Fundamentals on the package

Yuu Miino

March 11, 2023

1 Derivatives of a composite map

1.1 Singly composite map

Let V, W, and X be vector spaces over $\mathbb{R}$, and f and g be diffeomorphisms defined by

$$f : V \to W, \quad g : W \to X. \tag{1}$$

Then, the Jacobian matrix of f and g are elements of tensor products

$$J_f \in W \otimes V^*, \quad J_g \in X \otimes W^*, \tag{2}$$

where V^* and W^* are the dual spaces of V and W, respectively. J_f is also a map $V \to W$; J_g is $W \to X$. If J_f and J_g are differentiable in V and W, the Hessian tensors H_f and H_g are available as elements of another tensor products

$$H_f \in W \otimes V^* \otimes V^*, \quad H_g \in X \otimes W^* \otimes W^*. \tag{3}$$

On the other hand, a composite map $g \circ f$ is described as

$$g \circ f : V \to X. \tag{4}$$

From the chain-rule, the Jacobian matrix of $g \circ f$ is

$$J_{g\circ f} = J_g J_f \quad \in X \otimes V^*. \tag{5}$$

In the context of the tensor, Eq. (5) is equivalent to the following contraction of the tensor product.

$$J_{g\circ f} = \mathrm{tr}_{23}\left(J_g \otimes J_f\right), \tag{6}$$

where

$$\begin{aligned} J_g \otimes J_f &\in (X \otimes W^*) \otimes (W \otimes V^*), \\ \mathrm{tr}_{23} &: (X \otimes W^*) \otimes (W \otimes V^*) \to X \otimes V^*. \end{aligned} \tag{7}$$

We write the (i, j) contraction of a tensor by tr_{ij}, which is also known as a generalization of the trace. The Hessian tensor of $g \circ f$ is an element of $X \otimes V^* \otimes V^*$. From the chain-rule, we have

$$H_{g\circ f} = \left(H_g J_f\right) J_f + J_g H_f, \tag{8}$$

where

$$\begin{aligned} H_g J_f &= \mathrm{tr}_{24}\left(H_g \otimes J_f\right) \quad \left(= \mathrm{tr}_{34}\left(H_g \otimes J_f\right)\right), \\ H_g \otimes J_f &\in (X \otimes W^* \otimes W^*) \otimes (W \otimes V^*), \\ \mathrm{tr}_{24}\,(= \mathrm{tr}_{34}) &: (X \otimes W^* \otimes W^*) \otimes (W \otimes V^*) \to X \otimes W^* \otimes V^*, \end{aligned} \tag{9}$$

$$\begin{aligned} \left(H_g J_f\right) J_f &= \mathrm{tr}_{24}\left(\left(H_g J_f\right) \otimes J_f\right), \\ \mathrm{tr}_{24} &: (X \otimes W^* \otimes V^*) \otimes (W \otimes V^*) \to X \otimes V^* \otimes V^*, \end{aligned} \tag{10}$$

$$\begin{aligned} J_g H_f &= \mathrm{tr}_{23}\left(J_g \otimes H_f\right), \\ J_g \otimes H_f &\in (X \otimes W^*) \otimes (W \otimes V^* \otimes V^*), \\ \mathrm{tr}_{23} &: (X \otimes W^*) \otimes (W \otimes V^* \otimes V^*) \to X \otimes V^* \otimes V^*. \end{aligned} \tag{11}$$

In the Python implementation with NumPy, an "@" operator (equivalently "numpy.matmul") works well to shorten the calculation code of some contractions. Listing 1 shows the example usage. The option `axes = 0` makes the function return the tensor product of specified tensors.

```python
import numpy as np

Hg = np.random.randint(100, size=18).reshape(2, 3, 3)   # X, W*, W*
Hg = (Hg + Hg.transpose(0, 2, 1)) / 2                   # Make Hg symmetric
Jf = np.random.randint(100, size=12).reshape(3, 4)      # W, V*

# (2, 4) contraction (for W*, W)
cont24 = np.trace(np.tensordot(Hg, Jf, axes=0), axis1=1, axis2=3)

# (3, 4) contraction (for W*, W)
cont34 = np.trace(np.tensordot(Hg, Jf, axes=0), axis1=2, axis2=3)

# @ operation
atop = Hg @ Jf

# Comparison
print(cont24 == atop) # (2, 3, 4) tensor in X, W*, V*, with All True
print(cont34 == atop) # (2, 3, 4) tensor in X, W*, V*, with All True
```

Listing 1: Example usage of "@" operator.

Note that the "@" operator just calculates the matrix multiplication of two matrices, which compose of the last two indices of the target tensors, for each index of the other axes. Hence, we cannot directly use "@" operator with Eq. (10) and Eq. (11). In the cases, we can use "@" operator after swapping the axes of the tensors, as shown in Lsts. 2 and 3.

```python
# (2, 4) contraction (for W*, W)
cont24 = np.trace(np.tensordot(Hg @ Jf, Jf, axes=0), axis1=1, axis2=3)

# @ operation
atop = (Hg @ Jf).transpose(0, 2, 1) @ Jf

print(cont24 == atop) # (2, 4, 4) tensor in X, V*, V*, with All True
```

Listing 2: Example usage of "@" operator after axes swapping (for Eq. (10)).

As commented in Lst. 3, the "@" operation between a matrix `Jg` and a tensor `Hf.transpose(1, 0, 2)` raise unexpected result in the mathematical sence. The operator yields $V^* \otimes X \otimes V^*$ from $(X \otimes W^*)$ and $(V^* \otimes W \otimes V^*)$ because it just calculates the matrix product $X \otimes V^*$ from $X \otimes W^*$ and $W \otimes V^*$ for each element in V^*. We can revert the order by `transpose` method again.

```python
Jg = np.random.randint(100, size=6).reshape(2, 3)       # X, W*
Hf = np.random.randint(100, size=48).reshape(3, 4, 4)   # W, V*, V*
Hf = (Hf + Hf.transpose(0, 2, 1)) / 2                   # Make Hf symmetric

# (2, 3) contraction (for W*, W)
cont23 = np.trace(np.tensordot(Jg, Hf, axes=0), axis1=1, axis2=2)

# @ operation
atop = Jg @ Hf.transpose(1, 0, 2)   # (4, 2, 4) tensor
atop = atop.transpose(1, 0, 2)      # (2, 4, 4) tensor

print(cont23 == atop) # (2, 4, 4) tensor in X, V*, V*, with All True
```

Listing 3: Example usage of "@" operator after axes swapping (for Eq. (11)).

One can use the Einstein notation instead of a complex combination of traces and transposes, as shown in Lst. 4.

```python
# Einstein notation
ein = np.einsum('ij, jkl->ikl', Jg, Hf) # X W*, W V* V* -> X V* V*

print(cont23 == ein)
```

Listing 4: Example usage of "numpy.einsum" function (for Eq. (11)).

1.2 Multiply composite map

Let T be a composite map of T_i

$$T = T_{m-1} \circ T_{m-2} \circ \cdots \circ T_1 \circ T_0, \quad T : M_0 \to M_m \tag{12}$$

where T_i is a C^∞ diffeomorphism $T_i : M_i \to M_{i+1}$ and $M_i \subset \mathbb{R}$. Given $\boldsymbol{x}_i \in M_i$, the Jacobian matrix of T is

$$J = \frac{\partial T}{\partial \boldsymbol{x}_0} = \prod_{k=0}^{m-1} \frac{\partial T_{m-1-k}}{\partial \boldsymbol{x}_{m-1-k}}. \tag{13}$$

Let us denote the product in the right-hand of equation as J_{m-1}, and we have

$$J_{m-1} = \frac{\partial T_{m-1}}{\partial \boldsymbol{x}_{m-1}} J_{m-2} \quad \in M_m \otimes M_0^*, \tag{14}$$

for $m \geq 2$. If J_k is Hessian tensor of T is available by differentiating Eq. (14),

$$H = \frac{\partial J}{\partial \boldsymbol{x}_0} = \left(\frac{\partial^2 T_{m-1}}{\partial \boldsymbol{x}_{m-1}{}^2} J_{m-2} \right) J_{m-2} + \frac{\partial T_{m-1}}{\partial \boldsymbol{x}_{m-1}} \frac{\partial J_{m-2}}{\partial \boldsymbol{x}_0}. \tag{15}$$

Rewriting the derivative of J_{m-1} by H_{m-1}, we get

$$H_{m-1} = \left(\frac{\partial^2 T_{m-1}}{\partial \boldsymbol{x}_{m-1}{}^2} J_{m-2} \right) J_{m-2} + \frac{\partial T_{m-1}}{\partial \boldsymbol{x}_{m-1}} H_{m-2} \quad \in M_m \otimes M_0^* \otimes M_0^*. \tag{16}$$

Notice that $(J_{m-2}\, J_{m-2})$ is sometimes not J_{m-2}^2 because the dimensions of M_i and M_{i+1}^* are not necessarily equal.

In the Python implementation with NumPy, one can write Eqs. (14) and (16) in oneline coding, as shown in Lst. 5. Remember that the listing is just an example of calculation ways but is not directly implemented in the package. In the example, we assume the composite map composed of 8 mappings which have random dimensions. The code implements the calculation of Eqs. (14) and Eqs. (16) using dummy values of Jacobian matrices and Hessian tensors. The output example included below implies the dimension correspondence before and after the calculation. All Jacobian matrices transform from $M_i \otimes M_0^*$ to $M_{i+1} \otimes M_0^*$; all Hesssian tensors do from $M_0^* \otimes M_i \otimes M_0^*$ to $M_0^* \otimes M_{i+1} \otimes M_0^*$. Notice that we swap the axes of the Hessian tensor from $M_i \otimes M_0^* \otimes M_0^*$ to $M_0^* \otimes M_i \otimes M_0^*$, to simplify the code of the recurrence relation.

```
dom = random_dimension() # Dimension of domain   of T0, hence M_0
cod = random_dimension() # Dimension of codomain of T0, hence M_1

# Jacobian matrix (dummy) in M_1, M_0*
jac = get_jac(dom, cod)

# Hessian tensor (dummy) in M_1, M_0*, M_0*
# (swapped with M_0*, M_1, M_0* for convinience)
hes = get_hes(dom, cod)

first_info(dom, cod, jac, hes)  # Just for print

# If composite 8 mappings
for i in range(1, 8):
    next_cod = random_dimension()           # Dimension of next codomain
    prev_info(i, cod, next_cod, jac, hes)   # Just for print

    pdv1 = get_jac(cod, next_cod)   # dTi/dxi
    pdv2 = get_hes(cod, next_cod)   # d2Ti/dxi2

    # Oneline code of the reccurence relations
    jac, hes = pdv1 @ jac, (pdv2 @ jac).T @ jac + pdv1 @ hes

    next_info(jac, hes) # Just for print
    cod = next_cod      # Rotation of codomain

print(jac.shape, hes.shape)

"""Output example
T0: M0 (6) -> M1 (2) jac: (6, 6) hes: (6, 6, 6) -> jac: (2, 6) hes: (6, 2, 6)
T1: M1 (2) -> M2 (4) jac: (2, 6) hes: (6, 2, 6) -> jac: (4, 6) hes: (6, 4, 6)
T2: M2 (4) -> M3 (5) jac: (4, 6) hes: (6, 4, 6) -> jac: (5, 6) hes: (6, 5, 6)
```

```
T3: M3 (5) -> M4 (2) jac: (5, 6) hes: (6, 5, 6) -> jac: (2, 6) hes: (6, 2, 6)
T4: M4 (2) -> M5 (5) jac: (2, 6) hes: (6, 2, 6) -> jac: (5, 6) hes: (6, 5, 6)
T5: M5 (5) -> M6 (2) jac: (5, 6) hes: (6, 5, 6) -> jac: (2, 6) hes: (6, 2, 6)
T6: M6 (2) -> M7 (3) jac: (2, 6) hes: (6, 2, 6) -> jac: (3, 6) hes: (6, 3, 6)
T7: M7 (3) -> M8 (1) jac: (3, 6) hes: (6, 3, 6) -> jac: (1, 6) hes: (6, 1, 6)
(1, 6) (6, 1, 6)
"""
```

Listing 5: Example implementation of the recurrence relations (14) and (16).

2 Derivative of a map for the continuous-time systems

Consider a C^∞ autonomous dynamical system

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}), \quad \boldsymbol{x} \in M \subset \mathbb{R}^n, \quad t \in \mathbb{R}, \tag{17}$$

where M is a state space, and $\boldsymbol{f}$ is a function such that $M \to X \subset \mathbb{R}^n$. We write the trajectory of the system (17) by $\boldsymbol{\varphi} : X \times M \to M$, where $\boldsymbol{\varphi}(0, \boldsymbol{x}_0) = \boldsymbol{x}_0$ is the initial state and $\boldsymbol{\varphi}(t, \boldsymbol{x}_0)$ is the state at t. Let ∂M be an $n-1$ dimensional manifold defined by a conditional function $q : M \to \mathbb{R}$

$$\partial M = \{\boldsymbol{x} \in M \mid q(\boldsymbol{x}) = 0\} \tag{18}$$

Suppose that T_0 is a local map from $\boldsymbol{x}_0 \in M$ to a point in ∂M such that $M \to \partial M$. Then, the Jacobian matrix of T_0 is described by

$$J_{T_0} = \frac{\partial T_0}{\partial \boldsymbol{x}_0} = \left[I - \frac{1}{\frac{dq}{d\boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x})} \left(\boldsymbol{f}(\boldsymbol{x}) \otimes \frac{dq}{d\boldsymbol{x}} \right) \right] \Bigg|_{\boldsymbol{x}=\boldsymbol{x}_1} \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(\tau) = B(\boldsymbol{x}_1) \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(\tau), \tag{19}$$

where I is an identity matrix, τ is the spent time during the trajectory $\boldsymbol{\varphi}$ moves from $\boldsymbol{x}_0$ to the boundary ∂M, which only depends on $\boldsymbol{x}_0$, and $\boldsymbol{x}_1 = \boldsymbol{\varphi}(\tau, \boldsymbol{x}_0)$. The partial derivative of the trajectory with respect to the initial state at the time t is the solution of the following ordinary differential equation

$$\frac{d}{dt}\left(\frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0} \right) = \frac{d\boldsymbol{f}}{d\boldsymbol{x}} \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}, \quad \frac{d\boldsymbol{\varphi}}{d\boldsymbol{x}_0}(0) = I. \tag{20}$$

The Hessian tensor of T_0 is the derivative of J_{T_0} with respect to the initial state

$$\begin{aligned} H_{T_0} &= \left(\frac{\partial B}{\partial \boldsymbol{x}}(\boldsymbol{x}_1) J_{T_0} \right) \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(\tau) + B(\boldsymbol{x}_1) \left[\frac{\partial^2 \boldsymbol{\varphi}}{\partial {\boldsymbol{x}_0}^2}(\tau) + \left(\frac{d}{dt} \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(\tau) \right) \otimes \frac{\partial \tau}{\partial \boldsymbol{x}_0} \right], \\ &= \left(\frac{\partial B}{\partial \boldsymbol{x}}(\boldsymbol{x}_1) J_{T_0} \right) \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(\tau) + B(\boldsymbol{x}_1) \left[\frac{\partial^2 \boldsymbol{\varphi}}{\partial {\boldsymbol{x}_0}^2}(\tau) - \frac{1}{\frac{dq}{d\boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x}_1)} \left(\frac{d\boldsymbol{f}}{d\boldsymbol{x}} \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(\tau) \right) \otimes \left(\frac{dq}{d\boldsymbol{x}} \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0}(\tau) \right) \right], \end{aligned} \tag{21}$$

where

$$\frac{\partial B}{\partial \boldsymbol{x}} = -\frac{1}{\left(\frac{dq}{d\boldsymbol{x}}\boldsymbol{f}(\boldsymbol{x}) \right)^2} \left\{ \left(\frac{d\boldsymbol{f}}{d\boldsymbol{x}} \otimes \frac{dq}{d\boldsymbol{x}} + \boldsymbol{f}(\boldsymbol{x}) \otimes \frac{d^2 q}{d\boldsymbol{x}^2} \right) \frac{dq}{d\boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}) - \left(\boldsymbol{f}(\boldsymbol{x}) \otimes \frac{dq}{d\boldsymbol{x}} \right) \otimes \left(\frac{d^2 q}{d\boldsymbol{x}^2} \boldsymbol{f}(\boldsymbol{x}) + \frac{dq}{d\boldsymbol{x}} \frac{d\boldsymbol{f}}{d\boldsymbol{x}} \right) \right\}. \tag{22}$$

Notice that

$$\begin{aligned} &B \in X \otimes X^*, \quad \frac{\partial B}{\partial \boldsymbol{x}} \in X \otimes X^* \otimes X^*, \quad \frac{dq}{d\boldsymbol{x}} \in X^*, \ \frac{d^2 q}{d\boldsymbol{x}^2} \in X^* \otimes X^*, \quad \boldsymbol{f}(\boldsymbol{x}) \in X, \ \frac{d\boldsymbol{f}}{d\boldsymbol{x}} \in X \otimes X^*, \\ &J_{T_0} \in X \otimes M^*, \quad \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0} \in X \otimes M^*, \quad \frac{\partial^2 \boldsymbol{\varphi}}{\partial {\boldsymbol{x}_0}^2} \in X \otimes M^* \otimes M^*, \quad H_{T_0} \in X \otimes M^* \otimes M^*. \end{aligned} \tag{23}$$

The second partial derivative of $\boldsymbol{\varphi}$ with respect to $\boldsymbol{x}_0$ is the solution of the following ordinary differential equation

$$\frac{d}{dt}\left(\frac{\partial^2 \boldsymbol{\varphi}}{\partial {\boldsymbol{x}_0}^2} \right) = \left(\frac{d^2 \boldsymbol{f}}{d\boldsymbol{x}^2} \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0} \right) \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{x}_0} + \frac{d\boldsymbol{f}}{d\boldsymbol{x}} \frac{\partial^2 \boldsymbol{\varphi}}{\partial {\boldsymbol{x}_0}^2}, \quad \frac{\partial^2 \boldsymbol{\varphi}}{\partial {\boldsymbol{x}_0}^2} = \boldsymbol{0}, \tag{24}$$

where $\boldsymbol{0}$ is a tensor with all elements of zeros, and

$$\frac{d^2 \boldsymbol{f}}{d\boldsymbol{x}^2} \in X \otimes X^* \otimes X^*. \tag{25}$$