转载]矩阵求导公式【转】

http://simg.sinajs.cn/blog7style/images/common/sg_trans.gif (2011-11-15 11:03:34)

[[http://simg.sinajs.cn/blog7style/images/common/sg_trans.gif](javascript:;)转载▼](javascript:;)

|  |  |
| --- | --- |
| 标签：  [转载](http://search.sina.com.cn/?c=blog&q=%D7%AA%D4%D8&by=tag) |  |

**原文地址：**[矩阵求导公式【转】](http://blog.sina.com.cn/s/blog_6c17a3a00100qg5w.html)**作者：**[三寅](http://blog.sina.com.cn/u/1813488544)

今天推导公式，发现居然有对矩阵的求导，狂汗--完全不会。不过还好网上有人总结了。吼吼，赶紧搬过来收藏备份。

基本公式：  
Y = A \* X --> DY/DX = A'  
Y = X \* A --> DY/DX = A  
Y = A' \* X \* B --> DY/DX = A \* B'  
Y = A' \* X' \* B --> DY/DX = B \* A'

1. 矩阵Y对标量x求导：

相当于每个元素求导数后转置一下，注意M×N矩阵求导后变成N×M了

Y = [y(ij)] --> dY/dx = [dy(ji)/dx]

2. 标量y对列向量X求导：

注意与上面不同，这次括号内是求偏导，不转置，对N×1向量求导后还是N×1向量

y = f(x1,x2,..,xn) --> dy/dX = (Dy/Dx1,Dy/Dx2,..,Dy/Dxn)'

3. 行向量Y'对列向量X求导：

注意1×M向量对N×1向量求导后是N×M矩阵。

将Y的每一列对X求偏导，将各列构成一个矩阵。

重要结论：

dX'/dX = I

d(AX)'/dX = A'

4. 列向量Y对行向量X’求导：

转化为行向量Y’对列向量X的导数，然后转置。

注意M×1向量对1×N向量求导结果为M×N矩阵。

dY/dX' = (dY'/dX)'

5. 向量积对列向量X求导运算法则：

注意与标量求导有点不同。

d(UV')/dX = (dU/dX)V' + U(dV'/dX)

d(U'V)/dX = (dU'/dX)V + (dV'/dX)U'

重要结论：

d(X'A)/dX = (dX'/dX)A + (dA/dX)X' = IA + 0X' = A

d(AX)/dX' = (d(X'A')/dX)' = (A')' = A

d(X'AX)/dX = (dX'/dX)AX + (d(AX)'/dX)X = AX + A'X

6. 矩阵Y对列向量X求导：

将Y对X的每一个分量求偏导，构成一个超向量。

注意该向量的每一个元素都是一个矩阵。

7. 矩阵积对列向量求导法则：

d(uV)/dX = (du/dX)V + u(dV/dX)

d(UV)/dX = (dU/dX)V + U(dV/dX)

重要结论：

d(X'A)/dX = (dX'/dX)A + X'(dA/dX) = IA + X'0 = A

8. 标量y对矩阵X的导数：

类似标量y对列向量X的导数，

把y对每个X的元素求偏导，不用转置。

dy/dX = [ Dy/Dx(ij) ]

重要结论：

y = U'XV = ΣΣu(i)x(ij)v(j) 于是 dy/dX = [u(i)v(j)] = UV'

y = U'X'XU 则 dy/dX = 2XUU'

y = (XU-V)'(XU-V) 则 dy/dX = d(U'X'XU - 2V'XU + V'V)/dX = 2XUU' - 2VU' + 0 = 2(XU-V)U'

9. 矩阵Y对矩阵X的导数：

将Y的每个元素对X求导，然后排在一起形成超级矩阵。

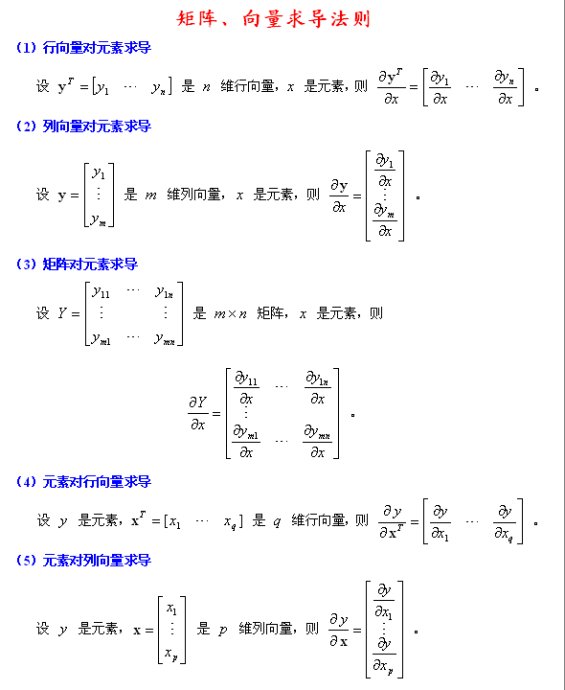
10.乘积的导数

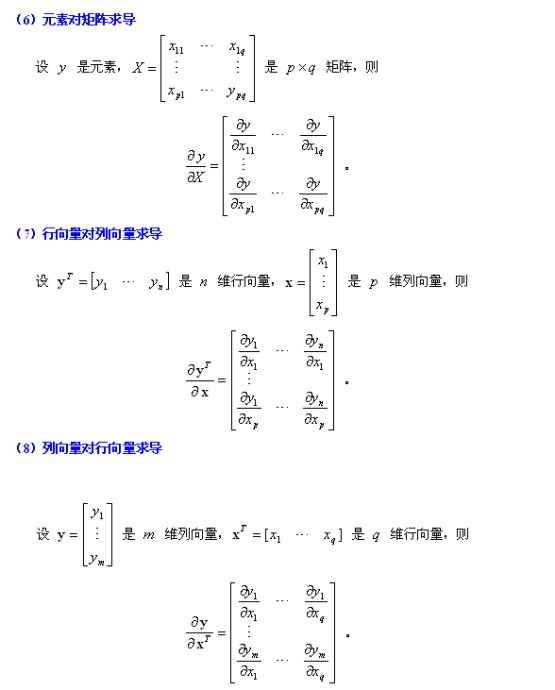
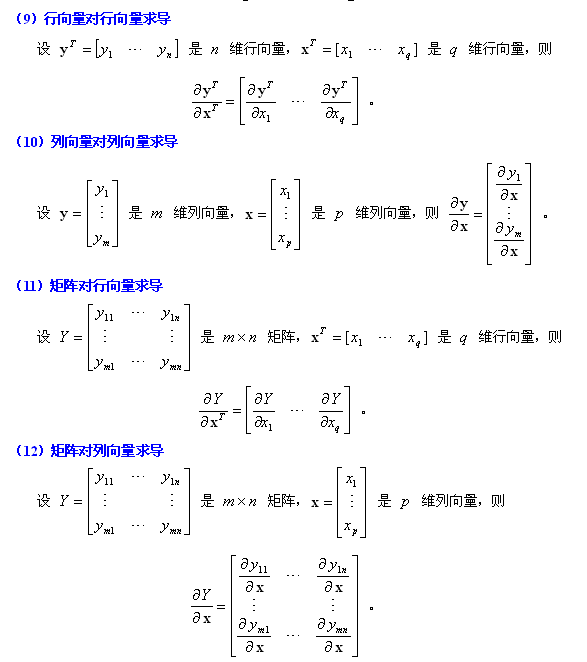
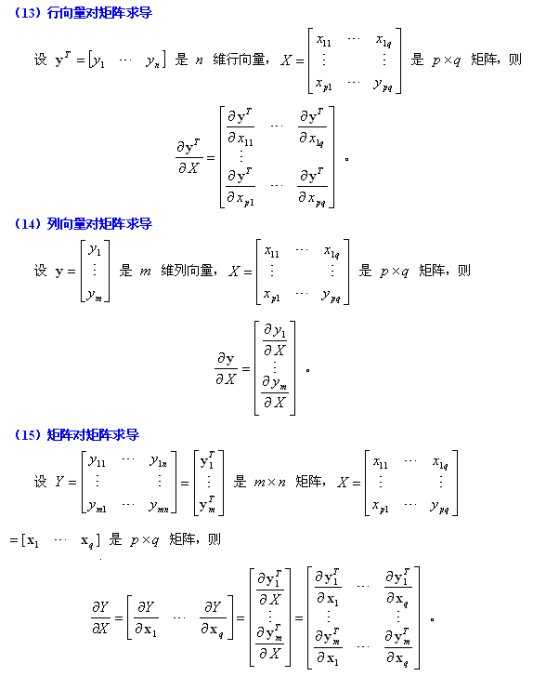
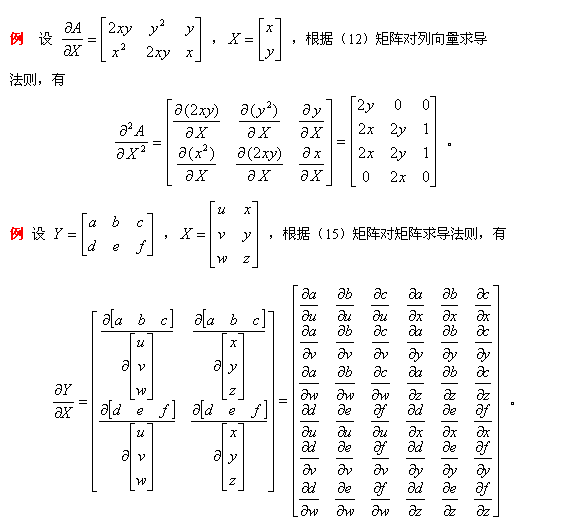
d(f\*g)/dx=(df'/dx)g+(dg/dx)f'

结论

d(x'Ax)=(d(x'')/dx)Ax+(d(Ax)/dx)(x'')=Ax+A'x   （注意：''是表示两次转置）

比较详细点的如下：

[](http://photo.blog.sina.com.cn/showpic.html#blogid=7959e7ed0100w2b3&url=http://s1.sinaimg.cn/orignal/6c17a3a0xa08fdfc73ea0)

[](http://photo.blog.sina.com.cn/showpic.html#blogid=7959e7ed0100w2b3&url=http://s15.sinaimg.cn/orignal/6c17a3a0xa08fdfd68bbe)  
  
[](http://photo.blog.sina.com.cn/showpic.html#blogid=7959e7ed0100w2b3&url=http://s15.sinaimg.cn/orignal/6c17a3a0xa08fdff509be)  
  
[](http://photo.blog.sina.com.cn/showpic.html#blogid=7959e7ed0100w2b3&url=http://s13.sinaimg.cn/orignal/6c17a3a0xa08fe0138efc)  
  
[](http://photo.blog.sina.com.cn/showpic.html#blogid=7959e7ed0100w2b3&url=http://s5.sinaimg.cn/orignal/6c17a3a0xa08fe022bf54)

http://lzh21cen.blog.163.com/blog/static/145880136201051113615571/

http://hi.baidu.com/wangwen926/blog/item/eb189bf6b0fb702b720eec94.html

其他参考：

**Contents**

* Notation
* Derivatives of Linear Products
* Derivatives of Quadratic Products

**Notation**

* *d/dx*(**y**)is a vector whose *(i)* element is *dy(i)/dx*
* *d/d***x** (y) is a vector whose *(i)* element is *dy/dx(i)*
* *d/d***x** (**y***T*) is a matrix whose *(i,j)* element is *dy(j)/dx(i)*
* *d/dx*(**Y**) is a matrix whose *(i,j)* element is *dy(i,j)/dx*
* *d/d***X** (y) is a matrix whose *(i,j)* element is *dy/dx(i,j)*

Note that the Hermitian transpose is not used because complex conjugates are not analytic.

In the expressions below matrices and vectors **A**, **B**, **C** do not depend on **X**.

**Derivatives of Linear Products**

* *d/dx*(**AYB**)*=***A** \**d/dx*(**Y**) \* **B**
  + *d/dx*(**Ay**)*=***A** \**d/dx*(**y**)
* *d/d***x**(**x***T***A**)*=***A**
  + *d/d***x**(**x***T*)*=***I**
  + *d/d***x**(**x***T***a**)*= d/d***x**(**a***T***x**) = **a**
* *d/d***X**(**a***T***Xb**) = **ab***T*
  + *d/d***X**(**a***T***Xa**) = *d/d***X**(**a***T***X***T***a**) = **aa***T*
* *d/d***X**(**a***T***X***T***b**) = **ba***T*
* *d/dx*(**YZ**)*=***Y** \**d/dx*(**Z**) + *d/dx*(**Y**)**\* Z**

**Derivatives of Quadratic Products**

* d/d**x** (**Ax**+**b**)*T***C**(**D**x+**e**) = **A***T***C(Dx+e)** + **D***T***C***T***(Ax+b)**
  + d/d**x** (**x***T***Cx**) = (**C**+**C***T*)**x**
    - [C: symmetric]: d/d**x** (**x***T***Cx**) = 2**Cx**
    - d/d**x** (**x***T***x**) = 2**x**
  + d/d**x** (**Ax**+**b**)*T* (**D**x+**e**) = **A***T***(Dx+e)** + **D***T***(Ax+b)**
    - d/d**x** (**Ax**+**b**)*T* (**A**x+**b**) = 2**A***T***(Ax+b)**
  + [C: symmetric]: d/d**x** (**Ax**+**b**)*T***C**(**A**x+**b**) = 2**A***T***C(Ax+b)**
* *d/d***X**(**a***T***X***T***Xb**) = **X(ab***T***+ ba***T***)**
  + *d/d***X**(**a***T***X***T***Xa**) = 2**Xaa***T*
* *d/d***X**(**a***T***X***T***CXb**) = **C***T***Xab***T***+ CXba***T*
  + *d/d***X**(**a***T***X***T***CXa**) = **(C + C***T***)Xaa***T*
  + [**C**:*Symmetric*] *d/d***X**(**a***T***X***T***CXa**) = **2CXaa***T*
* *d/d***X**((**Xa+b)***T***C(Xa+b)**) = (**C+C***T***)(Xa+b)a***T*

**Derivatives of Cubic Products**

* *d/d***x**(**x***T***Axx***T*) = (**A+A***T***)xx***T***+x***T***AxI**

**Derivatives of Inverses**

* *d/dx*(**Y**-1) = **-Y**-1*d/dx*(**Y**)**Y**-1

**Derivative of Trace**

Note: matrix dimensions must result in an *n\*n* argument for tr().

* *d/d***X**(tr**(X)**) = **I**
* *d/d***X**(tr**(X***k***)**) =*k***(X***k*-1**)***T*
* *d/d***X**(tr**(AX***k***)**) =**SUM***r=0:k*-1**(X***r***AX***k-r*-1**)***T*
* *d/d***X**(tr**(AX**-1**B)**) = **-(X**-1**BAX**-1**)***T*
  + *d/d***X**(tr**(AX**-1**)**) =*d/d***X**(tr**(X**-1**A)**) = **-X**-*T***A***T***X**-*T*
* *d/d***X**(tr**(A***T***XB***T*)) = *d/d***X**(tr**(BX***T***A**)) = **AB**
  + *d/d***X**(tr**(XA***T***)**) = *d/d***X**(**tr(A***T***X)**) =*d/d***X**(tr**(X***T***A)**) = *d/d***X**(**tr(AX***T***)**) **= A**
* *d/d***X**(tr**(AXBX***T***)**) =**A***T***XB***T* + **AXB**
  + *d/d***X**(tr**(XAX***T***)**) =**X(A+A***T***)**
  + *d/d***X**(tr**(X***T***AX)**) =**X***T***(A+A***T***)**
  + *d/d***X**(tr**(AX***T***X)**) =**(A+A***T***)X**
* *d/d***X**(tr**(AXBX)**) =**A***T***X***T***B***T* + **B***T***X***T***A***T*
* [C:*symmetric*] *d/d***X**(**tr((X***T***CX)**-1**A**) = *d/d***X**(**tr(A (X***T***CX)**-1) =**-(CX(X***T***CX)**-1**)(A+A***T***)(X***T***CX)**-1
* [B,C:*symmetric*] *d/d***X**(**tr((X***T***CX)**-1**(X***T***BX)**) = *d/d***X**(**tr( (X***T***BX)(X***T***CX)**-1) = **-2(CX(X***T***CX)**-1**)X***T***BX(X***T***CX)**-1**+ 2BX(X***T***CX)**-1

**Derivative of Determinant**

Note: matrix dimensions must result in an *n\*n* argument for det().

* *d/d***X**(det**(X)**) = *d/d***X**(det**(X***T***)**) = det**(X)\*X**-*T*
  + *d/d***X**(det**(AXB)**) = det**(AXB)\*X**-*T*
  + *d/d***X**(ln(det**(AXB))**) = **X**-*T*
* *d/d***X**(det**(X***k***)**) = *k*\*det**(X***k***)\*X**-*T*
  + *d/d***X**(ln(det**(X***k***)**)) = *k***X**-*T*
* [*Real*]*d/d***X**(det**(X***T***CX)**) = det**(X***T***CX)\*(C+C***T***)X(X***T***CX)**-1
  + [**C**: *Real,Symmetric*] *d/d***X**(det**(X***T***CX)**) = 2det**(X***T***CX)\* CX(X***T***CX)**-1
* [**C**: *Real,Symmetricc*] *d/d***X**(ln(det**(X***T***CX))**) = 2**CX(X***T***CX)**-1

**Jacobian**

If **y** is a function of **x**, then *d***y***T*/d**x** is the Jacobian matrix of **y** with respect to **x**.

Its determinant, |*d***y***T*/d**x**|, is the *Jacobian* of **y** with respect to **x** and represents the ratio of the hyper-volumes *d***y** and *d***x**. The Jacobian occurs when changing variables in an integration: Integral(f(**y**)d**y**)=Integral(f(**y**(**x**)) |*d***y***T*/*d***x**| d**x**).

**Hessian matrix**

If f is a function of **x** then the symmetric matrix d2f/d**x**2 = **d**/d**x***T*(df/d**x**) is the *Hessian* matrix of f(**x**). A value of **x** for which df/d**x** = **0** corresponds to a minimum, maximum or saddle point according to whether the Hessian is positive definite, negative definite or indefinite.

* d2/d**x**2 (**a***T***x**) = 0
* d2/d**x**2 (**Ax**+**b**)*T***C**(**D**x+**e**) = **A***T***CD** + **D***T***C***T***A**
  + d2/d**x**2 (**x***T***Cx**) = **C**+**C***T*
    - d2/d**x**2 (**x***T***x**) = 2**I**
  + d2/d**x**2 (**Ax**+**b**)*T* (**D**x+**e**) = **A***T***D** + **D***T***A**
    - d2/d**x**2 (**Ax**+**b**)*T* (**A**x+**b**) = 2**A***T***A**
  + [C: symmetric]: d2/d**x**2 (**Ax**+**b**)*T***C**(**A**x+**b**) = 2**A***T***CA**

http://www.psi.toronto.edu/matrix/calculus.html

<http://www.stanford.edu/~dattorro/matrixcalc.pdf>

<http://www.colorado.edu/engineering/CAS/courses.d/IFEM.d/IFEM.AppD.d/IFEM.AppD.pdf>

<http://www4.ncsu.edu/~pfackler/MatCalc.pdf>

<http://center.uvt.nl/staff/magnus/wip12.pdf>