# CORRIGENDUM TO “ALGORITHMS FOR DETERMINATION OF t-MODULE STRUCTURES ON SOME EXTENSION GROUPS”

FILIP GŁOCH[1], DAWID E. KĘDZIERSKI [1], AND PIOTR KRASOŃ [1,*]

Abstract. In our recent paper, “Algorithms for Determination of **t**-Module Structures on Some Extension Groups,” published in International Journal of Number Theory, Vol. 21, No. 8 (2025) 1889–1922, we identified a gap in Algorithm 2 that affects Example 4.2. In this corrigendum, we describe the nature of this issue and present a corrected version of the algorithm.

In the paper *Algorithms for determination of $t$-module structures on some extension groups*, there is a local error in Algorithm 2, more precisely in the function `Reduce1`, in the case of strictly pure **t**-modules.

The error appears both in the printed pseudocode and in the corresponding *Mathematica* implementation given in the Appendix. In certain cases, the function `Reduce1` does not completely reduce the matrix $\boldsymbol{V}$.

In the published version, the reduction of $\boldsymbol{V}$ starts at the upper-left entry and proceeds row by row, from left to right. However, when reducing the entry $(i, j)$, it may happen that the degree of an earlier entry $(k, l)$ with $k \leq i$ and $l \leq j$ increases. In such a situation, the original function does not return to that entry and, therefore, leaves it unreduced.

We stress that this error is local. Both the underlying reduction procedure and the proof of correctness remain valid. In particular, all proofs and conclusions presented in the paper remain correct. The correction concerns only the explicit form of the procedure `Reduce1` for strictly pure $t$-modules, as well as the corresponding implementation.

The corrected version of the pseudocode is as follows.

**Algorithm 2** Corrected version of Reduce1

```
function Reduce1($\boldsymbol{V}, \Phi, \Psi, \boldsymbol{A}_n^{-1}$)          ▷ Reduction of t-module using
                                                        inverse matrix
    Input
        $\boldsymbol{V}$      module to be reduced
        $\Phi$      strictly pure t-module
        $\Psi$      a t-module of degree less than degree $\Phi$
        $\boldsymbol{A}_n^{-1}$   inverse of the leading matrix of $\Phi$
    Output
        $\boldsymbol{V}$      reduced module
```

---

2020 *Mathematics Subject Classification.* 11G09, 11R58,18G50.

*Key words and phrases.* Drinfeld modules, Anderson **t**-modules, group of extensions, Hom-Ext exact sequence, biderivations, composition series, **t**-reduction algorithm.

```
    $n \leftarrow \textsc{Rows}(\boldsymbol{V})$
    $m \leftarrow \textsc{Cols}(\boldsymbol{V})$
    $r \leftarrow \textsc{Deg}(\Phi)$
    for $i = 1..n$ do
        for $j = 1..m$ do
            $r' \leftarrow \textsc{Deg}(\boldsymbol{V}[i,j])$
            if $r' \geq r$ then
                $a \leftarrow \textsc{Coefficient}(\boldsymbol{V}[i,j], r')$
                $\boldsymbol{G} \leftarrow \textsc{TMult}(E_{i\times j} a \tau^{r'-r}, \boldsymbol{A}_{\boldsymbol{n}}^{\boldsymbol{-1}})$
                    ▷ we are setting the leading coefficient of $\boldsymbol{G}[i,j]$ to be the same as leading coefficient of $\boldsymbol{V}[i,j]$
                $\boldsymbol{G} \leftarrow \textsc{TMult}(\boldsymbol{G}, \Phi) - \textsc{TMult}(\Psi, \boldsymbol{G})$
                $\boldsymbol{V} \leftarrow \boldsymbol{V} - \boldsymbol{G}$
                $i \leftarrow 1$
                $j \leftarrow 0$
    return $\boldsymbol{V}$
```

Thus, whenever a reduction is performed, the procedure restarts the traversal from the entry $(1,1)$. This ensures that any earlier entry whose $\tau$-degree becomes greater than or equal to $r = \deg_\tau \Phi$ during a later reduction step is not left unreduced.

Accordingly, the implementation in the Appendix (p. 1918) should also be replaced by the following corrected version:

```
reduce1[v__,r__,φ__,ψ__,s__]:=Module[{i,j,n,m,rp,a,g,vp},

vp=v;

n = Dimensions[v][[1]];

m = Dimensions[v][[2]];

For[i=1,i<=n,i++,

For[j=1,j<=m,j++,

rp=deg [vp[[i]][[j]]];

If[rp>=r,

a=Coefficient[vp[[i]][[j]],τ ,rp];

g=tMult[elementMatrix[i,j,n,m]*a*τ ^(rp-r),s];

g=tMult[g,φ ]-tMult[ψ ,g];

vp =PowerExpand[vp-g];

i=1; j=0; ] ] ];   vp ];
```

As a consequence, Example 4.2 on pp. 1904–1905 and the example in the Appendix on p. 1918 should also be corrected accordingly.

*Example* 4.2. Let

$$\Phi_t = \begin{bmatrix} \theta & \tau^3 \\ 1+\tau^3 & \theta \end{bmatrix} \quad \text{and} \quad \Psi_t = \begin{bmatrix} \theta+\tau^2 & 0 \\ 1 & \theta+\tau \end{bmatrix}.$$

Then $N_\Phi = N_\Psi = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$, and $A_3^{-1} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$. One readily verifies that $E_{i\times j}A_3^{-1}N_\Phi = N_\Psi E_{i\times j}A_3^{-1}$ only for $(i,j) = (2,2)$. Thus $s = 1$. Accordingly, the reduction algorithm yields

$$\operatorname{Ext}_\tau^1(\Phi,\Psi) = \begin{bmatrix}
\theta & -\tau^2 & 0 & 0 & 0 & -\tau^4 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \theta & -\tau^2 & 0 & 0 & (\theta-\theta^{(1)})\tau^2 & 0 & 0 & 0 & 0 & 0 & 0 \\
\tau^2 & 0 & \theta+\tau^6 & 0 & \tau^4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \theta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & (\theta-\theta^{(1)})\tau^2 & 0 & \theta & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \tau^4 & 0 & \tau^2 & 0 & \theta+\tau^6 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & \tau^4 & 0 & \tau^2 & 0 & \theta & 0 & -\tau & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & \tau^2 & \tau & \theta & 0 & 0 & 0 & \tau^2 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & \tau & \theta & 0 & 0 & 0 \\
0 & \tau^2 & 0 & 1 & 0 & \tau^4 & 0 & 0 & 0 & \theta & 0 & 0 \\
0 & 0 & \tau^2 & 0 & 1 & 0 & 0 & 0 & \tau^2 & \tau & \theta & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \tau & \theta
\end{bmatrix}$$

*Example* 4.3 (Corrected example in the Appendix on p. 1918).

```
(*example – Ext¹(ϕ, ψ)*)

ψ = {{θ + τ^2, 0}, {b, θ + τ}};

ϕ = {{θ, τ^3}, {τ^3 + a, θ}};

ϕ = {{θ, τ^3}, {τ^3 + a, θ}};

ψ//MatrixForm

ϕ//MatrixForm

formatQ/@extInverse[ϕ, ψ]//MatrixForm
```

$$\begin{pmatrix} \theta+\tau^2 & 0 \\ b & \theta+\tau \end{pmatrix}$$

$$\begin{pmatrix} \theta & \tau^3 \\ a+\tau^3 & \theta \end{pmatrix}$$

$$\begin{pmatrix}
\theta & -a\tau^2 & 0 & 0 & 0 & -a\tau^4 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \theta & -a^{(1)}\tau^2 & 0 & 0 & (\theta-\theta^{(1)})\tau^2 & 0 & 0 & 0 & 0 & 0 & 0 \\
\tau^2 & 0 & \theta+\tau^6 & 0 & \tau^4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \theta & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & (\theta-\theta^{(1)})\tau^2 & 0 & \theta & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \tau^4 & 0 & \tau^2 & 0 & \theta+\tau^6 & 0 & 0 & 0 & 0 & 0 & 0 \\
b & 0 & b\tau^4 & 0 & b\tau^2 & 0 & \theta & 0 & -a\tau & 0 & 0 & 0 \\
0 & b & 0 & 0 & 0 & b\tau^2 & \tau & \theta & 0 & 0 & 0 & \tau^2 \\
0 & 0 & b & 0 & 0 & 0 & 0 & \tau & \theta & 0 & 0 & 0 \\
0 & b\tau^2 & 0 & b & 0 & b\tau^4 & 0 & 0 & 0 & \theta & 0 & 0 \\
0 & 0 & b\tau^2 & 0 & b & 0 & 0 & 0 & \tau^2 & \tau & \theta & 0 \\
0 & 0 & 0 & 0 & 0 & b & 0 & 0 & 0 & 0 & \tau & \theta
\end{pmatrix}$$

To summarize, the correction concerns:

- Algorithm 2, function `Reduce1`, in the case of strictly pure $t$-modules,
- the corresponding *Mathematica* implementation in the Appendix,
- Example 4.2 on pp. 1904–1905,
- the example in the Appendix on p. 1918.

All proofs and all conclusions of the paper remain correct.

[1] Institute of Mathematics, Department of Exact and Natural Sciences, University of Szczecin, ul. Wielkopolska 15, 70-451 Szczecin, Poland
*Email address*: `filip.gloch@gmail.com`

*Corresponding author

*Email address*: `dawid.kedzierski@usz.edu.pl`

*Email address*: `piotrkras26@gmail.com`