No description has been provided for this image

Mario Cañadas Castro

Dpt. Patents and Technological Information - OEPM - ES

Email: mario.canadas@oepm.es

EPO - Patent Knowledge Forum 2024

Use case at OEPM: PATSTAT - Climate Change Mitigation

We present a simplified version of a previous study on **climate change mitigation technologies (https://www.oepm.es/export/sites/oepm/comun/documentos_relacionados/Memorias_de_Ac 2020.pdf)** (CCMT) using "PATSTAT online", which now has been replicated with the Technology Intelligence Platform (**TIP**). It covers:

- **Patent Trends**: evolution over the past ten years of EP applications.
- **Gender Gap Analysis**: A gender gap study is included as a new aspect, analyzing inventor names with WIPO gender dictionary (https://tind.wipo.int/record/49408?v=tab).
- **Regional Mapping**: A map of Spanish provinces with the highest patent activity is presented, enabled by the TIP's capabilities.

EP patents with origin in Spain about Climate Change Mitigation Technologies (CPC Y02)

Collecting patent applications:

We define ES origin of an invention (patent applications) if the first applicant has the residence in Spain.

First we import **PATSTAT** and Pandas libraries

In [1]:
```
from epo.tipdata.patstat import PatstatClient
import pandas as pd
patstat = PatstatClient(env='PROD')
```

In PATSTAT we import the needed Tables

In [2]:
```
from epo.tipdata.patstat.database.models import TLS201_APP
LN, TLS224_APPLN_CPC, TLS207_PERS_APPLN, TLS206_PERSON
```

Timeframe: published applications **between 2014 and 2023** (first publications: "A" kind code)

Output data is stored as two **dataframes (Pandas)**:

- **df_epo**: all EP applications
- **df_epo_clim**: EP applications related to climate change mitigation (Y02)

In [3]:
```
# Number of EPO applications from ES origin
q = patstat.sql_query("""
    SELECT A.earliest_publn_year AS publ_year, COUNT(DISTI
NCT A.appln_id) AS numb_appl
    FROM tls201_appln A
    JOIN tls224_appln_cpc C ON A.appln_id = C.appln_id
    JOIN tls207_pers_appln Q ON A.appln_id = Q.appln_id
    JOIN tls206_person P ON Q.person_id = P.person_id
    WHERE A.earliest_publn_year BETWEEN 2014 AND 2023
    AND A.appln_auth = 'EP'
    AND P.person_ctry_code = 'ES'
    AND A.appln_kind = 'A '
    AND Q.applt_seq_nr = 1
    --AND C.cpc_class_symbol LIKE 'Y02%'
    GROUP BY A.earliest_publn_year
    ORDER BY A.earliest_publn_year

""", use_legacy_sql = False)
```

In [4]:
```
df_epo = pd.DataFrame(q)
#df_epo
```

In [5]:
```
# Number of EPO applications from ES origin where CPC = Y0
2*
q2 = patstat.sql_query("""
    SELECT A.earliest_publn_year AS publ_year, COUNT(DISTI
NCT A.appln_id) AS numb_appl
    FROM tls201_appln A
    JOIN tls224_appln_cpc C ON A.appln_id = C.appln_id
    JOIN tls207_pers_appln Q ON A.appln_id = Q.appln_id
    JOIN tls206_person P ON Q.person_id = P.person_id
    WHERE A.earliest_publn_year BETWEEN 2014 AND 2023
    AND A.appln_auth = 'EP'
    AND P.person_ctry_code = 'ES'
    AND A.appln_kind = 'A '
    AND Q.applt_seq_nr = 1
    AND C.cpc_class_symbol LIKE 'Y02%'
    GROUP BY A.earliest_publn_year
    ORDER BY A.earliest_publn_year

""", use_legacy_sql = False)
```

In [6]:
```
df_epo_clim = pd.DataFrame(q2)
df_epo_clim
```

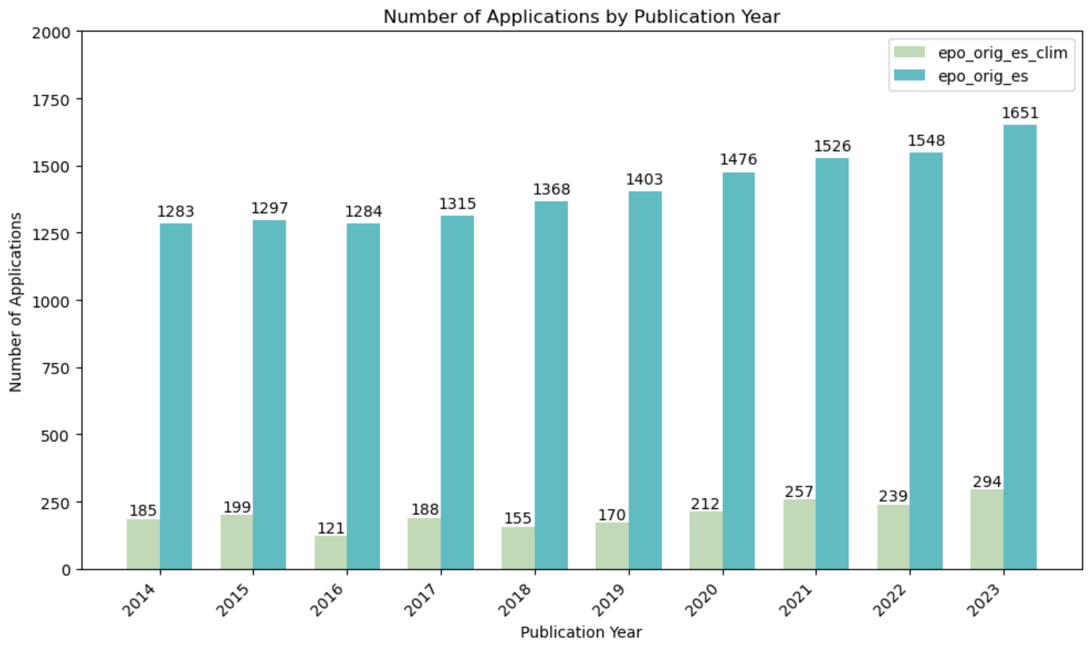
Out[6]:

	publ_year	numb_appl
0	2014	185
1	2015	199
2	2016	121
3	2017	188
4	2018	155
5	2019	170
6	2020	212
7	2021	257
8	2022	239
9	2023	294

Plot showing the evolution in the last 10 years

In [7]:
```
import matplotlib.pyplot as plt
import numpy as np

# Crear la figura y el eje
fig, ax = plt.subplots(figsize=(10, 6))  # Tamaño del gráf
ico
```

```
# Configuración de los valores de las barras
width = 0.35  # Ancho de las barras

# Crear las barras para el primer DataFrame (pat_orig_es_y
02)
bars1 = ax.bar(np.array(df_epo_clim['publ_year']) - width/
2, df_epo_clim['numb_appl'],
               width=width, color='#C1D9B7', label='epo_or
ig_es_clim')

# Crear las barras para el segundo DataFrame (pat_orig_es)
bars2 = ax.bar(np.array(df_epo['publ_year']) + width/2, df
_epo['numb_appl'],
               width=width, color='#60BCC1', label='epo_or
ig_es')

# Etiquetas y título en inglés
ax.set_ylim(0, 2000)  # Set y-axis range
ax.set_xlabel('Publication Year')
ax.set_ylabel('Number of Applications')
ax.set_title('Number of Applications by Publication Year')

# Añadir los valores sobre las barras
for bar in bars1:
    yval = bar.get_height()
    xval = bar.get_x() + bar.get_width() / 2
    ax.text(xval, yval + 0.01 * max(df_epo_clim['numb_app
l']), f'{yval:.0f}', ha='center', va='bottom')

for bar in bars2:
    yval = bar.get_height()
    xval = bar.get_x() + bar.get_width() / 2
    ax.text(xval, yval + 0.01 * max(df_epo['numb_appl']),
f'{yval:.0f}', ha='center', va='bottom')

# Configurar las etiquetas del eje X para que estén centra
das debajo de cada barra
ax.set_xticks(df_epo_clim['publ_year'])  # Asegura que los
años estén en el eje X
ax.set_xticklabels(df_epo_clim['publ_year'], rotation=45,
ha='right')  # Rota las etiquetas si es necesario

# Añadir leyenda
ax.legend()

# Ajustar el espacio entre los gráficos
plt.tight_layout()

# Mostrar el gráfico
plt.show()
```



Gender Gap Study

Adding a gender gap study to the previous case:

Now we want to study the **number of women inventors** in those patent applications (ES origin and also related to climate change mitigation -CPC Y02-). First we will collect **inventor names in PATSTAT**, then we will cross that data with an ad-hoc dictionary to assess the gender of each one according to his/her name.

Collecting names of inventors (new PATSTAT query)

Adding inventor names and countries: `P.person_ctry_code AS V_PAIS, P.person_name AS V_NOMBRE_COMPLETO`

In [8]:
```
# Name of inventors of EPO applications from ES origin
qgp = patstat.sql_query("""
SELECT DISTINCT A.earliest_publn_year AS YEAR, P.person_ct
ry_code AS V_PAIS, P.person_name AS V_NOMBRE_COMPLETO
    FROM tls201_appln A
    JOIN tls207_pers_appln Q ON A.appln_id = Q.appln_id
    JOIN tls206_person P ON Q.person_id = P.person_id
    WHERE Q.invt_seq_nr >= 1
    AND A.appln_id In ( SELECT DISTINCT A.appln_id
        FROM tls201_appln A
        JOIN tls224_appln_cpc C ON A.appln_id = C.appln_id
        JOIN tls207_pers_appln Q ON A.appln_id = Q.appln_i
d
        JOIN tls206_person P ON Q.person_id = P.person_id
        WHERE A.earliest_publn_year BETWEEN 2014 AND 2023
        AND A.appln_auth = 'EP'
        AND P.person_ctry_code = 'ES'
        AND A.appln_kind = 'A '
        AND Q.applt_seq_nr = 1
        AND C.cpc_class_symbol LIKE 'Y02%')
    ORDER BY  A.earliest_publn_year, P.person_name

""", use_legacy_sql = False)
```

In [9]:
```
df_names_epo_clim = pd.DataFrame()
# Convert to data frame
df_names_epo_clim = pd.DataFrame(qgp)
df_names_epo_clim
```

Out[9]:

	YEAR	V_PAIS	V_NOMBRE_COMPLETO
0	2014	ES	ABANADES GARCÍA, Juan Carlos
1	2014	ES	ABELLÁN SÁEZ, Gonzalo
2	2014	ES	AINZ IBARRONDO, Félix
3	2014	ES	ALLONA ALBERICH, Isabel
4	2014	ES	ALONSO BEDATE, Carlos
...	...	...	...
5464	2023	DE	Zeller, Lenz Simon
5465	2023	ES	ÁLVAREZ CARREÑO, Carlos
5466	2023	ES	ÁLVAREZ DE DIEGO, Javier
5467	2023	ES	ÁLVAREZ-HERMS, Jesús
5468	2023	ES	ÁVILA GARCÍA, Pedro

5469 rows × 3 columns

Importing WIPO "Gender name dictionary"

We upload into TIP a "csv" file with the **WIPO dictionary**, including contributions and refinement by OEPM.

In [10]:
```
# Load the gender dictionary if not already loaded
try:
    df_gender_dic
    print("El Diccionario de género ya estaba cargado.")
except NameError:
    print("Leyendo diccionario...")
    # Load the dictionary with the correct encoding
    df_gender_dic = pd.read_csv("~/generoPAT/DIM_EST_GENER
O_202410.csv", delimiter='}', encoding='latin1')
    print("Diccionario de género cargado.")
```

```
Leyendo diccionario...
Diccionario de género cargado.
```

Data wrangling to work with inventor names

In [11]:
```
# Separate V_NOMBRE_COMPLETO in 2 fields by comma
df_names_epo_clim[['V_APELLIDOS', 'V_NOMBRE']] = df_names_
epo_clim['V_NOMBRE_COMPLETO'].str.split(',', expand=True,
n=1)
#df_names_epo_clim

# Capitalize V_NOMBRE and remove spaces in front and behin
d
df_names_epo_clim['V_NOMBRE'] = df_names_epo_clim['V_NOMBR
E'].str.upper()
df_names_epo_clim['V_NOMBRE'] = df_names_epo_clim['V_NOMBR
E'].str.strip()
df_names_epo_clim
```

Out[11]:

	YEAR	V_PAIS	V_NOMBRE_COMPLETO	V_APELLIDOS	V_NOMBRE
0	2014	ES	ABANADES GARCÍA, Juan Carlos	ABANADES GARCÍA	JUAN CARLOS
1	2014	ES	ABELLÁN SÁEZ, Gonzalo	ABELLÁN SÁEZ	GONZALO
2	2014	ES	AINZ IBARRONDO, Félix	AINZ IBARRONDO	FÉLIX
3	2014	ES	ALLONA ALBERICH, Isabel	ALLONA ALBERICH	ISABEL
4	2014	ES	ALONSO BEDATE, Carlos	ALONSO BEDATE	CARLOS
...	...	...	...	...	...
5464	2023	DE	Zeller, Lenz Simon	Zeller	LENZ SIMON
5465	2023	ES	ÁLVAREZ CARREÑO, Carlos	ÁLVAREZ CARREÑO	CARLOS
5466	2023	ES	ÁLVAREZ DE DIEGO, Javier	ÁLVAREZ DE DIEGO	JAVIER
5467	2023	ES	ÁLVAREZ-HERMS, Jesús	ÁLVAREZ-HERMS	JESÚS
5468	2023	ES	ÁVILA GARCÍA, Pedro	ÁVILA GARCÍA	PEDRO

5469 rows × 5 columns

Determining the inventor's gender

Using "**merge**" function between dataframes (Pandas library)

In [12]:
```
df_names_epo_clim = pd.merge(df_names_epo_clim, df_gender_
dic, on=["V_NOMBRE", "V_PAIS"], how="left")
#df_names_epo_clim
```

Cleaning data

In [13]:
```
# Replace M by Male, F by Female and NaN by Unknown
# df_names_epo_clim=df
df_names_epo_clim['V_GENERO'] = df_names_epo_clim['V_GENER
O'].fillna('Unknown')
df_names_epo_clim['V_GENERO'] = df_names_epo_clim['V_GENER
O'].replace({'F': 'Female', 'M': 'Male'})
df_names_epo_clim
```

Out[13]:

	YEAR	V_PAIS	V_NOMBRE_COMPLETO	V_APELLIDOS	V_NOMBRE	V_GENE
0	2014	ES	ABANADES GARCÍA, Juan Carlos	ABANADES GARCÍA	JUAN CARLOS	M
1	2014	ES	ABELLÁN SÁEZ, Gonzalo	ABELLÁN SÁEZ	GONZALO	M
2	2014	ES	AINZ IBARRONDO, Félix	AINZ IBARRONDO	FÉLIX	M
3	2014	ES	ALLONA ALBERICH, Isabel	ALLONA ALBERICH	ISABEL	Fem
4	2014	ES	ALONSO BEDATE, Carlos	ALONSO BEDATE	CARLOS	M
...	...	...	...	...	...	
5467	2023	DE	Zeller, Lenz Simon	Zeller	LENZ SIMON	Unkno
5468	2023	ES	ÁLVAREZ CARREÑO, Carlos	ÁLVAREZ CARREÑO	CARLOS	M
5469	2023	ES	ÁLVAREZ DE DIEGO, Javier	ÁLVAREZ DE DIEGO	JAVIER	M
5470	2023	ES	ÁLVAREZ-HERMS, Jesús	ÁLVAREZ-HERMS	JESÚS	M
5471	2023	ES	ÁVILA GARCÍA, Pedro	ÁVILA GARCÍA	PEDRO	M

5472 rows × 6 columns

Second round:

Now we try to assess gender for those "Unknown" cases that are also compound names, by looking only for the first word of the name.

In [14]:
```
#New DataFrame to work with:
df_names_epo_clim2 = pd.DataFrame()
df_names_epo_clim2=df_names_epo_clim

# Rename column V_NOMBRE_ORIG with original names (to leav
e a column V_NOMBRE for merging with gender dictionary)
df_names_epo_clim2.rename(columns={"V_NOMBRE": "V_NOMBRE_O
RIG"}, inplace=True)

# Add the new column `V_NOMBRE` with first word of compoun
d names ONLY if already "Unkonwn" gender
df_names_epo_clim2['V_NOMBRE'] = df_names_epo_clim2.apply(
    lambda row: row['V_NOMBRE_ORIG'].split()[0] if (
        row['V_GENERO'] == "Unknown" and
        row['V_NOMBRE_ORIG'] and
        isinstance(row['V_NOMBRE_ORIG'], str) and
        len(row['V_NOMBRE_ORIG'].split()) > 1
    ) else "",
    axis=1
)

df_names_epo_clim2
```

Out[14]:

	YEAR	V_PAIS	V_NOMBRE_COMPLETO	V_APELLIDOS	V_NOMBRE_ORIG	V_
0	2014	ES	ABANADES GARCÍA, Juan Carlos	ABANADES GARCÍA	JUAN CARLOS	
1	2014	ES	ABELLÁN SÁEZ, Gonzalo	ABELLÁN SÁEZ	GONZALO	
2	2014	ES	AINZ IBARRONDO, Félix	AINZ IBARRONDO	FÉLIX	
3	2014	ES	ALLONA ALBERICH, Isabel	ALLONA ALBERICH	ISABEL	
4	2014	ES	ALONSO BEDATE, Carlos	ALONSO BEDATE	CARLOS	
...	...	...	...	...	...	
5467	2023	DE	Zeller, Lenz Simon	Zeller	LENZ SIMON	
5468	2023	ES	ÁLVAREZ CARREÑO, Carlos	ÁLVAREZ CARREÑO	CARLOS	
5469	2023	ES	ÁLVAREZ DE DIEGO, Javier	ÁLVAREZ DE DIEGO	JAVIER	
5470	2023	ES	ÁLVAREZ-HERMS, Jesús	ÁLVAREZ-HERMS	JESÚS	
5471	2023	ES	ÁVILA GARCÍA, Pedro	ÁVILA GARCÍA	PEDRO	

5472 rows × 7 columns

.

Check remaining names (V_NOMBRE) again using "merge" with the Gender dictionary

In [15]:
```
df_names_epo_clim2 = pd.merge(df_names_epo_clim2, df_gende
r_dic, on=["V_NOMBRE", "V_PAIS"], how="left")
#df_names_epo_clim2
#NOTE: this merge create two columns: V_GENERO_X and V_GEN
ERO_y, now we fuse them
```

In [16]:
```
# TO get only one column GENERO:
# Fuse Genero_x and GEnero_y

#df_names_epo_clim.drop(columns=['V_GENERO'], inplace=Tru
e)
# First Ensure all values in V_GENERO_x and V_GENERO_y are
treated as strings
df_names_epo_clim2['V_GENERO_x'] = df_names_epo_clim2['V_G
ENERO_x'].astype(str)
df_names_epo_clim2['V_GENERO_y'] = df_names_epo_clim2['V_G
ENERO_y'].astype(str)
df_names_epo_clim2['V_GENERO'] = df_names_epo_clim2.apply(
    lambda row: (row['V_GENERO_x'] if row['V_GENERO_x'] no
t in [None, "Unknown", "nan"] else "") +
                (row['V_GENERO_y'] if row['V_GENERO_y'] no
t in [None, "Unknown", "nan"] else ""),
    axis=1
)

# Replace any residual NaN values with empty Unknonw
df_names_epo_clim2['V_GENERO'] = df_names_epo_clim2['V_GEN
ERO'].fillna("Unknown")

# Drop the original columns if no longer needed
df_names_epo_clim2.drop(columns=['V_GENERO_x', 'V_GENERO_
y'], inplace=True)

# Cleaning data:
df_names_epo_clim2['V_GENERO'] = df_names_epo_clim2['V_GEN
ERO'].replace({'F': 'Female', 'M': 'Male'})
df_names_epo_clim2['V_GENERO'] = df_names_epo_clim2['V_GEN
ERO'].replace("", "Unknown")  # Replace empty strings
df_names_epo_clim2
```

Out[16]:

	YEAR	V_PAIS	V_NOMBRE_COMPLETO	V_APELLIDOS	V_NOMBRE_ORIG	V_
0	2014	ES	ABANADES GARCÍA, Juan Carlos	ABANADES GARCÍA	JUAN CARLOS	
1	2014	ES	ABELLÁN SÁEZ, Gonzalo	ABELLÁN SÁEZ	GONZALO	
2	2014	ES	AINZ IBARRONDO, Félix	AINZ IBARRONDO	FÉLIX	
3	2014	ES	ALLONA ALBERICH, Isabel	ALLONA ALBERICH	ISABEL	
4	2014	ES	ALONSO BEDATE, Carlos	ALONSO BEDATE	CARLOS	
...	...	...	...	...	...	
5468	2023	DE	Zeller, Lenz Simon	Zeller	LENZ SIMON	
5469	2023	ES	ÁLVAREZ CARREÑO, Carlos	ÁLVAREZ CARREÑO	CARLOS	
5470	2023	ES	ÁLVAREZ DE DIEGO, Javier	ÁLVAREZ DE DIEGO	JAVIER	
5471	2023	ES	ÁLVAREZ-HERMS, Jesús	ÁLVAREZ-HERMS	JESÚS	
5472	2023	ES	ÁVILA GARCÍA, Pedro	ÁVILA GARCÍA	PEDRO	

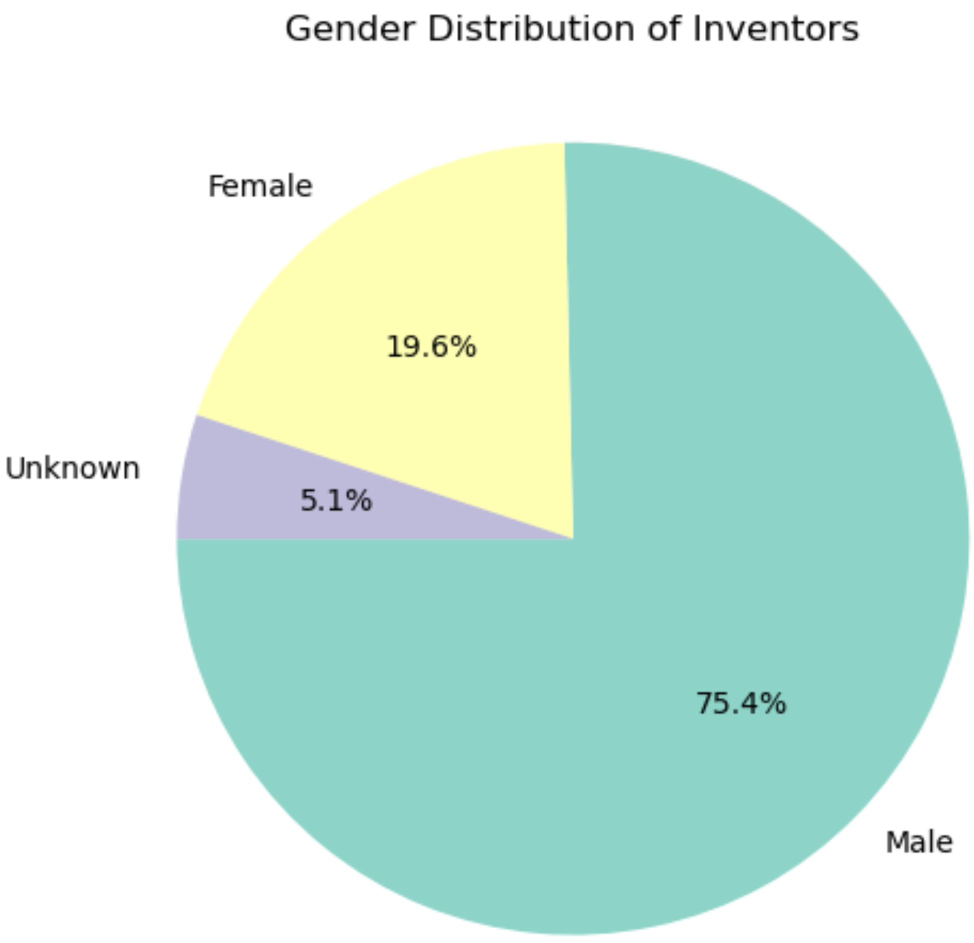
5473 rows × 7 columns

Gender gap chart

Pie graph showing the gender gap in EP inventions with ES origin, over the last 10 years and in the field of CCM.

It counts the occurences of each case: Male, Female, Unknown name; in the V_GENERO column.

In [17]:
```
import matplotlib.pyplot as plt

# First, it count occurrences
freq_clim = df_names_epo_clim2['V_GENERO'].value_counts()

# Plot the pie chart
plt.figure(figsize=(5, 5))
# Definir colores utilizando una paleta de colores
colors = plt.cm.Set3(range(len(freq_clim)))  # Colores par
a el primer gráfico

plt.pie(freq_clim, labels=freq_clim.index, autopct='%1.1
f%%', startangle=180, colors=colors)
plt.title("Gender Distribution of Inventors")
plt.axis('equal')  # Equal aspect ratio ensures the pie ch
art is circular.
plt.savefig("gender_distribution_pie_chart.png")  # Save t
he plot as an image file


# Mostrar los gráficos
plt.tight_layout()
plt.show()
freq_clim
```



Out[17]:
```
V_GENERO
Male       4124
Female     1072
Unknown     277
Name: count, dtype: int64
```

Geographic Distribution of Applicants

Installing "pygwalker" library

In [18]:
```
%pip install pygwalker
import pygwalker as pyg
```

```
Collecting pygwalker
  Downloading pygwalker-0.4.9.13-py3-none-any.whl.metadata
(20 kB)
```

```
Collecting anywidget (from pygwalker)
  Downloading anywidget-0.9.13-py3-none-any.whl.metadata (
7.2 kB)
Collecting appdirs (from pygwalker)
  Downloading appdirs-1.4.4-py2.py3-none-any.whl.metadata (
9.0 kB)
Requirement already satisfied: arrow in /opt/conda/lib/pyth
on3.11/site-packages (from pygwalker) (1.3.0)
Collecting astor (from pygwalker)
  Downloading astor-0.8.1-py2.py3-none-any.whl.metadata (4.
2 kB)
Requirement already satisfied: cachetools in /opt/conda/li
b/python3.11/site-packages (from pygwalker) (5.5.0)
Collecting duckdb<2.0.0,>=0.10.1 (from pygwalker)
  Downloading duckdb-1.1.3-cp311-cp311-manylinux_2_17_x86_6
4.manylinux2014_x86_64.whl.metadata (762 bytes)
Collecting gw-dsl-parser==0.1.49.1 (from pygwalker)
  Downloading gw_dsl_parser-0.1.49.1-py3-none-any.whl.metad
ata (1.2 kB)
Collecting ipylab<=1.0.0 (from pygwalker)
  Downloading ipylab-1.0.0-py3-none-any.whl.metadata (6.7 k
B)
Requirement already satisfied: ipython in /opt/conda/lib/py
thon3.11/site-packages (from pygwalker) (8.28.0)
Requirement already satisfied: ipywidgets in /opt/conda/li
b/python3.11/site-packages (from pygwalker) (8.1.5)
Requirement already satisfied: jinja2 in /opt/conda/lib/pyt
hon3.11/site-packages (from pygwalker) (3.1.4)
Collecting kanaries-track==0.0.5 (from pygwalker)
  Downloading kanaries_track-0.0.5-py3-none-any.whl.metadat
a (913 bytes)
Requirement already satisfied: numpy<2.0.0 in /opt/conda/li
b/python3.11/site-packages (from pygwalker) (1.26.4)
Requirement already satisfied: packaging in /opt/conda/lib/
python3.11/site-packages (from pygwalker) (24.1)
Requirement already satisfied: pandas in /opt/conda/lib/pyt
hon3.11/site-packages (from pygwalker) (2.2.3)
Requirement already satisfied: psutil in /opt/conda/lib/pyt
hon3.11/site-packages (from pygwalker) (5.9.8)
Requirement already satisfied: pyarrow in /opt/conda/lib/py
thon3.11/site-packages (from pygwalker) (17.0.0)
Requirement already satisfied: pydantic in /opt/conda/lib/p
ython3.11/site-packages (from pygwalker) (2.9.2)
Requirement already satisfied: pytz in /opt/conda/lib/pytho
n3.11/site-packages (from pygwalker) (2024.1)
Collecting quickjs (from pygwalker)
  Downloading quickjs-1.19.4-cp311-cp311-manylinux_2_17_x86
_64.manylinux2014_x86_64.manylinux_2_28_x86_64.whl.metadata
(590 bytes)
Requirement already satisfied: requests in /opt/conda/lib/p
ython3.11/site-packages (from pygwalker) (2.32.3)
Collecting segment-analytics-python==2.2.3 (from pygwalker)
  Downloading segment_analytics_python-2.2.3-py2.py3-none-a
```

```
ny.whl.metadata (1.6 kB)
Requirement already satisfied: sqlalchemy in /opt/conda/li
b/python3.11/site-packages (from pygwalker) (2.0.35)
Collecting sqlglot>=23.15.8 (from pygwalker)
  Downloading sqlglot-25.32.1-py3-none-any.whl.metadata (19
kB)
Requirement already satisfied: traitlets in /opt/conda/lib/
python3.11/site-packages (from pygwalker) (5.14.3)
Requirement already satisfied: typing-extensions in /opt/co
nda/lib/python3.11/site-packages (from pygwalker) (4.12.2)
Collecting wasmtime>=12.0.0 (from gw-dsl-parser==0.1.49.1->
pygwalker)
  Downloading wasmtime-27.0.2-py3-none-manylinux1_x86_64.wh
l.metadata (7.5 kB)
Collecting backoff>=2.2.1 (from kanaries-track==0.0.5->pygw
alker)
  Downloading backoff-2.2.1-py3-none-any.whl.metadata (14 k
B)
Collecting dateutils>=0.6.12 (from kanaries-track==0.0.5->p
ygwalker)
  Downloading dateutils-0.6.12-py2.py3-none-any.whl.metadat
a (1.3 kB)
Collecting monotonic~=1.5 (from segment-analytics-python==
2.2.3->pygwalker)
  Downloading monotonic-1.6-py2.py3-none-any.whl.metadata (
1.5 kB)
Requirement already satisfied: python-dateutil~=2.2 in /op
t/conda/lib/python3.11/site-packages (from segment-analytic
s-python==2.2.3->pygwalker) (2.8.2)
Requirement already satisfied: comm>=0.1.3 in /opt/conda/li
b/python3.11/site-packages (from ipywidgets->pygwalker) (0.
2.2)
Requirement already satisfied: widgetsnbextension~=4.0.12 i
n /opt/conda/lib/python3.11/site-packages (from ipywidgets-
>pygwalker) (4.0.13)
Requirement already satisfied: jupyterlab-widgets~=3.0.12 i
n /opt/conda/lib/python3.11/site-packages (from ipywidgets-
>pygwalker) (3.0.13)
Requirement already satisfied: decorator in /opt/conda/lib/
python3.11/site-packages (from ipython->pygwalker) (5.1.1)
Requirement already satisfied: jedi>=0.16 in /opt/conda/li
b/python3.11/site-packages (from ipython->pygwalker) (0.19.
1)
Requirement already satisfied: matplotlib-inline in /opt/co
nda/lib/python3.11/site-packages (from ipython->pygwalker)
(0.1.7)
Requirement already satisfied: prompt-toolkit<3.1.0,>=3.0.4
1 in /opt/conda/lib/python3.11/site-packages (from ipython-
>pygwalker) (3.0.48)
Requirement already satisfied: pygments>=2.4.0 in /opt/cond
a/lib/python3.11/site-packages (from ipython->pygwalker) (
2.18.0)
Requirement already satisfied: stack-data in /opt/conda/li
```

```
b/python3.11/site-packages (from ipython->pygwalker) (0.6.
2)
Requirement already satisfied: pexpect>4.3 in /opt/conda/li
b/python3.11/site-packages (from ipython->pygwalker) (4.9.
0)
Requirement already satisfied: charset-normalizer<4,>=2 in
/opt/conda/lib/python3.11/site-packages (from requests->pyg
walker) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/l
ib/python3.11/site-packages (from requests->pygwalker) (3.1
0)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/c
onda/lib/python3.11/site-packages (from requests->pygwalke
r) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in /opt/c
onda/lib/python3.11/site-packages (from requests->pygwalke
r) (2024.8.30)
Collecting psygnal>=0.8.1 (from anywidget->pygwalker)
  Downloading psygnal-0.11.1-cp311-cp311-manylinux_2_17_x86
_64.manylinux2014_x86_64.whl.metadata (7.2 kB)
Requirement already satisfied: types-python-dateutil>=2.8.1
0 in /opt/conda/lib/python3.11/site-packages (from arrow->p
ygwalker) (2.9.0.20241003)
Requirement already satisfied: MarkupSafe>=2.0 in /opt/cond
a/lib/python3.11/site-packages (from jinja2->pygwalker) (3.
0.1)
Requirement already satisfied: tzdata>=2022.7 in /opt/cond
a/lib/python3.11/site-packages (from pandas->pygwalker) (20
24.2)
Requirement already satisfied: annotated-types>=0.6.0 in /o
pt/conda/lib/python3.11/site-packages (from pydantic->pygwa
lker) (0.7.0)
Requirement already satisfied: pydantic-core==2.23.4 in /op
t/conda/lib/python3.11/site-packages (from pydantic->pygwal
ker) (2.23.4)
Requirement already satisfied: greenlet!=0.4.17 in /opt/con
da/lib/python3.11/site-packages (from sqlalchemy->pygwalke
r) (3.1.1)
Requirement already satisfied: parso<0.9.0,>=0.8.3 in /opt/
conda/lib/python3.11/site-packages (from jedi>=0.16->ipytho
n->pygwalker) (0.8.4)
Requirement already satisfied: ptyprocess>=0.5 in /opt/cond
a/lib/python3.11/site-packages (from pexpect>4.3->ipython->
pygwalker) (0.7.0)
Requirement already satisfied: wcwidth in /opt/conda/lib/py
thon3.11/site-packages (from prompt-toolkit<3.1.0,>=3.0.41-
>ipython->pygwalker) (0.2.13)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/p
ython3.11/site-packages (from python-dateutil~=2.2->segment
-analytics-python==2.2.3->pygwalker) (1.16.0)
Requirement already satisfied: importlib_resources>=5.10 in
/opt/conda/lib/python3.11/site-packages (from wasmtime>=12.
0.0->gw-dsl-parser==0.1.49.1->pygwalker) (6.4.5)
```

```
Requirement already satisfied: executing>=1.2.0 in /opt/con
da/lib/python3.11/site-packages (from stack-data->ipython->
pygwalker) (2.1.0)
Requirement already satisfied: asttokens>=2.1.0 in /opt/con
da/lib/python3.11/site-packages (from stack-data->ipython->
pygwalker) (2.4.1)
Requirement already satisfied: pure-eval in /opt/conda/lib/
python3.11/site-packages (from stack-data->ipython->pygwalk
er) (0.2.3)
Downloading pygwalker-0.4.9.13-py3-none-any.whl (4.5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.5/4.5 MB 122.
2 MB/s eta 0:00:00
Downloading gw_dsl_parser-0.1.49.1-py3-none-any.whl (956 k
B)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 956.2/956.2 kB
89.4 MB/s eta 0:00:00
Downloading kanaries_track-0.0.5-py3-none-any.whl (8.6 kB)
Downloading segment_analytics_python-2.2.3-py2.py3-none-an
y.whl (24 kB)
Downloading duckdb-1.1.3-cp311-cp311-manylinux_2_17_x86_64.
manylinux2014_x86_64.whl (20.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 20.1/20.1 MB 11
4.6 MB/s eta 0:00:00
Downloading ipylab-1.0.0-py3-none-any.whl (100 kB)
Downloading sqlglot-25.32.1-py3-none-any.whl (434 kB)
Downloading anywidget-0.9.13-py3-none-any.whl (213 kB)
Downloading appdirs-1.4.4-py2.py3-none-any.whl (9.6 kB)
Downloading astor-0.8.1-py2.py3-none-any.whl (27 kB)
Downloading quickjs-1.19.4-cp311-cp311-manylinux_2_17_x86_6
4.manylinux2014_x86_64.manylinux_2_28_x86_64.whl (2.2 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.2/2.2 MB 129.
7 MB/s eta 0:00:00
Downloading backoff-2.2.1-py3-none-any.whl (15 kB)
Downloading dateutils-0.6.12-py2.py3-none-any.whl (5.7 kB)
Downloading monotonic-1.6-py2.py3-none-any.whl (8.2 kB)
Downloading psygnal-0.11.1-cp311-cp311-manylinux_2_17_x86_6
4.manylinux2014_x86_64.whl (717 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 717.6/717.6 kB
59.7 MB/s eta 0:00:00
Downloading wasmtime-27.0.2-py3-none-manylinux1_x86_64.whl
(6.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.4/6.4 MB 140.
7 MB/s eta 0:00:00
Installing collected packages: quickjs, monotonic, appdirs,
wasmtime, sqlglot, psygnal, duckdb, backoff, astor, segment
-analytics-python, gw-dsl-parser, dateutils, kanaries-trac
k, ipylab, anywidget, pygwalker
Successfully installed anywidget-0.9.13 appdirs-1.4.4 astor
-0.8.1 backoff-2.2.1 dateutils-0.6.12 duckdb-1.1.3 gw-dsl-p
arser-0.1.49.1 ipylab-1.0.0 kanaries-track-0.0.5 monotonic-
1.6 psygnal-0.11.1 pygwalker-0.4.9.13 quickjs-1.19.4 segmen
t-analytics-python-2.2.3 sqlglot-25.32.1 wasmtime-27.0.2
Note: you may need to restart the kernel to use updated pac
```

```
kages.
```

PATSTAT query searching for the origin of the applicant: Spanish provinces (NUTS 3 (https://ec.europa.eu/eurostat/web/nuts/overview))

In [19]:

```
# Number of EPO applications from ES origin
q_ori = patstat.sql_query("""
    SELECT P.nuts AS NUTS, N.nuts_label AS Region, COUNT(D
ISTINCT A.appln_id) AS Num_applications
    FROM tls201_appln A
    JOIN tls224_appln_cpc C ON A.appln_id = C.appln_id
    JOIN tls207_pers_appln Q ON A.appln_id = Q.appln_id
    JOIN tls206_person P ON Q.person_id = P.person_id
    JOIN tls904_nuts N ON P.nuts = N.nuts
    WHERE A.earliest_publn_year BETWEEN 2014 AND 2023
    AND A.appln_auth = 'EP'
    AND P.person_ctry_code = 'ES'
    AND A.appln_kind = 'A '
    AND Q.applt_seq_nr = 1
    AND C.cpc_class_symbol LIKE 'Y02%'
    AND P.NUTS_LEVEL in (3,4)
    GROUP BY P.nuts, N.nuts_label
    ORDER BY P.nuts

""", use_legacy_sql = False)

#del df_INori
df_ori = pd.DataFrame(q_ori)
df_ori = df_ori.sort_values(by='Num_applications', ascendi
ng=False).reset_index(drop=True)
df_ori.head()
```

Out[19]:

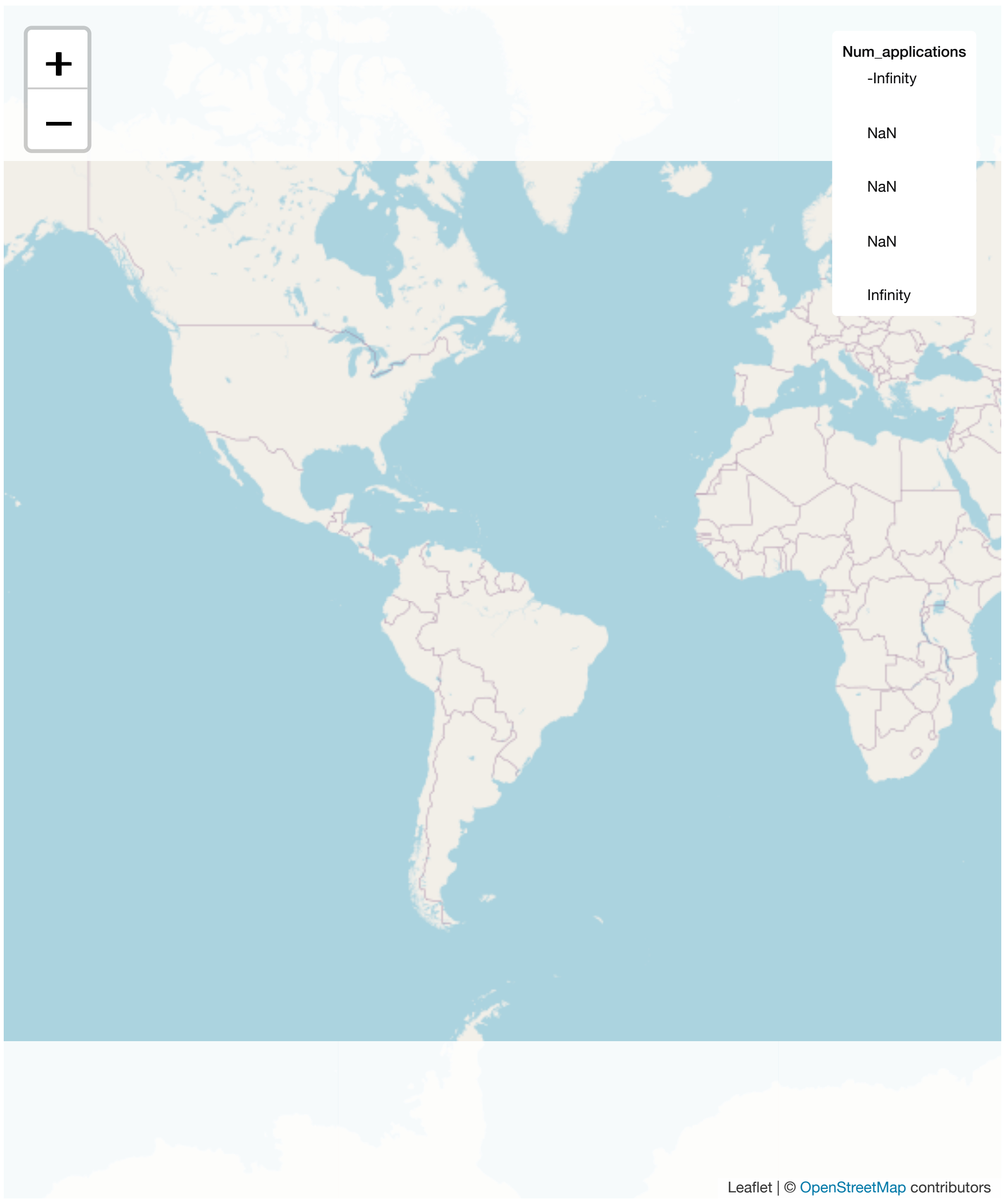
	NUTS	Region	Num_applications
0	ES300	Madrid	476
1	ES511	Barcelona	438
2	ES220	Navarra	272
3	ES213	Bizkaia	100
4	ES212	Gipuzkoa	93

Map of patent applications (CCMT) by Spanish provinces

In [20]:
```
import json

# Step 2: Load the saved JSON configuration
with open('pygwalker_spec_ESPprov.json', 'r') as file:  # Replace 'saved_graph.json' with your JSON file path
    saved_spec = json.load(file)

# Step 3: Use the JSON configuration to render the graph
pyg.walk(df_ori, spec=saved_spec)
```

```
Box(children=(HTML(value='\n<div id="ifr-pyg-00062809d1b74e1fLcjP04kKOGmT8iZ9" style="height: auto">\n    <hea…
```

Chart 1



Out[20]:
```
<pygwalker.api.pygwalker.PygWalker at 0x7b59b65b8b90>
```

CONCLUSIONS

Early stage tests with TIP

- **TIP** shows a **very good performance**
- **Python** libraries provide access to a wide variety of output formats, figures and data analysis features
- It opens new **patent data analysis** possibilities for us
- At present: patent coverage limitations (looking forward to new versions and improvements)

Thanks!

Special thanks to **Carlos Albert** and **Luis Priegue** (OEPM) for the preparation of the code; and to **Carlos Aitor Pérez de Unzueta** and **Borja Rojano** (EPO) for all the support with TIP.

Mario Cañadas Castro (**mario.canadas@oepm.es**)