Tensors in Computations II

Lek-Heng Lim

University of Chicago

recap from lecture I

recap: three definitions

- tensors capture three great ideas:
  1. equivariance
  2. multilinearity
  3. separability
- roughly correspond to three common definitions of a tensor
  1. a multi-indexed object that satisfies tensor transformation rules
  2. a multilinear map
  3. an element of a tensor product of vector spaces

recap: definition ①

- take home idea: definition ① is all about tensor transformation rules
- example: is this

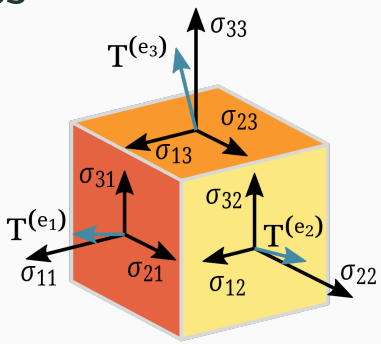
$$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 5 \end{bmatrix}$$

  a tensor?
- makes no sense
- definition ① requires a context

recap: definition ①

- if it comes from a measurement of stress

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



  then it is a contravariant 2-tensor

- if we are interested in its eigenvalues and eigenvectors

$$(XAX^{-1})X\mathbf{v} = \lambda X\mathbf{v}$$

  then it is a mixed 2-tensor

- if we are interested in its Hadamard product

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \circ \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} = \begin{bmatrix} a_{11}\,b_{11} & a_{12}\,b_{12} & a_{13}\,b_{13} \\ a_{21}\,b_{21} & a_{22}\,b_{22} & a_{23}\,b_{23} \\ a_{31}\,b_{31} & a_{32}\,b_{32} & a_{33}\,b_{33} \end{bmatrix}$$

  then it is not a tensor

recap: definition ①

- indices tell us nothing
- $A \in \mathbb{R}^{m \times n}$ has two indices but if transformation is

  $$A' = XA = [X\mathbf{a}_1, \ldots, X\mathbf{a}_n] \quad \text{or} \quad A' = X^{-\mathsf{T}}A = [X^{-\mathsf{T}}\mathbf{a}_1, \ldots, X^{-\mathsf{T}}\mathbf{a}_n]$$

  then it is covariant or contravariant 1-tensor respectively
- e.g., Householder QR algorithm treats the matrix as a covariant 1-tensor

recap: why important

saw three examples

full-rank least squares: solved by applying sequence of 0-, 1-, 2-tensor transformation rules

Krylov subspace methods: exploits relations in change-of-coordinates matrices of 2-tensor transformation rules

linearly constrained optimization: Newton method is tensorial, i.e., satisfies 0-, 1-, 2-tensor transformation rules, and thus insensitive to condition number; steepest descent is not

more examples

example: equivariant neural networks

- neural network $f : \mathbb{R}^n \to \mathbb{R}^n$ obtained by alternately composing $A_i \in \mathbb{R}^{n \times n}$ with pointwise activations $\sigma_i(\mathbf{v}) := \max(\mathbf{v}, \mathbf{b}_i)$, $\mathbf{b}_i \in \mathbb{R}^n$

$$f(\mathbf{v}) = A_k \sigma_{k-1} A_{k-1} \cdots \sigma_2 A_2 \sigma_1 A_1 \mathbf{v}$$

- G-equivariant function $f : \mathbb{R}^n \to \mathbb{R}^n$

$$f(X\mathbf{v}) = Xf(\mathbf{v})$$

  for all $X \in G \subseteq \mathrm{GL}(n)$

- observe

$$\begin{aligned} f(X\mathbf{v}) &= X(X^{-1}A_kX)(X^{-1}\sigma_{k-1}X)(X^{-1}A_{k-1}X) \\ &\qquad \cdots (X^{-1}A_2X)(X^{-1}\sigma_1X)(X^{-1}A_1X)\mathbf{v} \\ &= XA'_k\sigma'_{k-1}A'_{k-1}\cdots\sigma'_2A'_2\sigma'_1A'_1\mathbf{v} \end{aligned}$$

- equals $Xf(\mathbf{v})$ if

$$A'_i = X^{-1}A_iX = A_i, \quad \sigma'_i = X^{-1}\sigma_iX = \sigma_i$$

example: equivariant neural networks

- for image-based tasks n is on the order of millions of pixels, k typically 10 to 50 layers deep
- may not even have enough data to fit weights and biases

$$A_1, \ldots, A_k \in \mathbb{R}^{n \times n}, \qquad \mathbf{b}_1, \ldots, \mathbf{b}_{k-1} \in \mathbb{R}^n$$

- G-equivariance narrows range of possible A_i and $\mathbf{b}_i$ by requiring

$$A_i' = X^{-1} A_i X = A_i, \quad \sigma_i' = X^{-1} \sigma_i X = \sigma_i$$

- G-equivariance = mixed 2-tensor transformation rule

image-based tasks

- convolutional neural networks: group of translations

$$G = \left\{ \begin{bmatrix} 1 & 0 & m_1 \\ 0 & 1 & m_2 \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3} : \, m_1, m_2 \in \mathbb{Z} \right\}$$

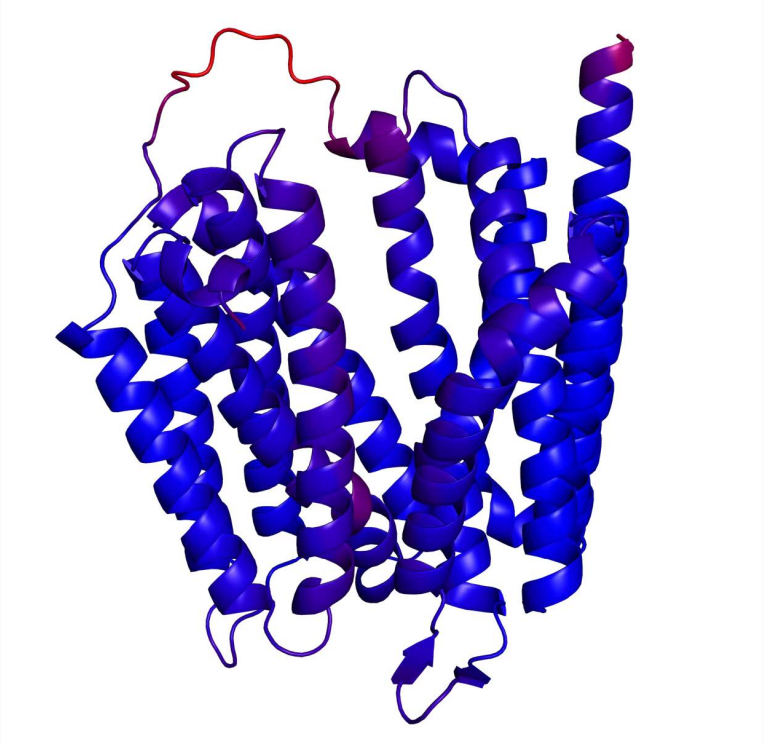
- $p4$ group of translations and right-angle rotations

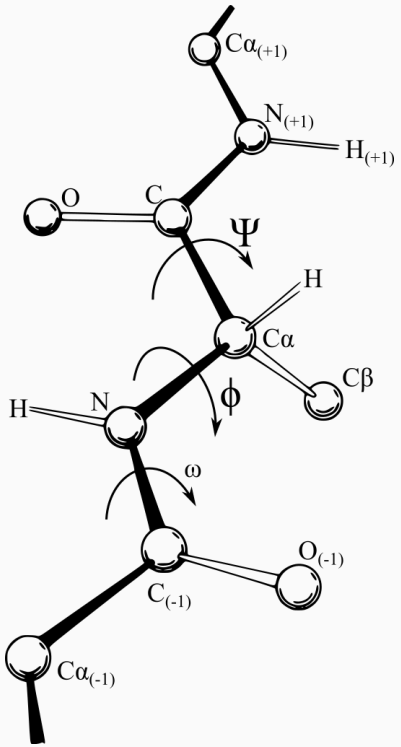
$$G = \left\{ \begin{bmatrix} \cos(k\pi/2) & -\sin(k\pi/2) & m_1 \\ \sin(k\pi/2) & \cos(k\pi/2) & m_2 \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3} : \begin{array}{r} k = 0, 1, 2, 3; \\ m_1, m_2 \in \mathbb{Z} \end{array} \right\}$$

- $p4m$ group: $p4$ plus reflections

$$G = \left\{ \begin{bmatrix} (-1)^j \cos(k\pi/2) & (-1)^{j+1} \sin(k\pi/2) & m_1 \\ \sin(k\pi/2) & \cos(k\pi/2) & m_2 \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{3\times 3} : \begin{array}{r} k = 0, 1, 2, 3; \\ j = 0, 1; \; m_1, m_2 \in \mathbb{Z} \end{array} \right\}$$

other tasks





- drug discovery: preserve pairwise distances between atoms in a molecule and chirality SO(3) or SE(3)
- DeepMind's AlphaFold 2: SE(3)-equivariant neural network and SE(3)-invariant attention module
- data from high energy phsyics: Lorentz groups $O(1,3)$ or $SO(1,3)$

example: cone programming

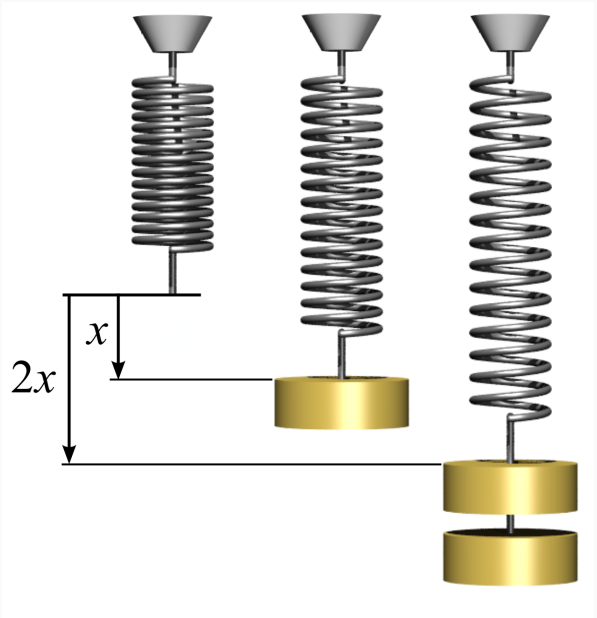
- primal and dual forms of cone programming problem over a symmetric cone $\mathbb{K} \subseteq \mathbb{V}$ conform to transformation rules for Cartesian 0-, 1-, 2-tensors
- but change-of-coordinates matrices would have to be replaced a linear map from the *orthgonal group of the cone*:

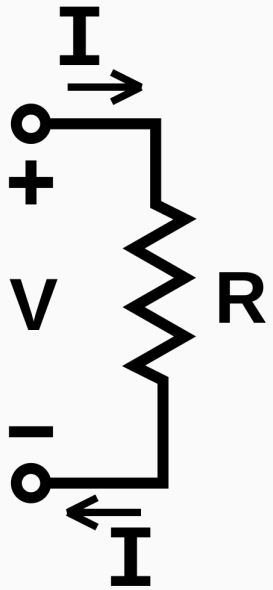
$$\mathrm{O}(\mathbb{K}) := \{\varphi : \mathbb{V} \to \mathbb{V} : \varphi \text{ linear, invertible, and } \varphi^* = \varphi^{-1}\}$$

- special cases include linear programming (LP), convex quadratic programming (QP), second-order cone programming (SOCP), and semidefinite programming (SDP)

motivation: multilinearity

why important





linearity principle: almost any natural process is linear in small amounts almost everywhere

multilinearity principle: if we keep all but one factors constant, the varying factor obeys principle of linearity

Hooke's and Ohm's laws both linear but not if we pass a current through the spring or stretch the resistor

coordinate-free tensors

Kip Thorne's Geometric Principle

The laws of physics must all be expressible as geometric (coordinate independent and reference frame independent) relationships between geometric objects (scalars, vectors, tensors, ...) that represent physical entitities.

laws of physics do not depend on coordinates, and thus the tensors used to express these laws should not depend on coordinates either

① a multi-indexed object that satisfies certain transformation rules

② a multilinear map

③ an element of a tensor product of vector spaces

tensors over arbitrary vector spaces

- vector space $\mathbb{V}$ may not be $\mathbb{R}^n$, e.g., $\mathbb{K} = \mathbb{S}^n_{++}$ and $\mathbb{V} = \mathbb{S}^n$ for SDP
- numerical linear algebra notations we have been using to describe definition ① awkward and unnatural
- want to work with tensors over arbitrary vector spaces
  - space of Toeplitz or Hankel or Toeplitz-plus-Hankel matrices
  - space of polynomials or differential forms or differential operators
  - space of L^2-functions on homogeneous spaces
- another impetus for coordinate-free approach in definitions ② and ③

tensors via multilinear maps

multilinear maps

- $\mathbb{V}_1, \dots, \mathbb{V}_d$ and $\mathbb{W}$ vector spaces
- multilinear map or d-linear map is $\varphi : \mathbb{V}_1 \times \cdots \times \mathbb{V}_d \to \mathbb{W}$ with

$$\Phi(\mathbf{v}_1, \dots, \lambda \mathbf{v}_k + \lambda' \mathbf{v}'_k, \dots, \mathbf{v}_d) = \lambda \Phi(\mathbf{v}_1, \dots, \mathbf{v}_k, \dots, \mathbf{v}_d) + \lambda' \Phi(\mathbf{v}_1, \dots, \mathbf{v}'_k, \dots, \mathbf{v}_d)$$

  for $\mathbf{v}_1 \in \mathbb{V}_1, \dots, \mathbf{v}_k, \mathbf{v}'_k \in \mathbb{V}_k, \dots, \mathbf{v}_d \in \mathbb{V}_d$, $\lambda, \lambda' \in \mathbb{R}$
- write $\mathrm{M}^d(\mathbb{V}_1, \dots, \mathbb{V}_d; \mathbb{W})$ for set of all such maps
- write $\mathrm{M}^1(\mathbb{V}; \mathbb{W}) = \mathrm{L}(\mathbb{V}; \mathbb{W})$ for linear maps
- tensor transformation rules = change-of-bases theorems for multilinear maps

$d = 1$

- $\mathscr{B} = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$ basis of $\mathbb{V}$, $\mathscr{B}^* = \{\mathbf{v}_1^*, \ldots, \mathbf{v}_m^*\}$ dual basis
- any $\mathbf{v} \in \mathbb{V}$ uniquely represented by $\mathbf{a} \in \mathbb{R}^m$

$$\mathbb{V} \ni \mathbf{v} = a_1\mathbf{v}_1 + \cdots + a_m\mathbf{v}_m \quad \longleftrightarrow \quad [v]_{\mathscr{B}} := \begin{bmatrix} a_1 \\ \vdots \\ a_m \end{bmatrix} \in \mathbb{R}^m$$

- any $\varphi \in \mathbb{V}^*$ uniquely represented by $\mathbf{b} \in \mathbb{R}^m$

$$\mathbb{V}^* \ni \varphi = b_1\mathbf{v}_1^* + \cdots + b_m\mathbf{v}_m^* \quad \longleftrightarrow \quad [\varphi]_{\mathscr{B}^*} := \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \in \mathbb{R}^m$$

$d = 1$

- if $\mathscr{C} = \{\mathbf{v}'_1, \ldots, \mathbf{v}'_m\}$ another basis and $X \in \mathbb{R}^{m \times m}$ is

$$\mathbf{v}'_j = \sum_{i=1}^{m} x_{ij} \mathbf{v}_i$$

- change-of-basis theorem:

$$[\mathbf{v}]_{\mathscr{C}} = X^{-1}[\mathbf{v}]_{\mathscr{B}}, \qquad [\varphi]_{\mathscr{C}^*} = X^{\mathsf{T}}[\varphi]_{\mathscr{B}^*}$$

- recover transformation rules for contravariant and covariant 1-tensors

$$\mathbf{a}' = X^{-1}\mathbf{a}, \qquad \mathbf{b}' = X^{\mathsf{T}}\mathbf{b}$$

- vectors $\longleftrightarrow$ contravariant 1-tensors
- linear functionals $\longleftrightarrow$ covariant 1-tensors

$d = 2$

- bases $\mathscr{A} = \{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$ on $\mathbb{U}$, $\mathscr{B} = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$ on $\mathbb{V}$
- linear operator $\Phi : \mathbb{U} \to \mathbb{V}$ has matrix representation

$$[\Phi]_{\mathscr{A},\mathscr{B}} = A \in \mathbb{R}^{m \times n}$$

where

$$\Phi(\mathbf{u}_j) = \sum_{i=1}^{m} a_{ij}\mathbf{v}_i$$

- new bases $\mathscr{A}'$ and $\mathscr{B}'$

$$[\Phi]_{\mathscr{A}',\mathscr{B}'} = A' \in \mathbb{R}^{m \times n}$$

- change-of-basis theorem: if $X \in \mathrm{GL}(m)$ change-of-basis matrix on $\mathbb{V}$, $Y \in \mathrm{GL}(n)$ change-of-basis matrix on $\mathbb{U}$, then

$$A' = X^{-1}AY$$

$d = 2$

- special case $\mathbb{U} = \mathbb{V}$ with $\mathscr{A} = \mathscr{B}$ and $\mathscr{A}' = \mathscr{B}'$

$$A' = X^{-1}AX$$

- recover transformation rules for mixed 2-tensors
- bilinear functional $\beta : \mathbb{U} \times \mathbb{V} \to \mathbb{R}$ with

$$\beta(\lambda\mathbf{u} + \lambda'\mathbf{u}', \mathbf{v}) = \lambda\beta(\mathbf{u}, \mathbf{v}) + \lambda'\beta(\mathbf{u}', \mathbf{v}),$$
$$\beta(\mathbf{u}, \lambda\mathbf{v} + \lambda'\mathbf{v}') = \lambda\beta(\mathbf{u}, \mathbf{v}) + \lambda'\beta(\mathbf{u}, \mathbf{v}')$$

  for all $\mathbf{u}, \mathbf{u}' \in \mathbb{U}$, $\mathbf{v}, \mathbf{v}' \in \mathbb{V}$, $\lambda, \lambda' \in \mathbb{R}$

- matrix representation of β

$$[\beta]_{\mathscr{A},\mathscr{B}} = A \in \mathbb{R}^{m \times n}$$

  given by

$$a_{ij} = \beta(\mathbf{u}_i, \mathbf{v}_j)$$

$d = 2$

- change-of-basis theorem: if

$$[\beta]_{\mathscr{A}',\mathscr{B}'} = A' \in \mathbb{R}^{m\times n},$$

then

$$A' = X^{\mathsf{T}} A Y$$

- special case $\mathbb{U} = \mathbb{V}$, $\mathscr{A} = \mathscr{B}$, $\mathscr{A}' = \mathscr{B}'$

$$A' = X^{\mathsf{T}} A X$$

- recover transformation rules for covariant 2-tensors
- linear operators $\longleftrightarrow$ mixed 2-tensors
- bilinear functionals $\longleftrightarrow$ covariant 2-tensors

1- and 2-tensor transformation rules

contravariant 1-tensor:	$\mathbf{a}' = X^{-1}\mathbf{a}$	$\mathbf{a}' = X\mathbf{a}$
covariant 1-tensor:	$\mathbf{a}' = X^{\mathsf{T}}\mathbf{a}$	$\mathbf{a}' = X^{-\mathsf{T}}\mathbf{a}$
covariant 2-tensor:	$A' = X^{\mathsf{T}}AX$	$A' = X^{-\mathsf{T}}AX^{-1}$
contravariant 2-tensor:	$A' = X^{-1}AX^{-\mathsf{T}}$	$A' = XAX^{\mathsf{T}}$
mixed 2-tensor:	$A' = X^{-1}AX$	$A' = XAX^{-1}$
contravariant 2-tensor:	$A' = X^{-1}AY^{-\mathsf{T}}$	$A' = XAY^{\mathsf{T}}$
covariant 2-tensor:	$A' = X^{\mathsf{T}}AY$	$A' = X^{-\mathsf{T}}AY^{-1}$
mixed 2-tensor:	$A' = X^{-1}AY$	$A' = XAY^{-1}$

$d = 3$

- bilinear operator $\mathrm{B} : \mathbb{U} \times \mathbb{V} \to \mathbb{W}$,

$$\mathrm{B}(\lambda \mathbf{u} + \lambda' \mathbf{u}', \mathbf{v}) = \lambda \mathrm{B}(\mathbf{u}, \mathbf{v}) + \lambda' \mathrm{B}(\mathbf{u}', \mathbf{v}),$$
$$\mathrm{B}(\mathbf{u}, \lambda \mathbf{v} + \lambda' \mathbf{v}') = \lambda \mathrm{B}(\mathbf{u}, \mathbf{v}) + \lambda' \mathrm{B}(\mathbf{u}, \mathbf{v}')$$

- bases $\mathscr{A} = \{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$, $\mathscr{B} = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$, $\mathscr{C} = \{\mathbf{w}_1, \ldots, \mathbf{w}_p\}$,

$$\mathrm{B}(\mathbf{u}_i, \mathbf{v}_j) = \sum_{i=1}^{p} a_{ijk} \mathbf{w}_k$$

- change-of-basis theorem: if

$$[\mathrm{B}]_{\mathscr{A},\mathscr{B},\mathscr{C}} = A \quad \text{and} \quad [\mathrm{B}]_{\mathscr{A}',\mathscr{B}',\mathscr{C}'} = A' \in \mathbb{R}^{m \times n \times p},$$

then

$$A' = (X^\mathsf{T}, Y^\mathsf{T}, Z^{-1}) \cdot A$$

$d = 3$

- trilinear functional $\tau : \mathbb{U} \times \mathbb{V} \times \mathbb{W} \to \mathbb{R}$,

$$\begin{aligned}\tau(\lambda\mathbf{u} + \lambda'\mathbf{u}', \mathbf{v}, \mathbf{w}) &= \lambda\tau(\mathbf{u}, \mathbf{v}, \mathbf{w}) + \lambda'\tau(\mathbf{u}', \mathbf{v}, \mathbf{w}),\\ \tau(\mathbf{u}, \lambda\mathbf{v} + \lambda'\mathbf{v}', \mathbf{w}) &= \lambda\tau(\mathbf{u}, \mathbf{v}, \mathbf{w}) + \lambda'\tau(\mathbf{u}, \mathbf{v}', \mathbf{w}),\\ \tau(\mathbf{u}, \mathbf{v}, \lambda\mathbf{w} + \lambda'\mathbf{w}') &= \lambda\tau(\mathbf{u}, \mathbf{v}, \mathbf{w}) + \lambda'\tau(\mathbf{u}, \mathbf{v}, \mathbf{w}')\end{aligned}$$

- bases $\mathscr{A} = \{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$, $\mathscr{B} = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\}$, $\mathscr{C} = \{\mathbf{w}_1, \ldots, \mathbf{w}_p\}$,

$$\tau(\mathbf{u}_i, \mathbf{v}_j, \mathbf{w}_k) = a_{ijk}$$

- change-of-basis theorem: if

$$[\tau]_{\mathscr{A},\mathscr{B},\mathscr{C}} = A \quad \text{and} \quad [\tau]_{\mathscr{A}',\mathscr{B}',\mathscr{C}'} = A' \in \mathbb{R}^{m\times n\times p},$$

then

$$A' = (X^\mathsf{T}, Y^\mathsf{T}, Z^\mathsf{T}) \cdot A$$

extends to arbitrary order

- bilinear operators $\longleftrightarrow$ mixed 3-tensor of covariant order 2 contravariant order 1
- trilinear functionals $\longleftrightarrow$ covariant 3-tensor
- recover all transformation rules in definition ①
  - covariant d-tensor $f : \mathbb{V}_1 \times \cdots \times \mathbb{V}_d \to \mathbb{R}$

    $$A' = (X_1^{\mathsf{T}}, X_2^{\mathsf{T}}, \ldots, X_d^{\mathsf{T}}) \cdot A$$

  - contravariant d-tensor $f : \mathbb{V}_1^* \times \cdots \times \mathbb{V}_d^* \to \mathbb{R}$

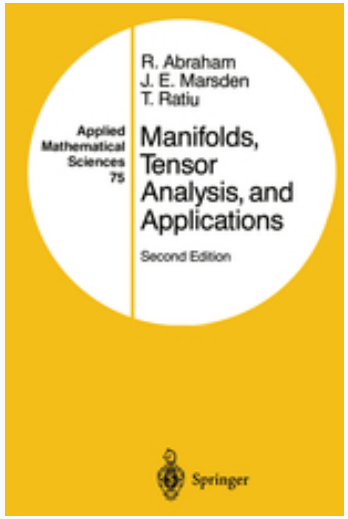
    $$A' = (X_1^{-1}, X_2^{-1}, \ldots, X_d^{-1}) \cdot A$$

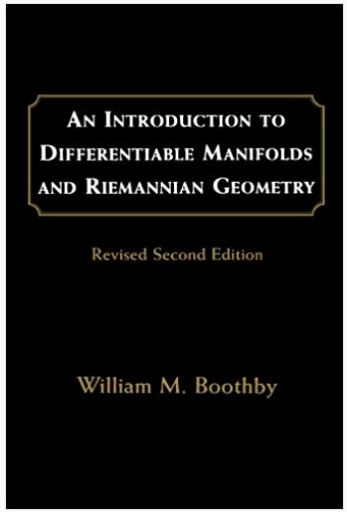
  - mixed d-tensor $f : \mathbb{V}_1^* \times \cdots \times \mathbb{V}_p^* \times \mathbb{V}_{p+1} \cdots \times \mathbb{V}_d \to \mathbb{R}$

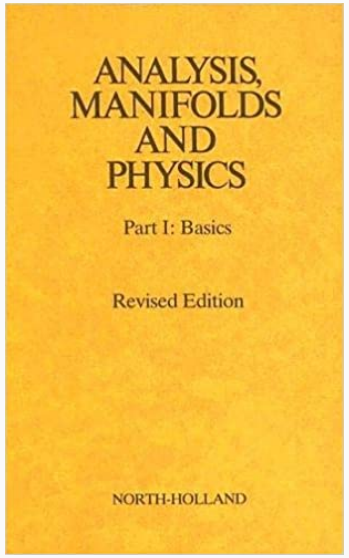
    $$A' = (X_1^{-1}, \ldots, X_p^{-1}, X_{p+1}^{\mathsf{T}}, \ldots, X_d^{\mathsf{T}}) \cdot A$$

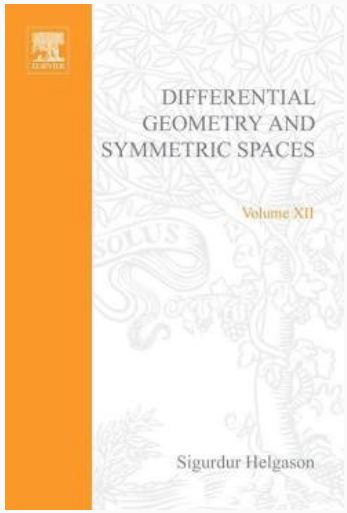
- definition ② is the easiest definition of a tensor (but it has issues)

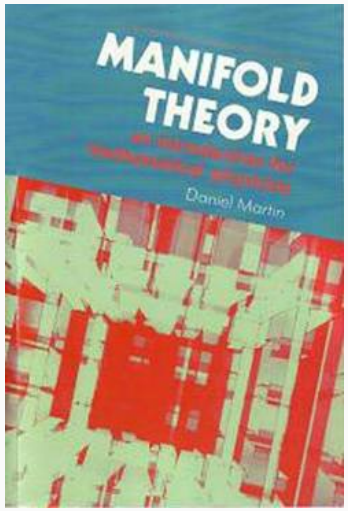
adopted in books c. 1980s

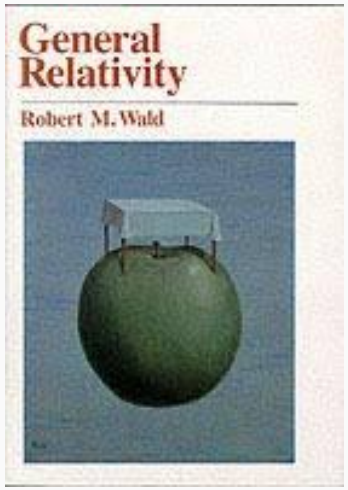


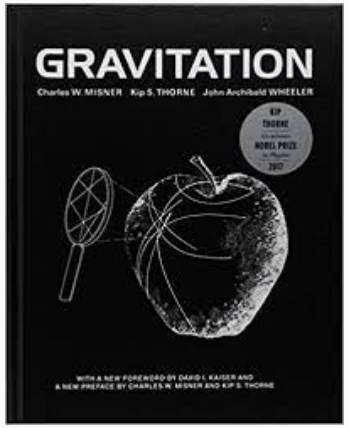


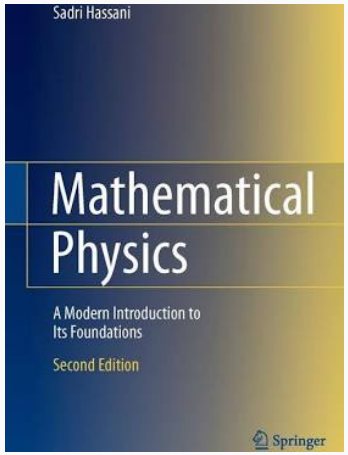












aside: why not hypermatrices

why not hypermatrices

- why not choose bases and just treat them as hypermatrices $A \in \mathbb{R}^{n_1 \times \cdots \times n_d}$?
- may not be computable: writing down a hypermatrix given bases in general #P-hard
- may not be possible: multilinear maps extend to modules, which may not have bases
- may not be useful: even for $d = 1, 2$, writing down matrix unhelpful when vector spaces have special structures
- may not be meaningful: 'indices' may be continuous

writing down hypermatrix is #P-hard

- $0 \leq d_1 \leq d_2 \leq \cdots \leq d_n$, generalized Vandermonde matrix

$$V_{(d_1,\ldots,d_n)}(\mathbf{x}) := \begin{bmatrix} x_1^{d_1} & x_2^{d_1} & \cdots & x_n^{d_1} \\ x_1^{d_2} & x_2^{d_2} & \cdots & x_n^{d_2} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{d_{n-1}} & x_2^{d_{n-1}} & \cdots & x_n^{d_{n-1}} \\ x_1^{d_n} & x_2^{d_n} & \cdots & x_n^{d_n} \end{bmatrix}$$

- usual Vandermonde matrix

$$V_{(0,1,\ldots,n-1)}(\mathbf{x}) = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{n-2} & x_2^{n-2} & \cdots & x_n^{n-2} \\ x_1^{n-1} & x_2^{n-1} & \cdots & x_n^{n-1} \end{bmatrix}$$

- if $d_i \geq i$, then $\det V_{(d_1,d_2,\ldots,d_n)}(\mathbf{x})$ divisible by $\det V_{(0,1,\ldots,n-1)}(\mathbf{x})$

writing down hypermatrix is #P-hard

- for any integers $0 \leq p_1 \leq p_2 \leq \cdots \leq p_n$,

$$s_{(p_1,p_2,\ldots,p_n)}(\mathbf{x}) := \frac{\det V_{(p_1,p_2+1,\ldots,p_n+n-1)}(\mathbf{x})}{\det V_{(0,1,\ldots,n-1)}(\mathbf{x})}$$

  is a symmetric polynomial in the variables $x_1, \ldots, x_n$

$$s(x_1, x_2, \ldots, x_n) = s\big(x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(n)}\big)$$

- $\mathbb{U}$, $\mathbb{V}$, $\mathbb{W}$ vector spaces of symmetric polynomials of degrees d, d', and $d + d'$ respectively
- $\mathrm{B} : \mathbb{U} \times \mathbb{V} \to \mathbb{W}$ bilinear operator given by

$$\mathrm{B}\big(s(\mathbf{x}), t(\mathbf{x})\big) = s(\mathbf{x})t(\mathbf{x})$$

  for $s(\mathbf{x})$ of degree d and $t(\mathbf{x})$ of degree d'

writing down hypermatrix is #P-hard

- $\mathscr{A}$ be basis of $\mathbb{U}$ given by

$$\left\{ s_{(p_1,p_2,\ldots,p_n)}(\mathbf{x}) \in \mathbb{U} : p_1 \leq p_2 \leq \cdots \leq p_n \text{ integer partition of } d \right\}$$

- $\mathscr{B}$, $\mathscr{C}$ similar bases for $\mathbb{V}$, $\mathbb{W}$
- B may in principle be written down as 3-dimensional hypermatrix

$$[\mathrm{B}]_{\mathscr{A},\mathscr{B},\mathscr{C}} = A \in \mathbb{R}^{d \times d' \times (d+d')}$$

with

$$\mathrm{B}(\mathbf{u}_i, \mathbf{v}_j) = \sum_{i=1}^{p} a_{ijk} \mathbf{w}_k$$

- a_{ijk}'s are Littlewood–Richardson coefficients, well-known to be #P-complete [Narayanan, 2006]
- used in resolution of Horn's conjecture on eigenvalues of sums of Hermitian matrices [Klyachko 1998; Knutson–Tao, 1999]

multilinear operators are useful

higher-order derivatives

- favorite example: higher derivatives of multivariate functions
- but need a norm on $\mathrm{M}^d(\mathbb{V}_1, \dots, \mathbb{V}_d; \mathbb{W})$
- $\mathrm{M}^d(\mathbb{V}_1, \dots, \mathbb{V}_d; \mathbb{W})$ is itself a vector space
- if $\mathbb{V}_1, \dots, \mathbb{V}_d$ and $\mathbb{W}$ endowed with norms, then

$$\|\Phi\|_\sigma := \sup_{\mathbf{v}_1, \dots, \mathbf{v}_d \neq 0} \frac{\|\Phi(\mathbf{v}_1, \dots, \mathbf{v}_d)\|}{\|\mathbf{v}_1\| \cdots \|\mathbf{v}_d\|}$$

  defines a norm on $\mathrm{M}^d(\mathbb{V}_1, \dots, \mathbb{V}_d; \mathbb{W})$
- slightly abused notation: same $\|\cdot\|$ denote norms on different spaces

higher-order derivatives

- $\mathbb{V}$, $\mathbb{W}$ normed spaces; $\Omega \subseteq \mathbb{V}$ open
- derivative of $f : \Omega \to \mathbb{W}$ at $\mathbf{v} \in \Omega$ is linear operator $Df(\mathbf{v}) : \mathbb{V} \to \mathbb{W}$,

$$\lim_{\mathbf{h} \to 0} \frac{\|f(\mathbf{v}+\mathbf{h}) - f(\mathbf{v}) - [Df(\mathbf{v})](\mathbf{h})\|}{\|\mathbf{h}\|} = 0$$

- since $Df(\mathbf{v}) \in \mathsf{L}(\mathbb{V};\mathbb{W})$, apply same definition to $Df : \Omega \to \mathsf{L}(\mathbb{V};\mathbb{W})$
- get $D^2 f(\mathbf{v}) : \mathbb{V} \to \mathsf{L}(\mathbb{V};\mathbb{W})$ as $D(Df)$,

$$\lim_{\mathbf{h} \to 0} \frac{\|Df(\mathbf{v}+\mathbf{h}) - Df(\mathbf{v}) - [D^2 f(\mathbf{v})](\mathbf{h})\|}{\|\mathbf{h}\|} = 0$$

- apply recursively to get derivatives of arbitrary order

$$Df(\mathbf{v}) \in \mathsf{L}(\mathbb{V};\mathbb{W}), \qquad D^2 f(\mathbf{v}) \in \mathsf{L}\big(\mathbb{V};\mathsf{L}(\mathbb{V};\mathbb{W})\big),$$
$$D^3 f(\mathbf{v}) \in \mathsf{L}\big(\mathbb{V};\mathsf{L}(\mathbb{V};\mathsf{L}(\mathbb{V};\mathbb{W}))\big), \quad D^4 f(\mathbf{v}) \in \mathsf{L}\big(\mathbb{V};\mathsf{L}\big(\mathbb{V};\mathsf{L}(\mathbb{V};\mathsf{L}(\mathbb{V};\mathbb{W}))\big)\big)$$

higher-order derivatives

- how to avoid nested spaces of linear maps?
- use multilinear maps

$$\mathrm{L}\big(\mathbb{V}; \mathrm{M}^{d-1}(\mathbb{V}, \dots, \mathbb{V}; \mathbb{W})\big) = \mathrm{M}^{d}(\mathbb{V}, \dots, \mathbb{V}; \mathbb{W})$$

- if $\Phi : \mathbb{V} \to \mathrm{M}^{d}(\mathbb{V}, \dots, \mathbb{V}; \mathbb{W})$ linear, then

$$[\Phi(\mathbf{h})](\mathbf{h}_1, \dots, \mathbf{h}_d)$$

  linear in $\mathbf{h}$ for fixed $\mathbf{h}_1, \dots, \mathbf{h}_d$, d-linear in $\mathbf{h}_1, \dots, \mathbf{h}_d$ for fixed $\mathbf{h}$
- $D^d f(\mathbf{v}) : \mathbb{V} \times \cdots \times \mathbb{V} \to \mathbb{W}$ may be regarded as multilinear operator
- Taylor's theorem

$$f(\mathbf{v} + \mathbf{h}) = f(\mathbf{v}) + [Df(\mathbf{v})](\mathbf{h}) + \frac{1}{2}[D^2 f(\mathbf{v})](\mathbf{h}, \mathbf{h}) + \cdots \\ \cdots + \frac{1}{d!}[D^d f(\mathbf{v})](\mathbf{h}, \dots, \mathbf{h}) + R(\mathbf{h})$$

  remainder $\|R(\mathbf{h})\|/\|\mathbf{h}\|^d \to 0$ as $\mathbf{h} \to 0$

multilinear maps in computations

example: semidefinite programming

- barrier function for positive definite cone $\mathbb{S}^n_{++}$

$$f : \mathbb{S}^n_{++} \to \mathbb{R}, \qquad f(X) = -\log\det X$$

- gradient is

$$\nabla f : \mathbb{S}^n_{++} \to \mathbb{S}^n, \qquad \nabla f(X) = -X^{-1}$$

- Hessian, i.e., $\nabla^2 f := D(\nabla f)$, at any $X \in \mathbb{S}^n_{++}$ is linear operator

$$\nabla^2 f(X) : \mathbb{S}^n \to \mathbb{S}^n, \qquad H \mapsto X^{-1}HX^{-1}$$

- standard formulas useless

$$\nabla f = \begin{bmatrix} \dfrac{\partial f}{\partial x_1} \\ \vdots \\ \dfrac{\partial f}{\partial x_n} \end{bmatrix}, \qquad \nabla^2 f = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \cdots & \dfrac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \dfrac{\partial f}{\partial x_n^2} \end{bmatrix}$$

example: semidefinite programming

- writing down (hyper)matrix representations of (multi)linear maps may not be useful even when $d = 1, 2$
- in SDP, vector space is $\mathbb{S}^n$, $n \times n$ real symmetric matrices
- bad idea to identify it with $\mathbb{R}^{n(n+1)/2}$
- what about higher derivatives of $f(X) = -\log\det X$?
- formulas like

$$\nabla^d f = \left[\frac{\partial^d f}{\partial x_i \partial x_j \cdots \partial x_k} \right]_{i,j,\ldots,k=1}^{n}$$

  even less illuminating
- need to view them as multilinear maps

example: semidefinite programming

- write $F = \nabla f$, i.e., $F(X) = -X^{-1}$
- then $DF(X) = \nabla^2 f(X)$ and now we want $D^2F(X)$
- by earlier discussion, this is bilinear operator

$$D^2F(X) : \mathbb{S}^n \times \mathbb{S}^n \to \mathbb{S}^n$$

- not hard to show that it is given by

$$(H_1, H_2) \mapsto -X^{-1}H_1X^{-1}H_2X^{-1} - X^{-1}H_2X^{-1}H_1X^{-1}.$$

- dth derivative is d-linear operator

$$D^dF(X) : \mathbb{S}^n \times \cdots \times \mathbb{S}^n \to \mathbb{S}^n$$

that sends $(H_1, H_2, \ldots, H_d)$ to

$$(-1)^{d+1} \sum_{\sigma \in \mathfrak{S}_d} X^{-1}H_{\sigma(1)}X^{-1}H_{\sigma(2)}X^{-1} \cdots X^{-1}H_{\sigma(d)}X^{-1}$$

example: self-concordance

- need third derivative to check self-concordance
- convex $f : \Omega \subseteq \mathbb{R}^n \to \mathbb{R}$ self-concordant at $\mathbf{x} \in \Omega$ if

$$\left|\nabla^3 f(\mathbf{x})(\mathbf{h}, \mathbf{h}, \mathbf{h})\right| \leq 2\sigma \left|\nabla^2 f(\mathbf{x})(\mathbf{h}, \mathbf{h})\right|^{3/2}$$

  for all $\mathbf{h} \in \mathbb{R}^n$ [Nesterov–Nemirovskii, 1994]
- convex programming problem may be solved to ε-accuracy in polynomial time if it has self-concordant barrier functions,
- e.g., LP, QP, SOCP, SDP, GP

example: self-concordance

- not useful when vector space is not $\mathbb{R}^n$:

$$\nabla^2 f(x)(\mathbf{h},\mathbf{h}) = \sum_{i,j=1}^{n} \frac{\partial^2 f(x)}{\partial x_i \partial x_j} h_i h_j,$$

$$\nabla^3 f(x)(\mathbf{h},\mathbf{h},\mathbf{h}) = \sum_{i,j,k=1}^{n} \frac{\partial^3 f(x)}{\partial x_i \partial x_j \partial x_k} h_i h_j h_k$$

- to check that $f(X) = -\log\det(X)$ is self-concordant, need to show

$$\left|\mathrm{tr}\left(H^{\mathrm{T}}[\nabla^3 f(X)](H,H)\right)\right| \leq 2\sigma\left|\mathrm{tr}\left(H^{\mathrm{T}}[\nabla^2 f(X)](H)\right)\right|^{3/2}$$

- easy with multilinear map formulas

$$[\nabla^2 f(X)](H) = X^{-1}HX^{-1},$$
$$[\nabla^3 f(X)](H,H) = -2X^{-1}HX^{-1}HX^{-1}$$

acknowledgment

- Simons Foundation Award no. 663281 granted to the Institute of Mathematics of the Polish Academy of Sciences for 2021–2023
- L.-H. Lim, "Tensors in computations," *Acta Numer.*, **30** (2021), pp. 555–764