The McLachlan Code

Peter Diener

Louisiana State University

August 12, 2015
Einstein Toolkit Workshop 2015
AlbaNova University Center, Stockholm, Sweden

Outline

- The ADM formulation.
- The BSSN formulation.
- Kranc.
- McLachlan.

The ADM formulation.

The 3+1 ADM evolution equations are

$$\begin{aligned}(\partial_t - \mathcal{L}_\beta)\,\gamma_{ij} &= -2\alpha K_{ij},\\ (\partial_t - \mathcal{L}_\beta)\,K_{ij} &= -D_i D_j \alpha + \alpha(R_{ij} + K K_{ij} - 2K_{ik}K^k{}_j),\end{aligned}$$

with the constraints

$$\begin{aligned}\mathcal{H} &\equiv R + K^2 - K_{ij}K^{ij} = 0,\\ \mathcal{M}^i &\equiv D_j(K^{ij} - \gamma^{ij}K) = 0.\end{aligned}$$

This set of PDE's is only weakly hyperbolic and is therefore not suitable for numerical evolution.
However, they provide a convenient starting point for a more stable formulation: The BSSN (Baumgarte-Shapiro-Shibata-Nakamura)[1] formulation.

---

[1] Should really include Oohara-Kojima and be BSSNOK.

The BSSN formulation (new variables).

Introduce a conformal rescaling of the three metric

$$\gamma_{ij} = \psi^4 \tilde{\gamma}_{ij}.$$

We choose $\psi = \gamma^{1/12}$ such that $\det(\tilde{\gamma}_{ij}) = 1$

In addition we introduce a trace decomposition of the extrinsic curvature.

$$K = \gamma^{ij} K_{ij},$$
$$A_{ij} = K_{ij} - \frac{1}{3}\gamma_{ij} K.$$

We then promote the following variables to evolution variables

$$\phi = \ln\psi = \frac{1}{12}\ln\gamma, \qquad K = \gamma_{ij} K^{ij},$$
$$\tilde{\gamma}_{ij} = e^{-4\phi}\gamma_{ij}, \qquad \tilde{A}_{ij} = e^{-4\phi} A_{ij},$$

as well as the conformal connection functions

$$\tilde{\Gamma}^i = \tilde{\gamma}^{jk} \tilde{\Gamma}^i{}_{jk} = -\partial_j \tilde{\gamma}^{ij}.$$

The BSSN formulation (evolution equations).

$$\begin{aligned}
\partial_t \tilde{\gamma}_{ij} &= -2\alpha \tilde{A}_{ij} + \beta^k \partial_k \tilde{\gamma}_{ij} + \tilde{\gamma}_{ik}\partial_j \beta^k + \tilde{\gamma}_{jk}\partial_i \beta^k - \frac{2}{3}\tilde{\gamma}_{ij}\partial_k \beta^k, \\
\partial_t \phi &= -\frac{1}{6}\alpha K + \beta^k \partial_k \phi + \frac{1}{6}\partial_k \beta^k, \\
\partial_t \tilde{A}_{ij} &= e^{-4\phi}[-D_i D_j \alpha + \alpha R_{ij}]^{TF} + \alpha(K\tilde{A}_{ij} - 2\tilde{A}_{ik}\tilde{A}^k{}_j) \\
&\quad + \beta^k \partial_k \tilde{A}_{ij} + \tilde{A}_{ik}\partial_j \beta^k + \tilde{A}_{jk}\partial_i \beta^k - \frac{2}{3}\tilde{A}_{ij}\partial_k \beta^k, \\
\partial_t K &= -D^i D_i \alpha + \alpha(\tilde{A}_{ij}\tilde{A}^{ij} + \frac{1}{3}K^2) + \beta^k \partial_k K, \\
\partial_t \tilde{\Gamma}^i &= \tilde{\gamma}^{jk}\partial_j \partial_k \beta^i + \frac{1}{3}\tilde{\gamma}^{ij}\partial_j \partial_k \beta^k + \beta^j \partial_j \tilde{\Gamma}^i - \tilde{\Gamma}^j \partial_j \beta^i + \frac{2}{3}\tilde{\Gamma}^i \partial_j \beta^j \\
&\quad - 2\tilde{A}^{ij}\partial_j \alpha + 2\alpha(\tilde{\Gamma}^i{}_{jk}\tilde{A}^{jk} + 6\tilde{A}^{ij}\partial_j \phi - \frac{2}{3}\tilde{\gamma}^{ij}\partial_j K).
\end{aligned}$$

Here $R_{ij} = \tilde{R}_{ij} + R^{\phi}_{ij}$, where

$$\begin{aligned}
\tilde{R}_{ij} &= -\frac{1}{2}\tilde{\gamma}^{lm}\partial_l \partial_m \tilde{\gamma}_{ij} + \tilde{\gamma}_{k(i}\partial_{j)}\tilde{\Gamma}^k + \tilde{\Gamma}^k \tilde{\Gamma}_{(ij)k} \\
&\quad + \tilde{\gamma}^{lm}\left(2\tilde{\Gamma}^k{}_{l(i}\tilde{\Gamma}_{j)km} + \tilde{\Gamma}^k{}_{im}\tilde{\Gamma}_{klj}\right), \\
R^{\phi}_{ij} &= -2\tilde{D}_i \tilde{D}_j \phi - 2\tilde{\gamma}_{ij}\tilde{D}^k \tilde{D}_k \phi + 4\tilde{D}_i \phi\, \tilde{D}_j \phi - 4\tilde{\gamma}_{ij}\tilde{D}^k \phi\, \tilde{D}_k \phi.
\end{aligned}$$

The BSSN formulation (constraint equations).

The constraints are

$$
\begin{aligned}
\tilde{\mathcal{H}} &\equiv R + \frac{2}{3}K^2 - \tilde{A}_{ij}\tilde{A}^{ij} = 0, \\
\tilde{\mathcal{M}}^i &\equiv \tilde{D}_j \tilde{A}^{ij} + 6\tilde{A}^{ij}\partial_j \phi - \frac{2}{3}\tilde{\gamma}^{ij}\partial_j K = 0, \\
\tilde{\mathcal{G}} &\equiv \tilde{\gamma} - 1 = 0, \\
\tilde{\mathcal{A}} &\equiv \tilde{\gamma}^{ij}\tilde{A}_{ij} = 0, \\
\tilde{\mathcal{L}}^i &\equiv \tilde{\Gamma}^i + \partial_j \tilde{\gamma}^{ij} = 0.
\end{aligned}
$$

The constraints $\tilde{\mathcal{G}}$ and $\tilde{\mathcal{A}}$ are enforced actively at each time-step.
The other constraints ($\tilde{\mathcal{H}}$, $\tilde{\mathcal{M}}^i$ and $\tilde{\mathcal{L}}^i$) are not enforced.
To improve stability and to help keep $\tilde{\mathcal{L}}^i$ small, the following rule is employed:

- Where derivatives of $\tilde{\Gamma}^i$ are needed, the evolved $\tilde{\Gamma}^i$ are used directly.
- Where $\tilde{\Gamma}^i$ are needed without taking derivatives, $\tilde{\gamma}^{jk}\tilde{\Gamma}^i{}_{jk}$ are used instead.

The BSSN formulation (gauge conditions).

$1 + \log$ family of lapse conditions.

$$\partial_t \alpha = -F\alpha^N K + \text{alphaDriver}(\alpha - 1) + \text{advectLapse}\beta^i \partial_i \alpha.$$

Harmonic slicing: $F = 1$, $N = 2$,
$1 + \log$ slicing: $F = 2$, $N = 1$.
There is also a variant using $A = \partial_t \alpha$ as an evolution variable.

Hyperbolic gamma driver condition:

$$\partial_t \beta^i = \text{shiftGammaCoeff}\, B^i + \text{advectShift}\, \beta^j \partial_j \beta^i,$$
$$\partial_t B^i = \partial_t \tilde{\Gamma}^i - \text{betaDriver}\, B^i + \text{advectShift}\, \beta^j \partial_j B^i.$$

Here shiftGammaCoeff $= 3/4$ and betaDriver has to be chosen appropriately for the mass of the black holes in the system.

The BSSN formulation (the W-method).

Instead of using $\phi = 1/12 \ln \gamma$ as an evolution variable it is also possible to use

$$W = \gamma^{-1/6} = e^{-2\phi}$$

in which case the evolution equation for W is

$$\partial_t W = \frac{1}{3} W(\alpha K - \partial_i \beta^i) + \beta^i \partial_i W,$$

and the expression for R^{ϕ}_{ij} is similarly converted to an expression involving derivatives of W.

We currently do not support the χ-method ($\chi = e^{-4\phi}$).

Kranc.

- Kranc is a set of mathematica scripts initially developed by Sascha Husa and Christiane Lechner and currently mainly developed by Ian Hinder for converting a set of tensorial evolution equations into Cactus code.
- It was originally created in order to allow easy experimentation
- with different formulations of the Einstein equations.
- Kranc produces a complete Cactus thorn including the configuration files.
- Kranc provides mathematica routines to define tensors and their properties and how they relate to the Cactus grid functions.
- Kranc interfaces with MoL and one of it's main functions is to produce the RHS evaluation routine for the evolution equations.
- In addition there are routines to define Cactus parameters.
- The user defines "Calculations" to operate on the tensors along with scheduling information.

McLachlan.

- McLachlan (named after the Canadian Singer/Songwriter Sarah McLachlan) is an implementation of the BSSN equations in Kranc.
- Supports any kind of matter through it's interface with TmunuBase.
- The RHS routine is split into smaller pieces to avoid instruction cache misses.
- Kranc can generate explicitly vectorized versions of the code.
- McLachlan supports the Llama multi-patch infrastructure.
- If desired, some parameters can be set at Kranc code generation time for improved optimizations (10–20%).
- The Kranc script is readable and extensible.
- McLachlan can also generate the conformal and covariant Z4-formulation (CCZ4).
- Kranc generates LoopControl loops so McLachlan is OpenMP parallelized by default.

McLachlan.

- Erik Schnetter also added support in `Kranc` for generating an OpenCL version.
- `McLachlan` currently needs the `ML_BSSN_Helper` thorn in order to handle Cactus related things that are not yet supported by Kranc itself.
- `McLachlan` will be able to run efficiently on GPU's with the development of `Chemora`.
- Etienne et. al have proposed a set of modifications to the gauge evolution equations that can reduce the constraint violations significantly.
- The gauge evolution part of the code is kind of messy and could use some cleanup and/or simplification.

McLachlan.

- The full BSSN equation description in Kranc is contained on 278 lines (including comments and empty lines).
- The total Kranc script is currently 1477 lines.
- The total number of C++ source lines in the generated code is more than $27,000$.

As an example

```
Gtl[la,lb,lc]  -> (1/2 (+ PD[gt[lb,la],lc]
                        + PD[gt[lc,la],lb]
                        - PD[gt[lb,lc],la]))
Gt[ua,lb,lc]   -> gtu[ua,ud] Gtl[ld,lb,lc]
```

McLachlan.

turns into (vectorization turned off, Jacobians turned on)

```
CCTK_REAL Gtl111 = 0.5*JacPDstandardNth1gt11;
CCTK_REAL Gtl112 = 0.5*JacPDstandardNth2gt11;
CCTK_REAL Gtl113 = 0.5*JacPDstandardNth3gt11;
CCTK_REAL Gtl122 = -0.5*JacPDstandardNth1gt22 +
  JacPDstandardNth2gt12;

CCTK_REAL Gt111 = Gtl111*gtu11 + Gtl211*gtu12 +
  Gtl311*gtu13;
CCTK_REAL Gt211 = Gtl111*gtu12 + Gtl211*gtu22 +
  Gtl311*gtu23;
CCTK_REAL Gt311 = Gtl111*gtu13 + Gtl211*gtu23 +
  Gtl311*gtu33;
CCTK_REAL Gt112 = Gtl112*gtu11 + Gtl212*gtu12 +
  Gtl312*gtu13;
```

+ many additional lines of codes for the remaining tensor components.

McLachlan.

turns into (vectorization turned on, Jacobians turned on)

```
CCTK_REAL_VEC Gtl111 =
  kmul(JacPDstandardNth1gt11,ToReal(0.5));
CCTK_REAL_VEC Gtl112 =
  kmul(JacPDstandardNth2gt11,ToReal(0.5));
CCTK_REAL_VEC Gtl113 =
  kmul(JacPDstandardNth3gt11,ToReal(0.5));
CCTK_REAL_VEC Gtl122 =
  kmadd(ToReal(-0.5),JacPDstandardNth1gt22,JacPDstandardNth2gt12);

CCTK_REAL_VEC Gt111 =
  kmadd(Gtl111,gtu11,kmadd(Gtl211,gtu12,kmul(Gtl311,gtu13)));
CCTK_REAL_VEC Gt211 =
  kmadd(Gtl111,gtu12,kmadd(Gtl211,gtu22,kmul(Gtl311,gtu23)));
CCTK_REAL_VEC Gt311 =
  kmadd(Gtl111,gtu13,kmadd(Gtl211,gtu23,kmul(Gtl311,gtu33)));
CCTK_REAL_VEC Gt112 =
  kmadd(Gtl112,gtu11,kmadd(Gtl212,gtu12,kmul(Gtl312,gtu13)));
```

+ many additional lines of codes for the remaining tensor components.