ch1_phase4

October 2, 2020

[1]:
```
from sympy import Matrix, Symbol, derive_by_array, Lambda, Function,␣
↪MatrixSymbol, Derivative, symbols, diff
from sympy import var
from sympy.abc import x, i, j, a, b
```

[2]:
```
def myvar(letter: str, i: int, j: int) -> Symbol:
    letter_ij = Symbol('{}_{}{}'.format(letter, i+1, j+1),␣
↪is_commutative=True)
    return letter_ij


n,m,p = 3,3,2

X = Matrix(n, m, lambda i,j : myvar('x', i, j)); X
```

[2]:
$$\begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix}$$

[3]:
```
W = Matrix(m, p, lambda i,j : myvar('w', i, j)); W
```

[3]:
$$\begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{bmatrix}$$

[4]:
```
A = MatrixSymbol('X',3,3); Matrix(A)
B = MatrixSymbol('W',3,2)
```

[5]:
```
v = lambda a,b: a*b

vL = Lambda((a,b), a*b)
```

```
vL2 = Lambda((A,B), A*B)

n = Function('v') #, Lambda((a,b), a*b))

vN = lambda mat1, mat2: Matrix(mat1.shape[0], mat2.shape[1], lambda i, j:
 ↪ Symbol("n_{}{}".format(i+1, j+1))); vN


Nelem = vN(X, W)
Nelem
```

[5]:

$$\begin{bmatrix} n_{11} & n_{12} \\ n_{21} & n_{22} \\ n_{31} & n_{32} \end{bmatrix}$$

[6]:
```
Nspec = v(X,W)
Nspec
```

[6]:

$$\begin{bmatrix} w_{11}x_{11} + w_{21}x_{12} + w_{31}x_{13} & w_{12}x_{11} + w_{22}x_{12} + w_{32}x_{13} \\ w_{11}x_{21} + w_{21}x_{22} + w_{31}x_{23} & w_{12}x_{21} + w_{22}x_{22} + w_{32}x_{23} \\ w_{11}x_{31} + w_{21}x_{32} + w_{31}x_{33} & w_{12}x_{31} + w_{22}x_{32} + w_{32}x_{33} \end{bmatrix}$$

[7]:
```
#N = v(X, W); N
N = n(A,B)
N
```

[7]:

$$v(X, W)$$

[8]:
```
# way 2 of declaring S (better way)
sigma = Function('sigma')

sigmaApply = Function("sigma_apply") #lambda matrix:  matrix.
 ↪applyfunc(sigma)

sigmaApply_ = lambda matrix: matrix.applyfunc(sigma)


S = sigmaApply(N); S
```

[8]:

$$\sigma_{apply}(v(X, W))$$

[9]:
```python
Sspec = S.subs({A:X, B:W}).replace(n, v).replace(sigmaApply, sigmaApply_)
Sspec
```

[9]:

$$\begin{bmatrix}\sigma(w_{11}x_{11} + w_{21}x_{12} + w_{31}x_{13}) & \sigma(w_{12}x_{11} + w_{22}x_{12} + w_{32}x_{13})\\ \sigma(w_{11}x_{21} + w_{21}x_{22} + w_{31}x_{23}) & \sigma(w_{12}x_{21} + w_{22}x_{22} + w_{32}x_{23})\\ \sigma(w_{11}x_{31} + w_{21}x_{32} + w_{31}x_{33}) & \sigma(w_{12}x_{31} + w_{22}x_{32} + w_{32}x_{33})\end{bmatrix}$$

[10]:
```python
Selem = S.replace(n, vN).replace(sigmaApply, sigmaApply_)
Selem
```

[10]:

$$\begin{bmatrix}\sigma(n_{11}) & \sigma(n_{12})\\ \sigma(n_{21}) & \sigma(n_{22})\\ \sigma(n_{31}) & \sigma(n_{32})\end{bmatrix}$$

[11]:
```python
import itertools

elemToSpecD = dict(itertools.chain(*[[(Nelem[i, j], Nspec[i, j]) for j␣
 ↪in range(2)] for i in range(3)]))

elemToSpec = list(elemToSpecD.items())

Matrix(elemToSpec)
```

[11]:

$$\begin{bmatrix}n_{11} & w_{11}x_{11} + w_{21}x_{12} + w_{31}x_{13}\\ n_{12} & w_{12}x_{11} + w_{22}x_{12} + w_{32}x_{13}\\ n_{21} & w_{11}x_{21} + w_{21}x_{22} + w_{31}x_{23}\\ n_{22} & w_{12}x_{21} + w_{22}x_{22} + w_{32}x_{23}\\ n_{31} & w_{11}x_{31} + w_{21}x_{32} + w_{31}x_{33}\\ n_{32} & w_{12}x_{31} + w_{22}x_{32} + w_{32}x_{33}\end{bmatrix}$$

[12]:
```python
specToElemD = {v:k for k,v in elemToSpecD.items()}
specToElem = list(specToElemD.items())
Matrix(specToElem)
```

[12]:

$$\begin{bmatrix}w_{11}x_{11} + w_{21}x_{12} + w_{31}x_{13} & n_{11}\\ w_{12}x_{11} + w_{22}x_{12} + w_{32}x_{13} & n_{12}\\ w_{11}x_{21} + w_{21}x_{22} + w_{31}x_{23} & n_{21}\\ w_{12}x_{21} + w_{22}x_{22} + w_{32}x_{23} & n_{22}\\ w_{11}x_{31} + w_{21}x_{32} + w_{31}x_{33} & n_{31}\\ w_{12}x_{31} + w_{22}x_{32} + w_{32}x_{33} & n_{32}\end{bmatrix}$$

[13]:
```
elemToSpecFuncD = dict(itertools.chain(*[[(Nelem[i, j],␣
↪Function("n_{}{}".format(i + 1, j + 1))(Nspec[i, j])) for j in␣
↪range(2)] for i in range(3)]))

elemToSpecFunc = list(elemToSpecFuncD.items())

Matrix(elemToSpecFunc)
```

[13]:
$$\begin{bmatrix} n_{11} & \mathrm{n}_{11}\left(w_{11}x_{11} + w_{21}x_{12} + w_{31}x_{13}\right) \\ n_{12} & \mathrm{n}_{12}\left(w_{12}x_{11} + w_{22}x_{12} + w_{32}x_{13}\right) \\ n_{21} & \mathrm{n}_{21}\left(w_{11}x_{21} + w_{21}x_{22} + w_{31}x_{23}\right) \\ n_{22} & \mathrm{n}_{22}\left(w_{12}x_{21} + w_{22}x_{22} + w_{32}x_{23}\right) \\ n_{31} & \mathrm{n}_{31}\left(w_{11}x_{31} + w_{21}x_{32} + w_{31}x_{33}\right) \\ n_{32} & \mathrm{n}_{32}\left(w_{12}x_{31} + w_{22}x_{32} + w_{32}x_{33}\right) \end{bmatrix}$$

[14]:
```
elemToSpecFuncArgsD = dict(itertools.chain(*[[(Nelem[i, j],␣
↪Function("n_{}{}".format(i + 1, j + 1))(*X,*W)) for j in range(2)]␣
↪for i in range(3)]))

elemToSpecFuncArgs = list(elemToSpecFuncArgsD.items())

Matrix(elemToSpecFuncArgs)
```

[14]:
$$\begin{bmatrix} n_{11} & \mathrm{n}_{11}\left(x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}, w_{11}, w_{12}, w_{21}, w_{22}, w_{31}, w_{32}\right) \\ n_{12} & \mathrm{n}_{12}\left(x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}, w_{11}, w_{12}, w_{21}, w_{22}, w_{31}, w_{32}\right) \\ n_{21} & \mathrm{n}_{21}\left(x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}, w_{11}, w_{12}, w_{21}, w_{22}, w_{31}, w_{32}\right) \\ n_{22} & \mathrm{n}_{22}\left(x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}, w_{11}, w_{12}, w_{21}, w_{22}, w_{31}, w_{32}\right) \\ n_{31} & \mathrm{n}_{31}\left(x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}, w_{11}, w_{12}, w_{21}, w_{22}, w_{31}, w_{32}\right) \\ n_{32} & \mathrm{n}_{32}\left(x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}, w_{11}, w_{12}, w_{21}, w_{22}, w_{31}, w_{32}\right) \end{bmatrix}$$

[15]:
```
elemToMatArgD = dict(itertools.chain(*[[(Nelem[i, j], Function("n_{}{}".
↪format(i+1,j+1))(A,B) ) for j in range(2)] for i in range(3)]))

elemToMatArg = list(elemToMatArgD.items())

Matrix(elemToMatArg)
```

[15]:

$$\begin{bmatrix} n_{11} & \mathrm{n}_{11}(X, W) \\ n_{12} & \mathrm{n}_{12}(X, W) \\ n_{21} & \mathrm{n}_{21}(X, W) \\ n_{22} & \mathrm{n}_{22}(X, W) \\ n_{31} & \mathrm{n}_{31}(X, W) \\ n_{32} & \mathrm{n}_{32}(X, W) \end{bmatrix}$$

[16]:
```
matargToSpecD = dict(zip(elemToMatArgD.values(), elemToSpecD.values()))

matargToSpec = list(matargToSpecD.items())

Matrix(matargToSpec)
```

[16]:

$$\begin{bmatrix} \mathrm{n}_{11}(X, W) & w_{11}x_{11} + w_{21}x_{12} + w_{31}x_{13} \\ \mathrm{n}_{12}(X, W) & w_{12}x_{11} + w_{22}x_{12} + w_{32}x_{13} \\ \mathrm{n}_{21}(X, W) & w_{11}x_{21} + w_{21}x_{22} + w_{31}x_{23} \\ \mathrm{n}_{22}(X, W) & w_{12}x_{21} + w_{22}x_{22} + w_{32}x_{23} \\ \mathrm{n}_{31}(X, W) & w_{11}x_{31} + w_{21}x_{32} + w_{31}x_{33} \\ \mathrm{n}_{32}(X, W) & w_{12}x_{31} + w_{22}x_{32} + w_{32}x_{33} \end{bmatrix}$$

[17]:
```
Selem
```

[17]:

$$\begin{bmatrix} \sigma(n_{11}) & \sigma(n_{12}) \\ \sigma(n_{21}) & \sigma(n_{22}) \\ \sigma(n_{31}) & \sigma(n_{32}) \end{bmatrix}$$

[18]:
```
Sspec = Selem.subs(elemToSpecD)
Sspec
```

[18]:

$$\begin{bmatrix} \sigma(w_{11}x_{11} + w_{21}x_{12} + w_{31}x_{13}) & \sigma(w_{12}x_{11} + w_{22}x_{12} + w_{32}x_{13}) \\ \sigma(w_{11}x_{21} + w_{21}x_{22} + w_{31}x_{23}) & \sigma(w_{12}x_{21} + w_{22}x_{22} + w_{32}x_{23}) \\ \sigma(w_{11}x_{31} + w_{21}x_{32} + w_{31}x_{33}) & \sigma(w_{12}x_{31} + w_{22}x_{32} + w_{32}x_{33}) \end{bmatrix}$$

[19]:
```
# CAN even replace elements after have done an operation on them!!!␣
↪replacing n_21 * 2 with the number 4.
Sspec.subs({Nspec[0, 0]: 3}).replace(sigma, lambda x: 2 * x).
↪replace(Nspec[2, 1] * 2, 4)
```

[19]:

$$\begin{bmatrix} 6 & 2w_{12}x_{11} + 2w_{22}x_{12} + 2w_{32}x_{13} \\ 2w_{11}x_{21} + 2w_{21}x_{22} + 2w_{31}x_{23} & 2w_{12}x_{21} + 2w_{22}x_{22} + 2w_{32}x_{23} \\ 2w_{11}x_{31} + 2w_{21}x_{32} + 2w_{31}x_{33} & 4 \end{bmatrix}$$

[20]:
```python
lambd = Function("lambda")
lambd_ = lambda matrix : sum(matrix)

i, j = symbols('i j')
M = MatrixSymbol('M', i, j)# abstract shape
#sigmaApply_L = Lambda(M, M.applyfunc(sigma))

lambda_L = Lambda(M, sum(M))

n = Function("nu",applyfunc=True)

L = lambd(sigmaApply(n(A,B))); L
```

[20]:
$$\lambda(\sigma_{apply}(\nu(X, W)))$$

[21]:
```python
L.replace(n,v).replace(sigmaApply, sigmaApply_).diff(B)
```

[21]:
$$\frac{d}{d\xi_1}\lambda(\xi_1)\Big|_{\xi_1=(d\mapsto\sigma(d))_\circ(XW)} X^T\left(d\mapsto\frac{d}{dd}\sigma(d)\right)_\circ(XW)$$

[22]:
```python
L.replace(n,vN).replace(sigmaApply, sigmaApply_)
```

[22]:
$$\lambda\left(\begin{bmatrix}\sigma(n_{11}) & \sigma(n_{12})\\ \sigma(n_{21}) & \sigma(n_{22})\\ \sigma(n_{31}) & \sigma(n_{32})\end{bmatrix}\right)$$

[23]:
```python
L.replace(n, vN).replace(sigmaApply, sigmaApply_).replace(lambd, lambd_)
```

[23]:
$$\sigma(n_{11})+\sigma(n_{12})+\sigma(n_{21})+\sigma(n_{22})+\sigma(n_{31})+\sigma(n_{32})$$

[24]:
```python
L.replace(n, vN).replace(sigmaApply, sigmaApply_).replace(lambd, lambd_).
↪subs(elemToSpecD)
```

[24]:
$$\sigma(w_{11}x_{11}+w_{21}x_{12}+w_{31}x_{13})+\sigma(w_{11}x_{21}+w_{21}x_{22}+w_{31}x_{23})+\sigma(w_{11}x_{31}+w_{21}x_{32}+w_{31}x_{33})+\sigma(w_{12}x_{11}+w_{22}x_{12}+w_{32}x_{13})+\sigma(w_{12}x_{21}+w_{22}x_{22}+w_{32}x_{23})+\sigma(w_{12}x_{31}+w_{22}x_{32}+w_{32}x_{33})$$

[25]:
```python
L.replace(n, vN).replace(sigmaApply, sigmaApply_).replace(lambd, lambd_).
↪subs(elemToSpecD).diff(W).subs(specToElemD)
```

[25]:

$$\begin{bmatrix} x_{11}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{11}}+x_{21}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{21}}+x_{31}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{31}} & x_{11}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{12}}+x_{21}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{22}}+x_{31}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{32}} \\ x_{12}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{11}}+x_{22}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{21}}+x_{32}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{31}} & x_{12}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{12}}+x_{22}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{22}}+x_{32}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{32}} \\ x_{13}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{11}}+x_{23}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{21}}+x_{33}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{31}} & x_{13}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{12}}+x_{23}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{22}}+x_{33}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{32}} \end{bmatrix}$$

[26]:
```
# Now verifying the above rule by applying the composition thing piece␣
↪by piece via multiplication:
L.replace(n,vL).replace(sigmaApply, sigmaApply_).diff(B)
```

[26]:

$$\left.\frac{d}{d\xi_1}\lambda(\xi_1)\right|_{\xi_1=(d\mapsto\sigma(d))_{\circ}(XW)} X^T\left(d\mapsto\frac{d}{dd}\sigma(d)\right)_{\circ}(XW)$$

[27]:
```
L.replace(n,vL).replace(sigmaApply,sigmaApply_)#.diff()
```

[27]:

$$\lambda((d\mapsto\sigma(d))_{\circ}(XW))$$

[28]:
```
L.replace(n,vL)
```

[28]:

$$\lambda(\sigma_{apply}(XW))$$

[29]:
```
L.replace(n,vL).diff(sigmaApply(A*B))
```

[29]:

$$\frac{\partial}{\partial\sigma_{apply}(XW)}\lambda(\sigma_{apply}(XW))$$

[30]:
```
#L.replace(n,vL).diff(sigmaApply(A*B)).replace(sigmaApply,sigmaApply_)
## ERROR cannot do that
```

[31]:
```
sigmaApply_L = Lambda(M, M.applyfunc(sigma))
#L.replace(n,vL).diff(sigmaApply(A*B)).subs(sigmaApply,sigmaApply_L) ##␣
↪ERROR

nL = Lambda((A,B), n(A,B)); nL
```

[31]:

$$((X,\ W)\mapsto\nu(X,W))$$

[32]:
```
# Try to create function cpmpositions :
from functools import reduce

def compose2(f, g):
    return lambda *a, **kw: f(g(*a, **kw))
```

```python
def compose(*fs):
    return reduce(compose2, fs)
```

[33]:
```python
f, g, h = symbols("f g h ", cls=Function)
diff(f(g(h(x))), x)
```

[33]: $\frac{d}{dg(h(x))}f(g(h(x)))\frac{d}{dh(x)}g(h(x))\frac{d}{dx}h(x)$

[34]:
```python
compose(f,g,h)(x)
```

[34]: $f(g(h(x)))$

[35]:
```python
compose(f,g,h)
```

[35]:
```
<function __main__.compose2.<locals>.<lambda>(*a, **kw)>
```

[36]:
```python
compose(lambd, sigmaApply, n)(A,B)
```

[36]: $\lambda(\sigma_{apply}(\nu(X, W)))$

[37]:
```python
#diff(compose(lambd, sigmaApply, n)(A.T,B), A)


#compose(lambd, sigmaApply, n)(A,B).diff(lambd)
diff(compose(lambd, sigmaApply, n)(A,B), compose(lambd, sigmaApply,␣
↪n)(A,B))
```

[37]: 1

[38]:
```python
d = diff(compose(lambd, sigmaApply, n)(A,B), n(A,B)); d
```

[38]: $\frac{\partial}{\partial\sigma_{apply}(\nu(X, W))}\lambda(\sigma_{apply}(\nu(X, W)))\frac{\partial}{\partial\nu(X, W)}\sigma_{apply}(\nu(X, W))$

[39]:
```python
#compose(lambd, sigmaApply, n)(A,B).diff(A) # ERROR
Lc = compose(lambd, sigmaApply, n)(A, B)
Lc.replace(n, vL).replace(sigmaApply, sigmaApply_)
```

[39]: $\lambda((d \mapsto \sigma(d))_{\circ}(XW))$

[40]:
```
# Same result as replacing in L
#Lc.replace(n, vL).replace(sigmaApply, sigmaApply_).diff(B)
Lc.replace(n, v).replace(sigmaApply, sigmaApply_).diff(B)
```

[40]:

$$\frac{d}{d\xi_1}\lambda(\xi_1)\Big|_{\xi_1=(d\mapsto\sigma(d))_\circ(XW)} X^T\left(d\mapsto\frac{d}{dd}\sigma(d)\right)_\circ(XW)$$

[41]:
```
funcToMat = lambda func: Matrix([func])
funcToMat(f)
A.applyfunc(sigma)
#funcToMat(f).applyfunc(sigma)
from sympy import FunctionMatrix
F = MatrixSymbol("F",3,3)#(FunctionMatrix(3,3, f))
FL = Lambda(F, n(A,B))
FL
gL = lambda A: A.applyfunc(sigma)
gL(A)
temp = lambda n : n
temp(n(A,B))
#sigmaApply_L(A).subs(A, Lambda((A,B), vL(A,B)))# arg must be matrix
 ↪instance
```

[41]:

$\nu(X,W)$

[42]:
```
sigmaApply_L(A*B).diff(B)
```

[42]:

$$X^T\left(d\mapsto\frac{d}{dd}\sigma(d)\right)_\circ(XW)$$

[43]:
```
sigmaApply_L(A).diff(A).subs(A,X).doit()
```

[43]:

$$\begin{bmatrix}\frac{d}{dx_{11}}\sigma(x_{11}) & \frac{d}{dx_{12}}\sigma(x_{12}) & \frac{d}{dx_{13}}\sigma(x_{13})\\ \frac{d}{dx_{21}}\sigma(x_{21}) & \frac{d}{dx_{22}}\sigma(x_{22}) & \frac{d}{dx_{23}}\sigma(x_{23})\\ \frac{d}{dx_{31}}\sigma(x_{31}) & \frac{d}{dx_{32}}\sigma(x_{32}) & \frac{d}{dx_{33}}\sigma(x_{33})\end{bmatrix}$$

[44]:
```

```

[44]:
```
sigmaApply_L(A*B).diff(B).subs(A,X).subs(B,W)#.doit()
### CANNOT go farther here because of noncommutative scalars in matrix
 ↪error
```

[44]:

$$\left(\begin{bmatrix}x_{11} & x_{12} & x_{13}\\x_{21} & x_{22} & x_{23}\\x_{31} & x_{32} & x_{33}\end{bmatrix}\right)^T \left(d \mapsto \frac{d}{dd}\sigma(d)\right)_\circ \left(\begin{bmatrix}x_{11} & x_{12} & x_{13}\\x_{21} & x_{22} & x_{23}\\x_{31} & x_{32} & x_{33}\end{bmatrix}\begin{bmatrix}w_{11} & w_{12}\\w_{21} & w_{22}\\w_{31} & w_{32}\end{bmatrix}\right)$$

[45]:
```
#sigmaApply_L(X*W).subs(specToElemD).subs(elemToMatArgD).diff(B)#
```

[46]:
```
sigmaApply_L(A*B).diff(B)#.replace(A,X).replace(B,W).replace(X*W,Nelem)
```

[46]:

$$X^T \left(d \mapsto \frac{d}{dd}\sigma(d)\right)_\circ (XW)$$

[47]:
```
sigmaApply_L(A*B).diff(B).subs({A*B : vN(A,B)})
```

[47]:

$$X^T \left(d \mapsto \frac{d}{dd}\sigma(d)\right)_\circ \left(\begin{bmatrix}n_{11} & n_{12}\\n_{21} & n_{22}\\n_{31} & n_{32}\end{bmatrix}\right)$$

[48]:
```
sigmaApply_L(A*B).diff(B).subs({A*B : vN(A,B)}).doit()
```

[48]:

$$X^T \begin{bmatrix}\frac{d}{dn_{11}}\sigma(n_{11}) & \frac{d}{dn_{12}}\sigma(n_{12})\\\frac{d}{dn_{21}}\sigma(n_{21}) & \frac{d}{dn_{22}}\sigma(n_{22})\\\frac{d}{dn_{31}}\sigma(n_{31}) & \frac{d}{dn_{32}}\sigma(n_{32})\end{bmatrix}$$

[49]:
```
sigmaApply_L(A*B).diff(B).subs({A*B : vN(A,B)}).subs(elemToMatArgD)
```

[49]:

$$X^T \left(d \mapsto \frac{d}{dd}\sigma(d)\right)_\circ \left(\begin{bmatrix}\mathrm{n}_{11}(X,W) & \mathrm{n}_{12}(X,W)\\\mathrm{n}_{21}(X,W) & \mathrm{n}_{22}(X,W)\\\mathrm{n}_{31}(X,W) & \mathrm{n}_{32}(X,W)\end{bmatrix}\right)$$

[50]:
```
sigmaApply_L(A*B).diff(B).subs({A*B : vN(A,B)}).subs(elemToMatArgD).
↪doit()
```

[50]:

$$X^T \begin{bmatrix}\frac{\partial}{\partial \mathrm{n}_{11}(X,W)}\sigma(\mathrm{n}_{11}(X,W)) & \frac{\partial}{\partial \mathrm{n}_{12}(X,W)}\sigma(\mathrm{n}_{12}(X,W))\\\frac{\partial}{\partial \mathrm{n}_{21}(X,W)}\sigma(\mathrm{n}_{21}(X,W)) & \frac{\partial}{\partial \mathrm{n}_{22}(X,W)}\sigma(\mathrm{n}_{22}(X,W))\\\frac{\partial}{\partial \mathrm{n}_{31}(X,W)}\sigma(\mathrm{n}_{31}(X,W)) & \frac{\partial}{\partial \mathrm{n}_{32}(X,W)}\sigma(\mathrm{n}_{32}(X,W))\end{bmatrix}$$

[51]:
```
sigmaApply_L(A*B).diff(B).subs({A*B : vN(A,B)}).doit()
```

[51]:

$$X^T \begin{bmatrix}\frac{d}{dn_{11}}\sigma(n_{11}) & \frac{d}{dn_{12}}\sigma(n_{12})\\\frac{d}{dn_{21}}\sigma(n_{21}) & \frac{d}{dn_{22}}\sigma(n_{22})\\\frac{d}{dn_{31}}\sigma(n_{31}) & \frac{d}{dn_{32}}\sigma(n_{32})\end{bmatrix}$$

[52]:
```
#part1=sigmaApply_L(A*B).diff(B).subs({A*B : vN(A,B)}).doit()
```

[53]:
```
part1 = compose(sigmaApply, n)(A,B).replace(n, v).replace(sigmaApply,␣
↪sigmaApply_).diff(B).subs({A*B : vN(A,B)}).doit()


part1.subs({A:X}) # replace won't work here
```

[53]:

$$\left(\begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix}\right)^T \begin{bmatrix} \frac{d}{dn_{11}}\sigma(n_{11}) & \frac{d}{dn_{12}}\sigma(n_{12}) \\ \frac{d}{dn_{21}}\sigma(n_{21}) & \frac{d}{dn_{22}}\sigma(n_{22}) \\ \frac{d}{dn_{31}}\sigma(n_{31}) & \frac{d}{dn_{32}}\sigma(n_{32}) \end{bmatrix}$$

[54]:
```
part1.subs({A:X}).doit()
```

[54]:

$$\begin{bmatrix} x_{11}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{21}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{31}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{11}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{21}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{31}\frac{d}{dn_{32}}\sigma(n_{32}) \\ x_{12}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{22}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{32}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{12}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{22}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{32}\frac{d}{dn_{32}}\sigma(n_{32}) \\ x_{13}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{23}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{33}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{13}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{23}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{33}\frac{d}{dn_{32}}\sigma(n_{32}) \end{bmatrix}$$

0.0.1 COMPARE: Symbolic form vs. Direct form vs. Step by Step form (which goes from symbolic to direct form by replacing)

Symbolic Abstract Form (with respect to W):

[55]:
```
Lc = compose(lambd, sigmaApply, n)(A, B)
symb = Lc.replace(n, v).replace(sigmaApply, sigmaApply_).diff(B)
symb
```

[55]:

$$\frac{d}{d\xi_1}\lambda(\xi_1)\Big|_{\xi_1=(d\mapsto\sigma(d))_{\circ}(XW)} X^T\left(d\mapsto\frac{d}{dd}\sigma(d)\right)_{\circ}(XW)$$

Direct form: (after the symbolic form)

[56]:
```
Lc.replace(n, vN).replace(sigmaApply, sigmaApply_).replace(lambd,␣
↪lambd_).subs(elemToSpecD).diff(W).subs(specToElemD)
```

[56]:

$$\begin{bmatrix} x_{11}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{11}} + x_{21}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{21}} + x_{31}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{31}} & x_{11}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{12}} + x_{21}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{22}} + x_{31}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{32}} \\ x_{12}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{11}} + x_{22}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{21}} + x_{32}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{31}} & x_{12}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{12}} + x_{22}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{22}} + x_{32}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{32}} \\ x_{13}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{11}} + x_{23}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{21}} + x_{33}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{31}} & x_{13}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{12}} + x_{23}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{22}} + x_{33}\left.\frac{d}{d\xi_1}\sigma(\xi_1)\right|_{\xi_1=n_{32}} \end{bmatrix}$$

```
Just placing the ``n'' values right in place of the ``epsilons'' using the
``doit'' function:
```

```
[57]: direct = Lc.replace(n, vN).replace(sigmaApply, sigmaApply_).
       ↪replace(lambd, lambd_).subs(elemToSpecD).diff(W).subs(specToElemD).
       ↪doit()
      direct
```

[57]:

$$\left[\begin{matrix}x_{11}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{21}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{31}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{11}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{21}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{31}\frac{d}{dn_{32}}\sigma(n_{32})\\x_{12}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{22}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{32}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{12}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{22}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{32}\frac{d}{dn_{32}}\sigma(n_{32})\\x_{13}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{23}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{33}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{13}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{23}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{33}\frac{d}{dn_{32}}\sigma(n_{32})\end{matrix}\right]$$

Step by Step Form:

```
[58]: assert symb == Lc.replace(n, v).replace(sigmaApply, sigmaApply_).diff(B)

      symb.subs({A*B : vN(A,B)})#.doit()
```

[58]:

$$\left.\frac{d}{d\xi_1}\lambda(\xi_1)\right|_{\xi_1=(d\mapsto\sigma(d))\circ\left(\left[\begin{matrix}n_{11} & n_{12}\\n_{21} & n_{22}\\n_{31} & n_{32}\end{matrix}\right]\right)} X^T\left(d\mapsto\frac{d}{dd}\sigma(d)\right)\circ\left(\left[\begin{matrix}n_{11} & n_{12}\\n_{21} & n_{22}\\n_{31} & n_{32}\end{matrix}\right]\right)$$

```
[59]: symb.subs({A*B : vN(A,B)}).doit() # the dummy variable for lambda still␣
       ↪stays unapplied
```

[59]:

$$\left.\frac{d}{d\xi_1}\lambda(\xi_1)\right|_{\xi_1=(d\mapsto\sigma(d))\circ\left(\left[\begin{matrix}n_{11} & n_{12}\\n_{21} & n_{22}\\n_{31} & n_{32}\end{matrix}\right]\right)} X^T\left[\begin{matrix}\frac{d}{dn_{11}}\sigma(n_{11}) & \frac{d}{dn_{12}}\sigma(n_{12})\\\frac{d}{dn_{21}}\sigma(n_{21}) & \frac{d}{dn_{22}}\sigma(n_{22})\\\frac{d}{dn_{31}}\sigma(n_{31}) & \frac{d}{dn_{32}}\sigma(n_{32})\end{matrix}\right]$$

```
[60]: symb.subs({A*B : vN(A,B)}).subs({A:X}).doit() # two doits are equivalent␣
       ↪to the last one at the end
```

[60]:

$$\left.\frac{d}{d\xi_1}\lambda(\xi_1)\right|_{\xi_1=(d\mapsto\sigma(d))\circ\left(\left[\begin{matrix}n_{11} & n_{12}\\n_{21} & n_{22}\\n_{31} & n_{32}\end{matrix}\right]\right)} \left[\begin{matrix}x_{11}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{21}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{31}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{11}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{21}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{31}\frac{d}{dn_{32}}\sigma(n_{32})\\x_{12}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{22}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{32}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{12}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{22}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{32}\frac{d}{dn_{32}}\sigma(n_{32})\\x_{13}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{23}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{33}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{13}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{23}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{33}\frac{d}{dn_{32}}\sigma(n_{32})\end{matrix}\right]$$

```
[61]: # Creating a special symbol just for the lambda_L function that has the␣
       ↪appropriate shape after multiplying A*B. Supposed to represent a␣
       ↪matrix R s.t. R == A * B (so that the indices after applying lambda_L␣
       ↪are correct)
```

```python
ABres = MatrixSymbol("R", A.shape[0], B.shape[1])
lambd_L = Lambda(ABres, sum(ABres))


symb.subs({A*B : vN(A,B)}).subs({A:X}).doit()#subs({lambd: lambd_L}).
 ↪doit()
```

[61]:

$$\frac{d}{d\xi_1}\lambda(\xi_1)\Bigg|_{\xi_1=(d\mapsto\sigma(d))\circ\left(\begin{bmatrix}n_{11} & n_{12}\\ n_{21} & n_{22}\\ n_{31} & n_{32}\end{bmatrix}\right)}\begin{bmatrix}x_{11}\frac{d}{dn_{11}}\sigma(n_{11})+x_{21}\frac{d}{dn_{21}}\sigma(n_{21})+x_{31}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{11}\frac{d}{dn_{12}}\sigma(n_{12})+x_{21}\frac{d}{dn_{22}}\sigma(n_{22})+x_{31}\frac{d}{dn_{32}}\sigma(n_{32})\\ x_{12}\frac{d}{dn_{11}}\sigma(n_{11})+x_{22}\frac{d}{dn_{21}}\sigma(n_{21})+x_{32}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{12}\frac{d}{dn_{12}}\sigma(n_{12})+x_{22}\frac{d}{dn_{22}}\sigma(n_{22})+x_{32}\frac{d}{dn_{32}}\sigma(n_{32})\\ x_{13}\frac{d}{dn_{11}}\sigma(n_{11})+x_{23}\frac{d}{dn_{21}}\sigma(n_{21})+x_{33}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{13}\frac{d}{dn_{12}}\sigma(n_{12})+x_{23}\frac{d}{dn_{22}}\sigma(n_{22})+x_{33}\frac{d}{dn_{32}}\sigma(n_{32})\end{bmatrix}$$

[62]:

```python
# yay finall got the dummy variable in the lambd to be applied!!
unapplied = sigmaApply_L(vN(A,B))
applied = unapplied.doit()

symb.subs({A*B : vN(A,B)}).subs({A:X}).doit().replace(unapplied, applied)
```

[62]:

$$\frac{d}{d\xi_1}\lambda(\xi_1)\Bigg|_{\xi_1=\begin{bmatrix}\sigma(n_{11}) & \sigma(n_{12})\\ \sigma(n_{21}) & \sigma(n_{22})\\ \sigma(n_{31}) & \sigma(n_{32})\end{bmatrix}}\begin{bmatrix}x_{11}\frac{d}{dn_{11}}\sigma(n_{11})+x_{21}\frac{d}{dn_{21}}\sigma(n_{21})+x_{31}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{11}\frac{d}{dn_{12}}\sigma(n_{12})+x_{21}\frac{d}{dn_{22}}\sigma(n_{22})+x_{31}\frac{d}{dn_{32}}\sigma(n_{32})\\ x_{12}\frac{d}{dn_{11}}\sigma(n_{11})+x_{22}\frac{d}{dn_{21}}\sigma(n_{21})+x_{32}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{12}\frac{d}{dn_{12}}\sigma(n_{12})+x_{22}\frac{d}{dn_{22}}\sigma(n_{22})+x_{32}\frac{d}{dn_{32}}\sigma(n_{32})\\ x_{13}\frac{d}{dn_{11}}\sigma(n_{11})+x_{23}\frac{d}{dn_{21}}\sigma(n_{21})+x_{33}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{13}\frac{d}{dn_{12}}\sigma(n_{12})+x_{23}\frac{d}{dn_{22}}\sigma(n_{22})+x_{33}\frac{d}{dn_{32}}\sigma(n_{32})\end{bmatrix}$$

[63]:

```python
# THIS SEEMS WRONG : ??? how to tell for sure?
lambd(Selem).diff(Selem).replace(lambd, lambd_L).doit()
```

[63]:

$$\left[\begin{matrix}\left[\begin{matrix}\left[\begin{matrix}1 & 1\\1 & 1\\1 & 1\end{matrix}\right] & 0\\0 & 0\\0 & 0\end{matrix}\right] & \left[\begin{matrix}0 & \left[\begin{matrix}1 & 1\\1 & 1\\1 & 1\end{matrix}\right]\\0 & 0\\0 & 0\end{matrix}\right]\\\left[\begin{matrix}0 & 0\\\left[\begin{matrix}1 & 1\\1 & 1\\1 & 1\end{matrix}\right] & 0\\0 & 0\end{matrix}\right] & \left[\begin{matrix}0 & 0\\0 & \left[\begin{matrix}1 & 1\\1 & 1\\1 & 1\end{matrix}\right]\\0 & 0\end{matrix}\right]\\\left[\begin{matrix}0 & 0\\0 & 0\\\left[\begin{matrix}1 & 1\\1 & 1\\1 & 1\end{matrix}\right] & 0\end{matrix}\right] & \left[\begin{matrix}0 & 0\\0 & 0\\0 & \left[\begin{matrix}1 & 1\\1 & 1\\1 & 1\end{matrix}\right]\end{matrix}\right]\end{matrix}\right]$$

[64]:
```
# THis seems right:
dL_dS = lambd(Selem).replace(lambd, lambd_L).diff(Selem)
dL_dS
```

[64]:

$$\left[\begin{matrix}1 & 1\\1 & 1\\1 & 1\end{matrix}\right]$$

[65]:
```
lambd(Selem)
```

[65]:

$$\lambda\left(\left[\begin{matrix}\sigma(n_{11}) & \sigma(n_{12})\\\sigma(n_{21}) & \sigma(n_{22})\\\sigma(n_{31}) & \sigma(n_{32})\end{matrix}\right]\right)$$

[66]:
```
# This is the X^T * dS/dN part
compose(sigmaApply, n)(A,B).replace(n, v).replace(sigmaApply,␣
↪sigmaApply_).diff(B).subs({A*B : vN(A,B)}).doit()
```

[66]:

$$X^T\left[\begin{matrix}\frac{d}{dn_{11}}\sigma(n_{11}) & \frac{d}{dn_{12}}\sigma(n_{12})\\\frac{d}{dn_{21}}\sigma(n_{21}) & \frac{d}{dn_{22}}\sigma(n_{22})\\\frac{d}{dn_{31}}\sigma(n_{31}) & \frac{d}{dn_{32}}\sigma(n_{32})\end{matrix}\right]$$

[67]:
```
N = MatrixSymbol("N", A.shape[0], B.shape[1])
Matrix(N)
```

[67]:

$$\begin{bmatrix} N_{0,0} & N_{0,1} \\ N_{1,0} & N_{1,1} \\ N_{2,0} & N_{2,1} \end{bmatrix}$$

[68]:
```
dS_dN = compose(sigmaApply)(N).replace(sigmaApply, sigmaApply_).diff(N).
↪subs({N : vN(A,B)}).doit()
# WRONG:
#dS_dN = sigmaApply(Nelem).replace(sigmaApply, sigmaApply_).
↪diff(Matrix(Nelem))
dS_dN
```

[68]:

$$\begin{bmatrix} \frac{d}{dn_{11}}\sigma(n_{11}) & \frac{d}{dn_{12}}\sigma(n_{12}) \\ \frac{d}{dn_{21}}\sigma(n_{21}) & \frac{d}{dn_{22}}\sigma(n_{22}) \\ \frac{d}{dn_{31}}\sigma(n_{31}) & \frac{d}{dn_{32}}\sigma(n_{32}) \end{bmatrix}$$

[69]:
```
from sympy.physics.quantum import TensorProduct

#TensorProduct( X.T, dS_dN)


dN_dW = X.T
dS_dW = dN_dW * dS_dN
dS_dW
#HadamardProduct(X.T, dS_dN)
```

[69]:

$$\begin{bmatrix} x_{11}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{21}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{31}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{11}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{21}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{31}\frac{d}{dn_{32}}\sigma(n_{32}) \\ x_{12}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{22}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{32}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{12}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{22}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{32}\frac{d}{dn_{32}}\sigma(n_{32}) \\ x_{13}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{23}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{33}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{13}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{23}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{33}\frac{d}{dn_{32}}\sigma(n_{32}) \end{bmatrix}$$

[70]:
```
from sympy import HadamardProduct

dL_dW = HadamardProduct(dS_dW , dL_dS)
dL_dW
```

[70]:

$$\begin{bmatrix} x_{11}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{21}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{31}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{11}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{21}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{31}\frac{d}{dn_{32}}\sigma(n_{32}) \\ x_{12}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{22}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{32}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{12}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{22}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{32}\frac{d}{dn_{32}}\sigma(n_{32}) \\ x_{13}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{23}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{33}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{13}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{23}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{33}\frac{d}{dn_{32}}\sigma(n_{32}) \end{bmatrix} \circ \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

One more time as complete symbolic form:

$$\begin{aligned}\frac{\partial L}{\partial W} &= \frac{\partial N}{\partial W} \times \frac{\partial S}{\partial N} \odot \frac{\partial L}{\partial S} \\ &= X^T \times \frac{\partial S}{\partial N} \odot \frac{\partial L}{\partial S}\end{aligned}$$

where $\odot$ signifies the Hadamard product and $\times$ is matrix multiplication.

[71]:
```
HadamardProduct(dN_dW * dS_dN, dL_dS)
```

[71]:

$$\begin{bmatrix} x_{11}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{21}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{31}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{11}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{21}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{31}\frac{d}{dn_{32}}\sigma(n_{32}) \\ x_{12}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{22}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{32}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{12}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{22}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{32}\frac{d}{dn_{32}}\sigma(n_{32}) \\ x_{13}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{23}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{33}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{13}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{23}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{33}\frac{d}{dn_{32}}\sigma(n_{32}) \end{bmatrix} \circ \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

[72]:
```
direct
```

[72]:

$$\begin{bmatrix} x_{11}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{21}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{31}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{11}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{21}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{31}\frac{d}{dn_{32}}\sigma(n_{32}) \\ x_{12}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{22}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{32}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{12}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{22}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{32}\frac{d}{dn_{32}}\sigma(n_{32}) \\ x_{13}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{23}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{33}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{13}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{23}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{33}\frac{d}{dn_{32}}\sigma(n_{32}) \end{bmatrix}$$

[73]:
```
assert HadamardProduct(dN_dW * dS_dN, dL_dS).equals(direct)
```

[74]:
```
# too long to see in editor:
symb.subs({A*B : vN(A,B)}).subs({A:X}).doit().replace(lambd, lambd_L)
```

[74]:

$$\left.\frac{d}{d\xi_1}\left(\xi_{1_{0,0}} + \xi_{1_{0,1}} + \xi_{1_{1,0}} + \xi_{1_{1,1}} + \xi_{1_{2,0}} + \xi_{1_{2,1}}\right)\right|_{\xi_1=(d\mapsto\sigma(d))_\circ\left(\begin{bmatrix} n_{11} & n_{12} \\ n_{21} & n_{22} \\ n_{31} & n_{32} \end{bmatrix}\right)} \begin{bmatrix} x_{11}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{21}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{31}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{11}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{21}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{31}\frac{d}{dn_{32}}\sigma(n_{32}) \\ x_{12}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{22}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{32}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{12}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{22}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{32}\frac{d}{dn_{32}}\sigma(n_{32}) \\ x_{13}\frac{d}{dn_{11}}\sigma(n_{11}) + x_{23}\frac{d}{dn_{21}}\sigma(n_{21}) + x_{33}\frac{d}{dn_{31}}\sigma(n_{31}) & x_{13}\frac{d}{dn_{12}}\sigma(n_{12}) + x_{23}\frac{d}{dn_{22}}\sigma(n_{22}) + x_{33}\frac{d}{dn_{32}}\sigma(n_{32}) \end{bmatrix}$$

[75]:
```
symb.subs({lambd: lambd_L})
```

[75]:

$$\left.\frac{d}{d\xi_1}\left(\xi_{1_{0,0}} + \xi_{1_{0,1}} + \xi_{1_{1,0}} + \xi_{1_{1,1}} + \xi_{1_{2,0}} + \xi_{1_{2,1}}\right)\right|_{\xi_1=(d\mapsto\sigma(d))_\circ(XW)} X^T\left(d \mapsto \frac{d}{dd}\sigma(d)\right)_\circ(XW)$$

[76]:
```
print(symb.subs({lambd: lambd_L}))
```

```
Subs(Derivative(_xi_1[0, 0] + _xi_1[0, 1] + _xi_1[1, 0] + _xi_1[1, 1] +␣
 ↪_xi_1[2,
0] + _xi_1[2, 1], _xi_1), _xi_1, Lambda(_d, sigma(_d)).(X*W))*X.
 ↪T*Lambda(_d,
Derivative(sigma(_d), _d)).(X*W)
```

[77]:
```
LcL = compose(lambd_L, sigmaApply, n)(A, B)
LcL
```

[77]:

$$(\sigma_{apply}(\nu(X,W)))_{0,0} + (\sigma_{apply}(\nu(X,W)))_{0,1} + (\sigma_{apply}(\nu(X,W)))_{1,0} + (\sigma_{apply}(\nu(X,W)))_{1,1} + (\sigma_{apply}(\nu(X,W)))_{2,0} + (\sigma_{apply}(\nu(X,W)))_{2,1}$$

[78]:
```
symbL = LcL.replace(n, v).replace(sigmaApply, sigmaApply_)#.diff(A)
symbL
```

[78]:

$$(d \mapsto \sigma(d))_{\circ}(XW)_{0,0} + (d \mapsto \sigma(d))_{\circ}(XW)_{0,1} + (d \mapsto \sigma(d))_{\circ}(XW)_{1,0} + (d \mapsto \sigma(d))_{\circ}(XW)_{1,1} + (d \mapsto \sigma(d))_{\circ}(XW)_{2,0} + (d \mapsto \sigma(d))_{\circ}(XW)_{2,1}$$

[79]:
```
compose(lambd, sigmaApply, n)(A,B).replace(n,v).subs({lambd:lambd_L})#.
 ↪subs({sigmaApply : sigmaApply_L})
```

[79]:

$$(\sigma_{apply}(XW))_{0,0} + (\sigma_{apply}(XW))_{0,1} + (\sigma_{apply}(XW))_{1,0} + (\sigma_{apply}(XW))_{1,1} + (\sigma_{apply}(XW))_{2,0} + (\sigma_{apply}(XW))_{2,1}$$

[80]:
```
compose(lambd, sigmaApply, n)(A,B).replace(n,v).replace(sigmaApply,␣
 ↪sigmaApply_).replace(lambd, lambd_L)
```

[80]:

$$(d \mapsto \sigma(d))_{\circ}(XW)_{0,0} + (d \mapsto \sigma(d))_{\circ}(XW)_{0,1} + (d \mapsto \sigma(d))_{\circ}(XW)_{1,0} + (d \mapsto \sigma(d))_{\circ}(XW)_{1,1} + (d \mapsto \sigma(d))_{\circ}(XW)_{2,0} + (d \mapsto \sigma(d))_{\circ}(XW)_{2,1}$$

[81]:
```
compose(lambd, sigmaApply, n)(A,B).replace(n,v).replace(sigmaApply,␣
 ↪sigmaApply_).replace(lambd, lambd_L).doit()
```

[81]:

$$\sigma(W_{0,0}X_{0,0} + W_{1,0}X_{0,1} + W_{2,0}X_{0,2}) + \sigma(W_{0,0}X_{1,0} + W_{1,0}X_{1,1} + W_{2,0}X_{1,2}) + \sigma(W_{0,0}X_{2,0} + W_{1,0}X_{2,1} + W_{2,0}X_{2,2}) + \sigma(W_{0,1}X_{0,0} + W_{1,1}X_{0,1} + W_{2,1}X_{0,2}) + \sigma(W_{0,1}X_{1,0} + W_{1,1}X_{1,1} + W_{2,1}X_{1,2}) + \sigma(W_{0,1}X_{2,0} + W_{1,1}X_{2,1} + W_{2,1}X_{2,2})$$

[82]:
```
compose(lambd, sigmaApply, n)(A,B).replace(n,v).diff(B).
 ↪doit()#replace(sigmaApply, sigmaApply_)#.replace(lambd, lambd_L).
 ↪diff(B)
```

[82]:

$$\frac{d}{d\xi_{1}}\sigma_{apply}(\xi_{1})\Big|_{\xi_{1}=XW}\frac{\partial}{\partial\sigma_{apply}(XW)}\lambda(\sigma_{apply}(XW))\frac{\partial}{\partial W}XW$$

[83]:
```
compose(lambd, sigmaApply, n)(A,B).replace(n,v).diff(B).replace(lambd, lambd_L)
```

[83]:
$$\frac{\partial}{\partial\sigma_{apply}(XW)}\left((\sigma_{apply}(XW))_{0,0}+(\sigma_{apply}(XW))_{0,1}+(\sigma_{apply}(XW))_{1,0}+(\sigma_{apply}(XW))_{1,1}+(\sigma_{apply}(XW))_{2,0}+(\sigma_{apply}(XW))_{2,1}\right)\frac{d}{d\xi_{1}}\sigma_{apply}(\xi_{1})\Big|_{\xi_{1}=XW}\frac{\partial}{\partial W}XW$$

[84]:
```
compose(lambd, sigmaApply, n)(A,B).replace(lambd, lambd_L)
```

[84]:
$$(\sigma_{apply}(\nu(X,W)))_{0,0}+(\sigma_{apply}(\nu(X,W)))_{0,1}+(\sigma_{apply}(\nu(X,W)))_{1,0}+(\sigma_{apply}(\nu(X,W)))_{1,1}+(\sigma_{apply}(\nu(X,W)))_{2,0}+(\sigma_{apply}(\nu(X,W)))_{2,1}$$

[85]:
```
compose(lambd, sigmaApply, n)(A,B).replace(lambd, lambd_L).replace(n, v).replace(sigmaApply, sigmaApply_)
```

[85]:
$$(d\mapsto\sigma(d))_{\circ}(XW)_{0,0}+(d\mapsto\sigma(d))_{\circ}(XW)_{0,1}+(d\mapsto\sigma(d))_{\circ}(XW)_{1,0}+(d\mapsto\sigma(d))_{\circ}(XW)_{1,1}+(d\mapsto\sigma(d))_{\circ}(XW)_{2,0}+(d\mapsto\sigma(d))_{\circ}(XW)_{2,1}$$

[86]:
```
compose(lambd, sigmaApply, n)(A,B).replace(lambd, lambd_L).replace(n, v).replace(sigmaApply, sigmaApply_).doit()#.diff(B)
```

[86]:
$$\sigma(W_{0,0}X_{0,0}+W_{1,0}X_{0,1}+W_{2,0}X_{0,2})+\sigma(W_{0,0}X_{1,0}+W_{1,0}X_{1,1}+W_{2,0}X_{1,2})+\sigma(W_{0,0}X_{2,0}+W_{1,0}X_{2,1}+W_{2,0}X_{2,2})+\sigma(W_{0,1}X_{0,0}+W_{1,1}X_{0,1}+W_{2,1}X_{0,2})+\sigma(W_{0,1}X_{1,0}+W_{1,1}X_{1,1}+W_{2,1}X_{1,2})+\sigma(W_{0,1}X_{2,0}+W_{1,1}X_{2,1}+W_{2,1}X_{2,2})$$

[87]:
```
compose(lambd, sigmaApply, n)(A,B).replace(lambd, lambd_L).replace(n, v).replace(sigmaApply, sigmaApply_).doit().diff(Matrix(B)).doit()
```

[87]:
$$\begin{bmatrix} X_{0,0}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,0}X_{0,0}+W_{1,0}X_{0,1}+W_{2,0}X_{0,2}}+X_{1,0}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,0}X_{1,0}+W_{1,0}X_{1,1}+W_{2,0}X_{1,2}}+X_{2,0}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,0}X_{2,0}+W_{1,0}X_{2,1}+W_{2,0}X_{2,2}} & X_{0,0}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,1}X_{0,0}+W_{1,1}X_{0,1}+W_{2,1}X_{0,2}}+X_{1,0}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,1}X_{1,0}+W_{1,1}X_{1,1}+W_{2,1}X_{1,2}}+X_{2,0}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,1}X_{2,0}+W_{1,1}X_{2,1}+W_{2,1}X_{2,2}} \\ X_{0,1}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,0}X_{0,0}+W_{1,0}X_{0,1}+W_{2,0}X_{0,2}}+X_{1,1}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,0}X_{1,0}+W_{1,0}X_{1,1}+W_{2,0}X_{1,2}}+X_{2,1}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,0}X_{2,0}+W_{1,0}X_{2,1}+W_{2,0}X_{2,2}} & X_{0,1}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,1}X_{0,0}+W_{1,1}X_{0,1}+W_{2,1}X_{0,2}}+X_{1,1}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,1}X_{1,0}+W_{1,1}X_{1,1}+W_{2,1}X_{1,2}}+X_{2,1}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,1}X_{2,0}+W_{1,1}X_{2,1}+W_{2,1}X_{2,2}} \\ X_{0,2}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,0}X_{0,0}+W_{1,0}X_{0,1}+W_{2,0}X_{0,2}}+X_{1,2}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,0}X_{1,0}+W_{1,0}X_{1,1}+W_{2,0}X_{1,2}}+X_{2,2}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,0}X_{2,0}+W_{1,0}X_{2,1}+W_{2,0}X_{2,2}} & X_{0,2}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,1}X_{0,0}+W_{1,1}X_{0,1}+W_{2,1}X_{0,2}}+X_{1,2}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,1}X_{1,0}+W_{1,1}X_{1,1}+W_{2,1}X_{1,2}}+X_{2,2}\frac{d}{d\xi_1}\sigma(\xi_1)\Big|_{\xi_1=W_{0,1}X_{2,0}+W_{1,1}X_{2,1}+W_{2,1}X_{2,2}} \end{bmatrix}$$

[88]:
```
sigmaApply = Function("sigma_apply", subscriptable=True)

#compose(lambd, sigmaApply, n)(A,B).replace(n,v).diff(B).replace(lambd, lambd_L).doit()# ERROR sigma apply is not subscriptable
```

```
#compose(lambd, sigmaApply, n)(A,B).replace(n,v).diff(B).
↪subs({sigmaApply: sigmaApply_L})
```

[89]:
```
compose(sigmaApply_L, sigmaApply_L)(A)
```

[89]: $(d \mapsto \sigma(d))_{\circ} ((d \mapsto \sigma(d))_{\circ} (X))$

[90]:
```
x = Symbol('x', applyfunc=True)
#compose(sigmaApply_, sigmaApply_)(x)##ERROR
compose(sigmaApply_, sigmaApply_)(A)#.replace(A,f(x))
```

[90]: $(d \mapsto \sigma(d))_{\circ} ((d \mapsto \sigma(d))_{\circ} (X))$

[91]:
```
compose(lambda_L, nL)(A,B)
```

[91]: 0

[92]:
```
n = Function("v", subscriptable=True) # doesn't work for below
#compose(lambda_L, n)(A,B).doit()
```

[93]:
```
VL = Lambda((A,B), Lambda((A,B), MatrixSymbol("V", A.shape[0], B.
↪shape[1])))
VL
```

[93]: $((X,\ W) \mapsto ((X,\ W) \mapsto V))$

[94]:
```
VL(A,B)
```

[94]: $((X,\ W) \mapsto V)$

[95]:
```
#saL = Lambda(A, Lambda(A, sigma(A)))
saL = Lambda(x, Lambda(x, sigma(x)))
#saL(n(A,B))## ERROR : the ultimate test failed: cannot even make this␣
↪take an arbitrary function
#saL(n)
#s = lambda x : Lambda(x, sigma(x))
s = lambda x : sigma(x)
s(n(A,B))
```

[95]: $\sigma(v(X,W))$

[96]:

```
#sL = lambda x : Lambda(x, sigma(x))
#sL = Lambda(x, lambda x: sigma(x))

sL = Lambda(x, Lambda(x, sigma(x)))
sL
#sL(A)
```

[96]:

$$(x \mapsto (x \mapsto \sigma(x)))$$