The ADM evolution equations. Code generation.

The job here is to take the ADM equations in the form

$$\partial_t g_{ab} = -2NK_{ab} \tag{dotgab.101}$$

$$\partial_t K_{ab} = -N_{|ab} + N\left(R_{ab} + \mathrm{tr}KK_{ab} - 2K_{ac}K_{bd}g^{cd}\right) \tag{dotKab.101}$$

$$\partial_t N = 0 \tag{dotN.101}$$

$$\mathcal{H} = R + K_{ab}g^{ab}K_{cd}g^{cd} - K_{ab}K_{cd}g^{ac}g^{bd} \tag{Ham.101}$$

$$\mathcal{D}_c = g^{ab}K_{ac|b} - \partial_c\left(g^{ab}K_{ab}\right) \tag{Mom.101}$$

where

$$\begin{aligned} R_{ab} = {} & \frac{1}{2}\partial_a g_{bc}\partial_d g^{cd} + \frac{1}{2}\partial_b g_{ac}\partial_d g^{cd} - \frac{1}{2}\partial_c g_{ab}\partial_d g^{cd} + \frac{1}{2}g^{cd}\partial_{ac}g_{bd} + \frac{1}{2}g^{cd}\partial_{bc}g_{ad} - \frac{1}{2}g^{cd}\partial_{cd}g_{ab} + \frac{1}{4}g^{cd}g^{ef}\partial_a g_{bc}\partial_d g_{ef} + \frac{1}{4}g^{cd}g^{ef}\partial_b g_{ac}\partial_d g_{ef} \\ & - \frac{1}{4}g^{cd}g^{ef}\partial_c g_{ab}\partial_d g_{ef} - \frac{1}{4}\partial_a g_{cd}\partial_b g^{cd} - \frac{1}{2}g^{cd}\partial_{ab}g_{cd} - \frac{1}{2}g^{cd}g^{ef}\partial_c g_{ae}\partial_f g_{bd} + \frac{1}{2}g^{cd}g^{ef}\partial_c g_{ae}\partial_d g_{bf} \end{aligned} \tag{Rab.112}$$

$$R = g^{ab}\partial_a g_{bc}\partial_d g^{cd} - g^{ab}\partial_c g_{ab}\partial_d g^{cd} + g^{ab}g^{cd}\partial_{ac}g_{bd} - g^{ab}g^{cd}\partial_{ab}g_{cd} - \frac{1}{4}g^{ab}g^{cd}g^{ef}\partial_a g_{cd}\partial_b g_{ef} - \frac{3}{4}g^{ab}\partial_a g_{cd}\partial_b g^{cd} + \frac{1}{2}g^{ab}\partial_c g_{ad}\partial_b g^{cd} \tag{R.110}$$

$$N_{|ab} = \partial_{ab}N - \frac{1}{2}g^{ce}\left(\partial_a g_{eb} + \partial_b g_{ae} - \partial_e g_{ab}\right)\partial_c N \tag{Nab.102}$$

$$\mathcal{D}_c = g^{ab}\partial_a K_{cb} + K_{ca}\partial_b g^{ab} + \frac{1}{2}g^{ab}g^{de}K_{ca}\partial_b g_{de} - \frac{1}{2}K_{ab}\partial_c g^{ab} - g^{ab}\partial_c K_{ab} \tag{Mom.110}$$

and to export the right hand sides as C-code. And that will be Cadabra's final curtain call.

```
from shared import *
import cdblib

Rscalar = cdblib.get ('Rscalar', 'adm-eqtns.json')
Rab     = cdblib.get ('Rab',     'adm-eqtns.json')
Nab     = cdblib.get ('Nab',     'adm-eqtns.json')
DgabDt  = cdblib.get ('DgabDt',  'adm-eqtns.json')
DKabDt  = cdblib.get ('DKabDt',  'adm-eqtns.json')
DNDt    = cdblib.get ('DNDt',    'adm-eqtns.json')
Ham     = cdblib.get ('Ham',     'adm-eqtns.json')
Mom     = cdblib.get ('Mom',     'adm-eqtns.json')

substitute (Rab, $ \partial_{a b}{g_{c d}} -> dg_{c d a b} $)
substitute (Rab, $ \partial_{a}{g_{b c}}   -> dg_{b c a}   $)

# ------------------------------------------------------------------------------
# Replace partial derivs with indices

def substitute_deriv (ex):
    substitute (ex, $ \partial_{a b}{g_{c d}} -> dg_{c d a b} $)
    substitute (ex, $ \partial_{a}{g_{b c}}   -> dg_{b c a}   $)
    substitute (ex, $ \partial_{a}{g^{b c}}   -> dg^{b c}_{a} $)
    substitute (ex, $ \partial_{a}{K_{b c}}   -> dK_{b c a}   $)
    substitute (ex, $ \partial_{a b}{N}       -> dN_{a b}     $)
    substitute (ex, $ \partial_{a}{N}         -> dN_{a}       $)
    substitute (ex, $ N_{a b}                 -> Hess_{a b}   $)
    return ex

Rscalar = substitute_deriv (Rscalar)
Rab     = substitute_deriv (Rab)
Nab     = substitute_deriv (Nab)
DgabDt  = substitute_deriv (DgabDt)
DKabDt  = substitute_deriv (DKabDt)
DNDt    = substitute_deriv (DNDt)
Ham     = substitute_deriv (Ham)
Mom     = substitute_deriv (Mom)

# ------------------------------------------------------------------------------
```

```
# build rules to export Cadabra expressions to Python
# use known symmetries for g_{a b}, dg_{ab,c,d} etc.
# note: replacements must not contain underscores (reserved for subscripts),
#       so g_{x x} -> g_xx is not allowed

gabRule := {g_{x x} -> gxx, g_{x y} -> gxy, g_{x z} -> gxz,
            g_{y x} -> gxy, g_{y y} -> gyy, g_{y z} -> gyz,
            g_{z x} -> gxz, g_{z y} -> gyz, g_{z z} -> gzz}.

iabRule := {g^{x x} -> ixx, g^{x y} -> ixy, g^{x z} -> ixz,
            g^{y x} -> ixy, g^{y y} -> iyy, g^{y z} -> iyz,
            g^{z x} -> ixz, g^{z y} -> iyz, g^{z z} -> izz}.

d1gabRule := {dg_{x x x} -> gxxx, dg_{x y x} -> gxyx, dg_{x z x} -> gxzx,
              dg_{y x x} -> gxyx, dg_{y y x} -> gyyx, dg_{y z x} -> gyzx,
              dg_{z x x} -> gxzx, dg_{z y x} -> gyzx, dg_{z z x} -> gzzx,

              dg_{x x y} -> gxxy, dg_{x y y} -> gxyy, dg_{x z y} -> gxzy,
              dg_{y x y} -> gxyy, dg_{y y y} -> gyyy, dg_{y z y} -> gyzy,
              dg_{z x y} -> gxzy, dg_{z y y} -> gyzy, dg_{z z y} -> gzzy,

              dg_{x x z} -> gxxz, dg_{x y z} -> gxyz, dg_{x z z} -> gxzz,
              dg_{y x z} -> gxyz, dg_{y y z} -> gyyz, dg_{y z z} -> gyzz,
              dg_{z x z} -> gxzz, dg_{z y z} -> gyzz, dg_{z z z} -> gzzz}.

d1iabRule := {dg^{x x}_{x} -> ixxx, dg^{x y}_{x} -> ixyx, dg^{x z}_{x} -> ixzx,
              dg^{y x}_{x} -> ixyx, dg^{y y}_{x} -> iyyx, dg^{y z}_{x} -> iyzx,
              dg^{z x}_{x} -> ixzx, dg^{z y}_{x} -> iyzx, dg^{z z}_{x} -> izzx,

              dg^{x x}_{y} -> ixxy, dg^{x y}_{y} -> ixyy, dg^{x z}_{y} -> ixzy,
              dg^{y x}_{y} -> ixyy, dg^{y y}_{y} -> iyyy, dg^{y z}_{y} -> iyzy,
              dg^{z x}_{y} -> ixzy, dg^{z y}_{y} -> iyzy, dg^{z z}_{y} -> izzy,

              dg^{x x}_{z} -> ixxz, dg^{x y}_{z} -> ixyz, dg^{x z}_{z} -> ixzz,
              dg^{y x}_{z} -> ixyz, dg^{y y}_{z} -> iyyz, dg^{y z}_{z} -> iyzz,
              dg^{z x}_{z} -> ixzz, dg^{z y}_{z} -> iyzz, dg^{z z}_{z} -> izzz}.

d2gabRule := {dg_{x x x x} -> gxxxx, dg_{x y x x} -> gxyxx, dg_{x z x x} -> gxzxx,
```

```
              dg_{y x x x} -> gxyxx, dg_{y y x x} -> gyyxx, dg_{y z x x} -> gyzxx,
              dg_{z x x x} -> gxzxx, dg_{z y x x} -> gyzxx, dg_{z z x x} -> gzzxx,
              dg_{x x y x} -> gxxxy, dg_{x y y x} -> gxyxy, dg_{x z y x} -> gxzxy,
              dg_{y x y x} -> gxyxy, dg_{y y y x} -> gyyxy, dg_{y z y x} -> gyzxy,
              dg_{z x y x} -> gxzxy, dg_{z y y x} -> gyzxy, dg_{z z y x} -> gzzxy,
              dg_{x x z x} -> gxxxz, dg_{x y z x} -> gxyxz, dg_{x z z x} -> gxzxz,
              dg_{y x z x} -> gxyxz, dg_{y y z x} -> gyyxz, dg_{y z z x} -> gyzxz,
              dg_{z x z x} -> gxzxz, dg_{z y z x} -> gyzxz, dg_{z z z x} -> gzzxz,

              dg_{x x x y} -> gxxxy, dg_{x y x y} -> gxyxy, dg_{x z x y} -> gxzxy,
              dg_{y x x y} -> gxyxy, dg_{y y x y} -> gyyxy, dg_{y z x y} -> gyzxy,
              dg_{z x x y} -> gxzxy, dg_{z y x y} -> gyzxy, dg_{z z x y} -> gzzxy,
              dg_{x x y y} -> gxxyy, dg_{x y y y} -> gxyyy, dg_{x z y y} -> gxzyy,
              dg_{y x y y} -> gxyyy, dg_{y y y y} -> gyyyy, dg_{y z y y} -> gyzyy,
              dg_{z x y y} -> gxzyy, dg_{z y y y} -> gyzyy, dg_{z z y y} -> gzzyy,
              dg_{x x z y} -> gxxyz, dg_{x y z y} -> gxyyz, dg_{x z z y} -> gxzyz,
              dg_{y x z y} -> gxyyz, dg_{y y z y} -> gyyyz, dg_{y z z y} -> gyzyz,
              dg_{z x z y} -> gxzyz, dg_{z y z y} -> gyzyz, dg_{z z z y} -> gzzyz,

              dg_{x x x z} -> gxxxz, dg_{x y x z} -> gxyxz, dg_{x z x z} -> gxzxz,
              dg_{y x x z} -> gxyxz, dg_{y y x z} -> gyyxz, dg_{y z x z} -> gyzxz,
              dg_{z x x z} -> gxzxz, dg_{z y x z} -> gyzxz, dg_{z z x z} -> gzzxz,
              dg_{x x y z} -> gxxyz, dg_{x y y z} -> gxyyz, dg_{x z y z} -> gxzyz,
              dg_{y x y z} -> gxyyz, dg_{y y y z} -> gyyyz, dg_{y z y z} -> gyzyz,
              dg_{z x y z} -> gxzyz, dg_{z y y z} -> gyzyz, dg_{z z y z} -> gzzyz,
              dg_{x x z z} -> gxxzz, dg_{x y z z} -> gxyzz, dg_{x z z z} -> gxzzz,
              dg_{y x z z} -> gxyzz, dg_{y y z z} -> gyyzz, dg_{y z z z} -> gyzzz,
              dg_{z x z z} -> gxzzz, dg_{z y z z} -> gyzzz, dg_{z z z z} -> gzzzz}.

KabRule := {K_{x x} -> Kxx, K_{x y} -> Kxy, K_{x z} -> Kxz,
            K_{y x} -> Kxy, K_{y y} -> Kyy, K_{y z} -> Kyz,
            K_{z x} -> Kxz, K_{z y} -> Kyz, K_{z z} -> Kzz}.

RabRule := {R_{x x} -> Rxx, R_{x y} -> Rxy, R_{x z} -> Rxz,
            R_{y x} -> Rxy, R_{y y} -> Ryy, R_{y z} -> Ryz,
            R_{z x} -> Rxz, R_{z y} -> Ryz, R_{z z} -> Rzz}.

HessRule := {Hess_{x x} -> Hessxx, Hess_{x y} -> Hessxy, Hess_{x z} -> Hessxz,
```

```
              Hess_{y x} -> Hessxy, Hess_{y y} -> Hessyy, Hess_{y z} -> Hessyz,
              Hess_{z x} -> Hessxz, Hess_{z y} -> Hessyz, Hess_{z z} -> Hesszz}.

NaRule := {dN_{x} -> Nx, dN_{y} -> Ny, dN_{z} -> Nz}.

NabRule := {dN_{x x} -> Nxx, dN_{x y} -> Nxy, dN_{x z} -> Nxz,
            dN_{y x} -> Nxy, dN_{y y} -> Nyy, dN_{y z} -> Nyz,
            dN_{z x} -> Nxz, dN_{z y} -> Nyz, dN_{z z} -> Nzz}.

dKabRule := {dK_{x x x} -> Kxxx, dK_{x y x} -> Kxyx, dK_{x z x} -> Kxzx,
             dK_{y x x} -> Kxyx, dK_{y y x} -> Kyyx, dK_{y z x} -> Kyzx,
             dK_{z x x} -> Kxzx, dK_{z y x} -> Kyzx, dK_{z z x} -> Kzzx,
             dK_{x x y} -> Kxxy, dK_{x y y} -> Kxyy, dK_{x z y} -> Kxzy,
             dK_{y x y} -> Kxyy, dK_{y y y} -> Kyyy, dK_{y z y} -> Kyzy,
             dK_{z x y} -> Kxzy, dK_{z y y} -> Kyzy, dK_{z z y} -> Kzzy,
             dK_{x x z} -> Kxxz, dK_{x y z} -> Kxyz, dK_{x z z} -> Kxzz,
             dK_{y x z} -> Kxyz, dK_{y y z} -> Kyyz, dK_{y z z} -> Kyzz,
             dK_{z x z} -> Kxzz, dK_{z y z} -> Kyzz, dK_{z z z} -> Kzzz}.

allRules =   gabRule + d1gabRule + d2gabRule \
           + iabRule + d1iabRule \
           + KabRule + dKabRule \
           + RabRule \
           + HessRule + NaRule + NabRule

evaluate (Rscalar, allRules, simplify=False)
evaluate (Rab,     allRules, simplify=False)
evaluate (Nab,     allRules, simplify=False)

evaluate (DgabDt,  allRules, simplify=False)
evaluate (DKabDt,  allRules, simplify=False)

evaluate (DNDt,    allRules, simplify=False)

evaluate (Ham,     allRules, simplify=False)
evaluate (Mom,     allRules, simplify=False)

# ------------------------------------------------------------------------------
```

```
# export to C-code

import writecode

writecode.cdb_write_code (Rscalar, 'R',       'ricci-scalar.c', 0)
writecode.cdb_write_code (Rab,     'Rab',     'ricci.c',        2)
writecode.cdb_write_code (DgabDt,  'dot_gab', 'dot-gab.c',      2)
writecode.cdb_write_code (DKabDt,  'dot_Kab', 'dot-Kab.c',      2)
writecode.cdb_write_code (DNDt,    'dot_N',   'dot-N.c',        0)
writecode.cdb_write_code (Nab,     'Hess',    'hessian.c',      2)

writecode.cdb_write_code (Ham,     'Ham',     'hamiltonian.c',  0)
writecode.cdb_write_code (Mom,     'Mom',     'momentum.c',     1)
```