

# Differential geometry with the open-source software system SageMath

Éric Gourgoulhon

Laboratoire Univers et Théories (LUTH)  
CNRS / Observatoire de Paris / Université Paris Diderot  
Paris Sciences et Lettres Research University  
92190 Meudon, France

<http://luth.obspm.fr/~luthier/gourgoulhon/>

**Cosmology, Particle Physics and Phenomenology - CP3**  
UCL, Louvain-la-Neuve, Belgium  
29 November 2016

# Outline

- 1 Computer differential geometry and tensor calculus
- 2 SageMath
- 3 The SageManifolds project
- 4 Let us practice!
- 5 Other examples
- 6 Conclusion and perspectives

# Outline

- 1 Computer differential geometry and tensor calculus
- 2 SageMath
- 3 The SageManifolds project
- 4 Let us practice!
- 5 Other examples
- 6 Conclusion and perspectives

# Introduction

- **Computer algebra system (CAS)** started to be developed in the 1960's; for instance **Macsyma** (to become **Maxima** in 1998) was initiated in 1968 at MIT

# Introduction

- **Computer algebra system (CAS)** started to be developed in the 1960's; for instance **Macsyma** (to become **Maxima** in 1998) was initiated in 1968 at MIT
- In 1965, J.G. Fletcher developed the **GEOM** program, to compute the Riemann tensor of a given metric

# Introduction

- **Computer algebra system (CAS)** started to be developed in the 1960's; for instance **Macsyma** (to become **Maxima** in 1998) was initiated in 1968 at MIT
- In 1965, J.G. Fletcher developed the **GEOM** program, to compute the Riemann tensor of a given metric
- In 1969, during his PhD under Pirani supervision, Ray d'Inverno wrote **ALAM (Atlas Lisp Algebraic Manipulator)** and used it to compute the Riemann tensor of Bondi metric. The original calculations took Bondi and his collaborators 6 months to go. The computation with ALAM took 4 minutes and yielded to the discovery of 6 errors in the original paper [J.E.F. Skea, *Applications of SHEEP* (1994)]

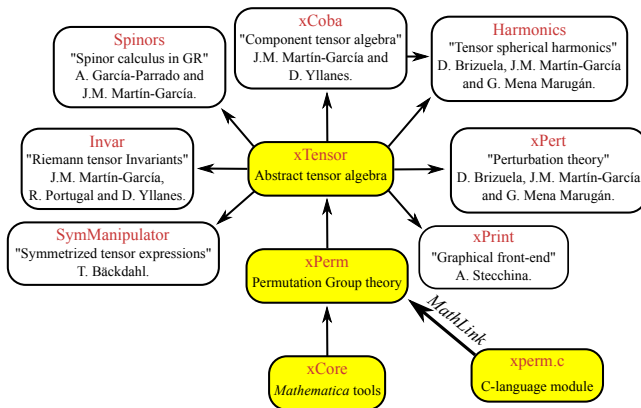
# Introduction

- **Computer algebra system (CAS)** started to be developed in the 1960's; for instance **Macsyma** (to become **Maxima** in 1998) was initiated in 1968 at MIT
- In 1965, J.G. Fletcher developed the **GEOM** program, to compute the Riemann tensor of a given metric
- In 1969, during his PhD under Pirani supervision, Ray d'Inverno wrote **ALAM (Atlas Lisp Algebraic Manipulator)** and used it to compute the Riemann tensor of Bondi metric. The original calculations took Bondi and his collaborators 6 months to go. The computation with ALAM took 4 minutes and yielded to the discovery of 6 errors in the original paper [[J.E.F. Skea, Applications of SHEEP \(1994\)](#)]
- Since then, many softwares for tensor calculus have been developed...

# An example of modern software: The xAct suite

Free packages for tensor computer algebra in Mathematica, developed by José Martín-García et al. <http://www.xact.es/>

## The xAct system



[García-Parrado Gómez-Lobo & Martín-García, *Comp. Phys. Comm.* **183**, 2214 (2012)]



# Software for differential geometry

## Packages for general purpose computer algebra systems:

- **xAct** free package for Mathematica [J.-M. Martin-Garcia]
- **Ricci** free package for Mathematica [J. L. Lee]
- **MathTensor** package for Mathematica [S. M. Christensen & L. Parker]
- **GRTensor** package for Maple [P. Musgrave, D. Pollney & K. Lake]
- **DifferentialGeometry** included in Maple [I. M. Anderson & E. S. Cheb-Terrab]
- **Atlas 2** for Maple and Mathematica
- ...

## Standalone applications:

- **SHEEP**, **Classi**, **STensor**, based on Lisp, developed in 1970's and 1980's (free) [R. d'Inverno, I. Frick, J. Åman, J. Skea, et al.]
- **Cadabra** field theory (free) [K. Peeters]
- **SnapPy** topology and geometry of 3-manifolds, based on Python (free) [M. Culler, N. M. Dunfield & J. R. Weeks]
- ...

cf. the complete list at <http://www.xact.es/links.html>

# Outline

- 1 Computer differential geometry and tensor calculus
- 2 SageMath**
- 3 The SageManifolds project
- 4 Let us practice!
- 5 Other examples
- 6 Conclusion and perspectives

# SageMath in a few words

- **SageMath** (*nickname: Sage*) is a **free open-source** mathematics software system

# SageMath in a few words

- SageMath (*nickname: Sage*) is a **free open-source** mathematics software system
- it is based on the Python programming language

# SageMath in a few words

- **SageMath** (*nickname: Sage*) is a **free open-source** mathematics software system
- it is based on the **Python** programming language
- it makes use of **many pre-existing open-sources packages**, among which

and provides a **uniform interface** to them

# SageMath in a few words

- **SageMath** (*nickname: Sage*) is a **free open-source** mathematics software system
- it is based on the **Python** programming language
- it makes use of **many pre-existing open-sources packages**, among which
  - **Maxima**, **Pynac** (symbolic calculations)

and provides a **uniform interface** to them

# SageMath in a few words

- **SageMath** (*nickname: Sage*) is a **free open-source** mathematics software system
- it is based on the **Python** programming language
- it makes use of **many pre-existing open-sources packages**, among which
  - **Maxima**, **Pynac** (symbolic calculations)
  - **GAP** (group theory)

and provides a **uniform interface** to them

# SageMath in a few words

- **SageMath** (*nickname: Sage*) is a **free open-source** mathematics software system
- it is based on the **Python** programming language
- it makes use of **many pre-existing open-sources packages**, among which
  - **Maxima**, **Pynac** (symbolic calculations)
  - **GAP** (group theory)
  - **PARI/GP** (number theory)

and provides a **uniform interface** to them



# SageMath in a few words

- **SageMath** (*nickname: Sage*) is a **free open-source** mathematics software system
- it is based on the **Python** programming language
- it makes use of **many pre-existing open-sources packages**, among which
  - **Maxima**, **Pynac** (symbolic calculations)
  - **GAP** (group theory)
  - **PARI/GP** (number theory)
  - **Singular** (polynomial computations)

and provides a **uniform interface** to them

# SageMath in a few words

- **SageMath** (*nickname: Sage*) is a **free open-source** mathematics software system
  - it is based on the **Python** programming language
  - it makes use of **many pre-existing open-sources packages**, among which
    - **Maxima**, **Pynac** (symbolic calculations)
    - **GAP** (group theory)
    - **PARI/GP** (number theory)
    - **Singular** (polynomial computations)
    - **matplotlib** (high quality 2D figures)
- and provides a **uniform interface** to them

# SageMath in a few words

- **SageMath** (*nickname: Sage*) is a **free open-source** mathematics software system
- it is based on the **Python** programming language
- it makes use of **many pre-existing open-sources packages**, among which
  - **Maxima**, **Pynac** (symbolic calculations)
  - **GAP** (group theory)
  - **PARI/GP** (number theory)
  - **Singular** (polynomial computations)
  - **matplotlib** (high quality 2D figures)

and provides a **uniform interface** to them

- William Stein (Univ. of Washington) created SageMath in 2005; since then, **~100 developers** (mostly mathematicians) have joined the SageMath team

# SageMath in a few words

- **SageMath** (nickname: **Sage**) is a **free open-source** mathematics software system
- it is based on the **Python** programming language
- it makes use of **many pre-existing open-sources packages**, among which
  - **Maxima**, **Pynac** (symbolic calculations)
  - **GAP** (group theory)
  - **PARI/GP** (number theory)
  - **Singular** (polynomial computations)
  - **matplotlib** (high quality 2D figures)

and provides a **uniform interface** to them

- William Stein (Univ. of Washington) created SageMath in 2005; since then, **~100 developers** (mostly mathematicians) have joined the SageMath team
- SageMath is now supported by European Union via the open-math project **OpenDreamKit** (2015-2019, within the *Horizon 2020* program)

# SageMath in a few words

- **SageMath** (nickname: **Sage**) is a **free open-source** mathematics software system
- it is based on the **Python** programming language
- it makes use of **many pre-existing open-sources packages**, among which
  - **Maxima**, **Pynac** (symbolic calculations)
  - **GAP** (group theory)
  - **PARI/GP** (number theory)
  - **Singular** (polynomial computations)
  - **matplotlib** (high quality 2D figures)

and provides a **uniform interface** to them

- William Stein (Univ. of Washington) created SageMath in 2005; since then, **~100 developers** (mostly mathematicians) have joined the SageMath team
- SageMath is now supported by European Union via the open-math project **OpenDreamKit** (2015-2019, within the *Horizon 2020* program)

# SageMath in a few words

- **SageMath** (nickname: **Sage**) is a **free open-source** mathematics software system
- it is based on the **Python** programming language
- it makes use of **many pre-existing open-sources packages**, among which
  - **Maxima**, **Pynac** (symbolic calculations)
  - **GAP** (group theory)
  - **PARI/GP** (number theory)
  - **Singular** (polynomial computations)
  - **matplotlib** (high quality 2D figures)

and provides a **uniform interface** to them

- William Stein (Univ. of Washington) created SageMath in 2005; since then, **~100 developers** (mostly mathematicians) have joined the SageMath team
- SageMath is now supported by European Union via the open-math project **OpenDreamKit** (2015-2019, within the *Horizon 2020* program)

## The mission

*Create a viable free open source alternative to Magma, Maple, Mathematica and Matlab.*

# Some advantages of SageMath

## SageMath is free

Freedom means

- 1 everybody can use it, by downloading the software from <http://sagemath.org>
- 2 everybody can examine the source code and improve it

# Some advantages of SageMath

## SageMath is free

Freedom means

- 1 everybody can use it, by downloading the software from <http://sagemath.org>
- 2 everybody can examine the source code and improve it

## SageMath is based on Python

- no need to learn any specific syntax to use it
- easy access for students
- Python is a very powerful *object oriented language*, with a neat syntax



# Some advantages of SageMath

## SageMath is free

Freedom means

- 1 everybody can use it, by downloading the software from <http://sagemath.org>
- 2 everybody can examine the source code and improve it

## SageMath is based on Python

- no need to learn any specific syntax to use it
- easy access for students
- Python is a very powerful *object oriented language*, with a neat syntax

## SageMath is developing and spreading fast

...sustained by an enthusiast community of developers

# Object-oriented notation in Python

As an **object-oriented language**, Python (and hence SageMath) makes use of the following **postfix notation** (same in C++, Java, etc.):

```
result = object.function(arguments)
```

In a **procedural language**, this would be written as

```
result = function(object, arguments)
```

# Object-oriented notation in Python

As an **object-oriented language**, Python (and hence SageMath) makes use of the following **postfix notation** (same in C++, Java, etc.):

```
result = object.function(arguments)
```

In a **procedural language**, this would be written as

```
result = function(object, arguments)
```

## Examples

1. `riem = g.riemann()`
2. `lie_t_v = t.lie_der(v)`

NB: no argument in example 1

# SageMath approach to computer mathematics

SageMath relies on a **Parent / Element** scheme: each object  $x$  on which some calculus is performed has a “parent”, which is another SageMath object  $X$  representing the set to which  $x$  belongs.

The calculus rules on  $x$  are determined by the *algebraic structure* of  $X$ .

*Conversion rules* prior to an operation, e.g.  $x + y$  with  $x$  and  $y$  having different parents, are defined at the level of the parents

# SageMath approach to computer mathematics

SageMath relies on a **Parent / Element** scheme: each object  $x$  on which some calculus is performed has a “parent”, which is another SageMath object  $X$  representing the set to which  $x$  belongs.

The calculus rules on  $x$  are determined by the *algebraic structure* of  $X$ .

*Conversion rules* prior to an operation, e.g.  $x + y$  with  $x$  and  $y$  having different parents, are defined at the level of the parents

## Example

```
sage: x = 4 ; x.parent()
```

```
Integer Ring
```

```
sage: y = 4/3 ; y.parent()
```

```
Rational Field
```

```
sage: s = x + y ; s.parent()
```

```
Rational Field
```

```
sage: y.parent().has_coerce_map_from(x.parent())
```

```
True
```

# SageMath approach to computer mathematics

SageMath relies on a **Parent / Element** scheme: each object  $x$  on which some calculus is performed has a “parent”, which is another SageMath object  $X$  representing the set to which  $x$  belongs.

The calculus rules on  $x$  are determined by the *algebraic structure* of  $X$ .

*Conversion rules* prior to an operation, e.g.  $x + y$  with  $x$  and  $y$  having different parents, are defined at the level of the parents

## Example

```
sage: x = 4 ; x.parent()
Integer Ring
sage: y = 4/3 ; y.parent()
Rational Field
sage: s = x + y ; s.parent()
Rational Field
sage: y.parent().has_coerce_map_from(x.parent())
True
```

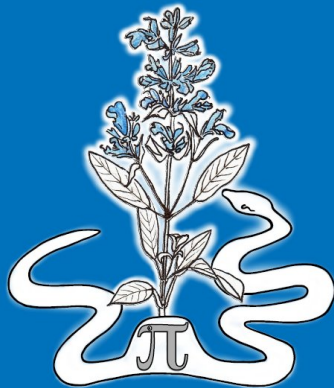
This approach is similar to that of Magma and is different from that of Mathematica, in which everything is a tree of symbols

# The Sage book

Calcul mathématique avec



# SAGE



by A. Casamayou, N. Cohen, G. Connan,  
T. Dumont, L. Fousse, F. Maltey, M. Meulien,  
M. Mezzarobba, C. Pernet, N.M. Thiéry &  
P. Zimmermann (2013)

Released under *Creative Commons* license:

- freely downloadable from  
<http://sagebook.gforge.inria.fr/>
- printed copies can be ordered at moderate price (10 €)

# Outline

- 1 Computer differential geometry and tensor calculus
- 2 SageMath
- 3 The SageManifolds project**
- 4 Let us practice!
- 5 Other examples
- 6 Conclusion and perspectives



# The SageManifolds project

<http://sagemanifolds.obspm.fr/>

## Aim

Implement **real smooth manifolds** of arbitrary dimension in Sage and **tensor calculus** on them

In particular:

- one should be able to introduce an arbitrary number of coordinate charts on a given manifold, with the relevant transition maps
- tensor fields must be manipulated as such and not through their components with respect to a specific (possibly coordinate) vector frame

# The SageManifolds project

<http://sagemanifolds.obspm.fr/>

## Aim

Implement **real smooth manifolds** of arbitrary dimension in Sage and **tensor calculus** on them

In particular:

- one should be able to introduce an arbitrary number of coordinate charts on a given manifold, with the relevant transition maps
- tensor fields must be manipulated as such and not through their components with respect to a specific (possibly coordinate) vector frame

Concretely, the project amounts to creating new Python classes, such as **Manifold**, **Chart**, **TensorField** or **Metric**, within SageMath's **Parent/Element framework**.

# Implementing coordinate charts

Given a (topological) manifold  $M$  of dimension  $n \geq 1$ , a **coordinate chart** is a homeomorphism  $\varphi : U \rightarrow V$ , where  $U$  is an open subset of  $M$  and  $V$  is an open subset of  $\mathbb{R}^n$ .

# Implementing coordinate charts

Given a (topological) manifold  $M$  of dimension  $n \geq 1$ , a **coordinate chart** is a homeomorphism  $\varphi : U \rightarrow V$ , where  $U$  is an open subset of  $M$  and  $V$  is an open subset of  $\mathbb{R}^n$ .

In general, more than one chart is required to cover the entire manifold:

## Examples:

- at least 2 charts are necessary to cover the  $n$ -dimensional sphere  $S^n$  ( $n \geq 1$ ) and the torus  $T^2$
- at least 3 charts are necessary to cover the real projective plane  $\mathbb{R}P^2$

# Implementing coordinate charts

Given a (topological) manifold  $M$  of dimension  $n \geq 1$ , a **coordinate chart** is a homeomorphism  $\varphi : U \rightarrow V$ , where  $U$  is an open subset of  $M$  and  $V$  is an open subset of  $\mathbb{R}^n$ .

In general, more than one chart is required to cover the entire manifold:

## Examples:

- at least 2 charts are necessary to cover the  $n$ -dimensional sphere  $S^n$  ( $n \geq 1$ ) and the torus  $T^2$
- at least 3 charts are necessary to cover the real projective plane  $\mathbb{R}P^2$

In SageManifolds, an arbitrary number of charts can be introduced

To fully specify the manifold, one shall also provide the *transition maps* on overlapping chart domains (SageManifolds class `CoordChange`)

# Implementing scalar fields

A **scalar field** on manifold  $M$  is a smooth mapping

$$\begin{aligned} f : U \subset M &\longrightarrow \mathbb{R} \\ p &\longmapsto f(p) \end{aligned}$$

where  $U$  is an open subset of  $M$ .

# Implementing scalar fields

A **scalar field** on manifold  $M$  is a smooth mapping

$$\begin{aligned} f : U \subset M &\longrightarrow \mathbb{R} \\ p &\longmapsto f(p) \end{aligned}$$

where  $U$  is an open subset of  $M$ .

A scalar field maps *points*, not *coordinates*, to real numbers

$\implies$  an object  $f$  in the `ScalarField` class has different **coordinate representations** in different charts defined on  $U$ .

# Implementing scalar fields

A **scalar field** on manifold  $M$  is a smooth mapping

$$\begin{aligned} f : U \subset M &\longrightarrow \mathbb{R} \\ p &\longmapsto f(p) \end{aligned}$$

where  $U$  is an open subset of  $M$ .

A scalar field maps *points*, not *coordinates*, to real numbers  
 $\implies$  an object  $f$  in the `ScalarField` class has different **coordinate representations** in different charts defined on  $U$ .

The various coordinate representations  $F, \hat{F}, \dots$  of  $f$  are stored as a *Python dictionary* whose keys are the charts  $C, \hat{C}, \dots$ :

$$f._\text{express} = \{C : F, \hat{C} : \hat{F}, \dots\}$$

$$\text{with } \underbrace{f(p)}_{\text{point}} = F(\underbrace{x^1, \dots, x^n}_{\text{coord. of } p \text{ in chart } C}) = \hat{F}(\underbrace{\hat{x}^1, \dots, \hat{x}^n}_{\text{coord. of } p \text{ in chart } \hat{C}}) = \dots$$



# The scalar field algebra

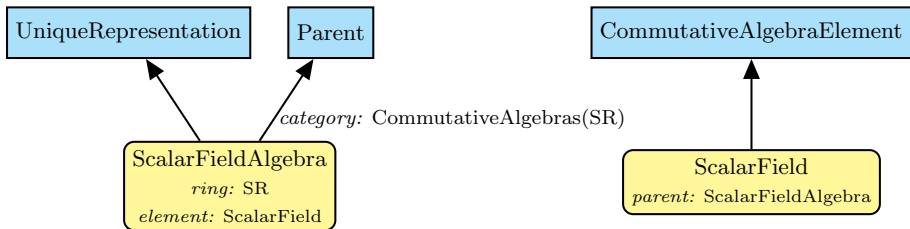
Given an open subset  $U \subset M$ , the set  $C^\infty(U)$  of scalar fields defined on  $U$  has naturally the structure of a **commutative algebra over**  $\mathbb{R}$ :

- 1 it is clearly a vector space over  $\mathbb{R}$
- 2 it is endowed with a commutative ring structure by pointwise multiplication:

$$\forall f, g \in C^\infty(U), \quad \forall p \in U, \quad (f \cdot g)(p) := f(p)g(p)$$

The algebra  $C^\infty(U)$  is implemented in SageManifolds via the class **ScalarFieldAlgebra**.

## Classes for scalar fields



- Native Sage class
- SageManifolds class  
(differential part)

# Outline

- 1 Computer differential geometry and tensor calculus
- 2 SageMath
- 3 The SageManifolds project
- 4 Let us practice!**
- 5 Other examples
- 6 Conclusion and perspectives

# Various ways to install/access SageMath

- **Install on your computer:**

3 options:

- compile from source (Linux, MacOS X):

```
git clone git://github.com/sagemath/sage.git
cd sage
```

```
MAKE='make -j8' make
```

- install a compiled binary version (Linux, MacOS X) from <http://www.sagemath.org/download.html>
- run in virtual machine (Windows)

- **Sage Debian Live USB key:**

<http://sagedebianlive.metelu.net/>

comes along with SageMath (boosted with octave, scilab), Geogebra, LaTeX, gimp, vlc, LibreOffice,...

- **SageMathCloud:**

<https://cloud.sagemath.com/>

- **SageMathCell:**

Single cell mode: <http://sagecell.sagemath.org/>

# Various ways to run SageMath

- **Console mode:**  
run the command `sage`
- **Jupyter Notebook:**  
run the command `sage -n jupyter`  
⇒ worksheet file format: `ipynb`
- **Legacy Sage Notebook:**  
run the command `sage -n`  
⇒ worksheet file format: `sws`
- **SageMathCloud:**  
in your browser, open <https://cloud.sagemath.com/>  
⇒ worksheet file format: `sagews`, `ipynb`

# A full example: Lemaître-Tolman solution

Let us use SageManifolds to derive the Lemaître-Tolman solution for the gravitational collapse of a ball of pressureless matter.

See the worksheet at

[http://nbviewer.jupyter.org/github/egourgoulhon/BHLectures/blob/master/sage/Lemaitre\\_Tolman.ipynb](http://nbviewer.jupyter.org/github/egourgoulhon/BHLectures/blob/master/sage/Lemaitre_Tolman.ipynb)

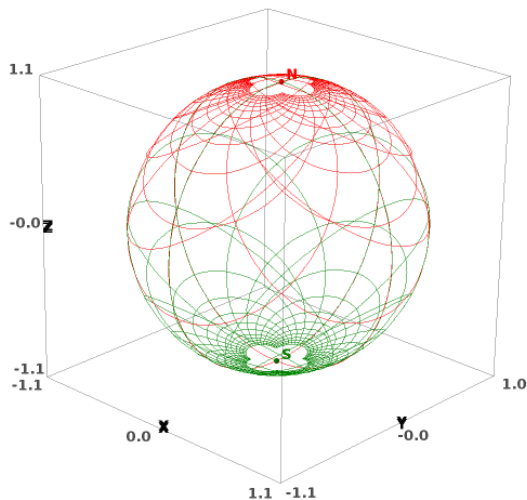
The source is stored at GitHub, from which it can be downloaded:

<https://github.com/egourgoulhon/BHLectures/>

# Outline

- 1 Computer differential geometry and tensor calculus
- 2 SageMath
- 3 The SageManifolds project
- 4 Let us practice!
- 5 Other examples**
- 6 Conclusion and perspectives

# The 2-sphere example



Stereographic coordinates on the 2-sphere

Two charts:

- $X_1: S^2 \setminus \{N\} \rightarrow \mathbb{R}^2$
- $X_2: S^2 \setminus \{S\} \rightarrow \mathbb{R}^2$

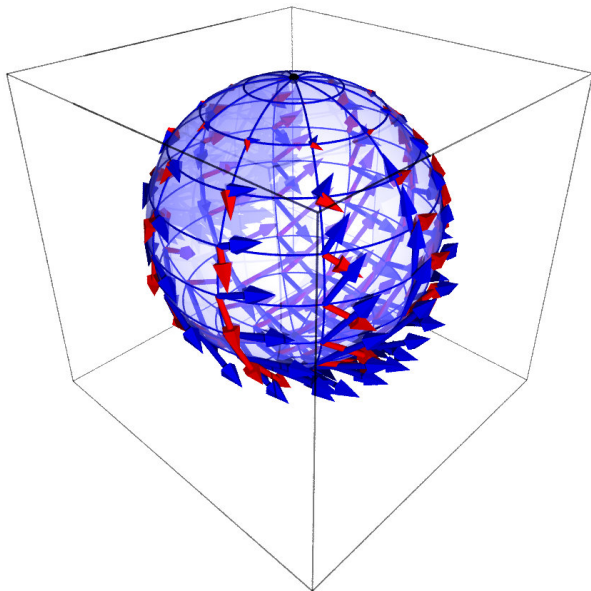
← picture obtained via function `Chart.plot()`

See the worksheet at

[http://sagemanifolds.obspm.fr/examples/html/SM\\_sphere\\_S2.html](http://sagemanifolds.obspm.fr/examples/html/SM_sphere_S2.html)



# The 2-sphere example



Vector frame associated with the stereographic coordinates  $(x, y)$  from the North pole

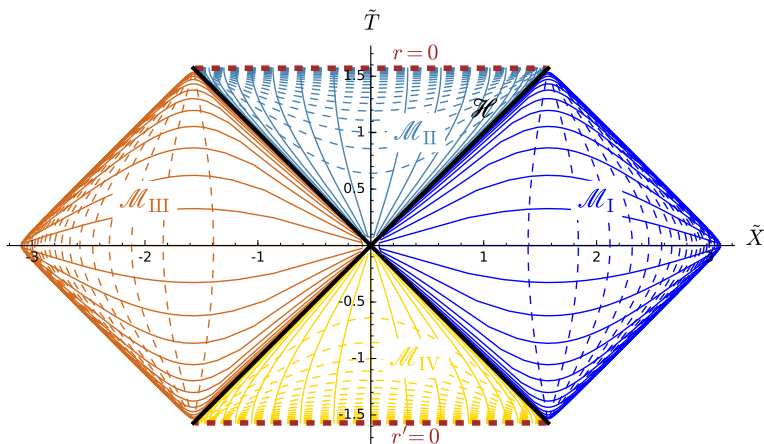
- $\frac{\partial}{\partial x}$
- $\frac{\partial}{\partial y}$

← picture obtained via function

`VectorField.plot()`

# Charts on Schwarzschild spacetime

## The Carter-Penrose diagram



Two charts of standard Schwarzschild-Droste coordinates  $(t, r, \theta, \varphi)$  plotted in terms of compactified coordinates  $(\tilde{T}, \tilde{X}, \theta, \varphi)$ ; see the worksheet at

<http://luth.obspm.fr/~luthier/gourgoulhon/bh16/sage.html>

# Outline

- 1 Computer differential geometry and tensor calculus
- 2 SageMath
- 3 The SageManifolds project
- 4 Let us practice!
- 5 Other examples
- 6 Conclusion and perspectives**

# Conclusion and perspectives

- **SageManifolds** is a **work in progress**
  - ~ 64,000 lines of Python code up to now (including comments and doctests)
- A preliminary version (v0.9.1) is freely available (GPL) at <http://sagemanifolds.obspm.fr/>

# Current status

*Already present (v0.9.1):*

- maps between manifolds, pullback operator
- submanifolds, pushforward operator
- curves in manifolds
- standard tensor calculus (tensor product, contraction, symmetrization, etc.), even on non-parallelizable manifolds
- all monotermin tensor symmetries
- exterior calculus (wedge product, exterior derivative, Hodge duality)
- Lie derivatives of tensor fields
- affine connections, curvature, torsion
- pseudo-Riemannian metrics, Weyl tensor
- some plotting capabilities (charts, points, curves, vector fields)
- parallelization (on tensor components) of CPU demanding computations, via the Python library `multiprocessing`

# Current status

- *Not implemented yet (but should be soon):*
  - extrinsic geometry of pseudo-Riemannian submanifolds
  - computation of geodesics (numerical integration via SageMath/GSL or **Gyoto**)
  - integrals on submanifolds

# Current status

- *Not implemented yet (but should be soon):*
  - extrinsic geometry of pseudo-Riemannian submanifolds
  - computation of geodesics (numerical integration via SageMath/GSL or **Gyoto**)
  - integrals on submanifolds
- *Future prospects:*
  - add more graphical outputs
  - add more functionalities: symplectic forms, fibre bundles, spinors, variational calculus, etc.
  - **connection with numerical relativity: using SageMath to explore numerically-generated spacetimes**

# Integration into SageMath

- The **algebraic part** (tensors on free modules of finite rank) has been submitted to SageMath Trac as ticket [#15916](#) and got a positive review  $\implies$  integrated in SageMath 6.6
- The **differential part** has been split in various tickets for submission to SageMath Trac (cf. the metaticket [#18528](#)); 4 tickets have been already accepted and integrated in SageMath 7.3
- **SageManifolds will be fully integrated in the next release of SageMath: version 7.5**
- Until then, SageManifold has to be downloaded from <http://sagemanifolds.obspm.fr/>
- SageManifolds v0.9.1 is installed in the SageMathCloud (in SageMath 7.4 Jupyter kernel)  $\implies$  open a free account and use it online: <https://cloud.sagemath.com/>

Want to join the project or simply to stay tuned?

visit <http://sagemanifolds.obspm.fr/>  
(download page, documentation, example worksheets, mailing list)