ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN


## ĐỒ ÁN MÔN HỌC

# TOÁN ỨNG DỤNG VÀ THỐNG KÊ

## ĐỀ TÀI: Bài toán khí hậu
## Bài toán 2

**Giảng viên lý thuyết:** PGS.TS Nguyễn Đình Thúc
**Lớp:** 20TN
**Thành viên thực hiện:**

- 20120131 – Nguyễn Văn Lộc
- 20120536 – Võ Trọng Nghĩa
- 20120572 – Nguyễn Kiều Minh Tâm

THÀNH PHỐ HỒ CHÍ MINH, THÁNG 3-4 NĂM 2022

# Mục lục



# Danh sách hình vẽ



# Danh sách bảng

# Lời nói đầu

# 1 Đặt vấn đề

## 1.1 Xác định và hình thức hóa mục tiêu của bài toán

**Bài toán:** Dự báo nhiệt độ trung bình năm ở 5 trạm khí tượng theo dữ liệu của Cơ quan Quản lý Khí quyển và Đại dương Quốc gia Hoa Kỳ, bao gồm:

- **Trạm 1:** Trạm Lieksa Lampela ở Phần Lan (FIE00144982).
- **Trạm 2:** Trạm Jena Sternwarte ở Đức (GM000004204).
- **Trạm 3:** Trạm Den Helder 1 ở Hà Lan (NLM00006235).
- **Trạm 4:** Trạm Uppsala Aut ở Thụy Điển (SWE00139148).
- **Trạm 5:** Trạm New York City's Central Park ở Hoa Kỳ (USW00094728).

**Hình thức hóa mục tiêu của bài toán:** dùng biểu đồ phân tán, tìm các hệ số hồi quy.

## 1.2 Dạng bài toán

Bài toán này thuộc dạng bài toán **Dự đoán** (từ dữ liệu hiện có trong hiện tại và quá khứ để đưa ra dự đoán dữ liệu trong tương lai chưa biết).

## 1.3 Đối tượng được chọn cho bài toán

Dữ liệu về nhiệt độ trung bình năm của 5 trạm nêu trên.
Cột TAVG của các tập tin **FIE00144982.csv**, **GM000004204.csv**, **NLM00006235.csv**, **SWE00139148.csv**, **USW00094728.csv** trong thư mục **gsoy-latest**, lấy từ dataset về Global Summary of the Year của NOAA.

## 1.4 Phạm vi, mức độ, quy mô của bài toán

Theo không gian: dữ liệu được xử lý trong bài toán đại diện cho 5 bang/tỉnh ở 5 quốc gia khác nhau.
Theo thời gian: dữ liệu được thu thập trong vòng 30 năm, từ năm 1991 đến năm 2020.

# 2 Thu thập và xử lý dữ liệu

## 2.1 Thu thập dữ liệu

Dữ liệu xử lý trong bài toán này được thu thập từ dữ liệu của Cơ quan Quản lý Khí quyển và Đại dương Quốc gia Hoa Kỳ (NOAA), phần dữ liệu tổng hợp theo năm (Global Summary of the Year).

## 2.2 Xử lý dữ liệu

### 2.2.1 Trích xuất dữ liệu

Dữ liệu nhiệt độ trung bình năm từ các tập tin được trích xuất thành các tập dữ liệu **data0.csv**, **data1.csv**, **data2.csv**, **data3.csv**, **data4.csv**, theo thứ tự được nêu ra bên trên nhờ các hàm của module `pandas`.

### 2.2.2 Xử lý phần dữ liệu bị khuyết

Dữ liệu từ tập tin gốc đầu tiên (tập tin **FIE00144982.csv**) bị khuyết phần dữ liệu của năm 2009., do đó, nhóm chúng em đã tìm cách "điền" vào những ô bị khuyết này. Nhiệt độ trung bình năm của năm 2009 trong tập tin này được tính bằng trung bình cộng của nhiệt độ trung bình năm 2008 và nhiệt độ trung bình năm 2010.

```
Station:  ../Data/FIE00144982.csv
Missing data:  [2009]
```

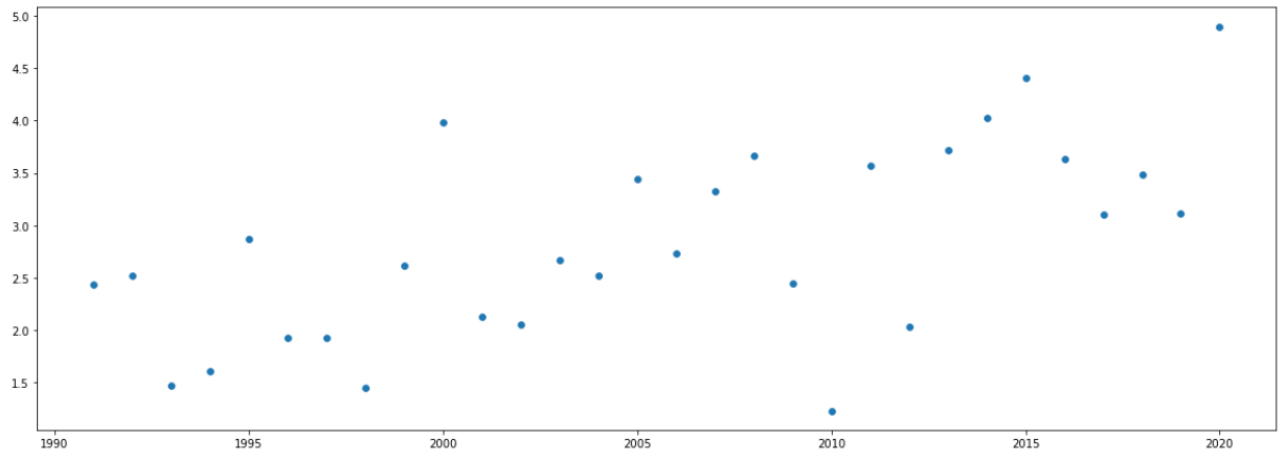
Hình 1: Phần dữ liệu bị khuyết

# 3 Phân tích, đánh giá và kết luận

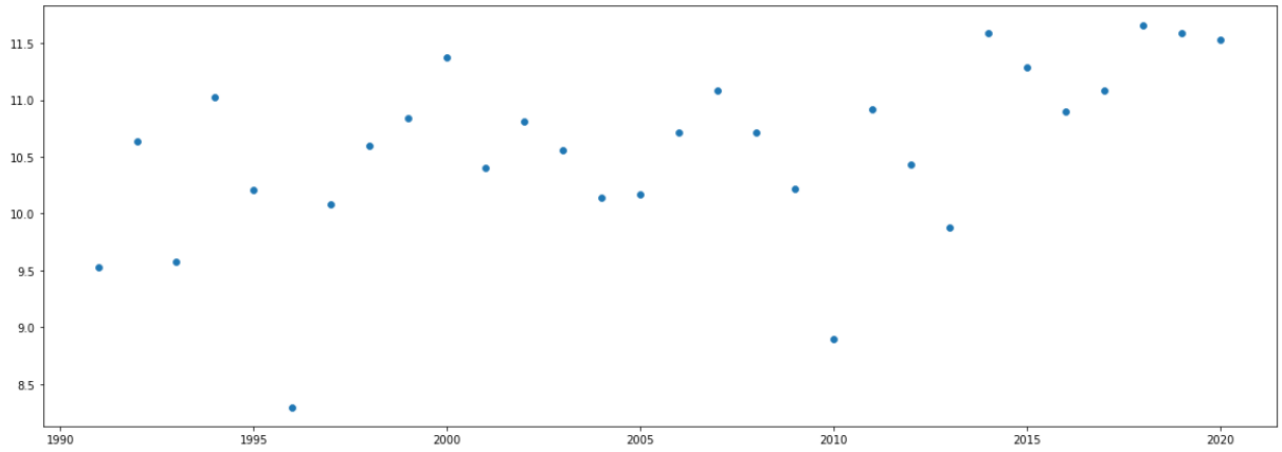
Sau khi được tiền xử lý, dữ liệu được trực quan hóa (visualized)thành biểu đồ phân tán nhờ hàm hỗ trợ của module `matplotlib`.

## 3.1 Biểu đồ phân tán của dữ liệu

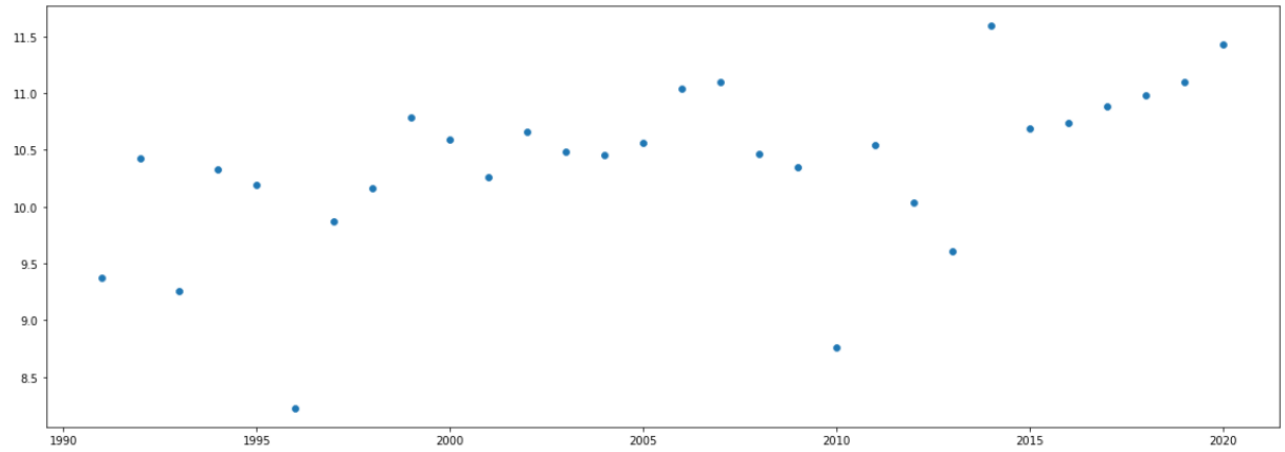
Biểu đồ phân tán về nhiệt độ trung bình năm đo được tại 5 trạm trên như sau:



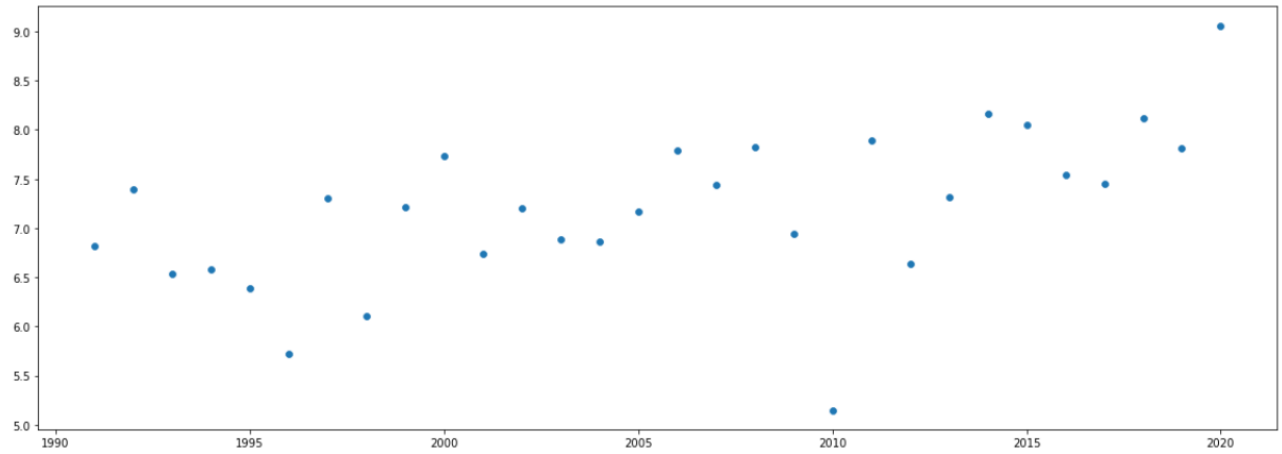

Hình 2: Biểu đồ phân tán dữ liệu của trạm 1



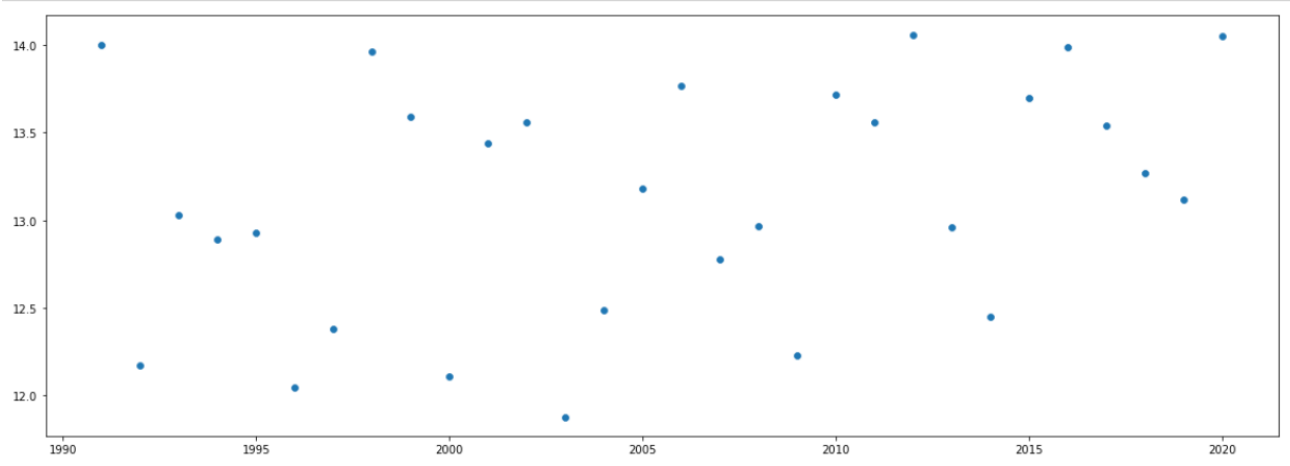

Hình 3: Biểu đồ phân tán dữ liệu của trạm 2




Hình 4: Biểu đồ phân tán dữ liệu của trạm 3




Hình 5: Biểu đồ phân tán dữ liệu của trạm 4




Hình 6: Biểu đồ phân tán dữ liệu của trạm 5

## 3.2 Tìm mô hình

Mô hình được chọn để dùng là mô hình **hồi quy** (regression).
Hàm hồi quy $f(x)$ được chọn từ tổ hợp có giá trị r-squared lớn nhất với các hệ số thích hợp từ những hàm $f_1(x) = 1$, $f_2(x) = x$, $f_3(x) = x^2$, $f_4(x) = x^3$, $f_5(x) = x^4$, $f_6(x) = \ln(x)$, $f_7(x) = \sin(x)$ và $f_8(x) = \ln(x) \cdot \sin(x)$.
Các hệ số của mô hình hồi quy được tìm như sau:
$x_1, x_2, ..., x_N$ là giá trị của các năm.

$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \in \mathbb{R}^N$ là các giá trị nhiệt độ trung bình năm được chọn.

$f_1(x), f_2(x), \ldots, f_p(x)$ là các hàm được chọn cho mô hình.

$$A = \begin{bmatrix} f_1(x_1) & f_2(x_1) & \ldots & f_p(x_1) \\ f_1(x_2) & f_2(x_2) & \ldots & f_p(x_2) \\ \vdots & \vdots & \vdots & \vdots \\ f_1(x_N) & f_2(x_N) & \ldots & f_p(x_N) \end{bmatrix} \in \mathbb{R}^{N \times p}.$$

$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_p \end{bmatrix} \in \mathbb{R}^p$ là ma trận các hệ số của mô hình hồi quy. Khi đó,

$$\theta_0 = \left(A^T A\right)^{-1} A^T Y.$$

Đoạn code tìm mô hình hồi quy như sau và vẽ đường hồi quy như sau:

```
from itertools import chain, combinations
from sklearn.metrics import r2_score
def model_generate(_col):
    return [
            (_col, 'x', lambda x: x),
            (np.power(_col, 2), 'x^2', lambda x: np.power(x, 2)),
            (np.power(_col, 3), 'x^3', lambda x: np.power(x, 3)),
```

```
            (np.power(_col, 4), 'x^4', lambda x: np.power(x, 4)),
            (np.log(_col), 'log(x)', lambda x: np.log(x)),
            (np.sin(_col), 'sin(x)', lambda x: np.sin(x)),
            (np.multiply(np.sin(_col), np.log(_col)), 'log(x) * sin(x)',
             lambda x: np.multiply(np.sin(x), np.log(x)))
    ]

def powerset(iterable):
    "powerset([1,2,3]) --> (1,) (2,) (3,) (1,2) (1,3) (2,3) (1,2,3)"
    s = list(iterable)
    return chain.from_iterable(combinations(s, r) for r in range(1, len(s)+1))

def calculate_theta(A, y):
    return np.linalg.inv(A.T * A) * A.T * y

def find_model(x, y):
    N = x.shape[0] # get number of rows
    _best_r2 = -np.Inf
    _best_model = None
    _best_theta = None
    _best_model_as_text = None
    for _model in powerset(model_generate(x)):
        A = np.ones((N, 1))
        for _elem in _model:
            A = np.concatenate((A, _elem[0]), axis = 1) # Merge columns
        try:
            theta = calculate_theta(A, y)
            y_hat = A * theta
            r2 = r2_score(np.squeeze(np.asarray(y, dtype=np.float64)),
                          np.squeeze(np.asarray(y_hat, dtype=np.float64)))
            if _best_r2 < r2:
                _best_model = [item[2] for item in _model]
                _best_model_as_text = [item[1] for item in _model]
                _best_r2 = r2
                _best_theta = theta
        except:
            continue
    _text = ""
    _text += str(float(_best_theta[0])) + " + "
    _text += "".join([str(np.round(a[0], 5)) + "*" + b + " + "
                      for a,b in zip(np.asarray(_best_theta[1:]),
                          _best_model_as_text)])[:-2]
    return (_best_r2, _text, _best_model, _best_theta)

def eval_value(_model, _theta, x):
    result = float(_theta[0])
    for model, theta in (zip(_model, _theta[1:])):
        result = result + float(model(x)) * float(theta)
    return result

def regression(file_name):
```

```
# regression model
df = pd.read_csv(file_name)
x = np.matrix(np.arange(1991, 2020 + 1), dtype = np.float64).T # create N
    * 1 matrix
y = np.matrix(list(df.iloc[:]['Temperature']), dtype = np.float64).T #
    create N * 1 matrix
plt.scatter(np.squeeze(np.asarray(x)),
            np.squeeze(np.asarray(y)),
            linewidths=0.6)


_r2, _text, _model, _theta = find_model(x, y)
print(f"Best r2: {_r2}")
print(f"Model: {_text}")

x_test = np.linspace(1990, 2050, 200)
y_test = np.array([eval_value(_model, _theta, _x) for _x in x_test])
plt.plot(x_test, y_test)
plt.show()
```