

SageManifolds tutorial

This worksheet provides a short introduction to [SageManifolds](#) (version 1.0, as included in SageMath 7.5).

Click [here](#) to download the worksheet file (ipynb format). To run it, you must start SageMath with the Jupyter notebook, via the command `sage -n jupyter`

The following assumes that you are using version 7.5 (or higher) of SageMath, since lower versions do not include all features of SageManifolds:

```
In [1]: version()
```

```
Out[1]: 'SageMath version 7.5.1, Release Date: 2017-01-15'
```

First we set up the notebook to display mathematical objects using LaTeX rendering:

```
In [2]: %display latex
```

Defining a manifold

As an example let us define a differentiable manifold of dimension 3 over \mathbb{R} :

```
In [3]: M = Manifold(3, 'M', latex_name=r'\mathcal{M}', start_index=1)
```

- The first argument, 3, is the manifold dimension. In SageManifolds, it can be any positive integer.
- The second argument, 'M', is a string defining the manifold's name; it may be different from the symbol set on the left-hand side of the = sign (here M): the latter stands for a mere Python variable, which refers to the manifold object in the computer memory, while the string 'M' is the mathematical symbol chosen for the manifold.
- The optional argument `latex_name=r'\mathcal{M}'` sets the LaTeX symbol to display the manifold. Note the letter 'r' in front on the first quote: it indicates that the string is a *raw* one, so that the backslash character in `\mathcal` is considered as an ordinary character (otherwise, the backslash is used to escape some special characters). If the argument `latex_name` is not provided by the user, it is set to the string used as the second argument (here 'M')
- The optional argument `start_index=1` defines the range of indices to be used for tensor components on the manifold: setting it to 1 means that indices will range in $\{1, 2, 3\}$. The default value is `start_index=0`.

Note that the default base field is \mathbb{R} . If we would have used the optional argument `field='complex'`, we would have defined a manifold over \mathbb{C} . See [list of all options](#) more details.

If we ask for M, it is displayed via its LaTeX symbol:

```
In [4]: M
```

```
Out[4]:  $\mathcal{M}$ 
```

If we use the `print` function instead, we get a short description of the object:

```
In [5]: print(M)
```

3-dimensional differentiable manifold M

Via the command `type`, we get the type of the Python object corresponding to M (here the Python class `DifferentiableManifold_with_category`):

```
In [6]: type(M)
```

```
Out[6]: <class 'sage.manifolds.differentiable.manifold.DifferentiableManifold'>
```

We can also ask for the category of M and see that it is the category of smooth manifolds over \mathbb{R} :

```
In [7]: category(M)
```

```
Out[7]: Smooth $\mathbb{R}$ 
```

The indices on the manifold are generated by the method `irange()`, to be used in loops:

```
In [8]: for i in M.irange():
        print(i)
```

```
1
2
3
```

If the parameter `start_index` had not been specified, the default range of the indices would have been $\{0, 1, 2\}$ instead:

```
In [9]: M0 = Manifold(3, 'M', r'\mathcal{M}')
        for i in M0.irange():
            print(i)
```

```
0
1
2
```

Defining a chart on the manifold

Let us assume that the manifold \mathcal{M} can be covered by a single chart (other cases are discussed below); the chart is declared as follows:

```
In [10]: X.<x,y,z> = M.chart()
```

The writing `.<x,y,z>` in the left-hand side means that the Python variables `x`, `y` and `z` are set to the three coordinates of the chart. This allows one to refer subsequently to the coordinates by their names.

In this example, the function `chart()` has no arguments, which implies that the coordinate symbols will be `x`, `y` and `z` (i.e. exactly the characters set in the `<...>` operator) and that each coordinate range is $(-\infty, +\infty)$. For other cases, an argument must be passed to `chart()` to specify the coordinate symbols and range, as well as the LaTeX symbol of a coordinate if the latter is different from the coordinate name (an example will be provided below).

The chart is displayed as a pair formed by the open set covered by it (here the whole manifold) and the coordinates:

```
In [11]: print(X)
Chart (M, (x, y, z))
```

```
In [12]: X
```

```
Out[12]: ( $\mathcal{M}$ , (x, y, z))
```

The coordinates can be accessed individually, by means of their indices, following the convention defined by `start_index=1` in the manifold's definition:

```
In [13]: X[1]
```

```
Out[13]: x
```

```
In [14]: X[2]
```

```
Out[14]: y
```

```
In [15]: X[3]
```

```
Out[15]: z
```

The full set of coordinates is obtained by means of the operator `[:]`:

```
In [16]: X[:]
```

```
Out[16]: (x, y, z)
```

Thanks to the operator `<x, y, z>` used in the chart declaration, each coordinate can be accessed directly via its name:

```
In [17]: z is X[3]
```

```
Out[17]: True
```

Coordinates are SageMath symbolic expressions:

```
In [18]: type(z)
```

```
Out[18]: <type 'sage.symbolic.expression.Expression'>
```

Functions of the chart coordinates

Real-valued functions of the chart coordinates (mathematically speaking, *functions defined on the chart codomain*) are generated via the method `function()` acting on the chart:

```
In [19]: f = X.function(x+y^2+z^3) ; f
```

```
Out[19]:  $z^3 + y^2 + x$ 
```

```
In [20]: f.display()
```

```
Out[20]:  $(x, y, z) \mapsto z^3 + y^2 + x$ 
```

```
In [21]: f(1,2,3)
```

```
Out[21]: 32
```

They belong to SageManifolds class `CoordFunctionSymb`:

```
In [22]: type(f)
```

```
Out[22]: <class 'sage.manifolds.coord_func_symb.CoordFunctionSymbRing_with_ca
```

and differ from SageMath standard symbolic functions by automatic simplifications in all operations. For instance, adding the two symbolic functions

```
In [23]: f0(x,y,z) = cos(x)^2 ; g0(x,y,z) = sin(x)^2
```

results in

```
In [24]: f0 + g0
```

```
Out[24]: (x, y, z) ↦ cos(x)2 + sin(x)2
```

while the sum of the corresponding functions in the class `CoordFunctionSymb` is automatically simplified:

```
In [25]: f1 = X.function(cos(x)^2) ; g1 = X.function(sin(x)^2)
         f1 + g1
```

```
Out[25]: 1
```

To get the same output with symbolic functions, one has to invoke the method `simplify_trig()`:

```
In [26]: (f0 + g0).simplify_trig()
```

```
Out[26]: (x, y, z) ↦ 1
```

Another difference regards the display; if we ask for the symbolic function `f0`, we get:

```
In [27]: f0
```

```
Out[27]: (x, y, z) ↦ cos(x)2
```

while if we ask for the chart function `f1`, we get only the coordinate expression:

```
In [28]: f1
```

```
Out[28]: cos(x)2
```

To get an output similar to that of `f0`, one should call the method `display()`:

```
In [29]: f1.display()
```

```
Out[29]: (x, y, z) ↦ cos(x)2
```

Note that the method `expr()` returns the underlying symbolic expression:

```
In [30]: f1.expr()
```

```
Out[30]: cos(x)2
```

```
In [31]: type(f1.expr())
```

```
Out[31]: <type 'sage.symbolic.expression.Expression'>
```

Introducing a second chart on the manifold

Let us first consider an open subset of \mathcal{M} , for instance the complement U of the region defined by $\{y = 0, x \geq 0\}$ (note that $(y \neq 0, x < 0)$ stands for $y \neq 0$ OR $x < 0$; the condition $y \neq 0$ AND $x < 0$ would have been written $[y \neq 0, x < 0]$ instead):

```
In [32]: U = M.open_subset('U', coord_def={X: (y!=0, x<0)})
```

Let us call X_U the restriction of the chart X to the open subset U :

```
In [33]: X_U = X.restrict(U) ; X_U
```

```
Out[33]: (U, (x, y, z))
```

We introduce another chart on U , with spherical-type coordinates (r, θ, ϕ) :

```
In [34]: Y.<r,th,ph> = U.chart(r'r:(0,+oo) th:(0,pi):\theta ph:(0,2*pi):\phi') ;
Y
```

```
Out[34]: (U, (r, \theta, \phi))
```

The function `chart()` has now some argument; it is a string, which contains specific LaTeX symbols, hence the prefix 'r' to it (for *raw* string). It also contains the coordinate ranges, since they are different from the default value, which is $(-\infty, +\infty)$. For a given coordinate, the various fields are separated by the character '.' and a space character separates the coordinates. Note that for the coordinate r , there are only two fields, since the LaTeX symbol has not to be specified. The LaTeX symbols are used for the outputs:

```
In [35]: th, ph
```

```
Out[35]: (\theta, \phi)
```

```
In [36]: Y[2], Y[3]
```

```
Out[36]: (\theta, \phi)
```

The declared coordinate ranges are now known to Sage, as we may check by means of the command `assumptions()`:

```
In [37]: assumptions()
```

```
Out[37]: [x is real, y is real, z is real, r is real, r > 0, th is real, \theta > 0,
          \theta < \pi, ph is real, \phi > 0, \phi < 2\pi]
```

They are used in simplifications:

```
In [38]: simplify(abs(r))
```

```
Out[38]: r
```

```
In [39]: simplify(abs(x)) # no simplification occurs since x can take any value
in R
```

```
Out[39]: |x|
```

After having been declared, the chart Y can be fully specified by its relation to the chart X_U , via a transition map:

```
In [40]: transit_Y_to_X = Y.transition_map(X_U, [r*sin(th)*cos(ph), r*sin(th)*si
n(ph), r*cos(th)])
```

```
In [41]: transit_Y_to_X
```

```
Out[41]: (U, (r, theta, phi)) -> (U, (x, y, z))
```

```
In [42]: transit_Y_to_X.display()
```

```
Out[42]: { x = r cos(phi) sin(theta)
          y = r sin(phi) sin(theta)
          z = r cos(theta)
```

The inverse of the transition map can be specified by means of the method `set_inverse()`:

```
In [43]: transit_Y_to_X.set_inverse(sqrt(x^2+y^2+z^2), atan2(sqrt(x^2+y^2), z), a
tan2(y, x))
transit_Y_to_X.inverse().display()
```

```
Out[43]: { r = sqrt(x^2 + y^2 + z^2)
          theta = arctan(sqrt(x^2 + y^2), z)
          phi = arctan(y, x)
```

At this stage, the manifold's **atlas** (the "user atlas", not the maximal atlas!) contains three charts:

```
In [44]: M.atlas()
```

```
Out[44]: [(M, (x, y, z)), (U, (x, y, z)), (U, (r, theta, phi))]
```

The first chart defined on the manifold is considered as the manifold's default chart (it can be changed by the method `set_default_chart()`):

```
In [45]: M.default_chart()
```

```
Out[45]: (M, (x, y, z))
```

Each open subset has its own atlas (since an open subset of a manifold is a manifold by itself):

```
In [46]: U.atlas()
```

```
Out[46]: [(U, (x, y, z)), (U, (r, theta, phi))]
```

```
In [47]: U.default_chart()
```

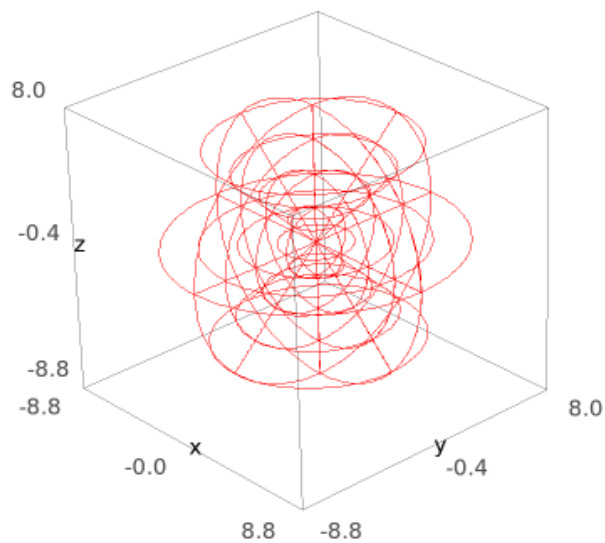
```
Out[47]: (U, (x, y, z))
```

We can draw the chart Y in terms of the chart X . Let us first define a viewer for 3D plots (use 'threejs' or 'jmol' for interactive 3D graphics):

```
In [48]: viewer3D = 'jmol' # must be 'threejs', 'jmol', 'tachyon' or None (default)
```

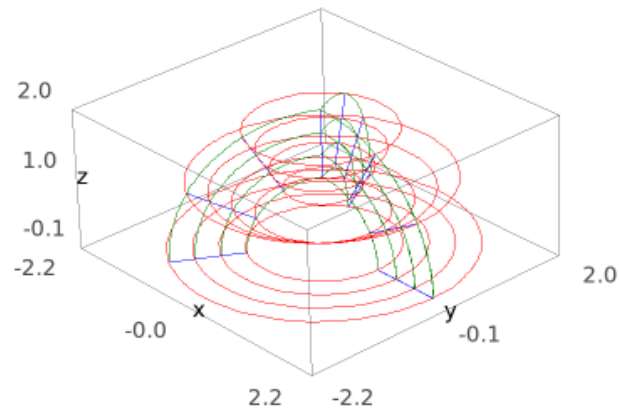
The plot shows lines of constant coordinates from the Y chart in a "Cartesian frame" based on the X coordinates:

```
In [49]: graph = Y.plot(X)
show(graph, viewer=viewer3D)
```



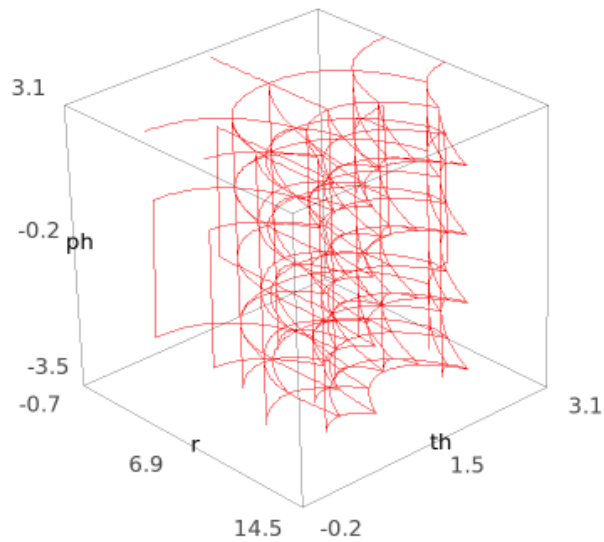
The command `plot()` allows for many options, to control the number of coordinate lines to be drawn, their style and color, as well as the coordinate ranges (cf. the [list of all options](#)):

```
In [50]: graph = Y.plot(X, ranges={r:(1,2), th:(0,pi/2)}, number_values=4,
        color={r:'blue', th:'green', ph:'red'})
        show(graph, aspect_ratio=1, viewer=viewer3D)
```



Conversely, the chart $X|_U$ can be plotted in terms of the chart Y (this is not possible for the whole chart X since its domain is larger than that of chart Y):


```
In [51]: graph = X_U.plot(Y)
show(graph, viewer=viewer3D, axes_labels=['r','theta','phi'])
```



Points on the manifold

A point on \mathcal{M} is defined by its coordinates in a given chart:

```
In [52]: p = M.point((1,2,-1), chart=X, name='p') ; print(p) ; p
```

Point p on the 3-dimensional differentiable manifold M

Out[52]: p

Since $X = (\mathcal{M}, (x, y, z))$ is the manifold's default chart, its name can be omitted:

```
In [53]: p = M.point((1,2,-1), name='p') ; print(p) ; p
```

Point p on the 3-dimensional differentiable manifold M

Out[53]: p

Of course, p belongs to \mathcal{M} :

```
In [54]: p in M
```

Out[54]: True

It is also in U :

```
In [55]: p in U
```

```
Out[55]: True
```

Indeed the coordinates of p have $y \neq 0$:

```
In [56]: p.coord(X)
```

```
Out[56]: (1, 2, -1)
```

Note in passing that since X is the default chart on \mathcal{M} , its name can be omitted in the arguments of `coord()`:

```
In [57]: p.coord()
```

```
Out[57]: (1, 2, -1)
```

The coordinates of p can also be obtained by letting the chart acting of the point (from the very definition of a chart!):

```
In [58]: X(p)
```

```
Out[58]: (1, 2, -1)
```

Let q be a point with $y = 0$ and $x \geq 0$:

```
In [59]: q = M.point((1,0,2), name='q')
```

This time, the point does not belong to U :

```
In [60]: q in U
```

```
Out[60]: False
```

Accordingly, we cannot ask for the coordinates of q in the chart $Y = (U, (r, \theta, \phi))$:

```
In [61]: try:
          q.coord(Y)
        except ValueError as exc:
          print("Error: " + str(exc))
```

```
Error: the point does not belong to the domain of Chart (U, (r, th, ph)
)
```

but we can for point p :

```
In [62]: p.coord(Y)
```

```
Out[62]: ( $\sqrt{3}\sqrt{2}, \pi - \arctan(\sqrt{5}), \arctan(2)$ )
```

```
In [63]: Y(p)
```

```
Out[63]: ( $\sqrt{3}\sqrt{2}, \pi - \arctan(\sqrt{5}), \arctan(2)$ )
```

Points can be compared:

```
In [64]: q == p
```

```
Out[64]: False
```

```
In [65]: p1 = U.point((sqrt(6), pi-atan(sqrt(5)), atan(2)), Y)
p1 == p
```

```
Out[65]: True
```

In SageMath's terminology, points are **elements**, whose **parents** are the manifold on which they have been defined:

```
In [66]: p.parent()
```

```
Out[66]:  $\mathcal{M}$ 
```

```
In [67]: q.parent()
```

```
Out[67]:  $\mathcal{M}$ 
```

```
In [68]: p1.parent()
```

```
Out[68]:  $U$ 
```

Scalar fields

A scalar field is a differentiable mapping $U \rightarrow \mathbb{R}$, where U is an open subset of \mathcal{M} .

The scalar field is defined by its expressions in terms of charts covering its domain (in general more than one chart is necessary to cover all the domain):

```
In [69]: f = U.scalar_field({X_U: x+y^2+z^3}, name='f') ; print(f)
```

```
Scalar field f on the Open subset U of the 3-dimensional differentiable manifold M
```

The coordinate expressions of the scalar field are passed as a Python dictionary, with the charts as keys, hence the writing $\{X_U: x+y^2+z^3\}$.

Since in the present case, there is only one chart in the dictionary, an alternative writing is

```
In [70]: f = U.scalar_field(x+y^2+z^3, chart=X_U, name='f') ; print(f)
```

```
Scalar field f on the Open subset U of the 3-dimensional differentiable manifold M
```

Since X_U is the domain's default chart, it can be omitted in the above declaration:

```
In [71]: f = U.scalar_field(x+y^2+z^3, name='f') ; print(f)
```

```
Scalar field f on the Open subset U of the 3-dimensional differentiable manifold M
```

As a mapping $U \subset \mathcal{M} \rightarrow \mathbb{R}$, a scalar field acts on points, not on coordinates:

```
In [72]: f(p)
```

```
Out[72]: 4
```

The method `display()` provides the expression of the scalar field in terms of a given chart:

```
In [73]: f.display(X_U)
```

```
Out[73]: f: U      -> R
          (x, y, z) -> z^3 + y^2 + x
```

If no argument is provided, the method `display()` shows the coordinate expression of the scalar field in all the charts defined on the domain (except for *subcharts*, i.e. the restrictions of some chart to a subdomain):

```
In [74]: f.display()
```

```
Out[74]: f: U      -> R
          (x, y, z) -> z^3 + y^2 + x
          (r, theta, phi) -> r^3 cos(theta)^3 + r^2 sin(phi)^2 sin(theta)^2 + r cos(phi) sin(theta)
```

Note that the expression of f in terms of the coordinates (r, θ, ϕ) has not been provided by the user but has been automatically computed by means of the change-of-coordinate formula declared above in the transition map.

```
In [75]: f.display(Y)
```

```
Out[75]: f: U      -> R
          (r, theta, phi) -> r^3 cos(theta)^3 + r^2 sin(phi)^2 sin(theta)^2 + r cos(phi) sin(theta)
```

In each chart, the scalar field is represented by a function of the chart coordinates (an object of the type `CoordFunctionSymb` described above), which is accessible via the method `coord_function()`:

```
In [76]: f.coord_function(X_U)
```

```
Out[76]: z^3 + y^2 + x
```

```
In [77]: f.coord_function(X_U).display()
```

```
Out[77]: (x, y, z) -> z^3 + y^2 + x
```

```
In [78]: f.coord_function(Y)
```

```
Out[78]: r^3 cos(theta)^3 + r^2 sin(phi)^2 sin(theta)^2 + r cos(phi) sin(theta)
```

```
In [79]: f.coord_function(Y).display()
```

```
Out[79]: (r, theta, phi) -> r^3 cos(theta)^3 + r^2 sin(phi)^2 sin(theta)^2 + r cos(phi) sin(theta)
```

The "raw" symbolic expression is returned by the method `expr()`:

```
In [80]: f.expr(X_U)
```

```
Out[80]: z^3 + y^2 + x
```

```
In [81]: f.expr(Y)
```

```
Out[81]: r^3 cos(theta)^3 + r^2 sin(phi)^2 sin(theta)^2 + r cos(phi) sin(theta)
```

```
In [82]: f.expr(Y) is f.coord_function(Y).expr()
```

```
Out[82]: True
```

A scalar field can also be defined by some unspecified function of the coordinates:

```
In [83]: h = U.scalar_field(function('H')(x, y, z), name='h') ; print(h)
```

Scalar field h on the Open subset U of the 3-dimensional differentiable manifold M

```
In [84]: h.display()
```

```
Out[84]: h : U      -> R
          (x, y, z)  -> H(x, y, z)
          (r, theta, phi) -> H(r cos(phi) sin(theta), r sin(phi) sin(theta), r cos(theta))
```

```
In [85]: h.display(Y)
```

```
Out[85]: h : U      -> R
          (r, theta, phi) -> H(r cos(phi) sin(theta), r sin(phi) sin(theta), r cos(theta))
```

```
In [86]: h(p) # remember that p is the point of coordinates (1,2,-1) in the char
          t X_U
```

```
Out[86]: H(1, 2, -1)
```

The parent of f is the set $C^\infty(U)$ of all smooth scalar fields on U , which is a commutative algebra over \mathbb{R} :

```
In [87]: CU = f.parent() ; CU
```

```
Out[87]: C^\infty(U)
```

```
In [88]: print(CU)
```

Algebra of differentiable scalar fields on the Open subset U of the 3-dimensional differentiable manifold M

```
In [89]: CU.category()
```

```
Out[89]: CommutativeAlgebrasSR
```

The base ring of the algebra is the field \mathbb{R} , which is represented here by SageMath's Symbolic Ring (SR):

```
In [90]: CU.base_ring()
```

```
Out[90]: SR
```

Arithmetic operations on scalar fields are defined through the algebra structure:

```
In [91]: s = f + 2*h ; print(s)
```

Scalar field on the Open subset U of the 3-dimensional differentiable manifold M

```
In [92]: s.display()
```

```
Out[92]: U      -> R
          (x, y, z) -> z^3 + y^2 + x + 2 H(x, y, z)
          (r, theta, phi) -> r^3 cos(theta)^3 + r^2 sin(phi)^2 sin(theta)^2 + r cos(phi) sin(theta) + 2
                               H(r cos(phi) sin(theta), r sin(phi) sin(theta), r cos(theta))
```

Tangent spaces

The tangent vector space to the manifold at point p is obtained as follows:

```
In [93]: Tp = M.tangent_space(p) ; Tp
```

```
Out[93]: T_p M
```

```
In [94]: print(Tp)
```

Tangent space at Point p on the 3-dimensional differentiable manifold M

$T_p M$ is a 2-dimensional vector space over \mathbb{R} (represented here by SageMath's Symbolic Ring (SR)):

```
In [95]: print(Tp.category())
```

Category of finite dimensional vector spaces over Symbolic Ring

```
In [96]: Tp.dim()
```

```
Out[96]: 3
```

$T_p M$ is automatically endowed with vector bases deduced from the vector frames defined around the point:

```
In [97]: Tp.bases()
```

```
Out[97]: [(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}), (\frac{\partial}{\partial r}, \frac{\partial}{\partial \theta}, \frac{\partial}{\partial \phi})]
```

For the tangent space at the point q , on the contrary, there is only one pre-defined basis, since q is not in the domain U of the frame associated with coordinates (r, θ, ϕ) :

```
In [98]: Tq = M.tangent_space(q)
          Tq.bases()
```

```
Out[98]: [(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z})]
```

A random element:

```
In [99]: v = Tp.an_element() ; print(v)
```

Tangent vector at Point p on the 3-dimensional differentiable manifold M

```
In [100]: v.display()
```

```
Out[100]:  $\frac{\partial}{\partial x} + 2\frac{\partial}{\partial y} + 3\frac{\partial}{\partial z}$ 
```

```
In [101]: u = Tq.an_element() ; print(u)
```

Tangent vector at Point q on the 3-dimensional differentiable manifold M

```
In [102]: u.display()
```

```
Out[102]:  $\frac{\partial}{\partial x} + 2\frac{\partial}{\partial y} + 3\frac{\partial}{\partial z}$ 
```

Note that, despite what the above simplified writing may suggest (the mention of the point p or q is omitted in the basis vectors), u and v are different vectors, for they belong to different vector spaces:

```
In [103]: v.parent()
```

```
Out[103]:  $T_p \mathcal{M}$ 
```

```
In [104]: u.parent()
```

```
Out[104]:  $T_q \mathcal{M}$ 
```

In particular, it is not possible to add u and v :

```
In [105]: try:
           s = u + v
         except TypeError as exc:
           print("Error: " + str(exc))
```

Error: unsupported operand parent(s) for '+': 'Tangent space at Point q on the 3-dimensional differentiable manifold M' and 'Tangent space at Point p on the 3-dimensional differentiable manifold M'

Vector Fields

Each chart defines a vector frame on the chart domain: the so-called **coordinate basis**:

```
In [106]: X.frame()
```

```
Out[106]:  $(\mathcal{M}, \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right))$ 
```

```
In [107]: X.frame().domain() # this frame is defined on the whole manifold
```

```
Out[107]:  $\mathcal{M}$ 
```

```
In [108]: Y.frame()
```

```
Out[108]:  $(U, \left(\frac{\partial}{\partial r}, \frac{\partial}{\partial \theta}, \frac{\partial}{\partial \phi}\right))$ 
```

```
In [109]: Y.frame().domain() # this frame is defined only on U
```

```
Out[109]:  $U$ 
```

The list of frames defined on a given open subset is returned by the method `frames()`:

```
In [110]: M.frames()
```

```
Out[110]: [(M, (∂/∂x, ∂/∂y, ∂/∂z)), (U, (∂/∂x, ∂/∂y, ∂/∂z)), (U, (∂/∂r, ∂/∂θ, ∂/∂φ))]
```

```
In [111]: U.frames()
```

```
Out[111]: [(U, (∂/∂x, ∂/∂y, ∂/∂z)), (U, (∂/∂r, ∂/∂θ, ∂/∂φ))]
```

```
In [112]: M.default_frame()
```

```
Out[112]: (M, (∂/∂x, ∂/∂y, ∂/∂z))
```

Unless otherwise specified (via the command `set_default_frame()`), the default frame is that associated with the default chart:

```
In [113]: M.default_frame() is M.default_chart().frame()
```

```
Out[113]: True
```

```
In [114]: U.default_frame() is U.default_chart().frame()
```

```
Out[114]: True
```

Individual elements of a frame can be accessed by means of their indices:

```
In [115]: e = U.default_frame() ; e2 = e[2] ; e2
```

```
Out[115]: ∂/∂y
```

```
In [116]: print(e2)
```

Vector field d/dy on the Open subset U of the 3-dimensional differentiable manifold M

We may define a new vector field as follows:

```
In [117]: v = e[2] + 2*x*e[3] ; print(v)
```

Vector field on the Open subset U of the 3-dimensional differentiable manifold M

```
In [118]: v.display()
```

```
Out[118]: ∂/∂y + 2x ∂/∂z
```

A vector field can be defined by its components with respect to a given vector frame. When the latter is not specified, the open set's default frame is of course assumed:


```
In [119]: v = U.vector_field(name='v') # vector field defined on the open set U
v[1] = 1+y
v[2] = -x
v[3] = x*y*z
v.display()
```

```
Out[119]: v = (y + 1) \frac{\partial}{\partial x} - x \frac{\partial}{\partial y} + xyz \frac{\partial}{\partial z}
```

Vector fields on U are Sage *element* objects, whose *parent* is the set $\mathcal{X}(U)$ of vector fields defined on U :

```
In [120]: v.parent()
```

```
Out[120]:  $\mathcal{X}(U)$ 
```

The set $\mathcal{X}(U)$ is a module over the commutative algebra $C^\infty(U)$ of scalar fields on U :

```
In [121]: print(v.parent())
```

```
Free module X(U) of vector fields on the Open subset U of the 3-dimensional differentiable manifold M
```

```
In [122]: print(v.parent().category())
```

```
Category of finite dimensional modules over Algebra of differentiable scalar fields on the Open subset U of the 3-dimensional differentiable manifold M
```

```
In [123]: v.parent().base_ring()
```

```
Out[123]:  $C^\infty(U)$ 
```

A vector field acts on scalar fields:

```
In [124]: f.display()
```

```
Out[124]: f: U      -> R
          (x, y, z)  -> z^3 + y^2 + x
          (r, theta, phi) -> r^3 cos(theta)^3 + r^2 sin(phi)^2 sin(theta)^2 + r cos(phi) sin(theta)
```

```
In [125]: s = v(f) ; print(s)
```

```
Scalar field v(f) on the Open subset U of the 3-dimensional differentiable manifold M
```

```
In [126]: s.display()
```

```
Out[126]: v(f): U      -> R
          (x, y, z)  -> 3xyz^3 - (2x - 1)y + 1
          (r, theta, phi) -> -3r^5 cos(phi) cos(theta)^5 sin(phi) + 3r^5 cos(phi) cos(theta)^3 sin(phi) - 2(theta)^2 + r sin(phi) sin(theta) + 1
```

```
In [127]: e[3].display()
```

```
Out[127]: \frac{\partial}{\partial z} = \frac{\partial}{\partial z}
```

```
In [128]: e[3](f).display()
```

$$\begin{aligned} \text{Out[128]: } \frac{\partial}{\partial z}(f) : U &\longrightarrow \mathbb{R} \\ (x, y, z) &\longmapsto 3z^2 \\ (r, \theta, \phi) &\longmapsto 3r^2 \cos(\theta)^2 \end{aligned}$$

Unset components are assumed to be zero:

```
In [129]: w = U.vector_field(name='w')
w[2] = 3
w.display()
```

$$\text{Out[129]: } w = 3 \frac{\partial}{\partial y}$$

A vector field on U can be expanded in the vector frame associated with the chart (r, θ, ϕ) :

```
In [130]: v.display(Y.frame())
```

$$\begin{aligned} \text{Out[130]: } v = &\left(\frac{xyz^2 + x}{\sqrt{x^2 + y^2 + z^2}} \right) \frac{\partial}{\partial r} + \left(-\frac{(x^3y + xy^3 - x)\sqrt{x^2 + y^2}z}{x^4 + 2x^2y^2 + y^4 + (x^2 + y^2)z^2} \right) \frac{\partial}{\partial \theta} \\ &+ \left(-\frac{x^2 + y^2 + y}{x^2 + y^2} \right) \frac{\partial}{\partial \phi} \end{aligned}$$

By default, the components are expressed in terms of the default coordinates (x, y, z) . To express them in terms of the coordinates (r, θ, ϕ) , one should add the corresponding chart as the second argument of the method `display()`:

```
In [131]: v.display(Y.frame(), Y)
```

$$\begin{aligned} \text{Out[131]: } v = &(r^3 \cos(\phi) \cos(\theta)^2 \sin(\phi) \sin(\theta)^2 + \cos(\phi) \sin(\theta)) \frac{\partial}{\partial r} \\ &+ \left(-\frac{r^3 \cos(\phi) \cos(\theta) \sin(\phi) \sin(\theta)^3 - \cos(\phi) \cos(\theta)}{r} \right) \frac{\partial}{\partial \theta} \\ &+ \left(-\frac{r \sin(\theta) + \sin(\phi)}{r \sin(\theta)} \right) \frac{\partial}{\partial \phi} \end{aligned}$$

```
In [132]: for i in M.irange():
show(e[i].display(Y.frame(), Y))
```

$$\begin{aligned} \frac{\partial}{\partial x} &= \cos(\phi) \sin(\theta) \frac{\partial}{\partial r} + \frac{\cos(\phi) \cos(\theta)}{r} \frac{\partial}{\partial \theta} - \frac{\sin(\phi)}{r \sin(\theta)} \frac{\partial}{\partial \phi} \\ \frac{\partial}{\partial y} &= \sin(\phi) \sin(\theta) \frac{\partial}{\partial r} + \frac{\cos(\theta) \sin(\phi)}{r} \frac{\partial}{\partial \theta} + \frac{\cos(\phi)}{r \sin(\theta)} \frac{\partial}{\partial \phi} \\ \frac{\partial}{\partial z} &= \cos(\theta) \frac{\partial}{\partial r} - \frac{\sin(\theta)}{r} \frac{\partial}{\partial \theta} \end{aligned}$$

The components of a tensor field w.r.t. the default frame can also be obtained as a list, via the command `[:]`:

In [133]: `v[:]`

Out[133]: $[y + 1, -x, xyz]$

An alternative is to use the method `display_comp()`:

In [134]: `v.display_comp()`

Out[134]:
$$\begin{aligned} v^x &= y + 1 \\ v^y &= -x \\ v^z &= xyz \end{aligned}$$

To obtain the components w.r.t. to another frame, one may go through the method `comp()` and specify the frame:

In [135]: `v.comp(Y.frame())[:]`

Out[135]:
$$\left[\frac{xyz^2 + x}{\sqrt{x^2 + y^2 + z^2}}, -\frac{(x^3y + xy^3 - x)\sqrt{x^2 + y^2}z}{x^4 + 2x^2y^2 + y^4 + (x^2 + y^2)z^2}, -\frac{x^2 + y^2 + y}{x^2 + y^2} \right]$$

However a shortcut is to provide the frame as the first argument of the square brackets:

In [136]: `v[Y.frame(), :]`

Out[136]:
$$\left[\frac{xyz^2 + x}{\sqrt{x^2 + y^2 + z^2}}, -\frac{(x^3y + xy^3 - x)\sqrt{x^2 + y^2}z}{x^4 + 2x^2y^2 + y^4 + (x^2 + y^2)z^2}, -\frac{x^2 + y^2 + y}{x^2 + y^2} \right]$$

In [137]: `v.display_comp(Y.frame())`

Out[137]:
$$\begin{aligned} v^r &= \frac{xyz^2 + x}{\sqrt{x^2 + y^2 + z^2}} \\ v^\theta &= -\frac{(x^3y + xy^3 - x)\sqrt{x^2 + y^2}z}{x^4 + 2x^2y^2 + y^4 + (x^2 + y^2)z^2} \\ v^\phi &= -\frac{x^2 + y^2 + y}{x^2 + y^2} \end{aligned}$$

Components are shown expressed in terms of the default's coordinates; to get them in terms of the coordinates (r, θ, ϕ) instead, add the chart name as the last argument in the square brackets:

In [138]: `v[Y.frame(), :, Y]`

Out[138]:
$$\left[r^3 \cos(\phi) \cos(\theta)^2 \sin(\phi) \sin(\theta)^2 + \cos(\phi) \sin(\theta), -\frac{r^3 \cos(\phi) \cos(\theta) \sin(\phi) \sin(\theta)^3 - \cos(\phi) \cos(\theta)}{r}, -\frac{r \sin(\theta) + \sin(\phi)}{r \sin(\theta)} \right]$$

or specify the chart in `display_comp()`:

```
In [139]: v.display_comp(Y.frame(), chart=Y)
```

```
Out[139]:
```

$$\begin{aligned} v^r &= r^3 \cos(\phi) \cos(\theta)^2 \sin(\phi) \sin(\theta)^2 + \cos(\phi) \sin(\theta) \\ v^\theta &= -\frac{r^3 \cos(\phi) \cos(\theta) \sin(\phi) \sin(\theta)^3 - \cos(\phi) \cos(\theta)}{r} \\ v^\phi &= -\frac{r \sin(\theta) + \sin(\phi)}{r \sin(\theta)} \end{aligned}$$

To get some vector component as a scalar field instead of a coordinate expression, use double square brackets:

```
In [140]: print(v[[1]])
```

Scalar field on the Open subset U of the 3-dimensional differentiable manifold M

```
In [141]: v[[1]].display()
```

```
Out[141]:
```

$$\begin{aligned} U &\longrightarrow \mathbb{R} \\ (x, y, z) &\longmapsto y + 1 \\ (r, \theta, \phi) &\longmapsto r \sin(\phi) \sin(\theta) + 1 \end{aligned}$$

```
In [142]: v[[1]].expr(X_U)
```

```
Out[142]:
```

$$y + 1$$

A vector field can be defined with components being unspecified functions of the coordinates:

```
In [143]: u = U.vector_field(name='u')
u[:] = [function('u_x')(x,y,z), function('u_y')(x,y,z), function('u_z')(x,y,z)]
u.display()
```

```
Out[143]:
```

$$u = u_x(x, y, z) \frac{\partial}{\partial x} + u_y(x, y, z) \frac{\partial}{\partial y} + u_z(x, y, z) \frac{\partial}{\partial z}$$

```
In [144]: s = v + u ; s.set_name('s') ; s.display()
```

```
Out[144]:
```

$$s = (y + u_x(x, y, z) + 1) \frac{\partial}{\partial x} + (-x + u_y(x, y, z)) \frac{\partial}{\partial y} + (xyz + u_z(x, y, z)) \frac{\partial}{\partial z}$$

Values of vector fields at a given point

The value of a vector field at some point of the manifold is obtained via the method `at()`:

```
In [145]: vp = v.at(p) ; print(vp)
```

Tangent vector v at Point p on the 3-dimensional differentiable manifold M

```
In [146]: vp.display()
```

```
Out[146]:
```

$$v = 3 \frac{\partial}{\partial x} - \frac{\partial}{\partial y} - 2 \frac{\partial}{\partial z}$$

Indeed, recall that, w.r.t. chart $X_U=(x, y, z)$, the coordinates of the point p and the components of the vector field v are

```
In [147]: p.coord(X_U)
```

```
Out[147]: (1, 2, -1)
```

```
In [148]: v.display(X_U.frame(), X_U)
```

```
Out[148]:  $v = (y + 1) \frac{\partial}{\partial x} - x \frac{\partial}{\partial y} + xyz \frac{\partial}{\partial z}$ 
```

Note that to simplify the writing, the symbol used to denote the value of the vector field at point p is the same as that of the vector field itself (namely v); this can be changed by the method `set_name()`:

```
In [149]: vp.set_name(latex_name='v|_p')
vp.display()
```

```
Out[149]:  $v|_p = 3 \frac{\partial}{\partial x} - \frac{\partial}{\partial y} - 2 \frac{\partial}{\partial z}$ 
```

Of course, $v|_p$ belongs to the tangent space at p :

```
In [150]: vp.parent()
```

```
Out[150]:  $T_p \mathcal{M}$ 
```

```
In [151]: vp in M.tangent_space(p)
```

```
Out[151]: True
```

```
In [152]: up = u.at(p) ; print(up)
```

Tangent vector u at Point p on the 3-dimensional differentiable manifold M

```
In [153]: up.display()
```

```
Out[153]:  $u = u_x(1, 2, -1) \frac{\partial}{\partial x} + u_y(1, 2, -1) \frac{\partial}{\partial y} + u_z(1, 2, -1) \frac{\partial}{\partial z}$ 
```

1-forms

A 1-form on \mathcal{M} is a field of linear forms. For instance, it can be the **differential of a scalar field**:

```
In [154]: df = f.differential() ; print(df)
```

1-form df on the Open subset U of the 3-dimensional differentiable manifold M

```
In [155]: df.display()
```

```
Out[155]:  $df = dx + 2ydy + 3z^2dz$ 
```

In the above writing, the 1-form is expanded over the basis (dx, dy, dz) associated with the chart (x, y, z) . This basis can be accessed via the method `coframe()`:

```
In [156]: dX = X.coframe() ; dX
```

```
Out[156]: ( $\mathcal{M}$ , (dx, dy, dz))
```

The list of all coframes defined on a given manifold open subset is returned by the method `coframes()`:

```
In [157]: M.coframes()
```

```
Out[157]: [( $\mathcal{M}$ , (dx, dy, dz)), (U, (dx, dy, dz)), (U, (dr, d $\theta$ , d $\phi$ ))]
```

As for a vector field, the value of the differential form at some point on the manifold is obtained by the method `at()`:

```
In [158]: dfp = df.at(p) ; print(dfp)
```

Linear form df on the Tangent space at Point p on the 3-dimensional differentiable manifold M

```
In [159]: dfp.display()
```

```
Out[159]:  $df = dx + 4dy + 3dz$ 
```

Recall that

```
In [160]: p.coord()
```

```
Out[160]: (1, 2, -1)
```

The linear form $df|_p$ belongs to the dual of the tangent vector space at p :

```
In [161]: dfp.parent()
```

```
Out[161]:  $T_p \mathcal{M}^*$ 
```

```
In [162]: dfp.parent() is M.tangent_space(p).dual()
```

```
Out[162]: True
```

As such, it is acting on vectors at p , yielding a real number:

```
In [163]: print(vp) ; vp.display()
```

Tangent vector v at Point p on the 3-dimensional differentiable manifold M

```
Out[163]:  $v|_p = 3\frac{\partial}{\partial x} - \frac{\partial}{\partial y} - 2\frac{\partial}{\partial z}$ 
```

```
In [164]: dfp(vp)
```

```
Out[164]: -7
```

In [165]: `print(up) ; up.display()`

Tangent vector u at Point p on the 3-dimensional differentiable manifold M

Out[165]: $u = u_x(1, 2, -1) \frac{\partial}{\partial x} + u_y(1, 2, -1) \frac{\partial}{\partial y} + u_z(1, 2, -1) \frac{\partial}{\partial z}$

In [166]: `dfp(up)`

Out[166]: $u_x(1, 2, -1) + 4 u_y(1, 2, -1) + 3 u_z(1, 2, -1)$

The differential 1-form of the unspecified scalar field h :

In [167]: `h.display() ; dh = h.differential() ; dh.display()`

Out[167]: $dh = \frac{\partial H}{\partial x} dx + \frac{\partial H}{\partial y} dy + \frac{\partial H}{\partial z} dz$

A 1-form can also be defined from scratch:

In [168]: `om = U.one_form('omega', r'\omega') ; print(om)`

1-form ω on the Open subset U of the 3-dimensional differentiable manifold M

It can be specified by providing its components in a given coframe:

In [169]: `om[:] = [x^2+y^2, z, x-z] # components in the default coframe (dx, dy, dz)`
`om.display()`

Out[169]: $\omega = (x^2 + y^2) dx + z dy + (x - z) dz$

Of course, one may set the components in a frame different from the default one:

In [170]: `om[Y.frame(), :, Y] = [r*sin(th)*cos(ph), 0, r*sin(th)*sin(ph)]`
`om.display(Y.frame(), Y)`

Out[170]: $\omega = r \cos(\phi) \sin(\theta) dr + r \sin(\phi) \sin(\theta) d\phi$

The components in the coframe (dx, dy, dz) are updated automatically:

In [171]: `om.display()`

Out[171]:
$$\omega = \left(\frac{x^4 + x^2 y^2 - \sqrt{x^2 + y^2 + z^2} y^2}{\sqrt{x^2 + y^2 + z^2} (x^2 + y^2)} \right) dx$$

$$+ \left(\frac{x^3 y + x y^3 + \sqrt{x^2 + y^2 + z^2} x y}{\sqrt{x^2 + y^2 + z^2} (x^2 + y^2)} \right) dy + \left(\frac{x z}{\sqrt{x^2 + y^2 + z^2}} \right) dz$$

Let us revert to the values set previously:

In [172]: `om[:] = [x^2+y^2, z, x-z]`
`om.display()`

Out[172]: $\omega = (x^2 + y^2) dx + z dy + (x - z) dz$

This time, the components in the coframe $(dr, d\theta, d\phi)$ are those that are updated:

```
In [173]: om.display(Y.frame(), Y)
```

```
Out[173]: 
$$\omega = (r^2 \cos(\phi) \sin(\theta)^3 + r(\cos(\phi) + \sin(\phi)) \cos(\theta) \sin(\theta) - r \cos(\theta)^2) dr$$


$$+ (r^2 \cos(\theta)^2 \sin(\phi) + r^2 \cos(\theta) \sin(\theta) + (r^3 \cos(\phi) \cos(\theta) - r^2 \cos(\phi)) \sin(\theta)^2) d\theta$$


$$+ (-r^3 \sin(\phi) \sin(\theta)^3 + r^2 \cos(\phi) \cos(\theta) \sin(\theta)) d\phi$$

```

A 1-form acts on vector fields, resulting in a scalar field:

```
In [174]: v.display(); om.display(); print(om(v)); om(v).display()
```

Scalar field $\omega(v)$ on the Open subset U of the 3-dimensional differentiable manifold M

```
Out[174]: 
$$\omega(v): U \longrightarrow \mathbb{R}$$


$$(x, y, z) \longmapsto -xyz^2 + x^2y + y^3 + x^2 + y^2 + (x^2y - x)z$$


$$(r, \theta, \phi) \longmapsto -r^2 \cos(\phi) \cos(\theta) \sin(\theta) + (r^4 \cos(\phi)^2 \cos(\theta) \sin(\phi) + r^3 \sin(\phi) \cos(\theta) \sin(\theta)^2) - (r^4 \cos(\phi) \cos(\theta)^2 \sin(\phi) - r^2) \sin(\theta)^2$$

```

```
In [175]: df.display(); print(df(v)); df(v).display()
```

Scalar field $df(v)$ on the Open subset U of the 3-dimensional differentiable manifold M

```
Out[175]: 
$$df(v): U \longrightarrow \mathbb{R}$$


$$(x, y, z) \longmapsto 3xyz^3 - (2x - 1)y + 1$$


$$(r, \theta, \phi) \longmapsto r \sin(\phi) \sin(\theta) + (3r^5 \cos(\phi) \cos(\theta)^3 \sin(\phi) - 2r^2 \cos(\phi) \cos(\theta) \sin(\theta)^2) \sin(\theta)$$

```

```
In [176]: u.display(); om(u).display()
```

```
Out[176]: 
$$\omega(u): U \longrightarrow \mathbb{R}$$


$$(x, y, z) \longmapsto x^2 u_x(x, y, z) + y^2 u_x(x, y, z) + z(u_y(x, y, z) - u_z(x, y, z)) + y^2 u_y(x, y, z) + x^2 u_y(x, y, z) + x^2 u_z(x, y, z) + y^2 u_z(x, y, z) + z^2 u_z(x, y, z)$$


$$(r, \theta, \phi) \longmapsto r^2 \sin(\theta)^2 u_x(r \cos(\phi) \sin(\theta), r \sin(\phi) \sin(\theta), r \cos(\theta)) + r^2 \sin(\theta)^2 u_y(r \cos(\phi) \sin(\theta), r \sin(\phi) \sin(\theta), r \cos(\theta)) + (r \cos(\phi) \sin(\theta) - r \cos(\theta)) u_z(r \cos(\phi) \sin(\theta), r \sin(\phi) \sin(\theta), r \cos(\theta))$$

```

In the case of a differential 1-form, the following identity holds:

```
In [177]: df(v) == v(f)
```

```
Out[177]: True
```

1-forms are Sage *element* objects, whose *parent* is the $C^\infty(U)$ -module $\Lambda^1(U)$ of all 1-forms defined on U :

```
In [178]: df.parent()
```

```
Out[178]:  $\Lambda^1(U)$ 
```

```
In [179]: print(df.parent())
```

Free module $\Lambda^1(U)$ of 1-forms on the Open subset U of the 3-dimensional differentiable manifold M


```
In [180]: print(om.parent())
```

Free module $\wedge^1(U)$ of 1-forms on the Open subset U of the 3-dimensional differentiable manifold M

$\Lambda^1(U)$ is actually the dual of the free module $\mathcal{X}(U)$:

```
In [181]: df.parent() is v.parent().dual()
```

```
Out[181]: True
```

Differential forms and exterior calculus

The **exterior product** of two 1-forms is taken via the method `wedge()` and results in a 2-form:

```
In [182]: a = om.wedge(df) ; print(a) ; a.display()
```

2-form omega/\df on the Open subset U of the 3-dimensional differentiable manifold M

```
Out[182]:
```

$$\omega \wedge df = (2x^2y + 2y^3 - z) dx \wedge dy + (3(x^2 + y^2)z^2 - x + z) dx \wedge dz + (3z^3 - 2xy + 2yz) dy \wedge dz$$

A matrix view of the components:

```
In [183]: a[:]
```

```
Out[183]:
```

$$\begin{pmatrix} 0 & 2x^2y + 2y^3 - z & 3(x^2 + y^2)z^2 - x + z \\ -2x^2y - 2y^3 + z & 0 & 3z^3 - 2xy + 2yz \\ -3(x^2 + y^2)z^2 + x - z & -3z^3 + 2xy - 2yz & 0 \end{pmatrix}$$

Displaying only the non-vanishing components, skipping the redundant ones (i.e. those that can be deduced by antisymmetry):

```
In [184]: a.display_comp(only_nonredundant=True)
```

```
Out[184]:
```

$$\begin{aligned} \omega \wedge df_{xy} &= 2x^2y + 2y^3 - z \\ \omega \wedge df_{xz} &= 3(x^2 + y^2)z^2 - x + z \\ \omega \wedge df_{yz} &= 3z^3 - 2xy + 2yz \end{aligned}$$

The 2-form $\omega \wedge df$ can be expanded on the $(dr, d\theta, d\phi)$ coframe:

In [185]: `a.display(Y.frame(), Y)`

Out[185]:

$$\begin{aligned}
 & \omega \wedge df \\
 &= (3 r^5 \cos(\phi) \sin(\theta)^4 \\
 &\quad - (3 r^5 \cos(\phi) - 3 r^4 \cos(\theta) \sin(\phi) - 2 r^3 \cos(\phi) \sin(\phi)^2) \sin(\theta)^2 \\
 &\quad - (3 r^4 \sin(\phi) + r^2 \cos(\phi)) \cos(\theta) - (2 r^3 \cos(\theta) \sin(\phi)^2 + (\sin(\phi)^2 - 1) r^2) \sin(\theta)) \cos(\theta) \\
 &\quad \wedge d\theta \\
 &\quad + (2 r^4 \sin(\phi) \sin(\theta)^5 + (3 r^5 \cos(\theta)^3 \sin(\phi) + 2 r^3 \cos(\phi)^2 \cos(\theta) \sin(\phi)) \sin(\theta) \\
 &\quad (\theta)^3 - (2 r^3 \cos(\phi) \cos(\theta)^2 \sin(\phi) + (\cos(\phi) \sin(\phi) + 1) r^2 \cos(\theta)) \sin(\theta)^2 \\
 &\quad - (3 r^4 \cos(\phi) \cos(\theta)^4 - r^2 \cos(\theta)^2 \sin(\phi)) \sin(\theta)) dr \wedge d\phi \\
 &\quad + (-r^3 \cos(\theta)^2 \sin(\theta) \\
 &\quad - (3 r^6 \cos(\theta)^2 \sin(\phi) + 2 r^4 \cos(\phi)^2 \sin(\phi) - 2 r^5 \cos(\theta) \sin(\phi)) \sin(\theta)^4 \\
 &\quad + (2 r^4 \cos(\phi) \cos(\theta) \sin(\phi) + r^3 \cos(\phi) \sin(\phi)) \sin(\theta)^3 \\
 &\quad + (3 r^5 \cos(\phi) \cos(\theta)^3 - r^3 \cos(\theta) \sin(\phi)) \sin(\theta)^2) d\theta \wedge d\phi
 \end{aligned}$$

As a 2-form, $A := \omega \wedge df$ can be applied to a pair of vectors and is antisymmetric:

In [186]: `a.set_name('A')`
`print(a(u,v)) ; a(u,v).display()`

Scalar field A(u,v) on the Open subset U of the 3-dimensional differentiable manifold M

Out[186]: $A(u, v) : U \longrightarrow \mathbb{R}$

$$\begin{aligned}
 (x, y, z) &\longmapsto 3xyz^4u_y(x, y, z) - 2x^2y^2u_y(x, y, z) - 2y^4u_x \\
 &\quad (xu_x(x, y, z) + u_y(x, y, z))y^3 + 3(x^3yu_x(x, y, z) + xy^3u_x \\
 &\quad - (3y^3u_z(x, y, z) - (2xu_y(x, y, z) - 3u_z(x, y, z))y) \\
 &\quad + (3x^2u_z(x, y, z) - xu_x(x, y, z))y) \\
 &\quad - (2x^3u_x(x, y, z) + 2x^2u_y(x, y, z) + (2x^2 - \\
 &\quad - (2x^2y^2u_y(x, y, z) + (x^2u_x(x, y, z) - (2x - 1)u_z(x \\
 &\quad - xu_x(x, y, z) - u_y(x, y, z) + u_z(x, y, z)) \\
 &\quad + xu_z(x, y, z)
 \end{aligned}$$

$$\begin{aligned}
 (r, \theta, \phi) &\longmapsto (r^4 \cos(\phi) \cos(\theta)^2 \sin(\phi) \sin(\theta)^2 + (\sin(\phi)^3 - \sin(\phi))r^4 \\
 &\quad (\phi) \cos(\theta) \sin(\theta) + (3r^7 \cos(\phi) \cos(\theta)^3 \sin(\phi) - 2r^4 \cos(\phi) \\
 &\quad (r \cos(\phi) \sin(\theta), r \sin(\phi) \sin(\theta), r \cos \\
 &\quad + (3r^6 \cos(\phi) \cos(\theta)^4 \sin(\phi) \sin(\theta)^2 + r^2 \cos(\theta) \sin \\
 &\quad ((\sin(\phi)^4 - \sin(\phi)^2)r^5 \cos(\theta) - r^4 \sin(\phi)^2) \sin(\theta) \\
 &\quad (r^5 \cos(\phi) \cos(\theta)^2 \sin(\phi)^2 - r^3 \sin(\phi)) \sin(\theta)^3 + \\
 &\quad (\phi) \sin(\theta), r \sin(\phi) \sin(\theta), r \cos(\theta) \\
 &\quad - ((3r^5 \cos(\theta)^2 \sin(\phi) - 2(\sin(\phi)^3 - \sin(\phi))r^3) \sin(\theta) \\
 &\quad + (3r^4 \cos(\theta)^2 - 2r^3 \cos(\phi) \cos(\theta) \sin(\phi) - r^2 \cos(\phi) \\
 &\quad (\theta) - (3r^4 \cos(\phi) \cos(\theta)^3 - r^2 \cos(\theta) \sin(\phi) + r \cos(\phi) \\
 &\quad (r \cos(\phi) \sin(\theta), r \sin(\phi) \sin(\theta), r \cos
 \end{aligned}$$

```
In [187]: a(u,v) == - a(v,u)
```

```
Out[187]: True
```

```
In [188]: a.symmetries()
```

```
no symmetry; antisymmetry: (0, 1)
```

The **exterior derivative** of a differential form:

```
In [189]: dom = om.exterior_derivative() ; print(dom) ; dom.display()
```

```
2-form domega on the Open subset U of the 3-dimensional differentiable manifold M
```

```
Out[189]: dω = -2ydx ∧ dy + dx ∧ dz - dy ∧ dz
```

Instead of invoking the method `exterior_derivative()`, one can use the function `xder`, after having imported it from `sage.manifolds.utilities`:

```
In [190]: from sage.manifolds.utilities import xder
          dom = xder(om)
```

```
In [191]: da = xder(a) ; print(da) ; da.display()
```

```
3-form dA on the Open subset U of the 3-dimensional differentiable manifold M
```

```
Out[191]: dA = (-6yz2 - 2y - 1) dx ∧ dy ∧ dz
```

The exterior derivative is nilpotent:

```
In [192]: ddf = xder(df) ; ddf.display()
```

```
Out[192]: ddf = 0
```

```
In [193]: ddom = xder(dom) ; ddom.display()
```

```
Out[193]: ddom = 0
```

Lie derivative

The Lie derivative of any tensor field with respect to a vector field is computed by the method `lie_derivative()`, with the vector field as the argument:

```
In [194]: lv_om = om.lie_derivative(v) ; print(lv_om) ; lv_om.display()
```

```
1-form on the Open subset U of the 3-dimensional differentiable manifold M
```

```
Out[194]: (-yz2 + (xy - 1)z + 2x) dx + (-xz2 + x2 + y2 + (x2 + xy)z) dy
          + (-2xyz + (x2 + 1)y + 1) dz
```

```
In [195]: lu_dh = dh.lie_derivative(u) ; print(lu_dh) ; lu_dh.display()
```

1-form on the Open subset U of the 3-dimensional differentiable manifold M

```
Out[195]:
```

$$\begin{aligned} & \left(u_x(x, y, z) \frac{\partial^2 H}{\partial x^2} + u_y(x, y, z) \frac{\partial^2 H}{\partial x \partial y} + u_z(x, y, z) \frac{\partial^2 H}{\partial x \partial z} + \frac{\partial H}{\partial x} \frac{\partial u_x}{\partial x} + \frac{\partial H}{\partial y} \frac{\partial u_y}{\partial x} \right. \\ & \quad \left. + \frac{\partial H}{\partial z} \frac{\partial u_z}{\partial x} \right) dx \\ & + \left(u_x(x, y, z) \frac{\partial^2 H}{\partial x \partial y} + u_y(x, y, z) \frac{\partial^2 H}{\partial y^2} + u_z(x, y, z) \frac{\partial^2 H}{\partial y \partial z} + \frac{\partial H}{\partial x} \frac{\partial u_x}{\partial y} \right. \\ & \quad \left. + \frac{\partial H}{\partial y} \frac{\partial u_y}{\partial y} + \frac{\partial H}{\partial z} \frac{\partial u_z}{\partial y} \right) dy \\ & + \left(u_x(x, y, z) \frac{\partial^2 H}{\partial x \partial z} + u_y(x, y, z) \frac{\partial^2 H}{\partial y \partial z} + u_z(x, y, z) \frac{\partial^2 H}{\partial z^2} + \frac{\partial H}{\partial x} \frac{\partial u_x}{\partial z} \right. \\ & \quad \left. + \frac{\partial H}{\partial y} \frac{\partial u_y}{\partial z} + \frac{\partial H}{\partial z} \frac{\partial u_z}{\partial z} \right) dz \end{aligned}$$

Let us check **Cartan identity** on the 1-form ω :

$$\mathcal{L}_v \omega = v \cdot d\omega + d\langle \omega, v \rangle$$

and on the 2-form A :

$$\mathcal{L}_v A = v \cdot dA + d(v \cdot A)$$

```
In [196]: om.lie_derivative(v) == v.contract(xder(om)) + xder(om(v))
```

```
Out[196]: True
```

```
In [197]: a.lie_derivative(v) == v.contract(xder(a)) + xder(v.contract(a))
```

```
Out[197]: True
```

The Lie derivative of a vector field along another one is the **commutator** of the two vectors fields:

```
In [198]: v.lie_derivative(u)(f) == u(v(f)) - v(u(f))
```

```
Out[198]: True
```

Tensor fields of arbitrary rank

Up to now, we have encountered tensor fields

- of type (0,0) (i.e. scalar fields),
- of type (1,0) (i.e. vector fields),
- of type (0,1) (i.e. 1-forms),
- of type (0,2) and antisymmetric (i.e. 2-forms).

More generally, tensor fields of any type (p, q) can be introduced in SageManifolds. For instance a tensor field of type (1,2) on the open subset U is declared as follows:

```
In [199]: t = U.tensor_field(1, 2, name='T') ; print(t)
```

Tensor field T of type (1,2) on the Open subset U of the 3-dimensional differentiable manifold M

As for vectors or 1-forms, the tensor's components with respect to the domain's default frame are set by means of square brackets:

```
In [200]: t[1,2,1] = 1 + x^2
          t[3,2,1] = x*y*z
```

Unset components are zero:

```
In [201]: t.display()
```

```
Out[201]: T = (x^2 + 1) \frac{\partial}{\partial x} \otimes dy \otimes dx + xyz \frac{\partial}{\partial z} \otimes dy \otimes dx
```

```
In [202]: t[:]
```

```
Out[202]: [[0, 0, 0], [x^2 + 1, 0, 0], [0, 0, 0]], [[0, 0, 0], [0, 0, 0], [0, 0, 0]],
           [[0, 0, 0], [xyz, 0, 0], [0, 0, 0]]
```

Display of the nonzero components:

```
In [203]: t.display_comp()
```

```
Out[203]: T^x_{yx} = x^2 + 1
          T^z_{yx} = xyz
```

Double square brackets return the component (still w.r.t. the default frame) as a scalar field, while single square brackets return the expression of this scalar field in terms of the domain's default coordinates:

```
In [204]: print(t[[1,2,1]]) ; t[[1,2,1]].display()
```

Scalar field on the Open subset U of the 3-dimensional differentiable manifold M

```
Out[204]: U          -> R
          (x, y, z)  -> x^2 + 1
          (r, \theta, \phi) -> r^2 \cos(\phi)^2 \sin(\theta)^2 + 1
```

```
In [205]: print(t[1,2,1]) ; t[1,2,1]
```

$x^2 + 1$

```
Out[205]: x^2 + 1
```

A tensor field of type (1,2) maps a 3-tuple (1-form, vector field, vector field) to a scalar field:

```
In [206]: print(t(om, u, v)) ; t(om, u, v).display()
```

Scalar field $T(\omega, u, v)$ on the Open subset U of the 3-dimensional differentiable manifold M

```
Out[206]: T(omega, u, v): U      -> R
          (x, y, z)  ->  (x^2 + 1)y^3 u_y(x, y, z) + (x^2 + 1)y^2 u_y(x, y, z) - (xy^2
                        + (x^4 + x^2)yu_y(x, y, z) + (x^2 y^2 u_y(x, y, z)
                        + (x^4 + x^2)u_y(x, y, z)
          (r, theta, phi) ->  (r^5 cos(phi)^2 sin(phi) sin(theta)^5 - ((cos(phi)^4 - cos(phi)^2)r^5
                        (theta)^4 + ((cos(phi)^3 - cos(phi))r^5 cos(theta)^2 + r^4 cos(phi)^2 cos
                        (theta)^3 - (r^4 cos(phi) cos(theta)^2 sin(phi) - r^2) sin(theta)^2)
                        (r cos(phi) sin(theta), r sin(phi) sin(theta), r
```

As for vectors and differential forms, the tensor components can be taken in any frame defined on the manifold:

```
In [207]: t[Y.frame(), 1,1,1, Y]
```

```
Out[207]: r^2 cos(phi)^4 sin(phi) sin(theta)^5 + (cos(phi)^4 - cos(phi)^2)r^3 sin(theta)^6
          - (cos(phi)^4 - cos(phi)^2)r^3 sin(theta)^4 + cos(phi)^2 sin(phi) sin(theta)^3
```

Tensor calculus

The **tensor product** \otimes is denoted by `**`:

```
In [208]: v.tensor_type() ; a.tensor_type()
```

```
Out[208]: (0, 2)
```

```
In [209]: b = v*a ; print(b) ; b
```

Tensor field $v \cdot A$ of type (1,2) on the Open subset U of the 3-dimensional differentiable manifold M

```
Out[209]: v \otimes A
```

The tensor product preserves the (anti)symmetries: since A is a 2-form, it is antisymmetric with respect to its two arguments (positions 0 and 1); as a result, b is antisymmetric with respect to its last two arguments (positions 1 and 2):

```
In [210]: a.symmetries()
```

```
no symmetry; antisymmetry: (0, 1)
```

```
In [211]: b.symmetries()
```

```
no symmetry; antisymmetry: (1, 2)
```

Standard **tensor arithmetics** is implemented:

```
In [212]: s = - t + 2*f* b ; print(s)
```

Tensor field of type (1,2) on the Open subset U of the 3-dimensional differentiable manifold M

Tensor contractions are dealt with by the methods `trace()` and `contract()`: for instance, let us contract the tensor T w.r.t. its first two arguments (positions 0 and 1), i.e. let us form the tensor c of components $c_i = T^k_{ki}$:

```
In [213]: c = t.trace(0,1)
print(c)
```

1-form on the Open subset U of the 3-dimensional differentiable manifold M

An alternative to the writing `trace(0,1)` is to use the **index notation** to denote the contraction: the indices are given in a string inside the `[]` operator, with '^' in front of the contravariant indices and '_' in front of the covariant ones:

```
In [214]: c1 = t['^k_ki']
print(c1)
c1 == c
```

1-form on the Open subset U of the 3-dimensional differentiable manifold M

Out[214]: True

The contraction is performed on the repeated index (here k); the letter denoting the remaining index (here i) is arbitrary:

```
In [215]: t['^k_kj'] == c
```

Out[215]: True

```
In [216]: t['^b_ba'] == c
```

Out[216]: True

It can even be replaced by a dot:

```
In [217]: t['^k_k.'] == c
```

Out[217]: True

LaTeX notations are allowed:

```
In [218]: t['^{k}_{ki}'] == c
```

Out[218]: True

The contraction $T^i_{jk} v^k$ of the tensor fields T and v is taken as follows (2 refers to the last index position of T and 0 to the only index position of v):

```
In [219]: tv = t.contract(2, v, 0)
print(tv)
```

Tensor field of type (1,1) on the Open subset U of the 3-dimensional differentiable manifold M

Since 2 corresponds to the last index position of T and 0 to the first index position of v , a shortcut for the above is

```
In [220]: tv1 = t.contract(v)
          print(tv1)
```

Tensor field of type (1,1) on the Open subset U of the 3-dimensional differentiable manifold M

```
In [221]: tv1 == tv
```

Out[221]: True

Instead of `contract()`, the **index notation**, combined with the `*` operator, can be used to denote the contraction:

```
In [222]: t['^i_jk']*v['^k'] == tv
```

Out[222]: True

The non-repeated indices can be replaced by dots:

```
In [223]: t['^._k']*v['^k'] == tv
```

Out[223]: True

Metric structures

A **Riemannian metric** on the manifold \mathcal{M} is declared as follows:

```
In [224]: g = M.riemannian_metric('g')
          print(g)
```

Riemannian metric g on the 3-dimensional differentiable manifold M

It is a symmetric tensor field of type (0,2):

```
In [225]: g.parent()
```

Out[225]: $\mathcal{T}^{(0,2)}()$

```
In [226]: print(g.parent())
```

Free module $T^{(0,2)}(M)$ of type-(0,2) tensors fields on the 3-dimensional differentiable manifold M

```
In [227]: g.symmetries()
```

symmetry: (0, 1); no antisymmetry

The metric is initialized by its components with respect to some vector frame. For instance, using the default frame of \mathcal{M} :

```
In [228]: g[1,1], g[2,2], g[3,3] = 1, 1, 1
          g.display()
```

Out[228]: $g = dx \otimes dx + dy \otimes dy + dz \otimes dz$

The components w.r.t. another vector frame are obtained as for any tensor field:

```
In [229]: g.display(Y.frame(), Y)
```

```
Out[229]: g = dr ⊗ dr + r2dθ ⊗ dθ + r2 sin(θ)2dφ ⊗ dφ
```

Of course, the metric acts on vector pairs:

```
In [230]: u.display(); v.display(); print(g(u,v)); g(u,v).display()
```

Scalar field $g(u,v)$ on the open subset U of the 3-dimensional differentiable manifold M

```
Out[230]: g(u,v) : U      → ℝ
           (x,y,z)   ↦   xyzuz(x,y,z) + yux(x,y,z) - xuy(x,y,z) + ux(x,y,z)
           (r,θ,φ)  ↦   r3 cos(φ) cos(θ) sin(φ) sin(θ)2uz(r cos(φ) sin(θ), r sin(φ) sin(θ),
           (φ) sin(θ)uy(r cos(φ) sin(θ), r sin(φ) sin(θ),
           + (r sin(φ) sin(θ) + 1)ux(r cos(φ) sin(θ), r sin(φ) sin(θ))
```

The **Levi-Civita connection** associated to the metric g :

```
In [231]: nabla = g.connection()
          print(nabla); nabla
```

Levi-Civita connection nabla_g associated with the Riemannian metric g on the 3-dimensional differentiable manifold M

```
Out[231]: ∇g
```

The Christoffel symbols with respect to the manifold's default coordinates:

```
In [232]: nabla.coef()[ : ]
```

```
Out[232]: [[[0, 0, 0], [0, 0, 0], [0, 0, 0]], [[0, 0, 0], [0, 0, 0], [0, 0, 0]],
           [[0, 0, 0], [0, 0, 0], [0, 0, 0]]]
```

The Christoffel symbols with respect to the coordinates (r, θ, ϕ) :

```
In [233]: nabla.coef(Y.frame())[ : , Y]
```

```
Out[233]: [[ [0, 0, 0], [0, -r, 0], [0, 0, -r sin(θ)2 ] ],
           [ [0, 1/r, 0], [1/r, 0, 0], [0, 0, -cos(θ) sin(θ)] ],
           [ [0, 0, 1/r], [0, 0, cos(θ)/sin(θ)], [1/r, cos(θ)/sin(θ), 0] ] ]
```

A nice view is obtained via the method `display()` (by default, only the nonzero connection coefficients are shown):

```
In [234]: nbla.display(frame=Y.frame(), chart=Y)
```

```
Out[234]:
```

$$\begin{aligned}\Gamma^r_{\theta\theta} &= -r \\ \Gamma^r_{\phi\phi} &= -r \sin(\theta)^2 \\ \Gamma^\theta_{r\theta} &= \frac{1}{r} \\ \Gamma^\theta_{\theta r} &= \frac{1}{r} \\ \Gamma^\theta_{\phi\phi} &= -\cos(\theta) \sin(\theta) \\ \Gamma^\phi_{r\phi} &= \frac{1}{r} \\ \Gamma^\phi_{\theta\phi} &= \frac{\cos(\theta)}{\sin(\theta)} \\ \Gamma^\phi_{\phi r} &= \frac{1}{r} \\ \Gamma^\phi_{\phi\theta} &= \frac{\cos(\theta)}{\sin(\theta)}\end{aligned}$$

The connection acting as a covariant derivative:

```
In [235]: nab_v = nbla(v)
print(nab_v) ; nab_v.display()
```

Tensor field nabla_g(v) of type (1,1) on the Open subset U of the 3-dimensional differentiable manifold M

```
Out[235]:
```

$$\nabla_g v = \frac{\partial}{\partial x} \otimes dy - \frac{\partial}{\partial y} \otimes dx + yz \frac{\partial}{\partial z} \otimes dx + xz \frac{\partial}{\partial z} \otimes dy + xy \frac{\partial}{\partial z} \otimes dz$$

Being a Levi-Civita connection, ∇_g is torsion.free:

```
In [236]: print(nbla.torsion()) ; nbla.torsion().display()
```

Tensor field of type (1,2) on the 3-dimensional differentiable manifold M

```
Out[236]: 0
```

In the present case, it is also flat:

```
In [237]: print(nbla.riemann()) ; nbla.riemann().display()
```

Tensor field Riem(g) of type (1,3) on the 3-dimensional differentiable manifold M

```
Out[237]: Riem(g) = 0
```

Let us consider a non-flat metric, by changing g_{rr} to $1/(1+r^2)$:

```
In [238]: g[Y.frame(), 1,1, Y] = 1/(1+r^2)
g.display(Y.frame(), Y)
```

```
Out[238]:
```

$$g = \left(\frac{1}{r^2 + 1} \right) dr \otimes dr + r^2 d\theta \otimes d\theta + r^2 \sin(\theta)^2 d\phi \otimes d\phi$$

For convenience, we change the default chart on the domain U to $Y=(U, (r, \theta, \phi))$:

```
In [239]: U.set_default_chart(Y)
```

In this way, we do not have to specify Y when asking for coordinate expressions in terms of (r, θ, ϕ) :

```
In [240]: g.display(Y.frame())
```

```
Out[240]: g = \left(\frac{1}{r^2 + 1}\right) dr \otimes dr + r^2 d\theta \otimes d\theta + r^2 \sin(\theta)^2 d\phi \otimes d\phi
```

We recognize the metric of the hyperbolic space \mathbb{H}^3 . Its expression in terms of the chart $(U, (x, y, z))$ is

```
In [241]: g.display(X_U.frame(), X_U)
```

```
Out[241]: g = \left(\frac{y^2 + z^2 + 1}{x^2 + y^2 + z^2 + 1}\right) dx \otimes dx + \left(-\frac{xy}{x^2 + y^2 + z^2 + 1}\right) dx \otimes dy
+ \left(-\frac{xz}{x^2 + y^2 + z^2 + 1}\right) dx \otimes dz + \left(-\frac{xy}{x^2 + y^2 + z^2 + 1}\right) dy \otimes dx
+ \left(\frac{x^2 + z^2 + 1}{x^2 + y^2 + z^2 + 1}\right) dy \otimes dy + \left(-\frac{yz}{x^2 + y^2 + z^2 + 1}\right) dy \otimes dz
+ \left(-\frac{xz}{x^2 + y^2 + z^2 + 1}\right) dz \otimes dx + \left(-\frac{yz}{x^2 + y^2 + z^2 + 1}\right) dz \otimes dy
+ \left(\frac{x^2 + y^2 + 1}{x^2 + y^2 + z^2 + 1}\right) dz \otimes dz
```

A matrix view of the components may be more appropriate:

```
In [242]: g[X_U.frame(), :, X_U]
```

```
Out[242]: \begin{pmatrix} \frac{y^2+z^2+1}{x^2+y^2+z^2+1} & -\frac{xy}{x^2+y^2+z^2+1} & -\frac{xz}{x^2+y^2+z^2+1} \\ -\frac{xy}{x^2+y^2+z^2+1} & \frac{x^2+z^2+1}{x^2+y^2+z^2+1} & -\frac{yz}{x^2+y^2+z^2+1} \\ -\frac{xz}{x^2+y^2+z^2+1} & -\frac{yz}{x^2+y^2+z^2+1} & \frac{x^2+y^2+1}{x^2+y^2+z^2+1} \end{pmatrix}
```

We extend these components, a priori defined only on U , to the whole manifold \mathcal{M} , by demanding the same coordinate expressions in the frame associated to the chart $X=(\mathcal{M}, (x, y, z))$:

```
In [243]: g.add_comp_by_continuation(X.frame(), U, X)
g.display()
```

```
Out[243]: g = \left(\frac{y^2 + z^2 + 1}{x^2 + y^2 + z^2 + 1}\right) dx \otimes dx + \left(-\frac{xy}{x^2 + y^2 + z^2 + 1}\right) dx \otimes dy
+ \left(-\frac{xz}{x^2 + y^2 + z^2 + 1}\right) dx \otimes dz + \left(-\frac{xy}{x^2 + y^2 + z^2 + 1}\right) dy \otimes dx
+ \left(\frac{x^2 + z^2 + 1}{x^2 + y^2 + z^2 + 1}\right) dy \otimes dy + \left(-\frac{yz}{x^2 + y^2 + z^2 + 1}\right) dy \otimes dz
+ \left(-\frac{xz}{x^2 + y^2 + z^2 + 1}\right) dz \otimes dx + \left(-\frac{yz}{x^2 + y^2 + z^2 + 1}\right) dz \otimes dy
+ \left(\frac{x^2 + y^2 + 1}{x^2 + y^2 + z^2 + 1}\right) dz \otimes dz
```

The Levi-Civita connection is automatically recomputed, after the change in g :

```
In [244]: nabla = g.connection()
```

In particular, the Christoffel symbols are different:

```
In [245]: nabla.display(only_nonredundant=True)
```

```
Out[245]:
```

$$\Gamma^x_{xx} = -\frac{xy^2+xz^2+x}{x^2+y^2+z^2+1}$$

$$\Gamma^x_{xy} = \frac{x^2y}{x^2+y^2+z^2+1}$$

$$\Gamma^x_{xz} = \frac{x^2z}{x^2+y^2+z^2+1}$$

$$\Gamma^x_{yy} = -\frac{x^3+xz^2+x}{x^2+y^2+z^2+1}$$

$$\Gamma^x_{yz} = \frac{xyz}{x^2+y^2+z^2+1}$$

$$\Gamma^x_{zz} = -\frac{x^3+xy^2+x}{x^2+y^2+z^2+1}$$

$$\Gamma^y_{xx} = -\frac{y^3+yz^2+y}{x^2+y^2+z^2+1}$$

$$\Gamma^y_{xy} = \frac{xy^2}{x^2+y^2+z^2+1}$$

$$\Gamma^y_{xz} = \frac{xyz}{x^2+y^2+z^2+1}$$

$$\Gamma^y_{yy} = -\frac{yz^2+(x^2+1)y}{x^2+y^2+z^2+1}$$

$$\Gamma^y_{yz} = \frac{y^2z}{x^2+y^2+z^2+1}$$

$$\Gamma^y_{zz} = -\frac{y^3+(x^2+1)y}{x^2+y^2+z^2+1}$$

$$\Gamma^z_{xx} = -\frac{z^3+(y^2+1)z}{x^2+y^2+z^2+1}$$

$$\Gamma^z_{xy} = \frac{xyz}{x^2+y^2+z^2+1}$$

$$\Gamma^z_{xz} = \frac{xz^2}{x^2+y^2+z^2+1}$$

$$\Gamma^z_{yy} = -\frac{z^3+(x^2+1)z}{x^2+y^2+z^2+1}$$

$$\Gamma^z_{yz} = \frac{yz^2}{x^2+y^2+z^2+1}$$

$$\Gamma^z_{zz} = -\frac{(x^2+y^2+1)z}{x^2+y^2+z^2+1}$$

```
In [246]: nbla.display(frame=Y.frame(), chart=Y, only_nonredundant=True)
```

$$\begin{aligned} \text{Out}[246]: \quad \Gamma^r_{rr} &= -\frac{r}{r^2+1} \\ \Gamma^r_{\theta\theta} &= -r^3 - r \\ \Gamma^r_{\phi\phi} &= -(r^3 + r) \sin(\theta)^2 \\ \Gamma^\theta_{r\theta} &= \frac{1}{r} \\ \Gamma^\theta_{\phi\phi} &= -\cos(\theta) \sin(\theta) \\ \Gamma^\phi_{r\phi} &= \frac{1}{r} \\ \Gamma^\phi_{\theta\phi} &= \frac{\cos(\theta)}{\sin(\theta)} \end{aligned}$$

The **Riemann tensor** is now

```
In [247]: Riem = nbla.riemann()
print(Riem) ; Riem.display(Y.frame())
```

Tensor field Riem(g) of type (1,3) on the 3-dimensional differentiable manifold M

$$\begin{aligned} \text{Out}[247]: \quad \text{Riem}(g) &= -r^2 \frac{\partial}{\partial r} \otimes d\theta \otimes dr \otimes d\theta + r^2 \frac{\partial}{\partial r} \otimes d\theta \otimes d\theta \otimes dr - r^2 \sin(\theta)^2 \frac{\partial}{\partial r} \\ &\otimes d\phi \otimes dr \otimes d\phi + r^2 \sin(\theta)^2 \frac{\partial}{\partial r} \otimes d\phi \otimes d\phi \otimes dr + \left(\frac{1}{r^2+1}\right) \frac{\partial}{\partial \theta} \otimes dr \otimes dr \\ &\otimes d\theta + \left(-\frac{1}{r^2+1}\right) \frac{\partial}{\partial \theta} \otimes dr \otimes d\theta \otimes dr - r^2 \sin(\theta)^2 \frac{\partial}{\partial \theta} \otimes d\phi \otimes d\theta \otimes d\phi \\ &+ r^2 \sin(\theta)^2 \frac{\partial}{\partial \theta} \otimes d\phi \otimes d\phi \otimes d\theta + \left(\frac{1}{r^2+1}\right) \frac{\partial}{\partial \phi} \otimes dr \otimes dr \otimes d\phi \\ &+ \left(-\frac{1}{r^2+1}\right) \frac{\partial}{\partial \phi} \otimes dr \otimes d\phi \otimes dr + r^2 \frac{\partial}{\partial \phi} \otimes d\theta \otimes d\theta \otimes d\phi - r^2 \frac{\partial}{\partial \phi} \otimes d\theta \\ &\otimes d\phi \otimes d\theta \end{aligned}$$

Note that it can be accessed directly via the metric, without any explicit mention of the connection:

```
In [248]: g.riemann() is nbla.riemann()
```

Out[248]: True

The **Ricci tensor** is

```
In [249]: Ric = g.ricci()
print(Ric) ; Ric.display(Y.frame())
```

Field of symmetric bilinear forms Ric(g) on the 3-dimensional differentiable manifold M

$$\text{Out}[249]: \quad \text{Ric}(g) = \left(-\frac{2}{r^2+1}\right) dr \otimes dr - 2r^2 d\theta \otimes d\theta - 2r^2 \sin(\theta)^2 d\phi \otimes d\phi$$

The **Weyl tensor** is:

```
In [250]: C = g.weyl()
          print(C) ; C.display()
```

Tensor field $C(g)$ of type (1,3) on the 3-dimensional differentiable manifold M

```
Out[250]: C(g) = 0
```

The Weyl tensor vanishes identically because the dimension of \mathcal{M} is 3.

Finally, the **Ricci scalar** is

```
In [251]: R = g.ricci_scalar()
          print(R) ; R.display()
```

Scalar field $r(g)$ on the 3-dimensional differentiable manifold M

```
Out[251]: r(g) :  M      ->  R
              (x,y,z)  -> -6
              on U :  (r,theta,phi) -> -6
```

We recover the fact that \mathbb{H}^3 is a Riemannian manifold of constant negative curvature.

Tensor transformations induced by a metric

The most important tensor transformation induced by the metric g is the so-called **musical isomorphism**, or **index raising** and **index lowering**:

```
In [252]: print(t)
```

Tensor field T of type (1,2) on the Open subset U of the 3-dimensional differentiable manifold M

```
In [253]: t.display()
```

```
Out[253]: T = (r^2 cos(phi)^2 sin(theta)^2 + 1) * d/dx \otimes dy \otimes dx + r^3 cos(phi) cos(theta) sin(phi) sin(theta)^2 * d/dz \otimes dy \otimes dx
```

```
In [254]: t.display(X_U.frame(), X_U)
```

```
Out[254]: T = (x^2 + 1) * d/dx \otimes dy \otimes dx + xyz * d/dz \otimes dy \otimes dx
```

Raising the last index of T with g :

```
In [255]: s = t.up(g, 2)
          print(s)
```

Tensor field of type (2,1) on the Open subset U of the 3-dimensional differentiable manifold M

Raising all the covariant indices of T (i.e. those at the positions 1 and 2):

```
In [256]: s = t.up(g)
          print(s)
```

Tensor field of type (3,0) on the Open subset U of the 3-dimensional differentiable manifold M

```
In [257]: s = t.down(g)
          print(s)
```

Tensor field of type (0,3) on the Open subset U of the 3-dimensional differentiable manifold M

Hodge duality

The volume 3-form (Levi-Civita tensor) associated with the metric g is

```
In [258]: epsilon = g.volume_form()
          print(epsilon) ; epsilon.display()
```

3-form eps_g on the 3-dimensional differentiable manifold M

```
Out[258]:
```

$$\epsilon_g = \left(\frac{1}{\sqrt{x^2 + y^2 + z^2 + 1}} \right) dx \wedge dy \wedge dz$$

```
In [259]: epsilon.display(Y.frame())
```

```
Out[259]:
```

$$\epsilon_g = \left(\frac{r^2 \sin(\theta)}{\sqrt{r^2 + 1}} \right) dr \wedge d\theta \wedge d\phi$$

```
In [260]: print(f) ; f.display()
```

Scalar field f on the Open subset U of the 3-dimensional differentiable manifold M

```
Out[260]:
```

$$\begin{aligned} f: U &\longrightarrow \mathbb{R} \\ (x, y, z) &\longmapsto z^3 + y^2 + x \\ (r, \theta, \phi) &\longmapsto r^3 \cos(\theta)^3 + r^2 \sin(\phi)^2 \sin(\theta)^2 + r \cos(\phi) \sin(\theta) \end{aligned}$$

```
In [261]: sf = f.hodge_dual(g)
          print(sf) ; sf.display()
```

3-form *f on the Open subset U of the 3-dimensional differentiable manifold M

```
Out[261]:
```

$$\star f = \left(\frac{r^3 \cos(\theta)^3 + r^2 \sin(\phi)^2 \sin(\theta)^2 + r \cos(\phi) \sin(\theta)}{\sqrt{r^2 + 1}} \right) dx \wedge dy \wedge dz$$

We check the classical formula $\star f = f \epsilon_g$, or, more precisely, $\star f = f \epsilon_g|_U$ (for f is defined on U only):

```
In [262]: sf == f * epsilon.restrict(U)
```

```
Out[262]: True
```

The Hodge dual of a 1-form is a 2-form:

In [263]: `print(om) ; om.display()`

1-form omega on the Open subset U of the 3-dimensional differentiable manifold M

Out[263]: $\omega = r^2 \sin(\theta)^2 dx + r \cos(\theta) dy + (r \cos(\phi) \sin(\theta) - r \cos(\theta)) dz$

In [264]: `som = om.hodge_dual(g)`
`print(som) ; som.display()`

2-form *omega on the Open subset U of the 3-dimensional differentiable manifold M

Out[264]:

$$\star\omega = \left(\frac{r^4 \cos(\phi) \cos(\theta) \sin(\theta)^3 - r^3 \cos(\theta)^3 - r \cos(\theta)}{\sqrt{r^2 + 1}} + \frac{r^3 (\cos(\phi) + \sin(\phi)) \cos(\theta)^2 + r \cos(\phi) \sin(\theta)}{\sqrt{r^2 + 1}} \right) dx \wedge dy$$

$$+ \left(-\frac{r^4 \cos(\phi) \sin(\phi) \sin(\theta)^4 - r^3 \cos(\theta)^2 \sin(\phi) \sin(\theta)}{\sqrt{r^2 + 1}} + \frac{(\cos(\phi) \sin(\phi) + \sin(\phi)^2) r^3 \cos(\theta) \sin(\theta)^2 + r \cos(\theta)}{\sqrt{r^2 + 1}} \right) dx \wedge dz$$

$$+ \left(\frac{r^4 \cos(\phi)^2 \sin(\theta)^4 - r^3 \cos(\phi) \cos(\theta)^2 \sin(\theta)}{\sqrt{r^2 + 1}} + \frac{((\cos(\phi)^2 + \cos(\phi) \sin(\phi)) r^3 \cos(\theta) + r^2) \sin(\theta)^2}{\sqrt{r^2 + 1}} \right) dy \wedge dz$$

The Hodge dual of a 2-form is a 1-form:

In [265]: `print(a)`

2-form A on the Open subset U of the 3-dimensional differentiable manifold M


```
In [266]: sa = a.hodge_dual(g)
print(sa) ; sa.display()
```

1-form *A on the Open subset U of the 3-dimensional differentiable manifold M

Out[266]:

$$\begin{aligned}
 & \star A \\
 & \left(\begin{aligned}
 & 3 r^5 \cos (\theta)^5 + 3 r^3 \cos (\theta)^3 \\
 & + \left(3 r^6 \cos (\phi) \cos (\theta)^2 \sin (\phi) - 2 r^5 \cos (\phi) \cos (\theta) \sin (\phi) - 2 r^4 \cos (\phi) \sin (\phi)^3 \right) \sin (\theta)^4 \\
 & + \left(2 r^4 \cos (\theta) \sin (\phi)^3 + \left(\sin (\phi)^3 - \sin (\phi) \right) r^3 \right) \sin (\theta)^3 \\
 & + \left(3 r^5 \cos (\theta)^3 \sin (\phi)^2 - 2 r^4 \cos (\phi) \cos (\theta)^2 \sin (\phi) + r^3 \cos (\phi) \cos (\theta) \sin (\phi) - r^2 \cos (\phi) \sin (\phi) \right) \\
 & \left(\theta \right)^2 + \left(2 r^4 \cos (\theta)^3 \sin (\phi) + r^3 \cos (\phi) \cos (\theta)^2 + 2 r^2 \cos (\theta) \sin (\phi) \right) \sin (\theta) \\
 & \end{aligned} \right) \frac{1}{\sqrt{r^2 + 1}} \\
 & + \left(\begin{aligned}
 & r^3 \cos (\theta)^3 \\
 & - \left(3 \left(\sin (\phi)^2 - 1 \right) r^6 \cos (\theta)^2 - 2 r^5 \cos (\theta) \sin (\phi)^2 - 2 \left(\sin (\phi)^4 - \sin (\phi)^2 \right) r^4 \right) \sin (\theta)^4 \\
 & + \left(2 r^4 \cos (\phi) \cos (\theta) \sin (\phi)^2 + \left(\cos (\phi) \sin (\phi)^2 - \cos (\phi) \right) r^3 \right) \sin (\theta)^3 \\
 & + \left(3 r^6 \cos (\theta)^4 + 3 r^5 \cos (\phi) \cos (\theta)^3 \sin (\phi) + 3 r^4 \cos (\theta)^2 - \left(\sin (\phi)^2 - 1 \right) r^3 \cos (\theta) \right) \\
 & \left(\theta \right)^2 + r \cos (\theta) - \left(r^3 \left(\cos (\phi) + \sin (\phi) \right) \cos (\theta)^2 + r \cos (\phi) \right) \sin (\theta) \\
 & \end{aligned} \right) \frac{1}{\sqrt{r^2 + 1}} \\
 & + \left(\begin{aligned}
 & 2 r^5 \sin (\phi) \sin (\theta)^5 \\
 & + \left(3 r^6 \cos (\theta)^3 \sin (\phi) + 2 r^4 \cos (\phi)^2 \cos (\theta) \sin (\phi) + 2 r^3 \sin (\phi) \right) \sin (\theta)^3 \\
 & - \left(2 r^4 \cos (\phi) \cos (\theta)^2 \sin (\phi) + \left(\cos (\phi) \sin (\phi) + 1 \right) r^3 \cos (\theta) \right) \sin (\theta)^2 - r \cos (\theta) \\
 & - \left(3 r^5 \cos (\phi) \cos (\theta)^4 - r^3 \cos (\theta)^2 \sin (\phi) \right) \sin (\theta) \\
 & \end{aligned} \right) \frac{1}{\sqrt{r^2 + 1}}
 \end{aligned}$$

Finally, the Hodge dual of a 3-form is a 0-form:

```
In [267]: print(da) ; da.display()
```

3-form dA on the Open subset U of the 3-dimensional differentiable manifold M

```
Out[267]: dA = (-2 (3 r^3 cos(theta)^2 sin(phi) + r sin(phi)) sin(theta) - 1) dx ^ dy ^ dz
```

```
In [268]: sda = da.hodge_dual(g)
print(sda) ; sda.display()
```

Scalar field *dA on the Open subset U of the 3-dimensional differentiable manifold M

```
Out[268]: *dA : U      -> R
          (x, y, z)  -> -(6 y z^2 + 2 y + 1) sqrt(x^2 + y^2 + z^2 + 1)
          (r, theta, phi) -> -sqrt(r^2 cos(theta)^2 + r^2 sin(theta)^2 + 1) (2 (3 r^3 cos(theta)^2 sin(phi) + r sin(phi)) sin(theta) - 1)
```

In dimension 3 and for a Riemannian metric, the Hodge star is idempotent:

```
In [269]: sf.hodge_dual(g) == f
```

```
Out[269]: True
```

```
In [270]: som.hodge_dual(g) == om
```

```
Out[270]: True
```

```
In [271]: sa.hodge_dual(g) == a
```

```
Out[271]: True
```

```
In [272]: sda.hodge_dual(g) == da
```

```
Out[272]: True
```

Getting help

To get the list of functions (methods) that can be called on a object, type the name of the object, followed by a dot and the TAB key, e.g.

```
sa.
```

To get information on an object or a method, use the question mark:

```
In [273]: nabla?
```

```
In [274]: g.ricci_scalar?
```

Using a double question mark leads directly to the **Python source code** (SageMath is **open source**, isn't it?)

```
In [275]: g.ricci_scalar??
```

Going further

Have a look at the [examples on SageManifolds page](#), especially the [2-dimensional sphere example](#) for usage on a non-parallelizable manifold (each scalar field has to be defined in at least two coordinate charts, the module $\mathcal{X}(\mathcal{M})$ is no longer free and each tensor field has to be defined in at least two vector frames).