A][X]$. We say that $f \in \mathcal{I}$ under $\sigma \in L^m$ if $\sigma(f) \in \sigma(\mathcal{I})$.

To determine $f \in \mathcal{I}$, we consider the (CGS) of $\mathcal{I}$ given 1. We check if $f \in \sigma_i(\mathcal{I})$ for each $i \in \{1, .., k\}$ and $\sigma_i \in A_i$. For each $\sigma_i(\mathcal{I})$, we have the corresponding Gröbner basis $\sigma_i(G_i)$. Thus, we divide f by the $\sigma_i(G_i)$. If the remainder is zero for some $i \in \{1, .., k\}$, then $f \in \sigma_i(\mathcal{I})$. Therefore, $f \in \mathcal{I}$. However, if the remainder is non-zero for all i, we conclude that $f \notin \mathcal{I}$.

Example 13. Let $\mathcal{F} = \{a^2x - y, -b^2xy^2 - x\}$ and let $\mathcal{I} \subset \mathbb{R}[a, b][x, y]$ be the ideal generated by $\mathcal{F}$. We will determine if $f(x, y) = -x + y^2 \in \mathcal{I}$. We compute the corresponding (CGS) add example 3
First let $A_1 = \{(a, b) \mid a \neq 0, b \neq 0\}$ and $A_2 = \{(a, b) \mid ab = 0\}$. We divide f by $\{\frac{y}{b^2} + y^3, x - \frac{y}{a^2}\}$ and we obtain the remainder $-\frac{y}{a^2} + y^2$ and since the exists no specialization in A_1 such that this remainder is zero we conclude that $f \notin \sigma_1(\mathcal{I})$ for $\sigma_1 \in A_1$. Now we repeat the reduction of f but with the second Gröbner basis $\{x, y\}$. We obtain a zero remainder, which implies that $f \in \sigma_2(\mathcal{I})$ for $\sigma_2 \in A_2$. Therefore, $f \in \mathcal{I}$ under a specialization $\sigma_2 \in A_2$.

6 The time-explicit case

In this section, we introduce Algorithm 1 for checking the conservation of a quantity for time-explicit schemes. We discuss the algorithm for general schemes in the next section. 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.

```
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
         $\{H^1, \ldots, H^m\} \subset \mathcal{S}(\mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m), \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}))$ 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 $\{H^1, \ldots, H^m\}$
3 Compute the discrete partial variational derivative
4 If the result is zero, conservation is True, else False
```

Example 14. We examine the conservation of $\sum_n u(n, t)$ under the following 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 15. We verify the result from Example 14, 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
```

ADD HERE TIME EXPLICIT EXAMPLE WITH PARAMETERS

7 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 19.

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

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 performed in our code using TIMEDIFFERENCE.

Example 16. 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. Now, we transition from difference algebra with infinite independent variables to polynomial algebra with finite variables to ensure the finiteness of the Gröbner basis. Hence, we need to compute a finite number of translations of the scheme and consider every instance of $u_j(n+e, t+l)$ that appears in either the translated scheme or the discrete time derivative as an independent polynomial variable. 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 17. 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 16. *Let* $u = (u_1, \dots, u_m) \in \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m)$ *and*

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

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* $\Big(\{x^1, \dots, x^{k_1}\}, \{x^{k_1+1}, \dots, x^{k_2}\}, \dots, \{x^{k_{m-1}+1}, \dots, x^{k_m}\}\Big)$. *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 we have $b < a$, 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 18. 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. $\Big\{\Big(\{(1,1),(0,0)\},\emptyset\Big),\Big(\emptyset,\{(0,1),(0,0)\}\Big)\Big\}$ with range $\Big\{\Big([0,1]\times[0,1],\emptyset\Big),\Big(\emptyset,[0,0]\times[0,1]\Big)\Big\}$. 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 Gröbner basis and reduce the discrete time derivative.

Polynomial ideals and Gröbner bases. In the previous step, we identified a finite set of polynomials comprising the translated scheme $H^1,\ldots,H^r$ and the discrete time derivative, each containing a finite number of instances of $u_j(n+e,t+l)$. We now proceed to reduce $\mathcal{T}$ by treating it as an element of a polynomial ring and performing multivariate polynomial division with respect to the translated scheme. To accomplish this, we adapt the definitions and theorems presented in [14] to suit our specific context.

Definition 17. *Let $\mathcal{K}$ be as in Definition 11. Let $\mathcal{R}$ be the polynomial ring generated by $\mathcal{K}$ and all instances of $u_j(n+e,t+l)$ that occur in $\{H^1,\ldots,H^r\}$. 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 18. *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 $\mathcal{T}=p^1H^1+\ldots+p^rH^r+p^{r+1}$. 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 $\langle H^1,\ldots,H^r\rangle$. 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, 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 employ two monomial orders: (a) the lexicographic order and (b) an experimental elimination order. To reduce $\mathcal{T}$, we first compute the Gröbner basis of $\langle H^1,\ldots,H^r\rangle$ with respect to the chosen monomial order and subsequently calculate the remainder resulting from the polynomial division of $\mathcal{T}$ with respect to this Gröbner basis.

The lexicographic order (a) is induced by the time-explicit ordering of the instances of $u_j(n+e, t+l)$, giving greater importance to instances at later times over those at earlier times. For the elimination order (b), we initially order all instances of $u_j(n+e, t+l)$ based on a predefined ordering. By default, our code employs the time-explicit ordering, but other options, such as implicit or user-defined orderings, can also be used. We then attempt to sequentially eliminate the instance with the highest ordering from the discrete time derivative, utilizing the elimination order induced by a weight-matrix. This elimination may not always be feasible, resulting in an unchanged discrete time derivative. In such cases, we repeat the process for the instance at the second latest time and continue until all instances of $u_j(n+e, t+l)$ have been eliminated.

It should be noted that the elimination order (b) is experimental in nature, as its weight-matrix lacks full rank and therefore does not induce a total order on the set of monomials. Finally, we select the result with the fewest distinct time instances from the two remainders produced by the lexicographic order (a) and the elimination order (b).

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 19. To illustrate Algorithm 2, we use the setting of Example 14. 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 $\Big(\big\{(0, 1), (0, 0), (1, 0), (-1, 0)\big\}\Big)$ 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 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 20. We verify our result from Example 19, using DiscreteConservedQ, which detects automatically if the scheme is time-explicit or not.

```
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 2. The algorithm is designed to identify conservation, but it cannot definitively confirm the absence of conservation. Therefore, a False result should be interpreted as the algorithm being unable to detect conservation, rather than an absolute assertion of non-conservation. The reason for this is that we are working with a Gröbner basis that does not generate the full difference ideal.

add an example with parameters

8 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 FindDiscreteConservedQuantityBasis.

```
Algorithm 3: FindDiscreteConservedQuantityBasis
  Input: right-hand side set of polynomials
         ${H^1, \dots, H^m} \subset S(\mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}^m), \mathcal{P}(\mathcal{I} \times \mathbb{N}, \mathbb{R}))$ ofa time-explicit
         scheme, list of generators, degree of polynomials
  Output: basis of conserved quantities
  Construct all monomials combined from the generators up to the total
    degree
  Form linear combinations of all monomials with undetermined coefficients
  Compute the discrete time derivative and substitute in the appropriate
    right-hand side(s) polynomials of the scheme
  Compute the discrete partial variational derivative
  Simplify the expression and apply the undetermined coefficients method to
    find coefficients that make the result vanish
```

Example 21. In the context of the heat equation from Example 15, we aim to find a basis for conserved quantities generated by u and n with degree up to 3.

```
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]}
```

The first conserved quantity corresponds to mass conservation and the second to center of mass conservation.

9 Examples and applications

We demonstrate our code's capabilities using two examples: the inviscid Burgers equation and a mean-field game system. Through these examples, we examine different discretizations, evaluate their conservation properties, and identify conserved quantities. Although the examples presented feature only first-order time derivatives, the algorithm can handle PDEs with derivatives of any order.

Burgers equation. In the first example, we focus on the inviscid Burgers equation, a well-known PDE with the form $u_t + uu_x = 0$ [18]. This PDE preserves the total mass $\int u \; dx$. Our first example demonstrates how the code verifies mass conservation by a forward-difference discretization. Next, we demonstrate the algorithm's ability to find a mass-preserving scheme by investigating conservation properties under a class of schemes involving a parameter. Finally, we explore time-implicit versions of the schemes and their conservation properties.

The inviscid Burgers equation is $u_t + uu_x = 0$ [18]. Any function of u is a conserved quantity. We search for discretizations preserving $\int u \; dx$.

Example 22. 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 23. We discretize u_x with 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==1/2
```

The conservation property holds when $a = \frac{1}{2}$, 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 24. 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
                           1
          True          a==─
                           2
```

Conserved quantities for a mean-field game. We analyze the conserved quantities preserved by the discretization of a PDE system in the context of mean-field games. The examples showcase our algorithm's effectiveness in identifying conserved quantities for various discretization schemes, including time-explicit, time-implicit, and mixed schemes.

Example 25. In [10], Gomes et al. derived the following system $v_t + \left(\frac{v^2}{2} - \frac{m^2}{2}\right)_x = 0$, and $m_t - (vm)_x = 0$. 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 replicate this result, observing that the scheme used here is time-explicit:

```
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, which 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 26. The related backward-forward system is $-v_t + \left(\frac{v^2}{2} - \frac{m^2}{2}\right)_x = 0$ and $m_t - (vm)_x = 0$. 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 by specifying an order for variable elimination, using an explicit monomial order for m and an implicit order 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
```

10 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. add remark on parametric groebner basis

References

1. Bihlo, A., Popovych, R.O.: Invariant discretization schemes for the shallow-water equations. SIAM Journal on Scientific Computing **34**(6), B810–B839 (2012)
2. Cheviakov, A.F.: Gem software package for computation of symmetries and conservation laws of differential equations. Computer Physics Communications **176**(1), 48–61 (2007)
3. Cheviakov, A.F.: Computation of fluxes of conservation laws. Journal of Engineering Mathematics **66**(1), 153–173 (2010)
4. Cheviakov, A.F., Dorodnitsyn, V.A., Kaptsov, E.I.: Invariant conservation law-preserving discretizations of linear and nonlinear wave equations. Journal of Mathematical Physics **61**(8), 081504 (2020)
5. Cox, D.A., Little, J., O'Shea, D.: Ideals, varieties, and algorithms. Undergraduate Texts in Mathematics, Springer, Cham, fourth edn. (2015), an introduction to computational algebraic geometry and commutative algebra
6. Dorodnitsyn, V.A., Kaptsov, E.I.: Discrete shallow water equations preserving symmetries and conservation laws. Journal of Mathematical Physics **62**(8), 083508 (2021)

7. Gao, M., Kato, Y., Ito, M.: A reduce package for finding conserved densities of systems of nonlinear difference–difference equations. Computer Physics Communications **148**(2), 242–255 (2002)
8. Gao, M., Kato, Y., Ito, M.: A reduce package for finding conserved densities of systems of implicit difference–difference equations. Computer Physics Communications **160**(1), 69–89 (2004)
9. Gerdt, V.P.: Consistency analysis of finite difference approximations to pde systems. In: Adam, G., Buša, J., Hnatič, M. (eds.) Mathematical Modeling and Computational Science. pp. 28–42. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
10. Gomes, D.A., Nurbekyan, L., Sedjro, M.: Conservation laws arising in the study of forward–forward mean-field games. In: Klingenberg, C., Westdickenberg, M. (eds.) Theory, Numerics and Applications of Hyperbolic Problems I. pp. 643–649. Springer International Publishing, Cham (2018)
11. Gomes, D.A., Safaryan, M., Ribeiro, R.d.L., Sayyari, M.: A surprisingly effective algorithm for the simplification of integrals and sums arising in the partial differential equations and numerical methods (2020)
12. Hereman, W., Adams, P.J., Eklund, H.L., Hickman, M.S., Herbst, B.M.: Direct methods and symbolic software for conservation laws of nonlinear equations. In: Yan, Z. (ed.) Advances in nonlinear waves and symbolic computation, pp. 19–78, loose errata. Nova Sci. Publ., New York (2009)
13. Hereman, W., Sanders, J., Sayers, J., Wang, J.P.: Symbolic computation of polynomial conserved densities, generalized symmetries, and recursion operators for nonlinear differential-difference equations. CRM Proceedings and Lecture Notes **39**, 133–148 (2004)
14. Hibi, T. (ed.): Gröbner Bases. Statistics and Software Systems, Springer, Tokyo Heidelberg New York Dodrecht London (2013)
15. Hydon, P.E.: Conservation laws of partial difference equations with two independent variables. Journal of Physics A: Mathematical and General **34**(48), 10347–10355 (nov 2001)
16. Kapur, D.: Comprehensive Gröbner basis theory for a parametric polynomial ideal and the associated completion algorithm. J. Syst. Sci. Complex. **30**(1), 196–233 (2017)
17. Kupershmidt, B.A.: Discrete Lax equations and differential-difference calculus. Astérisque (123), 212 (1985)
18. Smoller, J.: A Series of Comprehensive Studies in Mathematics, vol. 258. Springer-Verlag, New York Berlin Heidelberg, 2nd edn. (1994)
19. Weispfenning, V.: Comprehensive Gröbner bases. J. Symbolic Comput. **14**(1), 1–29 (1992)
20. Wolfram Research, I.: Mathematica, version 13.0.0 (2021), Champaign, IL