STeM: Symbolic Tetrad Manipulator. Part I

J.Carminati[1], K. T. Vu[2]
[1]School of Information Technology, Deakin University, Australia
[2]FIS, Australasia, Melbourne, Australia

July 14, 2025

Abstract

In this first part, of a two-part presentation, we present and describe, with illustrative examples, some of the basic functionality of the MAPLE computer algebra package *STeM*, which has a comprehensive collection of routines to allow the user to carry out most computations in the Newman-Penrose, Ellis-MacCallum, and Geroch-Held-Penrose formalisms, or any other tetrad formalism which is a modification of any one of these three systems.

1 Program summary

Program Title: STeMV1R1

CPC Library link to program files:

Developer's repository link: https://github.com/jcarm1930/STeMV1R1

Licensing provisions: None

Programming language: MAPLE internal language

Nature of Problem: The tetrad formalism is an approach to general relativity that generalizes the choice of basis for the tangent bundle from a coordinate basis to a more general choice of a local basis, i.e. a locally defined set of four linearly independent vector fields called a tetrad.[21] Usually the choice of the tetrad is guided by the geometric/physical properties of the problem at hand. The advantage of the tetrad formalism over the standard coordinate-based approach to general relativity lies in the ability to choose the tetrad basis to reflect important physical/geometric aspects of the spacetime. This can be quite useful when investigating solutions and their properties, of the Einstein field equations. In the tetrad formalism, all tensors are represented in terms of their components (scalars) in a chosen basis and computations are conveniently carried out in terms of these components. Since the tensorial equations are written in terms of their basis components, large expressions can arise. This can be an advantage as well as a drawback but in many cases, however, the properties that one seeks or assumes usually become more evident. Apart from the direct use of the Ricci identities, Einstein field equations and Bianchi identities, one usually extracts important additional information through the use of the commutator conditions present in all the three major formalisms, Newman-Penrose(NP),[17] Ellis-MacCallum(EM)[10 15] and Geroch-Held-Penrose(GHP),[12] mainly used in General Relativity.:Usually, the tetrad computations and resulting scalar equations, can be extensive and complicated, and well beyond hand computation ability.

Solution method: STeM, which runs under Maple, consists of a comprehensive collection of routines to allow the user to carry out most complex calculations and analyses in the three formalisms NP, EM, GHP or any other tetrad formalism which is a modification of any one of these three systems.

Keywords: Computer Algebra, General Relativity, Exact solutions, Tetrad formalisms.

2 Introduction: Tetrad Formalisms

In general relativity's original inception, the field equations (EFE) were written by Einstein in tensor form in a coordinate basis. In time and with the development of more sophisticated mathematics especially with the works of E. Cartan, it was recognised that the analyses, and subsequent clarification, of geometric aspects in the theory were more easily carried out with the introduction of a noncoordinated basis. Great impetus to this new approach was given by the seminal paper by Newman and Penrose (NP)[17] in 1962, where with the use of spinors and spin bases, the spin coefficient, null tetrad approach was basically born. In their approach,

at each point of the spacetime, a complex null tetrad $\{\mathbf{l},\mathbf{n},\mathbf{m},\bar{\mathbf{m}}\}$, with the full freedom of choice given by the 6-parameter Lorentz group, was introduced with the orthogonality properties

$$\begin{aligned} l_a l^a = n_a n^a = m_a m^a = l_a m^a = n_a m^a = 0, \\ l_a n^a = -m_a \bar{m}^a = 1, \end{aligned}$$

and tetrad metric

$$\eta_{ab} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

so that

$$g_{ab} = 2l_{(a}n_{b)} - 2m_{(a}\bar{m}_{b)}.$$

In applying the NP "method", one usually aligns the tetrad according to the symmetries of the spacetime or with the eigenvectors singled out by a particular tensor like the Weyl or Maxwell tensor with the consequence that substantial simplifications occur. The method has proven to be quite useful in a number of classification problems like those of the Weyl and Ricci tensors. We note that important solutions of the EFE were (and still are) quickly obtained, in particular the rotating Kerr solution, after the appearance of the NP formalism. This formalism is currently, more than ever, used in a wide range of applications with considerable success.
Moving on, sometime later, it was in particular noted by G. F. R. Ellis, M.A.H. MacCallum and others that the introduction of a null tetrad was not really suited to studying fluid sources and cosmological models of the EFE. Again, considerable impetus to this approach was given by the seminal papers of Ellis[10] and MacCallum.[15] In these spacetimes, there is a time-like unit vector $\mathbf{u}$ singled out by the fluid flow. It then seemed more appropriate to introduce an orthonormal tetrad $\{\mathbf{t},\mathbf{x},\mathbf{y},\mathbf{z}\}$ at each point of the spacetime, with the orthogonality properties

$$\begin{aligned} t_a x^a = t_a y^a = t_a z^a = x_a y^a = x_a z^a = y_a z^a = 0, \\ -t_a t^a = x_a x^a = y_a y^a = z_a z^a = 1, \end{aligned}$$

and tetrad metric

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

so that

$$g_{ab} = -t_a t_b + x_a x_b + y_a y_b + z_a z_b.$$

In applying this approach, one usually aligns the vector $\boldsymbol{t}$ with $\mathbf{u}$. One then projects all physical (geometric) quantities of interest, to spaces parallel and orthogonal to $\mathbf{u}$. In doing so, one obtains spacelike tensors/components like the vorticity, shear, expansion, of the fluid, and electric and magnetic parts of the Weyl tensor and so on. The tetrad is then further aligned (specialised), with the additional freedom of the SO(3) group, with these quantities, their eigenvectors, or with any symmetries present.
In 1973, R. Geroch, A. Held, and R. Penrose,[12] introduced an interesting variant of the Newman-Penrose formalism, where a pair of null directions was singled out. It is an intermediate formalism between a fully covariant one and the spin-coefficient approach. The key idea is to only retain quantities in the NP formalism which are properly weighted under the spin-boost 2-parameter subgroup of the Lorentz group. Of the original 12 complex spin coefficients only 8 remain with modifications of the derivative operators and commutator conditions. A remarkable, considerable simplification occurs. This formalism has been quite useful in the study/integration of algebraically special spacetimes or with the analysis of perturbations of them.
Regarding the *STeM* package, it is made up of three sub-systems corresponding to the three formalisms mentioned above. The orthonormal implementation follows closely the basic functionality and notation as that given by N. Van den Bergh in his program called Oframe.[24] The NP part follows the functionality and notation as that given by S Czapor and R.G. McLenghan in their NP package.[9] Finally, The GHP part which includes all of the routines in our previous package is an expansion of our previous GHP package.[29] For a very comprehensive list of computer algebra packages for general relativity, we refer the reader to the article *Computer algebra in gravity research* by M.A. H. MacCallum.[16]

3 The Maple STeM Package: Notation and Conventions

Throughout, *STeM* uses the (-,+,+,+) signature with Levi-Civita tensor defined by $\eta^{0123} = (-g)^{1/2}$. The Riemann, Ricci, and Weyl tensors are defined by

$$P^a{}_{;cd} - P^a{}_{;dc} = R^a{}_{bdc}P^b, R_{ab} = R^c{}_{acb},$$
$$C_{abcd} = R_{abcd} - \frac{1}{2}(g_{ac}R_{bd} + g_{bd}R_{ac} - g_{bc}R_{ad} - g_{ad}R_{bc}) + \frac{R}{6}(g_{ac}g_{bd} - g_{ad}g_{bc}).$$

where P is an arbitrary vector field.
The electric and magnetic parts of the Weyl tensor are defined by

$$E_{ab} = C_{agbh}u^g u^h, H_{ab} = \frac{1}{2}\eta_{ae}{}^{gh}C_{ghbd}u^e u^d.$$

Given an electromagnetic field represented by a skew tensor field F_{ab},the electric field E_a and magnetic field H_a,as measured by an observer with 4-velocity u^a are given by

$$E_a = F_{ab}u^b, H_a = \frac{1}{2}\eta_{ab}{}^{cd}u^b F_{cd}.$$

The NP form of Maxwell's equations are

$$\begin{aligned}
D\Phi_1 - \bar{\delta}\Phi_0 + \kappa\Phi_2 - 2\rho\Phi_1 + (2\alpha - \pi)\Phi_0 &= x_0 J_l,\\
\delta\Phi_1 - \Delta\Phi_0 + \sigma\Phi_2 - 2\tau\Phi_1 + (2\gamma - \mu)\Phi_0 &= x_0 J_m,\\
D\Phi_2 - \bar{\delta}\Phi_1 + (2\epsilon - \rho)\Phi_2 - 2\pi\Phi_1 + \lambda\Phi_0 &= x_0 J_{\overline{m}},\\
\delta\Phi_2 - \Delta\Phi_1 + (2\beta - \tau)\Phi_2 - 2\mu\Phi_1 + \nu\Phi_0 &= x_0 J_n.
\end{aligned}$$

where x_0 stands for the numerical constant 2π, and where

$$\begin{aligned}
J^a &= J_m\overline{m}^a + J_{\overline{m}}m^a - J_l n^a - J_n l^a,\\
E_a + iH_a &= \sqrt{2}\Phi_1(l_a - n_a) + \sqrt{2}(\Phi_2 m_a - \Phi_0\overline{m}_a),\\
\mathbf{E} + i\mathbf{H} &= 2\Phi_1\mathbf{e}_1 + (\Phi_2 - \Phi_0)\mathbf{e}_2 - i(\Phi_0 + \Phi_2)\mathbf{e}_3,\\
\mathbf{e}_0 &= \frac{1}{\sqrt{2}}(\mathbf{l} + \mathbf{n}), \mathbf{e}_1 = \frac{1}{\sqrt{2}}(\mathbf{l} - \mathbf{n}), \mathbf{e}_2 = \frac{1}{\sqrt{2}}(\mathbf{m} + \overline{\mathbf{m}}), \mathbf{e}_3 = \frac{-i}{\sqrt{2}}(\mathbf{m} - \overline{\mathbf{m}}),\\
E_1 &= (\Phi_1 + \overline{\Phi}_1), E_2 = \frac{1}{2}(\Phi_2 + \overline{\Phi}_2 - \Phi_0 - \overline{\Phi}_0), E_3 = -\frac{i}{2}(\Phi_2 - \overline{\Phi}_2 + \Phi_0 - \overline{\Phi}_0),\\
H_1 &= i(\overline{\Phi}_1 - \Phi_1), H_2 = \frac{i}{2}(\overline{\Phi}_2 - \Phi_2 + \Phi_0 - \overline{\Phi}_0), H_3 = -\frac{1}{2}(\Phi_0 + \overline{\Phi}_0 + \Phi_2 + \overline{\Phi}_2).
\end{aligned}$$

With geometrized units Einstein's equations read $G_{ab} = 4x_0 T_{ab}$. The electromagnetic contribution to the NP components of the Ricci tensor is $\Phi_{ab} = \Phi_a\bar{\Phi}_b$, with $\Lambda = 0$.
With our signature we have $l_a n^a = -1, m_a\overline{m}^a = 1$, and the Weyl and Ricci NP scalars need to change sign, to those given by NP, in order to maintain the same form for the Einstein- Maxwell equations. These quantities are explicitly given below:

$$\begin{aligned}
\Psi_0 &\equiv C_{abcd}l^a m^b l^c m^d, \Psi_1 \equiv C_{abcd}l^a n^b l^c m^d,\\
\Psi_2 &\equiv C_{abcd}l^a m^b \bar{m}^c n^d, \Psi_3 \equiv C_{abcd}l^a n^b \bar{m}^c n^d,\\
\Psi_4 &\equiv C_{abcd}n^a \bar{m}^b n^c \bar{m}^d,
\end{aligned}$$

and

$$\begin{aligned}
\Phi_{00} &\equiv \frac{1}{2}R_{ab}l^a l^b, \Phi_{11} \equiv \frac{1}{4}R_{ab}(l^a n^b + m^a \bar{m}^b), \Phi_{22} \equiv \frac{1}{2}R_{ab}n^a n^b, \Lambda \equiv \frac{R}{24},\\
\Phi_{01} &= \bar{\Phi}_{10} \equiv \frac{1}{2}R_{ab}l^a m^b, \Phi_{02} = \bar{\Phi}_{20} \equiv \frac{1}{2}R_{ab}m^a m^b,\\
\Phi_{12} &= \bar{\Phi}_{21} \equiv \frac{1}{2}R_{ab}m^a n^b.
\end{aligned}$$

The Maxwell and four-current NP scalars are

$$\Phi_0 \equiv F_{ab} l^a m^b = \phi_{11}, \Phi_1 \equiv \frac{1}{2} F_{ab}(l^a n^b + \overline{m}^a m^b) = \phi_{01}, \Phi_2 \equiv F_{ab}\overline{m}^a n^b = \phi_{00},$$
$$J_l \equiv J_a l^a, J_n \equiv J_a n^a, J_m \equiv J_a m^a, J_{\overline{m}} \equiv J_a \overline{m}^a = \overline{J}_m.$$

The three, 2-parameter null subgroups of the Lorentz group acting on the null basis vectors $\{l, n, m, \bar{m}\}$ are given below.

Null rotation leaving l^a fixed:

$$\begin{aligned}\tilde{l}^a &= l^a,\\ \tilde{n}^a &= n^a + d\overline{m}^a + \overline{d} m^a + d\,\overline{d}\, l^a,\\ \tilde{m}^a &= m^a + d l^a.\end{aligned}$$

Null rotation leaving n^afixed:

$$\begin{aligned}\tilde{n}^a &= n^a,\\ \tilde{l}^a &= l^a + f\bar{m}^a + \bar{f} m^a + f\overline{f}\, n^a,\\ \tilde{m}^a &= m^a + f n^a.\end{aligned}$$

Null rotation leaving directions of l^a and n^a fixed

$$\begin{aligned}\tilde{l}^a &= F^{-1} l^a,\\ \tilde{n}^a &= F n^a,\\ \tilde{m}^a &= e^{i\theta} m^a.\end{aligned}$$

The notation used for each formalism is given in the tables below.

Newman-Penrose Formalism: Notation

Original	STeM
D	D
Δ	V
δ	X
δ	Y
Ψ_0	$W0$
Ψ_1	$W1$
Ψ_2	$W2$
Ψ_3	$W3$
Ψ_4	$W4$
Λ	L
Φ_{00}	$R00$
Φ_{01}	$R01$
Φ_{02}	$R02$
Φ_{12}	$R12$

Original	STeM
Φ_{11}	$R11$
Φ_{22}	$R22$
κ	k
σ	s
ρ	r
τ	t
ν	n
λ	l
μ	m
π	p
ϵ	e
γ	g
α	a
β	b

Original	STeM
Φ_0	$F0$
Φ_1	$F1$
Φ_2	$F2$
V^1	Kl
V^2	Kn
V^3	Km
ψ	psi
ϕ_{00}	$B00$
ϕ_{10}	$B10$
ϕ_{11}	$B11$
J_l	$J0$
J_n	$J1$
J_m	$J2$

Ellis-MacCallum Formalism: Notation

Original	STeM
e_0	$e0$
e_1	$e1$
e_2	$e2$
e_3	$e3$
a_1	$a1$
a_2	$a2$
a_3	$a3$
$\dot{u}_1$	$u1$
$\dot{u}_2$	$u2$
$\dot{u}_3$	$u3$
ω_1	$w1$
ω_2	$w2$
ω_3	$w3$
n_{11}	$n11$
n_{12}	$n12$

Original	STeM
n_{12}	$n12$
n_{13}	$n13$
n_{22}	$n22$
n_{23}	$n23$
n_{33}	$n33$
σ_{11}	$s11$
σ_{12}	$s12$
σ_{13}	$s13$
σ_{22}	$s22$
σ_{23}	$s23$
σ_{33}	$s33$
V^1	$K0$
V^2	$K1$
V^3	$K2$
V^4	$K3$

Original	STeM
Ω_1	$O1$
Ω_2	$O2$
Ω_3	$O3$
$\theta_1 = \theta_{11}$	$H1$
$\theta_2 = \theta_{22}$	$H2$
$\theta_3 = \theta_{33}$	$H3$
θ	$H = H1 + H2 + H3$
μ	M
p	P
$E_{\alpha\beta}$	$E11..E33$
$H_{\alpha\beta}$	$M11..M33$
$E_{ab;c}u^c$	$dE11..$
$H_{ab;c}u^c$	$dM11..$
E_α	$E1..E3$
H_α	$B1..B3$

Geroch-Held-Penrose: Notation

Original	STeM
þ	T
þ′	TP
ð	E
ð′	EP

In our implementation, the NP components of the Killing (Homothetic) vector are given by[20]

$$K^a = Km\,\overline{m}^a + K\overline{m}\,m^a - Kl\,n^a - Kn\,l^a \ , \ K\overline{m} \ \equiv \overline{Km}$$

whereas in the orthonormal frame, the Killing vector $\mathbf{K}$ is expressed as

$$\mathbf{K} = -K0\,\mathbf{e}_0 + K1\,\mathbf{e}_1 + K2\,\mathbf{e}_2 + K3\,\mathbf{e}_3,$$
$$Kl = \frac{1}{\sqrt{2}}(K0 + K1), Kn = \frac{1}{\sqrt{2}}(K0 - K1), K\overline{m} = \frac{1}{\sqrt{2}}(K2 + iK3)$$

and the Killing equations are written as

$$K_{a;b} = Z_{ab} + \psi g_{ab},$$

where $\psi = 0$ for a Killing vector or ψ =constant $\neq 0$ for a homothetic vector.
The associated complex self-dual bivector is

$$\begin{aligned} Z^+_{ab} &\equiv \frac{1}{2}(Z_{ab} + iZ^*_{ab}) \\ &= 2\phi_{00} l_{[a} m_{b]} + 2\phi_{01}(l_{[a} n_{b]} - m_{[a} \bar{m}_{b]}) - 2\phi_{11} n_{[a} \bar{m}_{b]} \end{aligned}$$

A list of current *STeM* package routines, for each formalism with short descriptions is given in the tables below.

Available Commands, with Syntax, for all Formalisms[1]

Command/Syntax	Description
addpair(sys)	adds and factors all distinct pair additions of polys in list sys
alldeg(sys)	displays the degree of each variable in sys
autocomm(vars)	computes all 6 basic commutators of vars
buildchain(sys,chn,R)	constructs a new chain from inclusion of sys
chaincombo(PB,sys,N,T)	generates chains from all N subsets of sys, T=maximum time allowed
chainnormal(sys,chn,vars)	reduces each poly in sys to normal forms using a chain chn
clean(list)	removes a=a or 0 from a list
clearall()	clears all remember tables
clearAssumption()	clears all assumptions made by automated processes in STeM
forgetUnusedEqn()	clears all equations not used by stemnormal during process generation
forgetUsedEqn()	clears all equations used by stemnormal during process generation
comm(oper1,oper2,expr)	computes commutator of oper1,oper2, applied to expression
conj(expr)	computes complex conjugate of expr
convertGHP2NP(expr)	converts GHP expression/equation to NP
convertNP2GHP(expr)	converts NP expression/equation to GHP
convertNP2Orth(expr)	converts NP expression/equation to orthonormal
convertOrth2NP(expr)	converts orthonormal expression/equation to NP
defineOperator(Z$_{new}$, Z_{new}= $f(Z_{old})$)	defines new operator in terms of old ones
deglist(list)	applies alldeg to a list
eqns(m,n)	downloads m to n equations of designated formalism
findCombo(sys)	attempts to find a single "suitable" variable combination in sys
findComm(sysde)	finds and computes all possible commutator pairs in sysde
findHalfComm(sysde)	finds and computes all possible higher order commutators in sysde
findVariable(struc,sys,red)	determines possible variable changes, based on user given structure, "struc", which minimises sys according to Maple nops
findops(list)	determines elements of a list of polys which have multiple ops
forgetAssumption()	forces STeM to forget all assumptions
forgetCommutator()	forces STeM to forget computed commutators
formalism(form)	sets the formalism to ghp, np or orth environment
forgetUnusedEqn()	forces STeM to clear all unused equations generated by stemnormal
forgetUsedEqn()	forces STeM to clear all used equations generated by stemnormal
formalism(form)	sets the formalism to ghp, np or orth environment
genreslist(vars,nstrt,eqax,resmax,sys)[6]	computes all resultants, in pairs, w.r.t. vars of sys

getAssumption()	displays all assumptions made by certain routines used
getUnusedEqn()	retrieves all unused equations generated by automated processes
getUnusedFactor()	retrieves all unused factors generated by automated processes
getUsedEqn()	retrieves all used equations generated by automated processes
getUsedFactor()	retrieves all used factors generated by automated processes
groebcombo(PB,sys,N,T)	generates GB's from all N subsets of sys, T=maximum time allowed
groebreduce(expr,PB, GBL)	reduces expr using each element of GBL
indeppol(sys, nfacs, zerofacs)	routine removes nonzero factors (nfacs: list), and dependant polys which have factors 'zerofacs' from sys
init()	clears all computation results and definitions in session
isCommutable(expr)	checks if a particular expression is commutable[5]
linreslist(vars,num1,maxl,maxres,list)	computes resultants (length< maxres) wrt linear vars[i],starting from num1 to max.length maxl of polys in list
minimisePoly(sys)	minimises, in pairs of expr/eqns, Maple length of sys using GB
nullRotation(expr/eqn,preserve=l,n,or ln)[3]	performs null rotation of type "class" on expr/eqn
orthRotation(expr/eqn, preserve=e_i)[3]	performs 2-d rotation of type preserving e_i, on expr/eqn
picfac(expr)	selects a factor choice from ops of expr
polydecomp(poly,facexp)	decompose poly using factor facexp
polyiso(expr,combo)	isolates a combo from expr
powsubs2	allows multiple powsubs substitutions
prime(expr/eqn)	computes the prime of an expr/equation
properties(var)	displays zero/nonzero/complex/real/weight/prime/star of var
reduceMin(eqn,sys,keep=select)[9]	reduces the Maple length of eqn w.r.t. sys using LCM
reduceMinSys(sys,keep=select)[9]	reduces the Maple length of sys w.r.t. itself using term-by-term LCM
remdeg(sys,var,N)	removes all polys from sys with degree > N in var
remfacs(sys,[fac1,fac2,..]	removes factors fac1,fac2,.. from each poly of sys
removeNZfactor(expr)	removes known nonzero factors as given in showAssumption, from expr
remredun(sys)	removes redundant (by numerical factors) polys from sys(polys)
replaceOperator(Z_{new}, Z_{new} = f(Z_{old})	replaces old operator with defined new one
separate(list)	separates polys from des from a list
sepdeg(var,N,L)	selects polys of L with 0< deg(var) $\leq N$
sepeqns(L,varin,varout)	selects polys of L with variable varin but not varout
showAssumption()	displays cumulatively all nonzero assumptions made by automated processes
showCommutator()	shows all commutators calculated by findComm

showEInfo(poly)	displays information of poly
showStatistic(sys)	displays the statistics of sys
showUnusedEqnStat()	displays statistics of unused eqns by stemnormal
showUnusedFactorStat()	displays statistics of unused factors by stemnormal
simplifySys(lista,listb)	minimises pairwise, length of each element of lista against, listb using GB[4]
specs(sys)	displays indets, length, nops and type of sys
specsop(sys)	displays indets, length, nops and type of each op of sys
star(expr/eqn)	applies the star operation
stemCollect(expr,vars)	collects an expr w.r.t.vars list
stemHTerm(poly)	displays highest ordered term in poly
stemassume([expr1, expr2, ...])[5]	declares expr1, expr2,... as nonzero for automated processes
stemcomplete(sys,switch=[op1,op2,op3])[8]	proceeds in automated fashion, continuous computation of commutators and subsequently adjoins results to sys then normalises Stops when no new commutators can be found and system is fully normalised/reduced (see stemnormal)
stemdefine(var,weight=[p,q],prime=var1,star=var2)	defines the weight, prime and star of var in a session
stemdefoper([Z1,Z2,Z3,Z4]):	defines an ordering on the operators, from lowest to highest
stemdeforder([var1, var2,...])	defines an ordering on the variables, from lowest to highest
stemeval(expr)	same as maple eval plus complex quantity evaulation in np/ghp
steminvariant([il]) invsys=prime, star or both	establishes an environment automatically invariant under invsys imposing conditions that are invariant as indicated by type,invariant
stemiso(eqn)	solves for derivatives in an equation
stemisoh(eqn/expr)	solves eqn/expr for the highest ordered term,
stemnormal(sys,switch=[op1,op2,op3])	transforms a given system of equations into a normal form based on an ordering defined by stemdeforder and stemdefoper
stemreduce(eqn/expr,sysn)	reduces, a given eqn/expr w.r.t. sysn[1]
stemsorte(sys)	sorts a system, in a solved form with highest term on LHS, in ascending order with all polynomials listed first
stemundefine(var)	clears all defined properties of var.
subtractpair (sys)	subtracts and reduces in pairs of sys.
symtran(sys,nvars,symnvars)	performs discrete symmetry transformation on sys[7]
weight(expr/eqn/var)	displays the weight of a given expr/eqn/var

In the tables below, we indicate where the various equation groups of the basic formalisms, are located.

Newman-Penrose Equation System

Position	Description
1..18	Ricci
19..29	Bianchi
30..33	Maxwell
34..45	Killing
46..57	First IC Killing[2]
58..72	Second IC Killing[2]

Ellis-MacCallum Equation System

Position	Description (Eqns)
1..16	Jacobi
17..26	Field
27..42	Bianchi
43..48	E_{ab}
49..54	H_{ab}
55..58	Contracted Bianchi
59..66	Maxwell
67..76	Killing
77..81	$E_{ab;c}u^c$
82..86	$H_{ab;c}u^c$
87..104	Basic Variables
105..135	Basic EFE/I.C.
136..148	Ext./New Variables
149..153	λ and $e_i(\lambda)$

Geroch-Held-Penrose Equation System

Position	Description (Eqns)
1..12	Ricci
13..24	Bianchi
25..28	Maxwell
29..40	Killing
41..52	First IC Killing[2]
53..67	Second IC Killing[2]

In the above tables we have used the following abbreviations: (i). sys=list of expressions/equations of polynomials; (ii). expr=expression; (iii). Z1,Z2,..denote operators; (iv). x1,x2,..denote variables; (v). vars is a variables list, var is one variable (vi) expr is an expression with variables and/or operators; (vii). form=np,orth, or ghp; (viii) R= Polynomial Ring (ix). N=integer. (x) PB list of polynomials for ideal (xi) GBL: Groebner basis list.(xii): zerofacs: factors that are zero.[1]The given system of differential equations, *sysdeqn*, is required to be in a normal form, as given by *stemnormal*. Warning: if the given system is not in a normal form, this function might get into an infinite loop;[2] The integrability conditions are only for a Homothetic vector;[3] The *STeM* environment for the rotations must be free of global assignments, i.e. in its generic state with original variables;[4]Groebner Bases; [5]An expression is potentially commutable if when it is solved for the highest ordered derivative all other derivatives on r.h.s. are of lower order;[6]nstrt: poly number in sys to begin resultants; eqmax: maximum length of polys used; resmax: maximum size of resultant to retain;[7]nvars: new extra variables introduced and symnvars the associated symmetry mapping of nvars.[8]Optional entries are: "algebraic", "factor" and "immstop" for control of substitution of algebraic conditions, factors, and immediately stop option after each decision step. The user should exercising care when "cutting" output from *STeM* and then pasting it in as input, in the orthonormal formalism. There is no issue with the variables but the operators paste as "e[n]", n=0..3, which is not acceptable. These may be manually changed to "en" or done so through the use of the routine *convder*. [9]The switch "select=keep" gives the user the option of which factors to keep in the reduction process otherwise automatically the largest ordered/size factor is retained

---

STeM running under Maple, contains all of the standard equations of the Orthonormal, NP and GHP formalisms It automates virtually all of the regular computational tasks present in the tetrad frames, like propagation, commutation and normalisation of differential/polynomial conditions. The main key routine in *STeM* is *stemnormal* which reduces a combined list/set of polynomial and differential conditions to a "standard" form with the order of the variables and operators as specified by the user. In addition, there are five major simplifying routines in *STeM* called *stemreduce*, *minimisePoly*, *simplifySys*, *reduceMinSys* and *stemHTerm* to minimize/simplify resulting systems of equations. Briefly, *stemreduce* reduces a list of polynomial/differential conditions to a "minimial form" with respect to another list of conditions. *minimisePoly* is a routine which attempts to simplify, in pairs, elements of a set of polynomial conditions using Groebner basis methods. It can readily handle systems of large polynomial equations. *simplifySys* applies *minimisePoly* to two separate lists

minimizing/simplifying one against the other. *reduceMinSys* (*reduceMin* is for the reduction of a single condition with respect to a list of conditions) is a routine which performs a sort of exhaustive pair-term elimination using LCM's, to minimise the Maple length (or number of terms). It is computationally very intense and is more suited to small or medium sized systems. Large systems can also be addressed but would require considerably more time to complete the normalization/reduction process. *stemHTerm* presents the leading terms in a set of polynomial, based on the user's defined variable order, and then other routines in *STeM* decide, based on these leading terms, what equations are pairwise "suitable" for elimination of leading terms and hence is a sort of "restricted" Groebner method. Again, the governing principle is based on minimising Maple length, or variable elimination or term-number reduction in polynomials. Combined with the use of standard resultants, these tools can be quite effective in reducing/simplifying polynomial systems.
In the next section, we will present three illustrative applications showing how *STeM* facilitates the computations typically encountered in the three main tetrad formalisms

4 STeM Applications

In the applications that follow, we shall present the exact Maple input and some of the output that is required to achieve a particular result. For cases where intermediate steps may be large, we shall suppress the output print. The key feature that we want to emphasize is that in many problems intermediate expressions may be very large and it would be of little practical value to explicitly see them. Instead, the processes and advanced tools available in *STeM* allow the user to properly and intelligently navigate through a particular sequence of very large intermediate steps using only *STeM* auxiliary information, provided by *showStatistic*, *specs, maplen, sortl,* and other routines. Of course there are also generic Maple internal commands that can be used as well

4.1 Example 1: Type D vacuum spacetimes

In this example, we shall examine vacuum Petrov type D metrics, which are solutions of Einstein's field equations, all of which have been previously given by Kinnersley. However, it was some time later that it was shown by Czapor and McLenaghan[9] that all of these spacetimes possess the intriguing spin coefficient conditions $\tau\bar{\tau} = \pi\bar{\pi}$ and $\rho\bar{\mu} = \bar{\rho}\mu$, in the null frame that is aligned with the principal null directions of the Weyl tensor. In their computation using the Maple program NP,[9] these identities were established, though with considerable difficulty. They used the NP formalism with a special gauge choice of $\rho = \pi$ and then performed lengthy computations with trial-and-error attempts with various substitution/reduction steps. Clearly, this can be a difficult computational problem. We shall show, using *STeM,* how these identities can be established in a simple, virtually automated fashion and in just a few minutes. Almost all of the input and output are displayed below.
We begin with the *STeM* input to establish the appropriate GHP environment with conditions and equations

```
>read 'c:/STeMV1R1.mpl':with(STeM):formalism(ghp):eqs:=eqns(1,24):
>stemdeforder([m,p,r,t,W2]):stemdefoper([E,EP,T,TP]):
>R00:=0:R01:=0:R02:=0:R12:=0;R10:=0:R20:=0:R21:=0:R22:=0:R11:=0:W0:=0:W0_c:=0:W1:=0:
W1_c:=0:W3:=0:W3_c:=0:W4:=0:W4_c:=0:k:=0:k_c:=0:s:=0:s_c:=0:n:=0:n_c:=0:l:=0:l_c:=0:L:=0:
```

A few clarify points need to made at this stage. Firstly, we need to bear in mind that the conjugate variables are treated as independent from the variables themselves. Secondly, it is generally best to position the curvature at a higher order than the spin coefficients. However, having said that, we have noted that there have been some (very rare) exceptional cases where the best ordering is reversed. The ordering dictates to *stemnormal* (and also other routines) a key routine in *STeM*, which variables have a higher priority for the solving/reduction processes. There is no need to include the conjugates in the ordering as they are automatically positioned adjacent to their corresponding variables. If the user wishes, the conjugates can be repositioned as desired. Finally we have taken advantage of the Goldberg-Sachs theorem which in this cases yields $\kappa = \sigma = \nu = \lambda = 0$. In any case, one need not add these conditions in as they are otherwise quickly obtained by *STeM* in the reduction/normalization process.
Next, we generate the conjugates of all the equations *eqs* adjoin them to *eqs,* and apply *stemnormal* to the combined set.

```
>ceqs:=map(conj,eqs):
>toteqs:=stemnormal([op(eqs),op(ceqs)],switch=[algebraic,factor]);
```

toteqs := [EP(m) = -(m-m_c)p, TP(m) = -m^2, E(m_c) = p_c(m-m_c), TP(m_c) = -m_c^2,
E(p) = T(m)-mr_c-pp_c-W2, EP(p) = -p^2,TP(p) = -(t_c+p)m, E(p_c) = -p_c^2,
EP(p_c) = T(m_c)-m_cr-pp_c-W2_c, TP(p_c) = -m_c(t+p_c), E(r) = (r-r_c)t,
T(r) = r^2, EP(r_c) = -t_c(r-r_c), T(r_c) = r_c^2, E(t) = t^2,
EP(t) = TP(r)+m_cr+tt_c+W2,T(t) = (t+p_c)r, E(t_c) = TP(r_c)+mr_c+tt_c+W2_c,

EP(t_c) = t_c^2, T(t_c) = r_c(t_c+p),E(W2) = 3W2t, EP(W2) = -3W2p,
T(W2) = 3W2r, TP(W2) = -3W2m, E(W2_c) = -3W2_cp_c, EP(W2_c) = 3W2_ct_c,
T(W2_c) = 3W2_cr_c,TP(W2_c) = -3W2_cm_c]

We proceed, by computing all commutators initially possible with *toteqs.*

>allcom:=findComm(toteqs);

allcom := [E(r_c) = -T(p_c), EP(r) = -T(p), TP(r) = -tt_c-T(m)+pp_c, TP(r_c) = -tt_c-T(m_c)+pp_c,
TP(t) = -E(m), TP(t_c) = -EP(m_c)]

We can query *STeM* to see what commutators have been done.

>showCommutator();

Number of full commutators calculated is , 20
They are
[E, EP, W2], [E, T, W2], [E, TP, W2], [EP, T, W2], [EP, TP, W2], [T, TP, W2]
[E, EP, W2_c], [E, T, W2_c], [E, TP, W2_c], [EP, T, W2_c], [EP, TP, W2_c], [T, TP, W2_c]
[E, T, r]
[E, T, t]
[E, TP, m_c]
[E, TP, p_c]
[EP, T, r_c]
[EP, T, t_c]
[EP, TP, m]
[EP, TP, p]
Number of half commutators calculated is , 0

We note that the conjugate variable commutators are also done so that the user need not manually add them. There is also the added benefit that it checks that all results are self consistent. This would assist in catching input typing errors, if they occur. Also we have the added information that no higher order commutators, indicated by "half commutators", have been computed It is important to always check what assumptions *STeM* has made in given computations. This information can be retrieved as follows, and is always cumulative unless cleared by the user using the *clearAssumption* command at any stage.

>showAssumption();

Non-zero factors
W2 $\neq$ 0
W2_c $\neq$ 0

Now it is evident that the results we are seeking do not lie at the first-order commutator level. We need to address the issue of higher order commutators. We can complete this exercise by first applying *stemnormal* to the combined system consisting of *toteqs* and *allcom.*

>toteqsa:=stemnormal([op(toteqs),op(allcom)],switch=[algebraic,factor]);

toteqsa:=[EP(m) = -(m-m_c)p, TP(m) = -m^2, E(m_c) = p_c(m-m_c), TP(m_c) = -m_c^2,
E(p) = T(m)-mr_c-pp_c-W2, EP(p) = -p^2, TP(p) = -(t_c+p)m, E(p_c) = -p_c^2,
EP(p_c) = T(m_c)-m_cr-pp_c-W2_c,TP(p_c) = -m_c(t+p_c), E(r) = (r-r_c)t,EP(r) = -T(p),
T(r) = r^2, TP(r) = -tt_c-T(m)+pp_c, E(r_c) = -T(p_c), EP(r_c) = -t_c(r-r_c),T(r_c) = r_c^2,
TP(r_c) = -tt_c-T(m_c)+pp_c, E(t) = t^2, EP(t) = -T(m)+pp_c+m_cr+W2,T(t) = (t+p_c)r,
TP(t) = -E(m), E(t_c) = -T(m_c)+pp_c+mr_c+W2_c, EP(t_c) = t_c^2,T(t_c) = r_c(t_c+p),
TP(t_c) = -EP(m_c), E(W2) = 3W2t, EP(W2) = -3W2p,T(W2) = 3W2r, TP(W2) = -3W2m,
E(W2_c) = -3W2_cp_c, EP(W2_c) = 3W2_ct_c,T(W2_c) = 3W2_cr_c, TP(W2_c) = -3W2_cm_c]

We are now in a position to compute the half-commutators.

>halfcomm:=findHalfComm(toteqsa):

Calculate a commutator [E,EP](m)
Commutator result : -2T(m)m+2T(m)m_c+2W2m-W2_cm-pE(m)+2 mpp_c-2m_cpp_c-m_cW2-EP(E(m))
Calculate a commutator [T,EP](m)
Commutator result : -2pT(m)+pT(m_c)+T(p)m_c-T(p)m-EP(T(m))- rpm_c
Calculate a commutator [E,TP](m)
Commutator result : -TP(E(m))-3mE(m)
Calculate a commutator [T,TP](m)
Commutator result : -mp_ct_c-pE(m)-E(m)t_c+mpp_c-m_cpp_c-tpm_c-W2m-W2_cm-2T(m)m-TP(T(m))
Calculate a commutator [EP,E](m_c)
Commutator result : 2T(m_c)m-2T(m_c)m_c-m_cW2+2W2_cm_c-E(EP(m_c))-2mpp_c-W2_cm+2m_cpp_c-p_cEP(m_c)EP(m_c))-2mpp_c-W2_cm+2m_cpp_c-p_cEP(m_c)
Calculate a commutator [T,E](m_c)
Commutator result : T(p_c)m-T(p_c)m_c+p_cT(m)-2p_cT(m_c)-E(T(m_c))-r_cp_cm
Calculate a commutator [EP,TP](m_c)
Commutator result : -TP(EP(m_c))-3m_cEP(m_c)
Calculate a commutator [T,TP](m_c)
Commutator result : -tpm_c-p_cEP(m_c)-EP(m_c)t-mpp_c+m_cpp_c
-mp_ct_c-m_cW2-W2_cm_c-2T(m_c)m_c-TP(T(m_c))
Calculate a commutator [E,EP](p)
Commutator result : -2pT(m)+pT(m_c)+T(p)m_c-T(p)m-EP(T(m))-rpm_c
Calculate a commutator [T,EP](p)
Commutator result : -3pT(p)-EP(T(p))

Because of space limitations, here we have only presented about one third of the automated generated output. As expected, this routine has now generated expressions for second order derivatives. The final two steps consist of normalizing *halfcomm* with *toteqsa* and then computing the final set of commutators.

```
>toteqsb:=stemnormal([op(halfacomm),op(toteqsa)],switch=[algebraic,factor]):
>allcoma:=findComm(toteqsb);
```

allcoma := [t_c = (-mr_c+m_cr+pp_c)/t, t_c = -(-mr_c+m_cr-pp_c)/t, E(m) = mp_c+2mt-m_ct
, EP(m_c) = -mt_c+m_cp+2m_ct_c,T(p) = 2pr-pr_c+rt_c, T(p_c) = -p_cr+2p_cr_c+r_ct]

We have established the identities that we were looking for plus some additional information on the derivatives. Again, we should check for any additional assumptions made by *STeM*.

```
>showAssumption();
```

Non-zero factors
$W2 \neq 0$
$W2\_c \neq 0$
Non-zero denominators
$t \neq 0$
$t\_c \neq 0$

Note that we have picked up the extra assumption that $\tau \neq 0$. Presumably, to complete the investigation, one should check the $\tau = 0$ subcase. We wish to emphasize that at no stage was there any trial-and-error analysis and that all computations were carried out in an automated fashion. Ultimately, this whole computation can be carried out more efficiently and more rapidly if one uses the *stemcomplete* command as follows; all output suppressed except for the final result

```
>alleqs:=stemnormal([op(eqs),op(ceqs)],switch=[algebraic,factor]):
>alleqsa:=stemcomplete(alleqs):
>halfcomm:=findHalfComm(alleqsa):
>alleqsb:=stemnormal([op(alleqsa),op(halfcomm)],switch=[algebraic,factor]):
>fineq:=stemsorte(stemcomplete(alleqsb)):
```

We can actually know a good deal about the system *fineq,* without explicitly displaying it, by using *showStatistic*. This can be useful for when one is dealing with very large systems of large equations.

```
>showStatistic(fineq);
```

Number of equations is , 34
Number of polynomial equations is , 2
Number of differential equations is , 32
Equations' lengths are
[21, 21, 51, 34, 189, 20, 33, 74, 173, 24, 189, 20, 71, 42, 23, 174, 73, 38, 37, 76, 15, 190, 15, 190, 29, 52, 25, 26, 25, 26, 31, 40, 39, 32]
Equations' LHS are
[r_c, t_c, E(m), EP(m), T(m), TP(m), E(m_c), EP(m_c), T(m_c), TP(m_c), E(p), EP(p), T(p), TP(p), E(p_c), EP(p_c), T(p_c),TP(p_c), E(r), EP(r), T(r), TP(r), E(t), EP(t), T(t), TP(t), E(W2), EP(W2), T(W2), TP(W2), E(W2_c), EP(W2_c), T(W2_c), TP(W2_c),TP(W2_c)]

The total process has determined the desired two polynomial conditions, together with 32 differential conditions, which because of the *stemsorte* command will be located at the beginning of the list of equations *fineq*.

>fineq[1];fineq[2];

$$r\_c = m\_cr/m$$
$$t\_c = pp\_c/t$$

However, in this case, the system, through the use of *stemcomplete,* does make more nonzero assumptions. For example *STeM* assumes the following nonzero factors

Non-zero factors
$$p \neq 0$$
$$r \neq 0$$
$$W2 \neq 0$$
$$p\_c \neq 0$$
$$r\_c \neq 0$$
$$W2\_c \neq 0$$
$$p + 2\ t\_c \neq 0$$
$$p\_c + 2\ t \neq 0$$
$$2\ m\_c\ p\_c\ r + 2\ m\_c\ r\ t + 2\ p\ p\_c + 2\ p\ p\_c\ t + W2\ p\_c + W2\_c\ t \neq 0$$
$$2\ m\ p\ r\_c + 2\ m\ r\_c\ t\_c + 2\ p\ p\_c + 2\ p\ p\_c\ t\_c + W2\ t\_c + W2\_c\ p \neq 0$$

Similarly for nonzero assumptions made on denominators. In general the price one pays for more automation is the possible introduction of additional assumptions. These special cases need further analysis. As a final point, one can, using *stemassume,* declare that certain quantities are nonzero so that *STeM* carries out the computations assuming they are not branching decisions.

4.2 Example 2: Purely magnetic spacetimes

There are no known vacuum spacetimes with a purely magnetic Weyl tensor and so it has been conjectured that no such spacetimes exist. Necessarily, the Weyl tensor must be of Petrov I, D or O. The conjecture has been established for Type D and O but despite extensive attempts by many researchers, the proof of the conjecture or the presentation of a counter example is yet to be achieved for Petrov I. The problem is deceptively difficult. In this section, we will provide part of our proof of the conjecture, in the generic case of Type I. The complete proof with full details for all special sub-cases will be published elsewhere.
This example centres on an unusual *STeM* environment consisting of a hybrid blend of null and orthonormal formalisms. We have found that if this problem is tackled purely in the null NP frame or in the Orthonormal one, computational difficulties arise; expressions become large and unwieldy. This was easily seen with quick, short analyses using *STeM*. Part of the practical usefulness of *STeM* lies in its ability to allow the user to expeditiously "experiment" with various options composed of different formalisms and variables. Curiously, it is a hybrid choice, which we will describe below, that greatly facilitates the necessary computations. We note that for reasons of space limitations and aesthetic appeal we have chosen, via a Maple "*alias*" command to represent a complex conjugate by a preceding underscore instead of a trailing "_c". The user may wish to use other representations.
We begin with the NP environment where we initially choose the null tetrad to be the canonical one for a purely magnetic Weyl tensor of Petrov Type I, so that

$$\Psi_1 = \Psi_3 = 0, \Psi_0 = \Psi_4, \Psi_0 = -\bar{\Psi}_0, \Psi_2 = -\bar{\Psi}_2,$$

and since we have vacuum then

$$\Phi_{ab} = \Lambda = 0$$

We initialize the *STeM* environment as follows and indicate nonzero quantities.

```
>read ‘c:/STeMV1R1.mpl‘:with(STeM):formalism(np):with(student):
>alias(_c1=c1_c,_c2=c2_c,_c3=c3_c,_c4=c4_c,_t=t_c,_k=k_c,_n=n_c,_p=p_c,
_a=a_c,_b=b_c,_g=g_c,_e=e_c,_l=l_c,_m=m_c,_s=s_c,_r=r_c):
>W1 := 0: _W1 := 0:W3 := 0: _W3:= 0:_W0 := -W0: _W4 := -W4:W4 := W0:_W2:= -W2:
>L:=0:R00:=0: R01:=0: R02:=0:R10:=0: R11:= 0: R12:=0:R20:=0: R21:=0: R22:=0:
>stemassume({W0,W2}):
```

Next, we define an ordering on the spin coefficients based new variables $i1, .., i14$, $r1, .., r16$, and the curvature "compound" variables which will subsequently be introduced:

```
>stemdeforder([z1,z2,w6,w8,w9,w10,w13,w14,x9,x10,i6,i5,r6,r1,i1,r2,r3,i3,r4,i4,r5,r7,i7,r8,i8,r9,i9,r10,i10,r11,i11,r12,
i12,r13,r14,i13,i14,r15,r16]);
```

Based on extensive experience in using *STeM* over a period of more than 10 years, we have up to this point in time generally selected the "curvature type" variables to be in the highest ordered positions, which in this problem would amount to placing the w and x variables at the end of the ordering list. However, for this case it turns out that the lowest ordered positions is by far, the best. With these variables placed in the highest positions, one experiences significant issues regarding size and subsequent analysis of generated conditions. We stress that there are no hard-and-fast rules in deciding the order of the variables. This depends very much on the intricate structure of the system one is dealing with. As such we would recommend that the user should experiment a little with the ordering placed on variables before carrying out in-depth analysis on any problem. We now introduce the Ricci and Bianchi equations with complex conjugates, and combine.

```
>eins:=clean(map(eval,eqns(1,29))):ceins:=map(eval,map(conj,eins)):toteqs:=[op(eins),op(ceins)]:
```

The new operators [$E0$, $E1$, $E2$, $E3$], are introduced, given an ordering, properties defined, and then replaced, by issuing the following commands:

```
>defineOperator(‘E0‘, E0=(D+V)):defineOperator(‘E1‘, E1=(D-V)):defineOperator(‘E2‘,E2=(X+Y)):
>defineOperator(‘E3‘, E3=(X-Y)):stemdefoper([E3, E2, E1, E0]):
>stemdefine(E0,type=[real]):stemdefine(E1,type=[real]):stemdefine(E2,type=[real]):stemdefine(E3,type=
[imag]):replaceOperator(‘D‘ ,D=(E0+E1)/2):replaceOperator(‘V‘, V=(E0-E1)/2):
>replaceOperator(‘X‘, X=(E2+E3)/2):replaceOperator(‘Y‘, Y=(E2-E3)/2):
>
```

We note that because of space limitations, we have suppressed the obvious Maple output. Curiously, we have defined the new operators with a "lopsided" factor of 2. Presumably, for mainly aesthetic reasons one would want to see the "$\sqrt{2}$" introduced instead so as to mimic the correspondence with orthonormal operators as closely as possible. However, if one does this then Maple is forced to work with this irrational quantity and as a consequence, one finds that the performance is reduced.

Before proceeding further with the investigation of this problem, we would like to introduce new notions of *simple* and *compound* rotations. Now the spin coefficients $\{\kappa, \sigma, \rho, \pi, \tau, \mu, \lambda, \nu\}$ are proper weighted quantities but $\{\alpha, \beta, \gamma, \epsilon\}$ are not since under the spin-boost null rotations,$\{\widetilde{l^\mu} = A^{-1}l^\mu, \widetilde{n^\mu} = An^\mu, \widetilde{m^\mu} = e^{i\theta}m^\mu\}$, these quantities in general contain both derivatives of A and θ in their transformations. We say that these spin coefficients involve *compound* rotations. However consider the combinations $\{\alpha+\overline{\beta}, \alpha-\overline{\beta}, \epsilon+\overline{\epsilon}, \epsilon-\overline{\epsilon}, \gamma+\overline{\gamma}, \gamma-\overline{\gamma}\}$. They transform as

$$\widetilde{\alpha} + \widetilde{\overline{\beta}} = e^{-i\theta}\{\alpha + \overline{\beta} - A^{-1}\overline{\delta}A\}, \widetilde{\alpha} - \widetilde{\overline{\beta}} = e^{-i\theta}\{\alpha - \overline{\beta} + i\overline{\delta}\theta\},$$
$$\widetilde{\epsilon} + \widetilde{\overline{\epsilon}} = A^{-1}(\epsilon + \overline{\epsilon}) - A^{-2}DA, \widetilde{\epsilon} - \widetilde{\overline{\epsilon}} = A^{-1}(\epsilon - \overline{\epsilon}) + iA^{-1}D\theta,$$
$$\widetilde{\gamma} + \widetilde{\overline{\gamma}} = A(\gamma + \overline{\gamma}) - A^{-2}DA, \widetilde{\gamma} - \widetilde{\overline{\gamma}} = A(\gamma - \overline{\gamma}) - iA\Delta\theta.$$

Since the transformations of these quantities involve derivatives of A or θ but **not simultaneously** both, we say that they involve *simple* boost/rotations or are just *simple.* Therefore we would like to focus our attention on *simple* real and purely imaginary combinations. Hence we arrive at the following variable transformations:

$$r_1 = \alpha + \overline{\beta} + \overline{\alpha} + \beta, i_1 = \overline{\alpha} + \beta - \alpha - \overline{\beta},$$
$$r_2 = \overline{\alpha} - \beta + \alpha - \overline{\beta}, i_2 = \overline{\alpha} - \beta - \alpha + \overline{\beta},$$
$$r_3 = \epsilon + \overline{\epsilon} + \gamma + \overline{\gamma}, i_3 = \epsilon - \overline{\epsilon} + \overline{\gamma} - \gamma,$$
$$r_4 = \gamma + \overline{\gamma} - \epsilon - \overline{\epsilon}, i_4 = -\gamma + \overline{\gamma} - \epsilon + \overline{\epsilon}.$$

Next, we focus on the remaining spin coefficients. Now $\{\rho, \mu\}$ involve only the boost, and $\{\sigma, \overline{\lambda}\}$ both have the same spin factor. So continuing with the same general idea, by not mixing the different types, at this stage yet, and taking real and purely imaginary parts, modulo 2, we arrive at the following new variables for $\rho, \mu, \sigma - \overline{\lambda}, \sigma + \overline{\lambda}$:

$$r_5 = \rho + \overline{\mu} + \overline{\rho} + \mu, i_5 = \overline{\rho} + \mu - \rho - \overline{\mu},$$
$$r_6 = \rho - \overline{\mu} + \overline{\rho} - \mu, i_6 = \overline{\rho} - \mu - \rho + \overline{\mu},$$
$$r_7 = \sigma + \overline{\lambda} + \overline{\sigma} + \lambda, i_7 = \overline{\sigma} + \lambda - \sigma - \overline{\lambda},$$
$$r_8 = \sigma - \overline{\lambda} + \overline{\sigma} - \lambda, i_8 = \overline{\sigma} - \lambda - \sigma + \overline{\lambda}.$$

This leaves the remaining spin coefficients $\{\kappa, \tau, \pi, \nu\}$. By looking at similar spin transformation properties, we should consider the basic combinations: $\{\tau + \overline{\pi}, \tau - \overline{\pi}, \kappa + \overline{\nu}, \kappa - \overline{\nu}\}$. Thus we take these complex combinations, and form their real and purely imaginary parts, modulo 2,

$$c_1 = \kappa + \overline{\nu} + \tau + \overline{\pi}, c_2 = \tau + \overline{\pi} - \kappa - \overline{\nu},$$
$$c_3 = \overline{\nu} - \kappa + \overline{\pi} - \tau, c_4 = \overline{\pi} - \tau + \kappa - \overline{\nu},$$
$$r_9 = c_1 + \overline{c}_1, i_9 = c_1 - \overline{c}_1,$$
$$r_{10} = c_2 + \overline{c}_2, i_{10} = c_2 - \overline{c}_2,$$
$$r_{11} = c_3 + \overline{c}_3, i_{11} = c_3 - \overline{c}_3,$$
$$r_{12} = c_4 + \overline{c}_4, i_{12} = c_4 - \overline{c}_4,$$

Further, we find that it is advantageous to make the following additional changes:

$$i_5 = (i_{13} - i_{14})/2, i_7 = (i_{13} + i_{14})/2, r_4 = (r_{14} - r_{13} + 2r_6)/4,$$
$$r_7 = (r_{15} + r_{16})/2, r_8 = (r_{13} + r_{14})/2, r_5 = (r_{15} - r_{16})/2.$$

Interestingly, these new variables correspond to the orthonormal combinations, modulo a numerical factor, $n_{11} + n_{22} - n_{33}$, $n_{11} - n_{22} + n_{33}$, $\theta_{11} - \theta_{22}$, $\theta_{11} - \theta_{33}$, $a_1 + n_{23}$, $a_1 - n_{23}$, similarly for the others. These combinations are significantly simpler than the corresponding NP ones. As yet, it is not clear to us how these and the other variable combinations can be found while completely within the orthonormal formalism.
Our input is, with the addition of z_1 and z_2, as follows.

```
>set1:={z1=W0+3*W2,z2=W0-3*W2}
>set2:={r1 = _a + b + a + _b, i1 = _a + b - a - _b, r2 = _a - b + a - _b, i2 = _a - b - a + _b}:
>set3:={r3 = _g+e+_e+g, i3 = _g+e-g-_e, r4 = _g-e+g-_e, i4 = _g-e-g+_e}:
>set4:={r5=_r + m + r + _m, i5=_r + m - r - _m, r6=_r - m + r - _m, i6=_r - m - r + _m}:
>set5:={ c1 = k+t+_p+_n, c2 = t-k+_p-_n, c3 = _n-k+_p-t, c4 = _p-t-_n+k}:
>set5a=map(conj,set5):
>set6:={r7 = _s+l+s+_l, i7 = _s+l-s-_l, r8 = _s-l+s-_l, i8 = _s-l-s+_l}:
>set7:={c1=(r9+i9) /2, _c1 =( r9-i9)/2, c2=(r10+i10)/2, _c2 = (r10-i10)/2, c3=( r11+i11)/2 ,_c3=(r11- i11)/2,
c4=(r12 +i12)/2, _c4=(r12-i12)/2, i5 = (1/2)*i13-(1/2)*i14, i7 = (1/2)*i14+(1/2)*i13,
r4 = (1/4)*r14-(1/4)*r13+(1/2)*r6, r8 = (1/2)*r13+(1/2)*r14, r5 = (1/2)*r15-(1/2)*r16, r7 = (1/2)*r16+(1/2)*r15}:
```

Our semi-final variable set is $\{r1, r2, r3, r6, r9, .., r16, i1, .., i4, i6, i8, .., i14, z1, z2\}$, correctly yielding 24 independent real/imaginary functions. We introduce the properties of the new variables, into the session by two small "do" loops with just one entry:

```
>for j from 1 to 16 do stemdefine(r||j,type=['real']):od:
>for j from 1 to 14 do stemdefine(i||j,type=['imag']):od:
```

and

```
>stemdefine(z1,type=['imag']):stemdefine(z2,type=['imag']):
```

We need one final change which scales the variables $\{i6, i8, i9, i10, i13, i14, r9, r10\}$ by the curvature components and by $\{i1, i3, i4, r1\}$.

```
>set8:={i10 = w10*i1/z2, i13 = i3*w13/z1, i14 = i3*w14/z2, i6 = w6*i4/(z2*z1), i8 = w8*i4/(z2*z1),
i9 = w9*i1/z1, r10 = x10*r1/z1, r9 = r1*x9/z2}:
>stemdefine(z1,type=['imag']):stemdefine(z2,type=['imag']):stemdefine(x9,type=['imag']):stemdefine(x10,type=['imag']):
>stemdefine(w13,type=['imag']):stemdefine(w14,type=['imag']):stemdefine(w6,type=['real']):stemdefine(w8,type=['real']):
>stemdefine(w9,type=['imag']):stemdefine(w10,type=['imag']):
```

We can easily compare our new variables in terms of the standard orthonormal ones as follows, for example:

>factor(solve(map(convertNP2Orth,set2),{r1,i1,r2,i2}));

$$\{i1 = I\sqrt{2}(s13\text{-}w2),\ i2 = I\sqrt{2}(a3\text{-}n12),\ r1 = \sqrt{2}(s12+w3),\ r2 = (a2+n13)\sqrt{2}\}$$

>factor(solve(map(convertNP2Orth,set6),{r7,i7,r8,i8}));

$$\{i7 = I\sqrt{2}(n22\text{-}n33),\ i8 = (2I)\sqrt{2}s23,\ r7 = 2\sqrt{2}n23,\ r8 = -(H2\text{-}H3)\sqrt{2}\}$$

A check of all the new variables shows that they are intriguingly simple combinations of the orthonormal ones. However, curiously, we have found that, for this problem, a direct application of the standard othornormal system leads to insurmountable difficulties.
We complete the setup of the *STeM* environment by making a global total switch from the NP standard variables to our new ones.

```
>solve({op(set1),op(set2),op(set3),op(set4),op(set5),op(set5a),op(set6)},
{a,_a,b,_b,g,_g,p,_p,t,_t,e,_e,n,_n,m,_m,k,_k,l,_l,r,_r,s,_s,W0,W2}):assign(%):
>assign(set7):assign(set8):
```

The stage is now set for the analysis of the complete set of the extended field equations, with the application of *stemnormal*. We display some of the output with decision choices.

```
>toteqsa:=stemnormal(toteqs,switch=[algebraic,factor]):

                                "Obtain algebraic equation"

i1*w10-i1*w9+r1*x10-r1*x9

                    Substitute this algebraic equation into the system? (y,n) Y
                                "Obtain algebraic equation"

r1*(x10-x9)

                    Substitute this algebraic equation into the system? (y,n) Y
                                  factor , 1, : , r1
                                factor , 2, : , x10 - x9
                            Use factor number?(ter=terminate) 2
                                "Obtain algebraic equation"

i1

                    Substitute this algebraic equation into the system? (y,n) Y
                                "Obtain algebraic equation"

x10-2*z1-2*z2

                    Substitute this algebraic equation into the system? (y,n) Y
                                "Obtain algebraic equation"

x9-2*z1-2*z2

                    Substitute this algebraic equation into the system? (y,n) Y
                                "Obtain algebraic equation"

i3*w13*z1*z2+i3*w14*z1*z2+i4*w6*z1-i4*w6*z2-i4*w8*z1-i4*w8*z2

                    Substitute this algebraic equation into the system? (y,n) Y
                                "Obtain algebraic equation"

i3*(w13+w14)

                    Substitute this algebraic equation into the system? (y,n) Y
```

factor , 1, : , i3
factor , 2, : , w13 + w14
Use factor number?(ter=terminate) 2
"Obtain algebraic equation"

i4

Substitute this algebraic equation into the system? (y,n) Y
"Obtain algebraic equation"

w13+2*z1+2*z2

•
•
•

At each step, when *stemnormal* encounters/generates a polynomial or product of factors, it stops and queries the user for a decision. The user may or may not, as indicated by a "Y" or an "N", substitute it back into the system for further reduction/normalisation. For polynomials with multiple factors, the user must indicate which factor is to be substituted back into the normalisation process to continue. As indicated here, we have opted to substitute back in only the smaller conditions, whereas the larger conditions are set aside into the set *getUnusedEqn().* The polynomials substituted back in are placed in the set *getUsedEqn().* Next, we find out the details of *toteqsa* by

> showStatistic(toteqsa);

Number of equations is , 53
Number of polynomial equations is , 6
Number of differential equations is , 47
Equations' lengths are
[17, 17, 17, 16, 5, 5, 75, 75, 75, 88, 75, 75, 75, 88, 177, 441, 343, 445, 201, 525, 192, 791, 383, 582, 5135, 743, 547,· · ·]
Equations' LHS are
[w13, w14, x10, x9, i1, i4, E3(z1), E2(z1), E1(z1), E0(z1), E3(z2), E2(z2), E1(z2), E0(z2), E3(r1), E3(r11),
E1(r11), E0(r12), E3(r13), E2(r13), E2(r14), E1(r14), E1(r15), E0(r15), E3(r16), E2(r16), E0(r16), E1(r2),
E0(r2), E3(r3), E2(r3), E1(r3), E0(r3), E3(r6), E2(r6), E1(r6),· · ·]

Hence we have 53 equations in total, 6 being polynomials, which are placed at the beginning of the list *toteqsa,* and 47 differential equations. In addition, the Maple lengths and left hand sides of each equation are displayed. Next, we can collect all polynomial conditions generated and display the details of this set.

>totpols:=stemsorte([op(getUsedEqn()),op(getUnusedEqn()),op(getUsedFactor()),seq(toteqsa[i],i=1..6)]):
>showStatistic(totpols);indets(totpols);

Number of equations are , 24
Number of polynomial equations are , 24
Number of differential equations are , 0
Equations' length are
[17, 17, 17, 17, 17, 17, 16, 16, 21, 2, 5, 59, 22, 152, 193, 262, 325, 368, 393, 443, 467, 2, 5, 127]
Equations' LHS are
[none, w13, none, w14, none, x10, none, x9, none, none, i1, none, none, none, none, none, none, none, none, none, none,
none, i4, none]
{i1, i11, i12, i2, i3, i4, r1, r13, r14, w10, w13, w14, w6, w8, w9, x10, x9, z1, z2}

>showCommutator();

Number of full commutators calculated is , 13
They are
[E1, E0, z1], [E2, E0, z1], [E2, E1, z1], [E3, E0, z1], [E3, E1, z1], [E3, E2, z1]
[E1, E0, z2], [E2, E0, z2], [E2, E1, z2], [E3, E0, z2], [E3, E1, z2], [E3, E2, z2]
[E2, E1, i11]
Number of half commutators calculated is , 0

Therefore there are 24 polynomial conditions of the indicated lengths. Even though the system *totpols* is quite simple we can in fact simplify it much further, using the routines *minimisePoly* and *reduceMinSys*, as follows. Before that,we should query *STeM* to indicate what assumptions have been made.

```
>showAssumption();
```

Non-zero factors
i3 ≠ 0
r1 ≠ 0
Non-zero denominators
r1 ≠ 0
w6 ≠ 0
w8 ≠ 0
z1 ≠ 0
z2 ≠ 0
w6 - w8 ≠ 0
-z1 - z2 ≠ 0
-w10 + w9 ≠ 0
-w6 z1 + w6 z2 + w8 z1 + w8 z2 ≠ 0

```
>totpolb:=minimisePoly(totpols):
>totpolc:=reduceMinSys(totpolb):
>findops(totpolc);
```

11, 12

```
>totpolc[11]:=totpolc[11]/z1/z2;
```

totpolc[11] := 3*i11*r1*z2-3*i12*r1*z1-6*i2*r1*z1+6*i2*r1*z2-6*i3*r14*z1+2*z1^2-8*z1*z2

```
>totpolc[12]:=totpolc[12]/(x9-2*z1);
```

totpolc[12] := i11*r1*x9+2*i11*r1*z1-2*i12*r1*z1+2*i2*r1*x9-2*i3*r13*z1-i3*r14*x9-2*i3*r14*z1

```
>showStatistic(totpolc);
```

Number of equations is , 13
Number of polynomial equations is , 13
Number of differential equations is , 0
Equations' length are
[2, 2, 12, 12, 12, 16, 92, 104, 104, 115, 136, 140, 167]
Equations' LHS are
[none, none, none, none, none, none, none, none, none, none,
none, none, none]

We see that the system of polynomials has been reduced considerably. Finally, we use the routine *Polynomial-System,* in Maple, to solve *totpolc.*

```
> with(SolveTools):
>sola:=PolynomialSystem(totpolc,{z1<>0,z2<>0});
```

sola := {i1 = 0, i11 = i11, i12 = i12, i2 = i2, i3 = i3, i4 = 0, r1 = 2z1/(2i2+i12), r13 = -2z1/i3,
r14 = -2z1(-i12+i11)/(i3(2i2+i12)), w13 = 2z1, w14 = -2z1, x10 = -2z1, x9 = -2z1, z1 = z1, z2 = -2z1}

Note that $2i_2+i_{12} \neq 0$ since $z_1 z_2 \neq 0$ There is a solution to the conditions obtained so we are not quite done. To finish this example, we simply apply $E0$ to $z_2 = -2z_1$.

```
>fsubs(sola,stemreduce(E0(z2=-2*z1),toteqsa));
```

-12z1^2/i3

Thus the proof for this generic case is complete. We will stop here. There are of course, a number of special cases, that need to be analyzed, as given by $r_1 i_3 w_6 w_8 (w_6 - w_8)(w_9 - w_{10})(-w_6 z_1 + w_6 z_2 + w_8 z_1 + w_8 z_2) = 0$. It turns out that some of these subcases are more involved. However, the overall resulting systems, that we do find, have the desirable feature that there occurs a good measure of variable separation in the constituent conditions which assists the analysis considerably, though, admittedly, there results a large number of branching cases.
It is apparent from the above work, that it was not necessary, at any stage, to explicitly display the complete system. There are a number of routines in *STeM* which assist the user to extract essentially a great deal of the important/relevant information, of any given system without displaying it.
As a final point, it is conceivable that the new variables/system introduced here may prove to be useful for other applications. This matter is currently under investigation.

4.3 Example 3: Shear-free perfect fluids

We now present this last application as an interesting demonstration of a quite difficult problem made relatively straightforward by *STeM*. General relativistic, shear-free, perfect fluids which obey a barotropic equation of state $p = p(\mu)$ (μ is the energy density and p is the pressure) such that $\mu + p \neq 0$ have been studied extensively over the past fifty years. Much effort, by numerous researchers, has gone into showing that such fluids must be either nonexpanding (the expansion scalar $\theta = 0$) or nonrotating (the vorticity scalar $\omega = 0$). (The reader is referred to the review by Collins[5] and Van den Bergh and Slobodeanu[26] for comprehensive discussions of shear-free fluids and the associated conjecture.) Indeed, the conjecture has been demonstrated to hold in numerous special cases such as:

- spatial homogeneity,[1,13]the cases where $p^{'} = 0, \pm 1/3, 1/9$,[6,8,11,23,25]
- the case where the vorticity ω^a is parallel to the acceleration $\dot{u}^a$,[18,30]
- the case when $\theta = \theta(\mu)$,[14]
- the case when $\theta = \theta(\omega)$,[19]
- purely electric[4] or purely magnetic[8] spacetimes,
- Petrov Type O, Petrov Type N,[2] or Type III[3] spacetimes.

The conjecture has been also studied by Van den Bergh and Slobodeanu(VdBS)[26] for a linear equation of state and Van den Bergh for the case where $\mathfrak{D}_1 = \mathfrak{R}_2 = 0$, with a Killing vector.[27](see Appendix). In both cases the conjecture was established. A summary of their proofs will now be given.
The starting point is the setting up of the EFE and the alignment of the tetrad, as follows. The Einstein field equations are

$$R_{ab} - \frac{1}{2} R g_{ab} = \mu u_a u_b + p h_{ab}, \tag{1}$$

with $p = p(\mu)$,where $\mathbf{u}$ is the future-pointing (time-like) unit tangent vector to the flow, and $h_{ab} = g_{ab} + u_a u_b$ is the projection tensor into the rest space of the observers with 4-velocity $\mathbf{u}$. Next, $\mathbf{e}_0$ and $\mathbf{e}_3$ are aligned with $\mathbf{u}$ and ω, respectively, such that $\omega = \omega \mathbf{e}_3 \neq 0$. The relevant variables then become μ, p, θ, ω, $\dot{u}_\alpha$, Ω_α, $\sigma_{\alpha\beta} = 0$, together with the quantities $n_{\alpha\beta}$ and a_α. Latin indices will be spacetime indices while Greek indices will take the values 1, 2 and 3.
Secondly, it is always possible, using the remaining group freedom and making use of the Jacobi identities and field equations, to further specialize the tetrad so as to achieve

$$\Omega_1 = \Omega_2 = \Omega_3 + \omega = 0, \tag{2}$$

$$n_{11} = n_{22} \equiv n. \tag{3}$$

Now, if $n_{12} \neq 0$, the tetrad is completely fixed. On the other hand, if $n_{12} = 0$ ($\mathfrak{D}_3 = 0$) then there exists additional gauge freedom of a *basic* rotation in the $< \mathbf{e}_1, \mathbf{e}_2 >$ plane. Next,VdBS replace $n_{\alpha\beta}$ and a_α with the new variables q_α and r_α, defined by $n_{\alpha-1\,\alpha+1} = (r_\alpha + q_\alpha)/2$, and $a_\alpha = (r_\alpha - q_\alpha)/2$, expressions which must be read modulo 3. In addition, new extension variables, z_a, defined by

$$\partial_0 \theta = -\theta^2/3 + 2\omega^2 - (\mu + 3p)/2 + j, \;\; \partial_0 \theta = z_0, \partial_1 \theta = z_1, \partial_2 \theta = z_2, \partial_3 \theta = z_3, \tag{4}$$

are also introduced.

As regards the proof for the linear equation of state, it relies substantially on *basic* functions, which are functions f that are conserved along the flow of the fluid i.e. $\mathbf{e}_0(f) = 0$. In this case, VdBS, after some initial general results are established, introduce the following variables

$$\lambda \equiv \exp(\int \frac{d\mu}{3(p+\mu)}), p' = \frac{r}{3}-1, p = (\frac{r}{3}-1)\lambda^r - \mu_0, \mu = \lambda^r + \mu_0$$

(μ_0, r constants, since p is linear) The proof is essentially composed of two parts. The first part consists of showing that either $\dot{U}_3 \equiv \dot{u}_3/(\lambda p')$, is basic, in which case they are done by their theorem 2 or that with further analysis there exists a Killing vector parallel to the vorticity. Further, if $\theta\omega \neq 0$, they obtain

$$\dot{u}_3 = z_3 = q_3 = r_3 = E_{13} = E_{23} = 0, \tag{5}$$

together with an additional condition on E_{33}. In this case $n_{12} = 0$, and they use the additional basic rotation freedom to set $\mathfrak{b}_2 = \mathfrak{b}_1$.

Next, they translate these conditions into their basic variables equivalent: $\mathfrak{b}_3 = \mathfrak{D}_3 = \mathfrak{R}_3 = \mathfrak{C}_{13} = \mathfrak{C}_{23} = 0$ and start deriving conditions from the field equations and first and second Bianchi identities There follows a long analysis (with $p' \neq -1/11, 1/4$) of conditions involving the full set of basic variables and their extensions, $\dot{U}_\alpha$, and multiple time-like propagations (their repeated time-propagations should be on equation (80) not as indicated equation (79)) with resultants, to obtain three relations of the form

$$\mathcal{P}_i = a_i U^2 + b_i U\theta^2 + c_i U + d_i\theta^2 + e_i, \ i = 1, 2, 3.$$

a_i, b_i, c_i, d_i, and e_i have basic coefficients of degree 2 and 20 in respectively $p+\mu$ and λ, and $U = \dot{U}_1^2 + \dot{U}_2^2$, where $\dot{U}_\alpha \equiv \dot{u}_\alpha/(\lambda p')$. VdBS then eliminate θ,by resultants to obtain two relations of the form $\mathcal{R}_i(U, p+\mu, \lambda, \mu) = 0$, $i = 1, 2$.(We note a minor correction that the $\mathcal{R}_i$ must be a function of μ and the constant μ_0 in addition to the $\mu + p$ combination with U and λ). The $\mathcal{R}_i$ are of degree 3 in U, and of degree 9 in $p+\mu$ and 30 in λ.Taking one final resultant with respect to U yields $\mathcal{F}$, which factors as

$$\begin{aligned}\mathcal{F} = &\mathfrak{o} p'^{13}(p+\mu)^{18}\lambda^{18}(4p'-1)(6p'+1)(3p'-1)^9(21p'+11)^4 \\ &\times (11p'+1)^6(117p'^2 + 69p' + 2)(96p'^2 + 47p' + 1)\mathcal{F}_1\mathcal{F}_2\end{aligned}$$

which must vanish for consistency. $\mathcal{F}_1$and $\mathcal{F}_2$ respectively of degrees (6,20) and (21,70) in $p+\mu$ and λ,with coefficients depending on $\mathfrak{o}, \mathfrak{J}, \mathfrak{b}_1, \mathfrak{C}_3, \mathfrak{X}_3$. Presumably, these polynomials must be relatively large though the authors do not give any indication of the overall (Maple or Mathematica) size. They then argue that any such polynomial may be written in the form $\Sigma_{i,j} c_{i,j}\lambda^{ir+j} = 0$,where $c_{i,j}$ are functions of the essential indicated basic variables and one extension variable (see their Appendix B). We note that they next need to consider various possible values of rational and irrational values of r in combination with multiple applications of resultants. These considerations result in rather lengthy complex arguments which will not be summarized here due to space limitations. Eventually, the authors arrive at a number of separate case conclusions that all lead to $\theta\omega = 0$. They indicate that full details, in Maple or Mathematica worksheets are available, since they have only described the outline of the proof; the output being too lengthy for publication.

Using *STeM*, we have been able to construct a proof of the gamma law case which does not rely on any complicated arguments concerning rational/irrational powers of (possibly complex) λ or r, relies substantially on much smaller equations and, remarkably, does NOT require the use of any resultant computations. The full details of this new approach will be published elsewhere.

For the proof of the conjecture for the case when $\kappa \equiv \mathfrak{D}_1 - i\mathfrak{R}_2 = 0$, and general $p = p(\mu)$ Van den Bergh[27] utilizes a newly developed complex formalism, which is some what reminiscent of the GHP variant of NP, of the reduced orthonormal system with basic variables. Briefly, the complex scaled fluid velocity is defined by $\dot{u} = \dot{U}_1 + i\dot{U}_2$ The quantities $\kappa, \pi, \rho, \beta, \mathcal{E}_1, \mathcal{E}_2, \mathcal{E}$, which are simple complex combinations of the basic variables together with $\mathcal{O} \equiv \mathfrak{o}$ and $\mathcal{J}$, and which are well weighted under SO(2), appear explicitly in the equations whereas the quantities τ and ν which are not well weighted do not appear explicitly but form part of the well weighted operators $\eth, \eth'$ and $\mathcal{Z}$. Finally after some recombining, there results a complex system consisting of field equations with integrability conditions. The conditions $\kappa = 0$ and the existence of a Killing vector are then explicitly introduced with some immediately resulting simple conditions. Next he considers two separate cases based on whether $6p'G + 3p' + 1$ is zero or not. In the first case, he obtains a real equation for $|\dot{u}|^2$.Time propagating this condition and combining with other equations leads to a relatively large equation of the form

$$\lambda\bar{\beta}\dot{u} = -iQ_1\epsilon_0\theta\mathcal{J} + iQ_2\mathcal{O}^2\theta/\epsilon_0 + Q_3\mathcal{O}\mathcal{J}\lambda + Q_4\mathfrak{B}\mathcal{O}\lambda + Q_5\mathcal{O}\mathcal{J}\lambda + Q_6\mathfrak{B}\mathcal{O}\lambda + Q_7\lambda^3\epsilon_0\mathcal{O} + Q_8\lambda\mathcal{O}^3/\epsilon_0^2$$

where Q_i are rational functions of $p'$and coefficients are basic including $\mathfrak{B}$ (curiously, $\mathfrak{B}$ which is an extension or auxiliary variable, is never explicitly determined only the integrability condition that it satisfies). The key

feature of this equation is that θ is present only in its imaginary part. Taking the real part of this condition and combining with multiple time propagations of resulting equations leads to, after elimination of some variables presumably by resultants, a linear equation of state and hence the proof is complete for this branch. In the case $6p'G + 3p' + 1 \neq 0$, the situation is considerably more complicated. He begins with solving a large integrability equation for $\dot{u}$,which when combined with the time propagation of the same equation leads to a relatively lengthy equation of the form

$$C|\dot{u}|^2 + H_1\mathcal{J} + H_2\mathfrak{B} + H_3\mathcal{O}^2/\epsilon_0^2 + H_4\epsilon_0\lambda^2 = 0$$

where H_i are rational functions of G and p', C is a rational function of μ, p, p', G and G'. Firstly, he notes that if $C = 0$, then combining with two time propagations of the above equation and some variable elimination there results a condition of the form $P_1\mathcal{O}^2 + P_2 = 0$, where P_1 and P_2 are, presumably (the actual sizes are not stated) large polynomial functions, of degree 9 in G and of degree 20 and 17 in p', respectively. Finally, a resultant (presumably large hence with a mod) is used between P_1 and P_2 to eliminate G.Since the resultant does not vanish identically, it follows that the equation of state is linear and the proof is complete, in this case. Finally, for $C \neq 0$, then after some propagations with recombining of equations, he obtains an equation of the form

$$\mathfrak{B}\mathcal{F}_1 + \mathcal{J}\mathcal{F}_2 + \mathcal{O}^2\mathcal{F}_3 + \mathcal{F}_4 = 0,$$

where $\mathcal{F}_i$ are somewhat lengthy rational functions of $\lambda, \epsilon_0, \mu, p, p', G$, and G'. At this stage lemma 1 is invoked which is just basically a separation-of-variables argument focusing on the, constant coefficient linear independence of the set $\{\mathfrak{B},\mathcal{J}, \mathcal{O}^2, 1\}$. It follows that there are five distinct separate cases to consider delineated by the zero/nonzero or constant ratio properties of $\{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3, \mathcal{F}_4\}$.After a very lengthy and tedious computation, it is eventually shown that all possible cases lead to a linear equation of state. We have also examined this case and using *STeM*, have obtained a relatively short concise proof which does not require any complicated arguments. The full details of this new approach will also be published elsewhere.

In this our final example, we shall investigate spacetimes with a barotropic equation of state where the acceleration is NOT orthogonal to the vorticity and with a non-gamma law equation of state. This assumption is equivalent to requiring that there does NOT exist a Killing vector parallel to the vorticity. Our proof of the conjecture for the case when $\dot{\mathbf{u}} \cdot \omega \neq \mathbf{0}$ also relies substantially on *basic* functions (See Appendix) .We shall suppress much of the output generated by *STeM* (it is not necessary to see it) but display only statistically important information with key conditions, which are relatively small. We shall argue by contradiction, firstly assuming $\theta\omega \neq 0$, and then showing that either $\theta = 0$ or $\omega = 0$ must follow. Due to space limitations we shall only present one generic branch of the proof. The complete proof will be published elsewhere.

.

4.3.1 $\dot{\mathbf{u}} \cdot \omega \neq \mathbf{0} \Rightarrow \omega\theta = 0$

We begin our analysis by setting up initially the *STeM* environment the same as that of VdBS which includes the familiar start-up input for the EFE and tetrad alignment as described above.

```
>read `c:/STeMV1R1.mpl`:with(STeM):formalism(orth):
>stemdeforder([p,mu,theta,omega,P1,P2,P3,P4,P5,u1,u2,u3,J,b1,b2,b3,C0,C3,D1,D2,D3,R1,R2,R3,C12,C13,C23,
seq(X||i,i=1..16),z1,z2,z3,q1,q2,q3,r1,r2,r3,n,n33,E11,E22,E12,E13,E23,M11,M22,M12,M13,M23,j]);
>stemdefoper([e3,e2,e1,e0])::stemassume(P1,P+mu,omega,theta,P,P1+1,P1-1,mu):
>H:=theta:P:=p:M:=mu:H1:=H/3:H2:=H/3:H3:=H/3:s12:=0:s23:=0:s13:=0:w1:=0:w2:=0:w3:=omega:
O3:=-omega:n22:=n:n11:=n:
>e0(P):=P1*e0(M):e1(P):=P1*e1(M):e2(P):=P1*e2(M):e3(P):=P1*e3(M):
>e0(P1):=P2*e0(M):e1(P1):=P2*e1(M):e2(P1):=P2*e2(M):e3(P1):=P2*e3(M):
>e0(P2):=P3*e0(M):e1(P2):=P3*e1(M):e2(P2):=P3*e2(M):e3(P2):=P3*e3(M):
>e0(P3):=P4*e0(M):e1(P3):=P4*e1(M):e2(P3):=P4*e2(M):e3(P3):=P4*e3(M):
>E33:=-E22-E11:M33:=-M22-M11:O1:=0:O2:=0:O3:=-omega:
>H:=theta:P:=p:M:=mu:H1:=H/3:H2:=H/3:H3:=H/3:
>eqns(140,145);assign(%):
```

Next, we normalise the complete system of equations, collect all resulting polynomials and carry out consecutively two sets of commutators with subsequent normalisations.

```
>hqa:=stemnormal([op(eqns(1,58)),op(eqns(136,139))],switch=[factor,algebraic]):
```

"Obtain algebraic equation"

-3*n33*omega+2*z3

Substitute this algebraic equation into the system? (y,n) Y
"Obtain algebraic equation"

3*omega*r2-3*M23-z1

Substitute this algebraic equation into the system? (y,n) Y
"Obtain algebraic equation"

3*omega*q1+3*M13-z2

Substitute this algebraic equation into the system? (y,n) Y
"Obtain algebraic equation"

M12

Substitute this algebraic equation into the system? (y,n) Y
"Obtain algebraic equation"

omega*q3+omega*r3+M11-M22

Substitute this algebraic equation into the system? (y,n) Y
"Obtain algebraic equation"

omega*r3+omega*u3+M11

Substitute this algebraic equation into the system? (y,n) Y
"Obtain algebraic equation"

-omega*q3+omega*u3+M22

Substitute this algebraic equation into the system? (y,n) Y

>specs(hqa);

indets = {E11, E12, E13, E22, E23, M11, M12, M13, M22, M23, P1, j, μ, n, n33, ω, p, q1, q2, q3, r1, r2, r3, θ, u1, u2, u3, z1, z2, z3, e_0(E11), e_0(E12), e_0(E13), e_0(E22), e_0(E23), $e_0(\mu)$, e_0(n), $e_0(\omega)$, e_0(q1), e_0(q2), e_0(q3), e_0(r1), e_0(r2), e_0(r3), $e_0(\theta)$, e_0(u3), e_0(z1), e_0(z2), e_0(z3), e_1(E11), e_1(E12), e_1(E13), e_1(E22), e_1(E23), $e_1(\mu)$, e_1(n), $e_1(\omega)$, e_1(q1), e_1(q3), e_1(r1), e_1(r2), e_1(r3), $e_1(\theta)$, e_1(u1), e_1(u2), e_1(u3), e_1(z1), e_1(z2), e_1(z3), e_2(E11), e_2(E12), e_2(E13), e_2(E22), e_2(E23), $\cdots$}
length = 10056
nops = 59
type = list

>showAssumption();

Non-zero assumption predefined by user
P1 $\neq$ 0
Non-zero denominators
$\omega \neq 0$

>reva:=solve([3*omega*r2-3*M23-z1,3*omega*q1+3*M13-z2,omega*r3+omega*u3+M11, -omega*q3+omega*u3+M22,
-3*n33*omega+2*z3],{M13,M23,M11,M22,n33}):M12:=0:
>allcomas:=separate(findComm(hqa)):
>maplen(allcomas[1]);

[]

>showAssumption();

Non-zero assumption predefined by user
P1 $\neq$ 0
Non-zero factors
p + $\mu \neq 0$
Non-zero denominators
$\omega \neq 0$

>hqb:=stemnormal([op(hqa),op(allcomas[2])],switch=[algebraic,factor]):
>allcombs:=separate(findComm(hqb)):
>maplen(allcombs[1]);

[]

>maplen(allcombs[2]);

[256, 256, 256, 1028, 1028, 1100, 1100, 1270, 1323]

>showAssumption();

Non-zero assumption predefined by user
$P1 \neq 0$
Non-zero factors
$p + \mu \neq 0$
Non-zero denominators
$\omega \neq 0$
$3\ P1 + 1 \neq 0$

>showCommutator();

Number of full commutators calculated is , 36
They are
$[e_1, e_0, \mu]$, $[e_2, e_0, \mu]$, $[e_2, e_1, \mu]$, $[e_3, e_0, \mu]$, $[e_3, e_1, \mu]$, $[e_3, e_2, \mu]$ $[e_1, e_0, \omega]$, $[e_2, e_0, \omega]$, $[e_2, e_1, \omega]$,
$[e_3, e_0, \omega]$, $[e_3, e_1, \omega]$, $[e_3, e_2, \omega][e_1, e_0, \theta]$, $[e_2, e_0, \theta]$, $[e_2, e_1, \theta]$, $[e_3, e_0, \theta]$, $[e_3, e_1, \theta]$, $[e_3, e_2, \theta]$
$[e_1, e_0, u1]$, $[e_2, e_0, u1]$, $[e_2, e_1, u1]$, $[e_3, e_0, u1]$, $[e_3, e_1, u1]$, $[e_3, e_2, u1]$ $[e_1, e_0, u2]$, $[e_2, e_0, u2]$,
$[e_2, e_1, u2]$, $[e_3, e_0, u2]$, $[e_3, e_1, u2]$, $[e_3, e_2, u2]$ $[e_1, e_0, u3]$, $[e_2, e_0, u3]$, $[e_2, e_1, u3]$, $[e_3, e_0, u3]$,
$[e_3, e_1, u3]$, $[e_3, e_2, u3]$
Number of half commutators calculated is , 0

>hqc:=stemnormal([op(hqb),op(allcombs[2])],switch=[algebraic,factor]):
>specs(%);

indets = {E11, E12, E13, E22, E23, M11, M13, M22, M23, P1, P2, P3, j, μ, n, n33, ω, p, q1,
q2, q3, r1, r2, r3, θ, u1, u2, u3, z1, z2, z3, e_0(E11), e_0(E12), e_0(E13), e_0(E22), e_0(E23), e_0(j),
$e_0(\mu)$, e_0(n), $e_0(\omega)$, e_0(q1), e_0(q2), e_0(q3), e_0(r1), e_0(r2), e_0(r3), $e_0(\theta)$, e_0(u1), e_0(u2), e_0(u3),
e_0(z1), e_0(z2), e_0(z3),e_1(E11), e_1(E12), e_1(E13), e_1(E22), e_1(E23), e_1(j), $e_1(\mu)$, e_1(n), $e_1(\omega)$,
e_1(q1), e_1(q3), e_1(r1), e_1(r2), e_1(r3), $e_1(\theta)$, e_1(u1), e_1(u2), e_1(u3), e_1(z1), e_1(z2), e_1(z3),
e_2(E11),e_2(E12), e_2(E13), e_2(E22), e_2(E23), e_2(j), $\cdots$}
length = 40968
nops = 76
type = list

>showAssumption();

Non-zero assumption predefined by user
$P1 \neq 0$
Non-zero factors
$p + \mu \neq 0$
Non-zero denominators
$\omega \neq 0$
$3\ P1 + 1 \neq 0$

We proceed by manually carry out the indicated commutators and all subsequent polynomial conditions are collected and reduced

>comj:=stemreduce(autocomm(j),hqc):comz1:=stemreduce(autocomm(z1),hqc):
>comz2:=stemreduce(autocomm(z2),hqc):comz3:=stemreduce(autocomm(z3),hqc):
>com10q3:=stemreduce(comm(e1,e0,q3),hqc):com10r2:=stemreduce(comm(e1,e0,r2),hqc):
>com20r3:=stemreduce(comm(e2,e0,r3),hqc):com20q1:=stemreduce(comm(e2,e0,q1),hqc):
>allcomcs:=minimisePoly(sortl([com20q1,com20r3,com10r2,com10q3,op(comz3),op(comz2),op(comz1),op(comj)])):
>maplen(%);clearall();

[1456, 2256, 2256, 2859, 2859, 4408, 4408, 5889, 8572, 8572, 13777, 13777, 16076, 16980, 16980, 18746, 18746, 19117]

Next, we determine all possible higher order commutators. This process produces only differential conditions of the indicated Maple lengths

```
>half:=findHalfComm(hqc):
>halfs:=separate([op(half)]):
>des:=sortl(halfs[2]):
>maplen(%);
```

$$[76, 1090, 1210, 1314, 1314, 1458, 1458, 1998, 1998, 2028, 2028, 2063, 2093, 10221, 10221, 10813, 11055, \cdots]$$

We adjoin only the first thirty five smallest conditions (retaining larger conditions appears to cause normalisation issues; inordinate time/resource requirements) to the original system for normalisation and all generated polynomial conditions are not substituted in but retained in a separate list. Note the assumptions made by *STeM* are listed together with details of the polynomial list.

```
>clearall();hqd:=stemnormal([op(hqc),op([seq(des[i],i=1..35)])],switch=[algebraic,factor]):
```

Obtain algebraic equation with length = 10611
and containing {E13, E23, P1, P2, P3, P4, j, mu, omega, p, q1, q3, r3, theta, u1, u2, u3, z1, z2, z3}
Substitute this algebraic equation into the system? (y,n) N
Obtain algebraic equation with length = 8572
and containing {E13, P1, P2, P3, P4, j, mu, omega, p, r3, theta, u1, u2, u3, z1, z2, z3}
Substitute this algebraic equation into the system? (y,n) N
Obtain algebraic equation with length = 10611
and containing {E13, E23, P1, P2, P3, P4, j, mu, omega, p, q3, r2, r3, theta, u1, u2, u3, z1, z2, z3}
Substitute this algebraic equation into the system? (y,n) N
Obtain algebraic equation with length = 8572

•
•
•

```
>showAssumption();spola:=[op(getUnusedEqn())]:
```

Non-zero denominators

$$P1 \neq 0$$
$$\omega \neq 0$$
$$3\,P1 + 1 \neq 0$$

```
>specs(spola);
```

indets = {E11, E12, E13, E22, E23, P1, P2, P3, P4, j, μ, ω, p, q1, q3, r2, r3, θ, u1, u2, u3, z1, z2, z3}
length = 151223
nops = 12
type = list

```
>hqds:=separate(hqd):
```

Next, we minimise/reduce the complete set of polynomial conditions accumulated so far.

```
>ypola:=minimisePoly([op(getUnusedEqn()),op(allcomcs),op(hqds[1])]):
>maplen(%);
```

$$[24, 29, 29, 41, 41, 1456, 2156, 2156, 2859, 2859, 4247, 4247, 5889, 8572, 8572, 13777, 13777, 16076, 16980, \cdots]$$

```
>specs(hqd);
```

length = 294181
nops = 96
type = list

It is important to constantly reduce the differential system modulo the polynomial one when appropriate, as done below. This can have a beneficial affect of managing its overall size . We would recommend that in general one should keep the differential system separate from the polynomial one

```
>hqda:=simplifySys(hqd,ypola):
>specs(%);
```

length = 276200
nops = 91
type = list

```
>hqca:=simplifySys(hqc,ypola):
>specs(%);
```

length = 36924
nops = 71
type = list

It is now time to switch to the basic variables, as given by VdBS, together with the scaled operators, followed by a complete reduction/normalisation of the combined differential system. It turns out that replacing p by $x \equiv p + \mu$ is indeed very useful as there is a remarkable reduction in overall system size.

```
>l:=lambda:W:=omega:with(student):
>eqbasics:=separate(clean(eval(sortl([op(eqns(87,135))])))):
>solvar:=fsubs(p=x-mu,sortl(solve(eqbasics[1],{G,W,n,q1,q2,q3,r1,r2,r3,E13,E23,E12,E11,E22,z1,z2,z3,j})));
>eqns(150,153):assign(%):
>eqns(146,148):assign(%):
>basicvars:=[D0,D1,D2,D3,R,b1,b2,b3,R1,R2,R3,J,C0,C3,C12,C13,C23]:
>for i from 1 to 17 do e0(basicvars[i]):=0:od:
>p:=x-mu:
>e0(x) := -theta*x*(P1+1):e1(x):= -U1*lambda*x*(P1+1):
>e2(x) := -U2*lambda*x*(P1+1):e3(x) := -U3*lambda*x*(P1+1):
>solvar[8]:assign(%):
>hqe:=stemnormal([op(eqbasics[2]),op(hqca),op(solvar)],switch=[algebraic,factor]):
```

factor , 1, : , P1
factor, 2,:, U3λx+e$_3$(μ)
Use factor number?(ter=terminate) 2
factor, 1, :,ω
factor, 2, :, -6U1λ^2P1^{2}U3-6P2e$_3$(μ)U1λ+6nU2λP1-3U2λP1n33+6U3λP1q1-6e$_3$(U1)λP1+2u3λU1+6E13
Use factor number?(ter=terminate) 2
Obtain algebraic equation with length = 2144
and containing {C13, C23, D0, D1, P1, P2, P3, R2, R3, U1, U2,
U3, X1, X7, b1, b2, b3, lambda, n33, theta, x}
Substitute this algebraic equation into the system? (y,n) n
Obtain algebraic equation with length = 3337
and containing {C0, C12, C3, D0, D3, J, P1, P2, P3, R2, R3, U1,
U2, U3, X3, X5, b1, b2, b3, lambda, mu, n33, theta, x}
Substitute this algebraic equation into the system? (y,n) n
•
•
•
Obtain algebraic equation with length = 1905
and containing {C23, D0, D3, M23, P1, P2, P3, R2, R3, U1, U2,
U3, X8, b1, b2, b3, lambda, n33, theta, x}
Substitute this algebraic equation into the system? (y,n) n
"Obtain algebraic equation"
(2*D0*n33+b3*lambda)*(8*D0*R*lambda-6*D0*n33+3*b3*lambda)
Substitute this algebraic equation into the system? (y,n) y
factor , 1, : , 2D0n33 + b3λ
factor , 2, : , 8D0Rλ - 6D0n33 + 3b3λ
Use factor number?(ter=terminate) 1
•
•
•

```
>showAssumption();
```

Non-zero factors
P1 $\neq$ 0
8 D0Rλ - 6D0n33 + 3b3λ $\neq$ 0
Non-zero denominators
x $\neq$ 0
D0 $\neq$ 0

$$P1 \neq 0$$
$$P2 \neq 0$$
$$U1 \neq 0$$
$$U2 \neq 0$$
$$U3 \neq 0$$
$$\omega \neq 0$$
$$\lambda \neq 0$$
$$-3P1 - 1 \neq 0$$

A few key points need to be made here. The user should take note of the particular choices made on which conditions to substitute back into the normalisation procedure and which are to be kept aside. Note factor choices *must* be made. Intuitively, one can substitute in conditions that are "relatively small" perhaps less than 1K Maple length but the user should be aware that in making each choice it is important to consider the actual structure of the condition, like high degrees in the variables which can adversely affect the automated process.

```
>upol:=[op(getUsedEqn()),op(getUnusedEqn())]:
>specs(hqe);
```

length = 13502
nops = 72
type = list

```
>for i from 1 to 16 do e0(X||i):=0:od:
```

Higher order commutators are now again carried out followed by a global assignment of the basic variables together with a subsequent normalisation followed by an application of *findComm* on the resulting system.

```
>halfe:=findHalfComm(hqe):
>maplen(%);
```

[1914, 2110, 2112, 2161]

```
>assign(clean(solvar)):clearall();
>hqf:=stemnormal([op(hqe),op(halfe)],switch=[algebraic,factor]):
>showAssumption();
```

Non-zero denominators
$$x \neq 0$$
$$D0 \neq 0$$
$$\lambda \neq 0$$
$$3\ P1 + 1 \neq 0$$

```
>specs(hqf);
```

length = 19742
nops = 59
type = list

```
>hqfs:=separate(hqf):maplen(%);
```

[51]

```
>hqg:=stemnormal(hqf,switch=[algebraic,factor]):
>maplen(%);
```

[34, 36, 286, 286, 674, 53, 288, 346, 756, 53, 288, 346, 756, 73, 125, 125, 35, 35, 35, 29, 42, 80, 80, 422, 27, $\cdots$]

```
>xcomas:=separate(sortl(map(numer,map(foso,findComm(hqg))))):
>maplen(%);
```

[464, 464, 566, 652, 654, 1420, 1420, 1509, 1562, 1562, 2014, 2094, 2157, 2157, 2410, 2410, 2545, 2545,$\cdots$]

```
>maplen(xcomas[2]);
```

[122, 122, 127, 242, 339, 339, 464, 464, 482, 482, 513, 513]

In the next step, we globally assign the *reva* variables and minimise the complete set of accumulated polynomial conditions together with a normalisation/reduction of the augmented differential system

```
>assign(reva):megapola:=minimisePoly([op(upol),op(xcomas[1]),op(ypola)]):
>hqh:=stemnormal([op(xcomas[2]),op(hqg)],switch=[algebraic,factor]):
```

Before proceeding with yet another application of *findComm,* it is important that the *clearall* command be executed prior to carrying out the automated commutation process. This will then clear the memory tables of previous assumptions, automated commutators, stored polynomials in *getUnusedEqn()*, etc so that only the currently relevant assumptions appear. Of course this may lead to redundant computations. However, it is very important to bear in mind that if a system is altered by injection of new conditions, or subjected to further normalisations/reductions, carrying out the same commutators as done previously can lead to new interesting results!

```
>clearall();ycoms:=separate(findComm(hqh)):
>maplen(%);
```

[490, 490, 595, 682, 684, 1455, 1455, 1547, 1596, 1596, 2057, 2133, 2200, 2200, 2454, 2454, 2584, 2584, 2787, · · ·]

```
>showAssumption();
```

Non-zero factors

$x \neq 0$

$D0 \neq 0$

$P1 \neq 0$

$\lambda \neq 0$

$3P1 + 1 \neq 0$

Non-zero denominators

$x \neq 0$

$P1 \neq 0$

$\lambda \neq 0$

$-3P1 - 1 \neq 0$

```
>polb:=minimisePoly([op(megapola),op(ycoms[1])]):
>halfh:=findHalfComm(hqh):
>halfhs:=separate(sortl(clean([op(halfh)]))):
>maplen(halfhs[2]);
```

[3065, 3104, 3122, 32571, 34330, 35667, 38019, 40205, 41580]

```
>hqi:=stemnormal([op(hqh),op(halfhs[2][1..3])],switch=[algebraic,factor]):
>specs(hqi);clearall():
```

length = 33420

nops = 73

type = list

```
>comus:=separate(findComm(hqi)):
>maplen(%);
```

[490, 490, 595, 682, 684, 1455, 1455, 1547, 1596, 1596, 2057, 2133, 2200, 2200, 2454, 2454, 2584, 2584, · · ·]

```
>showAssumption();
```

Non-zero factors

$x \neq 0$

$D0 \neq 0$

$P1 \neq 0$

$\lambda \neq 0$

$3P1 + 1 \neq 0$

Non-zero denominators

$x \neq 0$

$P1 \neq 0$

$\lambda \neq 0$

$-3P1 - 1 \neq 0$

>polc:=minimisePoly([op(polb),op(comus[1])]):
>maplen(%);

[170, 464, 464, 566, 592, 652, 861, 861, 1420, 1420, 1509, 1748, 1854, 2014, 2094, 2177, 2177, 2410, 2410, 3275, 3275, ···]

>pold:=minimisePoly(remfacs(polc,[seq(l^i,i=1..4),seq(x^i,i=1..4),P1,3*P1+1])):

Next, we apply the symmetry operation *symtran* to *pold* to try and generate any, possibly new, conditions.

>spold:=symtran(pold,{},{}):pole:=minimisePoly([op(pold),op(spold)]):

To *pole*, we now add the time-propagation of the first smallest 18 equations.

>d0pole:=stemreduce(map(e0,pole[1..18]),hqi):
>maplen(%);

[779, 2335, 2335, 5201, 5195, 6545, 7281, 7281, 9858, 9858, 5348, 12139, 11069, 8102, 16441, 18595, 18595, 12107]

>specs(d0pole);

indets = {C0, C12, C13, C23, C3, D0, D1, D3, J, P1, P2, P3, P4, R2, R3, U1, U2, U3, X1, X10, X11, X12, X13, X15, X16, X2, X3, X4, X5, X6, X7, X8, X9, ···}
length = 159085
nops = 18
type = list

>polf:=sortl([op(pole),op(d0pole)]):

The next step consists of a conceptually novel application of the use of resultants. Normally resultants are used at the end stages of a particular mathematical computational scheme. In our case, we use it as a starting point so to speak. We systematically carry out all resultants on pairs of polynomials relative to a designated list of variables. This is an exhaustive measure with the aim of attempting to "separate" the variables which can be particularly useful when one is dealing with a large number of them. This is one of the key elements that has "unlocked" this difficult problem. The issue here is though, that this does indeed produce many hundreds of equations, as can be seen below. Typically, as a consequence, considerably redundancy is generated, so that it is desirable to have routines that conveniently and expeditiously remove "trivial" conditions like simple constant multiples, repeated redundant factors, reducible equations, etc. To assist the user with this task, we have introduced a number of routines suitable for this purpose, like *remredun*, *remfacs*, *indeppol, simplifySys*, etc. Note that *genreslist* computes the resultant using all possible pairs of polynomials with respect to each of the given variables, as long as both polynomials have that variable, of any degree, present. On the other hand *linreslist* only selects pairs of polynomials which are linear in each of the given variables. The starting point and maximum size of each resultant which is to be retained, is controlled by the user.

>resapolf:=sortl(genreslist([C0, C12, C13, C23, C3, D1, D3, J,R, P3, P4, R2, R3, X1, X10, X11, X12, X14, X15, X16, X2, X3, X4, X6, X7, X8, X9, b1, b2, b3],1,10000,5000,polf)):
>resaf:=map(numer,sortl(remredun(clean(resapolf)))):
>resaf:=remredun(remfacs(resaf,[C13,C23,seq(l^i,i=1..8),seq(H^i,i=1..6),seq(x^i,i=1..8),P1^2+2*P2*x+2*P1, seq(P1^i,i=1..6),seq(D0^i,i=1..6),seq((3*P1+1)^i,i=1..6),U3,U1,U2,seq(x^i,i=1..6),P1+1, 9*P1-1,3*P1-1,3*P1^2-3*P2*x-P1,6*P1^2-6*P2*x+P1,U1^2+U2^2+U3^2,3*P1^2+6*P2*x+8*P1])):
>specs(resaf);

indets = {C0, C12, C13, C23, C3, D0, D1, D3, J, P1, P2, P3, R2, R3, U1, U2, U3, X1, X10, X11, X12, X13, X14, X15, X16, X2, X3, X4, X5, X6, X7, X8, X9, b1, b2, b3, lambda, mu, theta, x}
length = 1354762
nops = 434
type = list

>for i from 1 to 434 do resag[i]:=picfac(resaf[i]):od:

$$1 = [C0],\ 2 = [-18D0^2D3P1\lambda^8-6D0^2D3\lambda^8+3C0P1^2U3x^2-3C13P1^2U1x^2+3C23P1^2U2x^2+3P1X11x^2+X11x^2]$$
2

$$1 = [18P1^2U3-9P2U3x-9P1R3-3P1U3-3R3-U3],\ 2 = [\text{length= } 656]$$
2

$$1 = [-36D0P1^2U1+18D0P2U1x+12D0D1P1+6D0P1U1+4D0D1+2D0U1+3P1b2+b2],\ 2 = [\text{length=}852]$$
2

$$1 = [36D0P1^2U2-18D0P2U2x+12D0P1R2-6D0P1U2+4D0R2-2D0U2+3P1b1+b1],\ 2 = [\text{length= } 852]$$
2

$$1 = [15P1U1^2+15P1U2^2+15P1U3^2+7U1^2+7U2^2+U3^2],\ 2 = [\text{length=}1513]$$
2

1 = [P2], 2 = [15*P1*U1^2+15*P1*U2^2+15*P1*U3^2+7*U1^2+7*U2^2+U3^2], 3 = [‘length=‘, 1513]

3

1 = [15*P1*U1^2+15*P1*U2^2+15*P1*U3^2+7*U1^2+7*U2^2+U3^2], 2 = [9*P1^2+18*P2*x+27*P1+8], 3 = [‘length=‘, 1513]

3

1 = [90P1^{3}U3-90P1P2U3x-81P1^{2}R3-33P1^{2}U3-6P2U3x-18P1R3-2P1U3+3R3+U3], 2 = [length= 1513]

2

1 = [P2], 2 = [90*P1^3*U3-90*P1*P2*U3*x-81*P1^2*R3-33*P1^2*U3-6*P2*U3*x-18*P1*R3-2*P1*U3+3*R3+U3], 3 = [‘length=‘, 1513]

3

1 = [9*P1^2+18*P2*x+27*P1+8], 2 = [90*P1^3*U3-90*P1*P2*U3*x-81*P1^2*R3-33*P1^2*U3-6*P2*U3*x-18*P1*R3-2*P1*U3+3*R3+U3], 3 = [‘length=‘, 1513]

3

1 = [-27P1b3λ^5+3b3λ^5+36P1^{2}U3θx-36P2U3θx^2-4P1U3θx], 2 = [length=1771]

2

1 = [-27P1b3λ^5+3b3λ^5+36P1^{2}U3θx-36P2U3θx^2-4P1U3θx], 2 = [length=1771]

2

1 = [27P1b3λ^5-3b3λ^5+24P1^{2}U3θx-24P2U3θx^2-16P1U3θx], 2 = [length=1771]

2

1 = [27P1b3λ^5-3b3λ^5+24P1^{2}U3θx-24P2U3θx^2-16P1U3θx], 2 = [length=1771]

2

1 = [81P1^2-18P2x-6P1-7], 2 = [length=1899]

2

1 = [72P1^2-18P2x-3P1-5], 2 = [length=2521]

2

1 = [72P1^2-18P2x-3P1-5], 2 = [length=2521]

2

1 = [length=740], 2 = [length=1997]

2

1 = [length=707], 2 = [length=2303]

2

1 = [P2], 2 = [length=3706]

2

1 = [P2], 2 = [length=3706]

2

1 = [C0], 2 = [length=4848]

2

1 = [b3], 2 = [length=4848]

2

After removal of some simple factors, which may need to be investigated further, we apply *linreslist* to the system *polf* and remove some redundancies. Hopefully, the next few steps should be self explanatory.

```
>resag:=sortl(convert(resag,list)):
>linresfk:=sortl(linreslist([D0,P2,U1,U2,U3,H,l,x,mu],1,4000,3000,polf)):
>linresf:=sortl(remfacs(linresfk,[U3,seq(l^i,i=1..10),seq(x^i,i=1..12),3*P1^2-3*P2*x-P1,D0,H,3*P1+1,C0,seq(P1^i,i=1..6),
P2,C13,P1^2+2*P2*x+2*P1,C23,U1,U2])):
>specs(linresfk);
```

indets = {C0, C12, C13, C23, C3, D0, D1, D3, P1, P2, P3, R2, R3, U1, U2, U3, X1, X10, X11, X12, X13, X14, X15, X16, X2, X5, X9, b1, b2, b3, lambda, theta, x}

length = 98269

nops = 64

type = list

```
>solU3:=C0*fiso(resag[1],U3);
```

solU3 := C0U3 = (1/3)(18D0^{2}D3P1λ^8+6D0^{2}D3λ^8+3C13P1^{2}U1x^2-3C23P1^{2}U2x^2-3P1X11x^2-X11x^2)/(P1^{2}x^2)

>eq1:=linresf[1]:eq2:=op(1,linresf[4]):
>eq3:=numer(powsubs(solU3,fres(eq1,eq2,H)))/2/x:
>eq4:=factor(powsubs(solU3,fres(eq1,linresf[3],H)/2/x)):
>eq5:=sqrt(fres(numer(oso(solU3)),eq3,x)/(D0^2*D3^2*lambda^16),symbolic):
>konU1:=fiso(solU3,U1):
>eq6:=fsubs(konU1,stemreduce(e0(konU1),hqi)):
>fsubs(konU1,fres(eq2,eq6,H)):eq7:=-fsubs(solU3,%)/2/l^5/(3*P1+2);

$$(P1^2+2P2x+2P1)(6D0^2D3\lambda^8-X11x^2)(324C23D0^3D3P1^2\lambda^8+72C23D0^3D3P1\lambda^8-12C23D0^3D3\lambda^8-54C0C23D0P1^3U3x^2-54C13^2D0P1^3U2x^2-54C23^2D0P1^3U2x^2+6C0C23D0P1^2U3x^2+6C13^2D0P1^2U2x^2+6C23^2D0P1^2U2x^2+9C0C13P1^2b3x^2-9C13^2P1^2b1x^2+9C13C23P1^2b2x^2-54C23D0P1^2X11x^2-12C23D0P1X11x^2+2C23D0X11x^2)$$

>case1:=op(3,eq7);
>conU1:=fiso(factor(powsubs(solU3,case1))/(3*P1^2*C13*x^2),U1);

$$\text{conU1} := U1 = (1/2)(-18C13D0P1U2+2C13D0U2+3C0b3-3C13b1+3C23b2)/(C23D0(9P1-1))$$

>d0conU1:=fsubs(conU1,stemreduce(e0(conU1),hqi))/D0;
>fres(d0conU1,eq2,H):fsubs(conU1,%):powsubs(solU3,%):factor(%):fres(numer(oso(konU1)),%,x):
finala:=sqrt(numer(factor(powsubs(solU3,conU1,%))),symbolic)/(12*(9*P1-1)^2*D0^3)/l^13/(3*P1+1);

$$\text{finala} := D3(18D0^2D3P1^2\lambda^8-60D0^2D3P1\lambda^8-32D0^2D3\lambda^8+9P1^2X11x^2+10P1X11x^2+4X11x^2)(-18C13^2D0P1U2-18C23^2D0P1U2+2C13^2D0U2+2C23^2D0U2+3C0C13b3-3C13^2b1-3C23^2b1)$$

>specsop(%);

$$1$$
$$\text{indets} = \{D3\}$$
$$\text{length} = 2$$
$$\text{nops} = 1$$
$$\text{type} = \text{symbol}$$
$$2$$
$$\text{indets} = \{D0,\ D3,\ P1,\ X11,\ \lambda,\ x\}$$
$$\text{length} = 136$$
$$\text{nops} = 6$$
$$\text{type} = +$$
$$3$$
$$\text{indets} = \{C0,\ C13,\ C23,\ D0,\ P1,\ U2,\ b1,\ b3\}$$
$$\text{length} = 143$$
$$\text{nops} = 7$$
$$\text{type} = +$$

We will consider here only the branch $18D0^2D3P1^2\lambda^8-60D0^2D3P1\lambda^8-32\ D0^2\ D3\lambda^8+9P1^2X11x^2+10P1X11x^2+4X11x^2=0$, $D3\neq 0 \Rightarrow X11 \neq 0$ since we are assuming a non-gamma law equation of state.
>kenD3:=fiso(18*D0^2*D3*P1^2*lambda^8-60*D0^2*D3*P1*lambda^8-32*D0^2*D3*lambda^8+9*P1^2*X11*x^2+10*P1*X11*x^2+4*X11*x^2,D3);

$$\text{kenD3} := D3 = -(1/2)X11x^2(9P1^2+10P1+4)/(D0^2\lambda^8(9P1^2-30P1-16))$$

>stemreduce(e0(kenD3),hqi):kenP2:=fiso(%,P2);

$$\text{kenP2} := P2 = (1/60)(3P1-1)(9P1^2+10P1+4)(9P1^2-30P1-16)/(x(9P1^2+9P1+1))$$

>kenP3:=fiso(fsubs(kenP2,stemreduce(e0(kenP2),hqi)),P3);

$$\text{kenP3} := P3 = (1/3600)(3P1-1)(9P1^2+10P1+4)(9P1^2-30P1-16)(6561P1^6-7290P1^5-39528P1^4-36936P1^3-14832P1^2-3156P1-548)/(x^2(9P1^2+9P1+1)^3)$$

>kenP4:=fiso(fsubs(kenP2,kenP3,stemreduce(e0(kenP3),hqi)),P4);
>fsubs(kenD3,eq2):kenU1:=fiso(%,U1);

$$\text{kenU1} := U1 = (1/2)(-18C13D0P1U2+2C13D0U2+3C0b3-3C13b1+3C23b2)/(C23D0(9P1-1))$$

>numer(fsubs(fiso(solU3,U3),kenP2,kenP3,kenP4,kenD3,kenU1,polf[5]));

$$-4(3P1+2)(3P1-1)(3P1+1)(243P1^5-351P1^4-234P1^3+138P1^2+148P1+64)X11\theta x^2$$

Hence we are done in one of the main branches. Of course, to fully establish that $\dot{\mathbf{u}}\cdot\omega \neq \mathbf{0} \Rightarrow \omega\theta = 0$, there still remains the other two main branches to consider, as well as all the other quite modest special cases. These are not particularly difficult, requiring no more than relatively straightforward computations for each case, though there are quite a few of them. Admittedly, the "intense" application of the many small resultants does lead to a large number of special cases, as expected, but this is the price one pays for having the desirable result of effective variable-separation

Several final points should now be made. Firstly, many of the above steps concern themselves with diagnostics. One should liberally use certain routines like *specs, maplen (with sortl)*, *showAssumption, showCommutator,* etc. so as to carefully monitor the general progression of the overall computation. Secondly, we stress that the generation/reduction processes are highly "non-linear" and hence final (and intermediate) system sizes and structures depend very much on intermediate steps and the order in which they are carried out. Also, the variable ordering made at the beginning is important in the overall scheme. Thirdly, at certain points, the user needs to decide which conditions to substitute back into the normalizing processes and which ones to keep aside. There are no hard-and-fast rules here. Depending on the nature of the problem, individual users must decide, perhaps guided by some experimentation or "intuition", on how large a system they wish to retain. Without exercising care at critical points, the computer may be overwhelmed with intermediate swell problems coupled with unacceptable time/memory requirements.

5 Comments

Typically, as seen in the examples above, in general relativity one usually encounters large complex systems of equations. Hence, to assist the user in the investigation of such systems, *STeM* provides interactive tools to help make important decisions at certain critical points in carrying out the analysis process. We emphasis that frequently, the best approach, which fundamentally relies on the ordering of the variables that affect *STeM*'s reduction routines, is not immediately obvious and therefore some experimentation may be required.

In assembling the different parts of *STeM*, we have been required to input many hundreds of equations from various sources. Clearly, the challenge has been to provide consistent, in the sense of signature, duals and object definitions, systems of equations without typographical error. We have done our best to try and avoid errors by performing very extensive cross-checking but however being that this is the first release of *STeM*, we would advise the users to exercise a certain amount of caution regarding the veracity of all equations. Separate independent checks is advised. We hope that through the widespread use of this package any errors that may be present will be found. In such cases, we would much appreciate receiving notification of these issues so that they may be addressed. In the second part exposition of *STeM*, we will demonstrate by examining further applications, the functionality of the remaining routines that we have not covered here. As a final point, in the near future, if there is sufficient interest, we will develop a version of *STeM* for Mathematica.

6 Acknowledgments

First of all, we would like to thank N.Van den Bergh for providing us with most of the orthonormal equations that are in *STeM*. It is his notation and conventions, except for the tetrad rotation coefficients renaming, that we have implemented in the orthonormal part of our package. We would also thank him for providing the Maple worksheets containing the expressions of the defining basic variables and basic integrability conditions/EFE as in Appendix B of his paper, and for his continued encouragement and suggestions throughout various aspects of this work. For the NP and GHP Killing equations, with resulting integrability conditions,(these were independently verified by *STeM*) we thank J.D. Steele.

7 Appendix: Basic variables

$$\mathfrak{D}_0 = \frac{(p+\mu)\omega}{\lambda^5}, \mathfrak{R} = \frac{n}{\lambda}, \mathfrak{b}_1 = -\frac{4}{3}\frac{p+\mu}{\lambda^6}z_1 - \frac{2(9p'-1)(p+\mu)\omega}{3\lambda^5}\dot{U}_2,$$

$$\mathfrak{b}_2 = -\frac{4}{3}\frac{p+\mu}{\lambda^6}z_2 + \frac{2(9p'-1)(p+\mu)\omega}{3\lambda^5}\dot{U}_1, \mathfrak{b}_3 = -\frac{4}{3}\frac{p+\mu}{\lambda^6}z_3,$$

$$\mathfrak{D}_1 = -\frac{1}{3}\dot{U}_1 + \frac{q_1}{\lambda}, \mathfrak{R}_1 = \frac{1}{3}\dot{U}_1 + \frac{r_1}{\lambda}, \mathfrak{D}_2 = -\frac{1}{3}\dot{U}_2 + \frac{q_2}{\lambda},$$

$$\mathfrak{R}_2 = \frac{1}{3}\dot{U}_2 + \frac{r_2}{\lambda}, \mathfrak{D}_3 + \mathfrak{R}_3 = -\frac{1}{3}\dot{U}_3 + \frac{q_3}{\lambda}, \mathfrak{D}_3 - \mathfrak{R}_3 = \frac{1}{3}\dot{U}_3 + \frac{r_3}{\lambda}$$

$$\mathfrak{J} = (1-2G)\dot{U}^2 + \lambda^{-2}(\theta^2 - 3\mu - 2\frac{j}{p'}) + 9\mathfrak{D}_0^2\frac{\lambda^8}{(p+\mu)^2},$$

$$\mathfrak{C}_{\alpha\beta} = \frac{3p'+1}{\lambda^2 p'}E_{\alpha\beta} + G\dot{U}_\alpha\dot{U}_\beta, \ \ (\alpha,\beta) = (1,2),(1,3),(2,3),$$

$$\mathfrak{C}_0 = \frac{3p'+1}{\lambda^2 p'}(E_{11} - E_{22}) + G(\dot{U}_1^2 - \dot{U}_2^2),$$

$$\mathfrak{C}_3 = \frac{3p'+1}{\lambda^2 p'}E_{33} - \frac{G}{3}(\dot{U}_1^2 + \dot{U}_2^2 - 2\dot{U}_3^2) + \frac{2(9p'+1)\mathfrak{D}_0^2\lambda^8}{3p'(p+\mu)^2},$$

$$G \equiv \frac{1}{3p'}\{3p''(p+\mu) + p' - 3(p')^2\}, \lambda \equiv \exp(\int\frac{d\mu}{3(p+\mu)}).$$

$$\dot{U}_\alpha \equiv \frac{\dot{u}_\alpha}{\lambda p'}.$$

References

1. Banerji S. 1968 Prog. Theor. Phys. **39** 365
2. Carminati J 1987 J. Math. Phys. **28** 1848
3. Carminati, J. and Cyganowski S 1997 Class. Quantum Grav. **14**, 1167
4. Collins C B 1984 J. Math. Phys. **25** 995
5. Collins C B 1986 Can. J. Phys. **64**, 191
6. Coley, A. A. 1991 Class. Quantum Grav. **8** 995
7. Complete details of the calculations for all cases, are available in Maple worksheets from the authors
8. Cyganowski S and Carminati J 2000 Gen. Rel. Grav. **32** 221
9. Czapor S. R. and McLenaghan R. G. 1987 Gen. Rel. Grav. **19** 623
10. Ellis G F R (1971) *Relativistic Cosmology* R Sachs (Ed.), *General Relativity and Cosmology*, Rendiconti S. I. F., XLVIII Corso, Academic Press,. New York.
11. Ellis G F R 1967 J. Math. Phys. **8** 1171
12. Geroch R, Held A and Penrose R 1973 *JMP* **14** 874
13. King A. R. and Ellis G. F. R. 1973 Commun. Math. Phys. **31** 209
14. Lang J M and Collins C B 1988 Gen. Rel. Grav. **20** 683
15. MacCallum M A H 1971 *Cosmological Models from a Geometric Point of View,* in *Cargese Lectures in Physics,* Vol. 6, Lectures at the International Summer School of Physics, Cargese, Corsica, edited by E. Schatzman (Gordon and Breach, New York, 1973)
16. MacCallum, M.A.H. 2018 Computer Algebra in Gravity Research. Living Rev Relativ **21**, 6.
17. Newman E T and Penrose R 1962 *J. Math. Phys.* **3** 556
18. Senovilla J. M. M., Sopuerta C. F. and Szekeres P. 1998 Gen. Rel. Grav. **30** 389
19. Sopuerta C F 1998 Class. Quantum Grav. **15** 1043
20. Steele J D 2012 arXiv:1212.1179v1 [gr-qc] 5 Dec
21. Stephani H, Kramer D, MacCallum M A H, Hoenselaers C and Herlt E 2003 *Exact Solutions of Einstein's Field Equations* 2nd Ed. (Cambridge: Cambridge University Press)
22. The notation used is as in Vu K T and Carminati J 2003 Gen. Rel. Grav. **35**, 263
23. Treciokas R and Ellis G F R 1971 Commun. Math. Phys. **23** 1
24. Van den Bergh, N. 1987 Class Quantum Grav 5:L169–79
25. Van den Bergh N 1999 Class. Quantum Grav. **16** 117
26. Van den Bergh, N and Slobodeanu, R 2016 Class. Quantum Grav. **33**, 085008
27. Van den Bergh, N 2023 arXiv:2309.10478v3 [gr-qc] 12 Oct 2023
28. Vu, K T and Carminati J 2001 Gen. Rel. Grav. **33** 295
29. Vu, K T and Carminati J 2003 Gen. Rel. Grav. **35** 263
30. White A. J. and Collins C. B. 1984 J. Math. Phys. **25** 332