

# 第一章

## Navier-Stokes Equations

Continuity Equation

$$\nabla \cdot \vec{V} = 0$$

Momentum Equations

$$\rho \frac{D\vec{V}}{Dt} = -\nabla p + \rho \vec{g} + \mu \nabla^2 \vec{V}$$

Total derivative

Pressure gradient

Body force term

Diffusion term

$$\rho \left[ \frac{\partial V}{\partial t} + (\vec{V} \cdot \nabla) V \right]$$

Fluid flows in the direction of largest change in pressure.

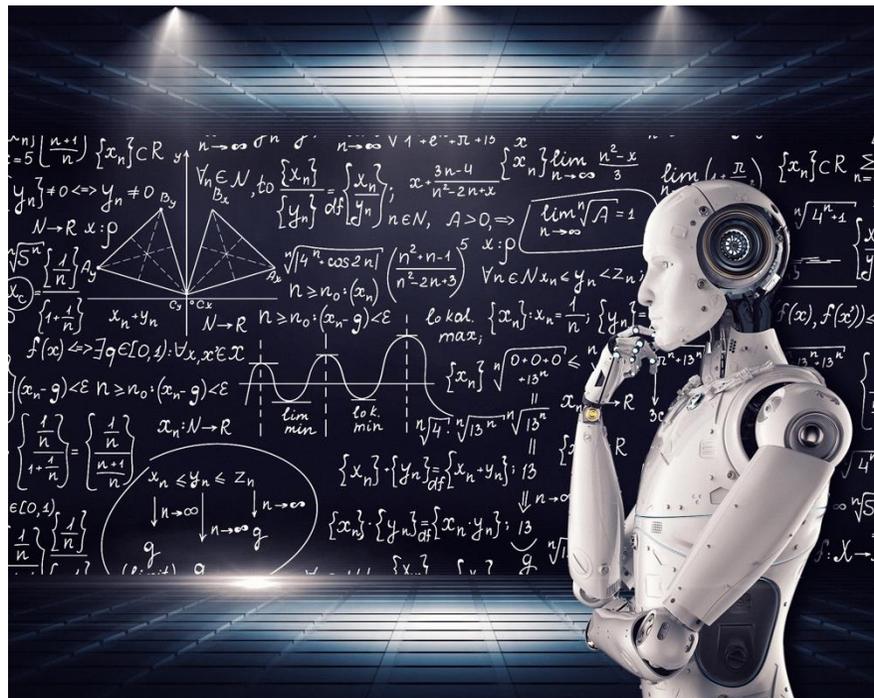
External forces, that act on the fluid (gravitational force or electromagnetic).

For a Newtonian fluid, viscosity operates as a diffusion of momentum.

Change of velocity with time

Convective term

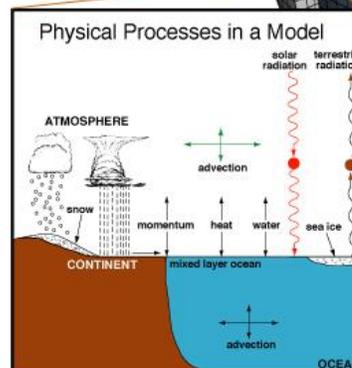
# 第二章



# 第三章

Horizontal Grid (Latitude-Longitude)

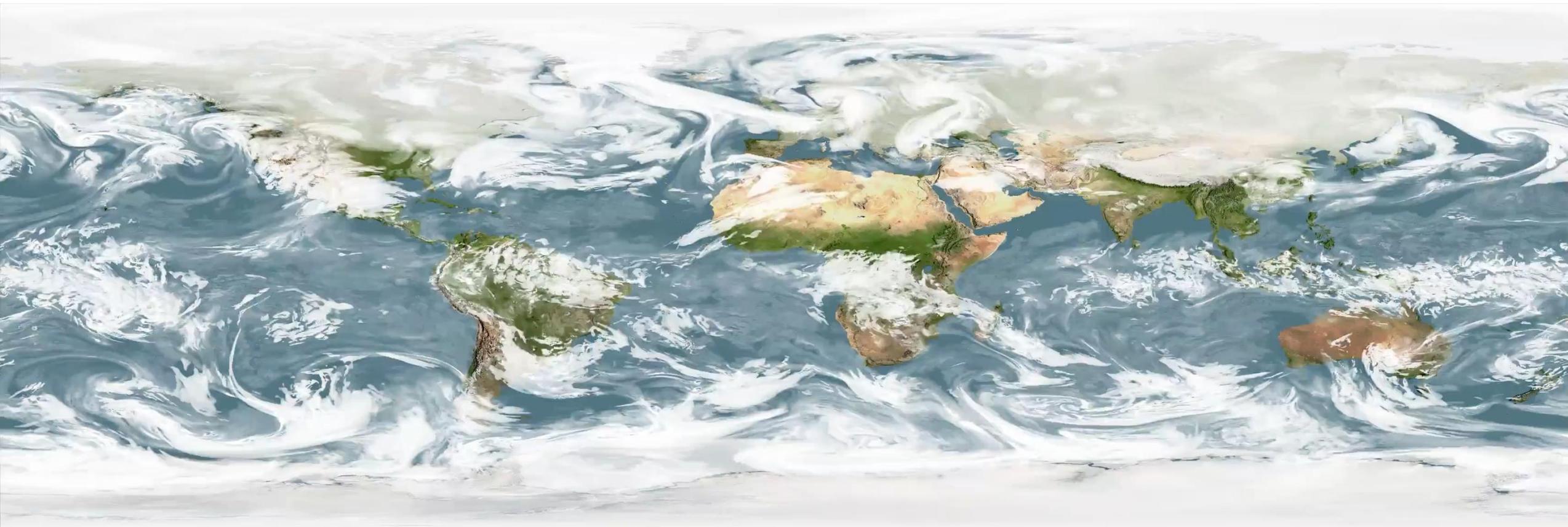
Vertical Grid (Height or Pressure)



# 地球流体的数值模拟与AI预测

任课老师：杨邱、闻新宇

2024年秋季学期

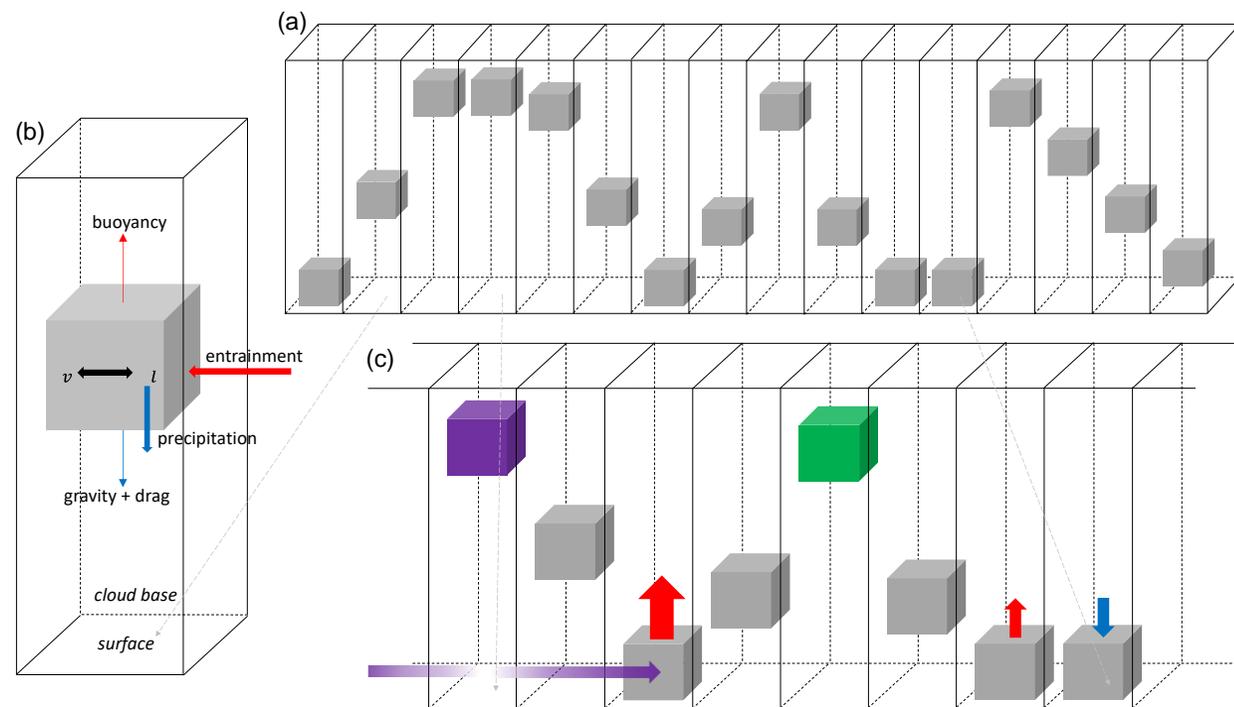


GEOS Model

通过数学建模的方法揭示大气中的对流组织化机制，并应用于实际



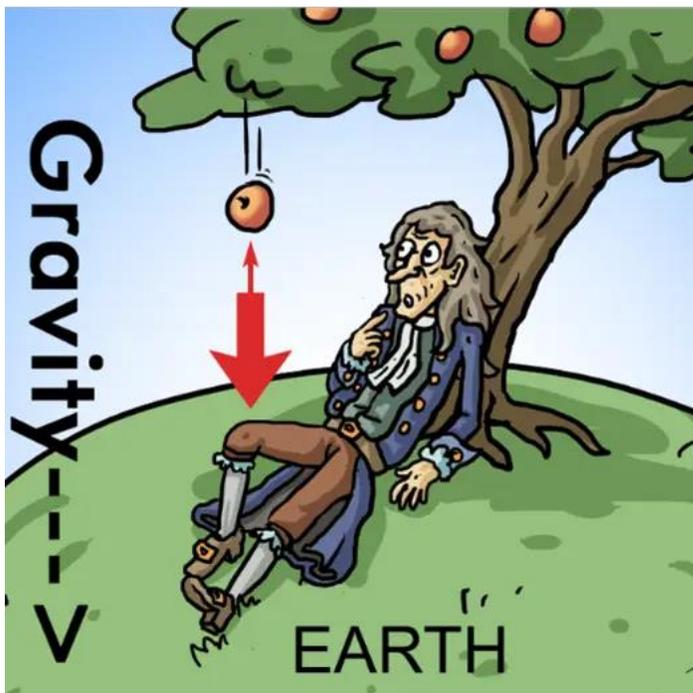
南中国海上的一次中尺度对流系统事件



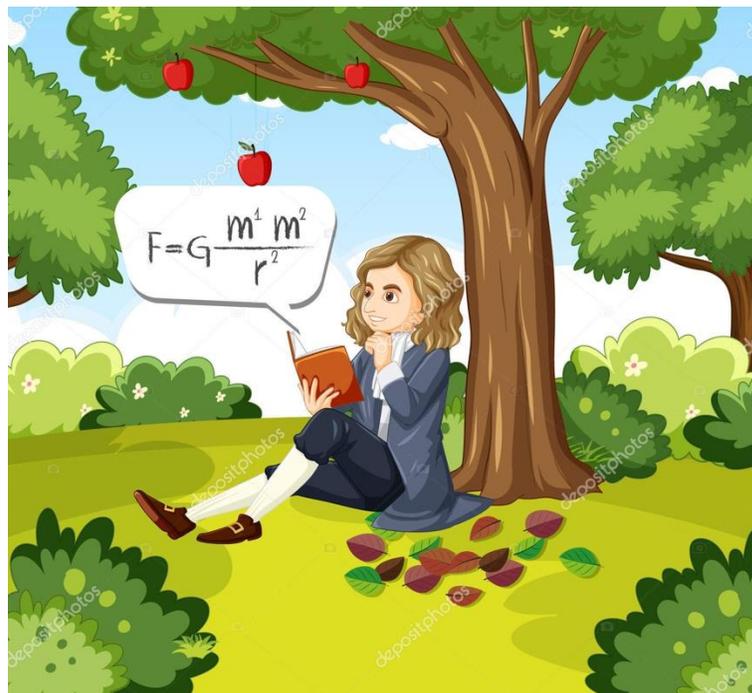
欢迎同学们访问我的科研主页 <https://qiuyang50.github.io/>



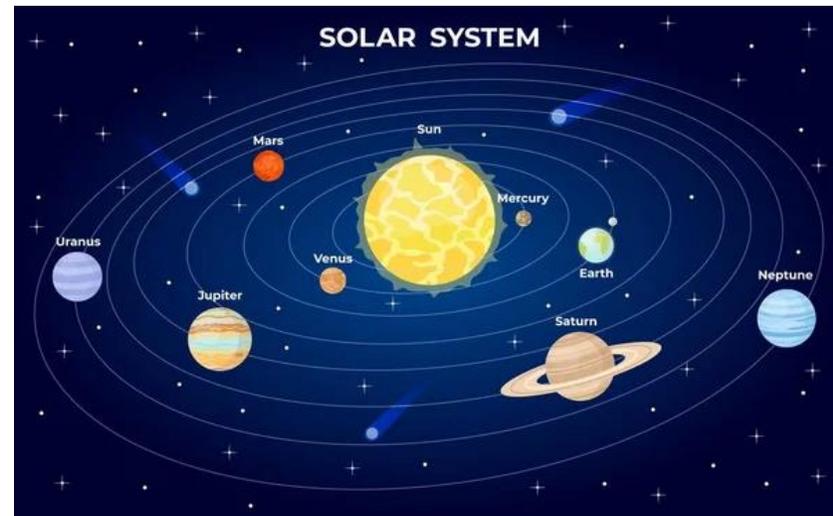
# 经典的物理建模方法取得了巨大的成功



观测、实验



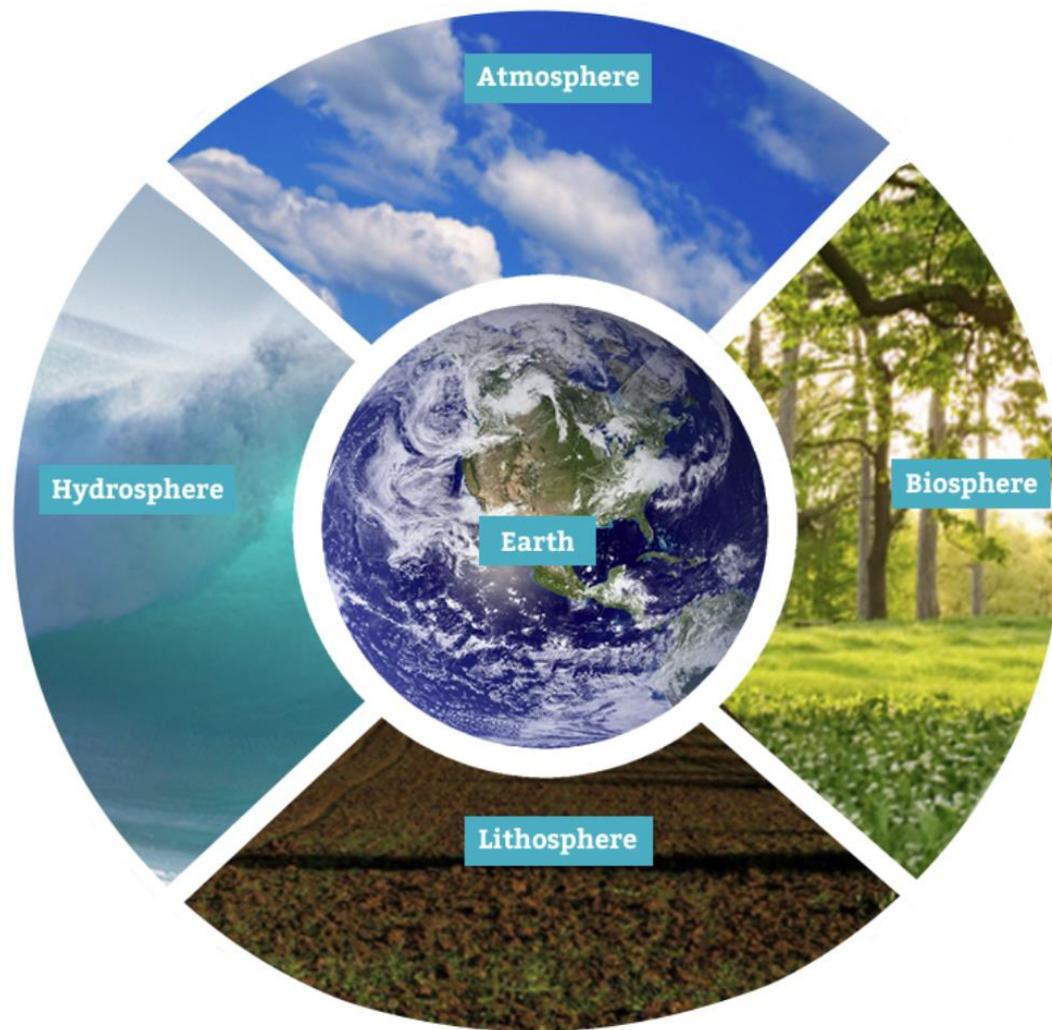
数学建模，数值求解



揭示机制、预测未来

但是，物理建模也有局限性

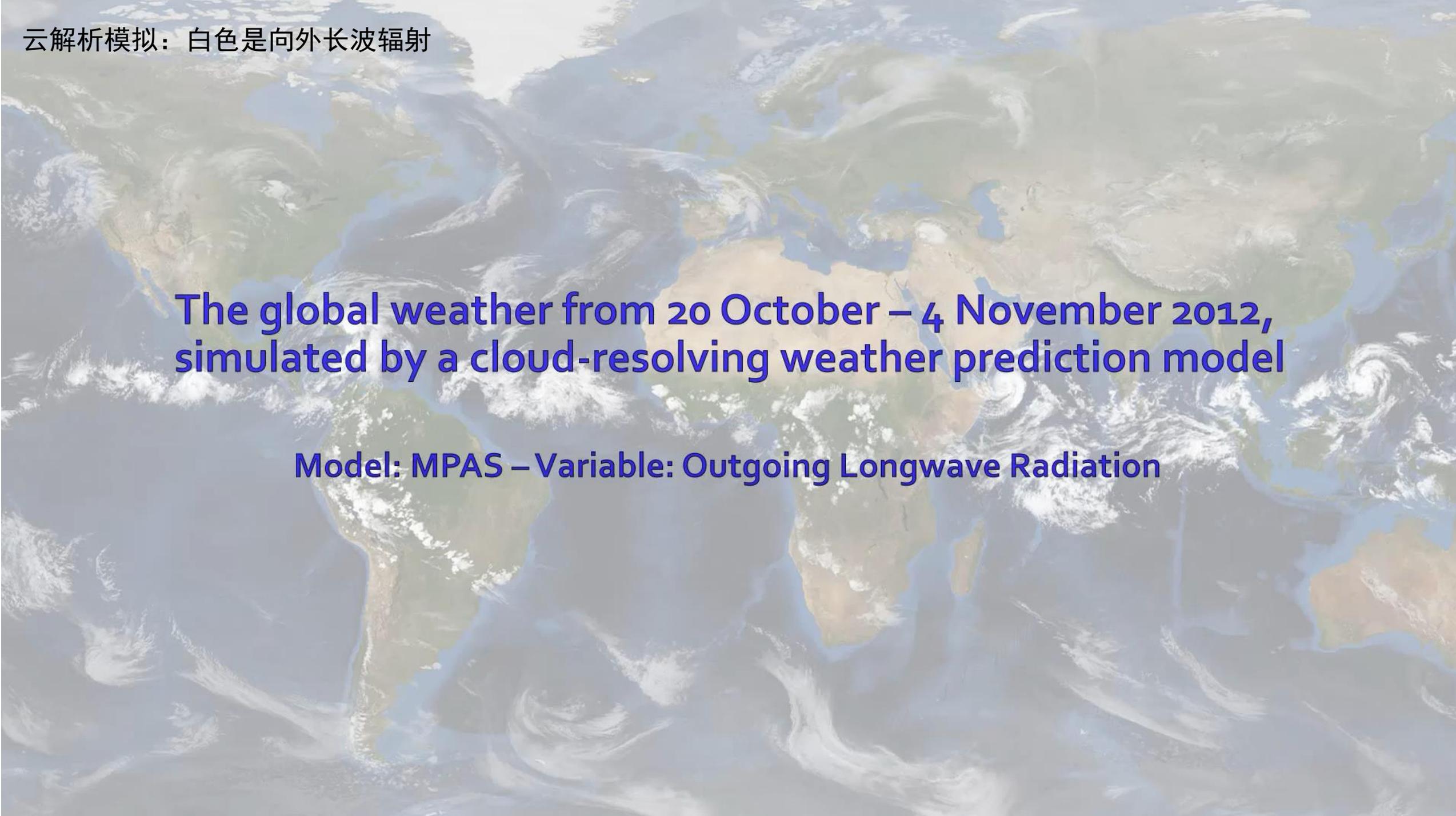
多圈层耦合造成了高度的系统复杂性



云解析模拟：白色是向外长波辐射

**The global weather from 20 October – 4 November 2012,  
simulated by a cloud-resolving weather prediction model**

**Model: MPAS – Variable: Outgoing Longwave Radiation**



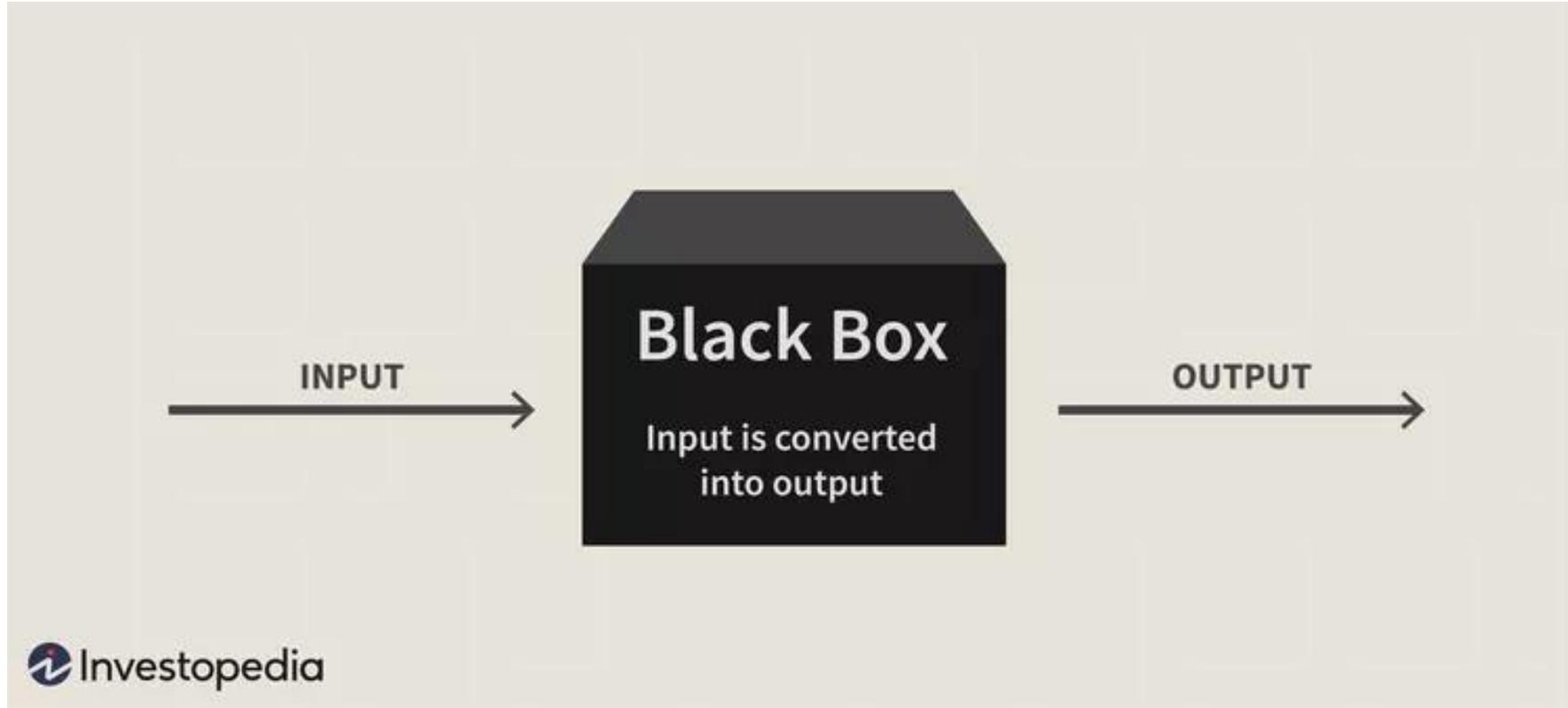
海洋模拟：白色是海表洋流

到底什么是AI？



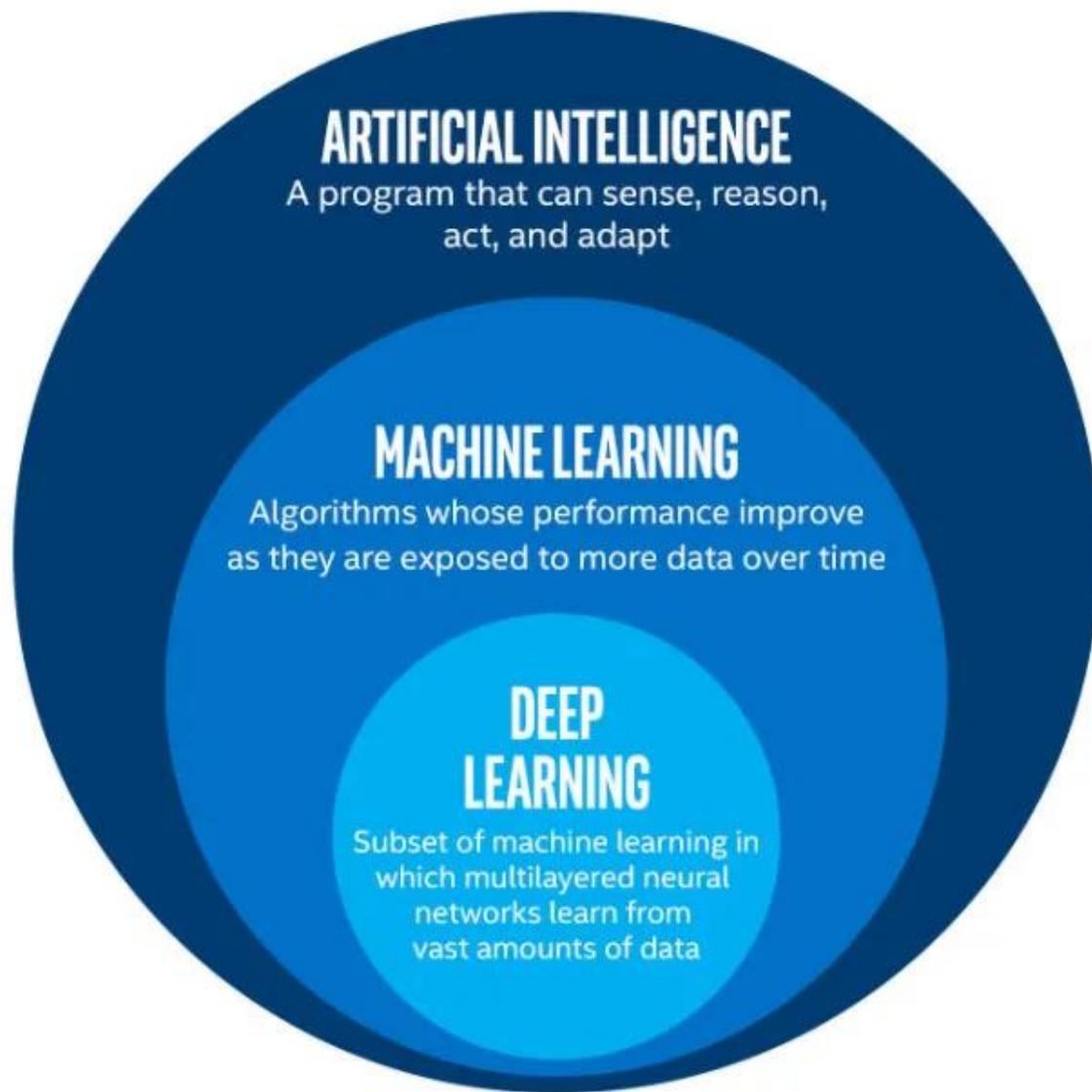
到底什么是AI？

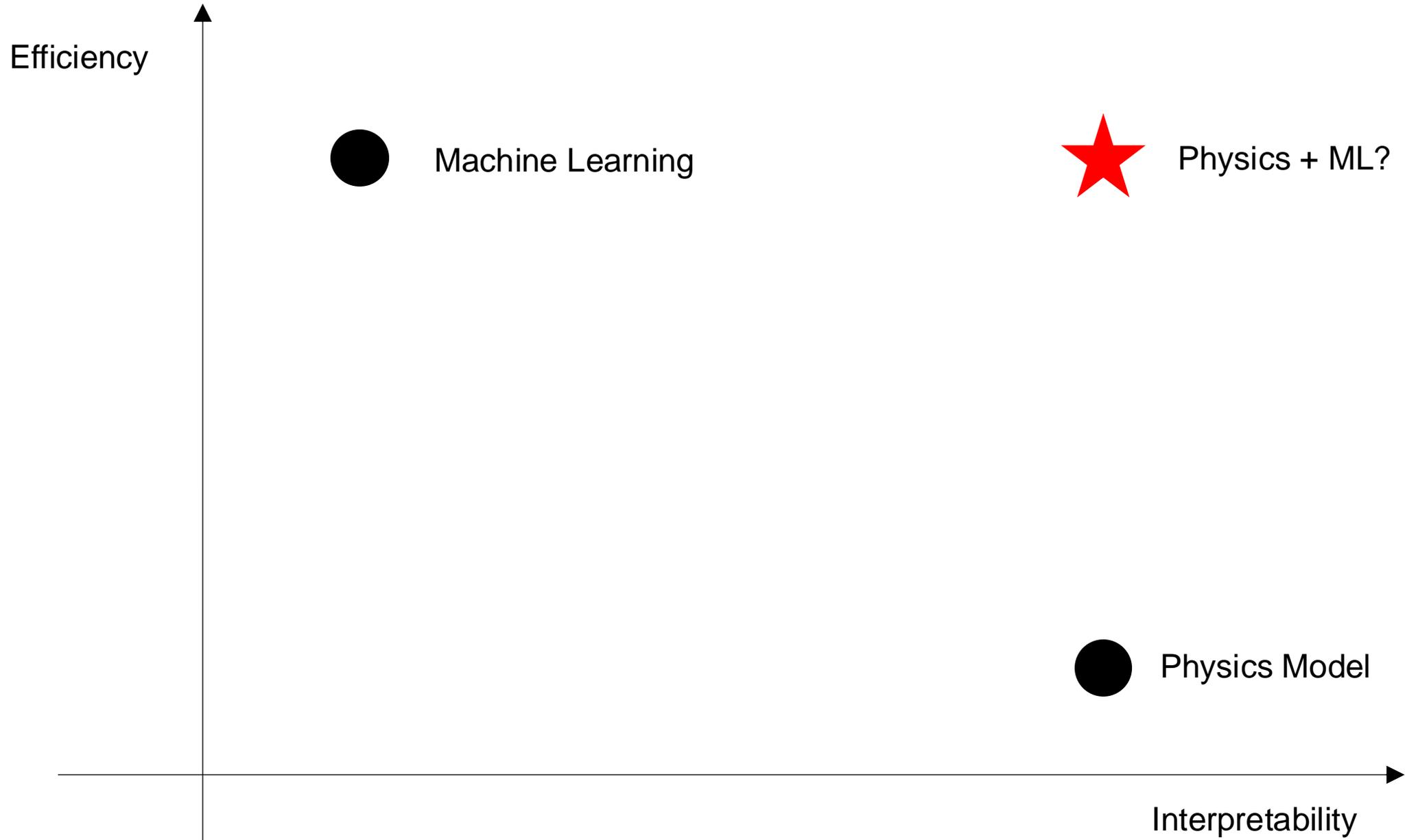




Physics Informed Neural Network (PINN)  
Layer-wise Relevance Propagation  
Explainable AI (XAI)

...





地球流体的数值模拟与AI预测 (2024 秋)

授课教师	周次	日期	上课内容	日期	上课内容	课程安排
杨	1	9月9日	1.1	9月12日	1.2	第一章：地球流体基本原理（杨）
	2	9月14日	1.3	9月19日	1.4	1.1 本课程简介
	3	9月23日	2.1	9月26日	2.2	1.2 基本控制方程
	4	9月30日	2.3	10月3日	国庆节放假	1.3 浅水模型及应用实例
	5	10月7日	国庆节放假	10月10日	2.4	1.4 准地转模型及应用实例
	6	10月14日	2.5	10月17日	2.5	
	7	10月21日	2.5	10月24日	2.5	第二章：机器学习原理与应用（杨）
	8	10月28日	2.6	10月31日	2.6	2.1 AI预测前沿进展
闻	9	11月4日	3.1	11月7日	3.2	2.2 线性回归分析
	10	11月11日	3.3	11月14日	3.3	2.3 决策树
	11	11月18日	3.4	11月21日	3.4	2.4 支持向量机
	12	11月25日	3.4	11月28日	3.5	2.5 深度学习
	13	12月2日	3.5	12月5日	3.5	2.6 非监督学习
	14	12月9日	3.6	12月12日	3.6	
	15	12月16日	3.6	12月19日	3.6	第三章：地球流体模式与模拟（闻）
闻、杨	16	12月23日	学生报告	12月26日	学生报告	3.1 LFR与Charney的故事
注：中秋节9月16日调课至9月14日						3.2 数值模式的寂静革命
课程网站 <a href="https://qiuyang50.github.io/pages/modeling_2024fall/">https://qiuyang50.github.io/pages/modeling_2024fall/</a>						3.3 天气模式
						3.4 天气模式WRF实践
						3.5 气候模式
						3.6 气候模式MITgcm实践

以培养学生科研能力为目标，强调具体应用范例，内容覆盖流体理论、数值编程、统计方法、实际应用

1. 请查看课程网站，上完每一章后将上传相应材料（安排、课件、作业）

[https://qiuyang50.github.io/\\_pages/modeling\\_2024fall/](https://qiuyang50.github.io/_pages/modeling_2024fall/)



2. 请加入课程微信群（通知、交流），群内请注明真实姓名

$$\text{平时作业}40 + \text{期末大作业}60 (\text{报告}30+\text{论文}30) = \text{总分}100$$

无上课考勤，无期中、期末考试。

平时作业大约4次，自发布之日起，两周后截止提交，将纸质版作业交给任课老师。

期末大作业包括口头报告和论文两部分，选题将在开学初给定范围自由选择，期末最后两次课用于口头报告，同学相互打分，论文由教师打分。

所需知识和技能：良好的数理基础，python (jupyterlab)，以及一颗好奇心！

期末大作业选题（需要与杨邱老师商量确定）：选择任一你感兴趣的章节，探讨机器学习或数值模拟在地球流体中的应用，须包含两部分，一是文献阅读，二是自己实践。

例如，应用卷积神经网络研究大气环境对热带气旋的影响。

#### 课程安排

##### 第一章：地球流体基本原理（杨）

- 1.1 本课程简介
- 1.2 基本控制方程
- 1.3 浅水模型及应用实例
- 1.4 准地转模型及应用实例

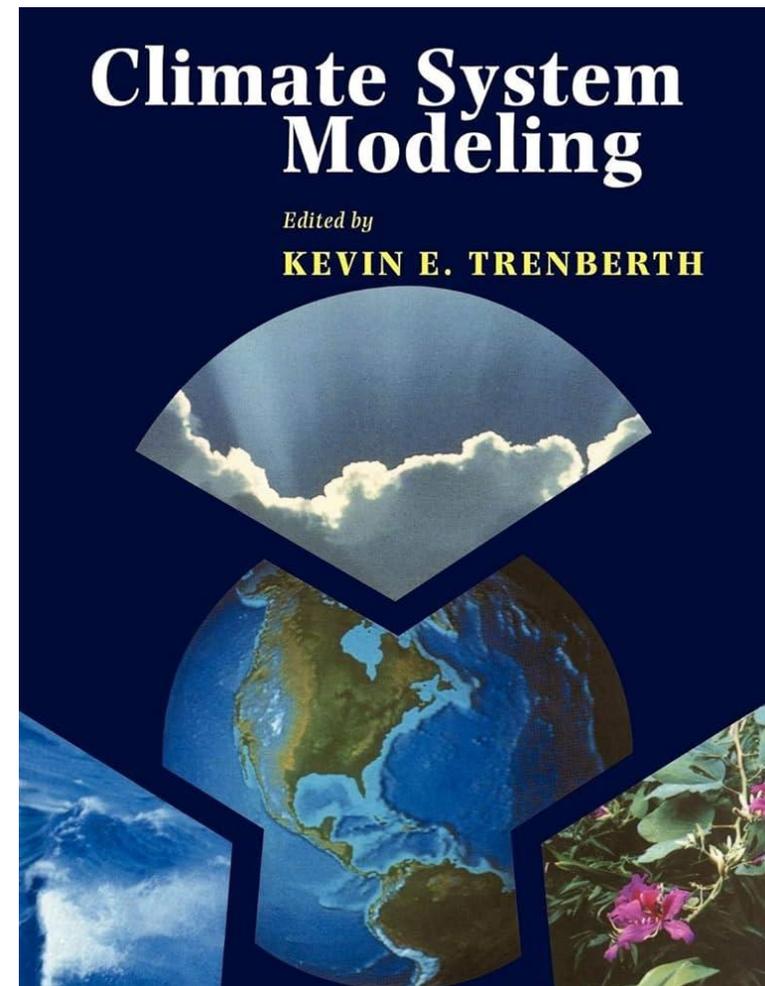
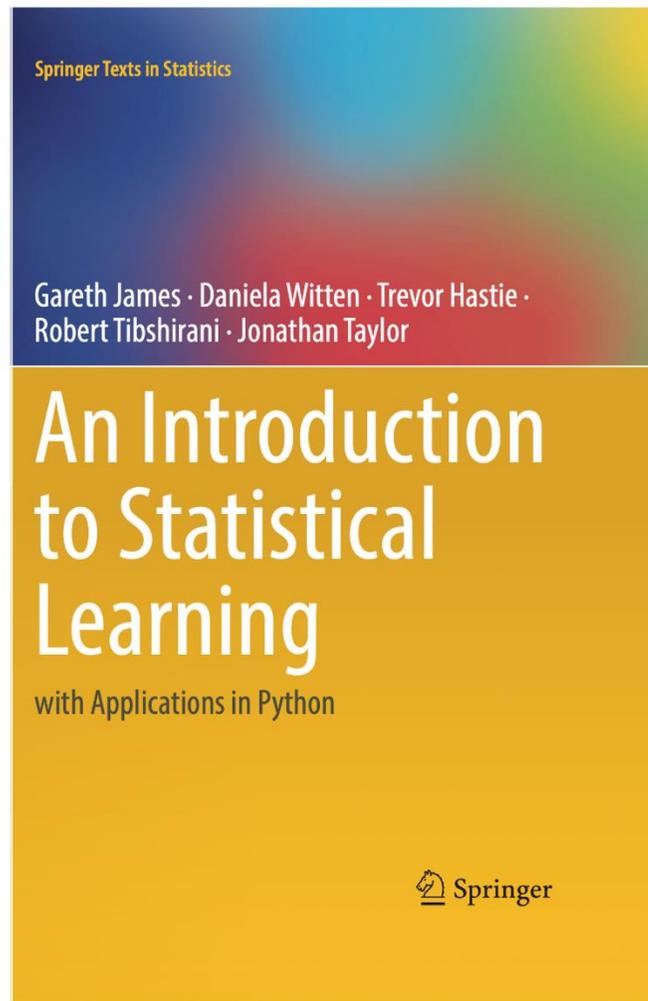
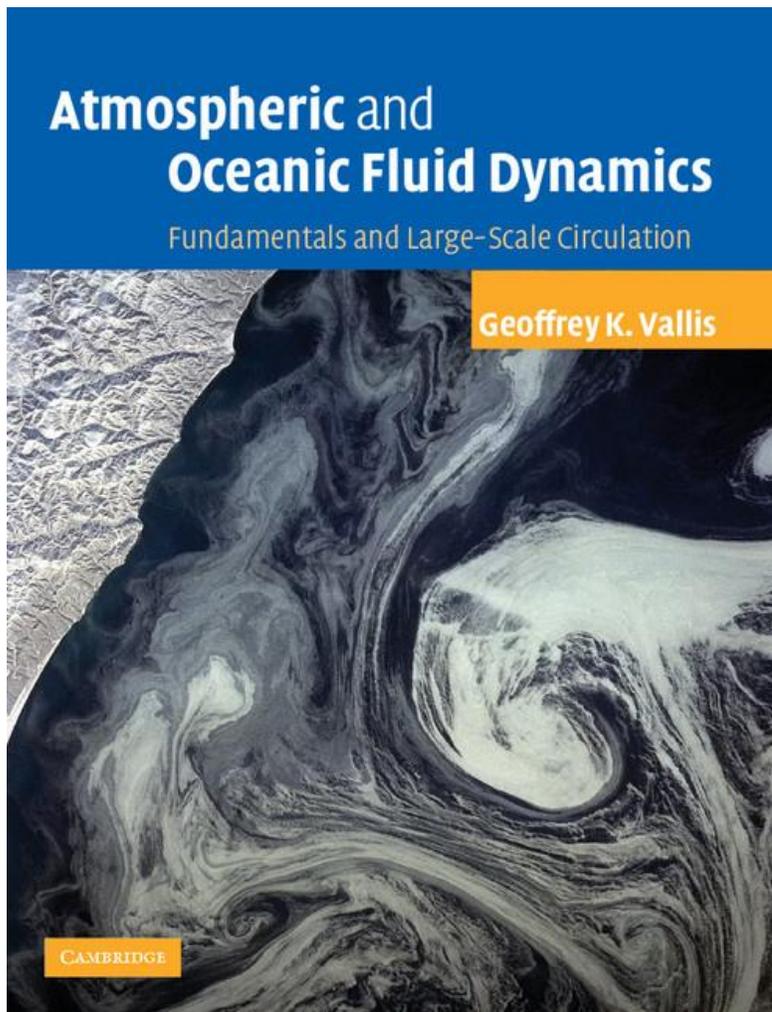
##### 第二章：机器学习原理与应用（杨）

- 2.1 AI预测前沿进展
- 2.2 线性回归分析
- 2.3 决策树
- 2.4 支持向量机
- 2.5 深度学习
- 2.6 非监督学习

##### 第三章：地球流体模式与模拟（闻）

- 3.1 LFR与Charney的故事
- 3.2 数值模式的寂静革命
- 3.3 天气模式
- 3.4 天气模式WRF实践
- 3.5 气候模式
- 3.6 气候模式MITgcm实践

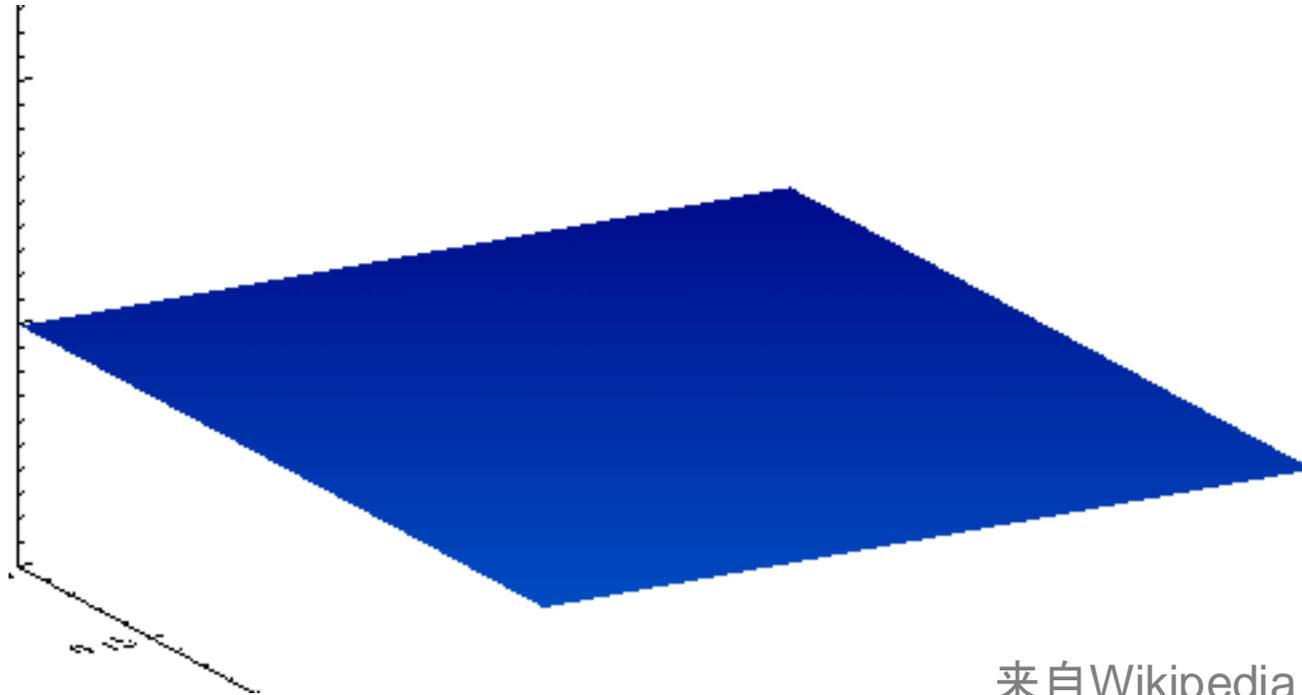
参考书



课件，代码，数据可从网站下载  
<https://www.statlearning.com/>

**关于本课程，你还有什么问题吗？**

# 第一章：地球流体基本原理

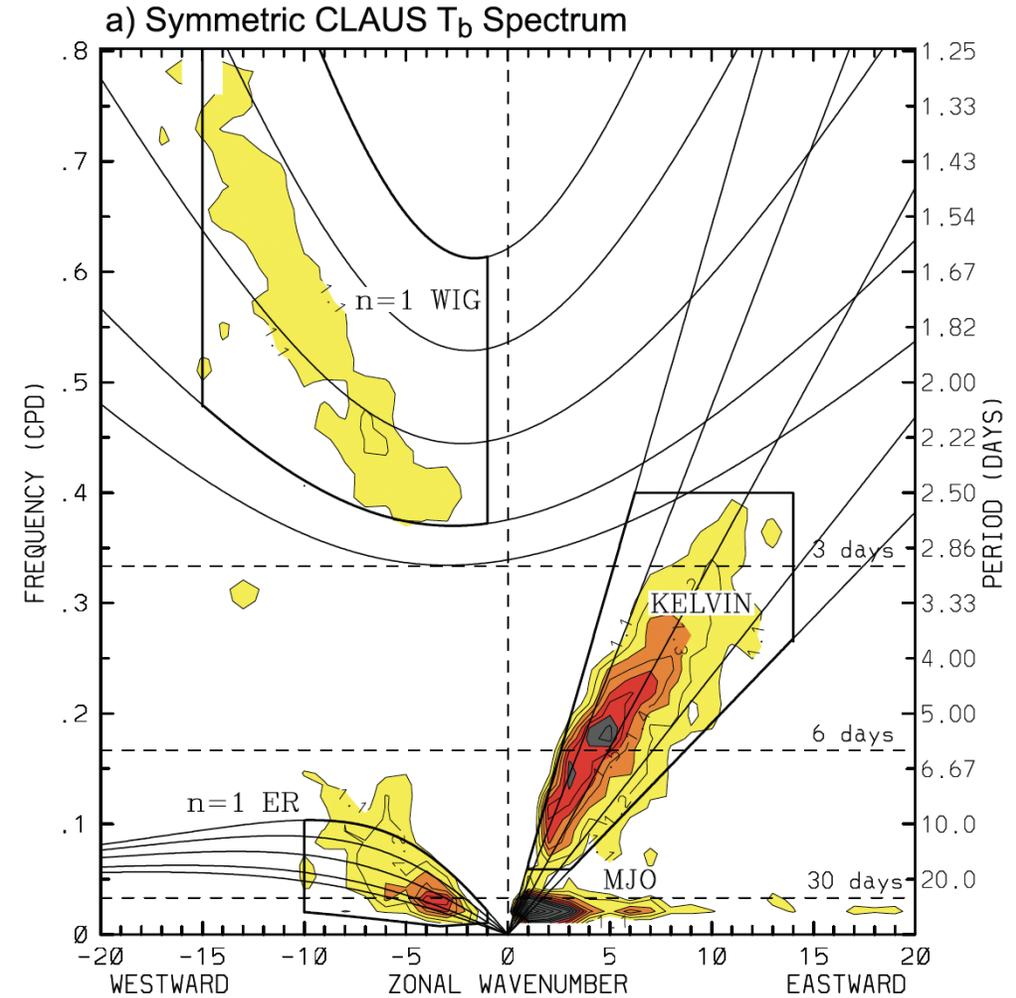


来自Wikipedia

浅水方程

momentum: 
$$\frac{D\mathbf{u}}{Dt} + \mathbf{f} \times \mathbf{u} = -g\nabla\eta.$$

mass continuity: 
$$\frac{Dh}{Dt} + h\nabla \cdot \mathbf{u} = 0$$



$$\frac{\partial \psi}{\partial t} + c \frac{\partial \psi}{\partial x} = 0$$

$$\frac{df}{dx}(x_0) = \lim_{\Delta x \rightarrow 0} \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x},$$

$$\frac{df}{dx}(x_0) = \lim_{\Delta x \rightarrow 0} \frac{f(x_0) - f(x_0 - \Delta x)}{\Delta x},$$

$$\frac{df}{dx}(x_0) = \lim_{\Delta x \rightarrow 0} \frac{f(x_0 + \Delta x) - f(x_0 - \Delta x)}{2\Delta x}.$$

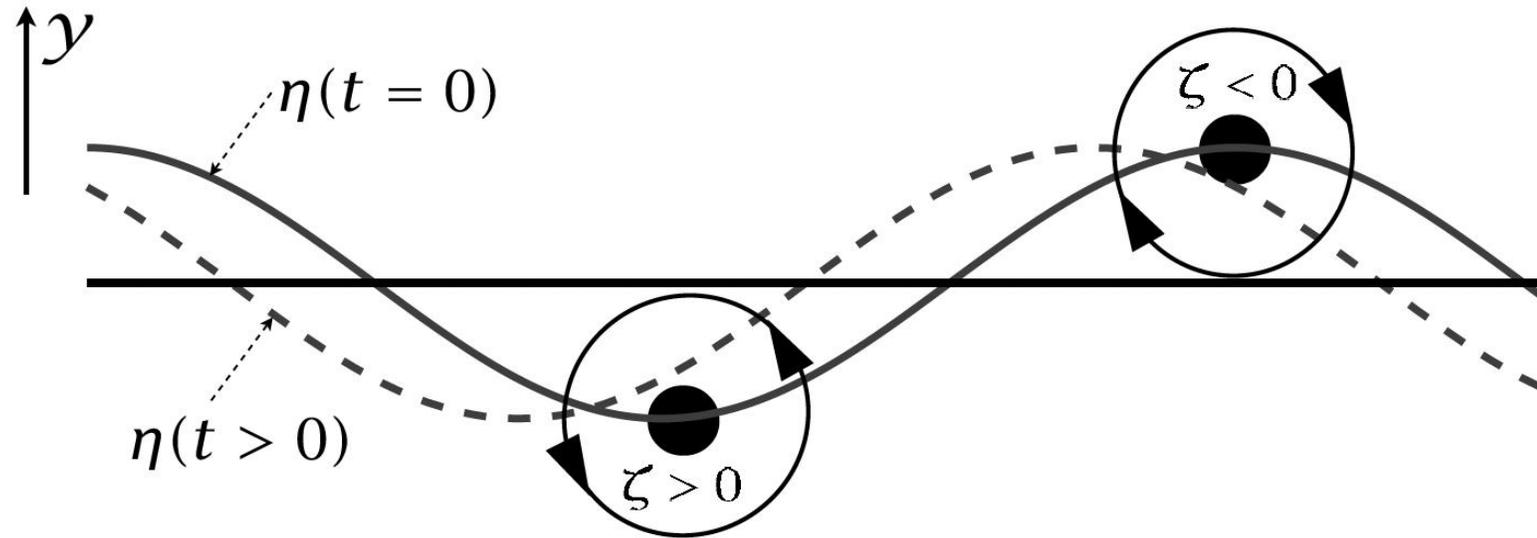
$$\frac{D}{Dt} \left( \nabla^2 \psi + \beta y - \frac{1}{L_d^2} \psi \right) = 0 ,$$

where  $\psi = (g/f_0)\eta$ ,  $L_d^2 = gH/f_0^2$ , and the advective derivative is

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + u_g \frac{\partial}{\partial x} + v_g \frac{\partial}{\partial y} = \frac{\partial}{\partial t} - \frac{\partial \psi}{\partial y} \frac{\partial}{\partial x} + \frac{\partial \psi}{\partial x} \frac{\partial}{\partial y} = \frac{\partial}{\partial t} + J(\psi, \cdot). \quad (5.67)$$

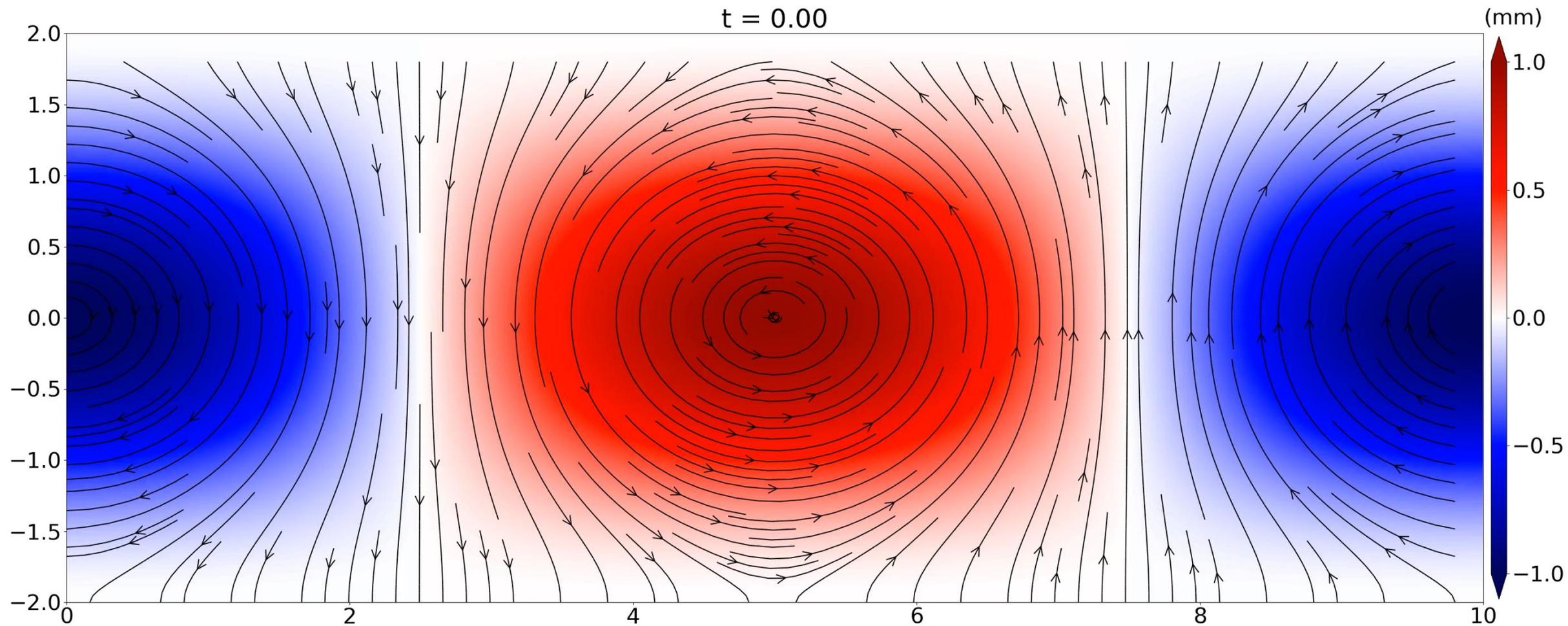
$$q \equiv \zeta + \beta y - \frac{f_0}{H} \eta = \nabla^2 \psi + \beta y - \frac{1}{L_d^2} \psi \quad (5.69)$$

is the *shallow water quasi-geostrophic potential vorticity*.



**Fig. 5.4** The mechanism of a two-dimensional ( $x$ - $y$ ) Rossby wave. An initial disturbance displaces a material line at constant latitude (the straight horizontal line) to the solid line marked  $\eta(t = 0)$ . Conservation of potential vorticity,  $\beta y + \zeta$ , leads to the production of relative vorticity, as shown for two parcels. The associated velocity field (arrows on the circles) then advects the fluid parcels, and the material line evolves into the dashed line. The phase of the wave has propagated westwards.

# Example 1



## 第二章：机器学习原理与应用（杨）

### 2.1 AI预测前沿进展

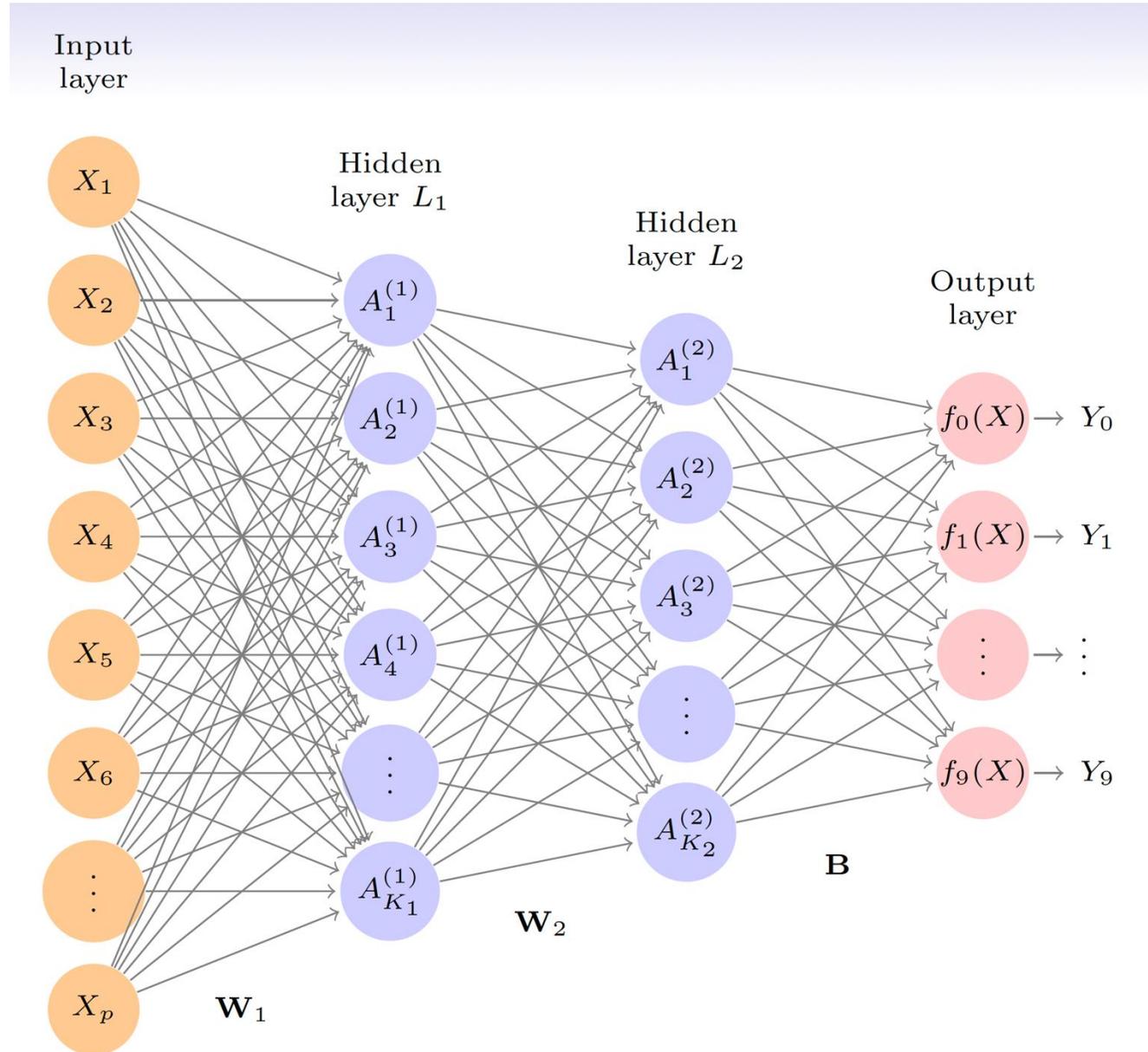
### 2.2 线性回归分析

### 2.3 决策树

### 2.4 支持向量机

### 2.5 深度学习

### 2.6 非监督学习



Launcher Ch10-deeplearning-lab.ipynb Python 3 (ipykernel)

## Deep Learning

launch binder

In this section we demonstrate how to fit the examples discussed in the text. We use the Python `torch` package, along with the `pytorch_lightning` package which provides utilities to simplify fitting and evaluating models. This code can be impressively fast with certain special processors, such as Apple's new M1 chip. The package is well-structured, flexible, and will feel comfortable to Python users. A good companion is the site [pytorch.org/tutorials](https://pytorch.org/tutorials). Much of our code is adapted from there, as well as the `pytorch_lightning` documentation. (The precise URLs at the time of writing are <https://pytorch.org/tutorials/beginner/basics/intro.html> and <https://pytorch-lightning.readthedocs.io/en/latest/>.)

We start with several standard imports that we have seen before.

```
[1]: import numpy as np, pandas as pd
      from matplotlib.pyplot import subplots
      from sklearn.linear_model import \
          (LinearRegression,
           LogisticRegression,
           Lasso)
      from sklearn.preprocessing import StandardScaler
      from sklearn.model_selection import KFold
      from sklearn.pipeline import Pipeline
      from ISLP import load_data
      from ISLP.models import ModelSpec as MS
      from sklearn.model_selection import \
          (train_test_split,
           GridSearchCV)
```

### Torch-Specific Imports

There are a number of imports for `torch`. (These are not included with `ISLP`, so must be installed separately.) First we import the main library and essential tools used to specify sequentially-structured networks.

```
[2]: import torch
      from torch import nn
      from torch.optim import RMSprop
      from torch.utils.data import TensorDataset
```

There are several other helper packages for `torch`. For instance, the `torchmetrics` package has utilities to compute various metrics to evaluate performance when fitting a model. The `torchinfo` package provides a useful summary of the layers of a model. We use the `read_image()` function when loading test images in Section-???.

If you have not already installed the packages `torchvision` and `torchinfo` you can install them by running `pip install torchinfo torchvision`. We can now import from `torchinfo`.

```
[3]: from torchmetrics import (MeanAbsoluteError,
                               R2Score)
      from torchinfo import summary
```

The package `pytorch_lightning` is a somewhat higher-level interface to `torch` that simplifies the specification and fitting of models by reducing the amount of boilerplate code needed (compared to using `torch` alone).

```
[4]: from pytorch_lightning import Trainer
      from pytorch_lightning.loggers import CSVLogger
```

In order to reproduce results we use `seed_everything()`. We will also instruct `torch` to use deterministic algorithms where possible.

```
[5]: from pytorch_lightning import seed_everything
      seed_everything(0, workers=True)
      torch.use_deterministic_algorithms(True, warn_only=True)
```

Python coding in jupyterlab

华为盘古大模型

# nature

[Explore content](#) ▾

[About the journal](#) ▾

[Publish with us](#) ▾

---

[nature](#) > [articles](#) > article

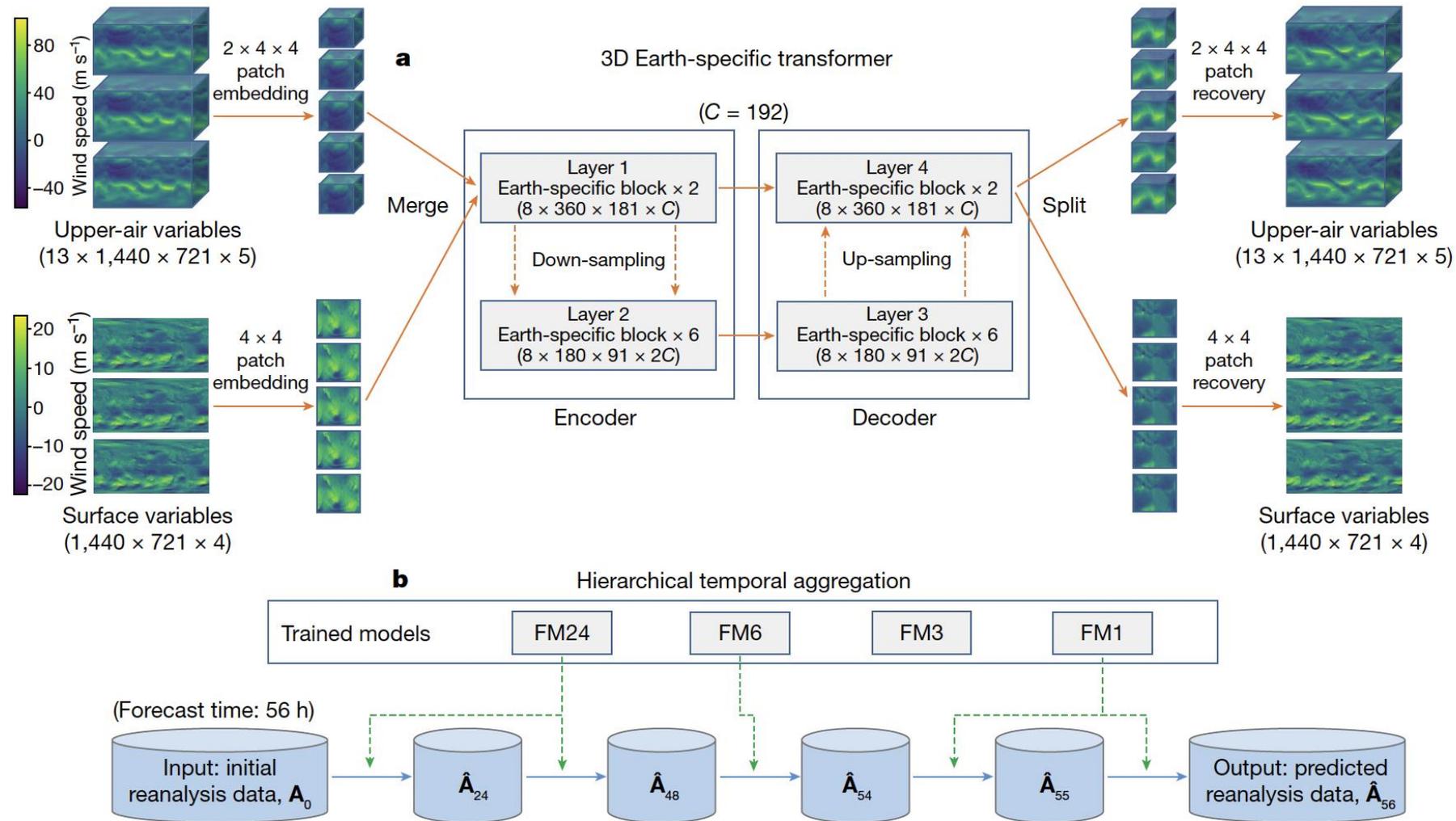
Article | [Open access](#) | Published: 05 July 2023

## Accurate medium-range global weather forecasting with 3D neural networks

[Kaifeng Bi](#), [Lingxi Xie](#), [Hengheng Zhang](#), [Xin Chen](#), [Xiaotao Gu](#) & [Qi Tian](#) 

[Nature](#) **619**, 533–538 (2023) | [Cite this article](#)

**209k** Accesses | **191** Citations | **1728** Altmetric | [Metrics](#)

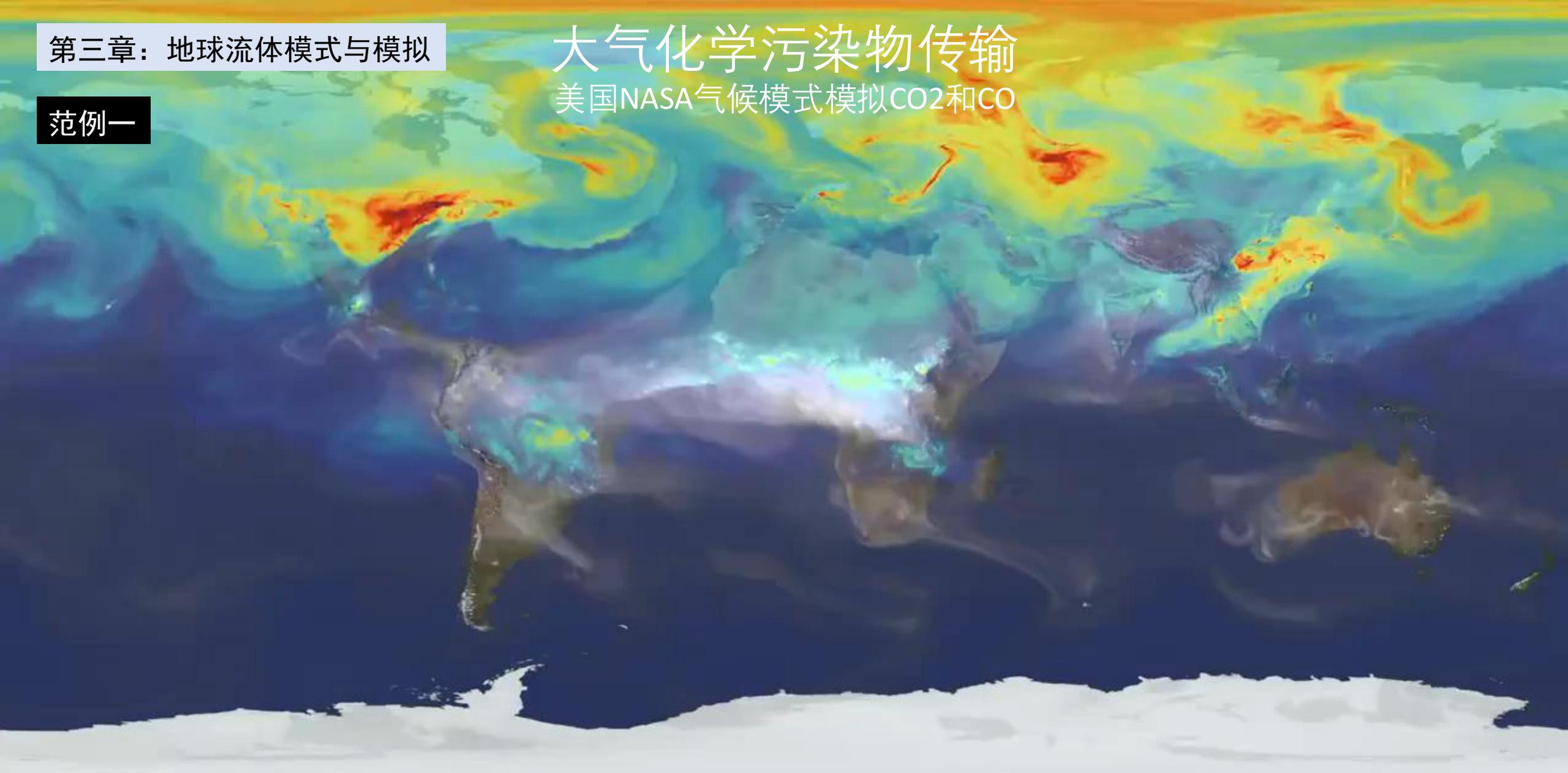


**Fig. 1 | Network training and inference strategies. a**, 3DEST architecture. Based on the standard encoder–decoder design of vision transformers, we adjusted the shifted-window mechanism<sup>19</sup> and applied an Earth-specific positional bias. **b**, Hierarchical temporal aggregation. Once given a lead time,

we used a greedy algorithm to perform forecasting with as few steps as possible. We use FM1, FM3, FM6 and FM24 to indicate the forecast models with lead times being 1 h, 3 h, 6 h or 24 h, respectively.  $\mathbf{A}_0$  is the input weather state and  $\hat{\mathbf{A}}_t$  denotes the predicted weather state at time  $t$  (in hours).

# 大气化学污染物传输

美国NASA气候模式模拟CO<sub>2</sub>和CO



2006 / 01 / 01

Global Modeling and Assimilation Office

Carbon Monoxide Column Abundance [ $1.0 \times 10^{18}$  molec  $\text{cm}^{-2}$ ]

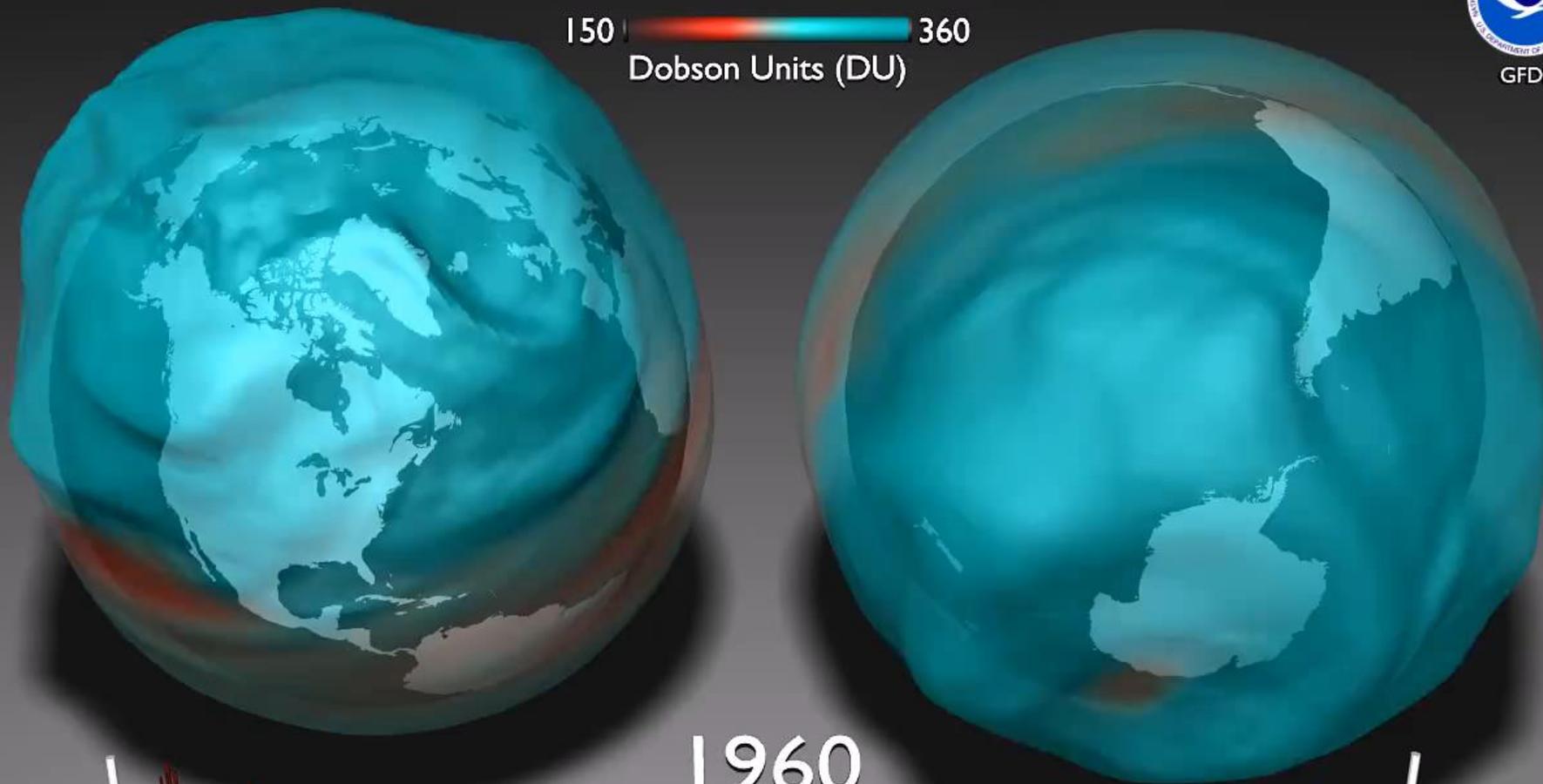


Carbon Dioxide Column Concentration [ppmv]

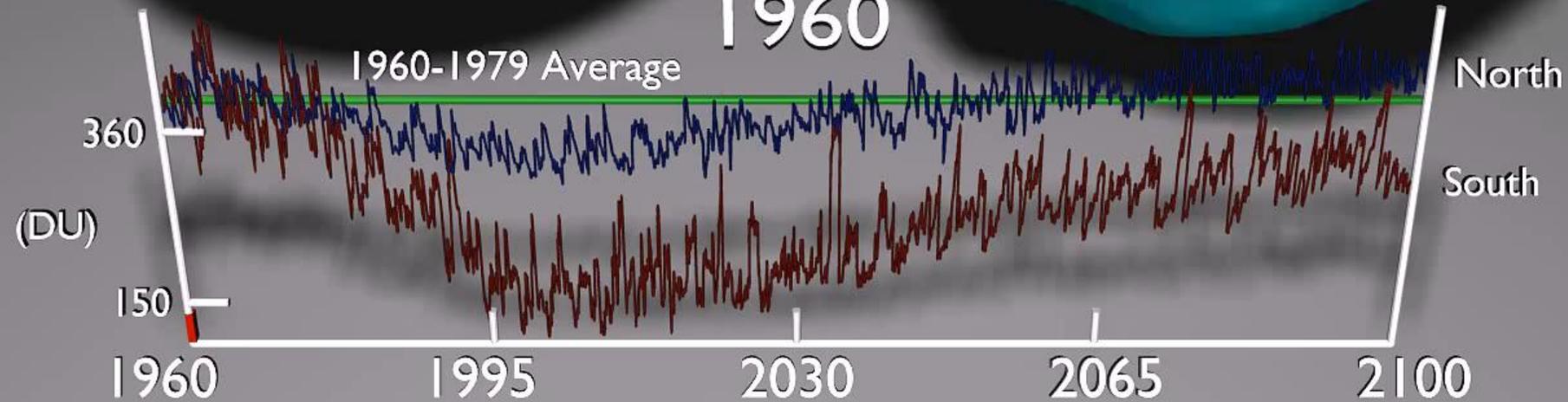


# Total Ozone Column

150  360  
Dobson Units (DU)

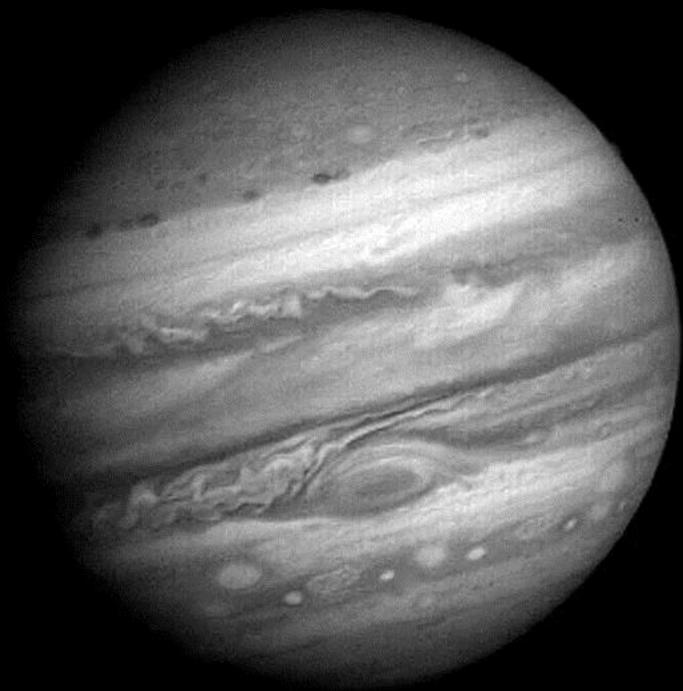


1960



### 范例三

范例四



**课堂讨论：你想从这门课中学到什么？**