Symbolic-numeric algorithm for intersecting linear differential varieties

Siyuan Deng[1] Zahra Mohammadi[2] Greg Reid[1]

[1]Univ. of Western Ontario

[2]Univ. of Guelph

SYNASC 2022

Motivation

Under and over determined systems of differential equations arise in applications and have hidden constraints. We can determine those constraints by prolongation and projection. Usually, models in application can be partitioned to two parts: exact and approximate. For example, consider the equation $\nabla^2 u = f(x, y, z)$. The left hand side of the following equation is the gravity potential, which is exact, where the right hand side is the density of instellar gas, which is approximate, since it is derived from data. So it is natural to exploit exact and approximate structure.

Prolongation and projection

If we consider $\underset{i}{u}$ as the ith order derivatives of u,

1. single prolongation:

$$D(R) = \left\{ (x, \underset{0}{u}, \ldots, \underset{q+1}{u}) \in J^{q+1} : R = 0,\ D_{x^1} R = \cdots = D_{x^n} R = 0 \right\}$$

2. single projection:

$$\pi(R) = \left\{ (x, \underset{0}{u}, \ldots, \underset{q-1}{u}) \in J^{q-1} : R(x, \underset{0}{u}, \underset{1}{u}, \ldots, \underset{q}{u}) = 0 \right\}.$$

3. multiple prolongation and projection: done by iteration

Geometric involutive form(GIF)

1. **Input:** linear approximate system & initial data list
2. prolongation & substitution of rif-form of exact subsystem
3. geometric involutive basis
4. **Output:** matrices incluidng info of dimension of kernal, row space, etc.

Joint exact and approximate

Suppose we have hybrid system R. Now we can partition it into exact part and approximate part. The exact subsystem is a general PDE system, and we can apply differential elimination, here we use **rifsimp**, a alreday-defined algorithm. Then we can apply our geometric invoulutive form to the approximate subsystem. We need to amalgamate these different methods, by using geometric invariants, such as differential Hilbert function(DHF). Apply DHF to exact part then we can use derived info to seek joint GIF.

Algorithm 1

Algorithm 1 SplitExactApprox

Input: Disjoint systems exact system ExSys, approximate system ApSys and a flag.

Output: [rExSys, SimpApSys, flag]
where rExSys is in rif-form, SimpApSys is an approximate system simplified with respect to rExSys.

1: Find the rif-form of ExSys w.r.t an orderly ranking:
rExSys := rif(ExSys)
2: Simplify the approximate system w.r.t the exact system:
SimpApSys := dsubs(rExSys, ApSys)
3: ExSimpApSys := ExactSystem(SimpApSys)
4: **if** ExSimpApSys = ∅ **then** flag:=**false**
else flag:=**true**
end if
5: ExSys := rExSys ∪ ExSimpApSys
6: ApSys := SimpApSys \ ExSimpApSys
7: **return** [ExSys, ApSys, flag]

Hybrid system of Poisson equation

Suppose we have an equation, with right hand side defined as approximate.

$$u_{xx} + u_{yy} + u_{zz} = \frac{1}{2}(G(x, y, z + 0.001) + G(x, y, z - 0.001)) \tag{1}$$

The linearized form of local Lie symmetry group is:

$$\begin{aligned} \tilde{x} &= x + \xi(x, y, z, u)\epsilon + O(\epsilon^2) \\ \tilde{y} &= y + \eta(x, y, z, u)\epsilon + O(\epsilon^2) \\ \tilde{z} &= z + \zeta(x, y, z, u)\epsilon + O(\epsilon^2) \\ \tilde{u} &= u + \phi(x, y, z, u)\epsilon + O(\epsilon^2) \end{aligned} \tag{2}$$

Determining the components ξ, η, ζ, ϕ of (2) leads a linear homogeneous system called determining equations [1,4]. Some existing computer algebra implementations are [6,2,3,5].

$$\begin{aligned} R = [&\phi_u - \frac{\phi}{u} = 0, \eta_u = 0, \eta_{u,u} = 0, \xi_u = 0, \xi_{u,u} = 0, \zeta_u = 0, \\ &-2\eta_y + 2\zeta_z = 0, -2\eta_{x,u} - 2\xi_{y,u} = 0, -2\eta_{y,u} + \phi_{u,u} = 0, \\ &-2\xi_{x,u} + \phi_{u,u} = 0, -2\zeta_x - 2\xi_z = 0, -2\zeta_y - 2\eta_z = 0, \\ &-2\zeta_{y,u} - 2\eta_{z,u} = 0, -2\zeta_{z,u} + \phi_{u,u} = 0, -2\eta_x - 2\xi_y = 0, \\ &-2\xi_x + 2\zeta_z = 0, \zeta_{u,u} = 0, -2\zeta_{x,u} - 2\xi_{z,u} = 0, \\ &-2\zeta_{y,u} - 2\eta_{z,u} = 0, -2\zeta_{z,u} + \phi_{u,u} = 0, \\ &-\eta_u G - \eta_{x,x} - \eta_{y,y} + 2\phi_{y,u} - \eta_{z,z} = 0, \\ &-\xi_u G - \xi_{x,x} + 2\phi_{x,u} - \xi_{y,y} - \xi_{z,z} = 0, \\ &-3\zeta_u G - \zeta_{x,x} - \zeta_{y,y} + 2\phi_{z,u} - \zeta_{z,z} = 0, \\ &\phi_{x,x} + \phi_{y,y} + \phi_{z,z} - \eta G_y - \zeta G_z - \xi G_x + \phi_u G - 2\zeta_z G = 0] \end{aligned}$$

Algorithm 2

Algorithm 2 HybridGeometricInvolutiveForm

Input: Linear Homogeneous differential system R.

Output: Geometric Involutive Form for system R

1: Lines 1 to 5: split the system into ExSys and ApSys
ExSys := ∅, ApSys := R
2: flag:=**true**
3: **while** flag = **true do**
4: [ExSys, ApSys, flag] := SplitExactApprox(ExSys, ApSys, flag)
5: **end do**
6: Compute the ID and Differential Hilbert Function for ExSys determining its involutivity order.
7: IDExSys := initialdata(ExSys)
8: HFExSys := DifferentialHilbertFunction(IDExSys,s)
9: **for** k **from** 0 **do**
Compute and simplify prolongations
10: DApSys[k] := dsubs(ExSys, D^kApSys)
11: **until** ExSys ∪ DApSys[k] tests projectively involutive
12: **return** [ExSys, ApSys, DApSys[k], HFExSys, IDExSys]

Application of algorithms 1 on Poisson equation

Applying rifsimp to ExSys yields

$$\begin{aligned} \text{rExSys} := [&\eta_{z,z,z} = 0, \xi_{z,z,z} = 0, \zeta_{z,z,z} = 0, \\ &\xi_{y,y} = \xi_{z,z}, \xi_{y,z} = 0, \eta_x = -\xi_y, \xi_x = \zeta_z, \\ &\zeta_x = -\xi_z, \eta_y = \zeta_z, \zeta_y = -\eta_z, \eta_u = 0, \\ &\phi_u = \frac{\phi}{u}, \xi_u = 0, \zeta_u = 0] \end{aligned}$$

Now we simplify ApSys with respect to rExSys using dsubs(rExSys, ApSys) and obtain:

$$\begin{aligned} &\text{SimpApSys} := \\ &[-\frac{\xi_{z,z}u - 2\phi_x}{u} = 0, \frac{-\eta_{z,z}u + 2\phi_y}{u} = 0, \frac{\zeta_{z,z}u + 2\phi_z}{u} = 0, \\ &\frac{G\phi}{u} + \phi_{x,x} + \phi_{y,y} + \phi_{z,z} - 2G\zeta_z - \eta G_y - \xi G_x - \zeta G_z = 0] \end{aligned}$$

Notice that the first 3 equations of SimpApSys are now exact and they can be removed to yield an updated

$$\text{ApSys} := [\frac{G\phi}{u} + \phi_{x,x} + \phi_{y,y} + \phi_{z,z} - 2G\zeta_z - \eta G_y - \xi G_x - \zeta G_z = 0]$$

The 3 exact equations can be appended to rExSys to give an updated ExSys:

$$\begin{aligned} &\text{ExSys} := \text{rExSys} \quad \cup \\ &[-\frac{\xi_{z,z}u - 2\phi_x}{u} = 0, \frac{-\eta_{z,z}u + 2\phi_y}{u} = 0, \frac{\zeta_{z,z}u + 2\phi_z}{u} = 0] \end{aligned}$$

Application of algorithm 2 on Poisson equation

Applying rifsimp to the new ExSys, yields:

$$\begin{aligned} \text{rExSys} := [&\xi_u = 0, \eta_u = 0, \zeta_u = 0, \xi_{y,z} = 0, \phi_{x,x} = 0, \\ &\phi_{x,y} = 0, \phi_{x,z} = 0, \phi_{y,y} = 0, \phi_{y,z} = 0, \phi_{z,z} = 0, \xi_x = \zeta_z, \\ &\eta_y = \zeta_z, \eta_x = -\xi_y, \zeta_x = -\xi_z, \zeta_y = -\eta_z, \phi_u = \frac{\phi}{u}, \\ &\xi_{y,y} = \frac{2\phi_x}{u}, \xi_{z,z} = \frac{2\phi_x}{u}, \eta_{z,z} = \frac{2\phi_y}{u}, \zeta_{z,z} = -\frac{2\phi_z}{u}] \end{aligned}$$

The initial data about a point $w^0 = (x^0, y^0, z^0, u^0)$ for this system is

$$\begin{aligned} [&\eta(w^0) = c_1, \eta_z(w^0) = c_2, \phi(w^0) = c_3, \phi_x(w^0) = c_4, \\ &\phi_y(w^0) = c_5, \phi_z(w^0) = c_6, \xi(w^0) = c_7, \xi_y(w^0) = c_8, \\ &\xi_z(w^0) = c_9, \zeta(w^0) = c_{10}, \zeta_z(w^0) = c_{11}] \end{aligned}$$

and the Differential Hilbert Function is

$$H(s) = 4 + 7s \tag{3}$$

Following our Intersection algorithm we simplify ApSys with respect to the new rExSys and obtain:

$$\text{SimpApSys} := [-\eta G_y - \zeta G_z - \xi G_x - 2\zeta_z G + \frac{\phi G}{u} = 0] \tag{4}$$

We note that both the order 2 prolongation of rExSys and indeed SimpApSys is also involutive. What remains is to prolong SimpApSys:

$$\text{SimpApSys}[k] := \text{dsubs}(\text{rExSys}, D^k\text{SimpApSys}) \tag{5}$$

until the joint system rExSys ∪ SimpApSys[k] tests projectively involutive. The dimension tests for involutivity are executed using the dimension information from the DifferentiaHilbertFunction for rExSys combined with the dimensions of the kernel and row space (co-kernel) of the projections of the prolonged approximate system. Since rExSys has 0 dimensional symbol all the calculations are efficiently carried out in J^2, actually J^1 after elimination from rExSys.

References

[1] S. Anco, G. Bluman, and T. Wolf.
Invertible mappings of nonlinear PDEs to linear PDEs through admitted conservation laws.
Acta Applicandae Mathematicae, 101, 21–38, 2008.

[2] J. Carminati and K. Vu.
Symbolic computation and differential equations: Lie symmetries.
Journal of Symbolic Computation, 29, 95–116, 2000.

[3] A. F. Cheviakov.
GeM software package for computation of symmetries and conservation laws of differential equations.
Computer Physics Communications, 176(1), 48–61, 2007.

[4] P. Olver.
Application of Lie groups to differential equations.
Springer-Verlag, 2nd edition, 1993.

[5] T. Rocha Filho and A. Figueiredo.
SADE: A Maple package for the symmetry analysis of differential equations.
Computer Physics Communications, 182(2), 467–476, 2011.

[6] T. Wolf.
Investigating differential equations with crack, liepde, applsymm and conlaw.
Handbook of Computer Algebra, Foundations, Applications, Systems, 37, 465–468, 2002.