Vector Analysis in Orthogonal Curvilinear Coordinate System

A symbolic approach

Y. Ma
mym@hust.edu.cn

School of Electrical and Electronic Engineering, HUST, Wuhan

May 2, 2023

Outline

Background

- Motivations
- Mathematics

Python approach

- introduction
- Mathematics
- Python version OVA

Mathematica approach

- introduction
- modify GVA package
- *Mathematica* version OVA

Summary

References

Motivations

- In plasma physics research, vector analysis is ubiquitous but often tedious, and the cumbersomeness has even become a trait of the field.
- The automatic symbolic derivation is necessary (at least for me).
- Two symbolic approaches towards the vector analysis in general coordinate systems have been developed.
  - GVA [1] (General Vector Analysis), by Prof. Qin in 1997, a *Mathematica* package. Non-normalized basis is used, which has great generality and theoretical conciseness in any well-defined coordinates. However, we prefer to normalize the basis vectors in orthogonal curvilinear coordinates as is commonly shown in textbooks and cheatsheets.
  - SymFields [2], by Dr. Chu in 2020, a python package. It is also developed for vector analysis in the general coordinates. The package normalize basis vectors in both orthogonal and non-orthogonal coordinates, which complicates the mathematics and brings inconvenience for further developments (e.g. add more supports about 2-rank tensor related calculation).

Mathematics

Basic rules

- The metric tensor completely determines the geometric structure of the space [3].
- We usually use Einstein summation convention in the suffix notation [4, 5], a free suffix represents a equation and the summation always has one suffix up and another down (one contravariant and another covariant).

- The matric tensor $\vec{\vec{G}}$ and its coefficients g_{ik} and g^{ik}

$$
\begin{aligned}
\vec{\vec{G}} &= g_{ik}\vec{e}^i\vec{e}^k = g^{ik}\vec{e}_i\vec{e}_k \\
g_{ik} &= \vec{e}_i \cdot \vec{e}_k = g_{ki} \\
g^{ik} &= \vec{e}^i \cdot \vec{e}^k = g^{ki} \\
g_{ik} &= \frac{\partial x}{\partial x^i}\frac{\partial x}{\partial x^k} + \frac{\partial y}{\partial x^i}\frac{\partial y}{\partial x^k} + \frac{\partial z}{\partial x^i}\frac{\partial z}{\partial x^k} \\
g^{ik} &= \frac{\partial x^i}{\partial x}\frac{\partial x^k}{\partial x} + \frac{\partial x^i}{\partial y}\frac{\partial x^k}{\partial y} + \frac{\partial x^i}{\partial z}\frac{\partial x^k}{\partial z}
\end{aligned}
\tag{1}
$$

Mathematics

- The vector notation

$$
\begin{aligned}
&\vec{v} = v^i \vec{e}_i = v_i \vec{e}^i \\
&v^i = \vec{v} \cdot \vec{e}^i = g^{ik} v_k \quad v_i = \vec{v} \cdot \vec{e}_i = g_{ik} v^k \\
&\vec{e}_i = g_{ik} \vec{e}^k \quad \vec{e}^k = g^{ik} \vec{e}_i \\
&\vec{e}_i = \frac{\partial}{\partial \xi^i} = \frac{\partial}{\partial x}\frac{\partial x}{\partial \xi^i} + \frac{\partial}{\partial y}\frac{\partial y}{\partial \xi^i} + \frac{\partial}{\partial z}\frac{\partial z}{\partial \xi^i} = \hat{x}_a \frac{\partial x^a}{\partial \xi^i} = \frac{\partial \vec{R}}{\partial \xi^i} \\
&\vec{e}^i = \nabla \xi^i = \frac{\partial \xi^i}{\partial x}\vec{e}_x + \frac{\partial \xi^i}{\partial y}\vec{e}_y + \frac{\partial \xi^i}{\partial z}\vec{e}_z \\
&\vec{e}^1 = \frac{1}{V}(\vec{e}_2 \times \vec{e}_3), \quad \vec{e}^2 = \frac{1}{V}(\vec{e}_3 \times \vec{e}_1), \quad \vec{e}^3 = \frac{1}{V}(\vec{e}_1 \times \vec{e}_2) \\
&\vec{e}_1 = V(\vec{e}^2 \times \vec{e}^3), \quad \vec{e}_2 = V(\vec{e}^3 \times \vec{e}^1), \quad \vec{e}_3 = V(\vec{e}^1 \times \vec{e}^2) \\
&\delta^i_j = \vec{e}^i \cdot \vec{e}_j
\end{aligned}
\tag{2}
$$

Mathematics

- The Jacobian $\mathcal{J}$ or V (volume of the space spaned by 3 covariant basis vectors)

$$
\begin{aligned}
\mathcal{J} &= \frac{\partial(x,y,z)}{\partial(x^1,x^2,x^3)} = \begin{vmatrix} \frac{\partial x}{\partial x^1} & \frac{\partial x}{\partial x^2} & \frac{\partial x}{\partial x^3} \\ \frac{\partial y}{\partial x^1} & \frac{\partial y}{\partial x^2} & \frac{\partial y}{\partial x^3} \\ \frac{\partial z}{\partial x^1} & \frac{\partial z}{\partial x^2} & \frac{\partial z}{\partial x^3} \end{vmatrix} \\
\mathcal{J}^{-1} &= \frac{\partial(x^1,x^2,x^3)}{\partial(x,y,z)} = \begin{vmatrix} \frac{\partial x^1}{\partial x} & \frac{\partial x^1}{\partial y} & \frac{\partial x^1}{\partial z} \\ \frac{\partial x^2}{\partial x} & \frac{\partial x^2}{\partial y} & \frac{\partial x^2}{\partial z} \\ \frac{\partial x^3}{\partial x} & \frac{\partial x^3}{\partial y} & \frac{\partial x^3}{\partial z} \end{vmatrix} \\
\mathcal{J} &= V = \sqrt{\det|g_{ij}|} = \sqrt{g} = \vec{e}_1 \cdot \vec{e}_2 \times \vec{e}_3 \\
\mathcal{J}^{-1} &= \frac{1}{V} = \frac{1}{\sqrt{\det|g_{ij}|}} = \frac{1}{\sqrt{g}} = \vec{e}^1 \cdot \vec{e}^2 \times \vec{e}^3 \\
g &= \det|g_{ij}| = \mathcal{J}^2 = V^2 = \frac{g_{22}g_{33} - g_{23}^2}{g^{11}} = \frac{g_{11}g_{33} - g_{13}^2}{g^{22}} = \frac{g_{11}g_{22} - g_{12}^2}{g^{33}}
\end{aligned} \tag{3}
$$

Mathematics

- Differentiation of basis vectors

$$\nabla \cdot \vec{e}_i = \frac{1}{V}\frac{\partial V}{\partial x^i} = \frac{1}{2g}\frac{\partial g}{\partial x^i}, \quad (i = 1, 2, 3)$$

$$\nabla \times \vec{e}^i = 0$$

$$(\vec{e}_\sigma \cdot \nabla)\,\vec{e}_\rho = \frac{1}{2}g^{\mu\lambda}\left(\frac{\partial g_{\rho\lambda}}{\partial x^\sigma} + \frac{\partial g_{\sigma\lambda}}{\partial x^\rho} - \frac{\partial g_{\rho\sigma}}{\partial x^\lambda}\right)\vec{e}_\mu = \Gamma^\mu_{\rho\sigma}\vec{e}_\mu, \quad (\rho, \sigma = 1, 2, 3) \tag{4}$$

The Christoffel symbol of the first kind is the jth covariant component of the vector $\partial\vec{e}_i/\partial x^k$

$$\Gamma_{jik} = \vec{e}_j \cdot \frac{\partial \vec{e}_i}{\partial x^k} = \frac{1}{2}\left(\frac{\partial g_{ji}}{\partial x^k} + \frac{\partial g_{jk}}{\partial x^i} - \frac{\partial g_{ik}}{\partial x^j}\right) \tag{5}$$

The Christoffel symbol of the second kind is the jth contravariant component of the vector $\partial\vec{e}_i/\partial x^k$

$$\Gamma^j_{ik} = \vec{e}_j \cdot \frac{\partial \vec{e}_i}{\partial x^k} = \frac{1}{2}g^{jn}\left(\frac{\partial g_{ni}}{\partial x^k} + \frac{\partial g_{nk}}{\partial x^i} - \frac{\partial g_{ik}}{\partial x^n}\right) \tag{6}$$

$$\Gamma_{jik} = \Gamma_{jki}, \quad \Gamma^j_{ik} = \Gamma^j_{ki}, \quad \Gamma_{jik} = g_{jn}\Gamma^n_{ik}, \quad \Gamma^j_{ik} = g^{jn}\Gamma_{nik}$$

Mathematics

- Dot, cross

$$
\begin{aligned}
\vec{a}\cdot\vec{b} &= a^i b_i = a_i b^i = g_{ij} a^i b^j = g^{ij} a_i b_j \\
\vec{a}\times\vec{b} &= \epsilon^{ijk} V a^i b^j \vec{e}^k = \epsilon_{ijk} \frac{1}{V} a_i b_j \vec{e}_k
\end{aligned} \tag{7}
$$

- Gradient

$$
\begin{aligned}
\nabla &= \vec{e}^i \frac{\partial}{\partial x^i} \\
\nabla\phi &= \frac{\partial \phi}{\partial x^i} \vec{e}^i \\
\nabla\vec{f} &= f_{\rho;\sigma} \vec{e}^\sigma \vec{e}^\rho \\
f_{\rho;\sigma} &= \vec{e}_\rho \vec{e}_\sigma : \nabla\vec{f} = \frac{\partial f_\rho}{\partial x^\sigma} - \Gamma^\mu_{\rho\sigma} f_\mu \\
\nabla\nabla\vec{f} &= f_{\rho;\sigma;\mu} \vec{e}^\mu \vec{e}^\sigma \vec{e}^\rho \\
f_{\rho;\sigma;\mu} &= \frac{\partial f_{\rho;\sigma}}{\partial x^\mu} - \Gamma^\lambda_{\rho\mu} f_{\lambda;\sigma} - \Gamma^\lambda_{\sigma\mu} f_{\rho;\lambda} \\
\vec{a}\cdot\nabla\vec{b} &= \left(a^i \frac{\partial b^j}{\partial x^i} + a^i b^k \Gamma^j_{ki} \right) \vec{e}_k
\end{aligned} \tag{8}
$$

Mathematics

- Divergence

$$\nabla \cdot \vec{f} = \nabla \cdot \left(f^i \vec{e}_i \right) = \vec{e}_i \cdot \nabla f^i + f^i \nabla \cdot \vec{e}_i = \frac{\partial f^i}{\partial x^i} + \frac{f^i}{2g} \frac{\partial g}{\partial x^i} = \frac{1}{\sqrt{g}} \frac{\partial}{\partial x^i} \left(\sqrt{g} f^i \right) \tag{9}$$

- Curl

$$\nabla \times \vec{f} = \frac{1}{\sqrt{g}} \frac{\partial f_i}{\partial x^j} \varepsilon_{kji} \vec{e}_k$$

$$\nabla \times \vec{f} = \frac{1}{\sqrt{g}} \left\{ \left(\frac{\partial f_3}{\partial x^2} - \frac{\partial f_2}{\partial x^3} \right) \vec{e}_1 + \left(\frac{\partial f_1}{\partial x^3} - \frac{\partial f_3}{\partial x^1} \right) \vec{e}_2 + \left(\frac{\partial f_2}{\partial x^1} - \frac{\partial f_1}{\partial x^2} \right) \vec{e}_3 \right\} \tag{10}$$

- Differential of position vector

$$\begin{gathered} d\vec{r} = \frac{\partial \vec{r}}{\partial x^i} dx^i = dx^i \vec{e}_i \\ d\vec{r} = dx_k \vec{e}^k = g_{ik} dx^i \vec{e}^k \\ dx_k = g_{ik} dx^i \quad dx^i = g^{ik} dx_k \\ dx^i = \vec{e}^i \cdot d\vec{r} = \vec{e}^i \cdot \vec{e}_j dx^j \end{gathered} \tag{11}$$

Outline

Background
- Motivations
- Mathematics

Python approach
- introduction
- Mathematics
- Python version OVA

Mathematica approach
- introduction
- modify GVA package
- *Mathematica* version OVA

Summary

References

Python approach

Python can do symbolic calculations using the package sympy [6].

- Open source.
- Can run in the jupyter notebook interactively, which supports multiple output forms.
- Symbolic calculation ability cannot surpass that of *Mathematica* due to the procedural programming nature of the Python.

The software package OVA (Orthogonal curvilinear coordinate Vector Analysis) of Python version has been developed.

- OVA supports vector analysis in any well-defined right-handed orthogonal coordinate system.
- The 'Cartesian', 'Cylindrical' and 'Spherical' coordinates are built into OVA. The user can define a new coordinate system easily by specifying the corresponding covariant metric matrix.
- Use formulas shown thereafter that can only do analysis in the orthogonal coordinate system.

Python approach

Basic rules

In an orthogonal coordinate system,

- the metric tensor is a diagonal matrix;
- the basis vectors are mutually perpendicular, and the three covariant basis vectors are just parallel to their corresponding three contravariant basis vectors. For this reason, we often normalize basis vectors using Lamé coefficients, which ensures the normalized covariant vectors coincide with their corresponding contravariant basis vectors.

Mathematics

- Vector notation

$$\begin{aligned} \vec{f} &= F_i\hat{e}_i, \quad F_i = \hat{e}_i \cdot \vec{f} = f_i/h_i = h_i f^i, \quad (i = 1,2,3) \\ \overleftrightarrow{K} &= K_{ij}\hat{e}_i\hat{e}_j, \quad K_{ij} = k_{ij}/\left(h_i h_j\right) = h_i h_j k^{ij}, \quad (i,j = 1,2,3) \end{aligned} \tag{12}$$

- Lamé coefficients

$$\begin{aligned} h_1 &= \sqrt{g_{11}} = \frac{1}{\sqrt{g^{11}}}, \quad h_2 = \sqrt{g_{22}} = \frac{1}{\sqrt{g^{22}}}, \quad h_3 = \sqrt{g_{33}} = \frac{1}{\sqrt{g^{33}}} \\ h_i &= |\vec{e}_i| = \left[\left(\frac{\partial x}{\partial x^i}\right)^2 + \left(\frac{\partial y}{\partial x^i}\right)^2 + \left(\frac{\partial z}{\partial x^i}\right)^2\right]^{1/2} \\ &= \left[\left(\frac{\partial x^i}{\partial x}\right)^2 + \left(\frac{\partial x^i}{\partial y}\right)^2 + \left(\frac{\partial x^i}{\partial z}\right)^2\right]^{-1/2}, \quad (i = 1,2,3) \\ \mathcal{J} &= V = \sqrt{g} = h_1 h_2 h_3 \\ \hat{e}_i &= \frac{\vec{e}_i}{h_i} = h_i\vec{e}^i, \quad F_i = F^i = \frac{f_i}{h_i} = f^i h_i \end{aligned} \tag{13}$$

Mathematics

- Lamé coefficients [3] - Cont'd

Cylindrical coordinate				Spherical coordinate			
coordinate	r	θ	z	coordinate	r	θ	φ
Lamé	1	r	1	Lamé	1	r	$r\sin\theta$
$\partial/\partial r$	0	1	0	$\partial/\partial r$	0	1	$\sin\theta$
$\partial/\partial\theta$	0	0	0	$\partial/\partial\theta$	0	0	$r\cos\theta$
$\partial/\partial z$	0	0	0	$\partial/\partial\varphi$	0	0	0
$\hat{e}_i$	$\hat{e}_r$	$\hat{e}_\theta$	$\hat{e}_z$	$\hat{e}_i$	$\hat{e}_r$	$\hat{e}_\theta$	$\hat{e}_\varphi$
$\partial/\partial r$	0	0	0	$\partial/\partial r$	0	0	0
$\partial/\partial\theta$	$\hat{e}_\theta$	$-\hat{e}_r$	0	$\partial/\partial\theta$	$\hat{e}_\theta$	$-\hat{e}_r$	0
$\partial/\partial z$	0	0	0	$\partial/\partial\varphi$	$\sin\theta\hat{e}_\varphi$	$\cos\theta\hat{e}_\varphi$	$-\sin\theta\hat{e}_r - \cos\theta\hat{e}_\theta$

- Lengths

$$
\begin{aligned}
(ds)^2 &= \left(h_1 dx^1\right)^2 + \left(h_2 dx^2\right)^2 + \left(h_3 dx^3\right)^2 \\
ds_1 &= h_1 dx^1 \hat{e}_1, \quad ds_2 = h_2 dx^2 \hat{e}_2, \quad ds_3 = h_3 dx^3 \hat{e}_3 \\
da_1 &= h_2 h_3 dx^2 dx^3 \hat{e}_1 \quad da_2 = h_1 h_3 dx^1 dx^3 \hat{e}_2 \quad da_3 = h_1 h_2 dx^1 dx^2 \hat{e}_3 \\
d\tau &= V dx^1 dx^2 dx^3 = h_1 h_2 h_3 dx^1 dx^2 dx^3
\end{aligned} \tag{14}
$$

Mathematics

- Differentiation of basis vectors

$$
\begin{aligned}
\nabla \cdot \hat{e}_i &= \frac{1}{Vh_i}\frac{\partial V}{\partial x_i} - \frac{1}{h_i^2}\frac{\partial h_i}{\partial x_i} = \frac{1}{V}\frac{\partial}{\partial x_i}\left(\frac{V}{h_i}\right), \quad (i = 1, 2, 3) \\
\nabla \times \hat{e}_i &= -h_i \nabla\left(\frac{1}{h_i}\right) \times \hat{e}_i = \frac{1}{h_i}\sum_{j=1}^{3}\left(\frac{1}{h_j}\frac{\partial h_i}{\partial x_j}\hat{e}_j\right) \times \hat{e}_i \\
\left(\hat{e}_k \cdot \nabla\right)\hat{e}_i &= -\frac{1}{h_k}\left(\nabla h_i\right)\delta_k^i + \frac{1}{h_i h_k}\frac{\partial h_k}{\partial x_i}\hat{e}_k, \quad (i, k = 1, 2, 3) \\
\frac{\partial \hat{e}_i}{\partial x_k} &= -\left(\nabla h_i\right)\delta_k^i + \frac{1}{h_i}\frac{\partial h_k}{\partial x_i}\hat{e}_k \quad (i, k = 1, 2, 3)
\end{aligned}
\tag{15}
$$

Mathematics

- Differentiation of basis vectors-Cont'd

$$\begin{cases} (\hat{e}_1 \cdot \nabla)\,\hat{e}_1 = -\frac{1}{h_1}\nabla h_1 + \frac{1}{h_1^2}\frac{\partial h_1}{\partial x_1}\hat{e}_1 = -\frac{1}{h_1 h_2}\frac{\partial h_1}{\partial x_2}\hat{e}_2 - \frac{1}{h_1 h_3}\frac{\partial h_1}{\partial x_3}\hat{e}_3, \\ (\hat{e}_2 \cdot \nabla)\,\hat{e}_2 = -\frac{1}{h_2}\nabla h_2 + \frac{1}{h_2^2}\frac{\partial h_2}{\partial x_2}\hat{e}_2 = -\frac{1}{h_1 h_2}\frac{\partial h_2}{\partial x_1}\hat{e}_1 - \frac{1}{h_2 h_3}\frac{\partial h_2}{\partial x_3}\hat{e}_3, \\ (\hat{e}_3 \cdot \nabla)\,\hat{e}_3 = -\frac{1}{h_3}\nabla h_3 + \frac{1}{h_3^2}\frac{\partial h_3}{\partial x_3}\hat{e}_3 = -\frac{1}{h_1 h_3}\frac{\partial h_3}{\partial x_1}\hat{e}_1 - \frac{1}{h_2 h_3}\frac{\partial h_3}{\partial x_2}\hat{e}_2. \end{cases} \tag{16}$$

$$\begin{cases} \frac{\partial \hat{e}_1}{\partial x_1} = -\nabla h_1 + \frac{1}{h_1}\frac{\partial h_1}{\partial x_1}\hat{e}_1 = -\frac{1}{h_2}\frac{\partial h_1}{\partial x_2}\hat{e}_2 - \frac{1}{h_3}\frac{\partial h_1}{\partial x_3}\hat{e}_3 \\ \frac{\partial \hat{e}_2}{\partial x_2} = -\nabla h_2 + \frac{1}{h_2}\frac{\partial h_2}{\partial x_2}\hat{e}_2 = -\frac{1}{h_1}\frac{\partial h_2}{\partial x_1}\hat{e}_1 - \frac{1}{h_3}\frac{\partial h_2}{\partial x_3}\hat{e}_3 \\ \frac{\partial \hat{e}_3}{\partial x_3} = -\nabla h_3 + \frac{1}{h_3}\frac{\partial h_3}{\partial x_3}\hat{e}_3 = -\frac{1}{h_1}\frac{\partial h_3}{\partial x_1}\hat{e}_1 - \frac{1}{h_2}\frac{\partial h_3}{\partial x_2}\hat{e}_2 \end{cases} \tag{17}$$

Mathematics

- Dot and cross product

$$
\begin{aligned}
\vec{a} \cdot \vec{b} &= a^i b_i = \frac{A_i}{h_i} B_i h_i = A_i B_i \\
\vec{a} \cdot \vec{\vec{T}} &= A_k \hat{e}_k \cdot T_{ij} \hat{e}_i \hat{e}_j = A_i T_{ij} \hat{e}_j \\
\vec{\vec{T}} \cdot \vec{a} &= T_{ij} \hat{e}_i \hat{e}_j \cdot a_k \hat{e}_k = A_j T_{ij} \hat{e}_i \\
\vec{a} \times \vec{b} &= A_i B_j \hat{e}_i \times \hat{e}_j = \epsilon_{ijk} A_i B_j \hat{e}_k
\end{aligned}
\tag{18}
$$

Mathematics

- Gradient

$$\nabla\phi = \frac{\partial\phi}{\partial x^i}\vec{e}^i = \frac{1}{h_i}\frac{\partial\phi}{\partial x_i}\hat{e}_i \tag{19}$$

$$\hat{e}_i \cdot \nabla\phi = \frac{1}{h_i}\frac{\partial\phi}{\partial x_i} \tag{20}$$

$$\begin{aligned}\nabla\vec{u} = \hat{e}_i\hat{e}_j\frac{1}{h_i}\frac{\partial U_j}{\partial x^i} + \hat{e}_i\frac{\partial\hat{e}_j}{\partial x^i}\frac{U_j}{h_i} &= \hat{e}_i\hat{e}_j\frac{1}{h_i}\frac{\partial U_j}{\partial x^i} + \hat{e}_i\frac{U_j}{h_i}\left(-(\nabla h_j)\delta_i^j + \frac{1}{h_j}\frac{\partial h_i}{\partial x_j}\hat{e}_i\right) \\ &= \hat{e}_i\hat{e}_j\frac{1}{h_i}\frac{\partial U_j}{\partial x^i} - \hat{e}_i\hat{e}_k\frac{U_i}{h_i}\frac{1}{h_k}\frac{\partial h_i}{\partial x_k} + \hat{e}_i\hat{e}_i\frac{U_j}{h_i}\frac{1}{h_j}\frac{\partial h_i}{\partial x_j}\end{aligned} \tag{21}$$

Mathematics

- Div

$$\nabla \cdot \vec{f} = \frac{1}{\sqrt{g}} \frac{\partial}{\partial x^i} \left(\sqrt{g} f^i \right) = \frac{1}{V} \sum_{i=1}^{3} \frac{\partial}{\partial x_i} \left(\frac{V F_i}{h_i} \right) \tag{22}$$

$$\nabla^2 \phi = \frac{1}{V} \frac{\partial}{\partial x_i} \left(\frac{V}{h_i^2} \frac{\partial \phi}{\partial x_i} \right) \tag{23}$$

$$\begin{aligned}
\nabla \cdot \vec{\vec{T}} &= \frac{1}{h_i} \frac{\partial T_{ik}}{\partial x_i} \hat{e}_k + \frac{T_{ik}}{V} \frac{\partial (\frac{V}{h_i})}{\partial x_i} \hat{e}_k + \frac{T_{ik}}{h_i} \frac{\partial \hat{e}_k}{\partial x_i} \\
&= \frac{1}{h_i} \frac{\partial T_{ik}}{\partial x_i} \hat{e}_k + \frac{T_{ik}}{V} \frac{\partial (\frac{V}{h_i})}{\partial x_i} \hat{e}_k + \frac{T_{ik}}{h_i} \left[-(\nabla h_k) \delta_i^k + \frac{1}{h_k} \frac{\partial h_i}{\partial x_k} \hat{e}_i \right] \\
&= \frac{1}{h_i} \frac{\partial T_{ik}}{\partial x_i} \hat{e}_k + \frac{T_{ik}}{V} \frac{\partial (\frac{V}{h_i})}{\partial x_i} \hat{e}_k + \frac{T_{ik}}{h_i} \frac{1}{h_k} \frac{\partial h_i}{\partial x_k} \hat{e}_i - \frac{T_{ii}}{h_i} \frac{1}{h_j} \frac{\partial h_i}{\partial x_j} \hat{e}_j
\end{aligned} \tag{24}$$

Mathematics

- Curl

$$\nabla \times \vec{f} = \frac{1}{\sqrt{g}} \frac{\partial f_i}{\partial x_j} \varepsilon_{kji} \vec{e}_k = \sum_{i,j,k=1}^{3} \frac{h_k}{V} \frac{\partial \left(h_i F_i \right)}{\partial x_j} \varepsilon_{kji} \hat{e}_k \tag{25}$$

$$\begin{aligned} \nabla \times \vec{f} = & \frac{1}{h_2 h_3} \left[\frac{\partial \left(h_3 F_3 \right)}{\partial x_2} - \frac{\partial \left(h_2 F_2 \right)}{\partial x_3} \right] \hat{e}_1 + \\ & \frac{1}{h_1 h_3} \left[\frac{\partial \left(h_1 F_1 \right)}{\partial x_3} - \frac{\partial \left(h_3 F_3 \right)}{\partial x_1} \right] \hat{e}_2 + \\ & \frac{1}{h_1 h_2} \left[\frac{\partial \left(h_2 F_2 \right)}{\partial x_1} - \frac{\partial \left(h_1 F_1 \right)}{\partial x_2} \right] \hat{e}_3 \end{aligned} \tag{26}$$

Mathematics

- Laplacian

$$\nabla^2\phi = \frac{1}{\sqrt{g}}\frac{\partial}{\partial x^j}\left(g^{ij}\sqrt{g}\frac{\partial\phi}{\partial x^i}\right) = \frac{1}{V}\sum_{i=1}^{3}\frac{\partial}{\partial x_i}\left(\frac{V}{h_i^2}\frac{\partial\phi}{\partial x_i}\right) \tag{27}$$

$$\nabla^2\phi = \frac{1}{h_1h_2h_3}\left\{\frac{\partial}{\partial x_1}\left(\frac{h_2h_3}{h_1}\frac{\partial\phi}{\partial x_1}\right) + \frac{\partial}{\partial x_2}\left(\frac{h_1h_3}{h_2}\frac{\partial\phi}{\partial x_2}\right) + \frac{\partial}{\partial x_3}\left(\frac{h_1h_2}{h_3}\frac{\partial\phi}{\partial x_3}\right)\right. \tag{28}$$

Python version OVA

- The Python version OVA was implemented using the normalized vector notations shown before.

```
In [1]: import sys
        sys.path.append("../")
        from OVA import *
        import sympy as sym
        sym.init_printing()
```

```
In [2]: r = sym.Symbol('r', positive=True)
        t, z = sym.symbols('theta, z')
        cylind = CoordinateSystem(r=r, t=t, p=z, coordinate='Cylindrical')
```

```
In [3]: # vector U
        Ur = sym.Function('U_r')(r, t, z); Ut = sym.Function('U_theta')(r, t, z); Uz = sym.Function('U_z')(r, t, z);
        U = [Ur, Ut, Uz]
        # scalar function phi
        phi = sym.Function('phi')(r, t, z)
        # 2-rank tensor T
        Trr = sym.Function('T_{rr}')(r, t, z); Trt = sym.Function('T_{rt}')(r, t, z); Trz = sym.Function('T_{rz}')(r, t, z);
        Ttr = sym.Function('T_{tr}')(r, t, z); Ttt = sym.Function('T_{tt}')(r, t, z); Ttz = sym.Function('T_{tz}')(r, t, z);
        Tzr = sym.Function('T_{zr}')(r, t, z); Tzt = sym.Function('T_{zt}')(r, t, z); Tzz = sym.Function('T_{zz}')(r, t, z);
        T = sym.Matrix([[Trr, Trt, Trz], [Ttr, Ttt, Ttz], [Tzr, Tzt, Tzz]])
        display(U)
        display(phi)
        display(T)
```

$$[\mathrm{U_r}(r,\theta,z),\ \mathrm{U_\theta}(r,\theta,z),\ \mathrm{U_z}(r,\theta,z)]$$

$$\phi(r,\theta,z)$$

$$\begin{bmatrix} \mathrm{T_{rr}}(r,\theta,z) & \mathrm{T_{rt}}(r,\theta,z) & \mathrm{T_{rz}}(r,\theta,z) \\ \mathrm{T_{tr}}(r,\theta,z) & \mathrm{T_{tt}}(r,\theta,z) & \mathrm{T_{tz}}(r,\theta,z) \\ \mathrm{T_{zr}}(r,\theta,z) & \mathrm{T_{zt}}(r,\theta,z) & \mathrm{T_{zz}}(r,\theta,z) \end{bmatrix}$$

Figure: Define coordinate system and variables

Python version OVA

- Parts of the benchmarks are shown here.

Grad of function $\phi(r,\theta,z)$:

$$\nabla\phi = \hat{e}_r\frac{\partial\phi}{\partial r} + \hat{e}_\theta\frac{1}{r}\frac{\partial\phi}{\partial\theta} + \hat{e}_z\frac{\partial\phi}{\partial z}$$

In [4]:
```
cylind.grad(phi)
```

Out[4]:

$$\left[\frac{\partial}{\partial r}\phi(r,\theta,z),\ \frac{\frac{\partial}{\partial\theta}\phi(r,\theta,z)}{r},\ \frac{\partial}{\partial z}\phi(r,\theta,z)\right]$$

Grad of vector $\vec{u}$:

$$\begin{aligned}\nabla\vec{u} = {} & \frac{\partial U_r}{\partial r}\hat{e}_r\hat{e}_r + \frac{\partial U_\theta}{\partial r}\hat{e}_r\hat{e}_\theta + \frac{\partial U_z}{\partial r}\hat{e}_r\hat{e}_z \\ & + \left(\frac{1}{r}\frac{\partial U_r}{\partial\theta} - \frac{U_\theta}{r}\right)\hat{e}_\theta\hat{e}_r + \left(\frac{1}{r}\frac{\partial U_\theta}{\partial\theta} + \frac{U_r}{r}\right)\hat{e}_\theta\hat{e}_\theta + \frac{1}{r}\frac{\partial U_z}{\partial\theta}\hat{e}_\theta\hat{e}_z \\ & + \frac{\partial U_r}{\partial z}\hat{e}_z\hat{e}_r + \frac{\partial U_\theta}{\partial z}\hat{e}_z\hat{e}_\theta + \frac{\partial U_z}{\partial z}\hat{e}_z\hat{e}_z\end{aligned}$$

In [5]:
```
cylind.grad(U)
```

Out[5]:

$$\begin{bmatrix} \frac{\partial}{\partial r}\mathrm{U_r}(r,\theta,z) & \frac{\partial}{\partial r}\mathrm{U_\theta}(r,\theta,z) & \frac{\partial}{\partial r}\mathrm{U_z}(r,\theta,z) \\ \frac{-\mathrm{U_\theta}(r,\theta,z)+\frac{\partial}{\partial\theta}\mathrm{U_r}(r,\theta,z)}{r} & \frac{\frac{\partial}{\partial\theta}\mathrm{U_\theta}(r,\theta,z)}{r} & \frac{\frac{\partial}{\partial\theta}\mathrm{U_z}(r,\theta,z)}{r} \\ \frac{\partial}{\partial z}\mathrm{U_r}(r,\theta,z) & \frac{\partial}{\partial z}\mathrm{U_\theta}(r,\theta,z) & \frac{\partial}{\partial z}\mathrm{U_z}(r,\theta,z) \end{bmatrix}$$

Figure: Benchmark gradient operator

Python version OVA

Vector $\vec{u}$ dot the grad of $\vec{u}$:

$$\vec{u}\cdot\nabla\vec{u} = \left[U_r\frac{\partial U_r}{\partial r} + U_\theta\left(\frac{1}{r}\frac{\partial U_r}{\partial\theta} - \frac{U_\theta}{r}\right) + U_z\frac{\partial U_r}{\partial z}\right]\hat{e}_r + \left[U_r\frac{\partial U_\theta}{\partial r} + U_\theta\left(\frac{1}{r}\frac{\partial U_\theta}{\partial\theta} + \frac{U_r}{r}\right) + U_z\frac{\partial U_\theta}{\partial z}\right]\hat{e}_\theta + \left[U_r\frac{\partial U_z}{\partial r} + U_\theta\frac{1}{r}\frac{\partial U_z}{\partial\theta} + U_z\frac{\partial U_z}{\partial z}\right]\hat{e}_z$$

In [6]:
```
cylind.dot(U, cylind.grad(U))
```

Out[6]:

$$\left[\frac{r\,U_r(r,\theta,z)\frac{\partial}{\partial r}U_r(r,\theta,z) + r\,U_z(r,\theta,z)\frac{\partial}{\partial z}U_r(r,\theta,z) - \left(U_\theta(r,\theta,z) - \frac{\partial}{\partial\theta}U_r(r,\theta,z)\right)U_\theta(r,\theta,z)}{r},\ U_r(r,\theta,z)\frac{\partial}{\partial r}U_\theta(r,\theta,z) + U_z(r,\theta,z)\frac{\partial}{\partial z}U_\theta(r,\theta,z) + \frac{U_\theta(r,\theta,z)\frac{\partial}{\partial\theta}U_\theta(r,\theta,z)}{r},\ U_r(r,\theta,z)\frac{\partial}{\partial r}U_z(r,\theta,z) + U_z(r,\theta,z)\frac{\partial}{\partial z}U_z(r,\theta,z) + \frac{U_\theta(r,\theta,z)\frac{\partial}{\partial\theta}U_z(r,\theta,z)}{r}\right]$$

Figure: Benchmark covariant derivative

Divergence of the tensor $\overleftrightarrow{T}$

$$\nabla\cdot\overleftrightarrow{T} = \left[\frac{1}{r}\frac{\partial}{\partial r}(rT_{rr}) + \frac{1}{r}\frac{\partial T_{\theta r}}{\partial\theta} + \frac{\partial T_{zr}}{\partial z} - \frac{T_{\theta\theta}}{r}\right]\hat{e}_r + \left[\frac{1}{r^2}\frac{\partial}{\partial r}(r^2T_{r\theta}) + \frac{1}{r}\frac{\partial T_{\theta\theta}}{\partial\theta} + \frac{\partial T_{z\theta}}{\partial z} - \frac{T_{r\theta} - T_{\theta r}}{r}\right]\hat{e}_\theta + \left[\frac{1}{r}\frac{\partial}{\partial r}(rT_{rz}) + \frac{1}{r}\frac{\partial T_{\theta z}}{\partial\theta} + \frac{\partial T_{zz}}{\partial z}\right]\hat{e}_z$$

In [8]:
```
cylind.div(T)
```

Out[8]:

$$\left[\frac{\partial}{\partial r}T_{rr}(r,\theta,z) + \frac{\partial}{\partial z}T_{zr}(r,\theta,z) + \frac{T_{rr}(r,\theta,z)}{r} - \frac{T_{tt}(r,\theta,z)}{r} + \frac{\frac{\partial}{\partial\theta}T_{tr}(r,\theta,z)}{r},\ \frac{\partial}{\partial r}T_{rt}(r,\theta,z) + \frac{\partial}{\partial z}T_{zt}(r,\theta,z) + \frac{T_{rt}(r,\theta,z)}{r} + \frac{T_{tr}(r,\theta,z)}{r} + \frac{\frac{\partial}{\partial\theta}T_{tt}(r,\theta,z)}{r},\ \frac{\partial}{\partial r}T_{rz}(r,\theta,z) + \frac{\partial}{\partial z}T_{zz}(r,\theta,z) + \frac{T_{rz}(r,\theta,z)}{r} + \frac{\frac{\partial}{\partial\theta}T_{tz}(r,\theta,z)}{r}\right]$$

Figure: Benchmark divergence of a tensor

Python version OVA

$$\vec{A}\times(\nabla\times\vec{B})-\left[\nabla\vec{B}\cdot\vec{A}-(\vec{A}\cdot\nabla)\vec{B}\right]=0$$

In [7]:
```
temp_1 = sym.Array(coord.cross(A, coord.curl(B)))
temp_2 = sym.Array(coord.dot(coord.grad(B), A))
temp_3 = sym.Array(coord.dot(A, coord.grad(B)))
res = temp_1 - (temp_2 - temp_3)
res.applyfunc(sym.simplify)
```

Out[7]: $\begin{bmatrix}0 & 0 & 0\end{bmatrix}$

$$\nabla\cdot(\phi\vec{A})-(\phi\nabla\cdot\vec{A}+\vec{A}\cdot\nabla\phi)=0$$

In [8]:
```
temp_pA = [phi * k for k in A]   # phi * \vec{A}
sym.simplify(coord.div(temp_pA) - (phi * coord.div(A) + coord.dot(A, coord.grad(phi))))
```

Out[8]: 0

$$\nabla\times(\phi\vec{A})-(\phi\nabla\times\vec{A}+\nabla\phi\times\vec{A})=0$$

In [9]:
```
temp_pA = [phi * k for k in A]
temp_1 = sym.Array(coord.curl(temp_pA))
temp_2 = sym.Array([phi * k for k in (coord.curl(A))])
temp_3 = sym.Array(coord.cross(coord.grad(phi), A))
sym.simplify(temp_1 - (temp_2 + temp_3))
```

Out[9]: $\begin{bmatrix}0 & 0 & 0\end{bmatrix}$

Figure: Benchmark vector identities

Outline

Background
- Motivations
- Mathematics

Python approach
- introduction
- Mathematics
- Python version OVA

Mathematica approach
- introduction
- modify GVA package
- *Mathematica* version OVA

Summary

References

introduction

- *Mathematica* has (the most?) powerful symbolic calculation capability.
  - *Mathematica* is a functional programming language [7] in which there is no distinction between functions and data.
  - The rule-based programming makes the *Mathematica* highly suitable for the symbolic calculation.
  - Everything in *Mathematica* is an expression.
  - Every expressions can be decomposed into atoms (symbols, numbers, strings) and the rules attached to symbols (DownValues, UpValues, OwnValues, SubValues, NValues, FormatValues).
  - To define our own vector analysis package, we just attach new rules to symbols 'div', 'curl', '∇' and so on.

introduction

- First, we modified the GVA package of version 1.0 by Prof. Qin in 1997 to ensure its compatibility with *Mathematica* versions newer than 8.0, eliminating any associated warnings.
  - The Version 1.0 GVA and the popular current version of Mathematica ($>$8.0) have some non-essential conflicts, such as conflicts between GVA symbols and built-in symbols, which results in warnings when used.
  - The updates can be classified into two categories: 1. all functions in the package now have the Protected attribute and begin with lowercase letters to avoid conflicts with symbols in Symstem‘ context. 2. Modifications in vectorNotation[] function are implemented to resolve issues with interpreting some of the notations such as $\nabla\times$ and $\nabla\cdot$.
- Second, our *Mathematica* version OVA package is implemented based on the modified GVA package.

modify GVA package

```
In[•]:= SetDirectory[NotebookDirectory[]];
        Needs["Calculus`GeneralVectorAnalysis`", "GeneralVectorAnalysis.m"]
        Begin["Calculus`GeneralVectorAnalysis`"]

Jacobian: Symbol Jacobian appears in multiple contexts {Calculus`GeneralVectorAnalysis`, System`}; definitions in context Calculus`GeneralVectorAnalysis` may shadow or be shadowed by other definitions.
Grad: Symbol Grad appears in multiple contexts {Calculus`GeneralVectorAnalysis`, System`}; definitions in context Calculus`GeneralVectorAnalysis` may shadow or be shadowed by other definitions.
Div: Symbol Div appears in multiple contexts {Calculus`GeneralVectorAnalysis`, System`}; definitions in context Calculus`GeneralVectorAnalysis` may shadow or be shadowed by other definitions.
Curl: Symbol Curl appears in multiple contexts {Calculus`GeneralVectorAnalysis`, System`}; definitions in context Calculus`GeneralVectorAnalysis` may shadow or be shadowed by other definitions.
Laplacian: Symbol Laplacian appears in multiple contexts {Calculus`GeneralVectorAnalysis`, System`}; definitions in context Calculus`GeneralVectorAnalysis` may shadow or be shadowed by other definitions.
UnitVector: Symbol UnitVector appears in multiple contexts {Calculus`GeneralVectorAnalysis`, System`}; definitions in context Calculus`GeneralVectorAnalysis` may shadow or be shadowed by other definitions.
VectorQ: Symbol VectorQ appears in multiple contexts {Calculus`GeneralVectorAnalysis`, System`}; definitions in context Calculus`GeneralVectorAnalysis` may shadow or be shadowed by other definitions.
The Vector Calculus on General Coordinate is loaded in
Use SetCoordinateSystem[ ] to set up a coordinate system
The default CoordinateSystem is  None
... Decrement: End is not a variable with a value, so its value cannot be changed.
... PreDecrement: End-- is not a variable with a value, so its value cannot be changed.

Out[•]= --(End--)

Out[•]= Calculus`GeneralVectorAnalysis`
```

Figure: Warnings occur when importing the GVA version 1.0

- Solution: all functions in the package now have the Protected attribute and begin with lowercase letters to avoid conflicts with symbols in Symstem‘ context.

modify GVA package

```
In[•]:= FullForm /@ {a · (b + c), ∇ × (a + a × (b × c)), ∇ · (a + b × d), (a · ∇) b, ∇² f ∇ · a, â, | a |, 𝒥[a]}
  ··· System`Curl: ∇ × (a + a × (b × c)) cannot be interpreted. In a curl, the del operator requires a subscript with differentiation variables.
In[•]:= FullForm /@ {a · b, a × b, ∇f, (a · ∇) b, ∇² a}
Out[•]= {DotProduct[a, b], CrossProduct[a, b], Grad[f], CovariantDerivative[a, b], Laplacian[a]}

In[•]:= FullForm@(∇ · a)
  ··· General: ∇ · a cannot be interpreted. The prefix operator ∇ cannot be combined with the binary operator ·.
In[•]:= FullForm@(∇ × a)
  ··· System`Curl: ∇ × a cannot be interpreted. In a curl, the del operator requires a subscript with differentiation variables.
In[•]:= Div[Curl[a]] // VectorExpand
Out[•]= ∇ · (∇ × a)
```

Figure: Errors occur when using some of the operator notations

- Solution: modifications in Calculus‘GeneralVectorAnalysis‘vectorNotation[] function are implemented to resolve issues with interpreting some of the notations such as $\nabla\times$ and $\nabla\cdot$.

Mathematica version OVA

- The *Mathematica* version OVA package is implemented based on the modified GVA package.
  - Vectors have 3 components with normalized basis vectors now (6 components before).
  - Every covariant and contravariant component is normalized using the Lamé coefficients.

$$
\begin{aligned}
\vec{f} = F_i \hat{e}_i, &\quad F_i = \hat{e}_i \cdot \vec{f} \\
f_i = F_i h_i, &\quad f^i = \frac{F_i}{h_i}
\end{aligned}
\tag{29}
$$

Mathematica version OVA: benchmark

```
In[1]:= SetDirectory[NotebookDirectory[]];
        Needs["OVA`", "OVA.m"]
        Begin["OVA`"]

        The Vector Calculus on Orthogonal Coordinate is loaded in
        Use SetCoordinateSystem[ ] to set up a coordinate system
        The default CoordinateSystem is  None

Out[3]= OVA`

In[4]:= (*basic usage*)

In[5]:= FullForm /@ {a · (b + c), ∇ × (a + a × (b × c)), ∇ · (a + b × d), (a · ∇) b, ∇²f ∇ · a, â, |a|, 𝒥[a]}

Out[5]= {dot[a, Plus[b, c]], curl[Plus[a, cross[a, cross[b, c]]]], div[Plus[a, cross[b, d]]],
         covariantDerivative[a, b], Times[div[a], laplacian[f]], unitVector[a], absoluteValue[a], jacobian[a]}

In[6]:= declareVector[A, B0, B, C, D, F, G, H, J, ξ, Q]
        declareScalar[a, b, c, d, e, f, g, h]

        {A, B0, B, C, D, F, G, H, J, ξ, Q} are Vectors now.
        {a, b, c, d, e, f, g, h} are Scalars now.

In[8]:= {A × A, A · (A × C), ∇ × (∇f), ∇ · (∇ × A), vectorExpand[A × (B × C)]}

Out[8]= {0, 0, 0, 0, -C A · B + B A · C}

In[9]:= vectorExpand /@ {A × (B × C) + B × (C × A) + C × (A × B), (A × B) × (C × D), ∇ · (f (A × B + ∇ × C))}

Out[9]= {0, -D A · (B × C) + C A · (B × D), -f A · (∇ × B) + f B · (∇ × A) + (A × B) · (∇f) + (∇ × C) · (∇f)}

In[10]:= (*an example from the ideal MHD*)
         Unprotect[div]; ∇ · (B0) = 0; Protect[div]; Q = ∇ × (ξ × B0); J = ∇ × Q

Out[10]= -∇ × (∇ × (B0 × ξ))

In[11]:= vectorExpand[J]

Out[11]= B0 × (∇ (∇ · ξ)) + ∇ × ((B0 · ∇) ξ) - ∇ × ((ξ · ∇) B0) - ∇ × B0 ∇ · ξ
```

Figure: The coordinate-independent vector analysis provided by OVA here is equivalent to GVA

Mathematica version OVA: benchmark

In[23]:= $\nabla^2 \text{phi}[r, \theta, z]$ // Assuming[r > 0, FullSimplify[#] // Expand] &

Out[23]= $$\text{phi}^{(0,0,2)}[r,\theta,z] + \frac{\text{phi}^{(0,2,0)}[r,\theta,z]}{r^2} + \frac{\text{phi}^{(1,0,0)}[r,\theta,z]}{r} + \text{phi}^{(2,0,0)}[r,\theta,z]$$

In[24]:= A_θ = defineVector[Ar[r, θ, z], At[r, θ, z], Az[r, θ, z]]

Out[24]= vector[{Ar[r, θ, z], At[r, θ, z], Az[r, θ, z]}]

In[25]:= $(A_\theta \cdot \nabla)\, A_\theta$ // Assuming[r > 0, FullSimplify[#] // Expand] &

Out[25]= $$\text{vector}\Big[\Big\{\frac{-\text{At}[r,\theta,z]^2 + r\,\text{Az}[r,\theta,z]\,\text{Ar}^{(0,0,1)}[r,\theta,z] + \text{At}[r,\theta,z]\,\text{Ar}^{(0,1,0)}[r,\theta,z] + r\,\text{Ar}[r,\theta,z]\,\text{Ar}^{(1,0,0)}[r,\theta,z]}{r},$$
$$\text{Az}[r,\theta,z]\,\text{At}^{(0,0,1)}[r,\theta,z] + \frac{\text{At}[r,\theta,z]\left(\text{Ar}[r,\theta,z] + \text{At}^{(0,1,0)}[r,\theta,z]\right)}{r} + \text{Ar}[r,\theta,z]\,\text{At}^{(1,0,0)}[r,\theta,z],$$
$$\text{Az}[r,\theta,z]\,\text{Az}^{(0,0,1)}[r,\theta,z] + \frac{\text{At}[r,\theta,z]\,\text{Az}^{(0,1,0)}[r,\theta,z]}{r} + \text{Ar}[r,\theta,z]\,\text{Az}^{(1,0,0)}[r,\theta,z]\Big\}\Big]$$

In[26]:= $\nabla \cdot A_\theta$ // Assuming[r > 0, FullSimplify[#] // Expand] &

Out[26]= $$\frac{\text{Ar}[r,\theta,z]}{r} + \text{Az}^{(0,0,1)}[r,\theta,z] + \frac{\text{At}^{(0,1,0)}[r,\theta,z]}{r} + \text{Ar}^{(1,0,0)}[r,\theta,z]$$

In[27]:= $\nabla \times A_\theta$ // Assuming[r > 0, FullSimplify[#] // Expand] &

Out[27]= $$\text{vector}\Big[\Big\{-\text{At}^{(0,0,1)}[r,\theta,z] + \frac{\text{Az}^{(0,1,0)}[r,\theta,z]}{r},\ \text{Ar}^{(0,0,1)}[r,\theta,z] - \text{Az}^{(1,0,0)}[r,\theta,z],\ \frac{\text{At}[r,\theta,z] - \text{Ar}^{(0,1,0)}[r,\theta,z]}{r} + \text{At}^{(1,0,0)}[r,\theta,z]\Big\}\Big]$$

In[28]:= ∇A_θ // Assuming[r > 0, FullSimplify[#] // Expand] &

Out[28]= $$\Big\{\big\{\text{Ar}^{(1,0,0)}[r,\theta,z],\ \text{At}^{(1,0,0)}[r,\theta,z],\ \text{Az}^{(1,0,0)}[r,\theta,z]\big\},$$
$$\Big\{-\frac{\text{At}[r,\theta,z]}{r} + \frac{\text{Ar}^{(0,1,0)}[r,\theta,z]}{r},\ \frac{\text{Ar}[r,\theta,z]}{r} + \frac{\text{At}^{(0,1,0)}[r,\theta,z]}{r},\ \frac{\text{Az}^{(0,1,0)}[r,\theta,z]}{r}\Big\},\ \big\{\text{Ar}^{(0,0,1)}[r,\theta,z],\ \text{At}^{(0,0,1)}[r,\theta,z],\ \text{Az}^{(0,0,1)}[r,\theta,z]\big\}\Big\}$$

In[29]:= $\nabla \cdot \nabla \times A_\theta$

Out[29]= 0

Figure: Some of the vector analysis operations in the cylindrical coordinate system have been benchmarked

Outline

Background
- Motivations
- Mathematics

Python approach
- introduction
- Mathematics
- Python version OVA

Mathematica approach
- introduction
- modify GVA package
- *Mathematica* version OVA

Summary

References

Summary

- The OVA (Orthogonal curvilinear coordinate Vector Analysis) package has been implemented in both Python and *Mathematica* versions.
  - The Python version OVA is implemented using the vector analysis formulas assuming the basis normalized by the Lamé coefficients (thus can only be used in orthogonal coordinates).
  - The GVA package, originally developed by Prof. Qin in 1997, has been modified to ensure compatibility with newer versions of Mathematica, beyond version 8.0.
  - The Mathematica version of the OVA package is built upon the modified GVA package, which includes updated vector definitions and the normalization of each component using Lamé coefficients.
- The formulas used by OVA are presented, which can serve as a helpful cheatsheet. Additionally, formulas specific to cylindrical coordinates are listed in the 'Backup' section, which may be useful for those interested in linear devices.

References I

[1] Hong Qin, WM Tang, and G Rewoldt. "Symbolic vector analysis in plasma physics". In: *Computer physics communications* 116.1 (1999), pp. 107–120.

[2] Nan Chu. "SymFields: An Open Source Symbolic Fields Analysis Tool for General Curvilinear Coordinates in Python". In: *Proceedings of the Future Technologies Conference (FTC) 2021, Volume 1*. Springer. 2022, pp. 685–696.

[3] 胡友秋. 曲线坐标系. 中国科学技术大学地球和空间科学学院, 2010.

[4] PC Matthews. *Vector calculus*. Springer, 1998.

[5] William D D'haeseleer et al. *Flux coordinates and magnetic field structure: a guide to a fundamental tool of plasma theory*. Springer Science & Business Media, 2012.

References II

[6] Aaron Meurer et al. "SymPy: symbolic computing in Python". In: *PeerJ Computer Science* 3 (Jan. 2017), e103. ISSN: 2376-5992. DOI: 10.7717/peerj-cs.103. URL: https://doi.org/10.7717/peerj-cs.103.

[7] David B Wagner. *Power programming with MATHEMATICA: the Kernel*. McGraw-Hill Companies, 1996.

https://github.com/ymma98/OVA.git

All the codes are avaliable on Github. Welcome issues and pull requests!

Thank You for Your Attention!

Backup: vector analysis in cylindrical coordinate

- The application of vector analysis in cylindrical coordinates is particularly useful for FRC research (which is important for me).
- I list the formulas in the cylindrical coordinate system (r, θ, z) separately here for easy use.

$$h_1 = 1 \quad h_2 = r \quad h_3 = 1 \tag{30}$$

$$\mathcal{J} = V = \sqrt{g} = h_1 h_2 h_3 = r \tag{31}$$

Backup: vector analysis in cylindrical coordinate

$$
\begin{aligned}
\vec{a} \cdot \vec{b} &= a^i b_i = \frac{A_i}{h_i} B_i h_i = A_i B_i \\
\vec{a} \cdot \vec{\vec{T}} &= A_k \hat{e}_k \cdot T_{ij} \hat{e}_i \hat{e}_j = A_i T_{ij} \hat{e}_j \\
\vec{\vec{T}} \cdot \vec{a} &= T_{ij} \hat{e}_i \hat{e}_j \cdot a_k \hat{e}_k = A_j T_{ij} \hat{e}_i \\
\vec{a} \times \vec{b} &= A_i B_j \hat{e}_i \times \hat{e}_j = \epsilon_{ijk} A_i B_j \hat{e}_k
\end{aligned} \tag{32}
$$

$$
\vec{a} \times \vec{b} = \hat{e}_r (A_\theta B_z - A_z B_\theta) + \hat{e}_\theta (A_z B_r - A_r B_z) + \hat{e}_z (A_r B_\theta - A_\theta B_r) \tag{33}
$$

Backup: vector analysis in cylindrical coordinate

$$\begin{aligned}\vec{a}\cdot\vec{b} &= a^i b_i = \frac{A_i}{h_i}B_i h_i = A_i B_i \\ \vec{a}\cdot\vec{\vec{T}} &= A_k\hat{e}_k\cdot T_{ij}\hat{e}_i\hat{e}_j = A_i T_{ij}\hat{e}_j \\ \vec{\vec{T}}\cdot\vec{a} &= T_{ij}\hat{e}_i\hat{e}_j\cdot a_k\hat{e}_k = A_j T_{ij}\hat{e}_i \\ \vec{a}\times\vec{b} &= A_i B_j\hat{e}_i\times\hat{e}_j = \epsilon_{ijk}A_i B_j\hat{e}_k\end{aligned} \tag{34}$$

$$\vec{a}\times\vec{b} = \hat{e}_r(A_\theta B_z - A_z B_\theta) + \hat{e}_\theta(A_z B_r - A_r B_z) + \hat{e}_z(A_r B_\theta - A_\theta B_r) \tag{35}$$

Backup: vector analysis in cylindrical coordinate

$$\begin{aligned} \nabla \cdot \hat{e}_r &= \frac{1}{r} \\ \nabla \cdot \hat{e}_\theta &= 0 \\ \nabla \cdot \hat{e}_z &= 0 \end{aligned} \tag{36}$$

$$\begin{aligned} \nabla \times \hat{e}_r &= 0 \\ \nabla \times \hat{e}_\theta &= \frac{1}{r} \\ \nabla \times \hat{e}_z &= 0 \end{aligned} \tag{37}$$

$$\left\{ \begin{array}{l} (\hat{e}_r \cdot \nabla)\,\hat{e}_r = 0 \\ (\hat{e}_\theta \cdot \nabla)\,\hat{e}_\theta = -\frac{1}{r}\hat{e}_r \\ (\hat{e}_z \cdot \nabla)\,\hat{e}_z = 0 \end{array} \right. \tag{38}$$

$$\left\{ \begin{array}{l} \frac{\partial \hat{e}_r}{\partial r} = 0 \\ \frac{\partial \hat{e}_\theta}{\partial \theta} = -\hat{e}_r \\ \frac{\partial \hat{e}_z}{\partial z} = 0 \end{array} \right. \tag{39}$$

Backup: vector analysis in cylindrical coordinate

$$
\begin{aligned}
\nabla &= \vec{e}^i \frac{\partial}{\partial x^i} = \hat{e}_i \frac{1}{h_i} \frac{\partial}{\partial x^i} = \hat{e}_r \frac{\partial}{\partial r} + \hat{e}_\theta \frac{1}{r} \frac{\partial}{\partial \theta} + \hat{e}_z \frac{\partial}{\partial z} \\
\nabla^2 &= \Delta = \frac{1}{V} \sum_{i=1}^{3} \frac{\partial}{\partial x_i} \left(\frac{V}{h_i^2} \frac{\partial}{\partial x_i} \right) = \frac{1}{r} \frac{\partial}{\partial r} \left(r \frac{\partial}{\partial r} \right) + \frac{1}{r^2} \frac{\partial^2}{\partial \theta^2} + \frac{\partial^2}{\partial z^2} \\
\vec{u} \cdot \nabla &= U_i \hat{e}_i \cdot \hat{e}_j \frac{1}{h_j} \frac{\partial}{\partial x^j} = U_i \frac{1}{h_i} \frac{\partial}{\partial x^i} = U_r \frac{\partial}{\partial r} + \frac{U_\theta}{r} \frac{\partial}{\partial \theta} + U_z \frac{\partial}{\partial z}
\end{aligned}
\tag{40}
$$

Backup: vector analysis in cylindrical coordinate

$$\nabla\phi = \hat{e}_r\frac{\partial\phi}{\partial r} + \hat{e}_\theta\frac{1}{r}\frac{\partial\phi}{\partial\theta} + \hat{e}_z\frac{\partial\phi}{\partial z}$$

$$\nabla\cdot\vec{U} = \frac{1}{\sqrt{g}}\frac{\partial}{\partial x^i}\left(\sqrt{g}U^i\right) = \frac{1}{\sqrt{g}}\frac{\partial}{\partial x^i}\left(\sqrt{g}\frac{U^i}{h_i}\right) = \frac{1}{r}\frac{\partial}{\partial r}(rU_r) + \frac{1}{r}\frac{\partial U_\theta}{\partial\theta} + \frac{\partial U_z}{\partial z}$$

$$\begin{aligned}\nabla\times\vec{u} &= \frac{1}{h_i}\sum_{j=1}^{3}\left(\frac{1}{h_j}\frac{\partial h_i}{\partial x_j}\hat{e}_j\right)\times\hat{e}_i \\ &= \left(\frac{1}{r}\frac{\partial U_z}{\partial\theta} - \frac{\partial U_\theta}{\partial z}\right)\hat{e}_r + \left(\frac{\partial U_r}{\partial z} - \frac{\partial U_z}{\partial r}\right)\hat{e}_\theta + \left[\frac{1}{r}\frac{\partial}{\partial r}(rU_\theta) - \frac{1}{r}\frac{\partial U_r}{\partial\theta}\right]\hat{e}_z\end{aligned} \tag{41}$$

$$
\begin{aligned}
\nabla\vec{u} &= \hat{e}_i\hat{e}_j\frac{1}{h_i}\frac{\partial U_j}{\partial x^i} + \hat{e}_i\frac{\partial \hat{e}_j}{\partial x^i}\frac{U_j}{h_i} = \frac{\partial U_r}{\partial r}\hat{e}_r\hat{e}_r + \frac{\partial U_\theta}{\partial r}\hat{e}_r\hat{e}_\theta + \frac{\partial U_z}{\partial r}\hat{e}_r\hat{e}_z \\
&\quad + \left(\frac{1}{r}\frac{\partial U_r}{\partial \theta} - \frac{U_\theta}{r}\right)\hat{e}_\theta\hat{e}_r + \left(\frac{1}{r}\frac{\partial U_\theta}{\partial \theta} + \frac{U_r}{r}\right)\hat{e}_\theta\hat{e}_\theta + \frac{1}{r}\frac{\partial U_z}{\partial \theta}\hat{e}_\theta\hat{e}_z \\
&\quad + \frac{\partial U_r}{\partial z}\hat{e}_z\hat{e}_r + \frac{\partial U_\theta}{\partial z}\hat{e}_z\hat{e}_\theta + \frac{\partial U_z}{\partial z}\hat{e}_z\hat{e}_z \\
\vec{u}\cdot\nabla\vec{u} &= \left[U_r\frac{\partial U_r}{\partial r} + U_\theta\left(\frac{1}{r}\frac{\partial U_r}{\partial \theta} - \frac{U_\theta}{r}\right) + U_z\frac{\partial U_r}{\partial z}\right]\hat{e}_r \\
&\quad + \left[U_r\frac{\partial U_\theta}{\partial r} + U_\theta\left(\frac{1}{r}\frac{\partial U_\theta}{\partial \theta} + \frac{U_r}{r}\right) + U_z\frac{\partial U_\theta}{\partial z}\right]\hat{e}_\theta \\
&\quad + \left[U_r\frac{\partial U_z}{\partial r} + U_\theta\frac{1}{r}\frac{\partial U_z}{\partial \theta} + U_z\frac{\partial U_z}{\partial z}\right]\hat{e}_z \\
\nabla\cdot\vec{\vec{T}} &= \left[\frac{1}{r}\frac{\partial}{\partial r}\left(rT_{rr}\right) + \frac{1}{r}\frac{\partial T_{\theta r}}{\partial \theta} + \frac{\partial T_{zr}}{\partial z} - \frac{T_{\theta\theta}}{r}\right]\hat{e}_r \\
&\quad + \left[\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2 T_{r\theta}\right) + \frac{1}{r}\frac{\partial T_{\theta\theta}}{\partial \theta} + \frac{\partial T_{z\theta}}{\partial z} - \frac{T_{r\theta} - T_{\theta r}}{r}\right]\hat{e}_\theta \\
&\quad + \left[\frac{1}{r}\frac{\partial}{\partial r}\left(rT_{rz}\right) + \frac{1}{r}\frac{\partial T_{\theta z}}{\partial \theta} + \frac{\partial T_{zz}}{\partial z}\right]\hat{e}_z
\end{aligned}
\tag{42}
$$