MATHML FOR DATA EXCHANGE

Robert Hornych, Zdeněk Hurák, Michael Šebek

Centre for Applied Cybernetics and
Department of Control Engineering
Czech Technical University, Prague
fax: +420-2-24916648
e-mail: hornycr@quick.cz, z.hurak, m.sebek@c-a-k.cz

Abstract: This paper deals with MathML (Mathematical Markup Language) utilization for writing mathematics elements and structures. It is useful for graphics presentations and data exchange with others products like Maple and Mathematica. For this purpose functions working in MATLAB workspace were developed.

Keywords: MathML, POL2MML, MML2POL, data exchange

1. BASIC DEFINITIONS

Polynomial Toolbox (2) for Matlab^{TM}(3) is a commercial Matlab-based package for systems, signals and control analysis and design based on advanced polynomial methods. Being developed by leading specialists in the field, the toolbox offers by far the most complete set of reliable algorithms for computation with polynomials and polynomial matrices. All the algorithms are numerical in nature and as such they are several orders of magnitude faster than their symbolic counterparts found in some computer algebra systems.

The toolbox relies on the object oriented programming features of Matlab. It defines a new Matlab class POL for storing all the information necessary for full specification of a polynomial matrix with constant coefficients.

Consider a simple polynom (1)

$P(s) = s^2 + 8.70605040302010$

Entering a polynomial matrix in Matlab prompt is very convenient as the Polynomial Toolbox follows the general Matlab syntax

```
>> P = s^2 + 8.70605040302010
P =
     8.7 + s^2
```

MathML is a new markup language for encoding the structure of mathematical expressions so that they can be displayed, manipulated and shared over the World Wide Web. *MathML* expression can be evaluated in a computer algebra system, rendered in a web browser, edited in a word processor, and printed on your laser printer. *MathML* 2.0, a W3C Recommendation was released on 21 Feb 2001. It is product of the W3C Math working group (1).

2. MATHML IN MATLAB

In MATLAB environment are implemented two functiones (`POL2MML`, `MML2POL`) for the export/import of polynomial matrices to/from *MathML*.

2.1 *POL2MML*

Return the *MathML* code corresponding to the POL representation of a polynoial matrix. All three forms of *MathML* code presentation can be explain by the following short example. For the above presented polynom (1) we obtain:

MathML code in presentation form:

```
>> pol2mml(P,'pm')
<math xmlns="http://www.w3.org/1998/Math/MathML">
<mrow>
<mi>P</mi>
<mo>&ApplyFunction;</mo>
<mfenced><mi>s</mi></mfenced>
<mo>=</mo>
<mn>8.71</mn>
<mo>+</mo>
<msup><mi>s</mi><mn>2</mn></msup>
</mrow>
</math>
```

MathML code in content form:

```
>> pol2mml(P,'cm')
<math xmlns="http://www.w3.org/1998/Math/MathML">
<apply>
  <eq/>
  <apply>
    <fn><ci>P</ci></fn>
    <ci>s</ci>
  </apply>
  <apply>
    <plus/>
    <cn type="real">8.706050403020100</cn>
    <apply>
      <power/>
      <ci>s</ci>
      <cn>2</cn>
    </apply>
  </apply>
</apply>
</math>
```

MathML code in mixed form [1] :

```
>> pol2mml(P)
<math xmlns="http://www.w3.org/1998/Math/MathML">
<semantics>
  <mrow>
  <mi>P</mi>
  <mo>&ApplyFunction;</mo>
  <mfenced><mi>s</mi></mfenced>
  <mo>=</mo>
  <mn>8.71</mn>
  <mo>+</mo>
  <msup><mi>s</mi><mn>2</mn></msup>
  </mrow>
  <annotation-xml encoding="MathML-Content">
  <apply>
    <eq/>
    <apply>
      <fn><ci>P</ci></fn>
      <ci>s</ci>
    </apply>
    <apply>
      <plus/>
      <cn type="real">8.706050403020100</cn>
      <apply>
        <power/>
        <ci>s</ci>
        <cn>2</cn>
      </apply>
    </apply>
  </apply>
  </annotation-xml>
</semantics>
</math>
```

By the expert experience the presentation form is useful for the web presentation (6), (7) and the content form for the data exchange. The mixed form contains advantages of both forms. Several features of the `POL2MML` function are listed below:

- Presentation, content and mixed markup are supported. Default method is mixed markup.
- Mixed markup is implemented via XML annotation mechanism.
- Optional insertion of Java applet tags specifying "webeq.Main" for the `code` parameter. Intended for WebEQ Math Viewer display method (8).
- The format of coefficients in the presentation markup is inherited from the display format of POL object in Matlab or can be specified explicitly.
- The format of coefficients in the content markup is 15-digit scaled fixed point (*long* format in Matlab) to assure as low loss of precision due to truncation as possible.
- Optionally copies the output in the clipboard (on Windows systems only).
- Optionally saves the MathML code to a specified file.
- Possible insertion of XML namespaces (m:) for displaying with MathPlayer engine (7).

[1] pol2mml(P,'mm') is the same as pol2mml(P) and it means the mixed markup export

2.2 *MML2POL*

The developers of both numerical and symbolic algorithms for polynomial matrices as well as practising engineers need to compare the results across the computational platforms. *MathML* appears to be a good format for this purpose. Indeed, packages like Mathematica and Maple in their latest versions support *MathML* language. MATLAB function `MML2POL` was developed that takes *MathML* code either from a specified file or directly from the clipboard (on Windows systems) and creates a corresponding POL object. Detailed error report is produced if inconsistencies in code are encountered.

By June 2003 the version 4.0 of `MML2POL` is released (2). Features and compatibility were improved.

3. MATHML IN MAPLE 8

MathML in Maple (4) has been implemented since the version 7. Logically, there were some imperfection. By the version 8 well compatibility is guaranteed. Maple supports all three forms of *MathML* markup.

3.1 *Data exchange from MATLAB to Maple*

The `POL2MML` function copies the *MathML* code to the clipboard. Using the paste (Ctrl-V) in the Maple window the conversion to Maple workspace is offered.

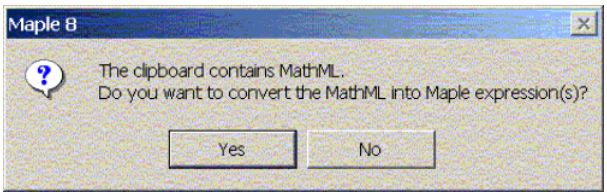

Fig. 1. Import *MathML* code into Maple 8

Importing the example (1) we obtain easily:

```
> P(s) = s^2+8.706050403;
                    2
          P(s) = s  + 8.706050403
```

There was a problem with the presentation form; Maple doesn't understand well the `&Applyfunction` construction. In this case is recommended to use `pol2mml(P,'pm','noname')` and better
`pol2mml(P,'pm','noname','%15f')` to save the precision [2] .

[2] Note that the presentation markup is determined for the web presentation, not for data exchange. It's implemented because sometimes it's required.

3.2 *Data exchange from Maple to MATLAB*

In the Maple workspace we introduce:

```
> P := s^2 + 8.70605040302010;
                    2
          P := s  + 8.70605040302010
```

MathML code in presentation form

```
> MathML[ExportPresentation](P);
> XMLTools[Print](%);
<math xmlns='http://www.w3.org/1998/Math/MathML'>
  <mrow>
    <msup>
      <mi>s</mi>
      <mn>2</mn>
    </msup>
    <mo>+</mo>
    <mn>8.70605040302010</mn>
  </mrow>
</math>
```

MathML code in content form

```
> MathML[ExportContent](P);
> XMLTools[Print](%);
<math xmlns='http://www.w3.org/1998/Math/MathML'>
  <apply id='id5'>
    <plus/>
    <apply id='id3'>
      <power/>
      <ci id='id1'>s</ci>
      <cn id='id2' type='integer'>2</cn>
    </apply>
    <cn id='id4' type='real'>8.70605040302010</cn>
  </apply>
</math>
```

MathML code in mixed form

```
> MathML[Export](P);
> XMLTools[Print](%);
<math xmlns='http://www.w3.org/1998/Math/MathML'>
  <semantics>
    <mrow xref='id5'>
      <mrow xref='id3'>
        <msup>
          <mi xref='id1'>s</mi>
          <mn xref='id2'>2</mn>
        </msup>
      </mrow>
      <mo>+</mo>
      <mn xref='id4'>8.70605040302010</mn>
    </mrow>
    <annotation-xml encoding='MathML-Content'>
      <apply id='id5'>
        <plus/>
        <apply id='id3'>
          <power/>
          <ci id='id1'>s</ci>
          <cn id='id2' type='integer'>2</cn>
        </apply>
        <cn id='id4' type='real'>8.70605040302010</cn>
      </apply>
    </annotation-xml>
    <annotation encoding='Maple'>
      s^2+8.70605040302010
    </annotation>
  </semantics>
</math>
```

In all cases we call the `MML2POL` function and obtain:

```
>> mml2pol
Getting the string from the clipboard.
ans =
     8.7 + s^2
```

Note that the `MML2POL` function get the contain of the clipboard and import the *MathML* code [3].

4. MATHML IN MATHEMATICA 4.2

The MathML export/import has been implemented sice the Mathematica (5) 4.1 version. In this version only the presentation form was supported. By the version 4.2 it generates the mixed markup form, but some bugs have stilled there.

4.1 *Data exchange from MATLAB to Mathematica*

Using the clipboard we can import *MathML* code into Mathematica workspace (Ctrl+V).

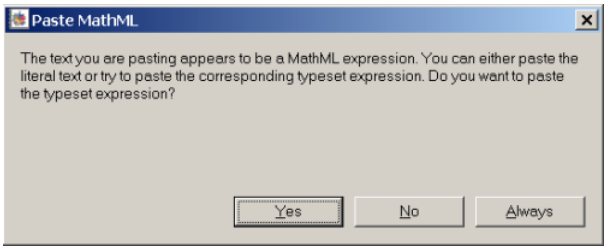

Fig. 2. Import *MathML* code into Mathematica 4.2

For MathML code from example (1) Mathematica offers

- code generated by pol2mml(P,'pm');

  ```
  > P[s] = 8.71 + s\^2
  ```

- code generated by pol2mml(P,'cm');

  ```
  > P[s] == s\^2 + 8.7060504030201‘
  ```

- code generated by pol2mml(P,'mm');

  ```
  > P[s] == s\^2 + 8.7060504030201‘
  ```

Uunfortunately you have to modify the offered command to follow Mathematica syntax.

```
> P = s\^2 + 8.7060504030201‘
```

It's more easy use the *MathML* code without the variable name:

- code generated by pol2mml(P,'pm','noname');

  ```
  > 8.71 + s\^2
  ```

- code generated by pol2mml(P,'cm','noname');

  ```
  > s\^2 + 8.7060504030201‘
  ```

[3] String assignment in MATLAB workspace is like str='all "my" text'. In Maple for example we can write str:="all 'my' text"; `MML2POL` and Maple have the input filter implemented.

- code generated by pol2mml(P,'mm','noname');

```
> s\^2 + 8.7060504030201‘
```

4.2 *Data exchange from Mathematica to MATLAB*

MathML code in mixed form we obtain by following commands:

```
> P=s^2+8.70605040302010
> MathMLForm[P]
<math xmlns='http://www.w3.org/1998/Math/MathML'>
 <semantics>
  <mrow>
   <msup>
    <mi>s</mi>
    <mn>2</mn>
   </msup>
   <mo>+</mo>
   <mn>8.7060504030201‘</mn>
  </mrow>
  <annotation-xml encoding='MathML-Content'>
   <apply>
    <plus/>
    <apply>
     <power/>
     <ci>s</ci>
     <cn type='integer'>2</cn>
    </apply>
    <cn type='real'>8.70605</cn>
   </apply>
  </annotation-xml>
 </semantics>
</math>
```

Then remain copy the *MathML* code to the clipboard and execute the `MML2POL` function.

```
>> mml2pol
Getting the string from the clipboard.
ans =
     8.7 + s^2
```

Note that the `MML2POL` function has a filter to correct some imperfection in the produced *MathML* code like:

- `<mn>8.7060504030201‘</mn>` [4]

Unfortunately numbers in the content part are not saved with the maximum precision. Perhaps Mathematica offers some options for the export.

5. CONCLUSIONS

Thanks to research well compatibility was reached and the desiderative aim was targetted. In spite of some imperfections in produced *MathML* code well usability is offered to engineers working with products like MATLAB, Maple and Mathematica.

ACKNOWLEDGMENTS

The work of the first author has been supported by the Grant Agency of the Czech Republic grant No. 102/02/0709. The work of the second author has been supported by the Ministry of Education of the Czech Republic under contract No. LN00B096.

[4] There hasn't to be the apostrof.

6. REFERENCES

[1] "MathML 2.0, a W3C Recommendation.", http://www.w3.org/Math, 2003
[2] Polyx, Ltd. "Polynomial Toolbox for Matlab", http://www.polyx.com, 2003
[3] The MathWorks, Inc. "Matlab", http://www.mathworks.com/products, 2003
[4] Waterloo Maple, Inc. "Maple 8", http://www.maplesoft.com/products, 2003
[5] Wolfram Research, Inc. "Mathematica 4.2", http://www.wolfram.com/products, 2003
[6] IBM, Ltd. "IBM techexplorer Hypermedia Browser", http://www-3.ibm.com/software/network/techexplorer/, 2002
[7] Design Science, Inc. "MathPlayer", http://www.dessci.com/webmath/mathplayer/, 2002
[8] Design Science, Inc. "WebEQ Viewer Control", http://www.mathtype.com/webmath/webeq/features.stm#viewer, 2001