Determinant and Laplace expansion

Robert Mařík

27. listopadu 2010

This worksheet defines new class for working with unevaluated determinant. For pedagogical purpouses.

1 Class definition

Sage code

```
class LaplaceExpansion(SageObject):
    """
    Class for explaining Laplace expansion and operations with determinant.

    For pedagogical purpouses only.

    Objects in this class are lists of pairs ``[A,i]`` where ``A`` is matrix and
    ``i`` is a number. The value is sum of all terms ``i*det(A)``.

    Methods defined on this class do not change matrices.

    AUTHORS:

    - Robert Marik (03-2010)
    """

    def __init__(self, input):
        r"""
        Initialization for LaplaceExpansion class

        ``input`` is either matrix or list of pairs [matrix,number]
        """
        value = 0
        if not isinstance(input,list):
            input = [[input,1]]
        for i in input:
            value += i[1]*(i[0].det())
        self.value = value
        self._data_ = input

    def _repr_(self):
        s = ''
        for i in self._data_:
            s = s + '+(%s)*determinant(\n%s\n)'%(i[1],i[0])
        return "Laplace expansion: \n"+s[1:]

    def show(self):
        return html(r'$%s$'%latex(self))
```

[0]Podporováno grantem FRVŠ 131/2010.

```
    def _latex_(self):
        save_del = latex.matrix_delimiters()
        latex.matrix_delimiters("|", "|")
        expr = ''
        for i in self._data_:
            if len(i[0].rows())==1:
                ii = '(%s)' %latex(i[0][0][0])
            else:
                ii = latex(i[0])
            #ii = latex(i[0])
            if i[1] == 1:
                expr = expr + '+%s' % (ii)
            elif i[1] == 0:
                expr = expr + '+0'
            else:
                expr = expr + '+(%s) %s' % (i[1],ii)
        latex.matrix_delimiters("(", ")")
        return expr[1:]

    def __add__(self,s):
        return LaplaceExpansion(self._data_+s._data_)

    def expand(self, row = None, column = None, term = 0, remove_zeros = True):
        r"""
        Returns Laplace expansion along given row or column.

        INPUT:

        - ``row`` - row for expansion (if ``column`` is None)
        - ``column`` - column for expansion (if ``row`` is None)
        - ``term`` - term for expansion, if ``self`` is a Laplace expansion with more terms
        - ``remove_zeros`` - if True, do not include minors which belong to zero elements

        Does not change ``self``. If neither row nor column are specified, uses the first row.
        """
        b = self._data_[term]
        s = []
        if row == None and column == None:
            row = 0
        if column == None:
            col = len(b[0].rows())
            for ii in range(col):
                s = s + [[matrix([[b[0][i][j] for j in range(col) if j!=ii]\
                 for i in range(col) if i!=row]),(-1)^(row+ii)*b[1]*b[0][row][ii]]]
            if remove_zeros:
                s = [w for w in s if w[1] != 0]
            ans = self._data_[:term]+s+self._data_[term+1:]
            return LaplaceExpansion(ans)
        if row == None:
            col = len(b[0].rows())
            for ii in range(col):
                s = s + [[matrix([[b[0][i][j] for j in range(col) if j!=column]\
                 for i in range(col) if i!=ii]),(-1)^(column+ii)*b[1]*b[0][ii][column]]]
            if remove_zeros:
                s = [w for w in s if w[1] != 0]
            ans = self._data_[:term]+s+self._data_[term+1:]
            return LaplaceExpansion(ans)
```

```
    def pivot_on(self, i = 0 , j = 0, term = 0):
        r"""
        Cleans elements in column j pivoting with the element in row i

        Does not change ``self``.
        """
        b = self._data_[term]
        ans_temp = [vector(s)-vector(b[0][i])*s[j]/b[0][i][j] for s in b[0].rows()]
        ans_temp[i] = b[0][i]  # restore pivot row
        ans = self._data_[:term]+[[matrix(ans_temp),b[1]]]+self._data_[term+1:]
        return LaplaceExpansion(ans)

    def add_multiple_of_row(self, i , j, s, term = 0):
        r"""
        Add s times row j to row i in given term.

        Does not change ``self``.
        """
        b = copy(self._data_[term][0][:])
        b.add_multiple_of_row(i,j,s)
        ans = self._data_[:term]+[[b,self._data_[term][1]]]+self._data_[term+1:]
        return LaplaceExpansion(ans)

    def collect(self):
        r"""
        Collects all determinants of order 1 at the end.
        """
        s = 0
        d = []
        for i in self._data_:
            if len(i[0].rows())!=1:
                d = d + [i]
            else:
                s=s+i[0][0][0]*i[1]
        if s != 0:
            d = d + [[matrix([[s]]),1]]
        return LaplaceExpansion(d)
```

2 Example

Sage code

```
A=matrix([[2,3,4,1],[1,1,4,0],[7,6,5,1],[1,1,1,0]])
B=LaplaceExpansion(A)
B
```

$$\begin{vmatrix} 2 & 3 & 4 & 1 \\ 1 & 1 & 4 & 0 \\ 7 & 6 & 5 & 1 \\ 1 & 1 & 1 & 0 \end{vmatrix}$$

We multiply the first row by -1 and add to the third row

Sage code

```
B.add_multiple_of_row(2,0,-1)
```

$$\begin{vmatrix} 2 & 3 & 4 & 1 \\ 1 & 1 & 4 & 0 \\ 5 & 3 & 1 & 0 \\ 1 & 1 & 1 & 0 \end{vmatrix}$$

Laplace expansion along the last column

Sage code

```
C=_.expand(column=3)
C
```

$$(-1)\begin{vmatrix} 1 & 1 & 4 \\ 5 & 3 & 1 \\ 1 & 1 & 1 \end{vmatrix}$$

We multiply the last row by -1 and add to the first row

Sage code

```
C.add_multiple_of_row(0,2,-1)
```

$$(-1)\begin{vmatrix} 0 & 0 & 3 \\ 5 & 3 & 1 \\ 1 & 1 & 1 \end{vmatrix}$$

We use Laplace expansion along the first row

Sage code

```
D=_.expand()
D
```

$$(-3)\begin{vmatrix} 5 & 3 \\ 1 & 1 \end{vmatrix}$$

We evaluate the determinat using rule $\begin{vmatrix} 5 & 3 \\ 1 & 1 \end{vmatrix} = 5 \cdot 1 - 3 \cdot 1 = 2$ and multiply by -3

Sage code

```
D.value
```

-6

3 Row operations

Sage code

```
(B.add_multiple_of_row(1,0,-1)).add_multiple_of_row(2,0,-1)
```

$$\begin{vmatrix} 2 & 3 & 4 & 1 \\ -1 & -2 & 0 & -1 \\ 5 & 3 & 1 & 0 \\ 1 & 1 & 1 & 0 \end{vmatrix}$$

Sage code

```
B.pivot_on(3,2)
```

$$\begin{vmatrix} -2 & -1 & 0 & 1 \\ -3 & -3 & 0 & 0 \\ 2 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{vmatrix}$$

Sage code

```
B.pivot_on(3,0)
```

$$\begin{vmatrix} 0 & 1 & 2 & 1 \\ 0 & 0 & 3 & 0 \\ 0 & -1 & -2 & 1 \\ 1 & 1 & 1 & 0 \end{vmatrix}$$

Sage code

```
B.pivot_on(2,2)
```

$$\begin{vmatrix} -\frac{18}{5} & -\frac{9}{5} & 0 & \frac{1}{5} \\ -\frac{23}{5} & -\frac{19}{5} & 0 & -\frac{4}{5} \\ 7 & 6 & 5 & 1 \\ -\frac{2}{5} & -\frac{1}{5} & 0 & -\frac{1}{5} \end{vmatrix}$$

Sage code

```
B.pivot_on(3,1)
```

$$\begin{vmatrix} -1 & 0 & 1 & 1 \\ 0 & 0 & 3 & 0 \\ 1 & 0 & -1 & 1 \\ 1 & 1 & 1 & 0 \end{vmatrix}$$

Sage code

```
B
```

$$\begin{vmatrix} 2 & 3 & 4 & 1 \\ 1 & 1 & 4 & 0 \\ 7 & 6 & 5 & 1 \\ 1 & 1 & 1 & 0 \end{vmatrix}$$

4 Laplace expansion

Sage code

```
B.expand(column=3)
```

$$(-1)\begin{vmatrix} 1 & 1 & 4 \\ 7 & 6 & 5 \\ 1 & 1 & 1 \end{vmatrix} + (-1)\begin{vmatrix} 2 & 3 & 4 \\ 1 & 1 & 4 \\ 1 & 1 & 1 \end{vmatrix}$$

Sage code

```
BB=(B).expand(row=1)
BB
```

$$(-1)\begin{vmatrix} 3 & 4 & 1 \\ 6 & 5 & 1 \\ 1 & 1 & 0 \end{vmatrix} + \begin{vmatrix} 2 & 4 & 1 \\ 7 & 5 & 1 \\ 1 & 1 & 0 \end{vmatrix} + (-4)\begin{vmatrix} 2 & 3 & 1 \\ 7 & 6 & 1 \\ 1 & 1 & 0 \end{vmatrix}$$

Sage code

```
(BB).expand(column=2,term=1)
```

$$(-1)\begin{vmatrix} 3 & 4 & 1 \\ 6 & 5 & 1 \\ 1 & 1 & 0 \end{vmatrix} + \begin{vmatrix} 7 & 5 \\ 1 & 1 \end{vmatrix} + (-1)\begin{vmatrix} 2 & 4 \\ 1 & 1 \end{vmatrix} + (-4)\begin{vmatrix} 2 & 3 & 1 \\ 7 & 6 & 1 \\ 1 & 1 & 0 \end{vmatrix}$$

Sage code

```
(BB).expand(row=0,term=2).expand(row=2)
```

$$(-1)\begin{vmatrix} 4 & 1 \\ 5 & 1 \end{vmatrix} + \begin{vmatrix} 3 & 1 \\ 6 & 1 \end{vmatrix} + \begin{vmatrix} 2 & 4 & 1 \\ 7 & 5 & 1 \\ 1 & 1 & 0 \end{vmatrix} + (-8)\begin{vmatrix} 6 & 1 \\ 1 & 0 \end{vmatrix} + (12)\begin{vmatrix} 7 & 1 \\ 1 & 0 \end{vmatrix} + (-4)\begin{vmatrix} 7 & 6 \\ 1 & 1 \end{vmatrix}$$

Sage code

```
(BB).expand(row=0,term=2).expand(row=2).expand(row=0, term=1)
```

$$(-1)\begin{vmatrix} 4 & 1 \\ 5 & 1 \end{vmatrix} + (3)(1) + (-1)(6) + \begin{vmatrix} 2 & 4 & 1 \\ 7 & 5 & 1 \\ 1 & 1 & 0 \end{vmatrix} + (-8)\begin{vmatrix} 6 & 1 \\ 1 & 0 \end{vmatrix} + (12)\begin{vmatrix} 7 & 1 \\ 1 & 0 \end{vmatrix} + (-4)\begin{vmatrix} 7 & 6 \\ 1 & 1 \end{vmatrix}$$

Sage code

```
(_).collect()
```

$$(-1)\begin{vmatrix} 4 & 1 \\ 5 & 1 \end{vmatrix} + \begin{vmatrix} 2 & 4 & 1 \\ 7 & 5 & 1 \\ 1 & 1 & 0 \end{vmatrix} + (-8)\begin{vmatrix} 6 & 1 \\ 1 & 0 \end{vmatrix} + (12)\begin{vmatrix} 7 & 1 \\ 1 & 0 \end{vmatrix} + (-4)\begin{vmatrix} 7 & 6 \\ 1 & 1 \end{vmatrix} + (-3)$$

Sage code

```
(_).expand(term=1,row=2)
```

$$(-1)\begin{vmatrix} 4 & 1 \\ 5 & 1 \end{vmatrix} + \begin{vmatrix} 4 & 1 \\ 5 & 1 \end{vmatrix} + (-1)\begin{vmatrix} 2 & 1 \\ 7 & 1 \end{vmatrix} + (-8)\begin{vmatrix} 6 & 1 \\ 1 & 0 \end{vmatrix} + (12)\begin{vmatrix} 7 & 1 \\ 1 & 0 \end{vmatrix} + (-4)\begin{vmatrix} 7 & 6 \\ 1 & 1 \end{vmatrix} + (-3)$$

Sage code

```
BBBB=_.expand(term=3, column=1).expand(term=2,column=1).\
expand(term=1,column=1).expand(term=0,column=1)
BBBB
```

$$(5) + (-1)(4) + (-1)(5) + (4) + (7) + (-1)(2) + (8)(1) + (12)\begin{vmatrix} 7 & 1 \\ 1 & 0 \end{vmatrix} + (-4)\begin{vmatrix} 7 & 6 \\ 1 & 1 \end{vmatrix} + (-3)$$

Sage code

```
BBBB.collect()
```

$$(12)\begin{vmatrix} 7 & 1 \\ 1 & 0 \end{vmatrix} + (-4)\begin{vmatrix} 7 & 6 \\ 1 & 1 \end{vmatrix} + (10)$$

Sage code

```
_.expand(term=1,column=1).expand(term=0,column=1)
```

$$(-12)(1) + (24)(1) + (-4)(7) + (10)$$

Sage code

```
_.collect()
```

$$(-6)$$

Sage code

```
A.det()
```

$$-6$$