1er_examen

June 20, 2025

1 1er examen parcial

1.0.1 Tópicos Selectos De Matemáticas Aplicadas II: Análisis de Datos con Python

20 de junio de 2025 Alan Badillo Salas

1.0.2 Ejercicio 1. (20 puntos)

Se tiene un conjunto de datos $X = [x_1, x_2, \dots, x_n]$. El proceso de estandarización **min-max** transforma cada valor x_i al rango $[0, 1]$ mediante:

$$z_i = \frac{x_i - \min(X)}{\max(X) - \min(X)}$$

Una vez estandarizados, definimos el siguiente **índice de dispersión normalizado (IDN)**:

$$\text{IDN}(X) = \frac{1}{n} \sum_{i=1}^{n} (z_i - \bar{z})^2$$

donde $\bar{z}$ es el promedio del vector estandarizado Z_X.

i) Implementa una función que:

- Estandarice un vector X usando la fórmula min-max.
- Calcule el valor de $\text{IDN}(X)$.

ii) Usa tu función para calcular $\text{IDN}(X)$ para:

$$X = [12.4, 8.1, 15.7, 9.3, 14.8, 10.2, 11.6]$$

iii) Compara el valor de $\text{IDN}(X)$ con la **varianza normalizada** calculada directamente con `np.var()` en Python. ¿Coinciden los valores?

---

[76]:
```
def estandarizar_minmax(x):
    z = []
    xmin = min(x)
    xmax = max(x)
    for xi in x:
        zi = (xi - xmin) / (xmax - xmin)
```

```
        z.append(zi)
    return z

def idn(z):
    zp = sum(z) / len(z)
    w = []
    for zi in z:
        wi = (zi - zp) ** 2
        w.append(wi)
    wp = sum(w) / len(w)
    return wp
```

[77]:
```
x = [12.4, 8.1, 15.7, 9.3, 14.8, 10.2, 11.6]

x
```

[77]:
```
[12.4, 8.1, 15.7, 9.3, 14.8, 10.2, 11.6]
```

[78]:
```
z = estandarizar_minmax(x)

z
```

[78]:
```
[0.5657894736842106,
 0.0,
 1.0,
 0.15789473684210542,
 0.8815789473684212,
 0.2763157894736842,
 0.46052631578947373]
```

[79]:
```
idn(z)
```

[79]:
```
0.11642828876703037
```

[80]:
```
import numpy

numpy.var(x)
```

[80]:
```
np.float64(6.724897959183674)
```

La diferencia entre $IDN(X)$ y $Var(X)$ es alta

1.0.3 Ejercicio 2. (20 puntos)

Se tiene la siguiente relación entre las variables x, y y z:

$$\sin(x+y) = \tan(\frac{e^{\cos(x+y)} - \ln(z)}{2}).$$

Visualice la función $z = f(x, y)$ en el dominio $[-10, 10] \times [-10, 10]$, en el caso de que $x + y \leq \frac{1}{2}$, entonces $z = 20$. Para este ejercicio evalue la función en una malla de 500 puntos en cada dirección.

---

[81]:
```
import numpy

x = numpy.linspace(-10, 10, 500)
y = numpy.linspace(-10, 10, 500)

X, Y = numpy.meshgrid(x, y)

X.shape, Y.shape
```

[81]:
```
((500, 500), (500, 500))
```

Para calcular z tenemos que despejar la variable:

$$e^{cos(x+y)} - ln(z) = 2 \cdot tan^{-1}(sin(x + y))$$

$$ln(z) = e^{cos(x+y)} - 2 \cdot tan^{-1}(sin(x + y))$$

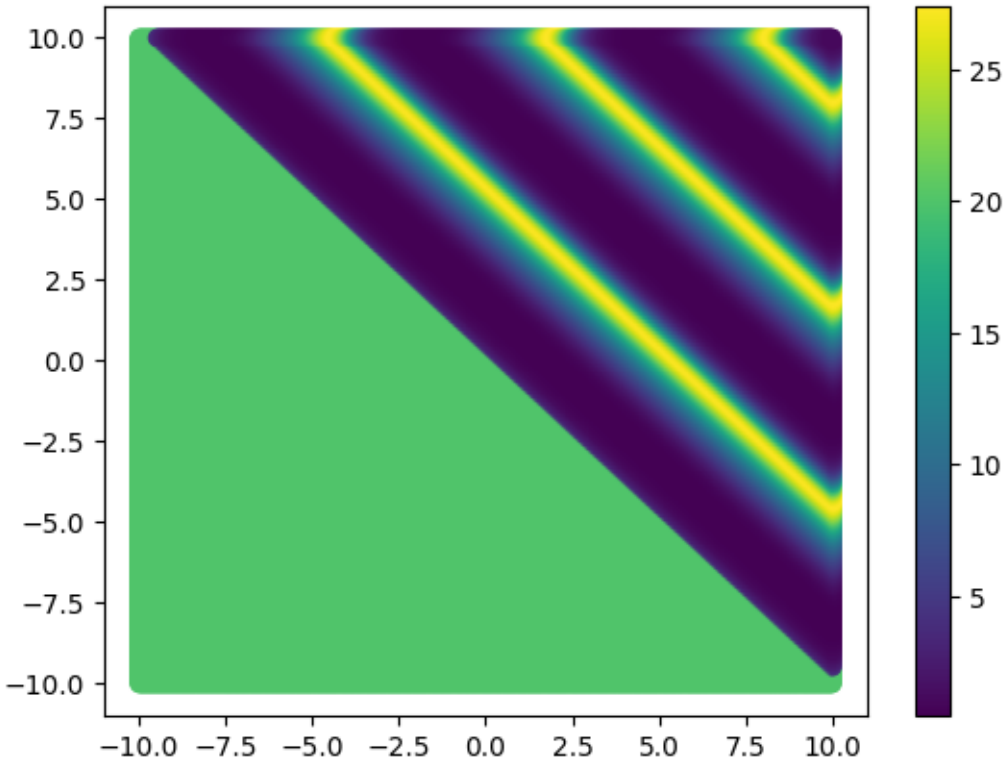
$$z = e^{e^{cos(x+y)} - 2 \cdot tan^{-1}(sin(x+y))}$$

[82]:
```
U = X + Y

Z = U * 0

Z = (U <= 0.5) * 20 + (U > 0.5) * numpy.exp(numpy.exp(numpy.cos(U)) - 2 * numpy.
↪atan(numpy.sin(U)))

Z
```

[82]:
```
array([[20.        , 20.        , 20.        , …, 20.        ,
        20.        , 20.        ],
       [20.        , 20.        , 20.        , …, 20.        ,
        20.        , 20.        ],
       [20.        , 20.        , 20.        , …, 20.        ,
        20.        , 20.        ],
       …,
       [20.        , 20.        , 20.        , …,  1.40224539,
         1.29109402,  1.1919243 ],
       [20.        , 20.        , 20.        , …,  1.29109402,
         1.1919243 ,  1.10341173],
       [20.        , 20.        , 20.        , …,  1.1919243 ,
         1.10341173,  1.02438041]])
```

[83]:
```
import matplotlib.pyplot as pyplot

pyplot.scatter(X, Y, c=Z)
pyplot.colorbar()
```

[83]:
```
<matplotlib.colorbar.Colorbar at 0x128c3e390>
```



1.0.4 Ejercicio 3. (20 puntos)

Utilice los datos de los nacimientos por estado del 2020 al 2022 del repositorio en Github y realice lo siguiente:

a) Obtenga un Dataframe que tenga como columnas el número de nacimientos de mujeres, el número de nacimientos de hombres y el número total de nacimientos por año durante el 2020 al 2022. Obtenga una gráfica de cada categoría.

b) Agregue a los datos de nacimientos del 2020 la columna de población por estado y la tasa de nacimientos por cada mil habitantes. Gráfique la tasa de nacimientos por estado.

---

[84]:
```
import pandas

nacimientos = []
```

```
for anio in [2020, 2021, 2022]:
    nacimientos_anio = pandas.read_csv(f"nacimientos_{anio}.csv")
    nacimientos_anio["Año"] = anio
    nacimientos.append(nacimientos_anio)

nacimientos_2020_2022 = pandas.concat(nacimientos)

nacimientos_2020_2022
```

[84]:

	Estado	Abreviatura	Regiones	Mujeres	Hombres	No_esp	Total	Año
0	Aguascalientes	AG	Noreste	9966	10404	0	20370	2020
1	Baja California	BC	Noroeste	23539	24406	0	47945	2020
2	Baja California Sur	BS	Noroeste	4982	5099	0	10081	2020
3	Campeche	CM	Sureste	5275	5454	0	10729	2020
4	Coahuila de Zaragoza	CO	Noreste	22594	23221	0	45815	2020
..	…	…	…	…	…	…	…	…
27	Tamaulipas	TM	Noreste	23177	24110	0	47287	2022
28	Tlaxcala	TL	Centro-Sur	9653	9964	1	19618	2022
29	Veracruz	VE	Sureste	53782	55329	0	109111	2022
30	Yucatan	YU	Sureste	13829	14103	0	27932	2022
31	Zacatecas	ZA	Noreste	13368	13402	0	26770	2022

[96 rows x 8 columns]

[85]:
```
reporte1 = nacimientos_2020_2022[["Año", "Mujeres", "Hombres", "Total"]].copy()

reporte1
```

[85]:

	Año	Mujeres	Hombres	Total
0	2020	9966	10404	20370
1	2020	23539	24406	47945
2	2020	4982	5099	10081
3	2020	5275	5454	10729

```
4   2020    22594    23221   45815
..   …       …       …       …
27  2022    23177    24110   47287
28  2022     9653     9964   19618
29  2022    53782    55329  109111
30  2022    13829    14103   27932
31  2022    13368    13402   26770

[96 rows x 4 columns]
```

[86]:
```
reporte1.groupby("Año").sum()
```

[86]:
```
      Mujeres  Hombres    Total
Año
2020   798452   827072  1625527
2021   938302   965088  1903393
2022   924579   956137  1880718
```

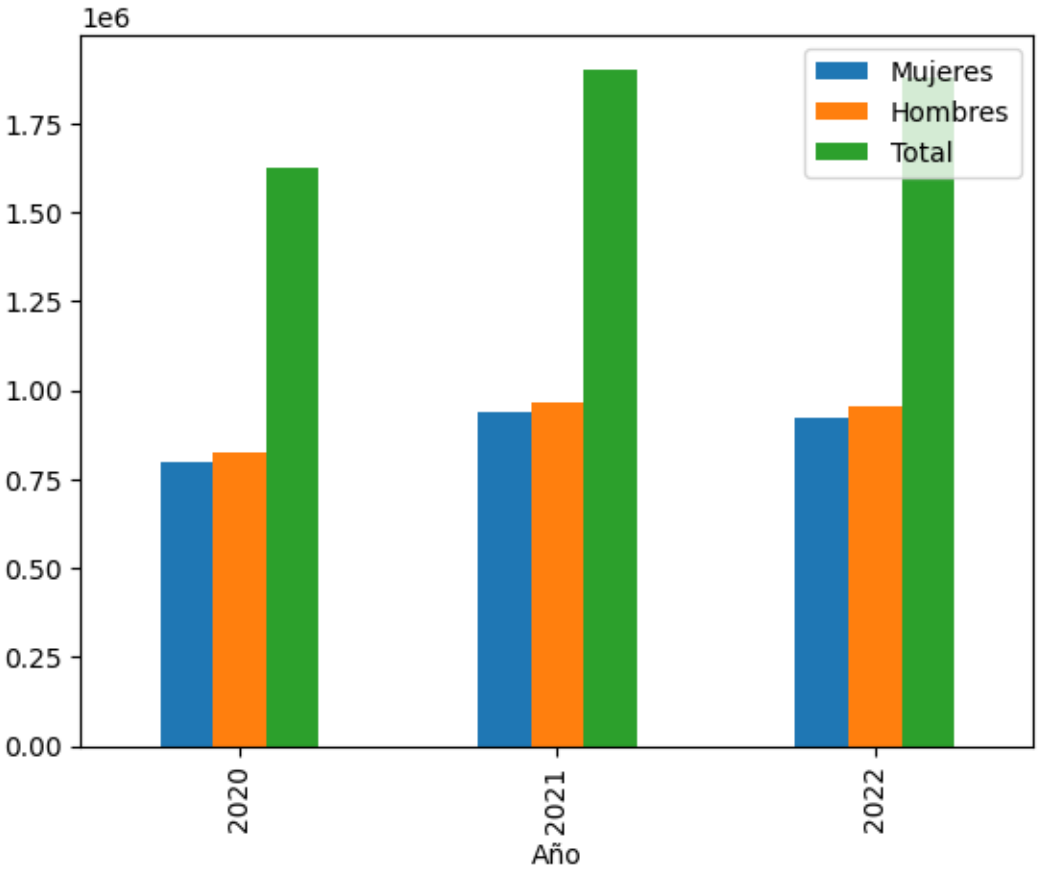
[87]:
```
reporte1.sum()
```

[87]:
```
Año          194016
Mujeres     2661333
Hombres     2748297
Total       5409638
dtype: int64
```

Entre el 2020 y 2022 nacieron $2,661,333$ mujeres y $2,748,297$ hombres, y en total $5,409,638$ hombres, mujeres y no especificados.

[88]:
```
reporte1.groupby("Año").sum().plot.bar()
```

[88]:
```
<Axes: xlabel='Año'>
```



[89]:
```
nacimientos_2020 = nacimientos[0]

nacimientos_2020
```

[89]:

	Estado	Abreviatura	Regiones	Mujeres	Hombres	No_esp	\
0	Aguascalientes	AG	Noreste	9966	10404	0	
1	Baja California	BC	Noroeste	23539	24406	0	
2	Baja California Sur	BS	Noroeste	4982	5099	0	
3	Campeche	CM	Sureste	5275	5454	0	
4	Coahuila de Zaragoza	CO	Noreste	22594	23221	0	
5	Colima	CL	Occidente	4325	4623	0	
6	Chiapas	CS	Sureste	47962	50136	0	
7	Chihuahua	CH	Noroeste	23436	24659	0	
8	CDMX	CX	Centro-Sur	32218	33485	0	
9	Durango	DG	Noroeste	13713	14275	0	
10	Guanajuato	GT	Noreste	41979	43278	2	
11	Guerrero	GR	Centro-Sur	29174	29976	0	
12	Hidalgo	HG	Occidente	18487	19011	0	
13	Jalisco	JC	Noroeste	58439	61165	0	

14	Mexico	EM	Occidente	90657	93790	0
15	Michoacan	MI	Occidente	36028	37086	1
16	Morelos	MO	Centro-Sur	12137	12525	0
17	Nayarit	NaN	Noroeste	8153	8371	0
18	Nuevo Leon	NL	Noreste	36678	37793	0
19	Oaxaca	OA	Sureste	25497	26534	0
20	Puebla	PU	Centro-Sur	51363	52865	0
21	Queretaro	QT	Noreste	16318	16893	0
22	Quintana Roo	QR	Sureste	11864	12199	0
23	San Luis Potosi	SL	Noreste	20432	21380	0
24	Sinaloa	SI	Noroeste	19080	19891	0
25	Sonora	SO	Noroeste	15005	15899	0
26	Tabasco	TB	Sureste	14289	14654	0
27	Tamaulipas	TM	Noreste	21876	22421	0
28	Tlaxcala	TL	Centro-Sur	10139	10471	0
29	Veracruz	VE	Sureste	45582	46743	0
30	Yucatan	YU	Sureste	14375	14946	0
31	Zacatecas	ZA	Noreste	12890	13419	0

	Total	Año
0	20370	2020
1	47945	2020
2	10081	2020
3	10729	2020
4	45815	2020
5	8948	2020
6	98098	2020
7	48095	2020
8	65703	2020
9	27988	2020
10	85259	2020
11	59150	2020
12	37498	2020
13	119604	2020
14	184447	2020
15	73115	2020
16	24662	2020
17	16524	2020
18	74471	2020
19	52031	2020
20	104228	2020
21	33211	2020
22	24063	2020
23	41812	2020
24	38971	2020
25	30904	2020
26	28943	2020

```
27   44297  2020
28   20610  2020
29   92325  2020
30   29321  2020
31   26309  2020
```

[90]:
```
poblacion = pandas.read_csv("poblacion.csv")

poblacion
```

[90]:

	Estado	Regiones	H_1990	M_1990	H_2000	M_2000	\
0	Aguascalientes	Noreste	350218	369441	456533	487752	
1	Baja California	Noroeste	832090	828765	1252581	1234786	
2	Baja California Sur	Noroeste	161833	155931	216250	207791	
3	Campeche	Sureste	268772	266413	344334	346355	
4	Coahuila de Zaragoza	Noreste	979097	993243	1140195	1157875	
5	Colima	Occidente	212543	215967	268192	274435	
6	Chiapas	Sureste	1604773	1605723	1941880	1979012	
7	Chihuahua	Noroeste	1213302	1228571	1519972	1532935	
8	CDMX	Centro-Sur	3939911	4295833	4110485	4494754	
9	Durango	Noroeste	664766	684612	709521	739140	
10	Guanajuato	Noreste	1926735	2055858	2233315	2429717	
11	Guerrero	Centro-Sur	1282220	1338417	1491287	1588362	
12	Hidalgo	Occidente	929138	959228	1081993	1153598	
13	Jalisco	Noroeste	2564892	2737797	3070241	3251761	
14	Mexico	Occidente	4834549	4981246	6407213	6689473	
15	Michoacan	Occidente	1718763	1829436	1911078	2074589	
16	Morelos	Centro-Sur	583785	611274	750799	804497	
17	Nayarit	Noroeste	411057	413586	456105	464080	
18	Nuevo Leon	Noreste	1542664	1556072	1907939	1926202	
19	Oaxaca	Sureste	1477438	1542122	1657406	1781359	
20	Puebla	Centro-Sur	2008531	2117570	2448801	2627885	
21	Queretaro	Noreste	516168	535067	680966	723340	
22	Quintana Roo	Sureste	254908	238369	448308	426655	
23	San Luis Potosi	Noreste	987315	1015872	1120837	1178523	
24	Sinaloa	Noroeste	1101621	1102433	1264143	1272701	
25	Sonora	Noroeste	915088	908518	1110590	1106379	
26	Tabasco	Sureste	749982	751762	934515	957314	
27	Tamaulipas	Noreste	1111698	1137883	1359874	1393348	
28	Tlaxcala	Centro-Sur	375130	386147	469948	492698	
29	Veracruz	Sureste	3077427	3150812	3355164	3553811	
30	Yucatan	Sureste	673892	689048	818205	840005	
31	Zacatecas	Noreste	623663	652660	653583	700027	

	H_2010	M_2010	H_2020	M_2020
0	576638	608358	696683	728924
1	1591610	1563460	1900589	1868431

```
2     325433   311593   405879   392568
3     407721   414720   456939   471424
4    1364197  1384194  1563669  1583102
5     322790   327765   360622   370769
6    2352807  2443773  2705947  2837881
7    1692545  1713920  1853822  1888047
8    4233783  4617297  4404927  4805017
9     803890   829044   904866   927784
10   2639425  2846947  2996454  3170480
11   1645561  1743207  1700612  1840073
12   1285222  1379796  1481379  1601462
13   3600641  3750041  4098455  4249696
14   7396986  7778876  8251295  8741123
15   2102109  2248928  2306341  2442505
16    858588   918639   950847  1020673
17    541007   543972   612278   623178
18   2320185  2333273  2890950  2893492
19   1819008  1982954  1974843  2157305
20   2769855  3009974  3160115  3423163
21    887188   940749  1156820  1211647
22    673220   652358   936779   921206
23   1260366  1325152  1372451  1449804
24   1376201  1391560  1494815  1532128
25   1339612  1322868  1472197  1472643
26   1100758  1137845  1173671  1228927
27   1616201  1652353  1736140  1791595
28    565775   604161   649894   693083
29   3695679  3947515  3871774  4190805
30    963333   992244  1140279  1180619
31    726897   763771   791058   831080
```

Verifiquemos que los estados estén en el mismo orden (sean los mismos)

[91]:
```
(nacimientos_2020["Estado"] != poblacion["Estado"]).sum()
```

[91]:
```
np.int64(0)
```

[92]:
```
nacimientos_2020["Población Mujeres"] = poblacion["M_2020"]
nacimientos_2020["Población Hombres"] = poblacion["H_2020"]

nacimientos_2020["Población"] = nacimientos_2020["Población Mujeres"] +␣
 ↪nacimientos_2020["Población Hombres"]

nacimientos_2020
```

[92]:
```
                 Estado Abreviatura    Regiones  Mujeres  Hombres  No_esp  \
0          Aguascalientes          AG     Noreste     9966    10404       0
1         Baja California          BC    Noroeste    23539    24406       0
```

2	Baja California Sur	BS	Noroeste	4982	5099	0
3	Campeche	CM	Sureste	5275	5454	0
4	Coahuila de Zaragoza	CO	Noreste	22594	23221	0
5	Colima	CL	Occidente	4325	4623	0
6	Chiapas	CS	Sureste	47962	50136	0
7	Chihuahua	CH	Noroeste	23436	24659	0
8	CDMX	CX	Centro-Sur	32218	33485	0
9	Durango	DG	Noroeste	13713	14275	0
10	Guanajuato	GT	Noreste	41979	43278	2
11	Guerrero	GR	Centro-Sur	29174	29976	0
12	Hidalgo	HG	Occidente	18487	19011	0
13	Jalisco	JC	Noroeste	58439	61165	0
14	Mexico	EM	Occidente	90657	93790	0
15	Michoacan	MI	Occidente	36028	37086	1
16	Morelos	MO	Centro-Sur	12137	12525	0
17	Nayarit	NaN	Noroeste	8153	8371	0
18	Nuevo Leon	NL	Noreste	36678	37793	0
19	Oaxaca	OA	Sureste	25497	26534	0
20	Puebla	PU	Centro-Sur	51363	52865	0
21	Queretaro	QT	Noreste	16318	16893	0
22	Quintana Roo	QR	Sureste	11864	12199	0
23	San Luis Potosi	SL	Noreste	20432	21380	0
24	Sinaloa	SI	Noroeste	19080	19891	0
25	Sonora	SO	Noroeste	15005	15899	0
26	Tabasco	TB	Sureste	14289	14654	0
27	Tamaulipas	TM	Noreste	21876	22421	0
28	Tlaxcala	TL	Centro-Sur	10139	10471	0
29	Veracruz	VE	Sureste	45582	46743	0
30	Yucatan	YU	Sureste	14375	14946	0
31	Zacatecas	ZA	Noreste	12890	13419	0

	Total	Año	Población Mujeres	Población Hombres	Población
0	20370	2020	728924	696683	1425607
1	47945	2020	1868431	1900589	3769020
2	10081	2020	392568	405879	798447
3	10729	2020	471424	456939	928363
4	45815	2020	1583102	1563669	3146771
5	8948	2020	370769	360622	731391
6	98098	2020	2837881	2705947	5543828
7	48095	2020	1888047	1853822	3741869
8	65703	2020	4805017	4404927	9209944
9	27988	2020	927784	904866	1832650
10	85259	2020	3170480	2996454	6166934
11	59150	2020	1840073	1700612	3540685
12	37498	2020	1601462	1481379	3082841
13	119604	2020	4249696	4098455	8348151
14	184447	2020	8741123	8251295	16992418

15	73115	2020	2442505	2306341	4748846
16	24662	2020	1020673	950847	1971520
17	16524	2020	623178	612278	1235456
18	74471	2020	2893492	2890950	5784442
19	52031	2020	2157305	1974843	4132148
20	104228	2020	3423163	3160115	6583278
21	33211	2020	1211647	1156820	2368467
22	24063	2020	921206	936779	1857985
23	41812	2020	1449804	1372451	2822255
24	38971	2020	1532128	1494815	3026943
25	30904	2020	1472643	1472197	2944840
26	28943	2020	1228927	1173671	2402598
27	44297	2020	1791595	1736140	3527735
28	20610	2020	693083	649894	1342977
29	92325	2020	4190805	3871774	8062579
30	29321	2020	1180619	1140279	2320898
31	26309	2020	831080	791058	1622138

[93]:
```
nacimientos_2020["Tasa nacimientos"] = (nacimientos_2020["Total"] /␣
↪nacimientos_2020["Población"]) * 1000

nacimientos_2020[["Estado", "Mujeres", "Hombres", "Total", "Población", "Tasa␣
↪nacimientos"]]
```

[93]:

	Estado	Mujeres	Hombres	Total	Población	\
0	Aguascalientes	9966	10404	20370	1425607	
1	Baja California	23539	24406	47945	3769020	
2	Baja California Sur	4982	5099	10081	798447	
3	Campeche	5275	5454	10729	928363	
4	Coahuila de Zaragoza	22594	23221	45815	3146771	
5	Colima	4325	4623	8948	731391	
6	Chiapas	47962	50136	98098	5543828	
7	Chihuahua	23436	24659	48095	3741869	
8	CDMX	32218	33485	65703	9209944	
9	Durango	13713	14275	27988	1832650	
10	Guanajuato	41979	43278	85259	6166934	
11	Guerrero	29174	29976	59150	3540685	
12	Hidalgo	18487	19011	37498	3082841	
13	Jalisco	58439	61165	119604	8348151	
14	Mexico	90657	93790	184447	16992418	
15	Michoacan	36028	37086	73115	4748846	
16	Morelos	12137	12525	24662	1971520	
17	Nayarit	8153	8371	16524	1235456	
18	Nuevo Leon	36678	37793	74471	5784442	
19	Oaxaca	25497	26534	52031	4132148	
20	Puebla	51363	52865	104228	6583278	
21	Queretaro	16318	16893	33211	2368467	

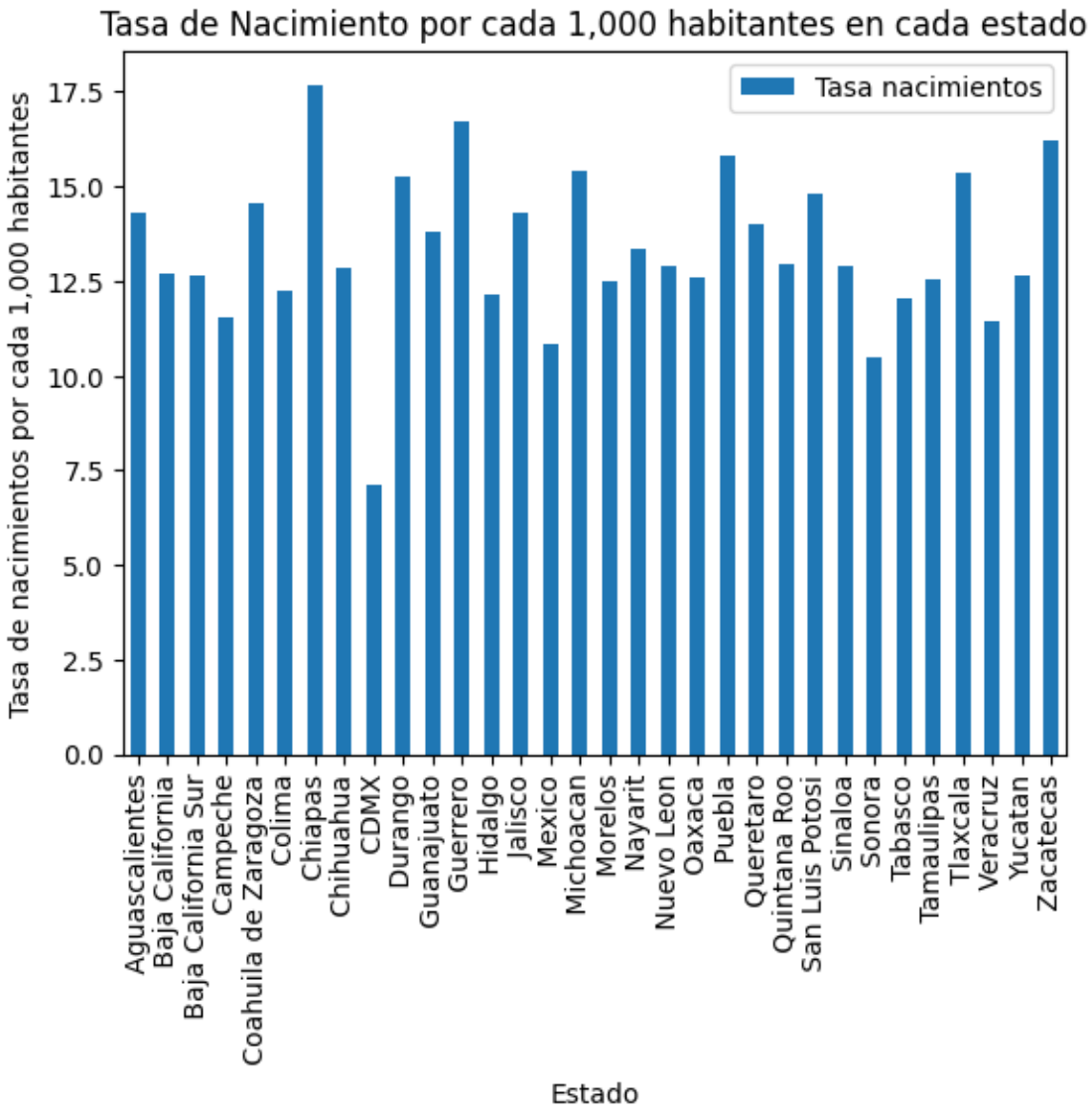
```
22           Quintana Roo    11864    12199    24063    1857985
23        San Luis Potosi    20432    21380    41812    2822255
24                Sinaloa    19080    19891    38971    3026943
25                 Sonora    15005    15899    30904    2944840
26                Tabasco    14289    14654    28943    2402598
27             Tamaulipas    21876    22421    44297    3527735
28                Tlaxcala    10139    10471    20610    1342977
29                Veracruz    45582    46743    92325    8062579
30                 Yucatan    14375    14946    29321    2320898
31              Zacatecas    12890    13419    26309    1622138
```

```
    Tasa nacimientos
0          14.288650
1          12.720813
2          12.625760
3          11.556902
4          14.559369
5          12.234222
6          17.694993
7          12.853203
8           7.133920
9          15.271874
10         13.825184
11         16.705807
12         12.163456
13         14.327005
14         10.854665
15         15.396372
16         12.509130
17         13.374819
18         12.874362
19         12.591756
20         15.832234
21         14.022150
22         12.951127
23         14.815104
24         12.874706
25         10.494288
26         12.046543
27         12.556782
28         15.346503
29         11.451051
30         12.633472
31         16.218719
```

[97]:
```
nacimientos_2020[["Estado", "Tasa nacimientos"]].plot.bar("Estado")
pyplot.title("Tasa de Nacimiento por cada 1,000 habitantes en cada estado")
```

```
pyplot.ylabel("Tasa de nacimientos por cada 1,000 habitantes")
```

[97]: `Text(0, 0.5, 'Tasa de nacimientos por cada 1,000 habitantes')`



1.0.5 Ejercicio 4. (20 puntos)

Guarde en un Dataframe los datos de los precios de las acciones de Tesla (*tesla.csv* en Github).

El **retorno simple** mide el cambio porcentual entre el precio de un activo financiero en dos momentos consecutivos. Es útil para analizar la variación de precios, por ejemplo, el precio de cierre diario de una acción. Dado un precio de cierre $ P_t $ en el tiempo $ t $, el retorno simple se calcula como:

$$r_t = \frac{P_t - P_{t-1}}{P_{t-1}} = \frac{P_t}{P_{t-1}} - 1$$

donde:

- $ r_t $: retorno simple en el tiempo $ t $,
- $ P_t $: precio de cierre en el tiempo $ t $,
- $ P_{t-1} $: precio de cierre en el tiempo anterior.

a) Agregue la columna correspondiente a la serie de **retornos simples** (utilice la función *.pct_change()*) y elimine la primera fila del *DataFrame*.

b) Grafique la serie de tiempo de los retornos simples y obtenga:

- El retorno simple más alto,
- El retorno simple más bajo,
- Las fechas en que ocurrieron.

c) Obtenga el histograma de frecuencia de la serie de retornos simples.
¿Estos se distribuyen normalmente? Justifique su respuesta calculando el coeficiente de asimetría, kurtosis y haciendo la prueba de Shapiro–Wilk.

d) Genere las gráficas de caja por año de los precios de cierre.

---

[122]:
```
import pandas

tesla = pandas.read_csv("tesla.csv", index_col=0, parse_dates=True)

tesla.head()
```

[122]:

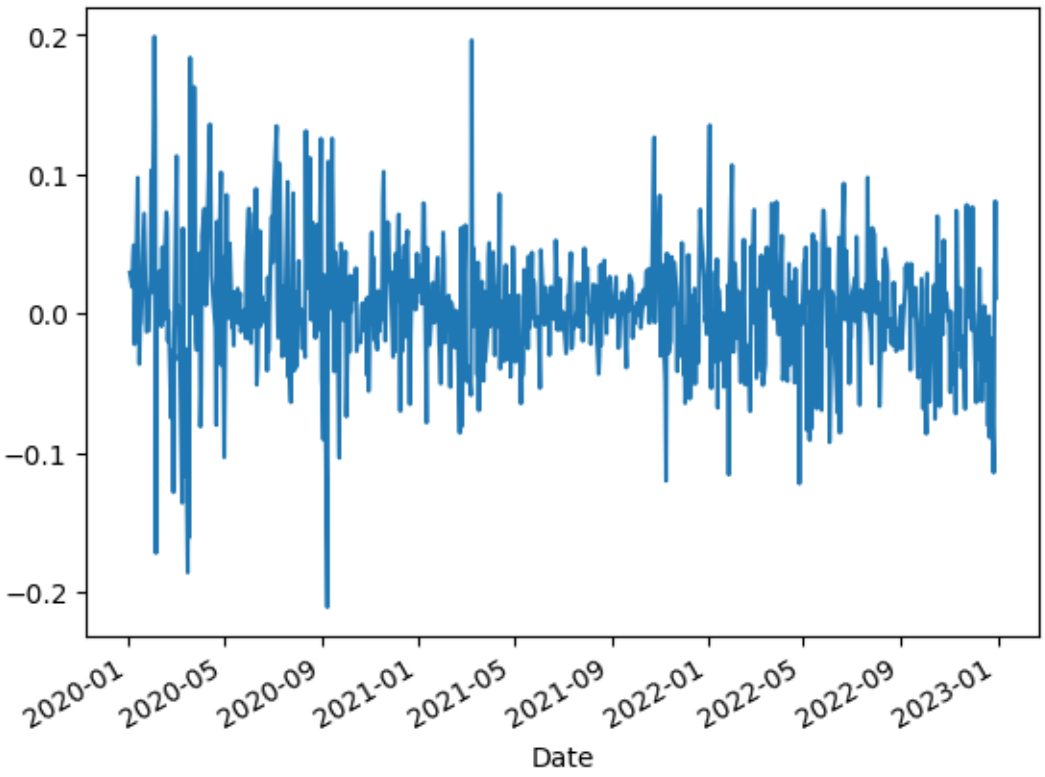
Date	Open	High	Low	Close	Adj Close
2020-01-02	28.299999	28.713333	28.114000	28.684000	28.684000
2020-01-03	29.366667	30.266666	29.128000	29.534000	29.534000
2020-01-06	29.364668	30.104000	29.333332	30.102667	30.102667
2020-01-07	30.760000	31.441999	30.224001	31.270666	31.270666
2020-01-08	31.580000	33.232666	31.215334	32.809334	32.809334

[123]:
```
r = tesla["Close"].pct_change().iloc[1:]

r.head()
```

[123]:
```
Date
2020-01-03    0.029633
2020-01-06    0.019255
2020-01-07    0.038801
2020-01-08    0.049205
2020-01-09   -0.021945
Name: Close, dtype: float64
```

[125]:
```
r.plot.line()
```

[125]:
```
<Axes: xlabel='Date'>
```



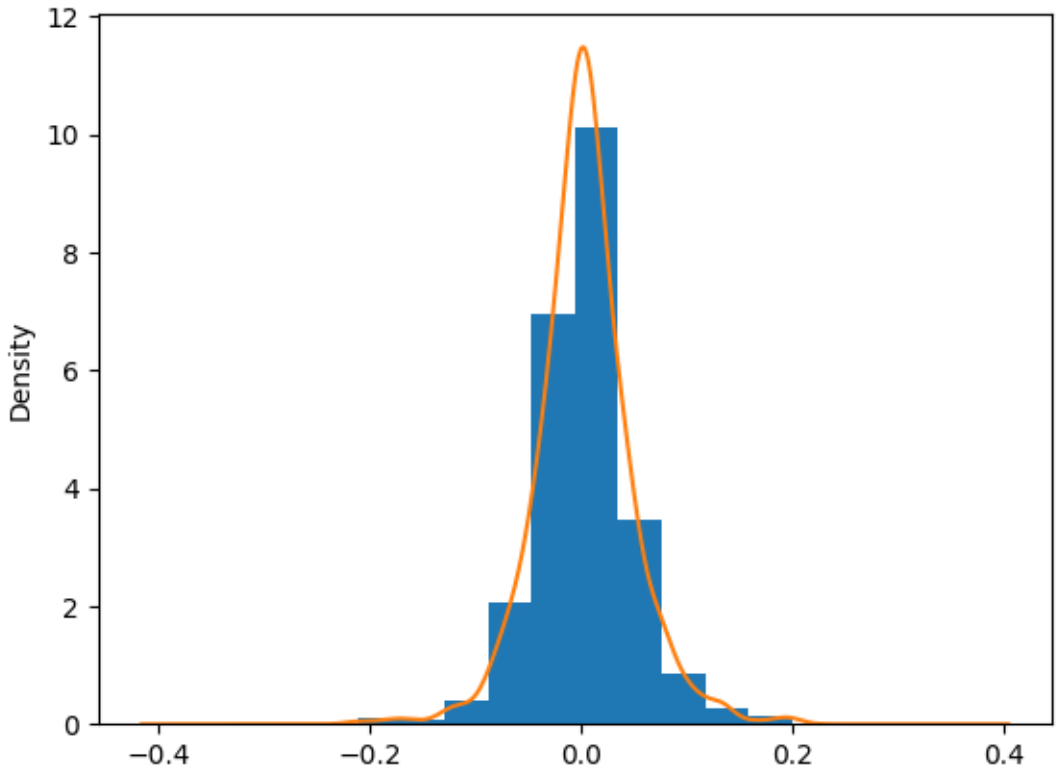
[133]:
```
imin = r.argmin()
imax = r.argmax()

pandas.DataFrame([
    (r.index[imin], r.iloc[imin]),
    (r.index[imax], r.iloc[imax]),
], columns=["Fecha", "Retorno"], index=["Mínimo", "Máximo"])
```

[133]:

	Fecha	Retorno
Mínimo	2020-09-08	-0.210628
Máximo	2020-02-03	0.198949

[138]:
```
r.plot.hist(density=True)
r.plot.density()
```

[138]:
```
<Axes: ylabel='Density'>
```



Aparentemente se distribuye normal

Asimetría

$$\eta = \frac{1}{n}\sum_{i=1}^{n}\frac{(x_i - \overline{x})^3}{S^3}$$

Tipo de sesgo:

- $\eta > 0$ - Sesgo positivo
- $\eta \approx 0$ - Casi insesgado
- $\eta < 0$ - Sesgo negativo

[139]:
```
r.skew()
```

[139]:
```
np.float64(0.08619747048670622)
```

Observamos que los datos son casi insegados

Curtosis

$$\beta = \frac{1}{n}\sum_{i=1}^{n}\frac{(x_i - \overline{x})^4}{S^4}$$

Tipos de curtosis:

- **Leptokurtic** - Estrecha ($\beta > 0$)
- **Mesokurtic** - Normal ($\beta \approx 0$)
- **Platykurtic** - Oval ($\beta < 0$)

[140]:
```
r.kurt()
```

[140]:
```
np.float64(2.7456294768903464)
```

Observamos que la curtosis es del tipo leptocurtosis

Prueba de Shapiro-Wilk

1. H_0: La muestra proviene de una distribución normal, la aceptamos con $p \geq 0.05$
2. H_1: La muestra no proviene de una distribución normal, si $p < 0.05$

[135]:
```
from scipy.stats import shapiro

r_value, p_value = shapiro(r)
print(f"r = {r_value:.4f} | p-value = {p_value:.10f}")
```

```
r = 0.9644 | p-value = 0.0000000000
```

Se rechaza la hipótesis nula H_0 dado que el valor-p es menor a 0.05

1.0.6 Ejercicio 5. (20 puntos)

Guarde en un *DataFrame* los datos de los 6 contaminantes principales de la ciudad de Puebla a partir del año 2016 (*contam_PUE_ugm3.csv* en GitHub). A continuación, realice lo siguiente:

a) Obtenga las gráficas de series de tiempo** de todos los contaminantes, los gráficos de caja, y calcule el número de datos atípicos de cada uno.

b) Realice una prueba de contraste de normalidad para los datos. Obtenga una matriz de correlación y un mapa de calor de los contaminantes utilizando el coeficiente de correlación apropiado.

c) Obtenga un *DataFrame* que muestre la correlación 2 a 2 entre los contaminantes.
Grafique 3 diagramas de dispersión** entre los contaminantes que presentan mayor correlación, ya sea positiva o negativa.

d) Aplique el test de Mann-Kendall a todos los contaminantes y obtenga una gráfica con la línea de tendencia de cada uno (en caso de que exista).

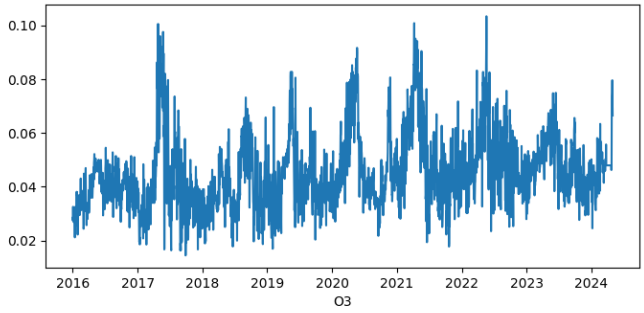
e) Compruebe la existencia de tendencia mediante el método ITA (Innovative Trend Analysis).

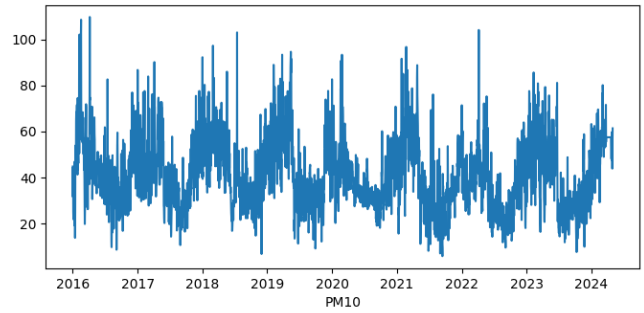
[142]:
```
import pandas

contam = pandas.read_csv("contam_PUE_ugm3.csv", index_col=0, parse_dates=True)

contam.head()
```

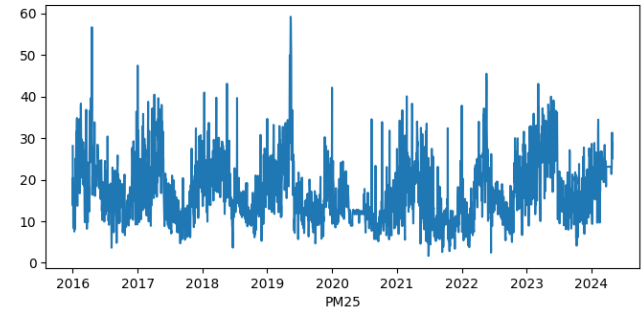
[142]:

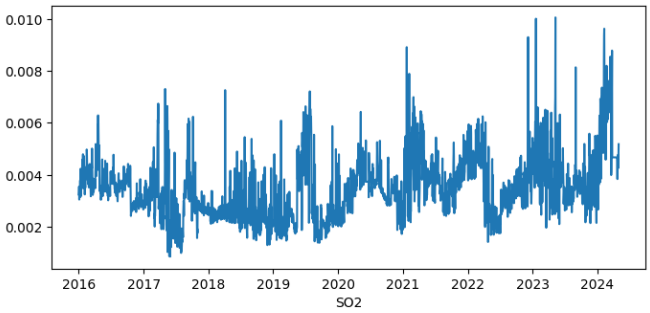
Fecha	O3	PM10	PM25	SO2	CO	NO2
2016-01-01	0.028414	44.694444	28.097222	0.003244	1.763388	0.020635
2016-01-02	0.027376	31.527778	13.791667	0.003339	1.796858	0.022468
2016-01-03	0.032512	44.333333	20.430556	0.003551	1.845788	0.023123
2016-01-04	0.028332	24.990338	9.569444	0.003051	1.756417	0.020897
2016-01-05	0.026939	21.763889	10.638889	0.003296	1.782832	0.021215

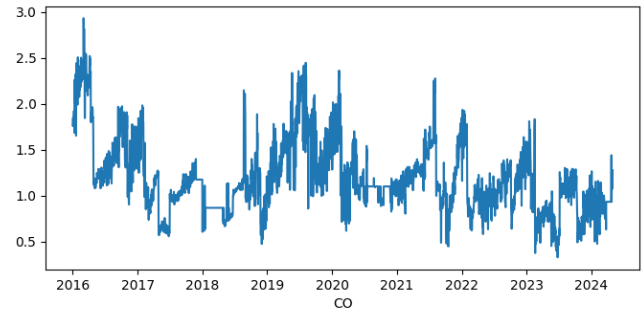
[171]:
```
import matplotlib.pyplot as pyplot

figure, axis = pyplot.subplots(len(contam.columns), 1, figsize=(8, 25))

for i, column in enumerate(contam.columns):
    axis[i].plot(contam.index, contam[column])
    axis[i].set_xlabel(column)
```

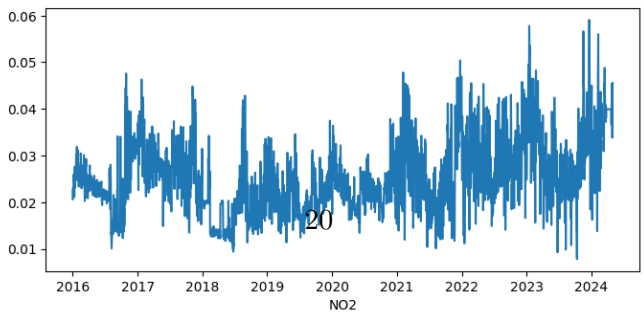

0.10
0.08
0.06
0.04
0.02
2016
2017
2018
2019
2020
2021
2022
2023
2024
O3


100
80
60
40
20
2016
2017
2018
2019
2020
2021
2022
2023
2024
PM10

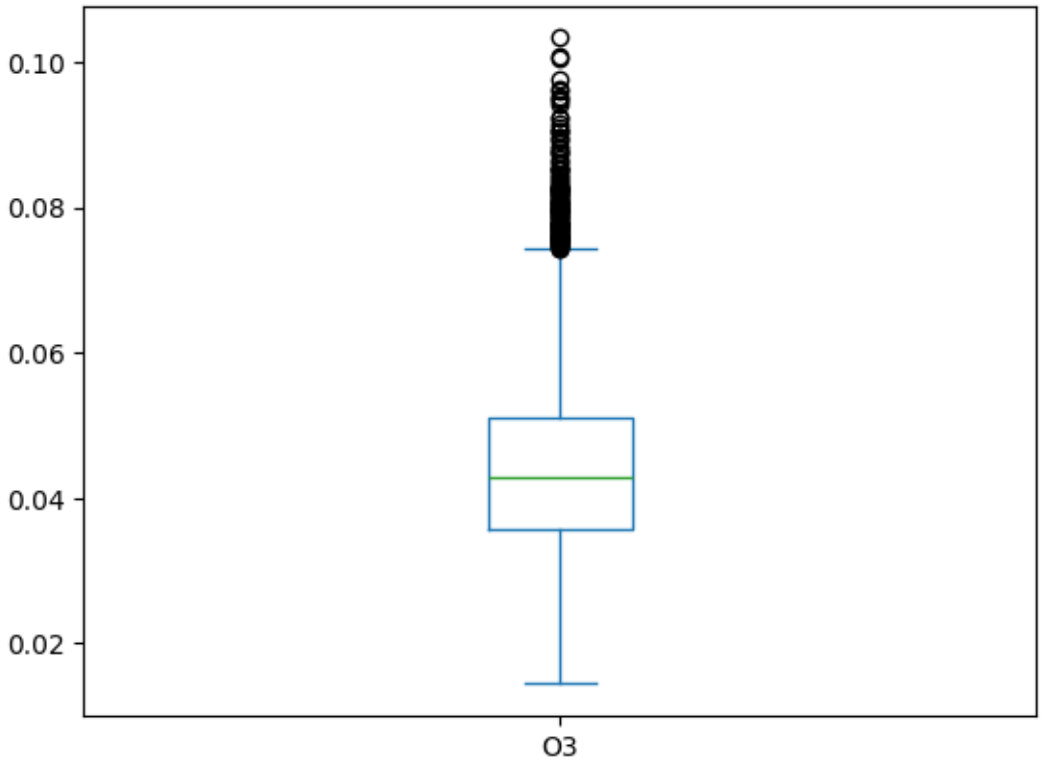

60
50
40
30
20
10
0
2016
2017
2018
2019
2020
2021
2022
2023
2024
PM25


0.010
0.008
0.006
0.004
0.002
2016
2017
2018
2019
2020
2021
2022
2023
2024
SO2

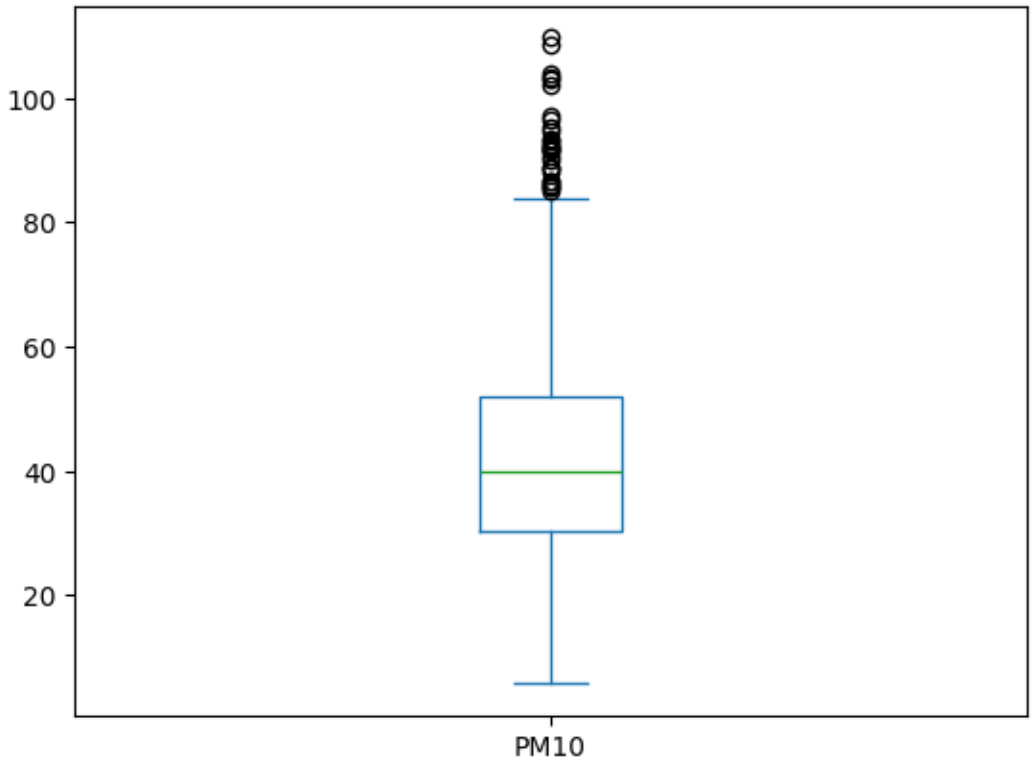

3.0
2.5
2.0
1.5
1.0
0.5
2016
2017
2018
2019
2020
2021
2022
2023
2024
CO


0.06
0.05
0.04
0.03
0.02
0.01
2016
2017
2018
2019
2020
2021
2022
2023
2024
NO2

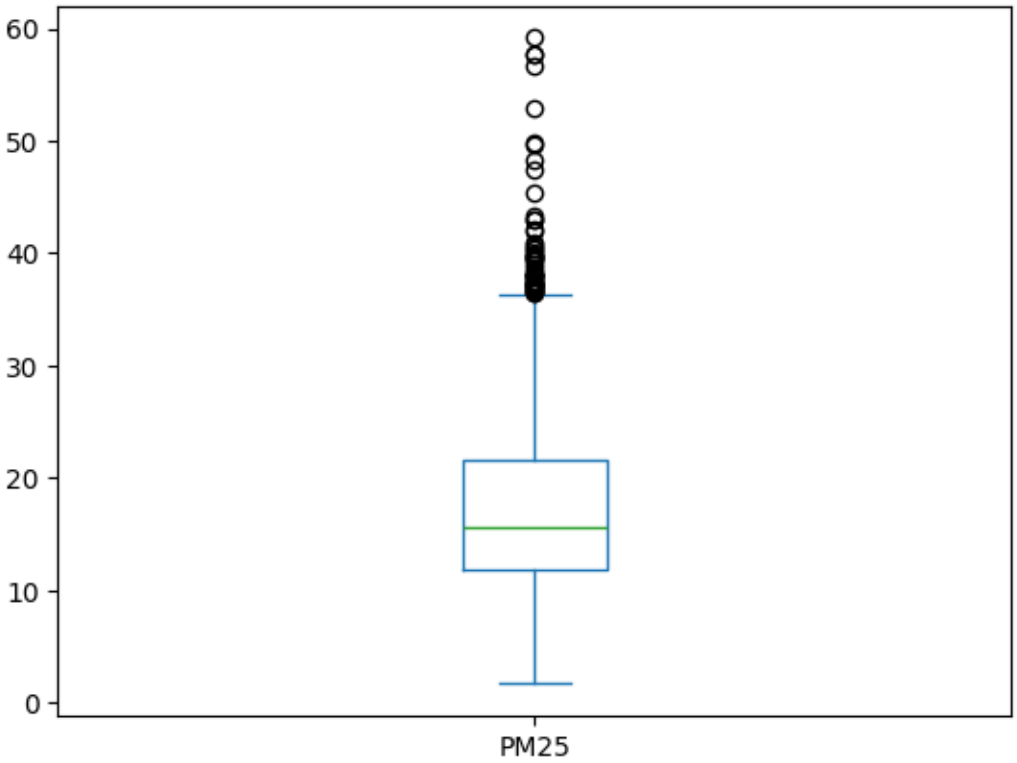
[161]:
```
for column in contam.columns:
    x = contam[column]

    x.plot.box()
    pyplot.show()

    Q1 = x.quantile(0.25)
    Q3 = x.quantile(0.75)
    IQR = Q3 - Q1
    sup = Q3 + 1.5 * IQR
    inf = Q1 - 1.5 * IQR

    print(f"Q1 = {Q1} | Q3 = {Q3} | IQR = {IQR}")

    print(f"Puntos atípicos superiores: {x[x > sup].count()} / {x.count()}")
    print(f"Puntos atípicos inferiores: {x[x < inf].count()} / {x.count()}")
```



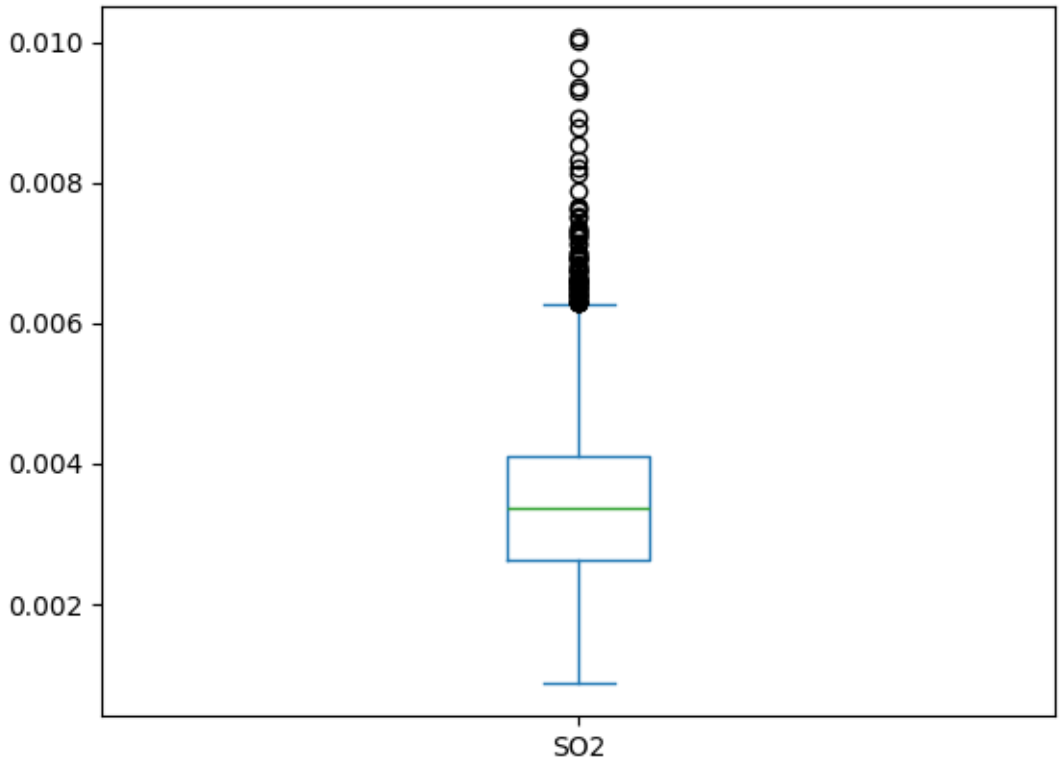

```
Q1 = 0.035606291228617096 | Q3 = 0.0510636588114753 | IQR = 0.015457367582858206
Puntos atípicos superiores: 136 / 3043
Puntos atípicos inferiores: 0 / 3043
```



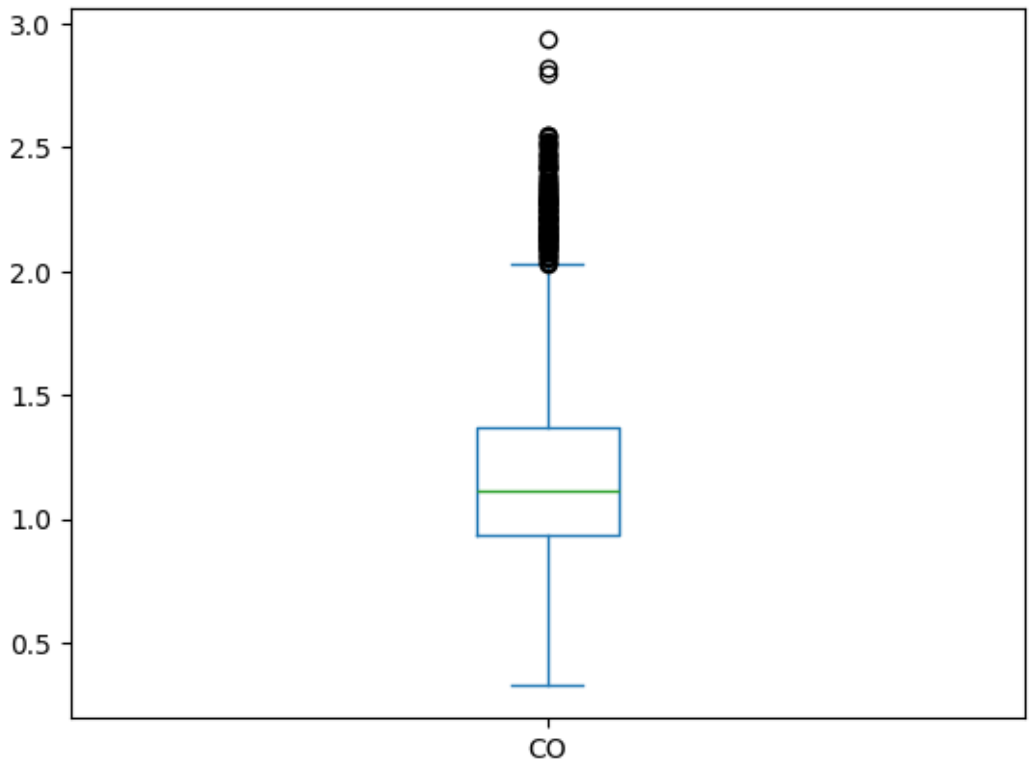

```
Q1 = 30.194444444444443 | Q3 = 51.951388888888886 | IQR = 21.756944444444443
Puntos atípicos superiores: 28 / 3043
Puntos atípicos inferiores: 0 / 3043
```



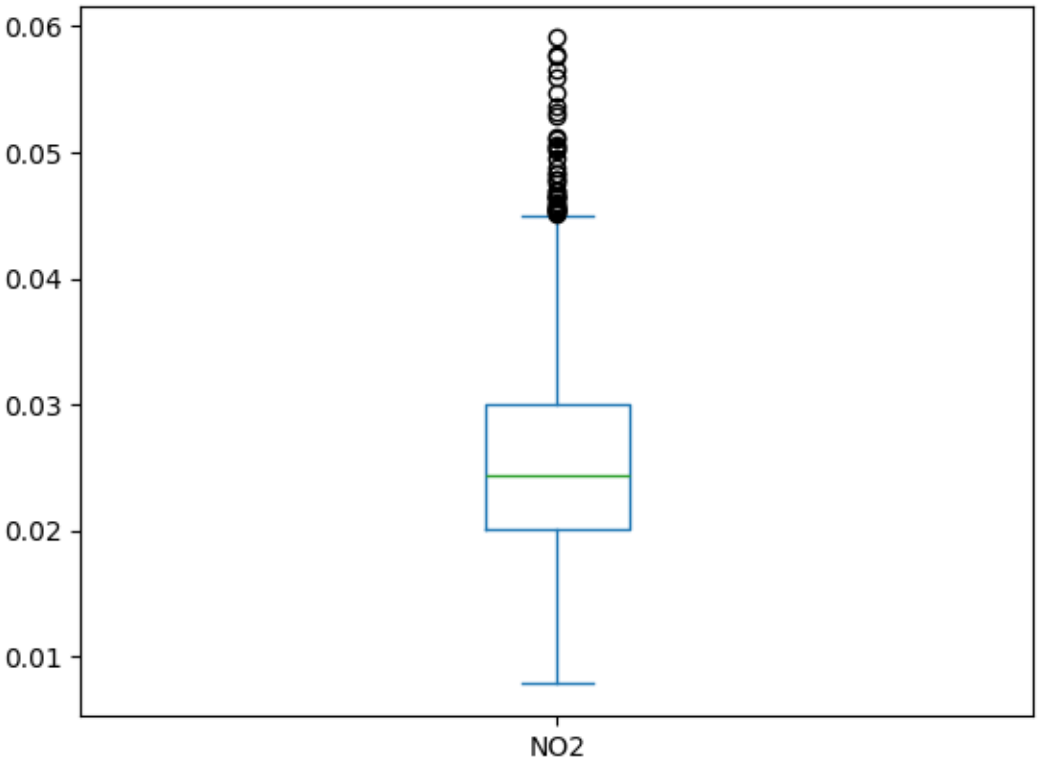

```
Q1 = 11.781840140535792 | Q3 = 21.65489130434783 | IQR = 9.87305116381204
Puntos atípicos superiores: 51 / 3043
Puntos atípicos inferiores: 0 / 3043
```




```
Q1 = 0.0026265850621353002 | Q3 = 0.00408715579763865 | IQR =
0.0014605707355033496
Puntos atípicos superiores: 82 / 3043
Puntos atípicos inferiores: 0 / 3043
```




```
Q1 = 0.931934977471 | Q3 = 1.3703324225865208 | IQR = 0.4383974451155208
Puntos atípicos superiores: 145 / 3043
Puntos atípicos inferiores: 0 / 3043
```




```
Q1 = 0.020078741888660948 | Q3 = 0.03005199112021835 | IQR =
0.009973249231557402
Puntos atípicos superiores: 41 / 3043
Puntos atípicos inferiores: 0 / 3043
```

Verificamos la normalidad de cada columna usando la prueba de Shpairo-Wilk y si se acepta la hipótesis nula H_0 usaremos la correlación de Pearon y si se rechaza (no es normal) usaremos la correlación de Spearmann.

[173]:
```python
from scipy.stats import shapiro

for column in contam.columns:
    x = contam[column]

    r_value, p_value = shapiro(x)

    if p_value > 0.05:
        print(f"{column} se distribuye normal")
    else:
        print(f"{column} no se distribuye normal")
```

```
O3 no se distribuye normal
PM10 no se distribuye normal
```

```
PM25 no se distribuye normal
SO2 no se distribuye normal
CO no se distribuye normal
NO2 no se distribuye normal
```

Cómo los datos en general no se distribuyen normales, usaremos la correlación de Spermann

Correlación de Spearman

$$\rho = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n \cdot (n^2 - 1)}$$

- d - Diferencia en el *rank* o posición en la pareja de observaciones, es decir, ordenados por el primer eje y ordenados por el segundo eje.
- n - Observaciones

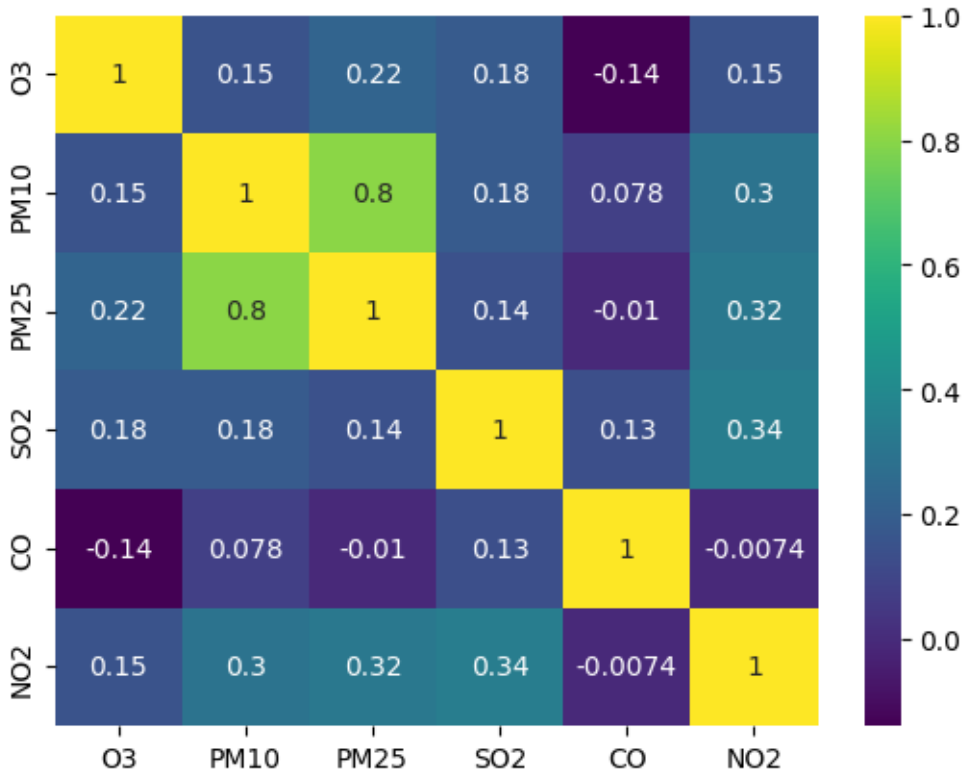
[174]:
```
contam.corr(method="spearman")
```

[174]:

	O3	PM10	PM25	SO2	CO	NO2
O3	1.000000	0.153624	0.222907	0.181364	-0.140142	0.152639
PM10	0.153624	1.000000	0.801181	0.184808	0.077874	0.295978
PM25	0.222907	0.801181	1.000000	0.144048	-0.010205	0.315740
SO2	0.181364	0.184808	0.144048	1.000000	0.133664	0.340807
CO	-0.140142	0.077874	-0.010205	0.133664	1.000000	-0.007370
NO2	0.152639	0.295978	0.315740	0.340807	-0.007370	1.000000

[182]:
```
import seaborn

seaborn.heatmap(contam.corr(method="spearman"), cmap="viridis", annot=True)
```

[182]:
```
<Axes: >
```



[186]:
```
import pingouin

pingouin.pairwise_corr(contam).sort_values(by="r", ascending=False).round(3)
```

[186]:

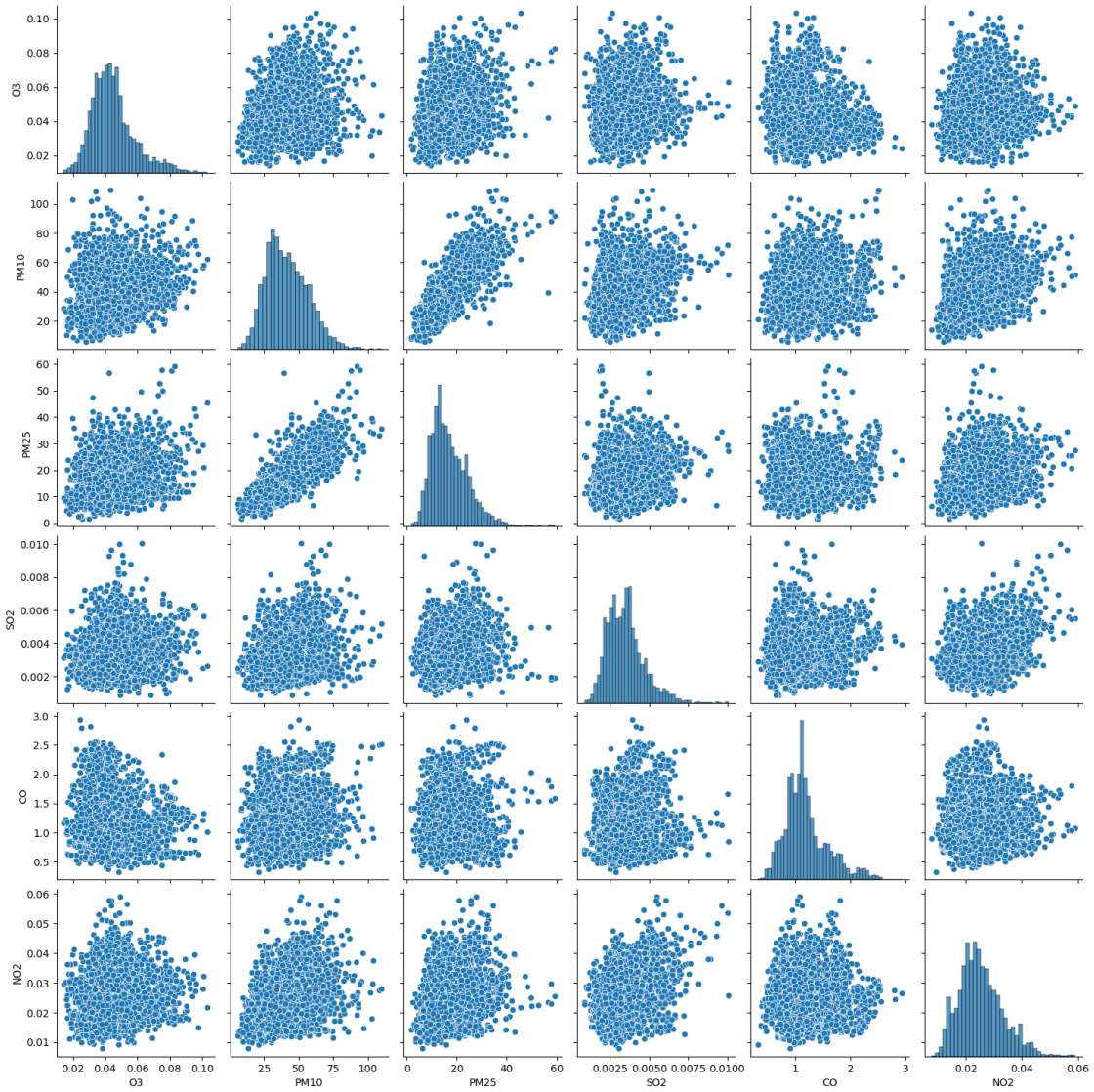
	X	Y	method	alternative	n	r	CI95%	p-unc	BF10	power
5	PM10	PM25	pearson	two-sided	3043	0.791	[0.78, 0.8]	0.000		
13	SO2	NO2	pearson	two-sided	3043	0.399	[0.37, 0.43]	0.000		
8	PM10	NO2	pearson	two-sided	3043	0.310	[0.28, 0.34]	0.000		
11	PM25	NO2	pearson	two-sided	3043	0.301	[0.27, 0.33]	0.000		
1	O3	PM25	pearson	two-sided	3043	0.250	[0.22, 0.28]	0.000		
6	PM10	SO2	pearson	two-sided	3043	0.225	[0.19, 0.26]	0.000		
9	PM25	SO2	pearson	two-sided	3043	0.178	[0.14, 0.21]	0.000		
0	O3	PM10	pearson	two-sided	3043	0.169	[0.13, 0.2]	0.000		
2	O3	SO2	pearson	two-sided	3043	0.163	[0.13, 0.2]	0.000		
12	SO2	CO	pearson	two-sided	3043	0.128	[0.09, 0.16]	0.000		
4	O3	NO2	pearson	two-sided	3043	0.127	[0.09, 0.16]	0.000		
7	PM10	CO	pearson	two-sided	3043	0.120	[0.09, 0.16]	0.000		
10	PM25	CO	pearson	two-sided	3043	0.032	[-0.0, 0.07]	0.077		
14	CO	NO2	pearson	two-sided	3043	-0.016	[-0.05, 0.02]	0.368		
3	O3	CO	pearson	two-sided	3043	-0.155	[-0.19, -0.12]	0.000		

```
5          inf  1.000
13  7.14e+112  1.000
8   6.826e+64  1.000
11   1.47e+61  1.000
1   5.914e+40  1.000
6   5.652e+32  1.000
9   4.521e+19  1.000
0   3.238e+17  1.000
2   1.074e+16  1.000
12  1.751e+09  1.000
4   1.225e+09  1.000
7   9.368e+07  1.000
10      0.108  0.424
14      0.034  0.147
3   2.385e+14  1.000
```

Observamos que *PM10* y *PM25* se correlacionan más positivamente y *CO* y *O3* se correlacionan más negativamente

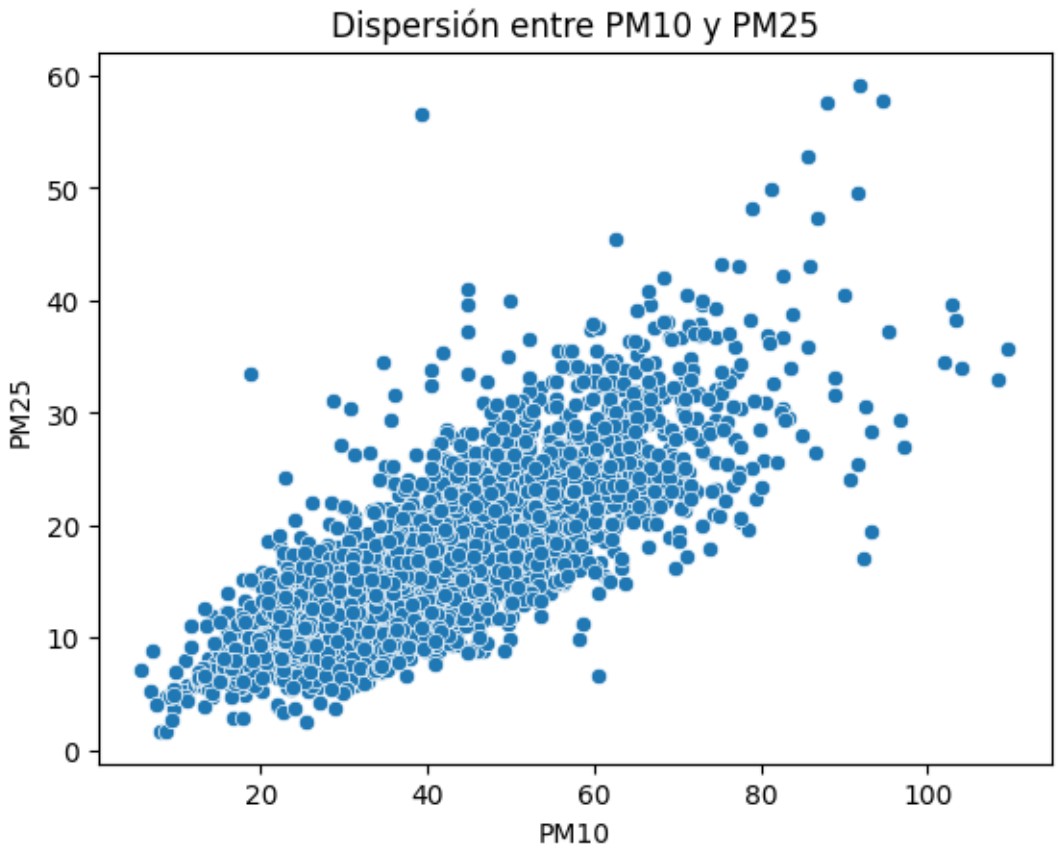
[179]:
```
seaborn.pairplot(contam)
```

[179]:
```
<seaborn.axisgrid.PairGrid at 0x16d8a8620>
```



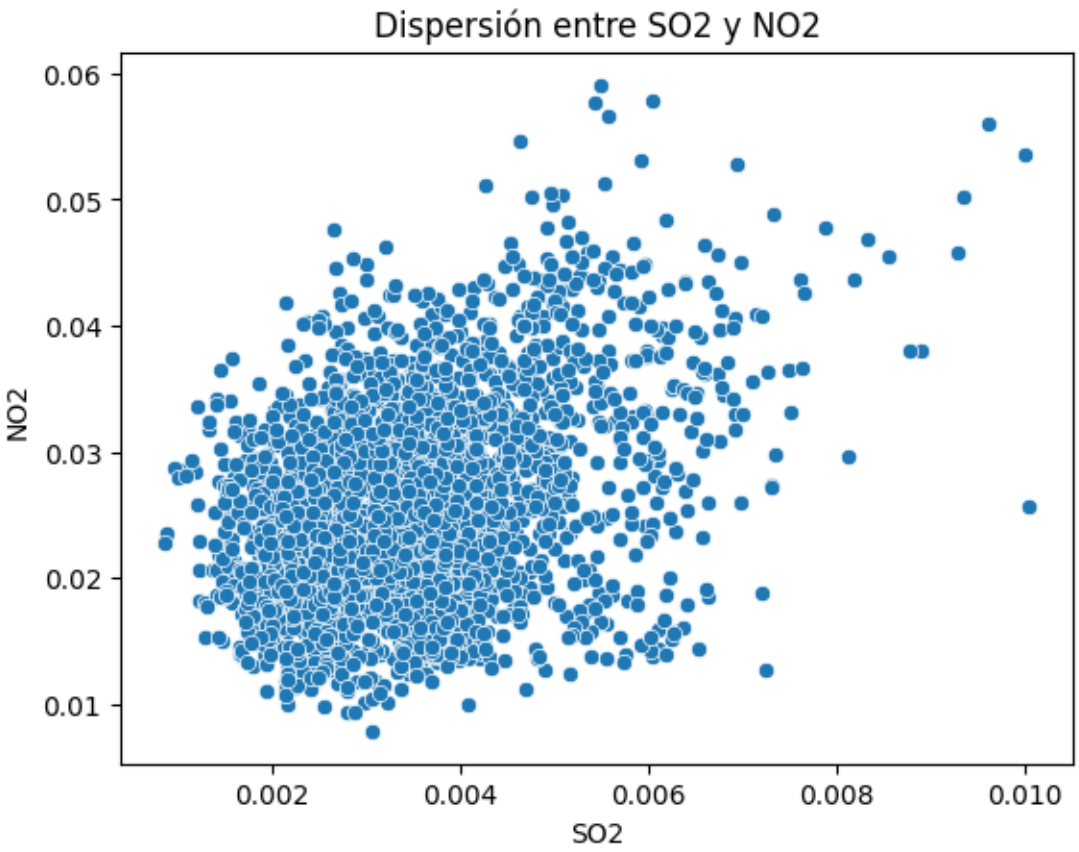
[185]:
```
seaborn.scatterplot(contam, x="PM10", y="PM25")
pyplot.title("Dispersión entre PM10 y PM25")
contam["PM10"].corr(contam["PM25"], method="spearman")
```

[185]: np.float64(0.8011807930223976)



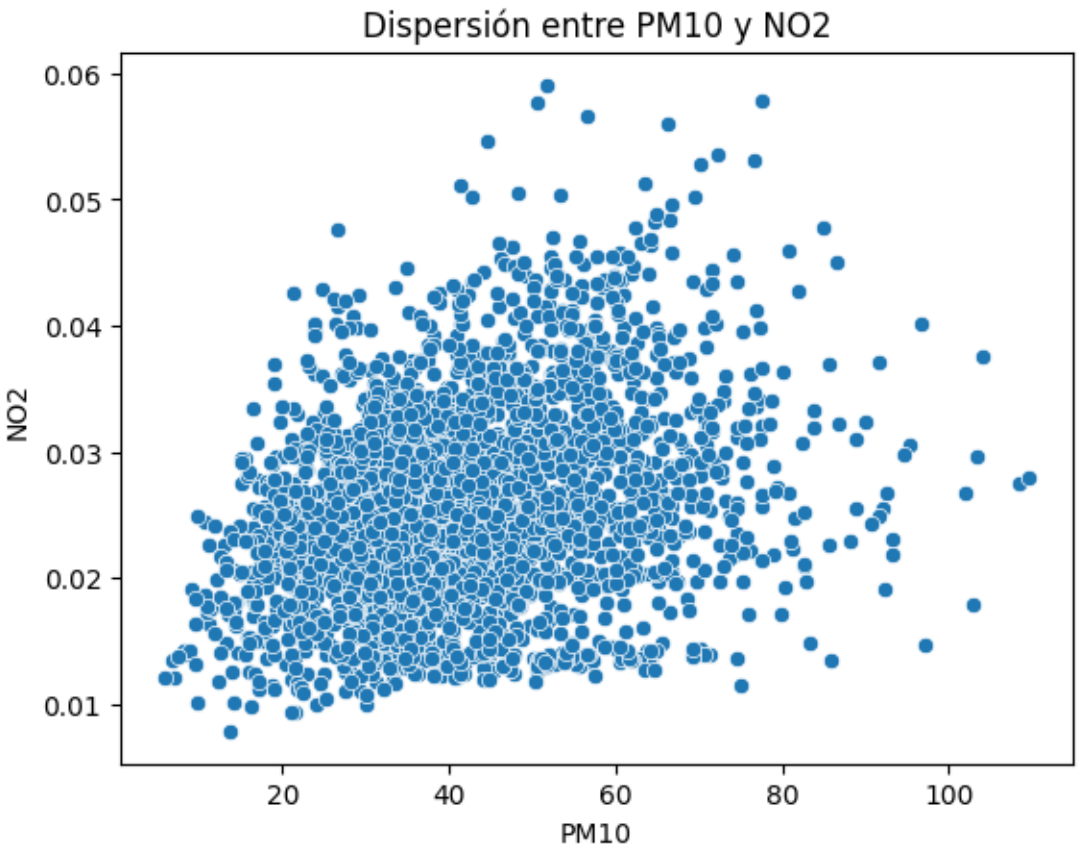
[184]:
```
seaborn.scatterplot(contam, x="SO2", y="NO2")
pyplot.title("Dispersión entre SO2 y NO2")
contam["SO2"].corr(contam["NO2"], method="spearman")
```

[184]: np.float64(0.3408070510798253)



[187]:
```
seaborn.scatterplot(contam, x="PM10", y="NO2")
pyplot.title("Dispersión entre PM10 y NO2")
contam["PM10"].corr(contam["NO2"], method="spearman")
```

[187]: np.float64(0.29597821821676923)



Test de Mann-Kendall

- H_0 - No hay tendencia monótona (la serie es aleatoria)
- H_a - Existe una tendencia nónotona (positiva o negativa)

$$S = \sum_{j=1}^{n-1} \sum_{k=j+1}^{n} sign(p_k - p_j)$$

$$Var(S) = ...$$

$$Z = \begin{cases} \frac{S-1}{Var(S)} & S > 0 \\ 0 & S = 0 \\ \frac{S+1}{Var(S)} & S < 0 \end{cases}$$

[201]:
```
import pymannkendall as kendall

reportes = []
```

```python
for column in contam.columns:

    test = kendall.original_test(contam[column], alpha=0.05)
    reportes.append((column, test))

    reporte = pandas.DataFrame(
        {
            "Contaminante": column,
            "Trend": [test.trend],
            "Ha": [test.h],
            "p-value": [test.p],
            "Z": [test.z],
            "S": [test.s],
            "Var(S)": [test.var_s],
            "b0": [test.intercept],
            "b1": [test.slope],
            "tau": [test.Tau],
        },
        index=["Value"],
    ).T

    print(reporte, end="\n\n")
```

```
                          Value
Contaminante                 O3
Trend                 increasing
Ha                          True
p-value                      0.0
Z                       17.591508
S                       984559.0
Var(S)          3132394657.666667
b0                       0.036618
b1                       0.000004
tau                      0.212721

                          Value
Contaminante               PM10
Trend                 decreasing
Ha                          True
p-value                      0.0
Z                       -7.791406
S                       -436069.0
Var(S)          3132396636.333333
b0                      43.740424
b1                      -0.002505
tau                     -0.094216

                          Value
```

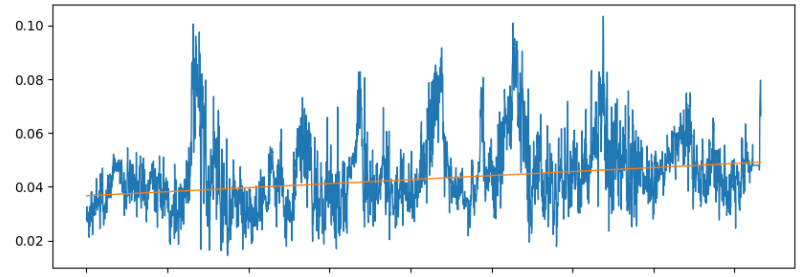
Contaminante	PM25
Trend	decreasing
Ha	True
p-value	0.011176
Z	-2.537141
S	-141999.0
Var(S)	3132386463.666667
b0	16.254297
b1	-0.000369
tau	-0.03068

	Value
Contaminante	SO2
Trend	increasing
Ha	True
p-value	0.0
Z	15.533966
S	869403.0
Var(S)	3132395482.333333
b0	0.002823
b1	0.0
tau	0.187841

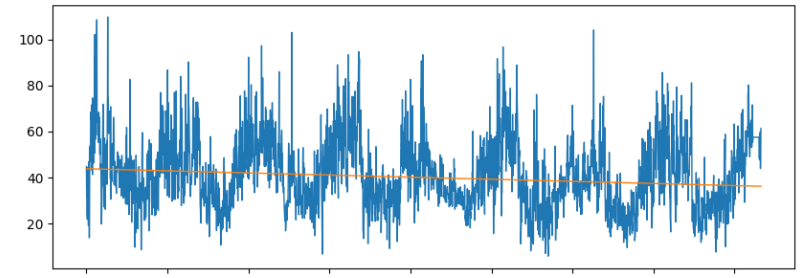
	Value
Contaminante	CO
Trend	decreasing
Ha	True
p-value	0.0
Z	-18.73453
S	-1048487.0
Var(S)	3132129109.666667
b0	1.312365
b1	-0.00013
tau	-0.226533

	Value
Contaminante	NO2
Trend	increasing
Ha	True
p-value	0.0
Z	17.235795
S	964650.0
Var(S)	3132391177.333333
b0	0.020121
b1	0.000003
tau	0.20842

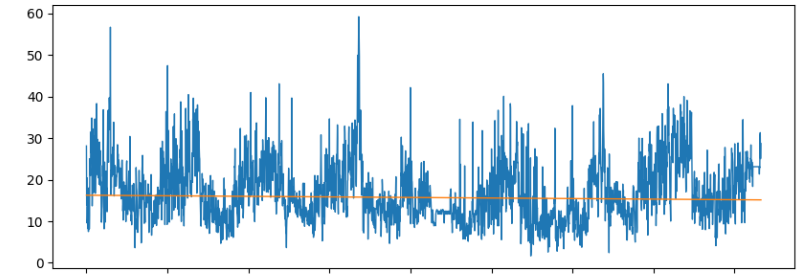
[210]:
```
figure, axis = pyplot.subplots(len(reportes), 1, figsize=(10, 25))

for i, (column, reporte) in enumerate(reportes):
    points = [reporte.intercept + i * reporte.slope for i in␣
 ↪range(contam[column].count())]
    y = pandas.Series(points, index=contam.index)
    axis[i].plot(contam.index, contam[column], lw=1)
    axis[i].plot(contam.index, points, lw=1)
    axis[i].set_xlabel(f"{column} {reporte.trend}")
```



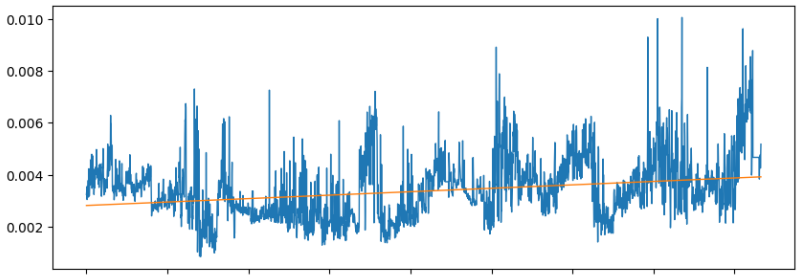
O3 increasing



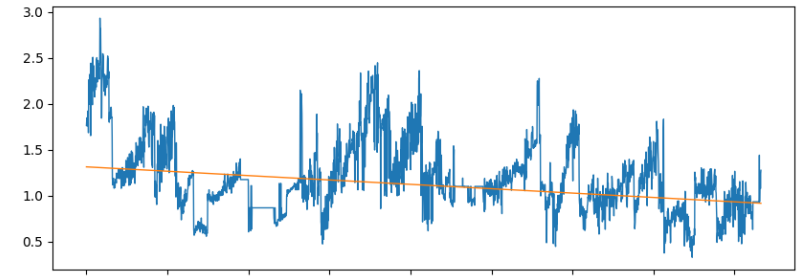
PM10 decreasing



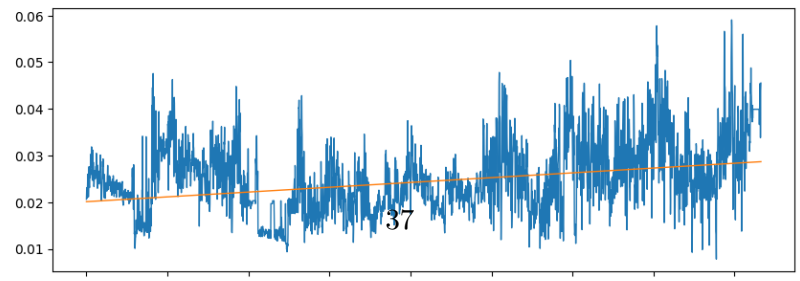
PM25 decreasing



SO2 increasing



CO decreasing



NO2 increasing

Innovative Trand Analysis (ITA Methodology)

- **Pasado:** $x_1 = x_0, \dots, x_{n/2-1}$
- **Presente:** $x_2 = x_{n/2}, \dots, x_{n-1}$

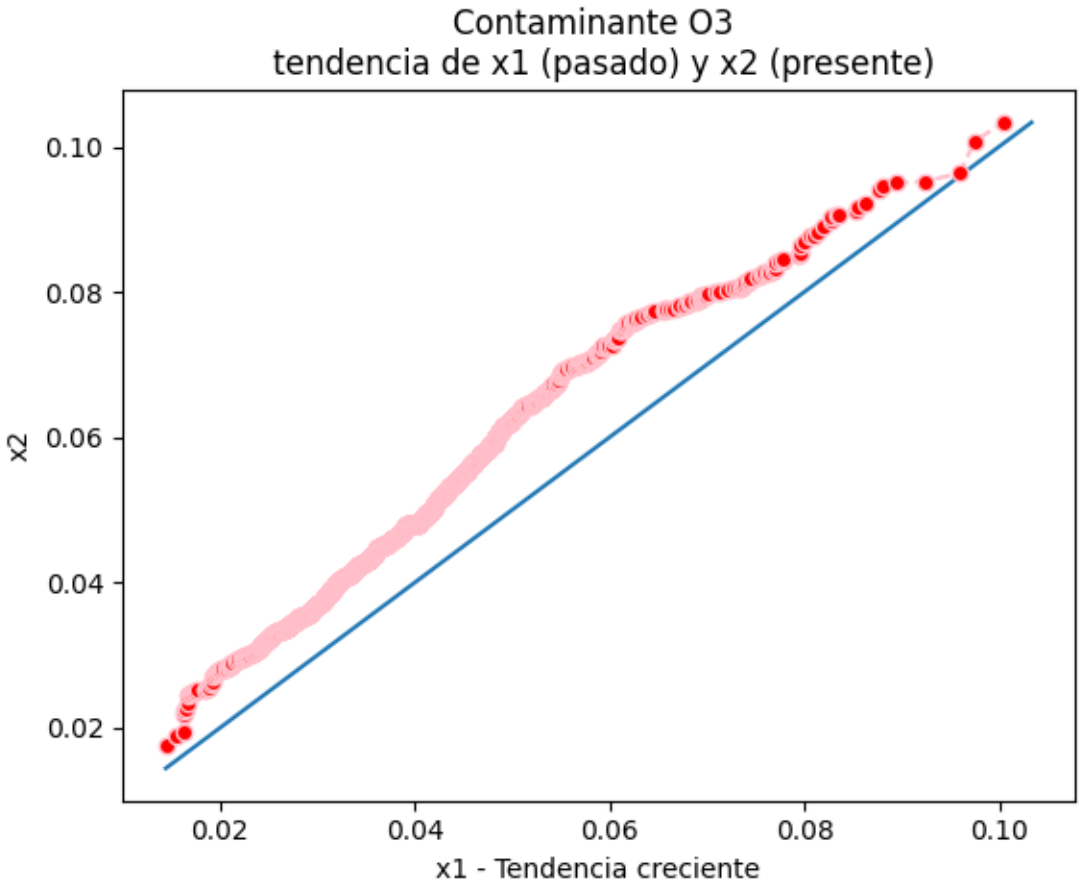
Interpretación gráfica

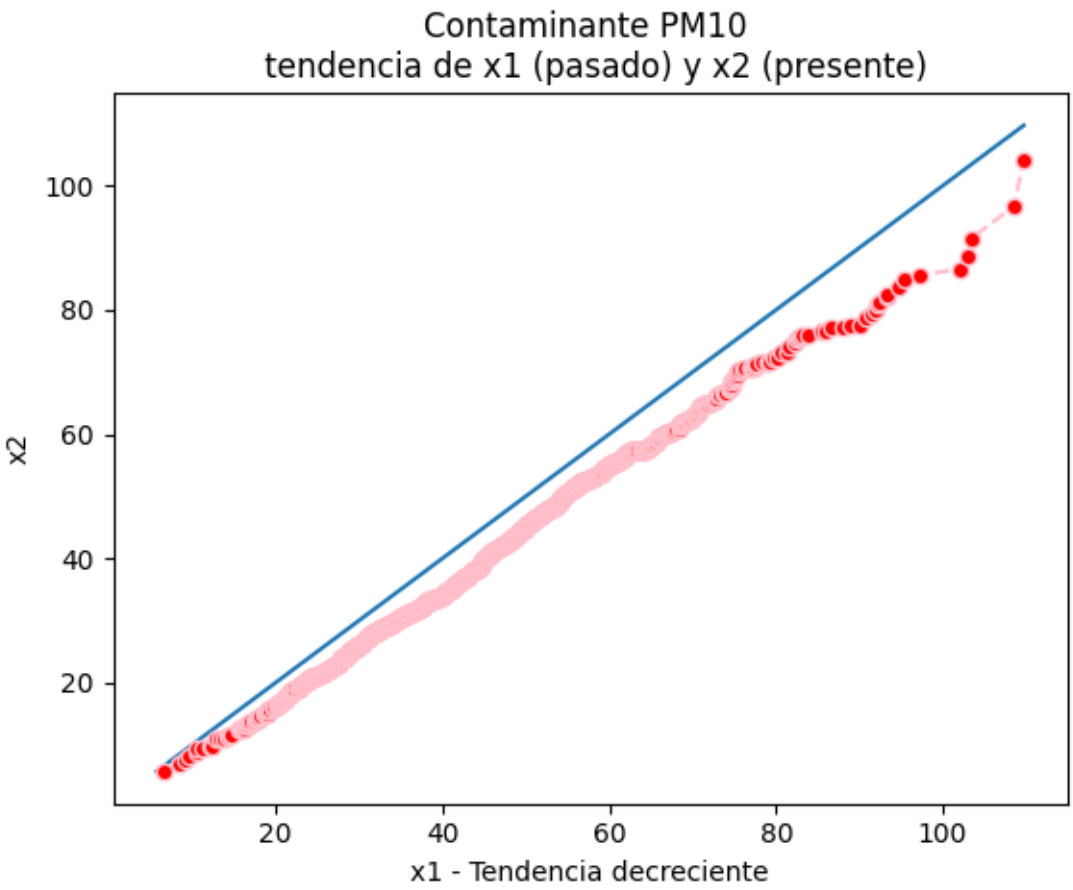
- **Puntos por encima de la diagonal**: señal de tendencia creciente.
- **Puntos por debajo**: tendencia decreciente.
- **Puntos dispersos simétricamente**: sin tendencia global.
- **Patrones específicos (como dispersión en extremos)**: pueden indicar tendencias parciales.

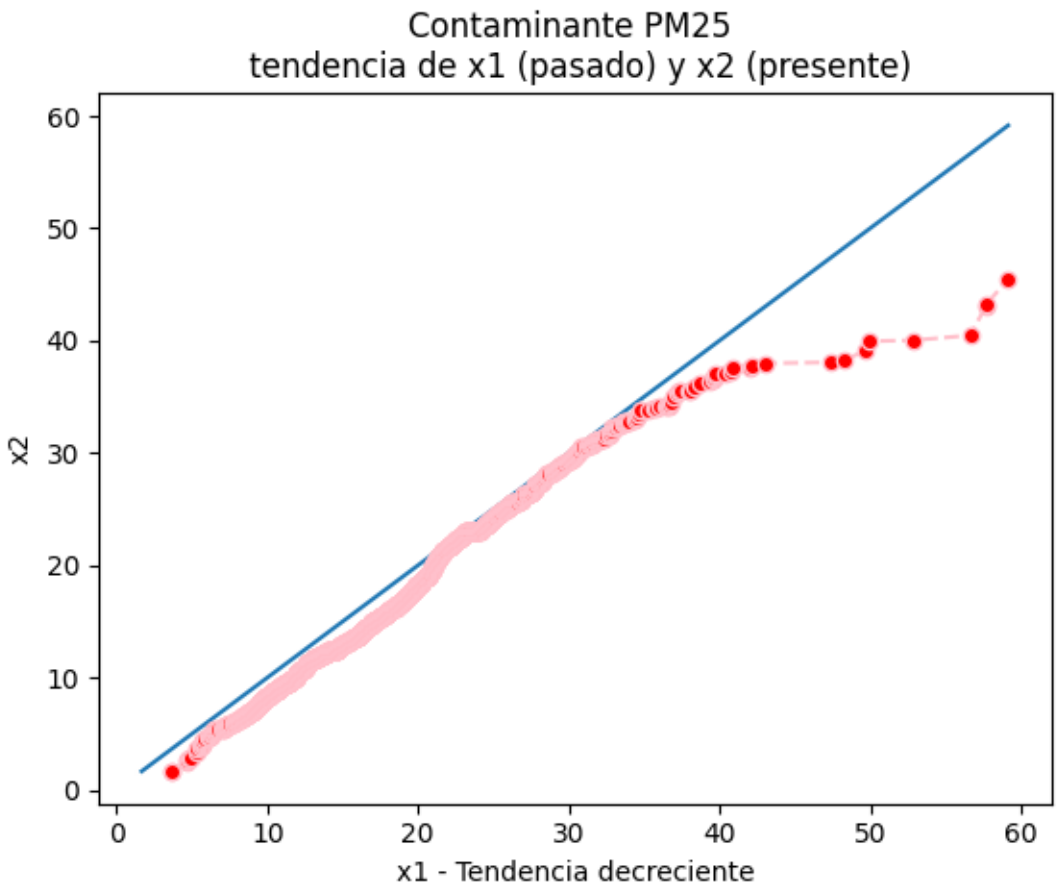
Sen Z. 2012. Innovative trend analysis methodology, J Hydrol Eng, 17 (9), pp. 1042–1046. https://doi.org/10.1061/(ASCE)HE.1943-5584.0000556

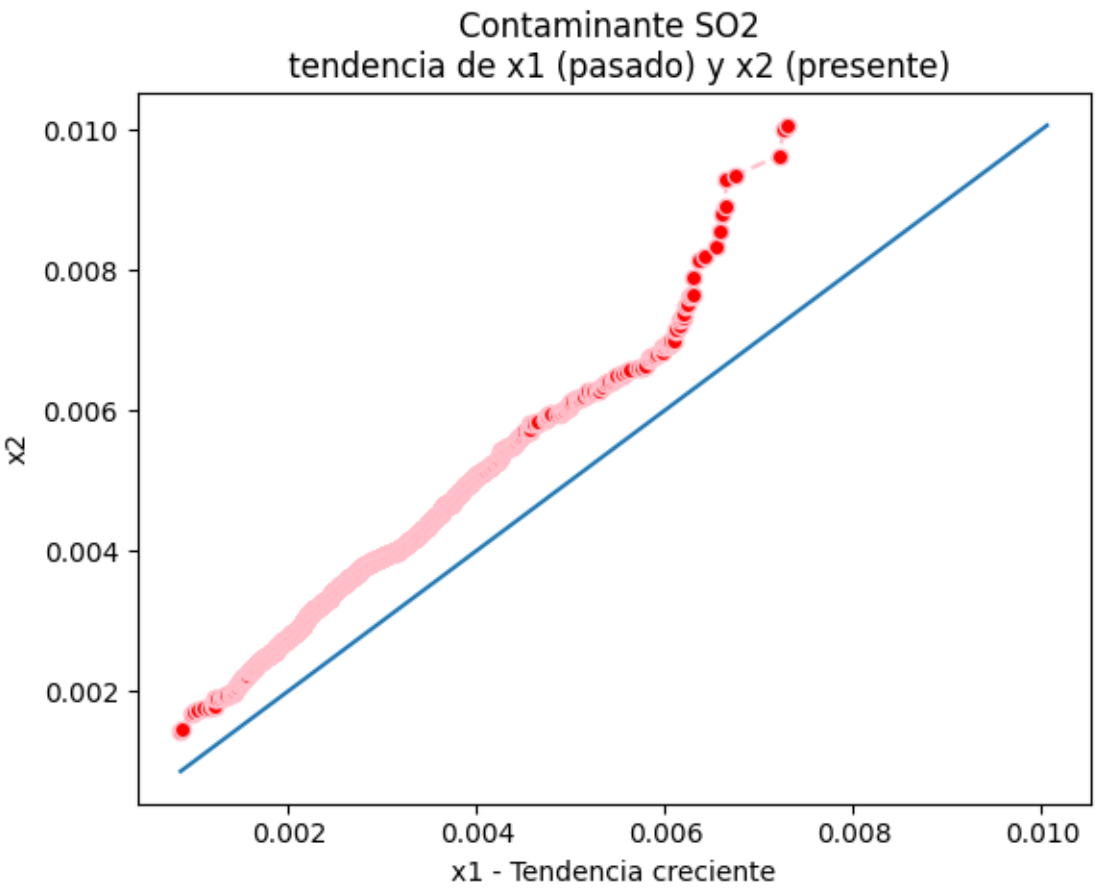
[222]:
```
import numpy
import matplotlib.pyplot as pyplot

for column in contam.columns:

    x = contam[column]

    n = x.count()

    m = int(n / 2)

    x1 = x[:m]
    x2 = x[m:2*m]

    xp = numpy.linspace(min(x1.min(), x2.min()), max(x1.max(), x2.max()))

    b = numpy.array(sorted(x2))
    a = numpy.array(sorted(x1))

    tendenciaArriba = (b / a).mean() > 1
    tendenciaAbajo = (b / a).mean() < 1

    tendencia = "Sin tendencia"

    if tendenciaArriba:
        tendencia = "Tendencia creciente"
    elif tendenciaAbajo:
        tendencia = "Tendencia decreciente"

    pyplot.plot(xp, xp)
    pyplot.plot(sorted(x1), sorted(x2),
```

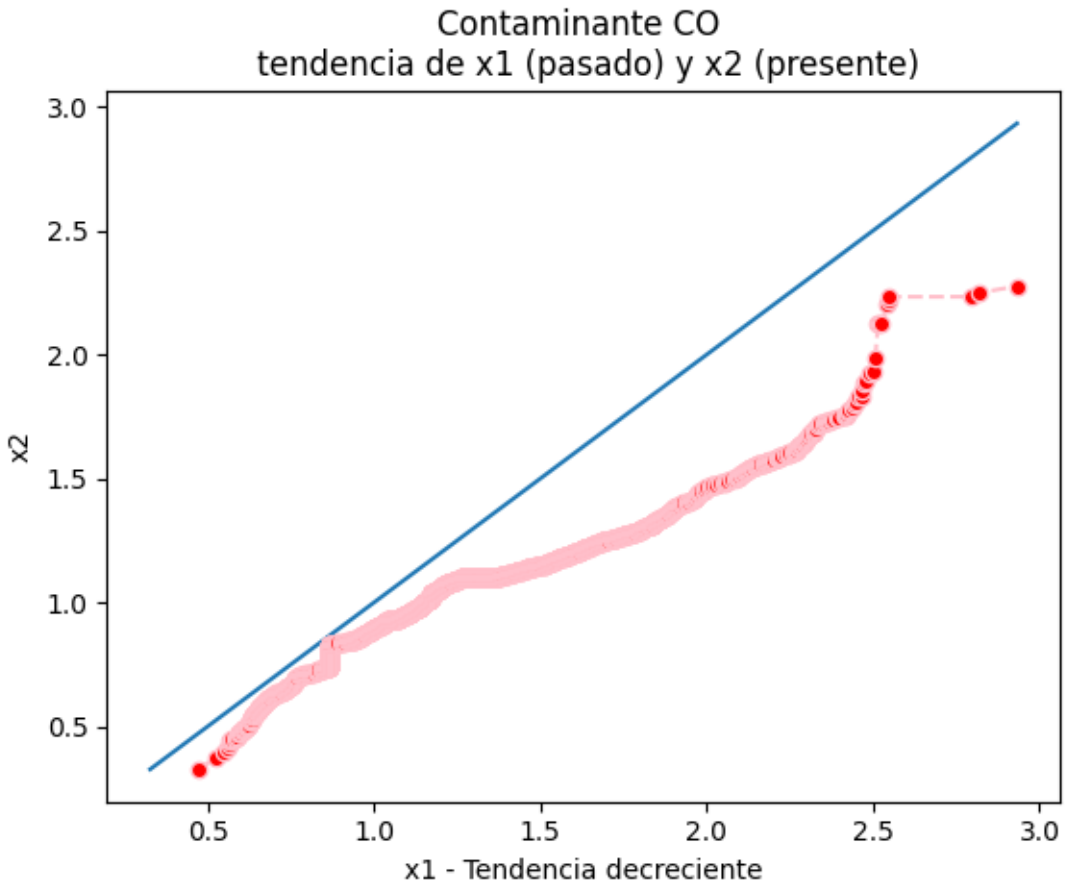
```
                marker="o", linestyle="--",
                color="pink", markerfacecolor="red")
    pyplot.xlabel(f"x1 - {tendencia}")
    pyplot.ylabel("x2")
    pyplot.title(f"Contaminante {column} \n tendencia de x1 (pasado) y x2␣
↪(presente)")
    pyplot.show()
```

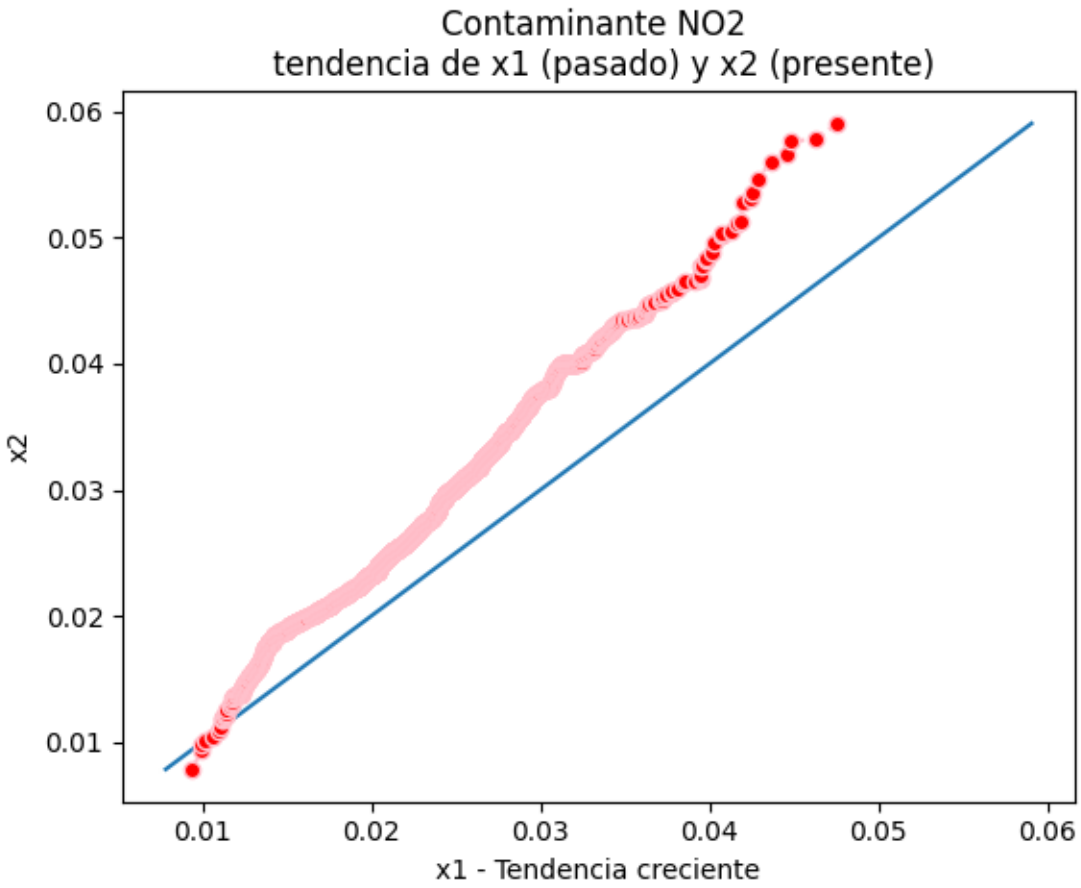



Contaminante PM10
tendencia de x1 (pasado) y x2 (presente)
x2
100
80
60
40
20
20
40
60
80
100
x1 - Tendencia decreciente


Contaminante PM25
tendencia de x1 (pasado) y x2 (presente)
x2
x1 - Tendencia decreciente
0
10
20
30
40
50
60


Contaminante SO2
tendencia de x1 (pasado) y x2 (presente)
0.010
0.008
0.006
0.004
0.002
x2
0.002
0.004
0.006
0.008
0.010
x1 - Tendencia creciente


Contaminante CO
tendencia de x1 (pasado) y x2 (presente)
3.0
2.5
2.0
1.5
1.0
0.5
x2
0.5
1.0
1.5
2.0
2.5
3.0
x1 - Tendencia decreciente


Contaminante NO2
tendencia de x1 (pasado) y x2 (presente)
0.06
0.05
0.04
0.03
0.02
0.01
x2
0.01
0.02
0.03
0.04
0.05
0.06
x1 - Tendencia creciente