Problem set on free resolutions of differential modules

Macaulay2 Mini-school, Minneapolis, 2023

Throughout, k denotes a field.

1. Let $R = k[x, y]$, $D = R^2$, and $\partial_D = \begin{pmatrix} xy & -x^2 \\ y^2 & -xy \end{pmatrix}$. The pair (D, ∂_D) gives a differential R-module.

(a) Prove that $H(D) \cong k$, and show there is no quasi-isomorphism between D and k. Suggestion: one can use M2 to find a generator for $H(D)$ (though M2 is by no means necessary for this problem).

(b) Prove that (D, ∂_D) is not a free flag. Equivalently: prove that, for all $f \in R$, we have

$$(D, \partial_D) \ncong (R^2, \begin{pmatrix} 0 & f \\ 0 & 0 \end{pmatrix}).$$

(c) Recall from the lecture that there is a free flag resolution of $(F, \partial_F) \xrightarrow{\simeq} (D, \partial_D)$, where $F = R^4$, and ∂_F is the matrix

$$\partial_F = \begin{pmatrix} 0 & y & x & 1 \\ 0 & 0 & 0 & x \\ 0 & 0 & 0 & -y \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

In fact, there is an isomorphism

$$(F, \partial_F) \cong (D, \partial_D) \oplus (R^2, \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}) \tag{1}$$

of differential modules. Construct an explicit isomorphism as in (1), using the "minimization" procedure described in the proof of [BE22, Proposition 4.1]. More explicitly:

(i) use the "1" entry to zero out the top row and last column of ∂_F, and

(ii) conjugate ∂_F by the matrices corresponding to the row/column operations in (i).

Aside: one goal for this week is to automate this "minimization" procedure in M2.

2. The point of this exercise is to explain how one comes up with the free flag resolution (F, ∂_F) in part 1(c). We begin with a definition:

Given a morphism $f : (A, \partial_A) \to (B, \partial_B)$ of differential modules, the **mapping cone of** f is the differential module $\text{cone}(f)$ with underlying module $B \oplus A$ and differential $\begin{pmatrix} \partial_B & f \\ 0 & -\partial_A \end{pmatrix}$. As in ordinary homological algebra, f is a quasi-isomorphism if and only if $\text{cone}(f)$ is exact.

Execute the following recursive algorithm to compute the free flag resolution (F, ∂_F) in 1(c); this process will, in particular, produce an explicit quasi-isomorphism $(F, \partial_F) \xrightarrow{\simeq} (D, \partial_D)$.

(a) Map a free module F_0 onto a set of cycles of D that descend to a minimal generating set of $H(D)$. This gives a morphism of differential modules $\varepsilon_0 : (F_0, 0) \to (D, \partial_D)$.

(b) If $\text{cone}(\varepsilon_0)$ is exact, we're done. Else, return to (a) with D replaced with $\text{cone}(\varepsilon_0)$.

I suggest you use M2 to compute the cycles and check exactness at each step. The following code should help get you started.

```
R = ZZ/101[x,y]
A = matrix{{x*y, -x^2}, {y^2, -x*y}}
d0 = map(R^2, R^2, A)
D0 = chainComplex(d0, d0) --this is an expression of our differential module D.
genList = mingens HH_1 D0 --a cycle that generates H(D)
conematrix = (A | genList) || map(source genList, source (A | genList), 0)
d1 = map(R^3, R^3, conematrix)
D1 = chainComplex(d1, d1) --this is cone(\epsilon_0)
```

This algorithm is similar to the usual one for building the minimal free resolution of a module. The output is always a free flag resolution, but, as you'll see, it need not be minimal.

3. We now turn to an application of differential modules to algebraic geometry. We let

$$S = k[x_0, x_1, x_2], \text{ where } \deg(x_0) = \deg(x_1) = 1 \text{ and } \deg(x_2) = 2; \text{ and}$$
$$E = \Lambda_k(e_0, e_1, e_2), \text{ where } \deg(e_0) = \deg(e_1) = (-1, -1) \in \mathbb{Z}^2 \text{ and } \deg(e_2) = (-2, -1) \in \mathbb{Z}^2.$$

Here, $\Lambda_k(e_0, e_1, e_2)$ denotes an exterior algebra over k on 3 variables. In this problem, a **differential E-module** is an E-module D equipped with a degree $(0, -1)$ differential. Let $\text{mod}(S)$ (resp. $\text{DM}(E)$) denote the category of finitely generated S-modules (resp. differential E-modules). Consider the functor

$$\mathbf{R} : \text{mod}(S) \to \text{DM}(E)$$

that sends M to the differential E-module $\bigoplus_{d \in \mathbb{Z}} M_d \otimes_k E(-d-4, -3)$ with differential given by $m \otimes e \mapsto \sum_{i=0}^{2} x_i m \otimes e_i e$.

Aside: the functor $\mathbf{R}$ determines an equivalence on bounded derived categories; this equivalence is an instance of the **toric BGG correspondence**. See [BE21, Section 2] for background. A goal for this week is to implement the toric BGG functors in M2, building on the existing BGG package for the standard graded case.

(a) Let $M = S/(x_0, x_1)$. Compute $\mathbf{R}(M)$.

(b) Compute the minimal free resolution $F \to \mathbf{R}(M)$. To do so, execute the first step of the algorithm in Problem (2), and observe that there is some periodic behavior. Suggestion to get started: use the isomorphism $H(\mathbf{R}(M))_{(d,i)} \cong \operatorname{Tor}_i^S(M,k)_d$ [BE21, Proposition 2.11].

(c) Let $T = \operatorname{cone}(F \to \mathbf{R}(M))$. (Aside: the object T is called a "Tate resolution" [BE21, EFS03].) Given $(i,j) \in \mathbb{Z}^2$, compute $\operatorname{Hom}_E(k, T)_{(j,-i)}$. Here, k is concentrated in degree $(0,0)$. Notice: the differential on T plays no role in this calculation.

Caution: since we're now in the graded setting, our definition of the mapping cone needs to be modified slightly. Given a morphism $f : D \to D'$ of differential E-modules, its mapping cone has underlying module $D' \oplus D(0,-1)$, with the same differential as defined in Problem (2).

You can check your answer for (c) in the following way. By results in [BE21], there is an isomorphism

$$H^i(X, \widetilde{M}(j)) \cong \operatorname{Hom}_E(k, T)_{(j,-i)}, \tag{2}$$

where X is the weighted projective stack $\mathbb{P}(1,1,2)$, and $\widetilde{M}$ is the sheaf on X associated to M. That is, what we're doing here is computing the sheaf cohomology of the stacky point $\widetilde{M}$ on X. If none of this stacky language is intelligible to you, don't worry: the main point is that the values on the left of line (2) can be computed in M2 as follows:

```
loadPackage "NormalToricVarieties"
X = weightedProjectiveSpace {1,1,2}
S = ring X
M = coker matrix{{x_0, x_1}}
F = sheaf(M)
i = 1
j = 2
HH^i(X, F**sheaf(S^{{j}}))
```

Plugging in various values of i and j, you should recover the formula you computed in (c). This idea leads to a BGG-flavored algorithm for computing sheaf cohomology over weighted projective stacks in M2, which we also aim to implement this week. This extends an algorithm due to Eisenbud-Fløystad-Schreyer over ordinary projective spaces [DE02, EFS03].

References

[BE21] Michael K. Brown and Daniel Erman, *Tate resolutions on toric varieties*, to appear in the Journal of the European Mathematical Society (JEMS) (2021).

[BE22] ———, *Minimal free resolutions of differential modules*, Trans. Amer. Math. Soc. **375** (2022), no. 10, 7509–7528.

[DE02] Wolfram Decker and David Eisenbud, *Sheaf algorithms using the exterior algebra*, Computations in algebraic geometry with Macaulay 2, Springer, 2002, pp. 215–249.

[EFS03] David Eisenbud, Gunnar Fløystad, and Frank-Olaf Schreyer, *Sheaf cohomology and free resolutions over exterior algebras*, Trans. Amer. Math. Soc. **355** (2003), no. 11, 4397–4426.