SurfaceComplex

`Domains: CMAP, SCMPLX`

Kurt Pagani
`nilqed@gmail.com`

December 20, 2016

Abstract

This manual decribes the FriCAS domains **CellMap** and **SurfaceComplex**. These domains provide methods to compute various differential geometric properties of so called p-surfaces in $\mathbb{R}^n$, a notion which is used by Walter Rudin in his famous *Principles of Mathematical Analysis*.

Contents

1 Introduction

Dealing with manifolds or similar geometric structures in a scientific computing system like FriCAS requires a great deal of abstraction as well as some simplifications compared to the usual mathematical definitions of such objects. It makes not much sense to speak of open sets, charts, diffeomorphisms and other topological properties in this context. Of course, it is not impossible to mimic such terms in FriCAS, however, it would be of little use since most of them were not computable anyway. Since we already have differential forms and jet bundles at hand, we are looking for *dual* objects, so that we will be able to integrate differential forms over *surfaces* for example, that is to have some form of *Stokes theorem*:

$$\int_M d\,\omega = \int_{\partial M} \omega$$

The identity above holds for a wide range of objects M and ω, and is - roughly speaking - the basis of geometric integration [2] and/or geometric measure theory [3]. The generalization of Stokes theorem in those theories looks like

$$\langle M, d\,\omega\rangle = \langle \partial M, \omega\rangle,$$

where the pairing $\langle\cdot,\cdot\rangle$ usually denotes a bilinear mapping between so called *chains* and *co-chains*. In the approach [2] by Whitney (and recently by Harrison [4]) the chains M are built by geometric primitives (e.g. polyhedral chains) forming normed linear spaces (for various norms like *sharp* or *flat*) whose duals are the corresponding Banach space of co-chains, where it is not clear a priori that they represent differential forms (this can be shown under certain conditions and is a highly non-trivial task). The other approach (as described by Federer [3]) starts from smooth differential forms and looks at the dual space, well known as *DeRham Currents*. The whole theory is more or less a fruitful generalization of Laurent Schwartz's concept of *distributions*. In this context Stokes theorem holds by definition:

$$T(d\,\omega) = \partial T(\omega),$$

for all smooth forms and all currents (comparable to the derivatives of distributions $T'(\varphi) = -T(\varphi')$). This means ∂ is a linear operator (usually called

boundary operator) which is loosely speaking the transpose of the differential operator d.

Here we will follow along the lines of the latter approach, that is we are going to deal with objects which might be seen as DeRham currents in a wider sense, though we only consider *nice* objects (currents form a really huge space and many elements look quite bizarre) which have some relation to differential geometry.

There is, however, **one notable difference** to the literature cited inasmuch as a differential form here is considered as an element of the graded algebra

$$\bigoplus_{p=0}^{n} \mathbf{\Omega}^p(\mathbb{R}^n),$$

whereas usually only homogeneous forms $\in \mathbf{\Omega^p}$ [1] are meant. This descends from the already existing domain *DeRhamComplex* on the one hand and computational convenience of such inhomogeneous forms on the other. So we have for instance

$$\omega(x) = a(x)\, d\, x_1 + b(x)\, dx_1 \wedge dx_2,$$

which is neither a 1-form nor a 2-form but just a form. If we speak of a p-form we always mean a homogeneous form otherwise we just speak of a *form*. This also is a reason why we speak of a *complex*.

2 $p-$Surfaces

From a computational point of view it seems very convenient to follow the method in ([1],chapter 10) how differential forms are introduced. We will not copy here parts of that theory but merely state the most important definitions on which the *code* will be based.

Definition 1 *A subset of* $\mathbb{R}^p$ *of the form*

$$\{(x_1, \ldots, x_p) \in \mathbb{R}^p : a_i \leq x_i \leq b_i, i = 1, \ldots, p\}$$

is called a $\mathbf{p-cell}$.

Usually $p > 1$ and $a_i < b_i$, but we will allow $p = 0$ and $a_j = b_j$ as a degenerate case (we will come back to this later on).

[1] Ω is a synonym for e.g. $\mathcal{D}, \mathcal{E}$ etc.

Definition 2 *Suppose E is an open set in $\mathbb{R}^n$. A* **p−surface** *in E is a $\mathcal{C}^1-$mapping Φ from a compact set $D \subset \mathbb{R}^p$ into $\mathbb{R}^n$.*

D is called the parameter domain of Φ.

In the following we will tacitly assume that D is some $p-$cell. This entails no loss of generality as pointed out in [1] too. One important thing we will cite ([1], def. 10.10) here is:

> We stress that $p-$surfaces in E are defined to be mappings into E, not subsets of E. This agrees with our earlier definition of curves (Definition 6.26). In fact, $1-$surfaces are precisely the same as continuously differentiable curves.

In FriCAS there is no $\mathbb{R}^n$, not to speak of an open set E. Instead of we will use the (pseudo-) Field *Expression R*, where is a *Ring* with some additional properties (usually $R = \mathbb{Z}$). Therefore points in $\mathbb{R}^n$ would correspond to

```
x:List Expression Integer:=[x[1],...,x[n]]
-- or
y:Vector Expression Integer:=[y[1],...,y[n]]
-- x <-> easy convertibe
```

and $p-$cells will be represented by a list of segments

```
Q:List Segment Expression Integer:=[a[1]..b[1],...,a[p]..b[p]]
```

Instead of `Integer` one might use `Float` of course. This way we have a correspondence to the definitions given above as soon we fix how the mappings from Q to E looks like. Obviously we can emulate E as lists of length n of a suitable type, thus our mapping Φ (P below) can be given by the λ expression:

```
X+->[P[1](X.1,...,X.p),...,P[n](X.1,...,X.p)]
```

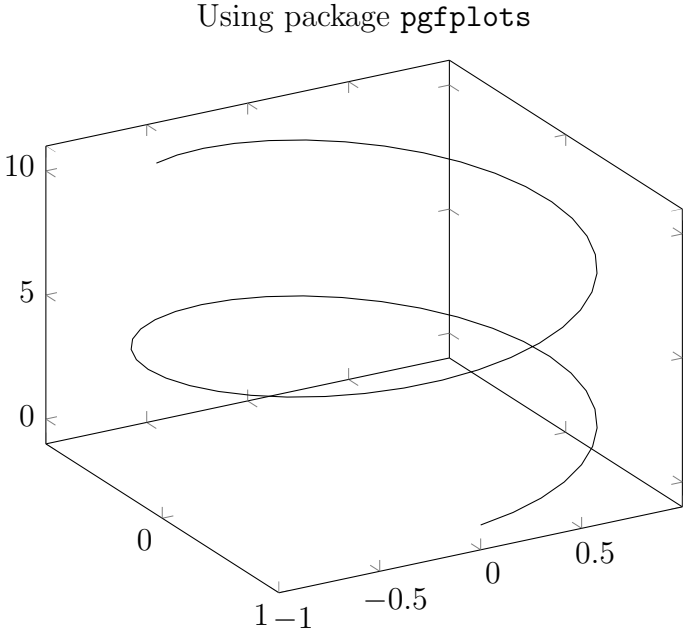
where $X = X.1, ..., X.p$ ranges over the $p-$cell Q.

We will call a $p-$surface in FriCAS a *CellMap*. This because it shall remind to the domain (a $p-$cell) on the one hand and to the fact that it is a mapping (and not a set) on the other hand. As an example we construct a $1-$surface (a curve) in three space as follows:

```
cellMap([0..10],x+->[cos x.1, sin x.1, x.1])$CellMap(Float,3)
```

```
                                                    1
|[0.0..10.0] -> [cos($ ),sin($ ),$ ]|
                      1       1   1
```

Type:CellMap(Float,3)



Notice that domain variable (x in the case above) is a list with the dimension of the cell (1−cell in the example), therefore the (only) parameter is `x.1`. Further note that x is just a dummy variable, we could have chosen any other symbol as well, so that a *CellMap* is independent of the coordinate symbol - which is also indicated by the $\$'s$ in the output format. One has to apply the mapping to a point in order to get a point on the *surface*.

2.1 Integration over $p-$Surfaces

Let ω be a general $p-$form in $\mathbb{R}^n$:

$$\omega(u) = \sum_I \omega_{i_1,\ldots,i_p}(u)\ du_{i_1} \wedge \ldots \wedge du_{i_p}$$

then the integral of ω over a $p-$surface Φ is defined as

$$\int_\Phi \omega = \int_Q \sum_I \omega_{i_1,\ldots,i_p}(\Phi(x))\ \frac{\partial(u_{i_1},\ldots,u_{i_p})}{\partial(x_1,\ldots,x_p)}\ dx_1 \cdots dx_p$$

where Q denotes the parameter domain of Φ (a $p-$cell in FriCAS). The functions $\omega_{i_1,\ldots,i_p}$ are assumed to be real and continuous and the *Jacobian* in the right hand side integral above is the one determined by the mapping

$$(x_1,\ldots,x_p) \mapsto (\Phi_{i_1}(x),\ldots,\Phi_{i_p}(x)),$$

which will be assumed to be continuously differentiable. Consequently the integral is an ordinary Riemann integral over a hypercube in $\mathbb{R}^p$. This widely corresponds to the statements given in [1]. The reader acquainted with the topic will immediately see that this is just integration of the pull back of ω by Φ over its ($\Phi's$) domain:

$$\int_{\Phi(Q)} \omega = \int_Q \Phi^\star \omega$$

This and more will be rigorously proved in [1], so we will end here with the theory and refer to the references given at the end.

3 Surface Complex

A cell map may be seen as a piece of a bigger object, comparable to a chart of a manifold which needs a whole atlas for its description. Since we have assumed that the domains of cell maps are compact we have a boundary ∂Q which is built of of cells of one lower dimension. In other words, a $p-$surface Φ has a boundary as well ($\partial\Phi$) which can be seen as a formal sum of other cell maps. Similar as the domain `DeRhamComplex` we define the domain `SurfaceComplex` as the free abelian group of cell maps over $\mathbb{Z}$. This will enable us to define a *boundary operator*.

Definition 3 *Let $\Sigma_p(\mathbb{R}^n)$ denote the set of $p-$surfaces, then the formal sums of these (with integer coefficients) build a surface complex.*

$$\mathcal{SC}_n = \bigoplus_{p=0}^{n} \mathbb{Z} \cdot \mathbf{\Sigma}_p(\mathbb{R}^n),$$

The boundary operator then is well defined

$$\partial : \mathcal{SC}_n \to \mathcal{SC}_n,$$

by the identity

$$\int_{\partial\Phi} \omega = \int_\Phi d\,\omega.$$

and the classical boundary of chains (simplexes in $\mathbb{R}^n$).

The last statement probably should be specified. Let $a, b \in \mathbb{R}^p$, then the line segment $[a, b]$ (the convex hull of the two points) induces a $1-$*current* in a natural way:

$$[[a, b]](\omega) = \int_0^1 \langle \omega((1-t)a + t + tb), b - a\rangle dt$$

for any one form $\omega \in \mathcal{D}^1(\mathbb{R}^p)$ [2]. If ω is exact (say $d\varphi$) we get

$$[[a,b]](d\varphi) = \int_0^1 \langle \nabla\varphi((1-t)a + t + tb), b-a\rangle dt = \varphi(b) - \varphi(a).$$

This means - by the definition $\partial T(\varphi) = T(d\varphi)$ -

$$\partial[[a,b]] = [[b]] - [[a]],$$

where $[[b]](\varphi)$ is the $0-$current δ_b, that is $[[b]](\varphi) = \varphi(b)$ for all $0-$forms. This way it can be shown (by induction) that the boundary of a $p-$cell is given by the formula:

$$\begin{aligned}\partial([[a_1,b_1]] \times \ldots \times [[a_p,b_p]]) = \sum_{k=1}^{p} (-1)^k [[a_+,b_1]] \times \ \ldots \times [[a_{k-1},b_{k-1}]] \\ \times([[b_k]] - [[a_k]]) \times [[a_{k+1},b_{k+1}]] \times \ldots \times [[a_p,b_p]].\end{aligned}$$

This is a $(p-1)-$current which obviously coincides with the usual geometric intuition (one may think of any $p-$cell as a simplicial chain). We have used this formula to implement the boundary operator (`bdry`).

4 Implementation

First we will describe the exported functions of the domain `CellMap`. The choice of the name may be debatable, however, *ParametricSurface* and other suitable names are already taken. For the other domain, `SurfaceComplex`, we think it is an adequate designator what it is for.

4.1 Domain: `CellMap (CMAP)`

The constructor function is also called `cellMap` and requires a $p-$cell and and a mapping of type `List X` $\rightarrow$ `List X`, where we will denote by X the type `Expression R`, with R the ring in `CellMap(R,n)`. To keep things as simple as possible we did not introduce new types for points and vectors of fixed size. It will be the responsibility of the user to respect the necessary dimensions of the lists, that is n for the range and p for the domain of the mapping.

[2] See [3],4.1.8 for details

4.1.1 cellMap

```
cellMap: (List Segment X,List X -> List X) -> $
```

As a first example let Q be the 3−cell defined by

$$0 \leq r \leq 1, \quad , 0 \leq \theta \leq \pi, \quad , 0 \leq \varphi \leq 2\pi.$$

We translate this into SPAD language as (notice the coercion `1::X`)

```
Q:=[0..1::X,0..%pi,0..2*%pi]
```

Now define $\Phi(r, \theta, \varphi) = (x, y, z)$, where

$$\begin{aligned} x &= r\sin(\theta)\cos(\varphi) \\ y &= r\sin(\theta)\sin(\varphi) \\ z &= r\cos(\theta). \end{aligned}$$

Obviously, Φ maps Q onto the closed unit ball of $\mathbb{R}^3$, and it is bijective in the interior of Q (but certain boundary points will be identified by Φ, so it is not $1-1$ on Q!). Again we translate this to SPAD:

```
F:List X -> List x
F:=Z+->[Z.1*sin(Z.2)*cos(Z.3),Z.1*sin(Z.2)*sin(Z.3),Z.1*cos(Z.2)]
```

where $Z_1 = r, Z_2 = \theta, Z_3 = \varphi$. Note that the symbol Z is irrelevant, we could have used any other admissible symbol. This is quite convenient to specify mappings and, moreover, it makes `cellMap` independent of coordinate symbols. Eventually we can create the 3−surface:

```
B := cellMap(Q,F)$CellMap(R,3)
```

```
|[0..1,0..%pi,0..(2%pi)] -> [$ cos($ )sin($ ),$ sin($ )sin($ ),$ cos($ )]|
                             1     3      2   1     2      3   1     2
Type: CellMap(Integer,3)
```

Note the dollar signs in the output whose meaning is that of dummy variables (Z will not occur anymore).

For later reference we note down the *Jacobian* of Φ:

$$J_\Phi(r, \theta, \varphi) = \frac{\partial(x, y, z)}{\partial(r, \theta, \varphi)} = r^2 \sin(\theta),$$

and the integral over the volume form in $\mathbb{R}^3$:

$$\int_\Phi dx \wedge dy \wedge dz = \int_Q J_\Phi \, dr d\theta d\varphi = \frac{4\pi}{3}.$$

The latter just gives the volume of $\Phi(Q)$.

4.1.2 getDom, getMap

The two functions getDom and getMap will recover the $p-$cell (domain) and the mapping respectively when applied to a CellMap. The $p-$surfaces are stored as

```
Rep := Record(d:DOM,f:MAP)
```

S.dom and S.map also work synonymously.

4.1.3 apply: elt

Elements of CellMap(R,n) can be applied to an argument which has to be a list of dimension equal or greater to the dimension of the cell (domain). Taking our example it looks like:

```
B [r,th,phi::X]

[r cos(phi)sin(th),r sin(phi)sin(th),r cos(phi)]
Type: List(Expression(Integer))
```

4.1.4 faces

This function returns as its name indicates the faces of the $p-$surface, that is the images of the boundary of the domain ($p-$cell). Note that the returned list contains pairs of faces corresponding to the endpoints of intervals. This function will be useful when later dealing with SurfaceComplex. Our example gives:

```
faces B

[
  [|[0..%pi,0..(2%pi)] -> [0,0,0]|,
   |[0..%pi,0..(2%pi)] -> [cos($ )sin($ ),sin($ )sin($ ),cos($ )]|],
                                2      1       1      2       1

    [|[0..1,0..(2%pi)] -> [0,0,$ ]|,
                                1

     |[0..1,0..(2%pi)] -> [0,0,-$ ]|],
                                 1

    [|[0..1,0..%pi] -> [$ sin($ ),0,$  cos($ )]|,
                         1     2     1      2

    |[0..1,0..%pi] -> [$ sin($ ),0,$  cos($ )]|]]
                        1     2     1      2
Type: List(List(CellMap(Integer,3)))
```

We can nicely see how the six faces of the 3−cell Q are mapped to.

4.1.5 coords, coordSymbols

These two functions are just for convenience in order to introduce coordinates as well as coordinate symbols on which `CellMap`'s can be applied to. The symbols are convenient when differentiation is needed.

```
M ==> CellMap(R,3)

Y:=coords('y,3)$M

[y ,y ,y ]
  1  2  3
Type: List(Expression(Integer))

YS:=coordSymbols('y,3)$M
 [y ,y ,y ]
   1  2  3
Type: List(Symbol)
```

```
-- apply B to Y

Z := B Y

[y cos(y )sin(y ),y sin(y )sin(y ),y cos(y )]
  1     3      2   1     2      3   1     2

Type: List(Expression(Integer))

-- differentiate w.r.t. Y

[D(Z.j,YS.j) for j in 1..3]

     [cos(y )sin(y ),y cos(y )sin(y ),0]
           3      2   1     2      3
Type: List(Expression(Integer))
```

4.1.6 jacobianMatrix

The function `jacobianMatrix` returns a $n \times p-$matrix valued function which gives the Jacobian matrix at each point in the cell.

```
jacobianMatrix(B)


theMap(CMAP;jacobianMatrix;$M;9!0,655)
Type: (List(Expression(Integer)) -> Matrix(Expression(Integer)))
```

When applied to a point of the $3-$cell we get

```
jacobianMatrix(B) Y
```

$$\begin{bmatrix} \cos(y_3)\sin(y_2) & y_1\cos(y_2)\cos(y_3) & -y_1\sin(y_2)\sin(y_3) \\ \sin(y_2)\sin(y_3) & y_1\cos(y_2)\sin(y_3) & y_1\cos(y_3)\sin(y_2) \\ \cos(y_2) & -y_1\sin(y_2) & 0 \end{bmatrix}$$

Type: Matrix(Expression(Integer))

Taking the determinant

```
simplify determinant %
```

$$y_1{}^2\sin(y_2)$$

we obtain the expected result when recalling $y_1 = r$ and $y_2 = \theta$.

4.1.7 tangentSpace, normalSpace

These two functions compute the tangent and normal space at each point respectively. Again, the return value is a function with domain a cell and the range is a list of vectors which span the spaces. Better names might be *tangentBundle* and *normalBundle*, however, there should be no confusion as neither of the nomenclature is really satisfactory. We will just apply the resulting maps and get by out example:

```
tangentSpace(B) Y
normalSpace(B) Y
```

```
   [[cos(y )sin(y ),sin(y )sin(y ),cos(y )],
         3      2       2      3       2
    [y cos(y )cos(y ),y cos(y )sin(y ),- y sin(y )],
      1     2      3   1     2      3     1     2
    [- y sin(y )sin(y ),y cos(y )sin(y ),0]]
        1     2      3   1     3      2

Type: List(Vector(Expression(Integer)))


    []  -- normal space is empty here

Type: List(Vector(Expression(Integer)))
```

4.1.8 normalField

The function `normalField` computes the unit normal at each point of a hypersurface (that is $p = n - 1$). The return value is as usual a map that can be applied to a cell point. Since our example has $p = n$ we cannot use this function, that is a message tells us:

```
normalField(B)
```

```
>> Error detected within library code:
Not a hypersurface
```

We will provide an example later.

4.1.9 metricTensor

The metric tensor g of a $p-$surface is computed by using the (Euclidean) scalar product of the ambient space: $g_{ij} = \langle g_i, g_j \rangle$ where the g_i are the tangent vectors. Let us compute the determinant of g for our example 3−surface B:

```
simplify determinant (metricTensor(B) Y)
```

$$-{y_1}^4 \cos(y_2)^2 + {y_1}^4$$

Type: Expression(Integer)

Recall that the volume form is defined by $\sqrt{g}\, dy_1 \wedge dy_2 \wedge dy_3$, thus we obtain for B:

$$\eta_B = y_1^2 \sin(y_2)\ dy_1 \wedge dy_2 \wedge dy_3.$$

4.1.10 christoffelSymbols

The *Christoffel symbols* of the second type are defined as follows:

$$\Gamma^i{}_{kl} = \tfrac{1}{2} g^{im} \left(\frac{\partial g_{mk}}{\partial x^l} + \frac{\partial g_{ml}}{\partial x^k} - \frac{\partial g_{kl}}{\partial x^m} \right) = \tfrac{1}{2} g^{im} (g_{mk,l} + g_{ml,k} - g_{kl,m})$$

and are computed by christoffelSymbols using this formula. [3] The returned value is a list in i of matrices with elements of k, l.

```
christoffelSymbols(B) Y
```

$$\left[\begin{bmatrix} 0 & 0 & 0 \\ 0 & -y_1 & 0 \\ 0 & 0 & y_1 \cos(y_2)^2 - y_1 \end{bmatrix}, \begin{bmatrix} 0 & \frac{1}{y_1} & 0 \\ \frac{1}{y_1} & 0 & 0 \\ 0 & 0 & -\cos(y_2)\ \sin(y_2) \end{bmatrix}, \begin{bmatrix} 0 & 0 & \frac{1}{y_1} \\ 0 & 0 & \frac{\cos(y_2)}{\sin(y_2)} \\ \frac{1}{y_1} & \frac{\cos(y_2)}{\sin(y_2)} & 0 \end{bmatrix} \right]$$

Sometimes it is necessary to **simplify** the resulting expressions, especially when trigonometric functions are involved. The list of matrices above has been simplified using the following method:

[3] en.wikipedia.orgwiki/Christoffel_symbols, math.stackexchange.com/questions/1985964/christoffel-symbols-for-spherical-polar-coordinates

```
TM ==> TranscendentalManipulations(INT,EXPR INT)
C:=christoffelSymbols(B) Y
[map(simplify$TM,C.j) for j in 1..3]
```

The function *simplify* will replace $\sin(x)^2 + \cos(x)^2$ by 1 for example.

4.2 Domain: SurfaceComplex (SCMPLX)

The constructor function construct is just the same as cellMap but its return value is of type SurfaceComplex instead of CellMap. It is also possible to coerce and retract between the two types. Let us make an example where it is best seen how it goes. The cell map I embeds its domain $[0,1] \times [0,1]$ into three space:

```
(1) -> I:=cellMap([0..1,0..1],z+->[z.1,z.2,0])$CellMap(INT,3)

   (1)  |[0..1,0..1] -> [$ ,$ ,0]|
                         1  2
                                                Type: CellMap(Integer,3)
(2) -> IC:=I::SurfaceComplex(INT,3)

   (2)  |[0..1,0..1] -> [$ ,$ ,0]|
                         1  2
                                                Type: SurfaceComplex(Integer,3)
(3) -> bdry IC

   (3)
     - |[0..1] -> [$ ,1,0]| + |[0..1] -> [$ ,0,0]| - |[0..1] -> [0,$ ,0]|
                   1                      1                        1
   +
     |[0..1] -> [1,$ ,0]|
                   1
                                                Type: SurfaceComplex(Integer,3)
(4) -> bdry %

   (4)  0
                                                Type: SurfaceComplex(Integer,3)
```

The boundary of I_C comprises four cell maps with integer coefficients indicating the orientation so that $\partial \circ \partial = 0$.

4.2.1 bdry

The boundary operator bdry computes the quantity ∂T as defined in the previous section. Note, however, that the result should not be confused with the topological or geometric boundary. First of all we ware dealing with mappings (not sets) and second the cell maps Φ need not be $1-1$ on the whole (compact) cell. Yet this does not matter because $p-$dimensional integration in $n-$space is blind with respect to lower dimensional $k-$surfaces and due to cancellations as well. As in the case of Schwartz distributions,

each DeRham current has a support `spt` which in case of taking boundaries may or may not coincide with the topological one (this will depend on the mapping of course). To illustrate this we will do example 10.32 in [1]:

Let $Q = \{u, v : 0 \leq u \leq \pi, 0 \leq v \leq 2\pi\}$ be a $2-$cell and

$$\Sigma(u, v) = (\sin u \cos v, \sin u \sin v, \cos u).$$

Then Σ is a $2-$surface in three space, whose range is the unit sphere $\mathcal{S}^2$. We all know that $\partial\mathcal{S}^2 = 0$ in the topological sense, but let us now write down $\partial\Sigma$:

$$\partial\Sigma = \Sigma(\partial Q) = \gamma_1 + \gamma_2 + \gamma_3 + \gamma_4$$

where

$$\begin{aligned}
\gamma_1(u) &= \Sigma(u, 0) = (\sin u, 0, \cos u) \\
\gamma_2(v) &= \Sigma(\pi, v) = (0, 0, -1) \\
\gamma_3(u) &= \Sigma(\pi - u, 2\,\pi) = (\sin u, 0, -\cos u) \\
\gamma_4(v) &= \Sigma(0, 2\,\pi - v) = (0, 0, 1)
\end{aligned}$$

Rudin describes this in geographic terms as follows:

> $\partial\Sigma$ starts at the north pole N, runs to the south pole S along a meridian, pauses at S, returns to N along the same meridian, and finally pauses at N. The two passages along the meridian are in opposite directions. The corresponding two line integrals therefore cancel each other. In Exercise 32 there is also one curve which occurs twice in the boundary, but without cancellation.)

So it is obvious that we must have $\partial\Sigma = 0$ since the constant maps γ_2 and γ_4 have zero Jacobians and will not contribute to integration over one forms. The other two mappings, γ_1 and γ_3 are opposite and will cancel.

Next let us do the same in FriCAS.

```
R ==> Integer
X ==> Expression R

-- 2-cell [0,pi]x[0,2*pi]
Q:=[0::X..%pi,0..2*%pi]

-- F mapping [Z.1,Z.2] -> [x,y,z]
F:List X -> List X
F:=Z+->[sin(Z.1)*cos(Z.2),sin(Z.1)*sin(Z.2),cos(Z.1)]

S:=cellMap(Q,F)$CellMap(R,3)
```

First we compute the faces of the cell map S,

```
faces S

   [[|[0..(2%pi)] -> [0,0,1]|,|[0..(2%pi)] -> [0,0,- 1]|],
    [|[0..%pi] -> [sin($ ),0,cos($ )]|,|[0..%pi] -> [sin($ ),0,cos($ )]|]]
                          1         1                       1         1
                                                Type: List(List(CellMap(Integer,3)))
```

where we recognize the γ's without direction. Now we compute the actual boundary, obtaining:

```
bdry(S::SCMPLX(R,3))

     - |[0..((2%pi))] -> [0,0,1]| + |[0..(2%pi)] -> [0,0,- 1]|

                                      Type: SurfaceComplex(Integer,3)
```

which is obviously not 0, actually it is the south pole minus the north pole. However, the integral over any $1-$form is zero of course, so that $\partial\Sigma = 0$ by definition. It is clear how to resolve this puzzle at once: the one dimensional content of a point is zero, so the meaning of $\partial\Sigma = 0$ always is with respect to a measure (in this sense our `bdry S` is equivalent to 0). Note that we use the Riemann integral here and Jordan content, while geometric measure theory is mostly based on Hausdorff measure (see [3]). The reason for this degeneracy is also found in the the vanishing of the determinant of the metric g at the values 0 and 2π.

4.2.2 Example: Solid Torus

In order to get a feeling how to deal with the elements of `SCMPLX`, sometimes also known as affine chains, we will try to verify example 10.47 in [1]. Let $0 < a < b$ be fixed and let K be the $3-$cell determined by $0 \leq t \leq a, 0 \leq u \leq 2\pi$ and $0 \leq v \leq 2\pi$.The equations

$$\begin{aligned} x &= t\cos u \\ y &= (b + t\sin u)\cos v \\ z &= (b + t\sin u)\sin v \end{aligned}$$

describe a mapping Ψ into three space which is $1-1$ in the interior of K, such that $\Psi(K)$ is a solid torus. Its Jacobian is claimed to be

$$J_\Psi = \frac{\partial(x, y, z)}{\partial(t, u, v)} = t(b + t\sin u)$$

. Next the $2-$chain $\Phi = \partial\Psi$ is considered (details see loc. cit.), and it is claimed that Φ is simply the $2-$surface obtained by setting $t = a$ in the

equations above, with parameter cell $[0, 2\pi] \times [0, 2\pi]$ (modulo a circle, which does not contribute to the integrals). Moreover, the normal to Φ at $(u, v) \in D$ should be the vector

$$N(u, v) = a(b + a \sin u)\mathbf{n}(u, v)$$

where $\mathbf{n}(u, v) = (\cos u, \sin u \cos v, \sin u \sin v)$.

Now the following sequence of commands will verify these claims step by step.

```
)clear all
R ==> Integer
X ==> Expression R

-- 3-cell [0,a]x[0,2*pi]x[0,2*%pi]
Q:=[0..a::X,0..2*%pi,0..2*%pi]

-- F mapping [Z.1,Z.2] -> [x,y,z]
F:List X -> List X
F:=Z+->[Z.1*cos(Z.2),(b+Z.1*sin(Z.2))*cos(Z.3),
        (b+Z.1*sin(Z.2))*sin(Z.3)]

T:=cellMap(Q,F)$CellMap(R,3)

J:=jacobianMatrix(T) [t,u,v]

simplify determinant(J) -- t^2 sin(u) + b t

S:= bdry(T::SCMPLX(R,3)) -- boundary = circle + claimed map

S2:=nthFactor(S,2)      -- get the mapping

nf:=normalField(S2) [u,v]
dot(nf,nf) -- equals 1

-- sometimes one has to simplify trig. expressions
nf:=vector [simplify s for s in nf::List X]

ts:=tangentSpace(S2) [u,v]
nc:=cross(ts.1,ts.2) -- cross product of tangent vectors

r1:=simplify dot(nc,nf)
simplify (denominator(r1)^2) -- equals 1 => a^2 sin(u) + a
```

References

[1] Walter Rudin, *Principles of Mathematical Analysis*, International series in pure and applied mathematics, 1976, McGraw-Hill.

[2] Hassler Whitney. *Geometric Integration Theory*, Princeton Mathematical Series, No. 21. Literary Licensing, LLC.

[3] Herbert Federer. *Geometric Measure Theory*. Springer, Reprint of the 1st ed. Berlin, Heidelberg, New York 1969 edition.

[4] Jenny Harrison, *Operator Calculus of Differential Chains and Differential Forms*, to appear in the Journal of Geometric Analysis, 2013.
Url: `math.berkeley.edu/~harrison`

[5] Singer, I. M., and Thorpe, J. A.*Lecture Notes on Elementary Topology and Geometry*, Scott, Foresman and Company.

[6] Spivak, M. *Calculus on Manifolds*, W. A. Benjamin, Inc., New York, 1965.