```
 12  max_speed           1452797 non-null  int64
 13  motorcar            1452797 non-null  int64
 14  geometry            1452797 non-null  geometry
 15  time                1452797 non-null  float64
 16  time_alt            1452797 non-null  float64
 17  isochrone_u         1452797 non-null  float64
 18  isochrone_v         1452797 non-null  float64
 19  isochrone_u_safety  1452797 non-null  float64
 20  isochrone_v_safety  1452797 non-null  float64
 21  code                1452797 non-null  object
dtypes: float64(7), geometry(1), int64(10), object(4)
memory usage: 254.9+ MB
```

In [123]:
```
df[['highway', 'level','bicycle', 'bicycle_safety', 'geometry']].to_file('iris-30min-safe.json', driver='GeoJSON')
```

jobs KDE

In [30]:
```
jobs = pd.read_excel('/Users/fabien/Dropbox/low-carbon-lille/cartographic-resources/etablissements.xlsx')
```

In [31]:
```
jobs.head()
```

Out[31]:

	siren	nic	siret	statutDiffusionEtablissement	dateCreationEtablissement
0	46350088	103	4635008800103	O	2011-01-01
1	47050174	78	4705017400078	O	1997-12-31
2	57812968	717	5781296800717	O	2008-09-01
3	62501275	93	6250127500093	O	2010-04-27
4	65801243	133	6580124300133	O	1998-12-18

5 rows × 80 columns

In [32]:
```
jobs = gpd.GeoDataFrame(jobs[['siren','nbr_emplois']], geometry=gpd.points_from_xy(jobs['lng'], jobs['lat']), crs=4326)
```

we need to remove the empty geometry and teh one outside the MEL boundaries

In [34]:
```
jobs = jobs[~jobs.isna()]
```

In [37]:
```
jobs = jobs.to_crs(3950)
```

In [44]:
```
jobsMel = gpd.sjoin(jobs,mel_boundary,op='intersects')
```

In [49]:
```
jobsMel.head()
```

Out[49]:

	siren	nbr_emplois	geometry	index_right	name
0	46350088	34.5	POINT (1704127.799 9272507.596)	0	mel_boundary
1	47050174	34.5	POINT (1705605.969 9274968.518)	0	mel_boundary
2	57812968	34.5	POINT (1708379.752 9264421.387)	0	mel_boundary
3	62501275	4.0	POINT (1712816.019 9276681.385)	0	mel_boundary
4	65801243	14.5	POINT (1692661.236 9272836.986)	0	mel_boundary

In [51]:
```
jobsMel.plot(markersize=0.1)
```

Out[51]: <AxesSubplot:>

In [52]:
```
jobsMel.nbr_emplois = round(jobsMel.nbr_emplois)
```

create one point for each job using nbr_emplois

In [54]:
```
jobsMelExtended = jobsMel.loc[jobsMel.index.repeat(jobsMel.nbr_emplois)].reset_index(drop=True)
```

In [55]:
```
jobsMelExtended.info()
```

```
<class 'geopandas.geodataframe.GeoDataFrame'>
RangeIndex: 501986 entries, 0 to 501985
Data columns (total 5 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   siren        501986 non-null  int64
 1   nbr_emplois  501986 non-null  float64
 2   geometry     501986 non-null  geometry
 3   index_right  501986 non-null  int64
 4   name         501986 non-null  object
dtypes: float64(1), geometry(1), int64(2), object(1)
memory usage: 19.1+ MB
```

version 1, simply map

now we can apply the KDE. We use geoplot for a kdeplot uncontrolled. See (https://residentmario.github.io/geoplot/gallery/plot_nyc_collision_factors.html)[geoplot kde]

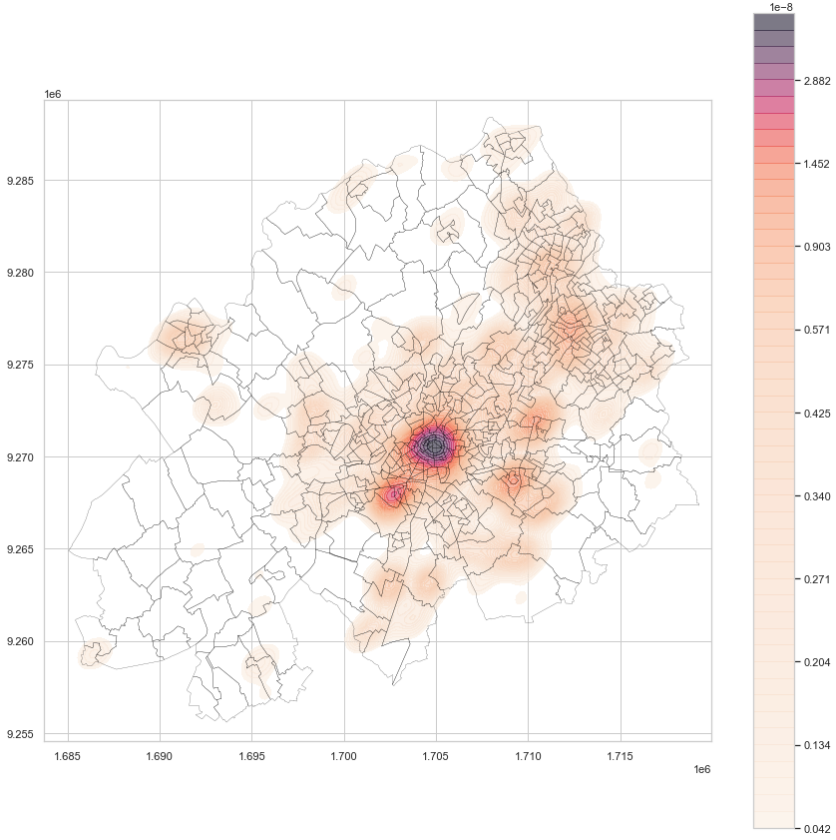
In [82]:
```
import descartes
```

In [68]:
```
sns.set_style('whitegrid')
```

In [107…
```
f, ax = plt.subplots(1, figsize=(15, 15))
# Generate and add KDE with a shading of 50 gradients
# coloured contours, 75% of transparency,
# and the reverse viridis colormap
sns.kdeplot(
    x = jobsMelExtended.geometry.x,
    y = jobsMelExtended.geometry.y,
    n_levels=50,
    fill=True,
    alpha=0.55,
    cmap='rocket_r',
    cbar=True,
    bw_adjust=1
)

irisMEL.boundary.plot(ax=ax,linewidth=0.2, color='black')
```

Out[107…
```
<AxesSubplot:>
```

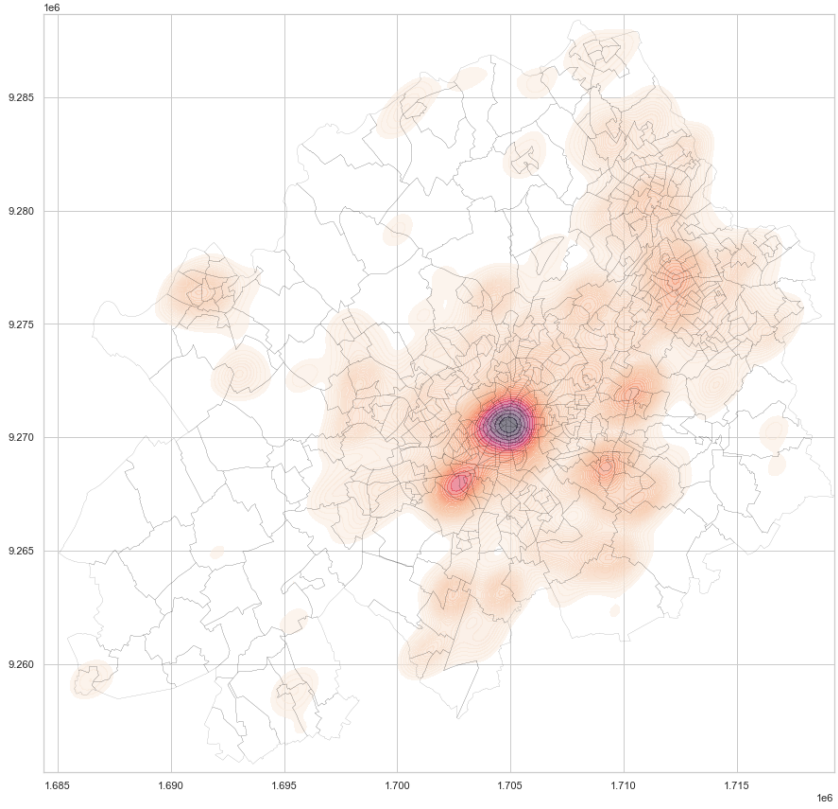


In [109…
```
f, ax = plt.subplots(1, figsize=(15, 15))

sns.kdeplot(
    x = jobsMel.geometry.x,
    y = jobsMel.geometry.y,
    n_levels=50,
    fill=True,
    alpha=0.5,
    multiple='layer',
    cmap='rocket_r',
    cbar=False,
    weights=jobsMel.nbr_emplois,
    bw_adjust=0.35
)

irisMEL.boundary.plot(ax=ax,linewidth=0.1, color='black')
```

Out[109…
```
<AxesSubplot:>
```



In [110…
```
f.savefig('kde-emplois.pdf', dpi=300)
```

In [111…
```
dfSafe30m = gpd.read_file('/Users/fabien/Dropbox/low-carbon-lille/iris-30min-safe/iris-30min-safe.json')
```

In [112…

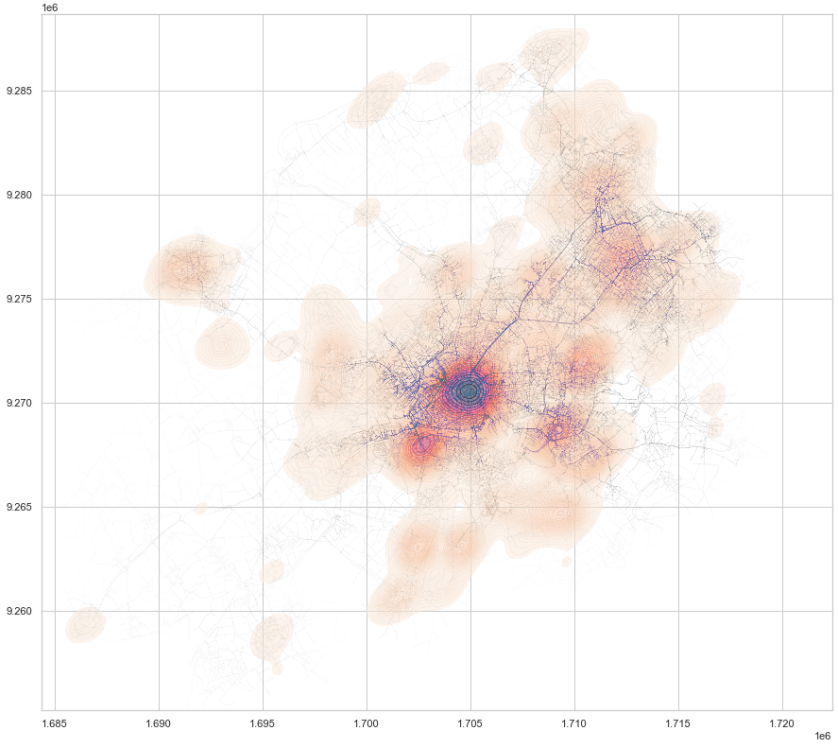
```
f, ax = plt.subplots(1, figsize=(15, 15))

sns.kdeplot(
    x = jobsMel.geometry.x,
    y = jobsMel.geometry.y,
    n_levels=50,
    fill=True,
    alpha=0.5,
    multiple='layer',
    cmap='rocket_r',
    cbar=False,
    weights=jobsMel.nbr_emplois,
    bw_adjust=0.35
)

dfSafe30m.plot(linewidth=0.1, alpha=0.05, color='b', ax=ax)
```

Out[112…

```
<AxesSubplot:>
```



In [113…

```
f.savefig('kde-emplois-iris30Msafe-stack.pdf', dpi=300)
```