Multivariate Polynomial Toolbox

Pete Seiler

1 Installation

- The toolbox was tested with MATLAB version 6.5 and, to a lesser degree, versions 6.0 and 6.1.
- Download the zip file and extract the contents to the directory where you want to install the toolbox.
- Add the `mulitpoly` directory and all subdirectories to the path. Note that the toolbox will not work if you are currently in the `@polynomial` directory. This is due to MATLAB's handling of class methods.
- Currently no mex functions exist.

2 Basic Manipulations

- Use the `pvar` command to create polynomial variables. For example, the following command creates three variables:

```
>> pvar x1 x2 x3
```

- Polynomial objects can now be created from these variables using addition, multiplication, and exponentiation:

```
>> p = x3^4+5*x2+x1^2
p =
  x3^4 + 5*x2 + x1^2
```

- Matrices of polynomials can be created from polynomials using horizontal/vertical concatenation and block diagonal augmentation:

```
>> M1=[p 2*x2]
M1 =
  [ x3^4 + 5*x2 + x1^2 , 2*x2 ]
>> M2=[p; 2*x2]
M2 =
  [ x3^4 + 5*x2 + x1^2 ]
  [               2*x2 ]
>> M3 = blkdiag(p,2*x2)
```

```
M3 =
  [ x3^4 + 5*x2 + x1^2 ,    0 ]
  [                   0 , 2*x2 ]
```

- Elements of a polynomial matrix can be referenced and assigned using the standard MATLAB referencing scheme:

```
>> M3
M3 =
  [ x3^4 + 5*x2 + x1^2 ,    0 ]
  [                   0 , 2*x2 ]
>> M3(2,2)
ans =
  2*x2
>> M3(1,1:2)
ans =
  [ x3^4 + 5*x2 + x1^2 , 0 ]
>> M3(1,2)=x1*x2
M3 =
  [ x3^4 + 5*x2 + x1^2 , x1*x2 ]
  [                   0 ,  2*x2 ]
```

- Let p denote an $N \times M$ polynomial in V variables consisting of T terms. This polynomial is stored as an $T \times NM$ sparse coefficient matrix, a $T \times V$ degree matrix, and a $V \times 1$ cell array of variable names. This information can be easily accessed:

```
>> p
p =
  x3^4 + 5*x2 + x1^2
>> pcoef = p.coefficient
pcoef =
   (1,1)        1
   (2,1)        5
   (3,1)        1
>> full(pcoef)
ans =
     1
     5
     1
>> full(p.degmat)
ans =
     0     0     4
     0     1     0
     2     0     0
>> p.varname
ans =
    'x1'
    'x2'
    'x3'
```

- Below is an example showing the '.'-reference for a matrix. It probably seems more natural to represent the coefficient matrix as an NxMxT array of coefficients. However, MATLAB does not support 3D sparse arrays. To exploit sparsity, the coefficient matrix is stored as an TxNM array.

```
>> M3
M3 =
  [ x3^4 + 5*x2 + x1^2 , x1*x2 ]
  [                  0 ,  2*x2 ]
>> N=2;M=2;T=4;V=3;
>> M3coef = M3.coefficient;
>> size(M3coef)
ans =
     4     4
>> temp = full(M3coef);
>> shiftdim(reshape(temp,T,N,M),1)
ans(:,:,1) =
     1     0
     0     0
ans(:,:,2) =
     5     0
     0     2
ans(:,:,3) =
     0     1
     0     0
ans(:,:,4) =
     1     0
     0     0
>> full(M3.degmat)
ans =
     0     0     4
     0     1     0
     1     1     0
     2     0     0
>> M3.varname
ans =
    'x1'
    'x2'
    'x3'
```

- The access to fields uses a case insensitive, partial-match. As shown below abbreviations can also be used to obtain the coefficients, degrees, and variable names. Finally, tab completion exists for accessing the field names.

```
>> p
p =
  x3^4 + 5*x2 + x1^2
>> full(p.c)
ans =
     1
     5
     1
>> full(p.d)
ans =
     0     0     4
     0     1     0
     2     0     0
>> p.v'
ans =
    'x1'    'x2'    'x3'
```

- A few additional operations exist in this initial version of the toolbox. Shown below are trace, transpose, determinant, differentiation, logical equal and logical not equal:

```
>> M3
M3 =
  [ x3^4 + 5*x2 + x1^2 , x1*x2 ]
  [                  0 ,  2*x2 ]
>> trace(M3)
ans =
  x3^4 + 7*x2 + x1^2
>> M3'
ans =
  [ x3^4 + 5*x2 + x1^2 ,    0 ]
  [              x1*x2 , 2*x2 ]
>> M3=M3+[0 0; 1 0]
M3 =
  [ x3^4 + 5*x2 + x1^2 , x1*x2 ]
  [                  1 ,  2*x2 ]
>> det(M3)
ans =
  2*x2*x3^4 + 10*x2^2 - x1*x2 + 2*x1^2*x2
>> diff(M3,x1)
ans =
  [ 2*x1 , x2 ]
  [    0 ,  0 ]
>> M3==x1*x2
ans =
     0     1
     0     0
>> M3~=(x3^4+5*x2+x1^2)
ans =
     0     1
     1     1
```