Prepared for submission to JHEP

Fast canonical analysis with artificial intelligence

Will Barker

Central European Institute for Cosmology and Fundamental Physics, Institute of Physics of the Czech Academy of Sciences, Na Slovance 1999/2, 182 00 Prague 8, Czechia

E-mail: barker@fzu.cz

Abstract: Canonical methods facilitate the study of classical field theories, and the quantization of gauge theories. The Dirac–Bergmann constraint algorithm is the gold standard approach to cataloging the propagating degrees of freedom and the gauge symmetries in Hamiltonian field theory. For gravity, however, this procedure is notoriously arduous. Firstly, Dirac–Bergmann is usually expressed in terms of Poisson brackets, whose computation is cumbersome. Secondly, Dirac–Bergmann often calls for a level of imagination which is surprising in an algorithmic procedure: reasoning and creativity are required both in the flow of the algorithm, and in the reconstruction of the final constraint algebroid. In this paper we present computer algebra tools that solve the first problem of computation, as has been possible in principle for some time. We then show that modern large language models have become powerful enough to contend with the second problem. We build a simple agentic system (*Hasdrubal*) which orchestrates Dirac–Bergmann, and which relies on a suite of tools (*Hamilcar*) to compute brackets without mistakes or hallucinations. The *Hamilcar* suite can also be used by humans, and we stress-test it here against pure GR at two loops. The *Hasdrubal* agent is shown to perform well against free field theories. This is significant, becuase *Hasdrubal* is a rudimentary agent by current standards: it relies on in-context learning and commercially available reasoning models without expensive post-training. This exercise is intended to highlight how routine workflows in theoretical physics are becoming vulnerable to disruption by advances in artificial intelligence.

Contents

1 Introduction

The importance of tool-use In the conduct of science it is useful to distinguish between *tools*, which are defined as providing some degree of specialised automation, and things which *use tools*, an ability which implies some degree of agency. Artificial intelligence (AI) is not a tool: much like human researchers it belongs firmly to the second category, and it is perhaps the first of our technologies to enjoy this distinction. Much like human researchers, AI may be *augmented* with tools, resulting in agentic systems, or agents. The agentic principle has served as a powerful companion to the recent empirical scaling-law and architectural breakthroughs that facilitate modern large language models (LLMs). For example, it is currently fashionable to observe that LLMs are not very good at arithmetic. They do,

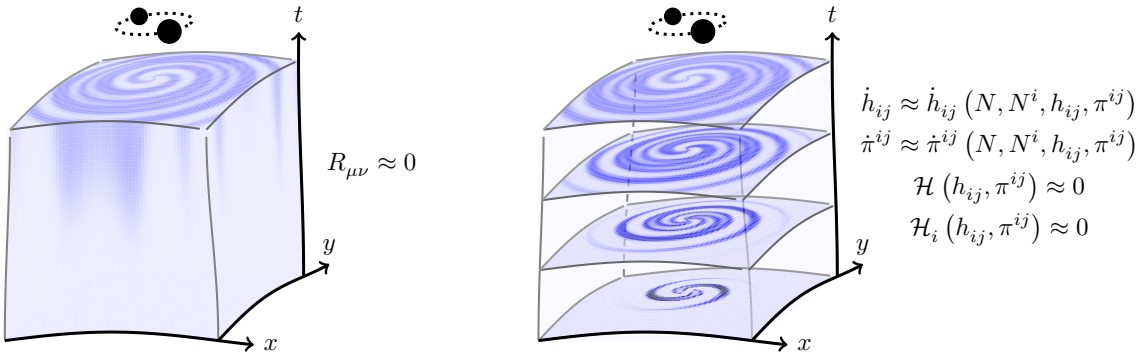


Figure 1. The canonical formulation of a field theory such as GR allows the propagation of initial data to subsequent foliations by means of first-order equations for phase-space variables (h_{ij} and π^{ij}), subject to additional constraints ($\mathcal{H}$ and $\mathcal{H}_i$) and a gauge choice to fix the values of undetermined Lagrange multipliers (N and N^i) on each foliation.

however, show a profound aptitude for abstract reasoning, and it turns out to be quite easy to furnish them with a calculator. Applications of reasoning models to theoretical physics are naturally promising, and have lead to bespoke benchmarks [1] used to assess test-time scaling techniques [2]. Viable agentic systems are currently entering the precision cosmology mainstream [3, 4] and have been proposed for the broader scientific community [5].

Canonical formulation In the study of classical field theories, the canonical formulation is reached by foliating the spacetime into equal-time hypersurfaces, and defining canonical variables (the field components and their conjugate momenta) which evolve between hypersurfaces according to first-order Hamilton equations. In the minimal transition to the canonical formulation, it frequently happens that not all of the velocities may be solved for in terms of canonical variables. In such cases, the definitions of the canonical momenta (as the variations of the action with respect to the velocities) lead to constraints. The minimal canonical formulation can be made consistent by enforcing these constraints with the aid of extra Lagrange multipliers, and further identifying all ancillary constraints which arise by enforcing the conservation (in time, via the Hamilton equations) of the original constraints. This algorithm leads to a counting of $N_{\rm Phy}$ pairs of Cauchy data, i.e., physical degrees of freedom. The total number of constraints is partitioned into $N_{\rm 1st}$ of the first class (commuting with all the rest), and $N_{\rm 2nd}$ of the second class (non-commuting with each other). The formula for the number of physical degrees of freedom is then

$$N_{\rm Phy} = \frac{1}{2}\Big(N_{\rm Can} - 2N_{\rm 1st} - N_{\rm 2nd}\Big)\,, \tag{1.1}$$

where $N_{\rm Can}$ is the original number of canonical variables. In the case of general relativity (GR) shown in fig. 1,[1] the induced metric h_{ij} and its conjugate momentum π^{ij} propagate subject to first-class Hamiltonian and momentum constraints $\mathcal{H}$ and $\mathcal{H}_i$, whilst the lapse N and shift N^i are undetermined multipliers whose values constitute a choice of gauge.

[1]Note that the Hamilton equations shown in the graphic omit the stress-energy sources to which gravity is coupled, and moreover they are *not* immediately suitable for use in numerical relativity.

For more general theories, the algorithm for determining and classifying all constraints is due to Dirac [6, 7] and Bergmann [8], see also [9–12]. This algorithm, and the formula in eq. (1.1), are exceptionally useful in providing a definitive counting of modes when a new classical theory is proposed. Moreover, constraints of the first class following from momentum definitions may be identified as the generators of gauge symmetries, allowing the latter to be reconstructed. In turn, this first-class algebra facilitates the quantisation of the theory [13, 14]. Certain steps in canonical analysis would benefit from automation. At the lower level, these principally include the computation of Poisson brackets between arbitrary functionals of canonical variables, and the re-expression of the results in terms of specified constraints. At a higher level, it would be useful if the multiple steps of the Dirac–Bergmann algorithm itself could be automated.

In this paper We present a set of computer algebra tools (*Hamilcar*) which facilitate canonical analysis. We also construct a rudimentary agent (*Hasdrubal*) that is capable of using these tools. The *Hamilcar* package is built on top of the *xAct* suite of packages for *Mathematica* [15–21], and automates the computation and simplification of Poisson brackets between arbitrary functionals of canonical variables in three spatial dimensions. *Hasdrubal* is a single-agent, tool-augmented LLM operating entirely at inference time. It is constructed using *OpenAI*'s *Agents SDK* platform, combined with *OpenAI*'s *GPT-5.2* (API identifier `gpt-5.2`) model. Whilst *Hamilcar* may plausibly be useful to the physics community as a software package, *Hasdrubal* is only a proof-of-concept that serves the current paper: it is expected that more sophisticated agents will be constructed in future work.

Structure of this paper The remainder of this paper is set out as follows. In section 2 we describe the *Hamilcar* package, and demonstrate its use by performing a full Dirac–Bergmann analysis of pure GR in section 2.2, pure R^2 theory in section 2.3 (see e.g. [22–25]), and finally pure GR at two loops, as described by the Goroff–Sagnotti action, in section 2.4 (see e.g. [26, 27]). In section 3 we use *Hasdrubal* to show that agents may easily be constructed around the *Hamilcar* framework. The agent is described in section 3.1, means for testing it in section 3.2, and some examples of its use in section 3.3. Technical appendices follow.

Notation and conventions Throughout this paper we use natural units in which $\hbar = c = 1$. Greek letters denote four-dimensional spacetime indices, while Roman letters denote three-dimensional spatial indices. The signature is $(-,+,+,+)$, other conventions are introduced as needed.

2 The *Hamilcar* tools

2.1 Overview of the package

2.1.1 Installation

Obtaining the package The *Hamilcar* package should only be installed after the *xAct* suite of packages has been installed. For information about *xAct*, see `xact.es`. The *Hamilcar* package is available at the *GitHub* repository `github.com/wevbarker/Hamilcar`, along with

installation instructions for various operating systems, including *Windows* and *macOS*. Here, we demonstrate a *GNU/Linux* installation.[2] One can use *Bash* to download *Hamilcar* into the home directory as follows:

```
[user@system ~]$ git clone https://github.com/wevbarker/Hamilcar
```

Installing the package To make the installation, the sources should simply be copied alongside the other *xAct* sources. If the installation of *xAct* is global, one can use:

```
[user@system ~]$ cd Hamilcar/xAct
[user@system xAct]$ sudo cp -r Hamilcar /usr/share/Wolfram/Applications/xAct/
```

Or, for a local installation of *xAct*, one may use:

```
[user@system xAct]$ cp -r Hamilcar ~/.Wolfram/Applications/xAct/
```

Syntax highlighting From this point on, we will occasionally present code listings which have syntax highlighting, our conventions for which are as follows. Symbols belonging to the *Wolfram Language* (*Mathematica*) are **BlackBold**, those belonging to *xAct* are **NavyBlueBold**, those belonging to *Hamilcar* are **BlueBold**, and those which will be defined as part of the user session are in **SkyBlueBold**. The start of each new input cell in a *Mathematica* notebook is denoted by '`In[#]:= `', and the start of each output cell is denoted by '`Out[#]=`'. Comments within the code appear in `(*Gray*)`, and strings (which are not symbols) are shown in `"GrayBold"`.

Loading the software The software is loaded via the **Get** command:

```
In[1]:= Get["xAct`Hamilcar`"];
```

2.1.2 Usage

Pre-defined geometry When `In[1]` is initially run, the software defines a three-dimensional spatial hypersurface with the ingredients shown in table 1. In the framework of *xAct*, note that calls to **DefManifold** and **DefMetric** are made internally at this stage. The package establishes a spatial manifold **M3**, creating the necessary geometric structure for canonical field theory calculations.

Wolfram Language	Output format	Meaning
a, **b**, **c**, ..., **z**	a, b, c, ... z	Spatial coordinate indices
G[-**a**,-**b**]	h_{ab}	Induced metric on the spatial hypersurface
CD[-**a**]@	∇_a	Spatial covariant derivative
epsilonG[-**a**,-**b**,-**c**]	ϵ_{abc}	Induced totally antisymmetric tensor

Table 1. Pre-defined geometric objects in *Hamilcar*. The spatial coordinate indices correspond to *adapted* coordinates in the ADM prescription.

[2]The syntax highlighting for *Bash* differs from that used for the *Wolfram Language* in other sections.

Function `DefCanonicalField` The command

```
DefCanonicalField[Fld[]]
```

defines a scalar canonical field `Fld` and its conjugate momentum `ConjugateMomentumFld`. The command

```
DefCanonicalField[Fld[Ind1,Ind2,...]]
```

defines a tensor canonical field `Fld` and its conjugate momentum `ConjugateMomentumFld` with indices `Ind1`, `Ind2`, etc. The command

```
DefCanonicalField[Fld[Ind1,Ind2,...],Symm]
```

defines a tensor canonical field `Fld` and its conjugate momentum `ConjugateMomentumFld` with indices `Ind1`, `Ind2`, etc. and symmetry `Symm`. The syntax of `DefCanonicalField` follows similar patterns to `DefTensor` in *xTensor*. Any number of comma-separated indices may be drawn from the contravariant `a`, `b`, `c`, up to `z`, the covariant `-a`, `-b`, `-c`, up to `-z`, or any admixture. The symmetry `Symm` can be one of the following (or any admixture allowed by `DefTensor`):

- `Symmetric[{SymmInd1,SymmInd2,...}]` denotes symmetrized indices.
- `Antisymmetric[{SymmInd1,SymmInd2,...}]` denotes antisymmetrized indices.

Note that if the global variable `$DynamicalMetric` is set to `True`, then:

- The spatial metric `G` is automatically registered, and `ConjugateMomentumG` is defined.
- All conjugate momenta are automatically defined as tensor densities of weight one (the effects of this are only seen by the `G`-variations performed internally by `PoissonBracket`).

The following options may be given:

- `FieldSymbol` is the symbol that `Fld` will use for display formatting.
- `MomentumSymbol` is the symbol that the conjugate momentum will use for display formatting.

Function `PoissonBracket` The command

```
PoissonBracket[Op1,Op2]
```

computes the Poisson bracket between operators `Op1` and `Op2`. The operators `Op1` and `Op2` must be expressions involving:

- Canonical fields and their conjugate momenta defined using `DefCanonicalField`.
- Tensors which have been defined on the manifold `M3` using `DefTensor`, and which are assumed always to be independent of the canonical fields unless their expansion in terms of canonical fields and more fundamental independent tensors has been registered using `PrependTotalFrom`.

- Derivatives via **CD** of canonical and non-canonical quantities, the spatial metric **G**, and the totally antisymmetric tensor **epsilonG**.
- Constant symbols defined using **DefConstantSymbol** (or **DefNiceConstantSymbol** from the *xTras* package).

The function automatically generates smearing tensors unless **$ManualSmearing** is set to **True**. When **$DynamicalMetric** is set to **True**, the **G**-sector contributions are included. The function computes variational derivatives with respect to all registered fields and momenta.

Function TotalFrom The command

```
TotalFrom[Expr]
```

expands composite expressions to canonical variable form by applying all registered expansion rules. The function converts composite quantities (like constraint expressions, traces, or field combinations) into explicit expressions involving only the fundamental canonical variables: fields, conjugate momenta, and their spatial derivatives. This expansion is essential before computing Poisson brackets, as bracket calculations require expressions to be written in terms of the canonical variables registered by **DefCanonicalField**. The function applies all rules stored in the internal list **$FromRulesTotal**, which are populated using **PrependTotalFrom**.

Function TotalTo The command

```
TotalTo[Expr]
```

converts expressions from canonical variable form back to compact notation using registered contraction rules. The function performs the inverse operation of **TotalFrom**, converting expressions written in terms of canonical variables back to more compact composite notation. This is primarily used for presentation purposes to make final results more readable. The function applies all rules stored in the internal list **$ToRulesTotal**, which are populated using **PrependTotalTo**. Unlike **TotalFrom**, this function is optional in most calculations. Moreover, for more advanced re-expression tasks, the **FindAlgebra** function is preferred.

Function PrependTotalFrom The command

```
PrependTotalFrom[Rle]
```

registers an expansion rule to convert a composite quantity to canonical variable form. The function adds **Rle** to the front of the internal list **$FromRulesTotal** used by **TotalFrom**. Typically this is used with **MakeRule** expressions that define composite quantities in terms of canonical variables, and is essential for setting up the expansion system before performing Poisson bracket calculations.

Function PrependTotalTo The command

```
PrependTotalTo[Rle]
```

registers a contraction rule to convert canonical variable expressions back to compact notation. The function adds `Rle` to the front of the internal list `$ToRulesTotal` used by `TotalTo`. Typically this is used with rules that convert expanded canonical expressions back to composite quantities.

Function `FindAlgebra` The command

```
FindAlgebra[Expr,{{Fctr1,Fctr2,...},...}]
```

seeks to express `Expr` as a sum of any number of terms, where each term corresponds to one of the sub-lists and has factors corresponding to indexed tensors whose heads are `Fctr1`, `Fctr2`, etc., where those tensor heads were defined using `DefCanonicalField` or `DefTensor`. The re-expression is achieved automatically by means of any required number of integrations by parts. The command

```
FindAlgebra[Expr,{{Fctr1,Fctr2,...,{CD,...,Fctr3,...}},...}]
```

additionally admits terms where one or more applications of the spatial covariant derivative `CD` act on any of a select group of factors, here `Fctr3`, etc. The following options may be given:

- `Constraints` is a list of special (and appropriately indexed) tensors which were passed as part of the ansatz, with respect to which the re-expression is expected to be homogeneously linear. The answer will be expressed with these tensors factored out.
- `Verify` is a boolean which, when set to **`True`**, causes the re-expression and `Expr` to be varied internally with respect to any tensors which appear exactly to the first power in all terms, after an application of `TotalFrom`. This usually includes smearing functions, but it may also include some canonical variables. The equality of the variations is checked to ensure that the re-expression is correct. Default is **`False`**.
- `DDIs` is a boolean which, when set to **`True`**, causes dimensionally dependent identities (DDIs) such as the Cayley–Hamilton theorem to be taken into account when performing the re-expression. Default is **`False`**.

2.2 Case study: pure GR

Overview of the theory The first theory we consider is Einstein's general relativity (GR) in the absence of matter, defined by the Einstein–Hilbert action

$$\mathscr{S}[g_{\mu\nu}] = \int \mathrm{d}^4x \, \sqrt{-g} \, \frac{R}{\kappa^2} \,, \tag{2.1}$$

where κ is the coupling constant, $g_{\mu\nu}$ is the spacetime metric, and $R \equiv g^{\mu\nu} R_{\mu\nu}$ is the Ricci scalar, defined in terms of the Riemannian curvature tensor by $R_{\mu\nu} \equiv R^{\rho}{}_{\mu\rho\nu}$ (see appendix A for further index conventions). General covariance requires the presence of the metric determinant $g \equiv \det(g_{\mu\nu})$.

2.2.1 Canonical formulation

Lapse, shift and induced metric The canonical formulation of the theory in eq. (2.23) is to be performed using the Arnowitt–Deser–Misner (ADM) decomposition [28] of the metric

$$g^{00} = -\frac{1}{N^2}\,, \qquad g_{0i} = N_i\,, \qquad g_{ij} = h_{ij}\,. \tag{2.2}$$

The ADM variables include the lapse scalar N, the shift vector N_i, and the induced metric tensor on the equal-time foliations h_{ij}. From eq. (2.2) we may also deduce the inverse metric components

$$g_{00} = -N^2 + N_i N^i\,, \qquad g^{0i} = \frac{N^i}{N^2}\,, \qquad g^{ij} = h^{ij} - \frac{N^i N^j}{N^2}\,, \tag{2.3}$$

where h^{ij} is the inverse metric on equal-time slices, such that $h_{ij}h^{jk} \equiv \delta_i^k$. This inverse induced metric will be used to raise spatial indices, for example $N^i \equiv h^{ij}N_j$. From eq. (2.2) it also follows that the metric determinant has a simple decomposition according to $\sqrt{-g} \equiv N\sqrt{h}$. In order for the four-dimensional metric to remain invertible, it is important that the lapse does not vanish, i.e., $N \neq 0$. The induced spatial metric `G[-a,-b]` and its conjugate momentum `ConjugateMomentumG[a,b]` are already pre-defined once `In[1]` has been executed; the latter is denoted by π^{ij}. It will be useful, however, to additionally define the trace $\pi \equiv h_{ij}\pi^{ij}$ of the conjugate momentum using the standard `DefTensor` command from *xAct*:

```
In[2]:= DefTensor[TraceConjugateMomentumG[], M3, PrintAs -> "\[Pi]"];
↪ FromTraceConjugateMomentumG = MakeRule[{TraceConjugateMomentumG[],
↪ Scalar[ConjugateMomentumG[a, -a]]}, MetricOn -> All, ContractMetrics ->
↪ True]
```

At the point of definition, *Hamilcar* is not aware that this trace has any special dependence on the canonical variables. We therefore need to teach *Hamilcar* how to expand this trace in terms of the canonical variables. The programmatic rule `FromTraceConjugateMomentumG`, which performs this expansion, is defined above using the `MakeRule` command from *xAct*; this rule is then registered with *Hamilcar* using the `PrependTotalFrom` command:

```
In[3]:= PrependTotalFrom[FromTraceConjugateMomentumG]
```

Having executed `In[3]`, it will be possible to feed expressions containing the trace π to other *Hamilcar* functions, which will expand it correctly in terms of the canonical variables. Next, the lapse function N is defined using:

```
In[4]:= DefTensor[Lapse[], M3, PrintAs -> "\[ScriptCapitalN]"]
```

The shift vector N^i is defined using:

```
In[5]:= DefTensor[Shift[m], M3, PrintAs -> "\[ScriptCapitalN]"]
```

Note that, once again, we use `DefTensor` from *xAct* to perform the definitions. Properly, these quantities are themselves canonical fields for which `DefCanonicalField` from *Hamilcar* should be used. Since, however, the lapse and shift are non-dynamical Lagrange multipliers

in the case of GR, we do not need to compute their Poisson brackets, and so the simpler `DefTensor` command suffices. Finally, the gravitational coupling constant κ from eq. (2.1) is defined using:

```
In[6]:= DefConstantSymbol[GravitationalCoupling, PrintAs -> "\[Kappa]"]
```

Note that the standard command `DefConstantSymbol` from *xAct* is used.

Extrinsic curvature Apart from the induced spatial metric h_{ij}, we also need to set up some helpful notation for the time derivative $\dot{h}_{ij} \equiv \partial_t h_{ij}$. The first time derivative (velocity) is encoded by the extrinsic curvature tensor

$$K_{ij} \equiv -\frac{1}{2N}\left(\dot{h}_{ij} - \nabla_i N_j - \nabla_j N_i\right), \tag{2.4}$$

where ∇_i is the induced three-dimensional covariant derivative, which satisfies the metricity condition with respect to h_{ij}. For the purpose of dealing with higher-derivative theories in section 2.3, we also need a convenient variable to encode the second time derivative of the metric, i.e., the first time derivative of the extrinsic curvature $\dot{K}_{ij} \equiv \partial_t K_{ij}$. This acceleration is provided by (see a similar construction in [22])

$$F_{ij} \equiv -\frac{1}{N}\dot{K}_{ij} - K_{ik}K^k_j + \frac{N^k}{N}\nabla_k K_{ij} + \frac{2}{N}K_{k(i}\nabla_{j)}N^k - \frac{1}{N}\nabla_i\nabla_j N\,, \tag{2.5}$$

with the trace $F \equiv h^{ij}F_{ij}$. After some standard manipulations (see appendix A and [29, 30]) the ADM decomposition of the Ricci scalar is found to be

$$R \equiv 2F + K^2 - K^{ij}K_{ij} + \mathcal{R}\,, \tag{2.6}$$

where $K \equiv K^i_i$, and $\mathcal{R}$ is the three-dimensional Ricci scalar associated with ∇_i.[3]

Extended action By substituting the expansion in eq. (2.6) into eq. (2.1) we obtain

$$\mathscr{S}[N, N^i, h_{ij}] = \frac{1}{\kappa^2}\int \mathrm{d}^4x\, N\sqrt{h}\left(2F + K^2 - K^{ij}K_{ij} + \mathcal{R}\right). \tag{2.7}$$

By examining eq. (2.5) we see that the action in eq. (2.7) makes explicit reference to the first time derivative of the extrinsic curvature, $\dot{K}_{ij}$ and hence the second time derivative of the spatial metric h_{ij}, which is not desirable for the canonical approach. This term is, however, amenable to integration by parts, using the manipulation

$$\begin{aligned}\mathscr{S}[N, N^i, h_{ij}] &\supset -\frac{2}{\kappa^2}\int \mathrm{d}^4x\,\sqrt{h}\,h^{ij}\dot{K}_{ij} \\ &= \frac{1}{\kappa^2}\int \mathrm{d}^4x\sqrt{h}\,\left(Kh^{ij} - 2K^{ij}\right)\left(2\nabla_{(i}N_{j)} - 2NK_{ij}\right),\end{aligned} \tag{2.8}$$

where we use eq. (2.4) in the second line. When eq. (2.8) is substituted back into eq. (2.7), the resulting action contains only first time derivatives, and simplifies to

$$\mathscr{S}[N, N^i, h_{ij}] = \frac{1}{\kappa^2}\int \mathrm{d}^4x\, N\sqrt{h}\,\left(\mathcal{R} - K^2 + K^{ij}K_{ij}\right). \tag{2.9}$$

[3] Apart from the change from Greek to Roman indices, our conventions for the curvature defined in three and four dimensions are identical.

To proceed towards the canonical formulation, we introduce an auxiliary variable $\mathcal{K}_{ij}$ which is forced on-shell to equal the extrinsic curvature $\mathcal{K}_{ij} \approx K_{ij}$ by means of a Lagrange multiplier π^{ij} — we know that this multiplier is precisely the momentum conjugate to h_{ij}, but this is an identification that we will allow to arise naturally. The canonical action is then

$$\mathscr{S}\left[N, N^i, h_{ij}, \mathcal{K}_{ij}, \pi^{ij}\right] = \int \mathrm{d}^4x \left[\frac{N\sqrt{h}}{\kappa^2}\left(\mathcal{R} - \mathcal{K}^2 + \mathcal{K}^{ij}\mathcal{K}_{ij}\right) + \pi^{ij}\left(\dot{h}_{ij} - 2\nabla_{(i}N_{j)} + 2N\mathcal{K}_{ij}\right)\right]. \tag{2.10}$$

A further application of integration by parts, this time on the foliation, allows eq. (2.10) to be separated into a symplectic part, and two further terms

$$\mathscr{S}\left[N, N^i, h_{ij}, \mathcal{K}_{ij}, \pi^{ij}\right] = \int \mathrm{d}^4x \left[\pi^{ij}\dot{h}_{ij} + 2N^i\nabla_j\pi^j_i + 2N\left(2\pi^{ij}\mathcal{K}_{ij} + \frac{\sqrt{h}}{\kappa^2}\left(\mathcal{R} - \mathcal{K}^2 + \mathcal{K}^{ij}\mathcal{K}_{ij}\right)\right)\right]. \tag{2.11}$$

These terms reveal N and N_i to be themselves acting as Lagrange multipliers.

Canonical action It is moreover possible in eq. (2.11) for π^{ij} to give up its role as a Lagrange multiplier, since $\mathcal{K}_{ij}$ appears only algebraically and may be integrated out as $\mathcal{K}_{ij} \approx \kappa^2\left(\pi h_{ij} - 2\pi_{ij}\right)/2\sqrt{h}$. By substituting this solution back into eq. (2.11), one obtains the standard canonical action of GR

$$\mathscr{S}\left[N, N^i, h_{ij}, \pi^{ij}\right] = \int \mathrm{d}^4x \left[\pi^{ij}\dot{h}_{ij} - N^i\mathcal{H}_i - N\mathcal{H}\right], \tag{2.12}$$

where the Hamiltonian and momentum constraints are defined as

$$\mathcal{H} \equiv \frac{\kappa^2}{\sqrt{h}}\left(2\pi^{ij}\pi_{ij} - \pi^2 - \frac{h\mathcal{R}}{\kappa^4}\right), \qquad \mathcal{H}_i \equiv -2\nabla_j\pi^j_i\,. \tag{2.13}$$

Referring to eq. (2.13), we proceed by defining $\mathcal{H}$ as the variable `SuperHamiltonian[]`, along with a rule `FromSuperHamiltonian` which expands it in terms of the canonical variables:

```
In[7]:= DefTensor[SuperHamiltonian[], M3, PrintAs -> "\[ScriptCapitalH]"];
↪ FromSuperHamiltonian = MakeRule[{SuperHamiltonian[],
↪ (1/((1/GravitationalCoupling^2) * Sqrt[DetG[]])) * (ConjugateMomentumG[i,
↪ j] * ConjugateMomentumG[-i, -j] - (1/2) * TraceConjugateMomentumG[]^2) -
↪ (1/GravitationalCoupling^2) * Sqrt[DetG[]] * RicciScalarCD[]}, MetricOn
↪ -> All, ContractMetrics -> True]
```

The syntax is again based on *xAct*'s `DefTensor` command, and we use `MakeRule` to define the expansion rule. Next, we define the momentum constraint $\mathcal{H}_i$ as the variable `SuperMomentum[-i]`, along with a rule `FromSuperMomentum` which expands it in terms of the canonical variables:

```
In[8]:= DefTensor[SuperMomentum[-i], M3, PrintAs -> "\[ScriptCapitalH]"];
↪ FromSuperMomentum = MakeRule[{SuperMomentum[-i], -2 *
↪ CD[-j][ConjugateMomentumG[j, -i]]}, MetricOn -> All, ContractMetrics ->
↪ True]
```

At the moment, both `FromSuperHamiltonian` and `FromSuperMomentum` are simply variables in the user session, so as with the trace of the conjugate momentum in `In[3]`, we need to register these rules with *Hamilcar*. For the super-Hamiltonian we use:

```
In[9]:= PrependTotalFrom[FromSuperHamiltonian]
```

For the super-momentum we use:

```
In[10]:= PrependTotalFrom[FromSuperMomentum]
```

With these commands, the kernel is now in a state where we can compute Poisson brackets of the constraints, and so implement the Dirac–Bergmann algorithm.

2.2.2 Dirac–Bergmann algorithm

Total Hamiltonian The variation of eq. (2.12) with respect to N and N^i reveals the constrained nature of the super-Hamiltonian and super-momentum

$$\mathcal{H} \approx 0\,, \quad \mathcal{H}_i \approx 0\,, \tag{2.14}$$

meanwhile the usual Legendre transformation is apparent in the simple format of eq. (2.12), leading directly to the total Hamiltonian

$$\mathscr{H} = \int \mathrm{d}^3x \, \left(N\mathcal{H} + N^i\mathcal{H}_i\right) \,. \tag{2.15}$$

Evidently, eq. (2.15) is a functional obtained by integrating canonical variables over equal-time slices. The Dirac–Bergmann algorithm is concerned with the conservation of the constraints in eq. (2.14) under time evolution generated by $\mathscr{H}$. This time evolution is expressed in terms of Poisson brackets between the constraints and the total Hamiltonian. What follows is seen also in sections 2.3 and 2.4, and is the standard scenario in canonical gravity theories: since $\mathscr{H}$ is itself expressed entirely in terms of the constraints, the Dirac–Bergmann algorithm reduces to computing Poisson brackets among the constraints themselves.

Smearing functions Since Poisson brackets are the central currency, we introduce smearing functions to make them easier to handle. As an example of how smeared quantities are to be denoted, we write

$$\mathcal{H}[f] \equiv \int \mathrm{d}^3x \, f(x)\mathcal{H}(x)\,, \qquad \mathcal{H}^i[f_i] \equiv \int \mathrm{d}^3x \, f_i(x)\mathcal{H}^i(x)\,. \tag{2.16}$$

Whenever smearing functions appear inside Poisson brackets, they are strictly assumed to be independent of the canonical variables *in their presented index configuration*. This means, for example, that when f^i and f_i appear inside two different brackets, we cannot

simply conclude $f^i = h^{ij} f_j$. The '[...]' notation of eq. (2.16) is also used more loosely outside of Poisson brackets to denote smearing by functions which may or may not depend on the canonical variables; for clarity, constraints are factored out of such brackets as far as possible. For the super-Hamiltonian, we define a scalar smearing function:

```
In[11]:= DefTensor[ScalarSmearingS[], M3, PrintAs -> "\[ScriptS]"]
```

A second scalar smearing function allows us to compute brackets between two independently smeared super-Hamiltonian constraints:

```
In[12]:= DefTensor[ScalarSmearingF[], M3, PrintAs -> "\[ScriptF]"]
```

For the super-momentum, we firstly define a covariant vector smearing function:

```
In[13]:= DefTensor[VectorSmearingCovariantS[-i], M3, PrintAs -> "\[ScriptS]"]
```

We then add a second independent covariant smearing function:

```
In[14]:= DefTensor[VectorSmearingCovariantF[-i], M3, PrintAs -> "\[ScriptF]"]
```

Similarly, we define a contravariant smearing function:

```
In[15]:= DefTensor[VectorSmearingContravariantS[i], M3, PrintAs ->
 ↪ "\[ScriptS]"]
```

And a second independent contravariant smearing function:

```
In[16]:= DefTensor[VectorSmearingContravariantF[i], M3, PrintAs ->
 ↪ "\[ScriptF]"]
```

By default, *Hamilcar* automatically smears constraints with internal, single-use variables when computing Poisson brackets. For more control over the smearing process, however, we can enable manual smearing:

```
In[17]:= $ManualSmearing = True
```

Fundamentally, all the Poisson brackets we compute in this section can be derived from the formal identity

$$\left\{h_{ij}(t,\vec{x}), \pi^{kl}(t,\vec{x}\,')\right\} \equiv \delta^k_{(i}\delta^l_{j)}\delta^3(\vec{x}-\vec{x}\,')\,. \tag{2.17}$$

Only the identity in eq. (2.17) is required because h_{ij} and π^{kl} are the only canonical variables appearing in the symplectic part of the action in eq. (2.12), and in the constraints in eq. (2.13) — extra fundamental commutators will appear in section 2.3. As with the variables `G[-a,-b]` and `ConjugateMomentumG[a,b]`, the bracket in eq. (2.17) is already pre-defined once `In[1]` has been executed, so no further action is required.

Super-Hamiltonian auto-commutator We can now compute the auto-commutator of the super-Hamiltonian constraint. We begin by setting up the smeared expression using two independent scalar smearing functions:

```
In[18]:= Expr = {ScalarSmearingF[] * SuperHamiltonian[], ScalarSmearingS[] *
 ↪ SuperHamiltonian[]}
```

We then compute the Poisson bracket by feeding `In[18]` into `PoissonBracket`:

```
In[19]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ Expr //= TotalTo
```

```
Out[19]=  -2 π_ab s ∇^b∇^a f + 2 π_ab f ∇^b∇^a s
```

The result of `In[19]` is, by default, expressed in terms of the canonical variables and their spatial derivatives, as seen in the output `Out[19]`. However, the goal of reconstructing the constraint algebra requires us to express this bracket in terms of the constraints themselves. The `FindAlgebra` function from *Hamilcar* performs this reconstruction automatically:

```
In[20]:= Expr //= FindAlgebra[#1, {{{SuperMomentum}, {CD, ScalarSmearingF,
 ↪ ScalarSmearingS}}}, Constraints -> {SuperMomentum[i]}, Method -> Solve,
 ↪ Verify -> True] & ;
```

```
Out[20]=  ℋ^i (-s ∇_i f + f ∇_i s)
```

We see that the `FindAlgebra` function requires us to specify in its second argument a schematic ansatz (with indices suppressed) for the final form of the bracket. In practice, such ansätze are usually easy to guess on dimensional grounds, although the specific index configurations can be tedious to enumerate in detail. In this case, we know that the final answer should be proportional to $\mathcal{H}^i$, and each of the smearing functions f and s, with one factor of ∇_i acting on either smearing function to provide the necessary spatial derivative. The additional structuring of the ansatz by means of nested lists signals that the ∇_i should be confined to act on the smearing functions only, and not on the constraint itself. The option `Constraints` allows the user to specify the constraints that should be factored out of the final result, which is remarkably helpful for formatting. The output `Out[20]` reflects

$$\big\{\mathcal{H}[f],\mathcal{H}[s]\big\} = \mathcal{H}^i\Big[f\nabla_i s - s\nabla_i f\Big]\,, \tag{2.18}$$

and eq. (2.18) is indeed the expected result.

Super-Hamiltonian and super-momentum Next, we compute the Poisson bracket between the super-Hamiltonian and the super-momentum, using a scalar smearing for the super-Hamiltonian and the contravariant vector smearing for the super-momentum:

```
In[21]:= Expr = {ScalarSmearingF[] * SuperHamiltonian[],
 ↪ VectorSmearingContravariantS[i] * SuperMomentum[-i]}
```

The computation proceeds by feeding `In[21]` into `PoissonBracket` as before:

```
In[22]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ Expr //= TotalTo;
```

```
Out[22]=  (2 κ^2 π^bc f s^a ∇_a π_bc)/√ħ - (κ^2 π^b_b f s^a ∇_a π^c_c)/√ħ + (κ^2 π_bc π^bc f ∇_a s^a)/√ħ - (κ^2 π^b_b π^c_c f ∇_a s^a)/(2 √ħ) -
          (√ħ R[∇] f ∇_a s^a)/κ^2 + (2 √ħ ∇_a s^a ∇_b ∇^b f)/κ^2 + (2 √ħ R[∇]_ab f ∇^b s^a)/κ^2 - (√ħ ∇_a ∇_b f ∇^b s^a)/κ^2 - (√ħ ∇_b ∇_a f ∇^b s^a)/κ^2
```

We use `FindAlgebra` to re-express the result in `Out[22]` cleanly in terms of the constraints:

```
In[23]:= Expr //= FindAlgebra[#1, {{{SuperHamiltonian}, {CD, ScalarSmearingF,
 ↪ VectorSmearingContravariantS}}}, Constraints -> {SuperHamiltonian[]},
 ↪ Method -> Solve, Verify -> True] & ;
```

Out[23]= $-\mathcal{H}\, s^a\, \nabla_a f$

Once again, `Out[23]` reflects

$$\{\mathcal{H}[f], \mathcal{H}_i[s^i]\} = \mathcal{H}\Big[-s^i \nabla_i f\Big], \tag{2.19}$$

and eq. (2.19) is again the expected result.

Super-momentum auto-commutator The final bracket to compute is the auto-commutator of the super-momentum constraint. It is helpful to compute this using both contravariant or covariant vector smearing functions, and to compare the results. We begin with the contravariant case, which requires the second contravariant vector smearing function:

```
In[24]:= Expr = {VectorSmearingContravariantF[i] * SuperMomentum[-i],
 ↪ VectorSmearingContravariantS[j] * SuperMomentum[-j]}
```

The computation proceeds by feeding `In[24]` into `PoissonBracket` as before:

```
In[25]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ Expr //= TotalTo;
```

Out[25]= $2\,\pi_{bc}\,\nabla_a s^c\,\nabla^b f^a + 2\,\pi_{ab}\,\nabla^b f^a\,\nabla_c s^c + 2\,s^a\,\nabla_a \pi_{bc}\,\nabla^c f^b - 2\,\pi_{ac}\,\nabla^b f^a\,\nabla^c s_b - 2\,f^a\,\nabla_a \pi_{bc}\,\nabla^c s^b - 2\,\pi_{bc}\,\nabla_a f^a\,\nabla^c s^b$

The result in `Out[25]` is then re-expressed in terms of constraints using `FindAlgebra`:

```
In[26]:= Expr //= FindAlgebra[#1, {{{SuperMomentum}, {CD,
 ↪ VectorSmearingContravariantF, VectorSmearingContravariantS}}},
 ↪ Constraints -> {SuperMomentum[i]}, Method -> Solve, Verify -> True] & ;
```

Out[26]= $\mathcal{H}^i\,(-s^a \nabla_a f_i + f^a\,\nabla_a s_i)$

The output `Out[26]` shows the super-momentum auto-commutator is proportional to the super-momentum itself:

$$\{\mathcal{H}_i[f^i], \mathcal{H}_j[s^j]\} = \mathcal{H}_i\Big[f^j \nabla_j s^i - s^j \nabla_j f^i\Big]. \tag{2.20}$$

Alternatively, we can set up essentially the same bracket using covariant vector smearing functions:

```
In[27]:= Expr = {VectorSmearingCovariantF[-i] * SuperMomentum[i],
 ↪ VectorSmearingCovariantS[-j] * SuperMomentum[j]}
```

We then feed `In[27]` into `PoissonBracket`:

```
In[28]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ Expr //= TotalTo;
```

```
Out[28]= 2 π_bc ∇_a s^c ∇^b f^a - 2 s^a ∇_a f^b ∇_c π_b^c + 2 f^a ∇_a s^b ∇_c π_b^c - 2 s^a ∇^b f_a ∇_c π_b^c + 2 f^a ∇^b s_a ∇_c π_b^c +
          2 π_ab ∇^b f^a ∇_c s^c + 2 s^a ∇_a π_bc ∇^c f^b - 2 π_ac ∇^b f^a ∇^c s_b - 2 f^a ∇_a π_bc ∇^c s^b - 2 π_bc ∇_a f^a ∇^c s^b
```

The result in **Out[28]** is then re-expressed using **FindAlgebra**:

```
In[29]:= Expr //= FindAlgebra[#1, {{{SuperMomentum}, {CD,
 ↪ VectorSmearingCovariantF, VectorSmearingCovariantS}}}, Constraints ->
 ↪ {SuperMomentum[i]}, Method -> Solve, Verify -> True] & ;
Out[29]= H^i (s^a ∇_i f_a - f^a ∇_i s_a)
```

This result is also correct, corresponding to

$$\{\mathcal{H}^i[f_i], \mathcal{H}^j[s_j]\} = \mathcal{H}^i\Big[s^j\nabla_i f_j - f^j\nabla_i s_j\Big]\,. \tag{2.21}$$

Both computations in **Out[26]** and **Out[29]** yield the same fundamental result, as seen also in eqs. (2.20) and (2.21): the super-momentum auto-commutator is proportional to the super-momentum itself. The apparent difference is due to the point mentioned above, that the smearing functions are assumed strictly to be independent of the canonical variables, including the spatial metric. At the level of *xAct*, this is a feature already of how **DefTensor** works: the defined index configuration is recorded as a property of the tensor. Thus, the choice of covariant versus contravariant smearing functions leads to an implicit extra factor of the spatial metric in the arguments of **PoissonBracket**, and interactions with the momentum conjugate to the spatial metric then lead to the different final forms.

Final summary Taken together, the brackets in eqs. (2.18) to (2.20) constitute the Dirac algebra of the constraints in GR. Since each bracket is proportional to the original constraints, it follows that the constraints are all first class. It is not possible, therefore, for further constraints to arise in the Dirac–Bergmann algorithm. The total number of first class constraints in $\mathcal{H}$ and $\mathcal{H}_i$ is $N_{1\text{st}} = 4$, and the total number of second class constraints is $N_{2\text{nd}} = 0$. Given that the number of canonical variables distributed over the independent components of the spatial metric h_{ij} and the conjugate momentum π^{ij} is $N_{\text{Can}} = 12$, the number of physical degrees of freedom is found by referring back to eq. (1.1) to be

$$N_{\text{Phy}} = \frac{1}{2}\Big(12 - 2\times 4\Big) = 2\,, \tag{2.22}$$

which are accounted for by the two polarizations of the massless graviton.

2.3 Case study: pure R^2 theory

Overview of the theory In the space of quadratic gravity theories, the pure R^2 theory is somewhat complementary to eq. (2.1), in that it contains no Einstein–Hilbert term. As shown in [25], this feature prohibits a perturbative treatment on flat spacetime, but the canonical analysis can still be performed non-perturbatively. The action for the pure R^2 theory is

$$\mathscr{S}[g_{\mu\nu}] = \int \mathrm{d}^4x\,\sqrt{-g}\,R^2\,. \tag{2.23}$$

Unlike for the case of eq. (2.1), there is no dimensionful coupling, and so we avoid introducing any coupling whatever in eq. (2.23).

2.3.1 Canonical formulation

Extended action Once again, the well-known identity in eq. (2.6) allows eq. (2.23) to be expanded in terms of ADM variables, as

$$\mathscr{S}[N, N^i, h_{ij}] = \int \mathrm{d}^4x\, N\sqrt{h}\left[2F + K^2 - K^{ij}K_{ij} + \mathcal{R}\right]^2. \tag{2.24}$$

The action in eq. (2.24) is still a higher-derivative action, and the trick in eq. (2.8) cannot be used to change this character. In order to make progress towards the canonical formulation, we again introduce auxiliary fields: this time we need $\mathcal{K}_{ij} \approx K_{ij}$ and $\mathcal{F}_{ij} \approx F_{ij}$, with the on-shell equivalence to the definitions in eqs. (2.4) and (2.5) ensured by Lagrange multipliers. This procedure was already used in moving from eq. (2.9) to eq. (2.10), but this time the introduction of the additional variable $\mathcal{F}_{ij}$ requires an extra Lagrange multiplier that we call ρ^{ij}. Thus, with reference to eqs. (2.4) and (2.5) we write

$$\begin{aligned}\mathscr{S}\left[N, N^i, h_{ij}, \mathcal{K}_{ij}, \mathcal{F}_{ij}, \pi^{ij}, \rho^{ij}\right] = \int \mathrm{d}^4x \Bigg[& N\sqrt{h}\left(2\mathcal{F} + \mathcal{K}^2 - \mathcal{K}^{ij}\mathcal{K}_{ij} + \mathcal{R}\right)^2 \\ & + \pi^{ij}\left(\dot{h}_{ij} - 2\nabla_{(i}N_{j)} + 2N\mathcal{K}_{ij}\right) \\ & + \rho^{ij}\Big(\dot{\mathcal{K}}_{ij} + N\mathcal{K}_{ik}\mathcal{K}^k_j - N^k\nabla_k\mathcal{K}_{ij} \\ & - 2\mathcal{K}_{k(i}\nabla_{j)}N^k + \nabla_i\nabla_j N + N\mathcal{F}_{ij}\Big)\Bigg]. \end{aligned} \tag{2.25}$$

This time, due to the presence of the new multiplier, the field $\mathcal{K}_{ij}$ cannot be integrated out as when moving from eq. (2.11) to eq. (2.12) without producing explicit squares of time derivatives. We thus retain $\mathcal{K}_{ij}$ as an independent canonical field, and identify ρ^{ij} as its conjugate momentum. In *Hamilcar*, we define this canonical pair using **DefCanonicalField**, which automatically defines the conjugate momentum based on the field's index structure and symbolic name:

```
In[30]:= DefCanonicalField[ExtrinsicCurvature[-m, -n], Symmetric[{-m, -n}],
 ↪ FieldSymbol -> "\[ScriptCapitalK]", MomentumSymbol -> "\[Rho]"];
```

The conjugate momentum is `ConjugateMomentumExtrinsicCurvature[a,b]`. We also define the traces of the extrinsic curvature and its conjugate momentum, along with rules for expanding and contracting these traces. First, for the extrinsic curvature:

```
In[31]:= DefTensor[TraceExtrinsicCurvature[], M3, PrintAs ->
 ↪ "\[ScriptCapitalK]"]; FromTraceExtrinsicCurvature =
 ↪ MakeRule[{TraceExtrinsicCurvature[], Scalar[ExtrinsicCurvature[a, -a]]},
 ↪ MetricOn -> All, ContractMetrics -> True];
 ↪ PrependTotalFrom[FromTraceExtrinsicCurvature]; ToTraceExtrinsicCurvature
 ↪ = MakeRule[{ExtrinsicCurvature[a, -a], TraceExtrinsicCurvature[]},
 ↪ MetricOn -> All, ContractMetrics -> True];
 ↪ PrependTotalTo[ToTraceExtrinsicCurvature];
```

Similarly, for its conjugate momentum:

```
In[32]:= DefTensor[TraceConjugateMomentumExtrinsicCurvature[], M3, PrintAs ->
 ↪ "\[Rho]"]; FromTraceConjugateMomentumExtrinsicCurvature =
 ↪ MakeRule[{TraceConjugateMomentumExtrinsicCurvature[],
 ↪ Scalar[ConjugateMomentumExtrinsicCurvature[a, -a]]}, MetricOn -> All,
 ↪ ContractMetrics -> True];
 ↪ PrependTotalFrom[FromTraceConjugateMomentumExtrinsicCurvature];
 ↪ ToTraceConjugateMomentumExtrinsicCurvature =
 ↪ MakeRule[{ConjugateMomentumExtrinsicCurvature[a, -a],
 ↪ TraceConjugateMomentumExtrinsicCurvature[]}, MetricOn -> All,
 ↪ ContractMetrics -> True];
 ↪ PrependTotalTo[ToTraceConjugateMomentumExtrinsicCurvature];
```

These trace definitions will be useful for simplifying expressions involving contracted indices in the subsequent analysis.

Canonical action Whilst $\mathcal{K}_{ij}$ cannot be integrated out, partial success is possible with the second auxiliary field $\mathcal{F}_{ij}$ whose trace $\mathcal{F} \equiv h^{ij}\mathcal{F}_{ij}$ can be determined on-shell according to the solution

$$\mathcal{F} \approx -\frac{1}{2}\Big(\mathcal{K}^2 - \mathcal{K}^{ij}\mathcal{K}_{ij} + \mathcal{R}\Big) - \frac{1}{24}\frac{\rho}{\sqrt{h}}\,. \tag{2.26}$$

When eq. (2.26) is substituted back into eq. (2.25), we notice how the effect of the remaining traceless portion $\mathcal{F}_{ij} - \frac{1}{3}h_{ij}\mathcal{F}$ is to enforce $\Phi^{ij} \approx 0$, where we define the quantity

$$\Phi^{ij} \equiv \rho^{ij} - \frac{\rho}{3}h^{ij}\,. \tag{2.27}$$

Since it vanishes on-shell, we see that Φ^{ij} is a new primary constraint. The corresponding *Hamilcar* definition is:

```
In[33]:= DefTensor[PrimaryConstraint[i, j], M3, Symmetric[{i, j}], PrintAs ->
 ↪ "\[CapitalPhi]"]; AutomaticRules[PrimaryConstraint,
 ↪ MakeRule[{PrimaryConstraint[i, -i], 0}, MetricOn -> All, ContractMetrics
 ↪ -> True]]; FromPrimaryConstraint = MakeRule[{PrimaryConstraint[i, j],
 ↪ ConjugateMomentumExtrinsicCurvature[i, j] - (1/3) * G[i, j] *
 ↪ TraceConjugateMomentumExtrinsicCurvature[]}, MetricOn -> All,
 ↪ ContractMetrics -> True]; PrependTotalFrom[FromPrimaryConstraint];
```

It is useful to define rules that impose the primary constraint shell, setting the traceless part of ρ^{ij} to zero:

```
In[34]:= ToPrimaryShell = MakeRule[{ConjugateMomentumExtrinsicCurvature[i,
 ↪ j], (1/3) * G[i, j] * TraceConjugateMomentumExtrinsicCurvature[]},
 ↪ MetricOn -> All, ContractMetrics -> True];
```

The rule `In[34]` will be used later when computing equations of motion, allowing us to simplify expressions by assuming the constraint is satisfied. We also define an explicit form that retains the primary constraint:

```
In[35]:= ToPrimaryShellExplicit =
 ↪ MakeRule[{ConjugateMomentumExtrinsicCurvature[i, j], (1/3) * G[i, j] *
 ↪ TraceConjugateMomentumExtrinsicCurvature[] + PrimaryConstraint[i, j]},
 ↪ MetricOn -> All, ContractMetrics -> True];
```

The rule `In[35]` is useful when we need to manipulate expressions on the constraint shell while keeping the constraint explicit in the result. Thus, $\mathcal{F}_{ij} - \frac{1}{3}h_{ij}\mathcal{F}$ is acting as a multiplier field, albeit one with an implicit dependence on the metric h_{ij} which ensures its tracelessness at all points on the foliation. To avoid such a dependence, which renders the evaluation of time derivatives tedious, it is convenient to introduce an alternative full-rank Lagrange multiplier field λ_{ij} which directly multiplies Φ^{ij} in the action.[4] Collecting the constraints enforced by λ_{ij}, and the lapse and shift, and so separating out the symplectic structure as we did in eq. (2.11), the canonical action may then be obtained. Contrary to the form given in eq. (2.12), however, we will follow [25] by opting here to collect with respect to N_i rather than N^i, so that the momentum constraint is taken to be contravariant — as discussed already in section 2.2.1 this choice is somewhat non-standard, and will result in complications later on. The canonical action is thus

$$\mathscr{S}\left[N, N_i, \lambda_{ij}, h_{ij}, \pi^{ij}, \mathcal{K}_{ij}, \rho^{ij}\right] = \int \mathrm{d}^4x \left[\pi^{ij}\dot{h}_{ij} + \rho^{ij}\dot{\mathcal{K}}_{ij} - N\left(\mathcal{H} + \lambda_{ij}\Phi^{ij}\right) - N_i\mathcal{H}^i\right], \tag{2.28}$$

where the Hamiltonian and momentum constraints are defined as

$$\mathcal{H} \equiv \frac{1}{144}\frac{\rho^2}{\sqrt{h}} - 2\mathcal{K}_{ij}\pi^{ij} + \frac{1}{6}\left(\mathcal{K}^2 - \mathcal{K}^{ij}\mathcal{K}_{ij} + \mathcal{R}\right)\rho - \mathcal{K}_{ik}\mathcal{K}^k_j\rho^{ij} - \nabla_i\nabla_j\rho^{ij}\,, \tag{2.29a}$$

$$\mathcal{H}^i \equiv -2\nabla_j\pi^{ij} + \rho^{kl}\nabla^i\mathcal{K}_{kl} - 2\nabla^k\left(\mathcal{K}^{il}\rho_{kl}\right). \tag{2.29b}$$

Evidently, the constraints in eqs. (2.29a) and (2.29b) differ from those in eq. (2.13). They can be further simplified by eliminating any dependence on Φ^{ij}, which is constrained, and this is equivalent to redefining the Lagrange multiplier λ_{ij}. Thus, a more compact (and physically equivalent) form for eqs. (2.29a) and (2.29b) is

$$\mathcal{H} \to \frac{1}{144}\frac{\rho^2}{\sqrt{h}} - 2\mathcal{K}_{ij}\pi^{ij} + \frac{1}{6}\left(\mathcal{K}^2 - 3\mathcal{K}^{ij}\mathcal{K}_{ij} + \mathcal{R}\right)\rho - \frac{1}{3}\nabla_i\nabla^i\rho\,, \tag{2.30a}$$

$$\mathcal{H}^i \to -2\nabla_j\pi^{ij} + \frac{1}{3}\rho\nabla^i\mathcal{K} - \frac{2}{3}\nabla_j\left(\mathcal{K}^{ij}\rho\right). \tag{2.30b}$$

The corresponding *Hamilcar* definitions are as follows. The super-Hamiltonian $\mathcal{H}$ is:

```
In[36]:= DefTensor[SuperHamiltonian[], M3, PrintAs -> "\[ScriptCapitalH]"];
 ↪ FromSuperHamiltonian = MakeRule[{SuperHamiltonian[], (1/(144 *
 ↪ Sqrt[DetG[]])) * TraceConjugateMomentumExtrinsicCurvature[]^2 - 2 *
 ↪ ConjugateMomentumG[m, n] * ExtrinsicCurvature[-m, -n] + (1/6) *
 ↪ TraceConjugateMomentumExtrinsicCurvature[] * (TraceExtrinsicCurvature[]^2
```

[4] Note that this step does not alter the physical content of the theory.

```
↪ - 3 * ExtrinsicCurvature[-m, -n] * ExtrinsicCurvature[m, n] +
↪ RicciScalarCD[]) - (Sqrt[DetG[]]/3) *
↪ CD[-m][CD[m][TraceConjugateMomentumExtrinsicCurvature[]/Sqrt[DetG[]]]},
↪ MetricOn -> All, ContractMetrics -> True];
↪ PrependTotalFrom[FromSuperHamiltonian];
```

The super-momentum $\mathcal{H}^i$ is defined as:

```
In[37]:= DefTensor[SuperMomentum[-m], M3, PrintAs -> "\[ScriptCapitalH]"];
↪ FromSuperMomentum = MakeRule[{SuperMomentum[-m], (1/3) *
↪ TraceConjugateMomentumExtrinsicCurvature[] *
↪ CD[-m][TraceExtrinsicCurvature[]] - 2 * Sqrt[DetG[]] * CD[-n][(3 *
↪ ConjugateMomentumG[-m, n] + TraceConjugateMomentumExtrinsicCurvature[] *
↪ ExtrinsicCurvature[-m, n])/(3 * Sqrt[DetG[]])]}, MetricOn -> All,
↪ ContractMetrics -> True]; PrependTotalFrom[FromSuperMomentum];
```

The action in eq. (2.28) with the constraints defined in eqs. (2.27), (2.30a) and (2.30b) is now the canonical formulation of the theory in eq. (2.23).

2.3.2 Dirac–Bergmann algorithm

Total Hamiltonian Varying eq. (2.28) with respect to Lagrange multipliers N, N_i, and λ_{ij} generates a total of ten primary constraints, to be compared with the four primary constraints in eq. (2.14) for GR, namely

$$\mathcal{H} \approx 0\,, \qquad \mathcal{H}^i \approx 0\,, \qquad \Phi^{ij} \approx 0\,. \tag{2.31}$$

By analogy with eq. (2.15), the Legendre transformation is now apparent in the simple format of eq. (2.28), leading directly to the total Hamiltonian

$$\mathscr{H} = \int \mathrm{d}^3x \left[N\left(\mathcal{H} + \lambda_{ij}\Phi^{ij}\right) + N_i \mathcal{H}^i \right], \tag{2.32}$$

and the Dirac–Bergmann algorithm is again concerned with the conservation of the constraints in eq. (2.31) under time evolution generated by eq. (2.32). Once again, the problem reduces to computing Poisson brackets among the various constraints. It will be necessary momentarily, for the purposes of computing the equations of motion, to define the total Hamiltonian density in eq. (2.32):

```
In[38]:= DefTensor[TotalHamiltonian[], M3, PrintAs -> "\!\(\*
↪ SubscriptBox[\(\[ScriptCapitalH]\),\(T\)]\)"]; FromTotalHamiltonian =
↪ MakeRule[{TotalHamiltonian[], Lapse[] * SuperHamiltonian[] + Shift[m] *
↪ SuperMomentum[-m] + Lapse[] * Multiplier[-i, -j] * PrimaryConstraint[i,
↪ j]}, MetricOn -> All, ContractMetrics -> True];
↪ PrependTotalFrom[FromTotalHamiltonian];
```

Note that in setting up `In[38]`, we used `In[7]`, `In[8]` and `In[33]`.

Smearing functions It is necessary to extend the smearing notation in eq. (2.16) to accommodate the new traceless constraint in eq. (2.27). Accordingly, we write

$$\Phi^{ij}[f_{ij}] \equiv \int \mathrm{d}^3x\, f_{ij}(x)\Phi^{ij}(x)\,. \tag{2.33}$$

Based on eq. (2.33), we denote e.g. $f \equiv f_{ij}h^{ij}$ in expressions where the rank-two f_{ij} has already been introduced inside a bracket; elsewhere f denotes a smearing scalar that is completely independent of h_{ij}, but this abuse of notation does not give rise to conflicts. The corresponding *Hamilcar* definitions for tensor smearing functions are:

```
In[39]:= DefTensor[TensorSmearingCovariantS[-i, -j], M3, Symmetric[{-i, -j}],
 ↪ PrintAs -> "\[ScriptS]"]; DefTensor[TensorSmearingContravariantS[i, j],
 ↪ M3, Symmetric[{i, j}], PrintAs -> "\[ScriptS]"];
 ↪ DefTensor[TensorSmearingCovariantF[-i, -j], M3, Symmetric[{-i, -j}],
 ↪ PrintAs -> "\[ScriptF]"]; DefTensor[TensorSmearingContravariantF[i, j],
 ↪ M3, Symmetric[{i, j}], PrintAs -> "\[ScriptF]"];
```

We will need the additional 'fundamental' non-vanishing Poisson bracket

$$\{\mathcal{K}_{ij}(t,\vec{x}), \rho^{kl}(t,\vec{x}')\} \equiv \delta^k_{(i}\delta^l_{j)}\delta^3(\vec{x}-\vec{x}')\,, \tag{2.34}$$

where eq. (2.34) and eq. (2.17) are both encoded in the symplectic part of eq. (2.28). To verify that the fundamental Poisson bracket is as expected, following the definition in `In[30]`, we first set up the bracket between the field and its momentum:

```
In[40]:= Expr = {ExtrinsicCurvature[-m, -n],
 ↪ ConjugateMomentumExtrinsicCurvature[a, b]};
```

We then feed `In[40]` into `PoissonBracket` and simplify:

```
In[41]:= Expr //= (PoissonBracket[#1, #2, Parallel -> False] & ) @@ #1 & ;
 ↪ Expr //= FullSimplify;
Out[41]= 1/2 α^ab (β_ab + β_ba)
```

As expected, the result `Out[41]` confirms the canonical commutation relation between $\mathcal{K}_{ij}$ and its conjugate momentum. Notice that for this basic check we did not take care to manually specify any smearing functions in `In[40]`, and so `PoissonBracket` provides them for us.

Hamilton equations Since they happen to have been derived already in [25], it is worth using *Hamilcar* to verify the equations of motion for the canonical variables in eq. (2.28) explicitly — note that this was not done in section 2.2 for GR. These equations follow immediately from the variations of eq. (2.28) with respect to all canoncial variables, but they can also be obtained by computing their Poisson brackets with the total Hamiltonian in eq. (2.32) according to

$$\begin{aligned} \dot{h}_{ij} &\approx \frac{\delta}{\delta f^{ij}}\{h_{kl}[f^{kl}], \mathscr{H}\}\,, \qquad \dot{\pi}^{ij} \approx \frac{\delta}{\delta f_{ij}}\{\pi^{kl}[f_{kl}], \mathscr{H}\}\,, \\ \dot{\mathcal{K}}_{ij} &\approx \frac{\delta}{\delta f^{ij}}\{\mathcal{K}_{kl}[f^{kl}], \mathscr{H}\}\,, \qquad \dot{\rho}^{ij} \approx \frac{\delta}{\delta f_{ij}}\{\rho^{kl}[f_{kl}], \mathscr{H}\}\,. \end{aligned} \tag{2.35}$$

The format in eq. (2.35) allows us to use *Hamilcar* to compute the equations of motion straightforwardly. We set up the Poisson bracket as follows:

```
In[42]:= Expr = {TensorSmearingContravariantF[i, j] * G[-i, -j],
 ↪ TotalHamiltonian[]};
```

Now we feed `In[42]` into `PoissonBracket`:

```
In[43]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ Expr //= TotalTo; Expr //= VarD[TensorSmearingContravariantF[i, j], CD];
 ↪ Expr //= TotalFrom; Expr //= TotalTo;
```

```
Out[43]= -2 K_ij N + ∇_i N_j + ∇_j N_i
```

The velocity of the induced metric is found from `Out[43]` to be

$$\dot{h}_{ij} \approx -2N\mathcal{K}_{ij} + 2\nabla_{(i}N_{j)} \,, \tag{2.36}$$

which simply confirms eq. (2.4) under the on-shell condition $\mathcal{K}_{ij} \approx K_{ij}$. Next, the velocity of the conjugate momentum to the induced metric is:

```
In[44]:= Expr = {TensorSmearingCovariantF[-i, -j] * ConjugateMomentumG[i, j],
 ↪ TotalHamiltonian[]};
```

Now we feed `In[44]` into `PoissonBracket`. We use `In[34]` to simplify the result by imposing the primary constraint shell:

```
In[45]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ Expr //= TotalTo; Expr //= VarD[TensorSmearingCovariantF[-i, -j], CD];
 ↪ Expr //= TotalFrom; Expr //= #1 /. ToPrimaryShell & ; Expr //= TotalTo;
```

```
Out[45]= -K^ia K^j_a N ρ + 1/6 K_ab K^ab h^ij N ρ + 1/9 h^ij N λ^a_a ρ - 1/3 N λ^ij ρ + 1/6 N R[∇]^ij ρ - 1/18 h^ij N R[∇] ρ - h^ij N ρ^2 / (864 √h) +
  1/3 K^ij N ρ K - 1/18 h^ij N ρ K^2 - N^j ∇_a π^ia + N^a ∇_a π^ij - N^i ∇_a π^ja - 1/3 N^j ρ ∇_a K^ia + 1/3 N^a ρ ∇_a K^ij -
  1/3 N^i ρ ∇_a K^ja + π^ij ∇_a N^a - 1/9 h^ij N^a ρ ∇_a K + 1/9 h^ij ρ ∇_a ∇^a N + 1/6 h^ij N ∇_a ∇^a ρ + 1/6 h^ij ∇_a ρ ∇^a N -
  π^j_a ∇^a N^i - π^i_a ∇^a N^j - 1/3 K^j_a N^i ∇^a ρ - 1/3 K^i_a N^j ∇^a ρ - 2/9 K_ab h^ij ρ ∇^b N^a + 1/3 K^j_a ρ ∇^i N^a +
  1/6 N^j ρ ∇^i K - 1/12 ρ ∇^i ∇^j N - 1/12 N ∇^i ∇^j ρ + 1/3 K^i_a ρ ∇^j N^a + 1/6 N^i ρ ∇^j K - 1/12 ρ ∇^j ∇^i N - 1/12 N ∇^j ∇^i ρ
```

Thus `Out[45]` allows us to conclude

$$\begin{aligned}
\dot{\pi}^{ij} \approx\; & -\frac{N}{864}\frac{\rho^2}{\sqrt{h}}h^{ij} - N\rho\Big(h^{ik}h^{jl} - \frac{1}{6}h^{ij}h^{kl}\Big)\Big(\mathcal{K}_{km}\mathcal{K}^m_l - \frac{\mathcal{K}}{3}\mathcal{K}_{kl}\Big) + \frac{N\rho}{6}\Big(\mathcal{R}^{ij} - \frac{\mathcal{R}}{3}h^{ij}\Big) \\
& - \frac{N}{6}\Big(\nabla^{(i}\nabla^{j)} - h^{ij}\nabla_k\nabla^k - h^{ij}(\nabla_k N)\nabla^k\Big)\rho - \frac{\rho}{6}\Big(\nabla^i\nabla^j - \frac{2}{3}h^{ij}\nabla^k\nabla_k\Big)N \\
& + \nabla_k\Big(N^k\pi^{ij} - 2N^{(i}\pi^{j)k}\Big) + \frac{\rho}{3}\Big(N^k\nabla_k\mathcal{K}^{ij} + 2\mathcal{K}^{k(i}\nabla^{j)}N_k - \frac{2}{3}h^{ij}\mathcal{K}^{kl}\nabla_k N_l\Big) \\
& - \frac{2}{3}N^{(i}\nabla_k\Big(\mathcal{K}^{j)k}\rho\Big) + \frac{\rho}{3}\Big(N^{(i}\nabla^{j)} - \frac{1}{3}h^{ij}N^k\nabla_k\Big)\mathcal{K} - \frac{N\rho}{3}\Big(\lambda^{ij} - \frac{\lambda}{3}h^{ij}\Big) \,.
\end{aligned} \tag{2.37}$$

For the auxiliary extrinsic curvature:

```
In[46]:= Expr = {TensorSmearingContravariantF[i, j] * ExtrinsicCurvature[-i,
 ↪ -j], TotalHamiltonian[]};
```

We feed `In[46]` into `PoissonBracket`:

```
In[47]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ Expr //= TotalTo; Expr //= VarD[TensorSmearingContravariantF[i, j], CD];
 ↪ Expr //= TotalFrom; Expr //= #1 /. ToPrimaryShell & ; Expr //= TotalTo;
```

$$\text{Out[47]}= -\frac{1}{2}\mathcal{K}_{ab}\,\mathcal{K}^{ab}\,h_{ij}\,\mathcal{N} - \frac{1}{3}\,h_{ij}\,\mathcal{N}\,\lambda^{a}{}_{a} + \mathcal{N}\,\lambda_{ij} + \frac{1}{6}\,h_{ij}\,\mathcal{N}\,\mathrm{R}[\nabla] + \frac{h_{ij}\,\mathcal{N}\,\rho}{72\,\sqrt{h}} + \frac{1}{6}\,h_{ij}\,\mathcal{N}\,\mathcal{K}^{2} + \frac{1}{3}\,h_{ij}\,\mathcal{N}^{a}\,\nabla_{a}\mathcal{K} - \frac{1}{3}\,h_{ij}\,\nabla_{a}\nabla^{a}\mathcal{N} + \frac{2}{3}\,\mathcal{K}_{ab}\,h_{ij}\,\nabla^{b}\mathcal{N}^{a}$$

The velocity in `Out[47]` is thus found to be

$$\dot{\mathcal{K}}_{ij} \approx \frac{h_{ij}}{3}\left[\frac{N}{2}\left(\frac{1}{12}\frac{\rho}{\sqrt{h}} + \mathcal{K}^{2} - 3\mathcal{K}^{kl}\mathcal{K}_{kl} + \mathcal{R}\right) - \nabla_{k}\nabla^{k}N + N_{k}\,\nabla^{k}\mathcal{K} + 2\mathcal{K}_{kl}\nabla^{k}N^{l}\right] + N\left(\lambda_{ij} - \frac{\lambda}{3}h_{ij}\right). \tag{2.38}$$

Finally, for the conjugate momentum to the auxiliary extrinsic curvature:

```
In[48]:= Expr = {TensorSmearingCovariantF[-i, -j] *
 ↪ ConjugateMomentumExtrinsicCurvature[i, j], TotalHamiltonian[]};
```

We feed `In[48]` into `PoissonBracket`:

```
In[49]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ Expr //= TotalTo; Expr //= VarD[TensorSmearingCovariantF[-i, -j], CD];
 ↪ Expr //= TotalFrom; Expr //= #1 /. ToPrimaryShell & ; Expr //= TotalTo;
```

$$\text{Out[49]}= 2\,\pi^{ij}\,\mathcal{N} + \mathcal{K}^{ij}\,\mathcal{N}\,\rho - \frac{1}{3}\,h^{ij}\,\mathcal{N}\,\rho\,\mathcal{K} + \frac{1}{3}\,h^{ij}\,\rho\,\nabla_{a}\mathcal{N}^{a} + \frac{1}{3}\,h^{ij}\,\mathcal{N}^{a}\,\nabla_{a}\rho - \frac{1}{3}\,\rho\,\nabla^{i}\mathcal{N}^{j} - \frac{1}{3}\,\rho\,\nabla^{j}\mathcal{N}^{i}$$

The velocity in `Out[49]` is thus found to be

$$\dot{\rho}^{ij} \approx 2N\pi^{ij} + \left(\mathcal{K}^{ij} - \frac{\mathcal{K}}{3}h^{ij}\right)\rho + \frac{h^{ij}}{3}\nabla_{k}\left(N^{k}\rho\right) - \frac{2\rho}{3}\nabla_{(i}N_{j)}\,. \tag{2.39}$$

In principle, the velocities in eqs. (2.36) to (2.39) can themselves be used to compute all future Poisson brackets. This is sometimes done, but it offers no computational advantage: in *Hamilcar* it is the Poisson bracket operation itself that is taken as fundamental.

Primary constraint algebra When computing the primary constraint algebra it will be useful to define the quantity

$$\Psi^{ij} \equiv \pi^{ij} - \frac{\pi}{3}h^{ij} + \frac{\rho}{6}\left(\mathcal{K}^{ij} - \frac{\mathcal{K}}{3}h^{ij}\right), \tag{2.40}$$

which we will presently identify with a secondary constraint. The corresponding *Hamilcar* definition is:

```
In[50]:= DefTensor[SecondaryConstraint[i, j], M3, Symmetric[{i, j}], PrintAs
 ↪ -> "\[CapitalPsi]"]; AutomaticRules[SecondaryConstraint,
 ↪ MakeRule[{SecondaryConstraint[i, -i], 0}, MetricOn -> All,
 ↪ ContractMetrics -> True]]; FromSecondaryConstraint =
 ↪ MakeRule[{SecondaryConstraint[i, j], Evaluate[Expr]}, MetricOn -> All,
 ↪ ContractMetrics -> True]; PrependTotalFrom[FromSecondaryConstraint];
```

The auto-commutator of the super-Hamiltonian is the Poisson bracket:

```
In[51]:= Expr = {ScalarSmearingF[] * SuperHamiltonian[], ScalarSmearingS[] *
 ↪ SuperHamiltonian[]};
```

Now we feed `In[51]` into `PoissonBracket`:

```
In[52]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ Expr //= TotalTo; Expr //= FullSimplify;
```

Out[52]=

$$\frac{1}{3}\left(-2\,s\,\pi\nabla_a\nabla^a f + 2\,f\,\pi\nabla_a\nabla^a s + 2\,\rho^{bc}\,\mathcal{K}_{bc}\,(s\,\nabla_a\nabla^a f - f\,\nabla_a\nabla^a s) + s\mathcal{K}\nabla_a\rho\nabla^a f - f\mathcal{K}\nabla_a\rho\nabla^a s - \mathcal{K}_{ab}\,s\,\rho\nabla^b\nabla^a f + \mathcal{K}_{ab}\,f\,\rho\nabla^b\nabla^a s\right)$$

We then use `FindAlgebra` to express `Out[52]` in terms of the constraints, making use of `In[50]` — which defines Ψ^{ij} — in our ansatz:

```
In[53]:= Expr //= FindAlgebra[#1, {{{PrimaryConstraint, ExtrinsicCurvature},
 ↪ {CD, CD, ScalarSmearingF, ScalarSmearingS}}, {{SuperMomentum}, {CD,
 ↪ ScalarSmearingF, ScalarSmearingS}}, {{SecondaryConstraint}, {CD, CD,
 ↪ ScalarSmearingF, ScalarSmearingS}}}, Constraints -> {SuperHamiltonian[],
 ↪ SuperMomentum[i], PrimaryConstraint[i, j], SecondaryConstraint[i, j]},
 ↪ Method -> Solve, Verify -> False] & ;
```

Out[53]=

$$\frac{2}{3}\,\mathcal{K}_{ij}\,\Phi^{ij}\,(s\,\nabla_a\nabla^a f - f\,\nabla_a\nabla^a s) + \mathcal{H}^i\,(-s\,\nabla_i f + f\,\nabla_i s) + \Psi^{ij}\,(2\,s\,\nabla_j\nabla_i f - 2\,f\,\nabla_j\nabla_i s)$$

The result `Out[53]` is

$$\{\mathcal{H}[f],\mathcal{H}[s]\} = \mathcal{H}^i\left[f\nabla_i s - s\nabla_i f\right] + \Phi^{ij}\left[\frac{2}{3}\mathcal{K}_{ij}\left(s\nabla_k\nabla^k f - f\nabla_k\nabla^k s\right)\right] + \Psi^{ij}\left[2\left(s\nabla_i\nabla_j f - f\nabla_i\nabla_j s\right)\right]. \tag{2.41}$$

We proceed similarly for the bracket between the super-Hamiltonian and super-momentum:

```
In[54]:= Expr = {ScalarSmearingF[] * SuperHamiltonian[],
 ↪ VectorSmearingCovariantS[i] * SuperMomentum[-i]};
```

Now we feed `In[54]` into `PoissonBracket`:

```
In[55]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ Expr //= TotalTo; Expr //= FullSimplify;
```

Out[55]=

$$\frac{1}{432}\Big(-864\,\pi^{bc}\,\mathcal{K}_{bc}\,f\,\nabla_a s^a + 24\,\mathrm{R}[\nabla]\,f\,\rho\,\nabla_a s^a - \frac{f\,\rho^2\,\nabla_a s^a}{\sqrt{h}} + 24\,f\,\rho\mathcal{K}^2\,\nabla_a s^a - 72\,\mathcal{K}^{bc}\,f\,(s^a\,(12\,\nabla_a\pi_{bc} + 4\,\rho\,\nabla_a\mathcal{K}_{bc} + 3\,\mathcal{K}_{bc}\,\nabla_a\rho) + \mathcal{K}_{bc}\,\rho\,\nabla_a s^a) - 144\,\nabla_a\rho\nabla^a f\,\nabla_b s^b - 48\,\rho\,\nabla_a s^a\,\nabla_b\nabla^b f - 144\,f\,\nabla_a s^a\,\nabla_b\nabla^b\rho + 1728\,\pi_b{}^c\,\mathcal{K}_{ac}\,f\,\nabla^b s^a + 576\,\rho^{cd}\,\mathcal{K}_{ab}\,\mathcal{K}_{cd}\,f\,\nabla^b s^a - 432\,\rho_{ab}\,\mathcal{K}_{cd}\,\mathcal{K}^{cd}\,f\,\nabla^b s^a + 144\,\rho_{ab}\,\mathrm{R}[\nabla]\,f\,\nabla^b s^a + \frac{12\,\rho_{ab}\,f\,\rho\,\nabla^b s^a}{\sqrt{h}} + 288\,\mathcal{K}_a{}^c\,\mathcal{K}_{bc}\,f\,\rho\,\nabla^b s^a - 144\,\mathrm{R}[\nabla]_{ab}\,f\,\rho\,\nabla^b s^a - 576\,\mathcal{K}_{ab}\,f\,\pi\,\nabla^b s^a - 288\,\mathcal{K}_{ab}\,f\,\rho\mathcal{K}\,\nabla^b s^a + 144\,\rho_{ab}\,f\,\mathcal{K}^2\,\nabla^b s^a + 72\,\rho\,\nabla_a\nabla_b f\,\nabla^b s^a + 72\,f\,\nabla_a\nabla_b\rho\,\nabla^b s^a + 72\,\rho\,\nabla_b\nabla_a f\,\nabla^b s^a + 72\,f\,\nabla_b\nabla_a\rho\,\nabla^b s^a + 6\,s^a\,(\nabla_a\rho\,(f\,(12\,\mathrm{R}[\nabla] + \frac{\rho}{\sqrt{h}} + 12\,\mathcal{K}^2) - 24\,\nabla_b\nabla^b f) + 48\,f\,((\rho^{bc}\,\mathcal{K}_{bc} - \pi)\,\nabla_a\mathcal{K} + 2\,\mathcal{K}_a{}^c\,\mathcal{K}_{bc}\,\nabla^b\rho - \mathcal{K}_{ab}\,\rho\,\nabla^b\mathcal{K} + 2\,\mathcal{K}_a{}^b\,(3\,\nabla_c\pi_b{}^c + \rho\,\nabla_c\mathcal{K}_b{}^c))) - 288\,\rho_{ab}\,\nabla^b s^a\,\nabla_c\nabla^c f\Big)$$

We then use **FindAlgebra** to express **Out[55]** in terms of the constraints, again making use of **In[50]** in our ansatz:

```
In[56]:= Expr //= FindAlgebra[#1, {{{PrimaryConstraint}, {CD, CD, CD,
 ↪ ScalarSmearingF, VectorSmearingCovariantS}}, {{PrimaryConstraint,
 ↪ RicciCD}, {CD, ScalarSmearingF, VectorSmearingCovariantS}},
 ↪ {{PrimaryConstraint, ConjugateMomentumExtrinsicCurvature}, {CD,
 ↪ ScalarSmearingF, VectorSmearingCovariantS}}, {{PrimaryConstraint}, {CD,
 ↪ ExtrinsicCurvature, ExtrinsicCurvature, ScalarSmearingF,
 ↪ VectorSmearingCovariantS}}, {{SecondaryConstraint}, {CD,
 ↪ ExtrinsicCurvature, ScalarSmearingF, VectorSmearingCovariantS}},
 ↪ {{SuperMomentum, ExtrinsicCurvature, ScalarSmearingF,
 ↪ VectorSmearingCovariantS}}, {{SuperHamiltonian}, {CD, ScalarSmearingF,
 ↪ VectorSmearingCovariantS}}}, Constraints -> {SuperHamiltonian[],
 ↪ SuperMomentum[i], PrimaryConstraint[i, j], SecondaryConstraint[i, j]},
 ↪ Method -> Solve, DDIs -> False, Verify -> False] & ;
```

Out[56]=

$$-2\,\mathcal{K}_{ia}\,f\,\mathcal{H}^{i}\,s^{a}-\mathcal{H}\,s^{a}\,\nabla_{a}f+2\,f\,\Psi^{ij}\,(s^{a}\,\nabla_{a}\mathcal{K}_{ij}+2\,\mathcal{K}_{ja}\,\nabla_{i}s^{a})+\frac{1}{36}\,\Phi^{ij}\,(24\,\mathcal{K}_{ij}\,f\,(s^{a}\,\nabla_{a}\mathcal{K}+2\,\mathcal{K}_{ab}\,\nabla^{b}s^{a})+(f\,(-36\,\mathcal{K}_{ab}\,\mathcal{K}^{ab}+12\,\mathrm{R}[\nabla]+\rho+12\,\mathcal{K}^{2})-24\,\nabla_{a}\nabla^{a}f)\,\nabla_{j}s_{i})$$

The result **Out[56]** is

$$\begin{aligned}\{\mathcal{H}[f],\mathcal{H}^{i}[s_{i}]\}=\mathcal{H}\Big[-s^{i}\nabla_{i}f\Big]+\mathcal{H}^{i}\Big[-2f\mathcal{K}_{i}^{j}s_{j}\Big]+\Phi^{ij}\Bigg[\frac{2}{3}\mathcal{K}_{ij}f\Big(s^{k}\nabla_{k}\mathcal{K}+2\mathcal{K}_{kl}\nabla^{k}s^{l}\Big)\\-\Big(\frac{2}{3}\nabla_{k}\nabla^{k}f+\Big(\mathcal{K}_{kl}\mathcal{K}^{kl}-\frac{\mathcal{R}}{3}-\frac{\rho}{36\sqrt{h}}-\frac{\mathcal{K}^{2}}{3}\Big)f\Big)\nabla_{i}s_{j}\Bigg]\\+\Psi^{ij}\Big[2f\Big(s^{k}\nabla_{k}\mathcal{K}_{ij}+2\mathcal{K}_{i}^{k}\nabla_{j}s_{k}\Big)\Big].\end{aligned}\tag{2.42}$$

For the auto-commutator of the super-momentum, we set up the Poisson bracket:

```
In[57]:= Expr = {VectorSmearingCovariantF[-i] * SuperMomentum[i],
 ↪ VectorSmearingCovariantS[-j] * SuperMomentum[j]};
```

Now we feed **In[57]** into **PoissonBracket**:

```
In[58]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ Expr //= TotalTo; Expr //= FullSimplify;
```

Out[58]=

$$\begin{aligned}&\tfrac{1}{9}\,(s^{a}\,(3\,\rho\,(\nabla_{a}f_{b}+\nabla_{b}f_{a})\,\nabla^{b}\mathcal{K}-18\,(\nabla_{a}f^{b}+\nabla^{b}f_{a})\,\nabla_{c}\pi_{b}{}^{c}-6\,\rho\,(\nabla_{a}f^{b}+\nabla^{b}f_{a})\,\nabla_{c}\mathcal{K}_{b}{}^{c}-\\&\quad 6\,\mathcal{K}_{bc}\,\nabla^{b}\rho\,(\nabla_{a}f^{c}+\nabla^{c}f_{a})+6\,(3\,\nabla_{a}\pi_{bc}+\rho\,\nabla_{a}\mathcal{K}_{bc}+\mathcal{K}_{bc}\,\nabla_{a}\rho)\,\nabla^{c}f^{b}-\nabla_{a}\mathcal{K}\,(\rho\,\nabla_{b}f^{b}+6\,\rho_{bc}\,\nabla^{c}f^{b}))+\\&\quad 18\,\pi_{bc}\,(\nabla_{a}s^{c}\,\nabla^{b}f^{a}-\nabla_{a}f^{a}\,\nabla^{c}s^{b})+f^{a}\,(3\,s^{b}\,(\nabla_{a}\mathcal{K}\,\nabla_{b}\rho-\nabla_{a}\rho\,\nabla_{b}\mathcal{K})+18\,(\nabla_{a}s^{b}+\nabla^{b}s_{a})\,\nabla_{c}\pi_{b}{}^{c}-18\,\nabla_{a}\pi_{bc}\,\nabla^{c}s^{b}+\\&\quad \nabla_{a}\mathcal{K}\,(\rho\,\nabla_{b}s^{b}+6\,\rho_{bc}\,\nabla^{c}s^{b})-3\,\rho\,((\nabla_{a}s_{b}+\nabla_{b}s_{a})\,\nabla^{b}\mathcal{K}-2\,(\nabla_{a}s^{b}+\nabla^{b}s_{a})\,\nabla_{c}\mathcal{K}_{b}{}^{c}+2\,\nabla_{a}\mathcal{K}_{bc}\,\nabla^{c}s^{b})+\\&\quad 6\,\mathcal{K}_{bc}\,(\nabla^{b}\rho\,(\nabla_{a}s^{c}+\nabla^{c}s_{a})-\nabla_{a}\rho\,\nabla^{c}s^{b}))+2\,(\mathcal{K}_{bc}\,\rho\,(3\,\nabla_{a}s^{c}\,\nabla^{b}f^{a}-\nabla_{a}f^{a}\,\nabla^{c}s^{b})+\\&\quad \nabla^{b}f^{a}\,(9\,\pi_{ab}\,\nabla_{c}s^{c}-3\,(3\,\pi_{ac}+\mathcal{K}_{ac}\,\rho)\,\nabla^{c}s_{b}-6\,\rho_{ab}\,\mathcal{K}_{cd}\,\nabla^{d}s^{c}+\mathcal{K}_{ab}\,(\rho\,\nabla_{c}s^{c}+6\,\rho_{cd}\,\nabla^{d}s^{c}))))\end{aligned}$$

We then use **FindAlgebra** to express **Out[58]** in terms of the constraints, again making use of **In[50]** in our ansatz:

```
In[59]:= Expr //= FindAlgebra[#1, {{{PrimaryConstraint}, {CD, CD,
 ↪ ExtrinsicCurvature, VectorSmearingCovariantF, VectorSmearingCovariantS}},
 ↪ {{SecondaryConstraint}, {CD, CD, VectorSmearingCovariantF,
 ↪ VectorSmearingCovariantS}}, {{SuperMomentum}, {CD,
 ↪ VectorSmearingCovariantF, VectorSmearingCovariantS}}}, Constraints ->
 ↪ {SuperHamiltonian[], SuperMomentum[i], PrimaryConstraint[i, j],
 ↪ SecondaryConstraint[i, j]}, Method -> Solve, Verify -> False] & ;
```

Out[59]= $\mathcal{H}^{i}\,(s^{a}\,\nabla_{i}f_{a}\text{-}\,f^{a}\,\nabla_{i}s_{a})+\frac{1}{3}\,\Phi^{ij}\,(\text{-}2\,(s^{a}\,\nabla_{a}\mathcal{K}+2\,\mathcal{K}_{ab}\,\nabla^{b}s^{a})\,\nabla_{j}f_{i}+2\,(f^{a}\,\nabla_{a}\mathcal{K}+2\,\mathcal{K}_{ab}\,\nabla^{b}f^{a})\,\nabla_{j}s_{i})$

The result Out[59] is

$$\{\mathcal{H}^{i}[f_{i}],\mathcal{H}^{j}[s_{j}]\}=\mathcal{H}^{i}\Big[s^{j}\nabla_{i}f_{j}-f^{j}\nabla_{i}s_{j}\Big]+\Phi^{ij}\Bigg[\frac{2}{3}\Big(\nabla_{k}\mathcal{K}\Big(f^{k}\nabla_{i}s_{j}-s^{k}\nabla_{i}f_{j}\Big)+2\mathcal{K}_{kl}\Big(\nabla^{k}f^{l}\nabla_{i}s_{j}-\nabla^{k}s^{l}\nabla_{i}f_{j}\Big)\Big)\Bigg]\,. \tag{2.43}$$

Next, we compute the commutator of the primary constraint with the super-Hamiltonian:

```
In[60]:= Expr = {TensorSmearingCovariantF[-i, -j] * PrimaryConstraint[i, j],
 ↪ ScalarSmearingS[] * SuperHamiltonian[]};
```

Now we feed In[60] into PoissonBracket:

```
In[61]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ Expr //= TotalTo;
```

Out[61]= $2\,\pi^{ab}\,s\,f_{ab}+\frac{2}{3}\,\rho^{ab}\,\mathcal{K}_{ab}\,s\,f^{c}{}_{c}+\frac{1}{3}\,\mathcal{K}^{ab}\,s\,f_{ab}\,\rho\text{-}\frac{2}{3}\,s\,f^{a}{}_{a}\,\pi\text{-}\frac{1}{3}\,s\,f^{a}{}_{a}\,\rho\mathcal{K}$

We then use FindAlgebra to express Out[61] in terms of the constraints, making use of In[50] in our ansatz:

```
In[62]:= Expr //= TotalFrom; Expr //= FindAlgebra[#1, {{{SecondaryConstraint,
 ↪ TensorSmearingCovariantF, ScalarSmearingS}}, {{PrimaryConstraint,
 ↪ ExtrinsicCurvature, TensorSmearingCovariantF, ScalarSmearingS}}},
 ↪ Constraints -> {PrimaryConstraint[i, j], SecondaryConstraint[i, j]},
 ↪ Method -> Solve, Verify -> False] & ;
```

Out[62]= $\frac{2}{3}\,\mathcal{K}_{ij}\,\Phi^{ij}\,s\,f^{a}{}_{a}+2\,s\,\Psi^{ij}\,f_{ij}$

The result Out[62] is

$$\{\Phi^{ij}[f_{ij}],\mathcal{H}[s]\}=\Phi^{ij}\Big[\frac{2}{3}sf\mathcal{K}_{ij}\Big]+\Psi^{ij}\Big[2sf_{ij}\Big]\,. \tag{2.44}$$

Similarly, we compute the commutator of the primary constraint with the super-momentum:

```
In[63]:= Expr = {TensorSmearingCovariantF[-i, -j] * PrimaryConstraint[i, j],
 ↪ VectorSmearingCovariantS[-j] * SuperMomentum[j]};
```

Now we feed In[63] into PoissonBracket:

```
In[64]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
↪ Expr //= TotalTo;
```

Out[64]= $\frac{2}{9} f^{b}{}_{b} \rho \nabla_{a} s^{a} - \frac{2}{3} \rho_{ab} f^{c}{}_{c} \nabla^{b} s^{a}$

We then use FindAlgebra to express Out[64] in terms of the constraints:

```
In[65]:= Expr //= TotalFrom; Expr //= FindAlgebra[#1, {{{PrimaryConstraint},
↪ {CD, TensorSmearingCovariantF, VectorSmearingCovariantS}}}, Constraints
↪ -> {SuperHamiltonian[], SuperMomentum[i], PrimaryConstraint[i, j],
↪ SecondaryConstraint[i, j]}, Method -> Solve, Verify -> False] & ;
```

Out[65]= $-\frac{2}{3} \Phi^{ij} f^{a}{}_{a} \nabla_{j} s_{i}$

The result Out[65] is

$$\left\{\Phi^{ij}[f_{ij}], \mathcal{H}^{k}[s_{k}]\right\} = \Phi^{ij}\left[-\frac{2}{3} f \nabla_{i} s_{j}\right]. \tag{2.45}$$

Finally, we can assert without detailed computation that the auto-commutator of the primary constraint vanishes, due to the definition in eq. (2.27), i.e.,

$$\left\{\Phi^{ij}[f_{ij}], \Phi^{kl}[s_{kl}]\right\} = 0\,. \tag{2.46}$$

Secondary constraint algebra The conservation of all the primary constraints, according to the algebroid in eqs. (2.41) to (2.46), necessitates us to identify eq. (2.40) as a secondary constraint, i.e.,

$$\Psi^{ij} \approx 0\,, \tag{2.47}$$

since eq. (2.47) alone implies that the full algebra between primary constraints weakly vanishes. Let us now compute the Poisson bracket of the secondary constraint with the primary constraint:

```
In[66]:= Expr = {TensorSmearingCovariantF[-i, -j] * SecondaryConstraint[i,
↪ j], TensorSmearingCovariantS[-k, -l] * PrimaryConstraint[k, l]};
```

Now we feed In[66] into PoissonBracket and simplify on the primary constraint shell using In[35]. We don't bother to use FindAlgebra for this task:

```
In[67]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
↪ Expr //= TotalTo; Expr //= #1 /. ToPrimaryShellExplicit & ; Expr //=
↪ ToCanonical; Expr //= ContractMetric; Expr //= ScreenDollarIndices; Expr
↪ //= FullSimplify;
```

Out[67]= $\frac{1}{18} (-((3 f^{ab} s_{ab} + f^{a}{}_{a} s^{b}{}_{b}) \rho) + 2 s^{c}{}_{c} (3 \Phi^{ab} f_{ab} + f_{a}{}^{a} \rho))$

From Out[67] the bracket between the two traceless constraints is

$$\left\{\Psi^{ij}[f_{ij}], \Phi^{kl}[s_{kl}]\right\} = \Phi^{ij}\left[\frac{s}{3} f_{ij}\right] + \rho\left[-\frac{1}{6}\left(f_{ij} s^{ij} - \frac{f s}{3}\right)\right]. \tag{2.48}$$

An important feature of eq. (2.48), which is discussed in [25], is that it vanishes at points where $\rho = 0$. In the bulk of the phase space, the bracket does not vanish, and so the two traceless constraints are generally taken to be second class. The auto-commutator of the secondary constraint self-evidently vanishes due to the definition in eq. (2.27), i.e.,

$$\big\{\Psi^{ij}[f_{ij}], \Psi^{kl}[s_{kl}]\big\} = 0\,. \tag{2.49}$$

From this point onwards for the R^2 model, we take the functionality of *Hamilcar* to be well-demonstrated, and we proceed directly by stating results.

Determination of the multiplier Because of the second class property following from eq. (2.48), the conservation of Ψ^{ij} is not expected to generate further constraints. Instead, the conservation law is satisfied so long as the traceless part of the Lagrange multiplier λ_{ij} obeys the following solution on the shell:

$$\begin{aligned}\lambda_{ij} - \frac{\lambda}{3}h_{ij} \approx \Big(\delta^k_{(i}\delta^l_{j)} - \frac{1}{3}h^{kl}h_{ij}\Big)\Bigg[&\mathcal{R}_{kl} - 2\Big(\mathcal{K}^m_k\mathcal{K}_{lm} - \frac{\mathcal{K}}{3}\mathcal{K}_{kl}\Big)\\ &- \frac{1}{N}\Big(\nabla_k\nabla_l N - 2\mathcal{K}^m_k\nabla_l N_m - N^m\nabla_m\mathcal{K}_{kl}\Big)\\ &- \frac{1}{\rho}\Big(2\pi\mathcal{K}_{kl} + \nabla_k\nabla_l\rho\Big)\Bigg].\end{aligned} \tag{2.50}$$

Once again, we note that eq. (2.50) has a relevant feature, which is a direct consequence of eq. (2.48): a *negative* power of ρ appears in the final term. As discussed in [25], this suggests that the model may have some curious properties in the vicinity of $\rho = 0$. We did not encounter any solutions such as eq. (2.50) in the case of GR in section 2.2. In that case, the multipliers were restricted to N and N^i, which remained entirely undetermined due to the diffeomorphism-invariance of the theory. The theory eq. (2.23) is also diffeomorphism-invariant, and so no further solutions are forthcoming.

Dressing of the constraints In order for N and N^i to remain undetermined, it is sufficient, but not necessary, that the $\mathcal{H}^i$ and $\mathcal{H}$ be first class. This would, however, give rise to an inconsistency, since the dynamical evolution of λ_{ij} in eq. (2.50) is non-trivial. At an even more prosaic level, commutators of first class functions are necessarily first class, whilst eqs. (2.41) to (2.43) are strongly linear in Φ^{ij} and Ψ^{ij}, which are currently thought to be second class. Accordingly, we define candidate first class constraints $\mathfrak{H}^i$ and $\mathfrak{H}$ by dressing the $\mathcal{H}^i$ and $\mathcal{H}$. The dressed constraints are defined as follows:

$$\mathfrak{H} \equiv \mathcal{H} + \Phi^{ij}\Big(\mathcal{R}_{ij} - 2\Big(\mathcal{K}^k_i - \frac{\mathcal{K}}{3}\delta^k_i\Big)\mathcal{K}_{jk}\Big) - \frac{\Phi^{ij}}{\rho}\left(2\pi\mathcal{K}_{ij} + \nabla_i\nabla_j\rho\right) - \nabla_i\nabla_j\Phi^{ij}, \tag{2.51a}$$

$$\mathfrak{H}^i \equiv \mathcal{H}^i + \Phi^{jk}\nabla^i\mathcal{K}_{jk} - 2\nabla_k\big(\Phi^{jk}\mathcal{K}^i_j\big). \tag{2.51b}$$

Notice that eq. (2.51b) simply restores the portion of eq. (2.29b) that was lost in the shift to eq. (2.30b), revealing a false economy in eliminating Φ^{ij} from $\mathcal{H}^i$. A similar effect is seen in eqs. (2.29a), (2.30a) and (2.51a), though in this case the dressed $\mathfrak{H}$ contains extra

structures. In particular, the negative power of ρ appears once again, as it did in eq. (2.50). The dressed constraints in eqs. (2.51a) and (2.51b) have the following commutators with the secondary constraint:

$$\begin{aligned}\big\{\Psi^{ij}[f_{ij}],\mathfrak{H}[s]\big\} &= \Phi^{ij}\bigg[2s\Big(f^{kl}-\frac{f}{3}h^{kl}\Big)\Big(\frac{1}{3}\mathcal{K}_{kl}\mathcal{K}_{ij}-\mathcal{K}_{ik}\mathcal{K}_{jl}\Big)-\frac{\rho s f_{ij}}{48\sqrt{h}}\\ &\qquad+\frac{1}{2\rho}\nabla^k\Big(\rho s\Big(\nabla_k f_{ij}-2\nabla_i\Big(f_{kj}-\frac{f}{3}h_{kj}\Big)\Big)\Big)\bigg]\\ &\quad+\Psi^{ij}\bigg[\frac{2sf}{3}\mathcal{K}_{ij}\bigg]+\mathcal{H}\Phi^{ij}\bigg[\frac{s}{\rho}f_{ij}\bigg]+\Phi^{ij}\Psi^{kl}\bigg[\frac{2s}{\rho}\Big(\mathcal{K}_{ij}f_{kl}+\mathcal{K}_{kl}f_{ij}\Big)\bigg]\\ &\quad+\Phi^{ij}\Phi^{kl}\bigg[-\frac{s}{\rho^2}f_{ij}\Big(2\mathcal{K}_{kl}\pi+\nabla_k\nabla_l\rho\Big)\bigg]\\ &\quad+\Phi^{ij}\bigg[f_{ij}\nabla_k\nabla_l\Big(\frac{s}{\rho}\Phi^{kl}\Big)\bigg],\end{aligned}\tag{2.52a}$$

$$\begin{aligned}\big\{\Psi^{ij}[f_{ij}],\mathfrak{H}^k[s_k]\big\} &= \mathcal{H}^i\bigg[f_i^j s_j-\frac{f}{3}s_i\bigg]+\Phi^{ij}\bigg[\Big(f_l^k-\frac{f}{3}\delta_l^k\Big)s^l\nabla_k\mathcal{K}_{ij}\\ &\qquad+2\mathcal{K}_i^l\nabla_j\bigg(\Big(f_l^k-\frac{f}{3}\delta_l^k\Big)s_k\bigg)\bigg]\\ &\quad+\Psi^{ij}\bigg[-s^k\nabla_k f_{ij}-2f_j^k\nabla_i s_k\bigg].\end{aligned}\tag{2.52b}$$

As required, the commutators with Ψ^{ij} in eqs. (2.52a) and (2.52b) vanish weakly. Moreover, the dressing corrections in eqs. (2.51a) and (2.51b) are all linear in Φ^{ij}, so that $\mathfrak{H}$ and $\mathfrak{H}^i$ preserve the property of $\mathcal{H}$ and $\mathcal{H}^i$ — as seen in eqs. (2.44) and (2.45) — of still weakly commuting with Φ^{ij}. Taken together, these observations support the hope that $\mathfrak{H}$ and $\mathfrak{H}^i$ may indeed be first class.

The deformed Dirac algebra This property is confirmed by the strong algebroid

$$\begin{aligned}\big\{\mathfrak{H}[f],\mathfrak{H}[s]\big\} &= \mathfrak{H}^i\Big[f\nabla_i s-s\nabla_i f\Big]+\mathfrak{H}^k\Phi_k^i\bigg[\frac{6}{\rho}\Big(f\nabla_i s-s\nabla_i f\Big)\bigg]\\ &\quad+\Phi^{ij}\bigg[\frac{12}{\rho}\nabla_k\Psi_j^k\Big(f\nabla_i s-s\nabla_i f\Big)\bigg]\\ &\quad+\Phi^{ij}\Phi^{kl}\bigg[\frac{6}{\rho}\Big(\nabla_j\mathcal{K}_{kl}+\frac{1}{\rho}\mathcal{K}_{kl}\nabla_j\rho\Big)\Big(s\nabla_i f-f\nabla_i s\Big)\bigg]\\ &\quad+\Phi^{ij}\bigg[\frac{2}{\rho}\Big(6\nabla_l\big(\mathcal{K}_{jk}\Phi^{kl}\big)+\nabla_j\big(\mathcal{K}_{kl}\Phi^{kl}\big)\Big)\Big(f\nabla_i s-s\nabla_i f\Big)\bigg],\end{aligned}\tag{2.53a}$$

$$\big\{\mathfrak{H}[f],\mathfrak{H}^i[s_i]\big\} = \mathfrak{H}\bigg[-s^i\nabla_i f\bigg]+\mathfrak{H}^i\bigg[-2f\mathcal{K}_i^j s_j\bigg]+\mathfrak{H}^i\Phi^{jk}\bigg[-\frac{2f}{\rho}s_i\mathcal{K}_{jk}\bigg],\tag{2.53b}$$

$$\big\{\mathfrak{H}^i[f_i],\mathfrak{H}^j[s_j]\big\} = \mathfrak{H}^i\Big[s^j\nabla_i f_j-f^j\nabla_i s_j\Big],\tag{2.53c}$$

which vanishes weakly. At the strong level, although the commutators in eqs. (2.53a) to (2.53c) are between first class constraints, they do not form a *closed* algebroid, as is the

case in GR.[5] Since, however, they are strongly equal to expressions which are linear in first class constraints and quadratic in constraints of either class, they do themselves have the expected property of being first class, in contrast to eqs. (2.41) to (2.43). In particular, the latter two terms in eq. (2.53b) are superfluous, and together with the non-standard format in eq. (2.53c) are a result of our poor choice of convention in which the contravariant components $\mathfrak{H}^i$ are used in the basis. It is the covariant components $\mathfrak{H}_i$ which carry physical significance as the generators of transverse diffeomorphisms, and indeed eqs. (2.53b) and (2.53c) can be readily replaced by

$$\big\{\mathfrak{H}[f],\mathfrak{H}_i[s^i]\big\} = \mathfrak{H}\Big[-s^i\nabla_i f\Big], \quad \big\{\mathfrak{H}_i[f^i],\mathfrak{H}_j[s^j]\big\} = \mathfrak{H}_i\Big[f^j\nabla_j s^i - s^j\nabla_j f^i\Big], \tag{2.54}$$

where eq. (2.54) represents the part of the standard Dirac hypersurface deformation algebroid given already in eqs. (2.19) and (2.20). By contrast, the structural difference between eq. (2.53a) and eq. (2.18) is more significant in nature, and distinguishes the R^2 model from pure GR.

Final summary To collect our results together, we find that the total number of first class constraints among $\mathfrak{H}$ and $\mathfrak{H}^i$ is $N_{\text{1st}}=4$. The total number of second class constraints among Φ^{ij} and Ψ^{ij} is $N_{\text{2nd}}=10$, where we recall that each of these tensorial constraints is traceless and symmetric. Compared to the case of GR, we have twice as many canonical variables distributed between the spatial metric h_{ij} and the conjugate momentum π^{ij}, and also the auxiliary extrinsic curvature $\mathcal{K}_{ij}$ and its conjugate momentum ρ^{ij} — a total of $N_{\text{Can}}=24$ canonical variables. The final number of propagating modes is found from eq. (1.1) to be

$$N_{\text{Phy}} = \frac{1}{2}\Big(24 - 2\times 4 - 10\Big) = 3\,. \tag{2.55}$$

These are accounted for by the two tensorial modes of the graviton, plus an additional scalar mode arising from the higher-derivative nature of the theory.

2.4 Case study: pure GR at two loops

Overview of the theory The final gravity theory we consider is an extension of eq. (2.1) with a cosmological constant and a cubic Riemann correction term, defined by the action

$$\mathscr{S}[g_{\mu\nu}] = \int \mathrm{d}^4x\,\sqrt{-g}\left[\frac{R-2\Lambda}{\kappa^2} + \alpha\kappa^2 R^{\mu\nu}{}_{\rho\sigma}R^{\rho\sigma}{}_{\alpha\beta}R^{\alpha\beta}{}_{\mu\nu}\right], \tag{2.56}$$

where Λ is the cosmological constant and α is a small coupling parameter. The cubic Riemann term arises as the leading quantum correction at two-loop order in perturbative quantum gravity. In *Hamilcar*, the cosmological constant is defined as:

```
In[68]:= DefConstantSymbol[CosmologicalConstant, PrintAs ->
 ↪ "\[CapitalLambda]"];
```

and the Wilson coefficient α as:

[5] See also [31] for another example of closure being spoiled by dressing of the Hamiltonian constraint.

```
In[69]:= DefConstantSymbol[WilsonCoefficient, PrintAs -> "\[Alpha]"];
```

The gravitational coupling κ was already defined in section 2.2 via `In[6]`, so we do not redefine it here.

2.4.1 Canonical formulation

Shorthand notation As shown in [27], the analysis to follow benefits from defining shorthand notation for three tensors, namely

$$\mathscr{K}_{ij} \equiv -\frac{\kappa^2}{\sqrt{h}}\left(\pi_{ij} - \frac{\pi}{2}h_{ij}\right), \quad \mathscr{L}^{i}{}_{jk} \equiv 2\nabla_{[k}\mathscr{K}_{j]}{}^{i}, \quad \mathscr{Q}^{ij}{}_{kl} \equiv 2\mathscr{K}^{i}_{[k}\mathscr{K}^{j}_{l]} + \mathcal{R}^{ij}{}_{kl}. \tag{2.57}$$

Their traces are $\mathscr{K} \equiv \mathscr{K}_i{}^i$, $\mathscr{L}_i \equiv \mathscr{L}^j{}_{ji}$, $\mathscr{Q}^i_j \equiv \mathscr{Q}^{ik}{}_{jk}$, and $\mathscr{Q} \equiv \mathscr{Q}^i_i$. The corresponding definitions in *Hamilcar* are as follows. First, the trace-shifted momentum $\mathscr{K}_{ij}$ is defined as:

```
In[70]:= DefTensor[ScriptK[-i, -j], M3, Symmetric[{-i, -j}], PrintAs ->
 ↪ "\[GothicCapitalK]"]; FromScriptK = MakeRule[{ScriptK[-i, -j],
 ↪ (-EinsteinConstant^2) * (ConjugateMomentumG[-i, -j]/Sqrt[DetG[]] - (G[-i,
 ↪ -j]/2) * (TraceConjugateMomentumG[]/Sqrt[DetG[]]))}, MetricOn -> All,
 ↪ ContractMetrics -> True]; PrependTotalFrom[FromScriptK];
```

Next, the antisymmetric gradient $\mathscr{L}^i{}_{jk}$ is defined as:

```
In[71]:= DefTensor[ScriptL[i, -j, -k], M3, Antisymmetric[{-j, -k}], PrintAs
 ↪ -> "\[GothicCapitalL]"]; FromScriptL = MakeRule[{ScriptL[i, -j, -k],
 ↪ CD[-k][ScriptK[-j, i]] - CD[-j][ScriptK[-k, i]]}, MetricOn -> All,
 ↪ ContractMetrics -> True]; PrependTotalFrom[FromScriptL];
```

Finally, the Riemann-like quantity $\mathscr{Q}^{ij}{}_{kl}$ is defined as:

```
In[72]:= DefTensor[ScriptQ[i, j, -k, -l], M3, {Antisymmetric[{i, j}],
 ↪ Antisymmetric[{-k, -l}]}, PrintAs -> "\[GothicCapitalQ]"]; FromScriptQ =
 ↪ MakeRule[{ScriptQ[i, j, -k, -l], ScriptK[i, -k] * ScriptK[j, -l] -
 ↪ ScriptK[i, -l] * ScriptK[j, -k] + RiemannCD[i, j, -k, -l]}, MetricOn ->
 ↪ All, ContractMetrics -> True]; PrependTotalFrom[FromScriptQ];
```

The corresponding traces are defined as follows. The scalar trace $\mathscr{K}$ is:

```
In[73]:= DefTensor[TraceScriptK[], M3, PrintAs -> "\[GothicCapitalK]"];
 ↪ FromTraceScriptK = MakeRule[{TraceScriptK[], G[i, j] * ScriptK[-i, -j]},
 ↪ MetricOn -> All, ContractMetrics -> True];
 ↪ PrependTotalFrom[FromTraceScriptK];
```

The vector trace $\mathscr{L}_i$ is:

```
In[74]:= DefTensor[ScriptLContraction[-i], M3, PrintAs ->
 ↪ "\[GothicCapitalL]"]; FromScriptLContraction =
 ↪ MakeRule[{ScriptLContraction[-i], ScriptL[j, -j, -i]}, MetricOn -> All,
 ↪ ContractMetrics -> True]; PrependTotalFrom[FromScriptLContraction];
```

The two-index trace $\mathscr{Q}^i_j$ is:

```
In[75]:= DefTensor[ScriptQSingleContraction[-i, -j], M3, Symmetric[{-i, -j}],
 ↪ PrintAs -> "\[GothicCapitalQ]"]; FromScriptQSingleContraction =
 ↪ MakeRule[{ScriptQSingleContraction[i, -j], ScriptQ[i, k, -j, -k]},
 ↪ MetricOn -> All, ContractMetrics -> True];
 ↪ PrependTotalFrom[FromScriptQSingleContraction];
```

The full scalar trace $\mathscr{Q}$ is:

```
In[76]:= DefTensor[TraceScriptQ[], M3, PrintAs -> "\[GothicCapitalQ]"];
 ↪ FromTraceScriptQ = MakeRule[{TraceScriptQ[], ScriptQ[i, j, -i, -j]},
 ↪ MetricOn -> All, ContractMetrics -> True];
 ↪ PrependTotalFrom[FromTraceScriptQ];
```

Reduced canonical action The *reduced* canonical action was found in [27]. The procedures used can be replicated in *Hamilcar*, but are not instructive for the purposes of this paper, so we simply state the final result as

$$\mathscr{S}\left[N, N_i, h_{ij}, \pi^{ij}\right] = \int d^4x \left[\pi^{ij}\dot{h}_{ij} - N\mathbb{H} - N_i\mathbb{H}^i\right], \tag{2.58}$$

where the reduced Hamiltonian and momentum density are, respectively,

$$\mathbb{H} = \sqrt{h}\Bigg[-\frac{(\mathscr{Q}-2\Lambda)}{\kappa^2} + 8\alpha\kappa^2\mathscr{Q}^i_j\Big(2\mathscr{Q}^j_k\mathscr{Q}^k_i - 3\mathscr{L}^j{}_{kl}\,\mathscr{L}_i{}^{kl}\Big) - 24\alpha\kappa^2\Lambda\Big(2\mathscr{Q}^i_j\mathscr{Q}^j_i - 2\Lambda^2 - \mathscr{L}_k{}^{ij}\,\mathscr{L}^k{}_{ij}\Big)\Bigg], \tag{2.59a}$$

$$\mathbb{H}^i = \sqrt{h}\left[-\frac{2\mathscr{L}^i}{\kappa^2}\right]. \tag{2.59b}$$

It is an essential point that eq. (2.58) has precisely the same structure as the canonical action of GR given in eq. (2.12). The symplectic part is identical, and only the definitions of the super-Hamiltonian and super-momentum in eqs. (2.59a) and (2.59b) differ from their GR counterparts in eq. (2.13). By contrast, the canonical action of R^2 theory in eq. (2.28) has a richer symplectic structure and different constraints: the idea is that the higher derivative order in this latter theory is faithfully reflected in the canonical analysis, whereas the two-loop cubic Riemann theory in eq. (2.56) has been through a process of order reduction to produce eq. (2.58). The corresponding *Hamilcar* definitions are as follows. The reduced super-Hamiltonian $\mathbb{H}$ is defined as:

```
In[77]:= DefTensor[ReducedHamiltonianConstraint[], M3, PrintAs -> "\!\(\*
 ↪ SubscriptBox[\(\[ScriptCapitalH]\),\(red\)]\)"];
 ↪ FromReducedHamiltonianConstraint =
 ↪ MakeRule[{ReducedHamiltonianConstraint[], Sqrt[DetG[]] *
 ↪ (-(TraceScriptQ[] - 2 * CosmologicalConstant)/EinsteinConstant^2 + 8 *
 ↪ WilsonCoefficient * EinsteinConstant^2 * ScriptQSingleContraction[i, -j]
```

```
↪ * (2 * ScriptQSingleContraction[j, -k] * ScriptQSingleContraction[k, -i]
↪ - 3 * ScriptL[j, -k, -l] * ScriptL[-i, k, l]) - 24 * WilsonCoefficient *
↪ EinsteinConstant^2 * CosmologicalConstant * (2 *
↪ ScriptQSingleContraction[i, -j] * ScriptQSingleContraction[j, -i] - 2 *
↪ CosmologicalConstant^2 - ScriptL[-k, i, j] * ScriptL[k, -i, -j]))},
↪ MetricOn -> All, ContractMetrics -> True];
↪ PrependTotalFrom[FromReducedHamiltonianConstraint];
```

The reduced super-momentum $\mathbb{H}^i$ is defined as:

```
In[78]:= DefTensor[ReducedMomentumConstraint[i], M3, PrintAs -> "\!\(\*
↪ SubscriptBox[\(\[ScriptCapitalH]\),\(red\)]\)"];
↪ FromReducedMomentumConstraint = MakeRule[{ReducedMomentumConstraint[i],
↪ Sqrt[DetG[]] * (-2 * (ScriptLContraction[i]/EinsteinConstant^2))},
↪ MetricOn -> All, ContractMetrics -> True];
↪ PrependTotalFrom[FromReducedMomentumConstraint];
```

2.4.2 Dirac–Bergmann algorithm

Total Hamiltonian The variation of eq. (2.58) with respect to N and N_i once again reveals the constrained nature of the reduced super-Hamiltonian and super-momentum

$$\mathbb{H} \approx 0\,, \quad \mathbb{H}^i \approx 0\,, \tag{2.60}$$

which is analogous to eq. (2.14), meanwhile the usual Legendre transformation is apparent in the simple format of eq. (2.58), leading to the total Hamiltonian

$$\mathscr{H} = \int \mathrm{d}^3x \, \left(N\mathbb{H} + N_i \mathbb{H}^i \right) , \tag{2.61}$$

which is analogous to eq. (2.15).

Expansion in the Wilson coefficient To facilitate the constraint algebra analysis, we decompose the reduced Hamiltonian constraint into zeroth-order and first-order parts in the Wilson coefficient α, denoted as $\mathbb{H} = \overset{(0)}{\mathbb{H}} + \overset{(1)}{\mathbb{H}}$, where

$$\overset{(0)}{\mathbb{H}} \equiv -\frac{\sqrt{h}}{\kappa^2} \left(\mathscr{Q} - 2\Lambda \right) , \tag{2.62a}$$

$$\overset{(1)}{\mathbb{H}} \equiv 8\alpha\kappa^2 \sqrt{h} \left[\mathscr{Q}^i_j \left(2\mathscr{Q}^j_k \mathscr{Q}^k_i - 3\mathscr{L}^j{}_{kl}\, \mathscr{L}_i{}^{kl} \right) - 3\Lambda \left(2\mathscr{Q}^i_j \mathscr{Q}^j_i - 2\Lambda^2 - \mathscr{L}_{ijk}\, \mathscr{L}^{ijk} \right) \right] . \tag{2.62b}$$

The corresponding *Hamilcar* definitions for the zeroth-order part $\overset{(0)}{\mathbb{H}}$ is:

```
In[79]:= DefTensor[ReducedHamiltonianOrderUnity[], M3, PrintAs -> "\!\(\*
↪ SubscriptBox[\(\*
↪ OverscriptBox[\(\[ScriptCapitalH]\),\((0)\)]\),\(red\)]\)"]; Expr =
↪ ReducedHamiltonianConstraint[]; Expr //= #1 /.
↪ FromReducedHamiltonianConstraint & ; Expr //= Coefficient[#1,
```

```
↪ WilsonCoefficient, 0] & ; FromReducedHamiltonianOrderUnity =
↪ MakeRule[{ReducedHamiltonianOrderUnity[], Evaluate[Expr]}, MetricOn ->
↪ All, ContractMetrics -> True];
↪ PrependTotalFrom[FromReducedHamiltonianOrderUnity];
```

The first-order part $\overset{(1)}{\mathbb{H}}$ is defined as:

```
In[80]:= DefTensor[ReducedHamiltonianOrderWilsonCoefficient[], M3, PrintAs ->
↪ "\!\(\*SubscriptBox[\(\*
↪ OverscriptBox[\(\[ScriptCapitalH]\),\((1)\)]\),\(red\)]\)"]; Expr =
↪ ReducedHamiltonianConstraint[]; Expr //= #1 /.
↪ FromReducedHamiltonianConstraint & ; Expr //= Coefficient[#1,
↪ WilsonCoefficient, 1] & ; Expr * = WilsonCoefficient;
↪ FromReducedHamiltonianOrderWilsonCoefficient =
↪ MakeRule[{ReducedHamiltonianOrderWilsonCoefficient[], Evaluate[Expr]},
↪ MetricOn -> All, ContractMetrics -> True];
↪ PrependTotalFrom[FromReducedHamiltonianOrderWilsonCoefficient];
```

The only interesting part of the constraint algebra is the auto-commutator of the reduced super-Hamiltonian. To analyze it, we compute the Poisson bracket as a sum of three contributions, to first order in the Wilson coefficient α, namely

$$\{\mathbb{H}[f],\mathbb{H}[s]\} = \left\{\overset{(0)}{\mathbb{H}}[f],\overset{(0)}{\mathbb{H}}[s]\right\} + \left\{\overset{(0)}{\mathbb{H}}[f],\overset{(1)}{\mathbb{H}}[s]\right\} + \left\{\overset{(1)}{\mathbb{H}}[f],\overset{(0)}{\mathbb{H}}[s]\right\} + \mathcal{O}(\alpha^2)\,. \tag{2.63}$$

The first term in eq. (2.63) is the auto-commutator of the zeroth-order part, while the second and third terms are the cross-brackets between zeroth and first-order parts. The term $\left\{\overset{(1)}{\mathbb{H}}[f],\overset{(1)}{\mathbb{H}}[s]\right\}$ is of order α^2, and is therefore neglected in this analysis. The computation proceeds by first setting up each bracket with the appropriate smearing functions, then invoking the **PoissonBracket** function from *Hamilcar*. For the leading contribution in eq. (2.63), we set up the bracket:

```
In[81]:= Expr = {ScalarSmearingF[] * ReducedHamiltonianOrderUnity[],
↪ ScalarSmearingS[] * ReducedHamiltonianOrderUnity[]};
```

We then compute the Poisson bracket from `In[81]` as follows:

```
In[82]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
↪ OrderUnityBracketValue = Expr;
```

```
Out[82]=  -2 π_ab s ∇^b∇^a f + 2 π_ab f ∇^b∇^a s
```

For the first cross-term in eq. (2.63), we set up the bracket:

```
In[83]:= Expr = {ScalarSmearingF[] * ReducedHamiltonianOrderUnity[],
↪ ScalarSmearingS[] * ReducedHamiltonianOrderWilsonCoefficient[]};
```

We then compute the bracket from `In[83]` as follows:

```
In[84]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
↪ CrossBracket01Value = Expr;
```

Out[84]=

$$(24\,\Lambda^3\,\kappa^4\,\alpha\,\pi^a{}_a\,f\,s + \frac{40\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi_b{}^d\,\pi^{bc}\,\pi_c{}^e\,\pi_d{}^f\,\pi_e{}^g\,\pi_{fg}\,f\,s}{\hbar^3} + \frac{12\,\kappa^{16}\,\alpha\,\pi_{ab}\,\pi^{ab}\,\pi_c{}^e\,\pi^{cd}\,\pi_d{}^f\,\pi_e{}^g\,\pi_{fg}\,f\,s}{\hbar^3} - \frac{66\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi^b{}_b\,\pi_c{}^e\,\pi^{cd}\,\pi_d{}^f\,\pi_e{}^g\,\pi_{fg}\,f\,s}{\hbar^3} - \frac{24\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi_{bc}\,\pi^{bc}\,\pi_d{}^f\,\pi^{de}\,\pi_e{}^g\,\pi_{fg}\,f\,s}{\hbar^3} + \frac{42\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi^b{}_b\,\pi^c{}_c\,\pi_d{}^f\,\pi^{de}\,\pi_e{}^g\,\pi_{fg}\,f\,s}{\hbar^3} + \frac{15\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi^b{}_b\,\pi_{cd}\,\pi^{cd}\,\pi_e{}^g\,\pi^{ef}\,\pi_{fg}\,f\,s}{\hbar^3} - \frac{25\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi^b{}_b\,\pi^c{}_c\,\pi^d{}_d\,\pi_e{}^g\,\pi^{ef}\,\pi_{fg}\,f\,s}{2\,\hbar^3} - \frac{3\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi^b{}_b\,\pi^c{}_c\,\pi_{de}\,\pi^{de}\,\pi_{fg}\,\pi^{fg}\,f\,s}{\hbar^3} + \frac{3\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi^b{}_b\,\pi^c{}_c\,\pi^d{}_d\,\pi^e{}_e\,\pi_{fg}\,\pi^{fg}\,f\,s}{2\,\hbar^3}) + [\ 753\ \text{similar terms}\ ]$$

For the second cross-term in eq. (2.63), we set up the bracket:

```
In[85]:= Expr = {ScalarSmearingF[] *
 ↪ ReducedHamiltonianOrderWilsonCoefficient[], ScalarSmearingS[] *
 ↪ ReducedHamiltonianOrderUnity[]};
```

We then compute the bracket from In[85] as follows:

```
In[86]:= Expr //= (PoissonBracket[#1, #2, Parallel -> True] & ) @@ #1 & ;
 ↪ CrossBracket10Value = Expr;
```

Out[86]=

$$(-24\,\Lambda^3\,\kappa^4\,\alpha\,\pi^a{}_a\,f\,s - \frac{40\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi_b{}^d\,\pi^{bc}\,\pi_c{}^e\,\pi_d{}^f\,\pi_e{}^g\,\pi_{fg}\,f\,s}{\hbar^3} - \frac{12\,\kappa^{16}\,\alpha\,\pi_{ab}\,\pi^{ab}\,\pi_c{}^e\,\pi^{cd}\,\pi_d{}^f\,\pi_e{}^g\,\pi_{fg}\,f\,s}{\hbar^3} + \frac{66\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi^b{}_b\,\pi_c{}^e\,\pi^{cd}\,\pi_d{}^f\,\pi_e{}^g\,\pi_{fg}\,f\,s}{\hbar^3} + \frac{24\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi_{bc}\,\pi^{bc}\,\pi_d{}^f\,\pi^{de}\,\pi_e{}^g\,\pi_{fg}\,f\,s}{\hbar^3} - \frac{42\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi^b{}_b\,\pi^c{}_c\,\pi_d{}^f\,\pi^{de}\,\pi_e{}^g\,\pi_{fg}\,f\,s}{\hbar^3} - \frac{15\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi^b{}_b\,\pi_{cd}\,\pi^{cd}\,\pi_e{}^g\,\pi^{ef}\,\pi_{fg}\,f\,s}{\hbar^3} + \frac{25\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi^b{}_b\,\pi^c{}_c\,\pi^d{}_d\,\pi_e{}^g\,\pi^{ef}\,\pi_{fg}\,f\,s}{2\,\hbar^3} + \frac{3\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi^b{}_b\,\pi^c{}_c\,\pi_{de}\,\pi^{de}\,\pi_{fg}\,\pi^{fg}\,f\,s}{\hbar^3} - \frac{3\,\kappa^{16}\,\alpha\,\pi^a{}_a\,\pi^b{}_b\,\pi^c{}_c\,\pi^d{}_d\,\pi^e{}_e\,\pi_{fg}\,\pi^{fg}\,f\,s}{2\,\hbar^3}) + [\ 753\ \text{similar terms}\ ]$$

The outputs from In[84] and In[86] (shown truncated in Out[84] and Out[86]) are too lengthy to display in full. Before combining the brackets, we define a utility function that will be used to simplify expressions to a canonical form, via a standard sequence of *xAct* routines:

```
In[87]:= StandardSimplify[Expr_] := Module[{Result = Expr}, Result //=
 ↪ ToCanonical; Result //= ContractMetric; Result //= ScreenDollarIndices;
 ↪ Result //= CollectTensors; Result];
```

With the aid of In[87], we combine all three brackets to obtain the total auto-commutator:

```
In[88]:= Expr = OrderUnityBracketValue + CrossBracket01Value +
 ↪ CrossBracket10Value; Expr //= StandardSimplify; BracketKValue = Expr;
```

The result contains terms at various orders in both the cosmological constant Λ and the Wilson coefficient α. To analyze different parts of the constraint algebra systematically, we define another utility function that extracts specific orders from the total bracket using series expansions:

```
In[89]:= Options[ExtractBracketAnatomy] = {CosmologicalConstantOrder -> All,
 ↪ WilsonCoefficientOrder -> All}; ExtractBracketAnatomy[OptionsPattern[]]
 ↪ := Module[{Expr = BracketKValue}, If[
```

```
↪ !OptionValue[CosmologicalConstantOrder] === All, Expr //= Series[#1,
↪ {CosmologicalConstant, 0, 10}] & ; Expr //= Normal; Expr =
↪ (Coefficient[Expr, CosmologicalConstant, #1] * CosmologicalConstant^#1 &
↪ ) /@ OptionValue[CosmologicalConstantOrder]; Expr //= Total; ]; If[
↪ !OptionValue[WilsonCoefficientOrder] === All, Expr //= Series[#1,
↪ {WilsonCoefficientOrder, 0, 10}] & ; Expr //= Normal; Expr =
↪ (Coefficient[Expr, WilsonCoefficient, #1] * WilsonCoefficient^#1 & ) /@
↪ OptionValue[WilsonCoefficientOrder]; Expr //= Total; ]; Expr //=
↪ StandardSimplify; Expr];
```

The function in In[89] allows us to isolate, for example, the zeroth-order contribution in Λ (which itself contains both zeroth and first-order terms in α), or the first-order contribution in Λ, enabling systematic verification against the results of [27].

Zeroth-order in Λ The most complicated part of the total bracket is the contribution at zeroth-order in the cosmological constant Λ, which contains both zeroth and first-order terms in the Wilson coefficient α. We extract this contribution using the utility function defined above:

```
In[90]:= Expr = ExtractBracketAnatomy[CosmologicalConstantOrder -> {0},
 ↪ WilsonCoefficientOrder -> All];
```

The raw extracted bracket from In[90] is extremely lengthy and difficult to interpret directly; we do not display its output here. To make progress, we apply the **FindAlgebra** function from *Hamilcar* iteratively, each time specifying a different ansatz structure that we want the bracket to match.[6] The **FindAlgebra** function attempts to re-express the bracket through a combination of integration by parts (modifying boundary terms) and applications of Cayley–Hamilton-like identities in three dimensions. These algebraic manipulations are performed automatically; the user need only specify the desired structural form. We proceed through four progressively refined representations. First, we express the bracket in terms of first-order gradients of the shorthand tensorial momentum $\mathscr{K}^i_j$:

```
In[91]:= Expr = ExtractBracketAnatomy[CosmologicalConstantOrder -> {0},
 ↪ WilsonCoefficientOrder -> All]; Expr //= FindAlgebra[#1, {{{CD, ScriptK},
 ↪ {CD, ScalarSmearingF, ScalarSmearingS}}, {{CD, ScriptK}, {RicciCD,
 ↪ RicciCD}, {CD, ScalarSmearingF, ScalarSmearingS}}, {{CD, ScriptK}, {CD,
 ↪ ScriptK}, {CD, ScriptK}, {CD, ScalarSmearingF, ScalarSmearingS}}, {{CD,
 ↪ ScriptK}, {RicciCD, ScriptK, ScriptK}, {CD, ScalarSmearingF,
 ↪ ScalarSmearingS}}, {{CD, ScriptK}, {ScriptK, ScriptK, ScriptK, ScriptK},
 ↪ {CD, ScalarSmearingF, ScalarSmearingS}}}, Method -> Solve, Verify ->
 ↪ False] & ;
```

[6] It is not necessary that we do this, and we show all the possible configurations to indicate the power and flexibility of **FindAlgebra**. One could skip ahead to the most compact form given in Out[94].

```
Out[91]= (-72 κ^2 α R[∇]^bc R[∇] s ∇_a 𝒦_bc ∇^a f + 24 κ^2 α R[∇]^bc s 𝒦_de 𝒦^de ∇_a 𝒦_bc ∇^a f -
          24 κ^2 α R[∇]^bc s 𝒦^d_d 𝒦^e_e ∇_a 𝒦_bc ∇^a f + (2 s ∇_a 𝒦^b_b ∇^a f)/κ^2 + 96 κ^2 α R[∇]_b^d R[∇]^bc s ∇_a 𝒦_cd ∇^a f +
          72 κ^2 α R[∇] s 𝒦_b^d 𝒦^bc ∇_a 𝒦_cd ∇^a f - 72 κ^2 α R[∇] s 𝒦^b_b 𝒦^cd ∇_a 𝒦_cd ∇^a f + 96 κ^2 α R[∇]^bc s 𝒦_b^d 𝒦^e_e ∇_a 𝒦_cd ∇^a f -
          96 κ^2 α R[∇]^bc s 𝒦_b^d 𝒦_d^e ∇_a 𝒦_ce ∇^a f - 96 κ^2 α s 𝒦^b_b 𝒦_c^e 𝒦^cd 𝒦_d^f ∇_a 𝒦_ef ∇^a f) + [ 142 similar terms ]
```

While the output from `In[91]` (shown truncated in `Out[91]`) confirms that the bracket can be written using only first derivatives (rather than higher-order gradients), the result remains too complex for direct interpretation. Next, we express the bracket in terms of the shorthand momentum gradient tensor $\mathscr{L}^i{}_{jk}$ and the spatial Ricci curvature $\mathcal{R}_{ij}$:

```
In[92]:= Expr = ExtractBracketAnatomy[CosmologicalConstantOrder -> {0},
 ↪ WilsonCoefficientOrder -> All]; Expr //= FindAlgebra[#1, {{{ScriptL},
 ↪ {CD, ScalarSmearingF, ScalarSmearingS}}, {{ScriptL, ScriptL, ScriptL},
 ↪ {CD, ScalarSmearingF, ScalarSmearingS}}, {{ScriptL, RicciCD, ScriptK,
 ↪ ScriptK}, {CD, ScalarSmearingF, ScalarSmearingS}}, {{ScriptL, ScriptK,
 ↪ ScriptK, ScriptK}, {CD, ScalarSmearingF, ScalarSmearingS}},
 ↪ {{ScriptL, RicciCD, RicciCD}, {CD, ScalarSmearingF, ScalarSmearingS}}},
 ↪ Method -> Solve, Verify -> False] & ;
```

```
Out[92]= (192 α R[∇]^bc s 𝒦_b^d 𝒦^e_e ℒ_acd ∇^a f + 72 α R[∇]^bc R[∇] s ℒ_bac ∇^a f -
          72 α R[∇]^bc s 𝒦_de 𝒦^de ℒ_bac ∇^a f + 72 α R[∇]^bc s 𝒦^d_d 𝒦^e_e ℒ_bac ∇^a f + 96 α R[∇]^bc s 𝒦_a^d 𝒦^e_e ℒ_bcd ∇^a f -
          96 α R[∇]^bc s 𝒦_a^d 𝒦_d^e ℒ_bce ∇^a f - 2 s ℒ^b_ab ∇^a f - 96 α R[∇]_b^d R[∇]^bc s ℒ_cad ∇^a f -
          72 α R[∇] s 𝒦_b^d 𝒦^bc ℒ_cad ∇^a f + 72 α R[∇] s 𝒦^b_b 𝒦^cd ℒ_cad ∇^a f) + [ 70 similar terms ]
```

The output from `In[92]` (shown truncated in `Out[92]`) is more compact, as it hides explicit gradients acting on non-smearing quantities, but further simplification is still possible. In the third refinement, we express the bracket entirely in terms of the shorthand functions $\mathscr{L}^i{}_{jk}$ and $\mathscr{Q}^{ij}{}_{kl}$:

```
In[93]:= Expr = ExtractBracketAnatomy[CosmologicalConstantOrder -> {0},
 ↪ WilsonCoefficientOrder -> All]; EinsteinConstant = 1;
 ↪ CosmologicalConstant = 1; WilsonCoefficient = 1; Expr //= FindAlgebra[#1,
 ↪ {{{ScriptL}, {CD, ScalarSmearingF, ScalarSmearingS}}, {{ScriptL, ScriptL,
 ↪ ScriptL}, {CD, ScalarSmearingF, ScalarSmearingS}}, {{ScriptL,
 ↪ ScriptQSingleContraction, ScriptQSingleContraction}, {CD,
 ↪ ScalarSmearingF, ScalarSmearingS}}}, Method -> Solve, Verify -> False,
 ↪ DDIs -> True] & ;
```

```
Out[93]= -2 s ℒ^b_ab ∇^a f - 96 s ℒ^b_a^c ℒ_dce ℒ^d_b^e ∇^a f + 96 s ℒ^b_a^c ℒ^d_bd ℒ^e_ce ∇^a f - 192 s ℒ^b_b^c 𝒬_a^d 𝒬_cd ∇^a f -
          48 s ℒ^b_ab 𝒬_cd 𝒬^cd ∇^a f + 96 s ℒ^b_b^c 𝒬_ac 𝒬^d_d ∇^a f - 24 s ℒ^b_a^c 𝒬_bc 𝒬^d_d ∇^a f + 48 s ℒ^b_ab 𝒬^c_c 𝒬^d_d ∇^a f +
          2 f ℒ^b_ab ∇^a s + 96 f ℒ^b_a^c ℒ_dce ℒ^d_b^e ∇^a s - 96 f ℒ^b_a^c ℒ^d_bd ℒ^e_ce ∇^a s + 192 f ℒ^b_b^c 𝒬_a^d 𝒬_cd ∇^a s +
          48 f ℒ^b_ab 𝒬_cd 𝒬^cd ∇^a s - 96 f ℒ^b_b^c 𝒬_ac 𝒬^d_d ∇^a s + 24 f ℒ^b_a^c 𝒬_bc 𝒬^d_d ∇^a s - 48 f ℒ^b_ab 𝒬^c_c 𝒬^d_d ∇^a s
```

The result `Out[93]` provides a more geometric representation, though it still does not directly reveal the constraint algebra structure. Finally, we express the bracket in terms of the reduced momentum constraint $\mathbb{H}_i$ and the zeroth-order reduced Hamiltonian constraint $\overset{(0)}{\mathbb{H}}$. Since $\overset{(0)}{\mathbb{H}}$ itself contains a term linear in Λ, and we are working at zeroth-order in Λ, we

must wrap this computation in a **Block** command that temporarily sets $\Lambda = 0$. This prevents **FindAlgebra** from attempting to expand terms that cannot appear at this order. Additionally, we use the **Constraints** option to instruct **FindAlgebra** to factor the result in terms of these constraint functions:

```
In[94]:= Expr = ExtractBracketAnatomy[CosmologicalConstantOrder -> {0},
 ↪ WilsonCoefficientOrder -> All]; Block[{CosmologicalConstant = 0}, Expr
 ↪ //= FindAlgebra[#1, {{{ReducedMomentumConstraint}, {CD, ScalarSmearingF,
 ↪ ScalarSmearingS}}, {{ReducedMomentumConstraint, ScriptL, ScriptL}, {CD,
 ↪ ScalarSmearingF, ScalarSmearingS}}, {{ReducedMomentumConstraint,
 ↪ ScriptQSingleContraction, ScriptQSingleContraction}, {CD,
 ↪ ScalarSmearingF, ScalarSmearingS}}, {{ReducedHamiltonianOrderUnity,
 ↪ ScriptQSingleContraction, ScriptL}, {CD, ScalarSmearingF,
 ↪ ScalarSmearingS}}}, Constraints -> {ReducedMomentumConstraint[i],
 ↪ ReducedHamiltonianOrderUnity[]}, Method -> Solve, Verify -> False] & ; ];
```

Out[94]=

$$-24\,\alpha\,\overset{(0)}{\mathcal{H}}_{\text{red}}\,(4\,\mathcal{L}^{b\ c}_{\ b\ }\,\mathcal{Q}_{ac} - \mathcal{L}^{b\ c}_{\ a\ }\,\mathcal{Q}_{bc} + 2\,\mathcal{L}^{b}_{\ ab}\,\mathcal{Q}^{c}_{\ c})\,(s\,\nabla^{a}f - f\,\nabla^{a}s) + \mathcal{H}_{\text{red}}{}^{i}\,(-48\,\alpha\,s\,\mathcal{L}_{bac}\,\mathcal{L}_{i}^{\ bc}\,\nabla^{a}f + 96\,\alpha\,s\,\mathcal{Q}_{ab}\,\mathcal{Q}_{i}^{\ b}\,\nabla^{a}f + 48\,\alpha\,f\,\mathcal{L}_{bac}\,\mathcal{L}_{i}^{\ bc}\,\nabla^{a}s - 96\,\alpha\,f\,\mathcal{Q}_{ab}\,\mathcal{Q}_{i}^{\ b}\,\nabla^{a}s + 24\,\alpha\,\mathcal{L}^{b}_{\ ib}\,\mathcal{L}^{c}_{\ ac}\,(s\,\nabla^{a}f - f\,\nabla^{a}s) + 48\,\alpha\,\mathcal{L}^{b\ c}_{\ i\ }\,\mathcal{L}_{cab}\,(-s\,\nabla^{a}f + f\,\nabla^{a}s) - s\,\nabla_{i}f - 24\,\alpha\,s\,\mathcal{L}^{abc}\,\mathcal{L}_{bac}\,\nabla_{i}f - 24\,\alpha\,s\,\mathcal{Q}_{ab}\,\mathcal{Q}^{ab}\,\nabla_{i}f + f\,(1 + 24\,\alpha\,(\mathcal{L}^{abc}\,\mathcal{L}_{bac} + \mathcal{Q}_{ab}\,\mathcal{Q}^{ab}))\,\nabla_{i}s)$$

The result **Out[94]** reveals the structure of the constraint algebra at zeroth-order in Λ and facilitates direct comparison with [27].

First-order in Λ We next analyze the contribution at first-order in the cosmological constant Λ. We extract this contribution using the same utility function from **In[89]**:

```
In[95]:= Expr = ExtractBracketAnatomy[CosmologicalConstantOrder -> {1},
 ↪ WilsonCoefficientOrder -> All];
```

The raw extracted bracket from **In[95]** is again too lengthy to display. Unlike the zeroth-order case, the first-order contribution cannot be cleanly expressed in terms of the reduced constraints alone, because the reduced Hamiltonian constraint itself contains both zeroth-order and first-order parts in Λ. Instead, we express the bracket directly in terms of the shorthand variables $\mathscr{L}^{i}_{\ jk}$ and $\mathscr{Q}^{ij}_{\ \ kl}$, along with the spatial curvature:

```
In[96]:= Expr = ExtractBracketAnatomy[CosmologicalConstantOrder -> {1},
 ↪ WilsonCoefficientOrder -> All]; Expr //= FindAlgebra[#1, {{{ScriptL},
 ↪ {CD, ScalarSmearingF, ScalarSmearingS}}, {{ScriptL,
 ↪ ScriptQSingleContraction}, {CD, ScalarSmearingF, ScalarSmearingS}}, {{CD,
 ↪ CD, ScriptL}, {CD, ScalarSmearingF, ScalarSmearingS}}, {{RicciCD,
 ↪ ScriptL}, {CD, ScalarSmearingF, ScalarSmearingS}}, {{RicciScalarCD,
 ↪ ScriptL}, {CD, ScalarSmearingF, ScalarSmearingS}}}, Method -> Solve,
 ↪ Verify -> True, DDIs -> True] & ;
```

Out[96]=

$$96\,s\,\mathcal{L}^{b\ c}_{\ b\ }\,\mathcal{Q}_{ac}\,\nabla^{a}f + 48\,s\,\mathcal{L}^{b\ c}_{\ a\ }\,\mathcal{Q}_{bc}\,\nabla^{a}f - 72\,s\,\mathcal{L}^{b}_{\ ab}\,\mathcal{Q}^{c}_{\ c}\,\nabla^{a}f - 96\,f\,\mathcal{L}^{b\ c}_{\ b\ }\,\mathcal{Q}_{ac}\,\nabla^{a}s - 48\,f\,\mathcal{L}^{b\ c}_{\ a\ }\,\mathcal{Q}_{bc}\,\nabla^{a}s + 72\,f\,\mathcal{L}^{b}_{\ ab}\,\mathcal{Q}^{c}_{\ c}\,\nabla^{a}s$$

The result Out[96] provides the first-order contribution in a form suitable for comparison with [27].

Second-order in Λ Finally, we analyze the contribution at second-order in the cosmological constant Λ. We extract this contribution with In[89]:

```
In[97]:= Expr = ExtractBracketAnatomy[CosmologicalConstantOrder -> {2},
 ↪ WilsonCoefficientOrder -> All];
```

```
Out[97]= 48 Λ^2 α s ∇^a f ∇_b π_a^b - 48 Λ^2 α f ∇^a s ∇_b π_a^b
```

The extracted bracket Out[97] is remarkably simple compared to the lower-order contributions. With a single application of FindAlgebra, we express it in terms of the reduced momentum constraint and the zeroth-order reduced Hamiltonian constraint, using the Constraints option to factor the result:

```
In[98]:= Expr = ExtractBracketAnatomy[CosmologicalConstantOrder -> {2},
 ↪ WilsonCoefficientOrder -> All]; Expr //= FindAlgebra[#1,
 ↪ {{{ReducedMomentumConstraint}, {CD, ScalarSmearingF, ScalarSmearingS}},
 ↪ {{ScriptL, ReducedHamiltonianOrderUnity}, {CD, ScalarSmearingF,
 ↪ ScalarSmearingS}}, {{ScriptQSingleContraction,
 ↪ ReducedMomentumConstraint}, {CD, ScalarSmearingF, ScalarSmearingS}},
 ↪ {{CD, CD, ScriptL}, {CD, ScalarSmearingF, ScalarSmearingS}}, {{RicciCD,
 ↪ ScriptL}, {CD, ScalarSmearingF, ScalarSmearingS}}}, Constraints ->
 ↪ {ReducedMomentumConstraint[i], ReducedHamiltonianOrderUnity[]}, Method ->
 ↪ Solve, Verify -> True] & ;
```

```
Out[98]= 24 Λ^2 α ℋ_red^i (-s ∇_i f + f ∇_i s)
```

The result Out[98] consists of a single term proportional to the reduced momentum constraint, demonstrating the simplified structure at second-order in Λ.

Reduced super-Hamiltonian auto-commutator Having computed the constraint algebra at each order in Λ, we now assemble the complete auto-commutator of the reduced super-Hamiltonian constraint to first order in α by combining Out[94], Out[96], and Out[98]. According to [27], this should take the form:

$$
\begin{aligned}
\{\mathbb{H}[f],\mathbb{H}[s]\} = {}& \mathbb{H}_i\big[f\nabla^i s - s\nabla^i f\big] + 24\alpha\kappa^4\mathbb{H}_i\Big[\mathscr{L}_k{}^{li}\mathscr{L}_l{}^{kj}\big(f\nabla_j s - s\nabla_j f\big)\Big] \\
& - 24\alpha\kappa^4\mathbb{H}_i\Big[\big(4\mathscr{Q}^i{}_j\mathscr{Q}^j{}_k - 2(\mathscr{Q}+\Lambda)\mathscr{Q}^i{}_k + \mathscr{L}^i_{jk}\mathscr{L}^j\big)\big(f\nabla^k s - s\nabla^k f\big)\Big] \\
& + 24\alpha\kappa^4\mathbb{H}_i\Big[\big(2\mathscr{Q}^j{}_{[k}\mathscr{Q}^k{}_{j]} + 2\Lambda\mathscr{Q} + \mathscr{L}^j_{kl}\mathscr{L}_j{}^{kl} - \mathscr{L}^j\mathscr{L}_j\big)\big(f\nabla^i s - s\nabla^i f\big)\Big] \\
& + 24\alpha\kappa^4\mathbb{H}\Big[\big(\mathscr{Q}^i{}_j\mathscr{L}^j_{ik} - \Lambda\mathscr{L}_k\big)\big(f\nabla^k s - s\nabla^k f\big)\Big] + \mathcal{O}(\alpha^2)\,. \qquad (2.64)
\end{aligned}
$$

It is easy to confirm using eqs. (2.59a), (2.59b), (2.62a) and (2.62b) that our computed results match eq. (2.64) perfectly. This concludes our discussion of the *Hamilcar* suite of tools.

3 The *Hasdrubal* agent

3.1 Design of the agent

Overview The *Hasdrubal* agent (see full sources in the supplement [32]) is a rudimentary proof-of-concept, and so we will not provide installation and usage instructions analogous to those given for *Hamilcar* in section 2.1. The agent is built on *OpenAI*'s *Agents SDK*, which provides an `Agent` class (an LLM with instructions and tools) and a `Runner` that orchestrates the agentic loop: receiving user input, invoking tools as needed, and iterating until the model produces a final response. A model context protocol (MCP) server is spawned as a subprocess and connected via `MCPServerStdio`, which communicates over standard input/output; the agent automatically discovers available tools through the MCP protocol's `list_tools` mechanism. The overall workflow is illustrated in fig. 2.

MCP server A persistent *Wolfram* kernel session with *Hamilcar* pre-loaded avoids startup overhead and maintains state across tool calls. The kernel connection is managed via *Wolfram Research*'s official `wolframclient` *Python* library, which provides a `WolframLanguageSession` class for inter-process communication. The MCP server exposes a single tool `tool_WolframScript` which allows the agent to send arbitrary *Wolfram Language* code to the kernel. Results are apparently not improved by providing specialized tools for specific *Hamilcar* functions: the model demonstrates a preference and aptitude for free composition.

System prompt The system prompt is composed of three components:

- Behavioural and stylistic guidelines as natural language. In particular, the Dirac–Bergmann algorithm itself is never explained to the agent. Rather, the guidelines that turn out to require emphasis are those enforcing:
  - Step-by-step kernel interaction, suppressing compound *Hamilcar* function calls.
  - The need for enhanced reasoning for certain steps.
  - Recovery from errors and interpretation of kernel messages.
- The *Hamilcar* sources as *Wolfram Language*, together with a navigation tree.
- Two worked examples of obfuscated test models as *Wolfram Language*, with natural language commentary: Proca theory and Kalb–Ramond theory. The theories are deliberately simple, and have limited overlap in their canonical formulation with the tests considered in section 3.3.

Language model The underlying language model is *OpenAI*'s *GPT-5.2* (API identifier `gpt-5.2`), announced in December 2025. This variant supports adaptive reasoning and performant tool-calling, with a 400,000 token context window (of which the *Hasdrubal* system prompt is a generically small fraction). Model parameters are left as default values, except for `reasoning_effort` which is set to `medium`. Reasoning is found to greatly impact performance by eliminating confusion over the correct mathematical interpretation

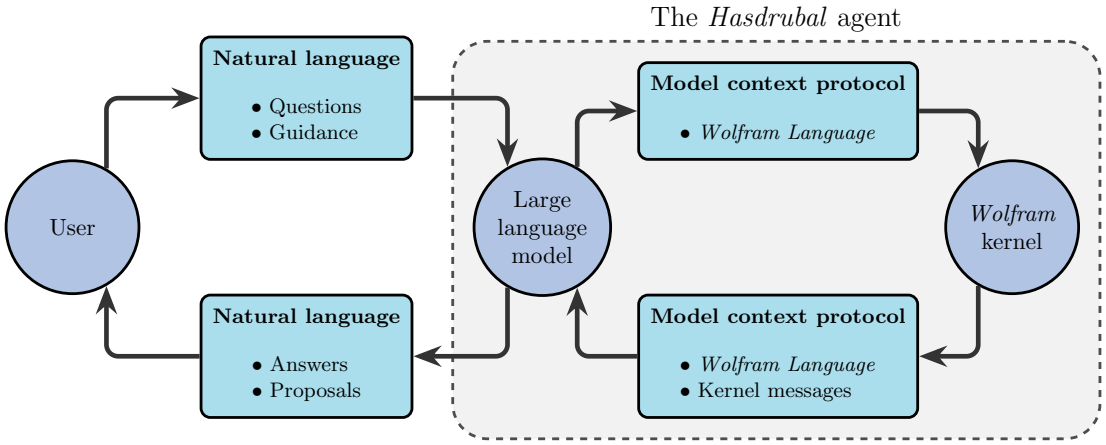


Figure 2. General workflow. The language model understands the Dirac–Bergmann algorithm, but cannot be relied upon to compute Poisson brackets. This problem is solved by allowing the model to interact with suitable tools.

of constraints and recovery from kernel messages. Loosely, this is consistent with the expectation that the Dirac–Bergmann algorithm requires non-trivial reasoning to apply correctly.

REPL handler In addition to the agentic loop native to the *Agents SDK* infrastructure, *Hasdrubal* is wrapped with an automated read-eval-print loop (REPL) handler that facilitates the tests shown in section 3.3. Whilst fig. 2 illustrates a generic user, the REPL handler monitors proposed tool calls, either encouraging continued evaluation or emphasizing the need for additional reasoning when certain conditions are met. The REPL also introduces the obfuscated theory at the start of the interaction, and requests the final summary of results at the end. The REPL handler completely removes the need for human intervention, allowing us to demonstrate that AI can indeed perform the Dirac–Bergmann algorithm from start to finish.

3.2 In-context learning

Overview The natural language portion of the system prompt requires tuning. It is easy to verify that the `gpt-5.2` model itself already understands the Dirac–Bergmann algorithm at a conceptual level: the algorithm and examples of its use feature widely in the pre-training corpus. Additional prompt engineering is, however, needed to ensure that the model applies the algorithm correctly in practice, and the correctness of the application must be verified through testing against a suite of theories for which the true physical results are known. To increase the number of examples, and so improve the quality of the prompt and gain confidence in the implementation, it is important that examples be created in an efficient, automated manner. The tuning of the system prompt is done manually, whenever the agent fails to produce correct results for any of the theories in the test suite. The prompt engineering pipeline is shown in fig. 3.

Fast verification of results To allow testing against a suite of canonical formulation problems, we focus exclusively on *linear* or *free* tensorial field theories on a Minkowski spacetime. For free theories, the results of the Dirac–Bergmann analysis — however protracted to implement — may be trivially obtained by independent and purely algorithmic means. The process of computing the number of propagating modes, as part of the particle spectrum (which additionally includes information about the masses and pole residues associated with the tree-level propagator), has a substantial history [33–47].[7] Recently, these kinds of analyses have been performed with computers [50–53], and for this paper we will employ the *PSALTer* software [54, 55]. In the context of this work, *PSALTer* is a tool, analogous to *Hamilcar*. It does not have any connection to an LLM, and implements a well-known algorithm whereby the tensor fields in the action are decomposed into their irreducible representations under the action of the massive little group (i.e., the rotation group SO(3)). This so-called *spin-projection* is amenable to a matrix formulation of the wave operator, from which pseudoinversion yields the propagator. The residues at the poles of the propagator then reveal the particle spectrum, including the number of propagating modes. For each of the linear theories proposed in the training catalogue, *PSALTer* is used to compute the *actual* number of propagating modes, which is then compared to the claim made by *Hasdrubal*.

Fast preparation of models The need for automatic generation of total Hamiltonia, analogous to those given already in eqs. (2.15), (2.32) and (2.61), is another reason to focus on free theories, since the REPL handler presents the total Hamiltonian to *Hasdrubal* as a means of posing the problem to be solved. In the free case, the total Hamiltonian can be computed automatically using a script based on *xAct*'s existing functionality. This script employs certain routines internal to the *PSALTer* codebase to break higher-rank fields into their irreducible components, by means of projection operators which are constructed from the Minkowski metric $g_{\mu\nu} = \eta_{\mu\nu}$ and the unit-timelike vector n^μ. In *PSALTer*, which uses a $(+,-,-,-)$ signature different from the $(-,+,+,+)$ signature of this work,[8] this unit-timelike vector has the property $n^\mu n_\mu \equiv 1$ and the interpretation that it be defined according to the massive particle four-momentum k^μ according to $n^\mu \equiv k^\mu/\sqrt{k^2}$, where $k^2 \equiv k^\mu k_\mu$. The projected irreps are naturally orthogonal to n^μ in all their indices, so that once the inverse metric $\eta^{\mu\nu} = -\delta^{\mu\nu} + n^\mu n^\nu$ is separated out, the unit-timelike vector appears only within the operator $n^\mu \partial_\mu$. Irreps upon which this operator acts are replaced by time derivatives with the same index structure. The canonical momenta are then defined as usual, by varying the resulting action with respect to the velocities. This results in a linear system of equations that may be programmatically inverted — as far as possible — to express the velocities in terms of the canonical momenta. The part of the system that does not admit any solution corresponds to the primary constraints. The total Hamiltonian is then constructed by means of a Legendre transformation, adding the primary constraints with Lagrange multipliers. Residual velocities may be substituted with their corresponding

[7] See additionally the general works [48, 49].

[8] In the following, expressions where this choice of convention affects the interpretation will be clearly indicated.

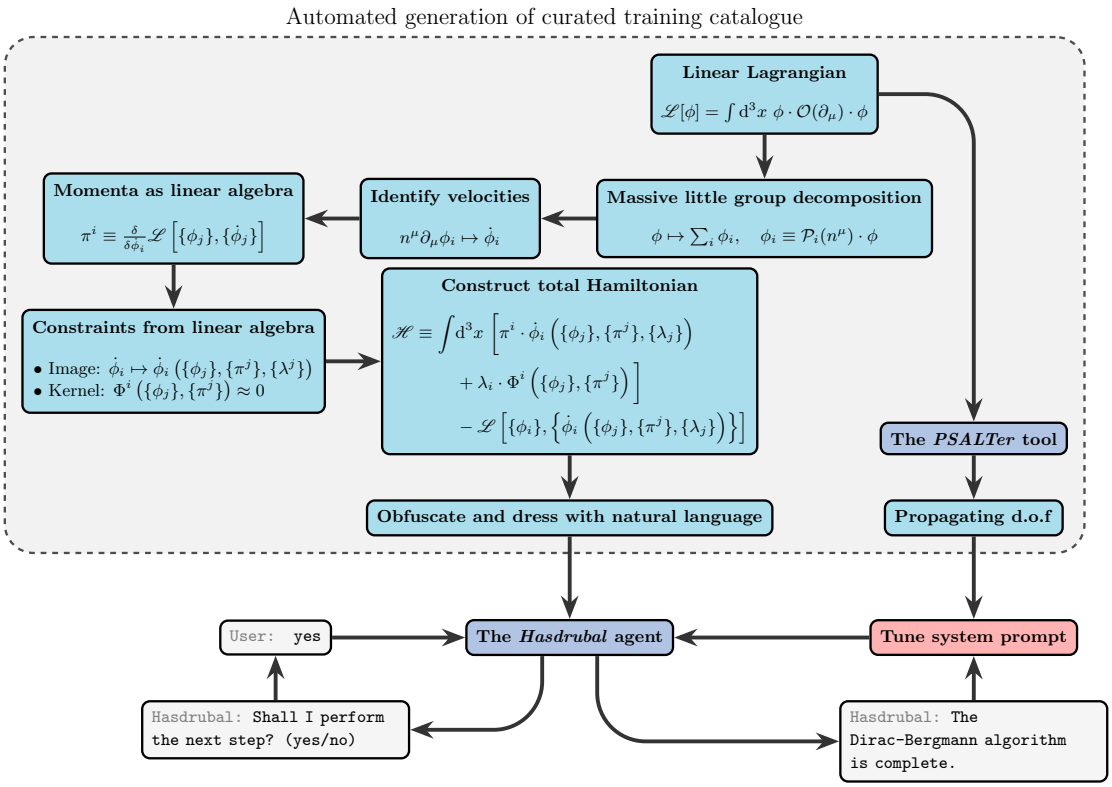


Figure 3. Pipeline for efficient prompt engineering, relying on fast construction of curated test-cases.

multipliers, which corresponds to a shift of the multiplier definitions that leaves the physics unchanged.

Model obfuscation Whilst the unavoidable presence of explicit examples in the pre-training corpus is doubtless beneficial for the capabilities of the system, it causes a problem for assessing the agent's flexibility and generalization. It is therefore important to remove explicit reference to specific theories from the prompt. To achieve this, the set of fields, conjugate momenta and coupling constants in each theory are systematically obfuscated by renaming them with hashes, and the problem is referred to anonymously in the natural language dressing. This approach at least places all the test theories on a maximally equal footing at inference time, though it is remarkable to observe that the model is still perfectly capable of recognizing familiar theories (such as Maxwell or Fierz–Pauli) from their canonical structure alone, even when obfuscated.

3.3 Samples from the training catalogue

3.3.1 Standard theories

Massive case We first consider a class of theories that is simple enough to plausibly have appeared in the training corpus of the LLM, specifically the vector theory in which there is a gauge invariance under transverse shifts of the vector. The general action for such a

Fundamental field	Symmetries	Decomposition into SO(3) irrep(s)	Source
$\mathcal{A}_\alpha$	(None)	$\mathcal{A}^{\#1}_{1^-\,\alpha} + \mathcal{A}^{\#1}_{0^+}\, n_\alpha$	$\mathcal{J}_\alpha$
SO(3) irrep	Symmetries	Expansion in terms of the fundamental field	Source SO(3) irrep
$\mathcal{A}^{\#1}_{0^+}$	(None)	$\mathcal{A}^\alpha\, n_\alpha$	$\mathcal{J}^{\#1}_{0^+}$
$\mathcal{A}^{\#1}_{1^-\,\alpha}$	(None)	$\mathcal{A}_\alpha - \mathcal{A}^\beta\, n_\alpha\, n_\beta$	$\mathcal{J}^{\#1}_{1^-\,\alpha}$

Table 2. Output generated by *PSALTer*, which assumes a $(+,-,-,-)$ signature different from the $(-,+,+,+)$ signature used throughout this work. The field kinematics of the vector field A_μ. These definitions are used in figs. 4 to 6. The unit-timelike vector is $n_\mu \equiv k_\mu/\sqrt{k^\nu k_\nu}$, where k_μ is the massive particle four-momentum.

theory is

$$\mathscr{S}[A_\mu] = \int \mathrm{d}^4x \left[-\frac{\alpha}{4}\partial_\mu A^\mu \partial_\nu A^\nu - \frac{\beta}{2} A_\mu A^\mu \right] . \tag{3.1}$$

The field kinematics of the vector in eq. (3.1) are shown in table 2. The spin-parity decomposition yields $A^{\#1}_{0^+}$ and $A^{\#1}_{1^-i}$, in the standard notation of [54, 55] where the subscript J^P describes the spin J and parity P of each mode, and the possibility of multiple modes with the same J^P is indicated by a superscript.[9] This notation extends naturally to conjugate momenta $\pi(A)^{\#1}_{0^+}$ and $\pi(A)^{\#1i}_{1^-}$, and Lagrange multipliers $\lambda(A)^{\#1}_{0^+}$ and $\lambda(A)^{\#1}_{1^-i}$. It is easy to see that the total Hamiltonian corresponding to eq. (3.1) is

$$\begin{aligned} \mathscr{H} = \int \mathrm{d}^3x \Bigg[& -\frac{\pi(A)^{\#1\,2}_{0^+}}{\alpha} + \frac{\beta}{2} A^{\#1\,2}_{0^+} - \frac{\beta}{2} A^{\#1}_{1^-i} A^{\#1i}_{1^-} \\ & + \pi(A)^{\#1}_{1^-i} \lambda(A)^{\#1i}_{1^-} + \pi(A)^{\#1}_{0^+} \partial_i A^{\#1i}_{1^-} \Bigg] , \end{aligned} \tag{3.2}$$

and that a single primary constraint arises as

$$\phi(A)^{\#1}_{1^-i} \equiv \pi(A)^{\#1}_{1^-i} \approx 0 . \tag{3.3}$$

Conservation of the primary constraint yields

$$\dot{\phi}(A)^{\#1}_{1^-i} \approx \frac{\delta}{\delta f i} \left\{ \phi(A)^{\#1j}_{1^-}[f_j], \mathscr{H} \right\} = \partial_i \pi(A)^{\#1}_{0^+} + \beta A^{\#1}_{1^-i} \approx 0 . \tag{3.4}$$

In the general case, this implies the secondary constraint

$$\chi(A)^{\#1}_{1^-i} \equiv \partial_i \pi(A)^{\#1}_{0^+} + \beta A^{\#1}_{1^-i} \approx 0 . \tag{3.5}$$

The primary and secondary constraints are found to be second class,

$$\left\{ \phi(A)^{\#1i}_{1^-}[f_i], \chi(A)^{\#1j}_{1^-}[s_j] \right\} = \beta[f_i s^i] , \tag{3.6}$$

[9] Note that the spin-parity modes may also carry Greek indices, but in such cases it is assumed that contraction of any Greek indices with n^μ yields zero; equivalently, n^μ defines the foliation.

which allows the multiplier $\lambda(A)^{\#1\,i}_{1^-}$ to be determined. The total number of second class constraints is $N_{\rm 2nd}=6$, while the total number of canonical variables is $N_{\rm Can}=8$. Since there are no first class constraints, the final number of propagating modes is found from eq. (1.1) to be

$$N_{\rm Phy}=\frac{1}{2}\Big(8-6\Big)=1\,. \tag{3.7}$$

Thus, the theory propagates a single massive scalar. This is confirmed by the *PSALTer* analysis, as shown in fig. 4. The total Hamiltonian defined in eq. (3.2) can be obfuscated as follows:

```
In[99]:= ( CouplingConstante5f53ecd * CanonicalFieldccb6bb11[]^2 ) / 2 -
 ↪ ConjugateMomentumCanonicalFieldccb6bb11[]^2 / CouplingConstant750dbd1c -
 ↪ ( CouplingConstante5f53ecd * CanonicalField1ea68330[-a] *
 ↪ CanonicalField1ea68330[a] ) / 2 +
 ↪ ConjugateMomentumCanonicalField1ea68330[-a] *
 ↪ LagrangeMultiplier1ea68330[a] + ConjugateMomentumCanonicalFieldccb6bb11[]
 ↪ * CD[-a][CanonicalField1ea68330[a]]
```

When `In[99]` is presented to *Hasdrubal* the response is:

> **Hasdrubal:** *This total Hamiltonian describes a theory on flat spatial slices with canonical variables* `CanonicalFieldccb6bb11[]`, `ConjugateMomentumCanonicalFieldccb6bb11[]` *and* `CanonicalField1ea68330[-a]`, `ConjugateMomentumCanonicalField1ea68330[a]`, *and with constant symbols* `CouplingConstant750dbd1c` *and* `CouplingConstante5f53ecd`. *The Dirac–Bergmann analysis yields the following constraints in terms of canonical variables. The primary constraint is* `ConjugateMomentumCanonicalField1ea68330[a] == 0`, *which is a spatial vector constraint and hence corresponds to 3 independent constrained degrees of freedom (one for each spatial component). Preservation in time of this primary constraint produces the secondary constraint* `CouplingConstante5f53ecd * CanonicalField1ea68330[a] + CD[a] @ConjugateMomentumCanonicalFieldccb6bb11[] == 0`, *also a spatial vector and hence likewise imposing 3 independent constrained degrees of freedom. No tertiary constraints arise.*
> *For generic nonzero* `CouplingConstante5f53ecd`, *these two vector constraints are second-class as a pair: their smeared Poisson bracket is nonvanishing and proportional to* `CouplingConstante5f53ecd`, *while the primary–primary and secondary–secondary brackets vanish. Consistency (time preservation) fixes the Lagrange multiplier* `LagrangeMultiplier1ea68330[-a]` *(all 3 of its components are determined by the consistency conditions). No other Lagrange multiplier is fixed by consistency; in particular,* `LagrangeMultiplierccb6bb11[]` *does not generate a constraint from the given total Hamiltonian and remains undetermined.*
> *The total number of canonical degrees of freedom is 2 times the number of configuration components, i.e. 2*(1+3)=8 from the scalar* `CanonicalFieldccb6bb11[]` *and the spatial vector* `CanonicalField1ea68330[-a]`. *With 6 second-class constrained degrees of freedom in total (3 primary + 3 secondary) and no first-class constraints, the number of physical degrees of freedom is (8-6)/2 = 1.*

Thus, *Hasdrubal* correctly reproduces the full chain of reasoning in eqs. (3.3) to (3.7).

Massless case In the case that $\beta=0$, eq. (3.4) instead implies the secondary constraint

$$\chi(A)^{\#1}_{0^+}\equiv\pi(A)^{\#1}_{0^+}\approx 0\,. \tag{3.8}$$

It is obvious from eq. (3.2) that the conservation in time of $\chi(A)^{\#1}_{0^+}$ is identically satisfied, and that $\phi(A)^{\#1}_{1^-i}$ and $\chi(A)^{\#1}_{0^+}$ commute and are first class. The total number of first class

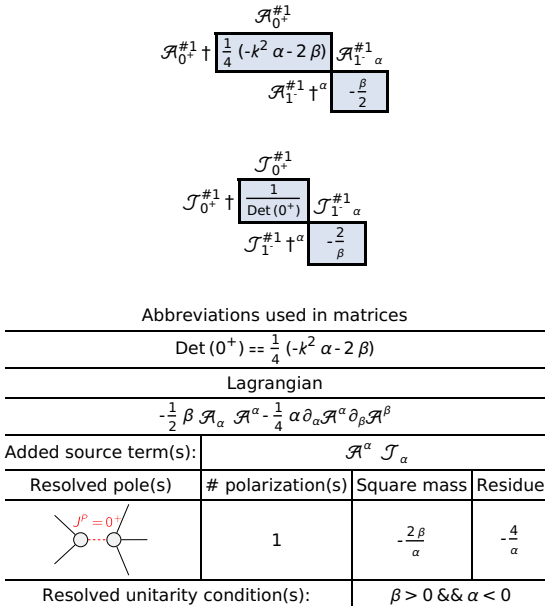


Figure 4. Output generated by *PSALTer*, which assumes a $(+,-,-,-)$ signature different from the $(-,+,+,+)$ signature used throughout this work. The spectrograph of the general theory in eq. (3.1). The theory propagates a massive scalar. The upper matrices show the structure of the wave operator in k-space, following the massive little-group decomposition; the lower matrices show the saturated propagator. All notation is defined in table 2.

constraints is $N_{\rm 1st} = 4$, while the total number of canonical variables is $N_{\rm Can} = 8$. Since there are no second class constraints, the final number of propagating modes is found from eq. (1.1) to be

$$N_{\rm Phy} = \frac{1}{2}\Big(8 - 2\times 4\Big) = 0\,. \tag{3.9}$$

This is a slightly surprising result, but the vanishing of all modes is due to the gauge symmetry which emerges in the massless limit, and which is generated by the primary first class constraint $\phi(A)^{\#1}_{1^-i}$. This is confirmed by the *PSALTer* analysis, as shown in fig. 5. The $\beta = 0$ case of the total Hamiltonian defined in eq. (3.2) can be obfuscated as follows:

```
In[100]:= - ( ConjugateMomentumCanonicalFieldccb6bb11[]^2 /
 ↪ CouplingConstant750dbd1c ) + ConjugateMomentumCanonicalField1ea68330[-a]
 ↪ * LagrangeMultiplier1ea68330[a] +
 ↪ ConjugateMomentumCanonicalFieldccb6bb11[] *
 ↪ CD[-a][CanonicalField1ea68330[a]]
```

When `In[100]` is presented to *Hasdrubal* the response is:

> **Hasdrubal:** *This total Hamiltonian describes a theory with canonical variables* `CanonicalFieldccb6bb11[]`, `ConjugateMomentumCanonicalFieldccb6bb11[]`, `CanonicalField1ea68330[-a]`, *and* `ConjugateMomentumCanonicalField1ea68330[a]` *(with coupling* `CouplingConstant750dbd1c`*) and with the multiplier sector including* `LagrangeMultiplierccb6bb11[]` *and* `LagrangeMultiplier1ea68330[-a]` *as provided. The Dirac–Bergmann analysis yields no primary*

constraint from `LagrangeMultiplierccb6bb11[]` *(its variational derivative of the total Hamiltonian density vanishes identically), while variation with respect to* `LagrangeMultiplier1ea68330[-a]` *yields the primary constraint* `ConjugateMomentumCanonicalField1ea68330[a] == 0`*, which is a rank-1 constraint with 3 independent components. Requiring time preservation of this primary constraint produces a secondary constraint* `CD[a] @ConjugateMomentumCanonicalFieldccb6bb11[] == 0`*; although written as a covector equation (3 components), it is a pure gradient of the scalar momentum* `ConjugateMomentumCanonicalFieldccb6bb11[]` *and therefore contributes 1 independent local constraint component (equivalently, it constrains* `ConjugateMomentumCanonicalFieldccb6bb11[]` *to vanish up to spatially constant modes, which are irrelevant for local degree-of-freedom counting). Preservation of the secondary constraint closes the algorithm with no tertiary constraints.*
All Poisson brackets among the constraints vanish (in particular, the bracket between `ConjugateMomentumCanonicalField1ea68330[a]` *and* `CD[b] @ ConjugateMomentumCanonicalFieldccb6bb11[]` *vanishes), so both constraints are first class. No consistency condition determines the time evolution of any Lagrange multiplier: neither* `LagrangeMultiplier1ea68330[-a]` *nor* `LagrangeMultiplierccb6bb11[]` *is fixed by constraint preservation, and both remain undetermined.*
The total number of canonical phase space degrees of freedom is `2 * ( 1 + 3 )= 8`*, corresponding to one scalar canonical pair* `( CanonicalFieldccb6bb11[] , ConjugateMomentumCanonicalFieldccb6bb11[] )` *and one spatial vector canonical pair* `( CanonicalField1ea68330[-a] , ConjugateMomentumCanonicalField1ea68330[a] )`*. The first-class constraints remove* `2 * 3` *degrees of freedom from* `ConjugateMomentumCanonicalField1ea68330[a] == 0` *plus* `2 * 1` *degrees of freedom from the independent content of* `CD[a] @ ConjugateMomentumCanonicalFieldccb6bb11[] == 0`*, for a total reduction of* `2 * ( 3 + 1 )= 8`*, leaving* `8 - 8 = 0` *physical phase space degrees of freedom, i.e. 0 physical configuration-space degrees of freedom.*

Thus, *Hasdrubal* correctly reproduces the full chain of reasoning in eqs. (3.8) to (3.9).

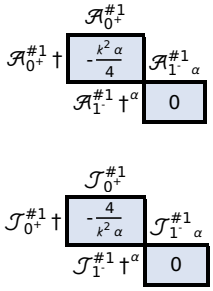
3.3.2 Unseen theories

Massive multi-particle theory We now move beyond the standard theories of section 3.3.1, to consider models which are very unlikely to have appeared in the training corpus of the LLM. We consider the theory

$$\begin{aligned}\mathscr{S}[\phi, A_\mu, B_\mu] = \int \mathrm{d}^4x \Big[&\alpha \partial_\mu A^\mu \partial_\nu A^\nu + \beta \partial_\mu B^\mu \partial_\nu B^\nu + \gamma \partial_\mu B_\nu \partial^\mu B^\nu \\ &+ \delta \partial_\mu \phi \partial^\mu \phi + \epsilon \partial_\mu A^\mu \phi + \zeta B_\mu B^\mu + \eta \phi^2 \Big] ,\end{aligned} \tag{3.10}$$

in which A_μ has been set up with the same transverse gauge invariance as in eq. (3.1), while B_μ admits the full kinetic structure of a vector field (i.e. no symmetry whatever). In addition to these vectors, a scalar field ϕ is also included in such a way as to preserve the gauge invariance of A_μ. The particle spectrum of eq. (3.10) is shown in fig. 6 — the trivial field kinematics of the scalar are shown in table 3, and the decomposition of the vector B_μ is analogous to that of A_μ in table 2. The theory is seen to have a fully massive spectrum.[10] We will not provide the full Dirac–Bergmann analysis here, but proceed to the obfuscated total Hamiltonian:

[10] Unitarity for this model is impossible without fine-tuning of the constants, however this is inessential.



Lagrangian		
$-\frac{1}{4}\,\alpha\,\partial_\alpha \mathcal{A}^\alpha\,\partial_\beta \mathcal{A}^\beta$		
Added source term(s):	$\mathcal{A}^\alpha\ \mathcal{J}_\alpha$	
Source constraint(s)	# constraint(s)	Covariant form
$\mathcal{J}_{1^-}^{\#1\,\alpha} == 0$	3	$\partial_\beta \partial^\alpha \mathcal{J}^\beta == \partial_\beta \partial^\beta \mathcal{J}^\alpha$
Total # constraint(s):	3	
Resolved unitarity condition(s):		True

Figure 5. Output generated by *PSALTer*, which assumes a $(+,-,-,-)$ signature different from the $(-,+,+,+)$ signature used throughout this work. The spectrograph of the theory in eq. (3.1) for the case $\beta = 0$. The spectrum is trivial — devoid of any particles — due to the transverse symmetry. In pology terms, the transverse symmetry is revealed by constraints on the source currents, in this case the transverse constraint has three independent components. All notation is defined in table 2.

```
In[101]:= ( CouplingConstantad6be3ca^2 * CanonicalFieldf25eb593[]^2 ) / ( 4 *
 ↪ CouplingConstant750dbd1c ) (*omitting 2553 characters for brevity*)
 ↪ CouplingConstantef18d0ee * CD[-b][CanonicalFieldfa1caa2d[-a]] *
 ↪ CD[b][CanonicalFieldfa1caa2d[a]]
```

When `In[101]` is presented to *Hasdrubal* the response is:

> **Hasdrubal:** *This total Hamiltonian describes a quadratic field theory on flat spatial slices with five canonical configuration variables, namely the scalars* `CanonicalFieldf25eb593[]`, `CanonicalFieldccb6bb11[]`, `CanonicalField8eca3bb0[]` *and the spatial covectors* `CanonicalField1ea68330[-a]`, `CanonicalFieldfa1caa2d[-a]`, *together with their conjugate momenta* `ConjugateMomentumCanonicalFieldf25eb593[]`, `ConjugateMomentumCanonicalFieldccb6bb11[]`, `ConjugateMomentumCanonicalField8eca3bb0[]`, `ConjugateMomentumCanonicalField1ea68330[a]`, *and* `ConjugateMomentumCanonicalFieldfa1caa2d[a]`, *and constant symbols* { `CouplingConstant750dbd1c` , `CouplingConstantad6be3ca` , `CouplingConstante5f53ecd` , `CouplingConstant83bce689` , `CouplingConstanta5eb6381` , `CouplingConstant6d9a09c9` , `CouplingConstantef18d0ee` } . *The only Lagrange multiplier entering the given Hamiltonian is* `LagrangeMultiplier1ea68330[a]`, *appearing linearly as* `ConjugateMomentumCanonicalField1ea68330[-a] * LagrangeMultiplier1ea68330[a]`.
> *The primary-constraint analysis by variational differentiation with respect to the listed multipliers yields a single nontrivial primary constraint from* `LagrangeMultiplier1ea68330[-a]`, *namely* `ConjugateMomentumCanonicalField1ea68330[a] == 0`, *which is a contravariant spatial vector and therefore corresponds to 3 independent constrained degrees of freedom (one per spatial component). All other multiplier variations vanish, i.e. there are no primary constraints associated with* `LagrangeMultiplierf25eb593[]`, `LagrangeMultiplierccb6bb11[]`, `LagrangeMultiplier8eca3bb0[]`, *or* `LagrangeMultiplierfa1caa2d[-a]` *for the Hamiltonian provided.*

Fundamental field	Symmetries	Decomposition into SO(3) irrep(s)	Source
ϕ	(None)	$\phi_{0^+}^{\#1}$	ρ
SO(3) irrep	Symmetries	Expansion in terms of the fundamental field	Source SO(3) irrep
$\phi_{0^+}^{\#1}$	(None)	ϕ	$\rho_{0^+}^{\#1}$

Table 3. Output generated by *PSALTer*, which assumes a $(+,-,-,-)$ signature different from the $(-,+,+,+)$ signature used throughout this work. The field kinematics of a scalar field. These definitions are used in fig. 6. The unit-timelike vector is defined in table 2.

> *Preservation in time of the primary constraint gives the condition* `CD[a] @ ConjugateMomentumCanonicalFieldccb6bb11[] == 0`. *Since this is a pure spatial gradient of the scalar* `ConjugateMomentumCanonicalFieldccb6bb11[]`, *the consistency condition is equivalent (modulo spatial zero-modes, as in the standard local Dirac analysis) to the scalar secondary constraint* `ConjugateMomentumCanonicalFieldccb6bb11[] == 0`, *corresponding to 1 independent constrained degree of freedom. The secondary constraint is preserved in time without generating further constraints.*
> *All Poisson brackets among the constraints vanish (in particular, the brackets between* `ConjugateMomentumCanonicalField1ea68330[a]` *and* `ConjugateMomentumCanonicalFieldccb6bb11[]`, *as well as each self-bracket, vanish), so both constraints are first-class: the 3-component vector constraint* `ConjugateMomentumCanonicalField1ea68330[a] == 0` *is first-class, and the 1-component scalar constraint* `ConjugateMomentumCanonicalFieldccb6bb11[] == 0` *is first-class. No consistency condition fixes the time-evolution of any Lagrange multiplier; in particular,* `LagrangeMultiplier1ea68330[a]` *remains undetermined by the Dirac-Bergmann algorithm for this Hamiltonian, and the remaining listed multipliers (which do not enter the Hamiltonian provided) are likewise undetermined.*
> *The configuration-space dimension is* `N = 3 ( from CanonicalFieldf25eb593[] , CanonicalFieldccb6bb11[] , CanonicalField8eca3bb0[] )+ 6 ( from CanonicalField1ea68330[-a]` and `CanonicalFieldfa1caa2d[-a] )= 9`, *so the phase-space dimension is* `2 N = 18`. *With* `F = 4` *first-class constraint components in total (3 from* `ConjugateMomentumCanonicalField1ea68330[a] == 0` *and 1 from* `ConjugateMomentumCanonicalFieldccb6bb11[] == 0`*) and* `S = 0` *second-class constraints, the number of physical degrees of freedom is* `( 1 / 2 )( 2 N - 2 F - S )= ( 1 / 2 )( 18 - 8 - 0 )= 5`.

Thus, *Hasdrubal* correctly reproduces results matching those of fig. 6.

Massless multi-gauge theory We now consider the theory

$$\begin{aligned}\mathscr{S}[\phi, A_\mu, B_\mu, C_\mu] = \int \mathrm{d}^4x \Big[& \alpha \partial_\mu A^\mu \partial_\nu A^\nu + \beta \partial_\mu B_\nu \partial^\mu B^\nu - \beta \partial_\mu B_\nu \partial^\nu B^\mu \\ & + \gamma \partial_\mu B_\nu \partial^\mu C^\nu - \gamma \partial_\mu B_\nu \partial^\nu C^\mu \\ & + \delta \partial_\mu C_\nu \partial^\mu C^\nu - \delta \partial_\mu C_\nu \partial^\nu C^\mu \\ & + \delta \partial_\mu \phi \partial^\mu \phi + \epsilon \partial_\mu A^\mu \phi + \zeta \phi^2 \Big] ,\end{aligned} \tag{3.11}$$

in which the A_μ field has the same role as in eq. (3.10), but the B_μ and C_μ fields are now both endowed with the full Maxwell kinetic structure, and are mixed. The scalar field ϕ is again included. The theory in eq. (3.11) has a richer gauge structure than that of eq. (3.10), spanning multiple fields. The particle spectrum of eq. (3.11) is shown in fig. 7. The theory is seen to have a massless spectrum in addition to a massive scalar sector, supported by the

	$\mathcal{A}_{0^+}^{\#1}$	$\phi_{0^+}^{\#1}$	$\mathcal{B}_{0^+}^{\#1}$
$\mathcal{A}_{0^+}^{\#1}\dagger$	$k^2\,\alpha$	$-\frac{1}{2}\,i\,k\,\epsilon$	0
$\phi_{0^+}^{\#1}\dagger$	$\frac{i\,k\,\epsilon}{2}$	$k^2\,\delta+\eta$	0
$\mathcal{B}_{0^+}^{\#1}\dagger$	0	0	$\Upsilon_1\,k^2+\zeta$

	$\mathcal{A}_{1^-\,\alpha}^{\#1}$	$\mathcal{B}_{1^-\,\alpha}^{\#1}$
$\mathcal{A}_{1^-}^{\#1}\dagger^{\alpha}$	0	0
$\mathcal{B}_{1^-}^{\#1}\dagger^{\alpha}$	0	$\zeta+k^2\,\gamma$

	$\mathcal{J}_{0^+}^{\#1}$	$\rho_{0^+}^{\#1}$	$\mathcal{K}_{0^+}^{\#1}$
$\mathcal{J}_{0^+}^{\#1}\dagger$	$\frac{\Upsilon_1\,k^4\,\delta+\zeta\,\eta+k^2\,(\zeta\,\delta+\Upsilon_1\,\eta)}{\text{Det}\,(0^+)}$	$\frac{i\,\Upsilon_1\,k^3\,\epsilon+i\,k\,\zeta\,\epsilon}{2\,\text{Det}\,(0^+)}$	0
$\rho_{0^+}^{\#1}\dagger$	$\frac{-i\,\Upsilon_1\,k^3\,\epsilon-i\,k\,\zeta\,\epsilon}{2\,\text{Det}\,(0^+)}$	$\frac{\Upsilon_1\,k^4\,\alpha+k^2\,\alpha\,\zeta}{\text{Det}\,(0^+)}$	0
$\mathcal{K}_{0^+}^{\#1}\dagger$	0	0	$\frac{4\,k^4\,\alpha\,\delta+k^2\,(-\epsilon^2+4\,\alpha\,\eta)}{4\,\text{Det}\,(0^+)}$

	$\mathcal{J}_{1^-\,\alpha}^{\#1}$	$\mathcal{K}_{1^-\,\alpha}^{\#1}$
$\mathcal{J}_{1^-}^{\#1}\dagger^{\alpha}$	0	0
$\mathcal{K}_{1^-}^{\#1}\dagger^{\alpha}$	0	$\frac{1}{\zeta+k^2\,\gamma}$

Abbreviations used in matrices

$$\Upsilon_1 == \beta+\gamma \;\&\&\; \text{Det}\,(0^+) == \frac{1}{4}\,k^2\,(-\epsilon^2+4\,\alpha\,(k^2\,\delta+\eta))\,(\zeta+k^2\,(\beta+\gamma))$$

Lagrangian

$$\eta\,\phi^2+\zeta\,\mathcal{B}_{\alpha}\;\mathcal{B}^{\alpha}+\epsilon\,\phi\,\partial_{\alpha}\mathcal{A}^{\alpha}+\delta\,\partial_{\alpha}\phi\,\partial^{\alpha}\phi+\gamma\,\partial_{\alpha}\mathcal{B}^{\beta}\,\partial^{\alpha}\mathcal{B}_{\beta}+\alpha\,\partial_{\alpha}\mathcal{A}^{\alpha}\,\partial_{\beta}\mathcal{A}^{\beta}+\beta\,\partial_{\alpha}\mathcal{B}^{\alpha}\,\partial_{\beta}\mathcal{B}^{\beta}$$

Added source term(s):	$\phi\,\rho+\mathcal{A}^{\alpha}\;\mathcal{J}_{\alpha}+\mathcal{B}^{\alpha}\;\mathcal{K}_{\alpha}$	
Source constraint(s)	# constraint(s)	Covariant form
$\mathcal{J}_{1^-}^{\#1\,\alpha} == 0$	3	$\partial_{\beta}\partial^{\alpha}\mathcal{J}^{\beta} == \partial_{\beta}\partial^{\beta}\mathcal{J}^{\alpha}$
Total # constraint(s):	3	

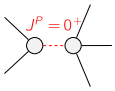
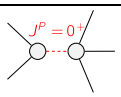
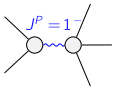
Resolved pole(s)	# polarization(s)	Square mass	Residue
	1	$\frac{\epsilon^2-4\,\alpha\,\eta}{4\,\alpha\,\delta}$	$\frac{\alpha\,\epsilon^2+\delta\,\epsilon^2-4\,\alpha^2\,\eta}{\alpha\,\delta\,\epsilon^2-4\,\alpha^2\,\delta\,\eta}$
	1	$-\frac{\zeta}{\beta+\gamma}$	$\frac{1}{\beta+\gamma}$
	3	$-\frac{\zeta}{\gamma}$	$-\frac{1}{\gamma}$
Resolved unitarity condition(s):	(Demonstrably impossible)		

Figure 6. Output generated by *PSALTer*, which assumes a $(+,-,-,-)$ signature different from the $(-,+,+,+)$ signature used throughout this work. The spectrograph of the combined theory. The spectrum is entirely massive. All notation is defined in tables 2 and 3 — the notation for B_μ is analogous to that for A_μ.

multiple gauge invariances.[11] Once again, we will rely entirely on *Hasdrubal* for the actual Hamiltonian structure. The obfuscated total Hamiltonian is:

```
In[102]:= ( CouplingConstantad6be3ca^2 * CanonicalFieldf25eb593[]^2 ) / ( 4 *
 ↪ CouplingConstant750dbd1c ) (*omitting 16470 characters for brevity*)
 ↪ CouplingConstant83bce689 * CD[-b][CanonicalFieldbc2ebc94[-a]] *
 ↪ CD[b][CanonicalFieldbc2ebc94[a]]
```

When `In[102]` is presented to *Hasdrubal* the response is:

> **Hasdrubal:** *This total Hamiltonian describes a theory on flat spatial slices with seven canonical configuration variables, namely the scalars* `CanonicalFieldf25eb593[]`, `CanonicalFieldccb6bb11[]`, `CanonicalField8eca3bb0[]`, `CanonicalFieldc8d16b6b[]` *and the spatial covectors* `CanonicalField1ea68330[-a]`, `CanonicalFieldfa1caa2d[-a]`, `CanonicalFieldbc2ebc94[-a]`, *together with their conjugate momenta* `ConjugateMomentumCanonicalFieldf25eb593[]`, `ConjugateMomentumCanonicalFieldccb6bb11[]`, `ConjugateMomentumCanonicalField8eca3bb0[]`, `ConjugateMomentumCanonicalFieldc8d16b6b[]` *and* `ConjugateMomentumCanonicalField1ea68330[a]`, `ConjugateMomentumCanonicalFieldfa1caa2d[a]`, `ConjugateMomentumCanonicalFieldbc2ebc94[a]`. *The constant symbols entering the Hamiltonian are* `CouplingConstant750dbd1c`, `CouplingConstantad6be3ca`, `CouplingConstante5f53ecd`, `CouplingConstant83bce689`, `CouplingConstanta5eb6381`, *and* `CouplingConstantef18d0ee`.
>
> *The Dirac–Bergmann analysis yields three primary constraints. The first is the vector constraint* `ConjugateMomentumCanonicalField1ea68330[a] == 0`, *which has 3 independent components. The second is the scalar constraint* `ConjugateMomentumCanonicalField8eca3bb0[] == 0`, *which has 1 independent component. The third is the scalar constraint* `ConjugateMomentumCanonicalFieldc8d16b6b[] == 0`, *which has 1 independent component. Preservation in time of* `ConjugateMomentumCanonicalField1ea68330[a] == 0` *yields a secondary constraint via a spatial gradient, which reduces to the scalar constraint* `ConjugateMomentumCanonicalFieldccb6bb11[] == 0` *with 1 independent component. Preservation in time of* `ConjugateMomentumCanonicalField8eca3bb0[] == 0` *yields the secondary scalar constraint* `CD[-a] @ConjugateMomentumCanonicalFieldfa1caa2d[a] == 0` *with 1 independent component. Preservation in time of* `ConjugateMomentumCanonicalFieldc8d16b6b[] == 0` *yields the secondary scalar constraint* `CD[-a] @ConjugateMomentumCanonicalFieldbc2ebc94[a] == 0` *with 1 independent component. No tertiary constraints arise: the time evolution of* `ConjugateMomentumCanonicalFieldccb6bb11[]`, `CD[-a] @ConjugateMomentumCanonicalFieldfa1caa2d[a]`, *and* `CD[-a] @ConjugateMomentumCanonicalFieldbc2ebc94[a]` *vanishes on the constraint surface.*
>
> *All six constraints listed above are first-class: all computed mutual Poisson brackets between smeared forms of these constraints vanish. No second-class constraints occur. Consequently, no Lagrange multiplier is fixed by consistency conditions. In particular, the multipliers that explicitly enforce primary constraints in the given total Hamiltonian, namely* `LagrangeMultiplier1ea68330[-a]`, `LagrangeMultiplier8eca3bb0[]`, *and* `LagrangeMultiplierc8d16b6b[]`, *remain undetermined by the Dirac–Bergmann algorithm. The remaining multiplier symbols provided in the model input,* `LagrangeMultiplierf25eb593[]`, `LagrangeMultiplierccb6bb11[]`, `LagrangeMultiplierfa1caa2d[-a]`, *and* `LagrangeMultiplierbc2ebc94[-a]`, *do not generate primary constraints for this Hamiltonian and are likewise not determined by consistency conditions.*
>
> *The physical degree-of-freedom count is as follows. The configuration space has 13 independent components (4 scalars plus 3 vectors, i.e.* `4 + 3 + 3 + 3 = 13`*), so the phase space has dimension* `2N = 26`. *There are* `F = 8` *independent first-class constraints in total (3 from* `ConjugateMomentumCanonicalField1ea68330[a] == 0`, *and 1 each from* `ConjugateMomentumCanonicalField8eca3bb0[] == 0`, `ConjugateMomentumCanonicalFieldc8d16b6b[] == 0`, `ConjugateMomentumCanonicalFieldccb6bb11[] == 0`, `CD[-a] @ConjugateMomentumCanonicalFieldfa1caa2d[a] == 0`, *and* `CD[-a] @`

[11] Unitarity for this model is again impossible without fine-tuning of the constants.

> `ConjugateMomentumCanonicalFieldbc2ebc94[a] == 0`*), and* `S = 0` *second-class constraints. Therefore the number of physical degrees of freedom is* `( 2N - 2F - S )/ 2 = ( 26 - 16 - 0 )/ 2 = 5`.

Thus, *Hasdrubal* correctly navigates multiple constraint chains to reproduce results matching those of fig. 7.

4 Conclusions

In this paper we have presented:

- *Hamilcar* — a *Wolfram Language* package for computing and simplifying Poisson brackets in canonical field theory. This package may plausibly be of use beyond the context of this paper, and is intended for long-term maintenance.
- *Hasdrubal* — a rudimentary agent built using *OpenAI*'s *Agents SDK* which demonstrates that *Hamilcar* may be used autonomously by artificial intelligence (AI). This agent is a proof-of-concept: it is anticipated that more sophisticated agents may be built in the future.

The *Hamilcar* package was shown to be effective against highly non-trivial operations in the canonical formulation of gravity. This includes the automated reconstruction of closed-form constraint algebroids through integration-by-parts manipulations, and dimensionally-dependent identities such as the Cayley–Hamilton theorem. Many of the techniques used by *Hamilcar* have been available for some years, though an effective implementation was lacking. It is expected that *Hamilcar* will be useful for the efficient order-reduction of those modified gravity theories, and theories beyond the standard model of particle physics, which present as truncated effective theories. The expected outputs in such cases are quantum-corrected Hamiltonia which are suitable for phenomenological study.

The *Hasdrubal* agent confirmed that commercially available large language models (LLMs) are capable of autonomously implementing the multi-step Dirac–Bergmann algorithm from start to finish, with the assistance of tool calls to *Hamilcar*. The agent was shown to be capable of finding the correct number of propagating modes in unseen linear field theories, including cases with multiple gauge symmetries and fields. Whilst free theories are often thought of as 'simple', their gauge structure can be highly non-trivial when multiple non-scalar fields are mixed: the state of the art for determining the particle spectra of such theories is still an active area of research [54, 55]. The ability of the agent is remarkable given the reliance on in-context learning alone. The agent requires no domain-specific fine-tuning, reinforcement learning, or gradient-based training. All domain knowledge is supplied either symbolically (via *Hamilcar*) or procedurally (via in-context demonstrations). Further work may focus on more sophisticated agent or multi-agent design, or on the use of fine-tuning to improve performance. In this paper, the importance of automating the generation of training examples was emphasized: these techniques will be crucial for future work.

	$\phi_{0^+}^{\#1}$	$\mathcal{A}_{0^+}^{\#1}$	$\mathcal{B}_{0^+}^{\#1}$	$C_{0^+}^{\#1}$
$\phi_{0^+}^{\#1}\,\dagger$	$\zeta + k^2\,\delta$	$\frac{i\,k\,\epsilon}{2}$	0	0
$\mathcal{A}_{0^+}^{\#1}\,\dagger$	$-\frac{1}{2}\,i\,k\,\epsilon$	$k^2\,\alpha$	0	0
$\mathcal{B}_{0^+}^{\#1}\,\dagger$	0	0	0	0
$C_{0^+}^{\#1}\,\dagger$	0	0	0	0

	$\mathcal{A}_{1^-\,\alpha}^{\#1}$	$\mathcal{B}_{1^-\,\alpha}^{\#1}$	$C_{1^-\,\alpha}^{\#1}$
$\mathcal{A}_{1^-}^{\#1}\,\dagger^{\alpha}$	0	0	0
$\mathcal{B}_{1^-}^{\#1}\,\dagger^{\alpha}$	0	$k^2\,\beta$	$\frac{k^2\,\gamma}{2}$
$C_{1^-}^{\#1}\,\dagger^{\alpha}$	0	$\frac{k^2\,\gamma}{2}$	$k^2\,\delta$

	$\rho_{0^+}^{\#1}$	$\mathcal{J}_{0^+}^{\#1}$	$\mathcal{K}_{0^+}^{\#1}$	$\mathcal{L}_{0^+}^{\#1}$
$\rho_{0^+}^{\#1}\,\dagger$	$\frac{k^2\,\alpha}{\text{Det}\,(0^+)}$	$-\frac{i\,k\,\epsilon}{2\,\text{Det}\,(0^+)}$	0	0
$\mathcal{J}_{0^+}^{\#1}\,\dagger$	$\frac{i\,k\,\epsilon}{2\,\text{Det}\,(0^+)}$	$\frac{\zeta + k^2\,\delta}{\text{Det}\,(0^+)}$	0	0
$\mathcal{K}_{0^+}^{\#1}\,\dagger$	0	0	0	0
$\mathcal{L}_{0^+}^{\#1}\,\dagger$	0	0	0	0

	$\mathcal{J}_{1^-\,\alpha}^{\#1}$	$\mathcal{K}_{1^-\,\alpha}^{\#1}$	$\mathcal{L}_{1^-\,\alpha}^{\#1}$
$\mathcal{J}_{1^-}^{\#1}\,\dagger^{\alpha}$	0	0	0
$\mathcal{K}_{1^-}^{\#1}\,\dagger^{\alpha}$	0	$\frac{k^2\,\delta}{\text{Det}\,(1^-)}$	$-\frac{k^2\,\gamma}{2\,\text{Det}\,(1^-)}$
$\mathcal{L}_{1^-}^{\#1}\,\dagger^{\alpha}$	0	$-\frac{k^2\,\gamma}{2\,\text{Det}\,(1^-)}$	$\frac{k^2\,\beta}{\text{Det}\,(1^-)}$

Abbreviations used in matrices

$$\text{Det}\,(0^+) == \frac{1}{4}\,k^2\,(-\epsilon^2 + 4\,\alpha\,(\zeta + k^2\,\delta))\;\&\&\;\text{Det}\,(1^-) == \frac{1}{4}\,k^4\,(4\,\delta\,\beta - \gamma^2)$$

Lagrangian

$$\zeta\,\phi^2 + \epsilon\,\phi\,\partial_\alpha\mathcal{A}^\alpha + \delta\,\partial_\alpha\phi\,\partial^\alpha\phi + \beta\,\partial_\alpha\mathcal{B}_\beta\,\partial^\alpha\mathcal{B}^\beta + \gamma\,\partial_\alpha\mathcal{B}_\beta\,\partial^\alpha C^\beta +$$
$$\delta\,\partial_\alpha C_\beta\,\partial^\alpha C^\beta + \alpha\,\partial_\alpha\mathcal{A}^\alpha\,\partial_\beta\mathcal{A}^\beta - \beta\,\partial_\alpha\mathcal{B}_\beta\,\partial^\beta\mathcal{B}^\alpha - \gamma\,\partial_\alpha\mathcal{B}_\beta\,\partial^\beta C^\alpha - \delta\,\partial_\alpha C_\beta\,\partial^\beta C^\alpha$$

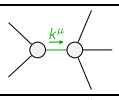
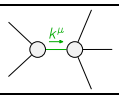
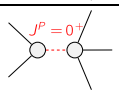
Added source term(s):	$\phi\,\rho + \mathcal{A}^\alpha\;\mathcal{J}_\alpha + \mathcal{B}^\alpha\;\mathcal{K}_\alpha + C^\alpha\;\mathcal{L}_\alpha$		
Source constraint(s)	# constraint(s)	Covariant form	
$\mathcal{L}_{0^+}^{\#1} == 0$	1	$\partial_\alpha\mathcal{L}^\alpha == 0$	
$\mathcal{K}_{0^+}^{\#1} == 0$	1	$\partial_\alpha\mathcal{K}^\alpha == 0$	
$\mathcal{J}_{1^-}^{\#1\,\alpha} == 0$	3	$\partial_\beta\partial^\alpha\mathcal{J}^\beta == \partial_\beta\partial^\beta\mathcal{J}^\alpha$	
Total # constraint(s):	5		
Resolved pole(s)	# polarization(s)	Square mass	Residue
	2	0	$-\frac{\delta+\beta+\sqrt{\delta^2-2\,\delta\,\beta+\beta^2+\gamma^2}}{4\,\delta\,\beta-\gamma^2}$
	2	0	$-\frac{\delta+\beta-\sqrt{\delta^2-2\,\delta\,\beta+\beta^2+\gamma^2}}{4\,\delta\,\beta-\gamma^2}$
	1	$\frac{\epsilon^2-4\,\alpha\,\zeta}{4\,\alpha\,\delta}$	$\frac{\alpha\,\epsilon^2-4\,\alpha^2\,\zeta+\epsilon^2\,\delta}{\alpha\,\epsilon^2\,\delta-4\,\alpha^2\,\zeta\,\delta}$
Resolved unitarity condition(s):		(Demonstrably impossible)	

Figure 7. Output generated by *PSALTer*, which assumes a $(+,-,-,-)$ signature different from the $(-,+,+,+)$ signature used throughout this work. The spectrograph of the second combined theory. The spectrum contains massless modes supported by the large gauge sector; the source constraints include two gauge parameters for the Maxwell-like fields. All notation is defined in tables 2 and 3 — the notation for B_μ and C_μ is analogous to that for A_μ.

Acknowledgements

This work was made possible by useful discussions with Boris Bolliet, Justin Feng, Dražen Glavan, Will Handley, Syksy Räsänen, Ignacy Sawicki, Richard Woodard and Tom Zlosnik.

I am grateful for the support of Girton College, Cambridge, Marie Skłodowska-Curie Actions and the Institute of Physics of the Czech Academy of Sciences.

This work used the DiRAC Data Intensive service (CSD3 www.csd3.cam.ac.uk) at the University of Cambridge, managed by the University of Cambridge University Information Services on behalf of the STFC DiRAC HPC Facility (www.dirac.ac.uk). The DiRAC component of CSD3 at Cambridge was funded by BEIS, UKRI and STFC capital funding and STFC operations grants. DiRAC is part of the UKRI Digital Research Infrastructure.

This work also used the Newton compute server, access to which was provisioned by Will Handley using an ERC grant.

Co-funded by the European Union (Physics for Future – Grant Agreement No. 101081515). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Research Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

A Gauss–Codazzi and the Ricci scalar

In this appendix, we provide a more conventional introduction to the ADM formalism than that given in section 2.2.1, allowing for a self-contained derivation of the well-known formulae of eqs. (2.5) and (2.6) which lie at the heart of our approach — we do not rely on computer manipulation [27, 56–58]. To begin with, our conventions for the curvature are as follows. Given some vector A^μ, the Ricci tensor is $R_{\sigma\nu} \equiv R^\lambda{}_{\sigma\lambda\nu}$ and the Riemann tensor $R^\rho{}_{\sigma\mu\nu}$ is given by

$$D_{[\mu}D_{\nu]}A^\rho \equiv \frac{1}{2}R^\rho{}_{\sigma\mu\nu}A^\sigma, \quad R^\rho{}_{\sigma\mu\nu} \equiv \partial_\mu\Gamma^\rho_{\sigma\nu} - \partial_\nu\Gamma^\rho_{\sigma\mu} + \Gamma^\alpha_{\sigma\nu}\Gamma^\rho_{\mu\alpha} - \Gamma^\alpha_{\sigma\mu}\Gamma^\rho_{\nu\alpha}, \tag{A.1}$$

where the covariant derivative D_μ and Christoffel symbol $\Gamma^\rho_{\sigma\mu}$ are

$$D_\mu A^\nu \equiv \partial_\mu A^\nu + \Gamma^\nu_{\mu\sigma}A^\sigma, \quad \Gamma^\rho_{\sigma\mu} \equiv \frac{1}{2}g^{\rho\alpha}\left(\partial_\sigma g_{\mu\alpha} + \partial_\mu g_{\sigma\alpha} - \partial_\alpha g_{\mu\sigma}\right). \tag{A.2}$$

It is a defining characteristic of the canonical formulation that a definite sense of time is agreed upon in the calculations; this we parameterise with the coordinate $t \equiv x^0$. The remaining coordinates x^i, with holonomic Roman indices (excluding t) from the middle of the alphabet running from one to three, are taken to be '*adapted*' in the sense that they span each spatial hypersurface for each respective value of t. Having set up the coordinate system, the conventional starting point for the ADM formalism is to introduce the unit-timelike vector field $n^\mu n_\mu \equiv -1$ with the property of being normal to spatial hypersurfaces $(\partial_i)^\mu\, n_\mu \equiv 0$, i.e., it is locally normal to the tangent space of the adapted x^i at every point on each surface. The relationship between t and n^μ is variable, and parameterised by the lapse and shift via the formula

$$\partial_t x^\mu \equiv N n^\mu + N^i \delta^\mu_i. \tag{A.3}$$

The extrinsic curvature in eq. (2.4) may then be defined in terms of the covariant derivative of the unit-timelike vector field according to

$$K_{ij} \equiv -D_j n_i \equiv -\frac{1}{2N}\left(\dot{h}_{ij} - 2\nabla_{(i}N_{j)}\right). \tag{A.4}$$

The conventional precursor to the canonical expansion of the Ricci scalar is the Gauss–Codazzi equation [59, 60] for the projected components of the Riemann tensor, which reads, together with its full contraction

$$R^{i}{}_{jkl} \equiv \mathcal{R}^{i}{}_{jkl} - K_{jk}K^{i}_{l} + K_{jl}K^{i}_{k}, \quad R^{ij}{}_{ij} \equiv \mathcal{R} + K^2 - K_{ij}K^{ij}. \tag{A.5}$$

Since the latter expression in eq. (A.5) can also be given by $R^{ij}{}_{ij} \equiv R + 2R^{\mu}{}_{\nu\mu\lambda}n^{\nu}n^{\lambda}$, we can use eq. (A.1) to show that

$$R^{\mu}{}_{\nu\mu\lambda}n^{\nu}n^{\lambda} \equiv 2n^{\lambda}D_{[\mu}D_{\lambda]}n^{\mu} \equiv 2D_{[\mu}\left(n^{\lambda}D_{\lambda]}n^{\mu}\right) - 2D_{[\mu}n^{\lambda}D_{\lambda]}n^{\mu}. \tag{A.6}$$

It follows from eq. (A.4) that $D_{\mu}n^{\mu} \equiv -K$ and $D_{\mu}n^{\lambda}D_{\lambda}n^{\mu} \equiv K_{ij}K^{ij}$, so that eqs. (A.5) and (A.6) may be used to deduce

$$\begin{aligned} R &\equiv \mathcal{R} - K^2 + K_{ij}K^{ij} + 4D_{[\mu}\left(n^{\mu}D_{\nu]}n^{\nu}\right) \\ &\equiv \mathcal{R} + K^2 - K_{ij}K^{ij} + 2n^{\mu}D_{\mu}K - 2n^{\mu}D_{\nu}D_{\mu}n^{\nu}. \end{aligned} \tag{A.7}$$

By similar reasoning, the final term in eq. (A.7) is $n^{\mu}D_{\nu}D_{\mu}n^{\nu} \equiv D_{\nu}\left(n^{\mu}D_{\mu}n^{\nu}\right) - K_{ij}K^{ij}$, whilst a consequence of eq. (A.3) is that $n^{\nu}D_{\nu}n_{\mu} \equiv \left(\delta^{\nu}_{\mu} + n^{\nu}n_{\mu}\right)D_{\nu}\ln N$. After a tedious calculation in which the latter identity is used several times it may be shown that

$$n^{\mu}D_{\nu}D_{\mu}n^{\nu} \equiv \frac{1}{N}\nabla_i\nabla^i N - K_{ij}K^{ij}. \tag{A.8}$$

Meanwhile, we can use eqs. (A.3) and (A.7) to write $n^{\mu}D_{\mu}K \equiv \left(\dot{K} - N^i\nabla_i K\right)/N$. The explicit time derivative of the whole trace requires some further manipulation: since $\dot{K} \equiv h^{ij}\dot{K}_{ij} - K^{ij}\dot{h}_{ij}$ we can use eq. (A.4) to write

$$n^{\mu}D_{\mu}K \equiv \frac{1}{N}\left(h^{ij}\dot{K}_{ij} + 2NK_{ij}K^{ij} - 2K^{ij}\nabla_i N_j - N^i\nabla_i K\right). \tag{A.9}$$

Finally, eqs. (2.5) and (2.6) are obtained by plugging eqs. (A.8) and (A.9) into eq. (A.7).