Cartographie élémentaire

ATTENTION RELANCEZ LES PROGRAMMES POUR OBTNIR LE RÉSULTAT VISUEL QUAND IL Y A UNE FONCTION DYNAMIQUE, NOTAMMENT DANS LA CONCLUSION

Les outils cartographiques intégrés dans Mathematica sont très puissants et très variés. Nous en livrons une première présentation, relativement simple. ***Pour obtenir les résultats affichés, il faut disposer d'une liaison Internet afin d'accéder aux données externes, qui sont dans la base de connaissance de Mathematica***.

Modèles cartographiques généraux

Plusieurs fonctions de cartographie sont générales. Elles construisent des types de cartes. Les fonctions essentielles à connaître sont : **GeoListPlot[], GeoBubbleChart[], GeoRegionValuePlot[] et GeoGraphics[]**.

GeoListPlot[] pour localiser des points, des entités spatiales et des lignes sur une carte

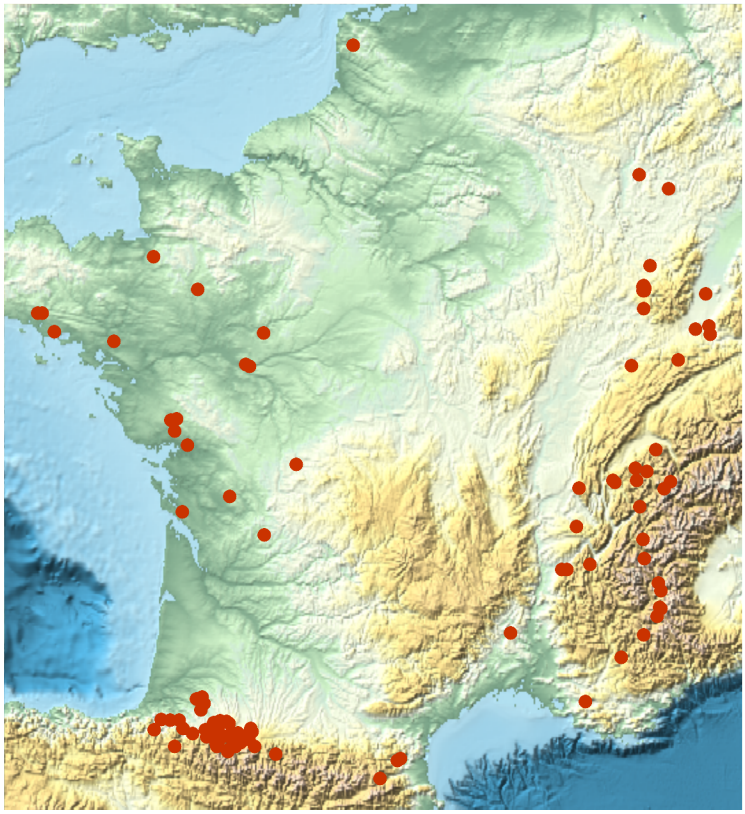
Avec **GeoListPlot[]**, on localise des points, soit des données issues de la base de connaissance, soit ses propres données. Prenons dans la base de connaissance les séismes de magnitude 4 enregistrés en France du 01/01/1980 au 31/12/2015, avec la ligne d'instruction suivante :

In[146]:=
```
seisme = EarthquakeData[Entity["Country", "France"],
          (données sismiques)  (entité)
  4, {{1980, 1, 1}, {2014, 12, 31}}, "Position"]["Values"];
                                      (position)  (valeurs)
```

Il est alors possible d'en faire une représentation cartographique avec la ligne d'instruction suivante :

In[147]:=
```
GeoListPlot[seisme, GeoBackground → "ReliefMap"]
```
représentation géographiqu··· fond géographique

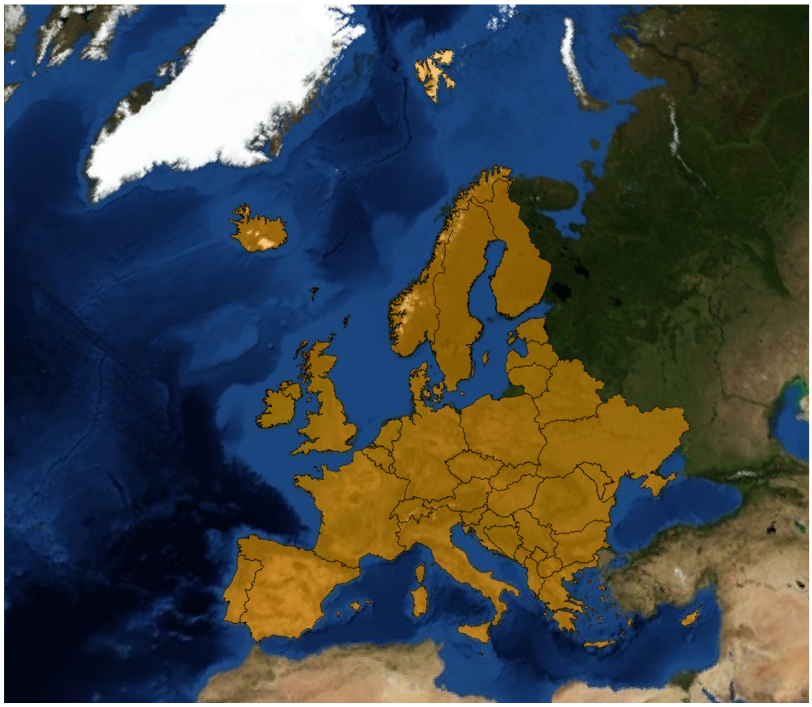
Out[147]=



Volontairement, nous avons retenu le relief en option comme fond de carte.
GeoListPlot[] s'utilise aussi pour cartographier des entités, par exemple l'Europe avec en option une photographie satellite en fonde de carte :

In[105]:=
```
GeoListPlot[EntityClass["Country", "Europe"], GeoBackground -> "Satellite"]
```
représentation ··· classe d'entités fond géographique

Out[105]=



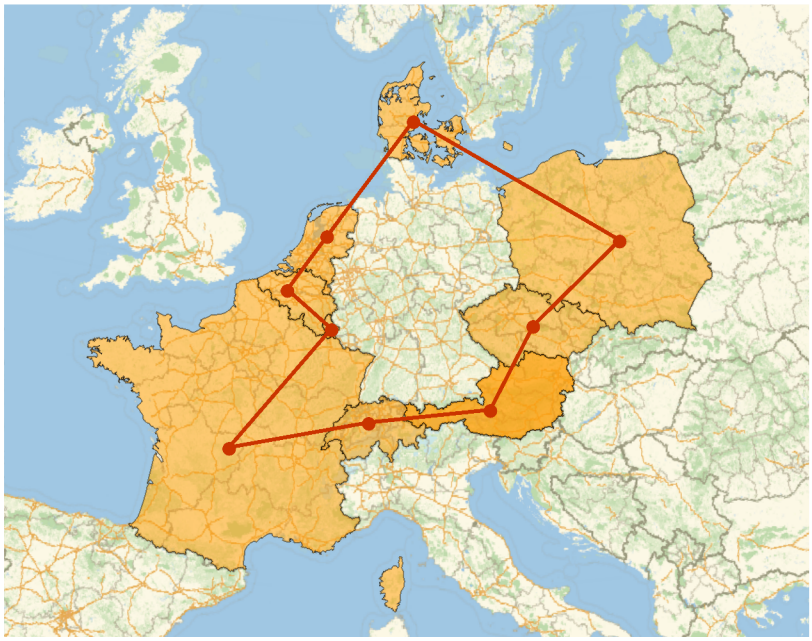
Il est possible d'illustrer ses propres données en fournissant les coordonnées et même des données calculées. Dans le programme ci-dessous Mathematica trace le plus court chemin entre les pays qui bordent l'Allemagne.

In[•]:=
```
etat = CountryData["Germany", "BorderingCountries"]
       données de pays
order = Last[FindShortestTour[GeoPosition[etat]]]
        dern··· trouve le circuit le plus ··· position géographique
GeoListPlot[etat[[order]], Joined → True]
représentation géographique de lieux  joint  vrai
```

Out[•]= { Austria , Belgium , Czech Republic , Denmark ,
France , Luxembourg , Netherlands , Poland , Switzerland }

Out[•]= {1, 3, 8, 4, 7, 2, 6, 5, 9, 1}

Out[•]=



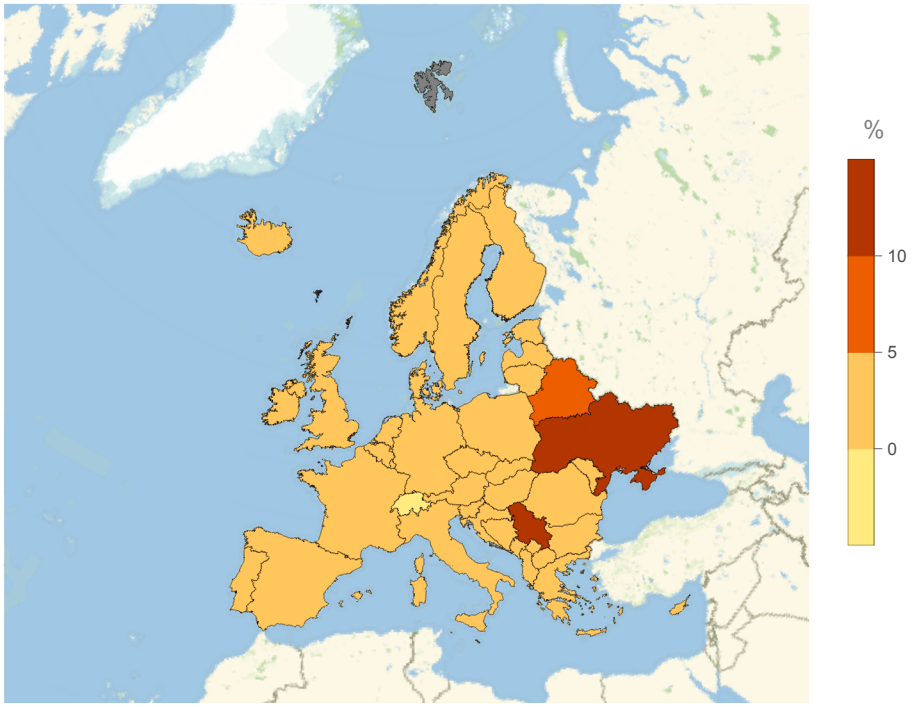
Volontairement, nous affichons sur l'écran les résultats intermédiaires. Apparaît donc la liste des États, puis le plus court chemin, et enfin la carte qui correspond aux États voisins avec le tracé du plus court chemin.

Pour associer des valeurs fréquentielles à des régions employer la fonction GeoRegionValuePlot[]

Quand le géographe traite des données fréquentielles, analyse des taux, les représentations par plage de couleurs sont à privilégier. C'est ce que réalise la fonction **GeoRegionValuePlot**[]

In[ ]:=
```
GeoRegionValuePlot[EntityClass["Country", "Europe"] -> "DepositInterestRate"]
```
représentation géographiq··· classe d'entités

Out[ ]=



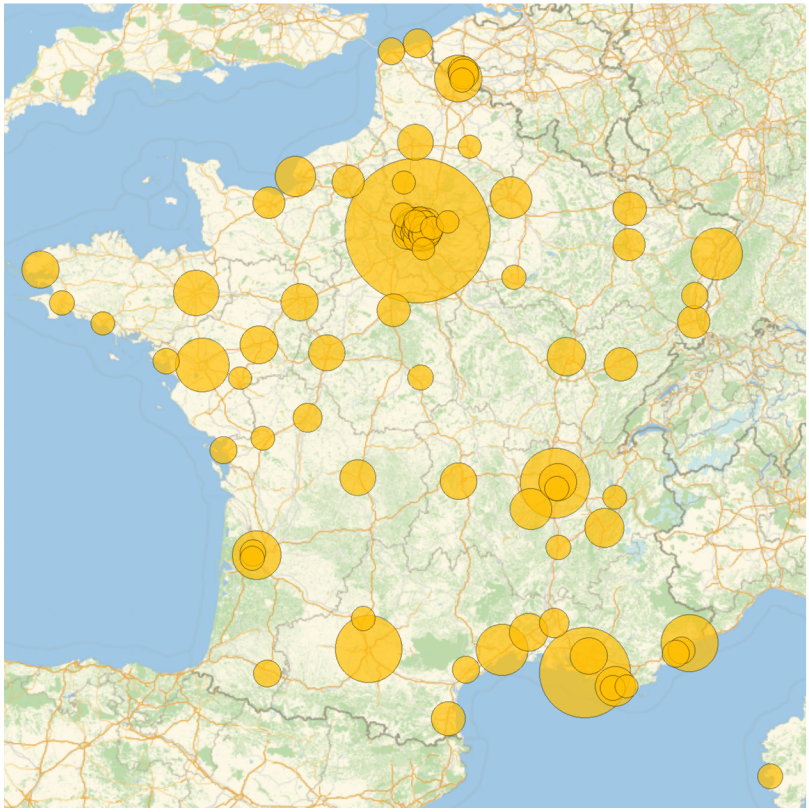
Le programme ci-dessus donne la répartition pat pays des taux d'intérêt accordés aux dépôts financiers. De très nombreuses options sont accessibles pour modifier les couleurs, la légende et même faire apparaître le nom des États quand le curseur passe dessus. Rappelons que ce type de carte est surtout valable pour des taux. Pour des valeurs brutes, il est préférable d'utiliser la fonction **GeoBubbleChart[].**

Une représentation des quantités avec GeoBubbleChart[]

Attention, il faut attendre que le programme cherche les données dans la base de données **CityData**. Ce qui prend quelques minutes.

In[ ]:=
```
data = Table[{city, city["Population"]},
  table
    {city, CountryData["France", "LargestCities"]}];
          données de pays
GeoBubbleChart[
bulles aux positions géographiques
 data]
```

Out[ ]=



Un outil complet : la fonction GeoGraphics[]

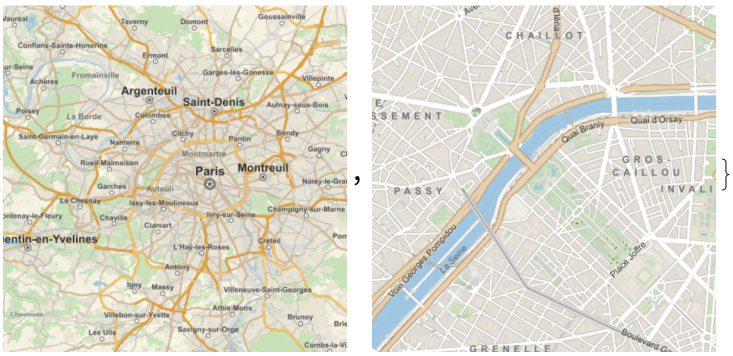
La fonction **GeoGraphics**[] permet de réaliser des cartes de toute nature. Et en premier lieu d'importer des cartes à des échelles différentes. Ces cartes sont des objets graphiques et non pas des objets images.

```
carte1 =
  GeoGraphics[ Eiffel Tower BUILDING , GeoResolution → Quantity[100, "Meters"]];
              carte géographique          résolution géographi···  quantité

carte2 = GeoGraphics[ Eiffel Tower BUILDING , GeoResolution → Quantity[10, "Yards"]];
            carte géographique                 résolution géographi···  quantité

carto = {carte1, carte2}
```

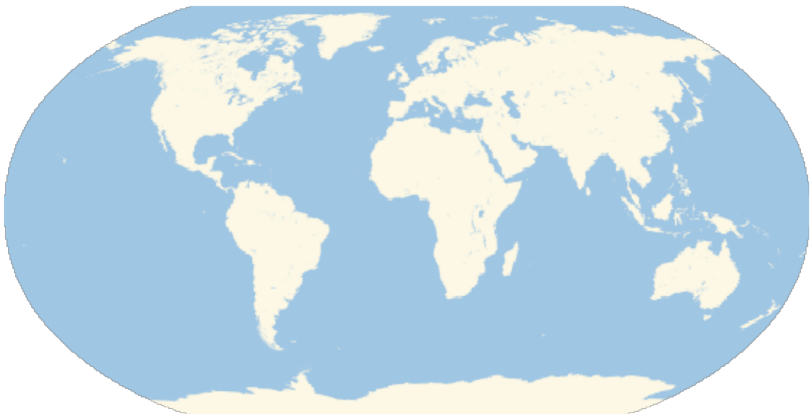
Out[ ]=



Les deux cartes sont élaborées à partir de la Tour Eiffel à Paris, avec deux résolutions différentes. Les deux instructions se terminant par un ; elles n'apparaissent pas à l'écran. La troisième ligne, positionne les deux cartes créées avec la fonction **GeoGraphics**[] dans une liste, comme l'attestent les deux accolades. Ces cartes peuvent provenir du serveur Wolfram, mais aussi d'autres serveurs, par exemple le serveur *Open-street*. Et le géographe peut faire appel à différentes projections. Ci-dessous la carte du monde en projection Robinson.

In[ ]:=
```
GeoGraphics[GeoRange → "World", GeoProjection → "Robinson"]
carte géograph···  gamme géographique  projection cartographique
```

Out[ ]=

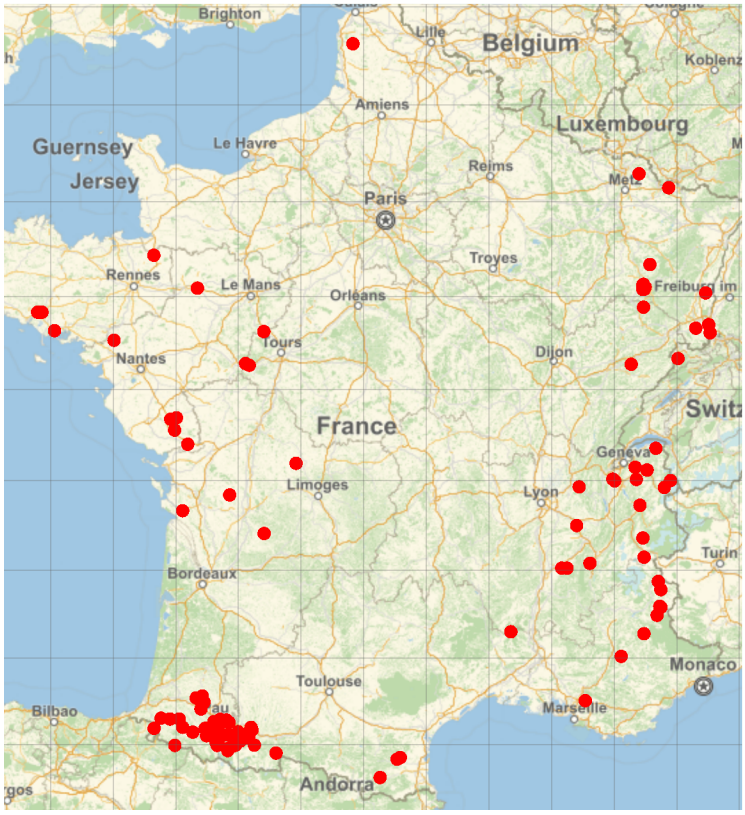


Il est possible, avec quelques aménagements de reproduire une carte réalisée avec une autre fonction, par exemple la carte des séismes présentée au début de ce notebook avec **GeoListPlot**[].

In[107]:=
```
GeoGraphics[{Polygon[Entity["France"]], Red, PointSize[Large], Point[seisme]},
 GeoGridLines → Automatic]
```
carte géographique | polygone | entité | rouge | taille des poi… | taille gr… | point
réticule géographique | automatique

GeoGraphics: Unable to obtain location information for Entity[France].

Out[107]=



Et surtout, il est possible de combiner des graphiques. Le lecteur doit se référer à l'aide pour voir toutes les options fort nombreuses qui sont disponibles. Dans le programme ci-dessous, le graphique dessine le tracé d'une équation sur la carte de France.

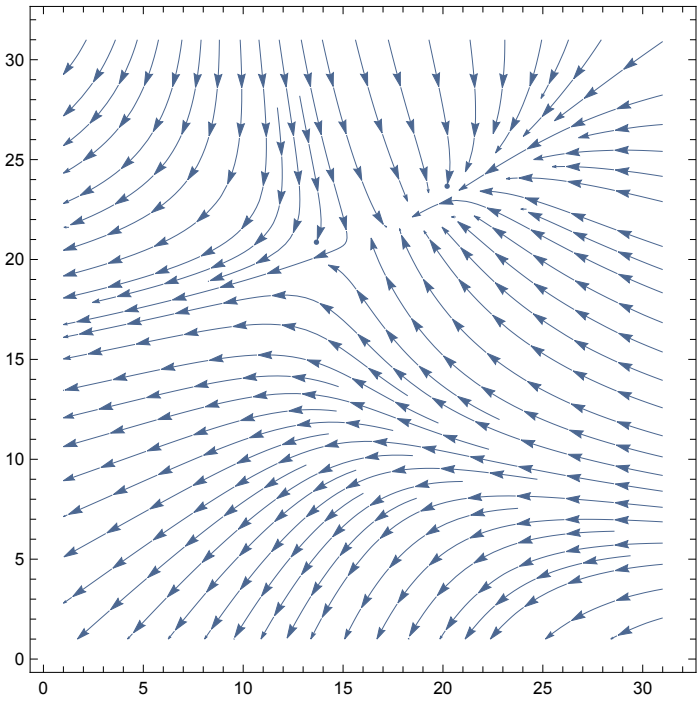
In[206]:=
```
function = Table[{-1 - x^2 + y, 1 + x - y^2}, {x, -3, 3, 0.2}, {y, -3, 3, 0.2}];
```
table

```
tracefunction = ListStreamPlot[function]
```
tracé stream de liste

```
GeoGraphics[{GeoStyling[{"Image", Rasterize[tracefunction]}],
```
carte géographique | style de carte gé··· | image | convertis en carte de bits

```
  Polygon[Entity["Country", "France"]]}, Frame → True]
```
polygone | entité | cadre | vrai

Out[207]=



Out[208]=

Zoomer les cartes avec la fonction DynamicGeoGraphics[]

Il est possible de réaliser des zoom comme sur Google map. Pour cela on utilise la fonction **DynamicGeo-Graphics**[] :

```
In[•]:= DynamicGeoGraphics[ Eiffel Tower BUILDING , GeoRange → Quantity[1, "Miles"]]
```

graphiques géographiques dynamiques | gamme géog·· | quantité

Out[•]=

Il suffit de cliquer sur les cases + ou -, en bas à froide pour obtenir des cartes à différentes échelles. Et cette fonction offre des options similaires à celles qui fonctionnent pour **GeoGraphics**[].

Modèles cartographiques spécifiques

Mathematica offre aussi des outils de cartographie spécifiques, adaptées à un type particulier de données

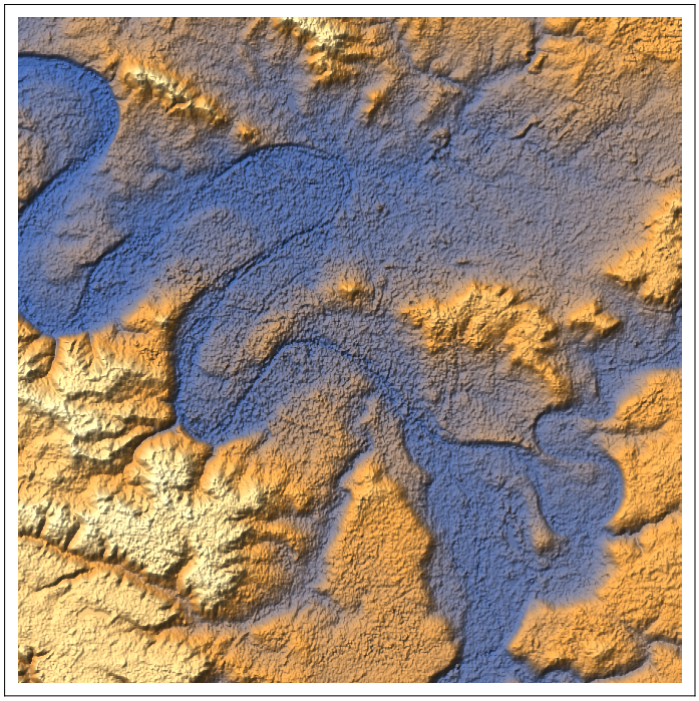
ou de traitement.

Réaliser des cartes d'un Modèle Numérique de Terrain avec ReliefPlot[]

Avec la fonction **GeoElevationData**[], le géographe récupère les données d'altitude et les coordonnées d'un espace étudié, autour de la ville de Paris dans cet exemple. Et divers graphiques peuvent donner une représentation visuelle de ces données, notamment une carte en relief ou un graphique en trois dimensions. Le programme ci-dessous importe les données, puis trace une carte en relief avec la fonction **ReliefPlot**[].

```
dataMnt = GeoElevationData[Entity["City", {"Paris", "IleDeFrance", "France"}],
          |données d'élévation gé··· |entité
   GeoProjection -> Automatic];
   |projection cartographique |automatique
ReliefPlot[dataMnt, DataReversed -> True]
|tracé en relief          |données inversées    |vrai
```

Out[•]=

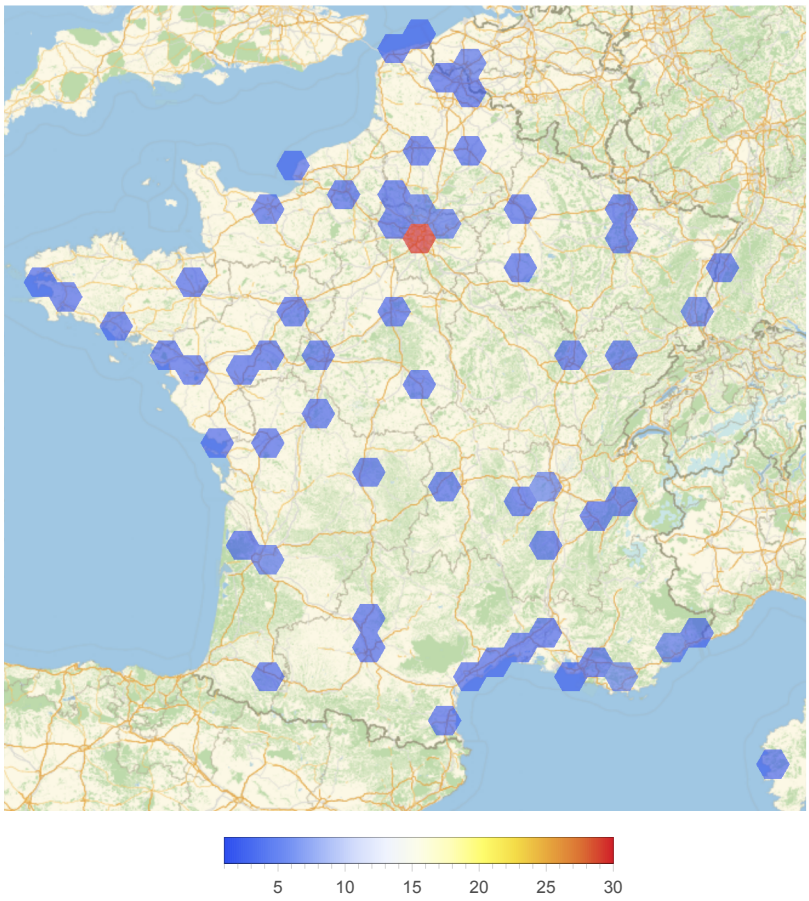


Illustrer la répartition de points avec la fonction GeoHistogram[]

D'abord un exemple simple, prenant en compte uniquement la localisation des plus grandes villes françaises.

In[148]:=
```
datavilles = EntityValue[CountryData["France", "LargestCities"], "Position"];
GeoHistogram[datavilles, ColorFunction → "TemperatureMap",
 PlotLegends → Placed[Automatic, Below]]
```

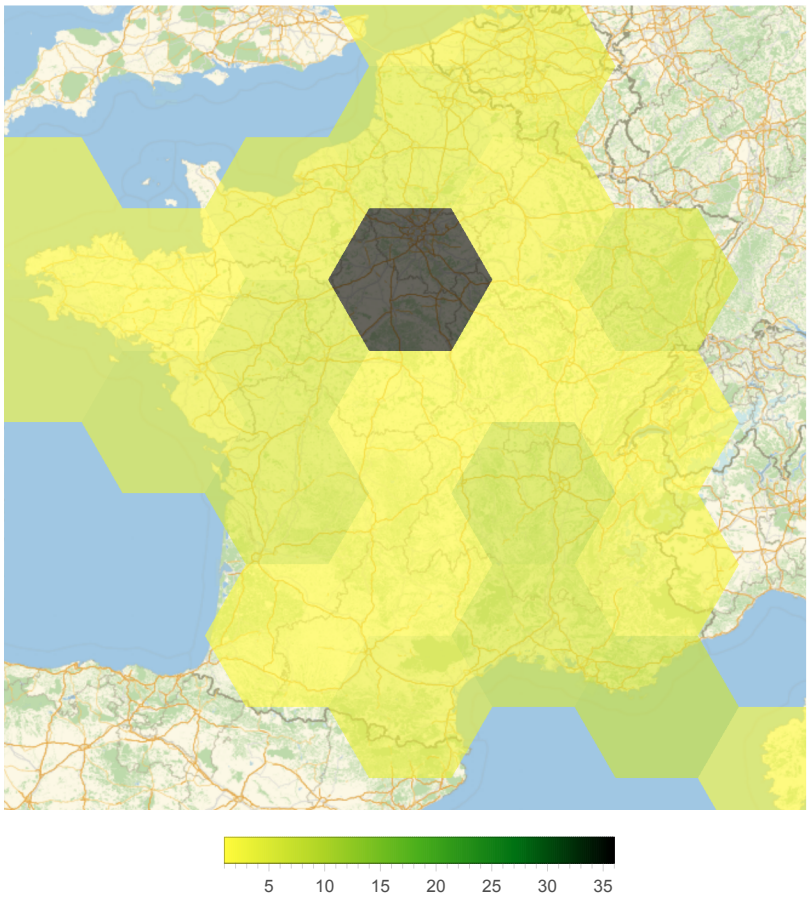
Out[149]=



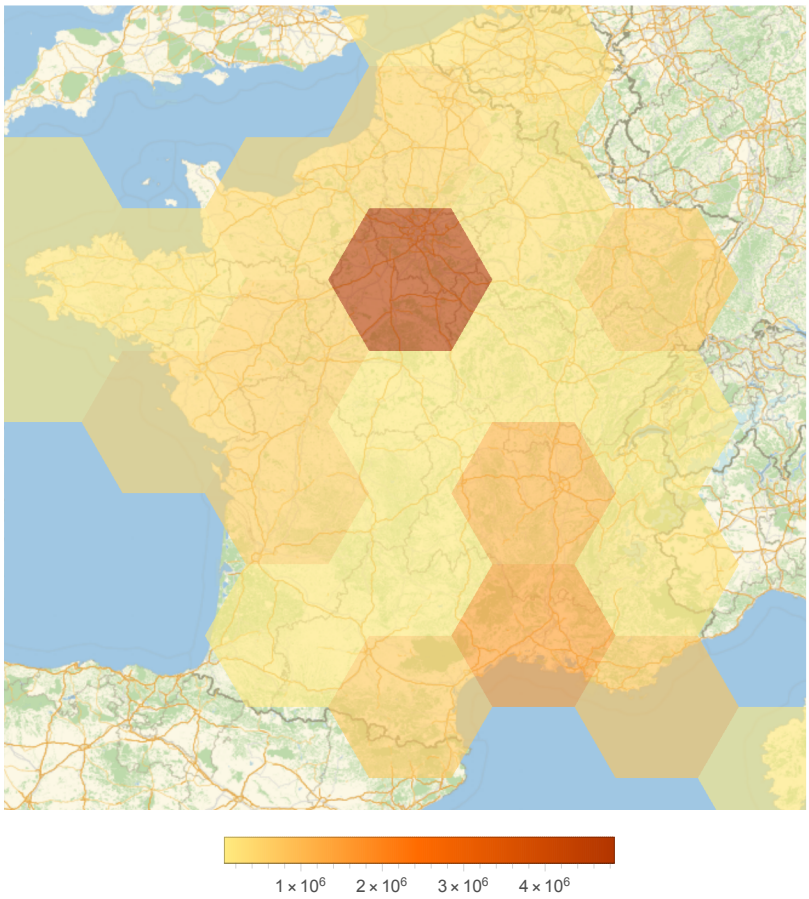
Il est possible de modifier le nombre de classes, puis si on prend en compte le poids démographique des villes, il faut collecter la donnée population pour calculer des histogrammes pondérés, et on obtient alors :

In[164]:=
```
GeoHistogram[datavilles, {"Total", 24},
 ColorFunction → ColorData[{"AvocadoColors", "Reverse"}],
 PlotLegends → Placed[Automatic, Below]]
datavillespop = WeightedData[
    CountryData["France", "LargestCities"], #["Population"] &];
GeoHistogram[datavillespop, {"Total", 24},
 PlotLegends → Placed[Automatic, Below]]
```

Out[164]=



Out[166]=

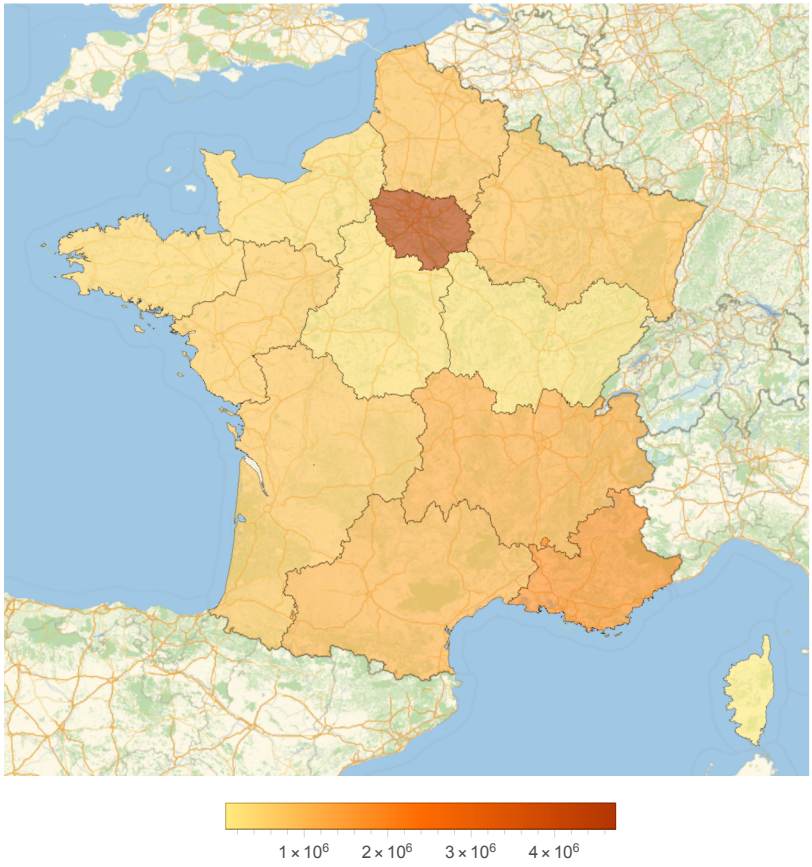


Avec les valeurs pondérées, on remarque que l'aire lyonnaise et marseillaise s'individualisent. Les options permettent de modifier le type d'histogramme et sa forme. Il est même possible de retenir comme forme les divisions administratives, comme le montre l'exemple ci-dessous :

```
In[ ]:= regions = EntityValue[ administrative division , "Entities"];
                  valeur d'entité
```

```
GeoHistogram[datavillespop, regions, PlotLegends → Placed[Automatic, Below]]
histogramme géo                    légendes de tracé  placé  automatique  sous
```

Out[ ]=

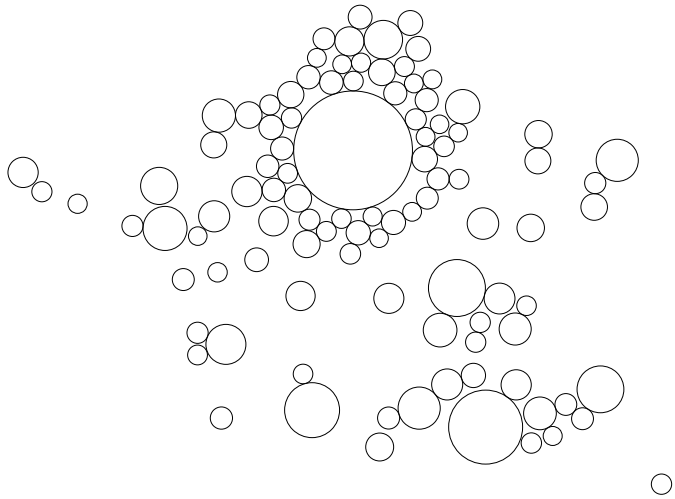


Outre ces fonctions, le lecteur peut utiliser d'autres fonctions pour cartographier des courbes de niveaux, des densités ou des vecteurs. Pour cela il retiendra les fonction **GeoContourPlot[], GeoDensityPlot[] et GeoVectorPlot[]**. Et de nouvelles fonctions sont disponibles dans les Resource functions pour élaborer des cartogrammes. Les cartogrammes tracent les surfaces des espaces considérés en fonction de l'intensité du phénomène à représenter.

Le cartogramme de Dorling des 100 plus grandes villes françaises

```
In[•]:= ResourceFunction["DorlingCartogram"][
          fonction ressource
          EntityValue[EntityClass["City", {"Country" → Entity["Country", "France"],
          valeur d'entité   classe d'entités                  entité
              "Population" → TakeLargest[100]}], "Entities"] → "Population"]
                             prends le plus grand
```

Out[•]=

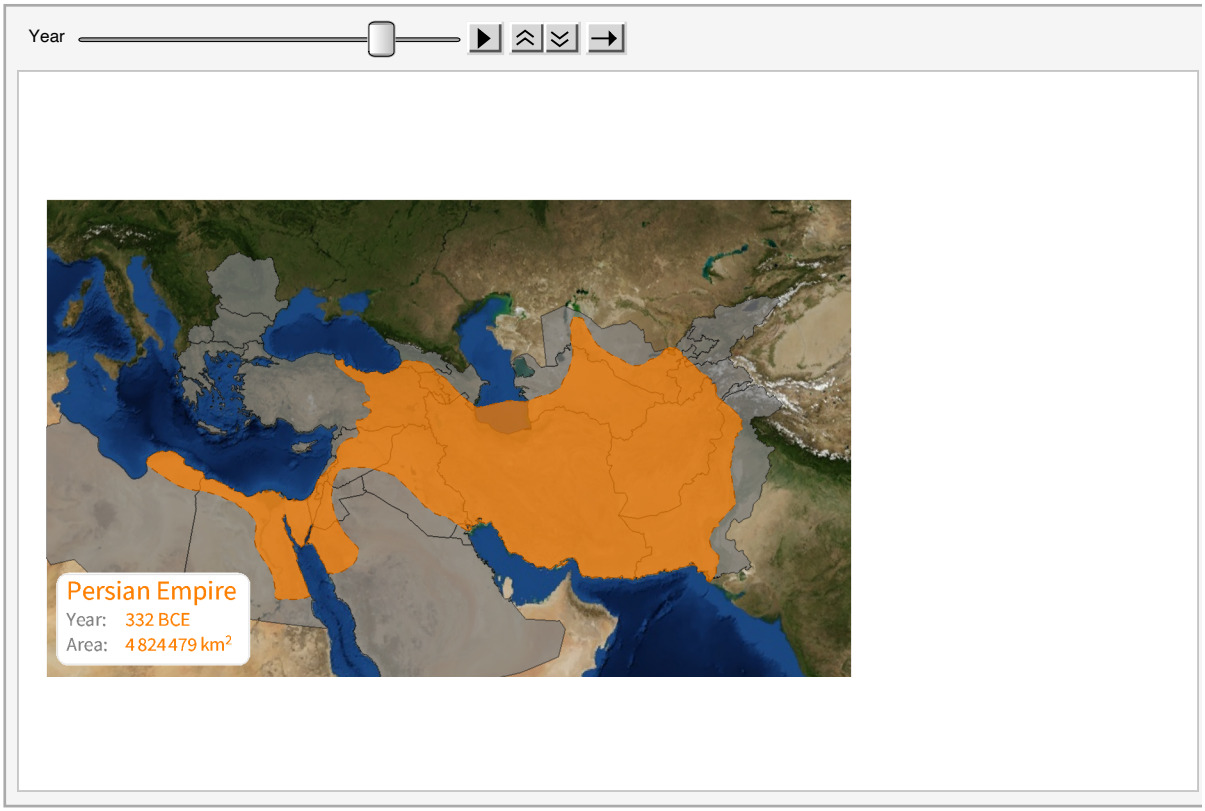


Conclusion

Bien qu'élémentaire, la présentation sommaire de ces fonctions doit montrer la richesse cartographique de Mathematica. Il convient d'ajouter qu'outre les options, les objets graphiques obtenus se combinent facilement avec les autres fonctions intégrées dans le logiciel. De plus, il est facile de construire des Atlas dynamiques ou historiques. Le programme ci-dessous montre l'évolution de l'empire perse. Pour stopper l'animation, donc stopper le déroulement historique, il faut cliquer sur l'icône qui représente les deux barres verticales.

In[153]:=
```
ResourceFunction["HistoricalCountryAnimate"][
 Entity["HistoricalCountry", "PersianEmpire"]]
```
fonction ressource
entité

Out[153]=



Et pour finir, un atlas dynamique de 5 cartes démographiques de l'Afrique. Cliquer sur le menu déroulant pour voir s'afficher la carte. Difficile de faire plus concis :

In[ ]:=
```
Manipulate[GeoRegionValuePlot[ Africa COUNTRIES → i],
 {i, {"Population", "AdultPopulation", "BirthRateFraction",
   "DeathRateFraction", "LifeExpectancy", "LiteracyRate"}}]
```
manipule — représentation géographique de régions associées à des valeurs

Out[ ]=