M@PPE MONDE

Numéro numéro 118 · Géovisualisation · Mise en ligne : 07/2016

Un exemple de géovisualisation de phénomène temporel avec la bibliothèque D3.js

par : Antoine Moriceau, Julie Laforêt et Corentin Rey

Mots-clés : Amérique du Sud, d3, géovisualisation, tourisme

⇒ Cliquer pour consulter et interagir avec la géovisualisation

Introduction

L'objectif de cet article est de présenter un exemple de géovisualisation, c'est-à-dire la visualisation de données géographiques. Par extension, cette technique désigne toutes les formes d'exploration et d'expérience visuelle rendues possibles par les outils géomatiques, par exemple en représentant de manière visuelle et accessible des aspects complexes du réel selon différents angles, ce qui permet leur comparaison et leur croisement.

Le projet Open-Source D3 (*Data-Driven Documents*) a connu un grand développement grâce à la participation de nombreux développeurs et testeurs (Dewar, 2012). Cette bibliothèque de fonctions en langage JavaScript est largement utilisée de par le monde, notamment dans la presse en ligne. Elle offre des possibilités très diverses et de portée très variable, comme le montre le site D3 Gallery (par Ch. Viau) qui contient de nombreux exemples d'application. D3 permet de contrôler la création de formes graphiques dynamiques et interactives avec un simple navigateur internet. Les applications sont actives « côté client » c'est-à-dire qu'elles s'exécutent dans le navigateur internet de l'utilisateur, à partir de données fournies directement ou en provenance d'un fichier ou d'un flux externe. Un programme développé en JavaScript va construire le type de représentation et d'interactivité dans la page web. Cette bibliothèque repose sur la manipulation d'objets vectoriels au format SVG (Scalable Vector Graphics) et nécessite la connaissance des langages HTML (*Hyper Text Markup Langage*) pour « l'écriture web » et de la norme CSS (*Cascading Style Sheet*) pour la gestion des styles (Murray, 2013).

Par le biais de ces outils relativement accessibles et gratuits, la représentation de données géographiques dynamique et interactive est ouverte à des utilisateurs familiers des langages de programmation web (HTML, CSS, bases de JavaScript). Cet article a pour ambition de toucher un public assez large en décrivant une application simple mais comprenant divers types de représentation géographique, tout en rendant possible une réappropriation du code JavaScript mis à disposition.

Dans cet objectif, notre réflexion nous a menés vers la représentation d'une dimension qu'il est difficile de représenter en cartographie statique : le temps. L'aspect temporel, potentiellement inhérent à toute analyse géographique, est une problématique constante de la représentation cartographique comme en témoigne l'article de Jean-Paul Cheylan paru dans *M@ppemonde* (n°87, 2007). Les techniques de géovisualisation offrent la dynamique et l'interactivité qui permettent de répondre au besoin de représentation de la variation temporelle d'un phénomène spatial.

Choix du sujet et du mode de représentation

La première étape de toute application de géovisualisation est l'analyse géographique de l'information

disponible et la définition des dimensions à représenter. Même si cette première définition du sujet sera probablement amenée à être modifiée à la marge, il est important de prendre le temps de mener cette première étape aussi loin que possible. En effet, le traitement des données et le codage de l'application sont des processus lourds et la refonte des objectifs plus tard au cours du projet constituerait une perte de temps.

Le thème abordé dans cet article est le tourisme en Amérique du Sud de 1996 à 2013. Deux variables statistiques complémentaires sont représentées : l'évolution du nombre de touristes et leur dépense moyenne annuelle, par pays. Ces deux variables sont représentées simultanément de deux manières différentes, selon deux modalités graphiques : une carte et un histogramme.

Sur la carte, l'évolution du nombre de touristes est une quantité relative et sera représentée en choroplèthe (aplat d'un dégradé de couleur). La dépense moyenne annuelle en dollars (quantité absolue) sera représentée sous la forme de cercles proportionnels. Ce mode de représentation est en accord avec les règles de sémiologie graphique (Bertin, 1967). Sur le graphique, l'évolution de la fréquentation touristique est représentée classiquement sous la forme de barres verticales annuelles formant histogramme. La dépense est visualisée sur une courbe linéaire superposée à l'histogramme.

L'utilisateur peut interagir avec l'application de deux manières. L'année courante est modifiable par un curseur temporel proposé à gauche de la page, qui modifie dynamiquement l'information représentée. On peut également cliquer sur les pays pour obtenir plus d'informations : le graphique statistique spécifique à ce pays, sur toute la plage temporelle, ainsi que des informations générales pour le pays sélectionné. De plus, un second graphique représentant l'évolution des deux variables sur le continent entier en fonction du temps est disponible dans la partie basse de la composition.

Après avoir présenté l'objectif général de l'application, décrivons plus en détail et plus concrètement la gestion des données et les différentes modalités de leur représentation.

Gestion des données

Si le rapatriement des données est réalisé au début du développement de l'application, leur mise en forme dépend du choix des dimensions et des modalités de leur représentation par l'application. Deux jeux de données sont nécessaires pour cette application : les données statistiques sur le tourisme et les données géographiques des pays d'Amérique du Sud.

Les données statistiques

Les données ont été acquises sur le site de diffusion de données libres (*Opendata*) de l'ONU. Celles-ci sont disponibles en téléchargement en CSV, un format texte lisible par de très nombreux logiciels (tableurs, SIG). Les métadonnées sont disponibles sur le site de l'Organisation Mondiale du Tourisme (UNTWO). Les informations générales sur les pays et les images de drapeaux sont issues de Wikipedia.

Les données utilisées par l'application ont été filtrées sur un tableur, par aire géographique et thématique. Les informations générales et les valeurs brutes des variables (non relativisées) ont été compilées. Ces dernières ont ensuite été mises en forme pour faciliter l'export au format JSON, une norme courante du web, utilisée par D3. Il est notamment important de transformer les virgules des nombres décimaux par des points. De plus les valeurs non renseignées doivent être laissées vides, lors de l'export elles seront affectées de la mention « null » qui est gérée comme cas particulier dans les scripts de l'application. La mise au format a été réalisée grâce à un convertisseur en ligne. Il est important de noter qu'un encodage a été réalisé sur le nom des pays en trois lettres, qui sera utilisé pour filtrer le jeu de données dynamiquement, comme on peut le voir dans le fichier DATA.json. Ce code constitue la valeur-clé de repérage des pays dans les données.

Les données spatiales

Les données cartographiques des pays d'Amérique du Sud ont été acquises au format dit « fichiers de forme » (SHP) sur le site *Natural Earth Data*, selon une licence d'utilisation libre. Ces données ont ensuite été filtrées pour ne conserver que les pays de la zone d'étude grâce au logiciel libre QGIS. Le logiciel QGIS permet d'enregistrer directement ces fichiers dans le format GeoJSON, directement exploitable en JavaScript avec la bibliothèque D3.

La même valeur-clé que pour les données statistiques est renseignée dans un champ de l'objet GeoJSON pour identifier les pays. Ceci permettra d'accéder à ces champs selon la date courante de travail de l'application. Les données sont situées dans le fichier Pays_donnes.geojson. Dans les cas où il n'y a pas de données, lors de la jointure, le logiciel de SIG conserve la valeur 0 dans les champs correspondants. Cela ne pose pas de problème pour les valeurs de dépense (un cercle de rayon nul n'apparaît pas). Cependant, il est nécessaire de changer manuellement la valeur d'un champ nul pour l'évolution avec la valeur « null » pour gérer ce cas lors de l'affichage de la carte choroplèthe. Un autre fichier Pays_hors_zone.geojson est créé, il contient les pays hors de la zone d'étude (Guyane française et Amérique centrale), qui vont constituer l'habillage de la carte.

Organisation des scripts de l'application

L'application est contenue dans une seule page HTML pour une incorporation plus facile dans une page web. Celle-ci appelle plusieurs fichiers : des bibliothèques de fonctions, des scripts JavaScript, un fichier de styles CSS et des fichiers de données JSON et GeoJSON comme le présente le schéma de la **figure 1**.

Figure 1. Organisation des scripts de l'application.

Quatre variables globales sont déclarées dans le fichier HTML, leur valeurs seront modifiées par l'interaction des utilisateurs : la date courante, le pays sélectionné, les données statistiques et les classes de la discrétisation (le découpage en classes utilisé dans choroplèthe, les histogrammes et la légende). Les variables globales sont utilisées par les différentes fonctions déclarées dans les scripts pour modifier interactivement l'affichage des informations.

Les grandes étapes de construction des différents éléments affichés sont présentées ci-après mais, pour une réelle appropriation du fonctionnement, il est possible de se laisser guider par les commentaires qui jalonnent les scripts eux-mêmes.

La carte

Les instructions pour l'affichage d'une carte complète (choroplèthe et symboles proportionnels) ainsi que ses éléments indissociables (légende, échelle) sont contenus dans le script Carte.js. Celui-ci déclare la fonction « map » qui est déclenchée avec la date en paramètre lors d'un évènement de clic sur la barre de temps ou les histogrammes. C'est le script Date.js qui va initialiser l'application, notamment en remplissant les variables globales, dont la plus importante contient les données statistiques.

Tout d'abord, le script va rechercher ces données dans le fichier DATA.json, puis il les filtre sur la date courante. Le script Carte.js prend ensuite le relais. Il lui faut définir les couleurs qui seront utilisées ainsi que les bornes des classes correspondantes. Une nouvelle variable « couleurs » est donc créée avec un tableau des couleurs HTML choisies à l'avance. De la même manière la variable tableau globale « classes » est initialisée

avec les bornes des classes choisies. Ces variables seront utilisées à la fois dans la carte choroplèthe et dans la légende associée. Avant d'entamer la création de la carte à proprement parler, un objet SVG dans lequel les données seront stockées est déclaré.

La carte choroplèthe

La création de la carte choroplèthe constitue la première étape du travail cartographique et représente la base visuelle de l'application. Les informations cartographiques, de fond de carte, sont lues dans le fichier Pays_donnees.geojson. Ces données sont utilisées afin de créer, au travers des objets SVG et des données statistiques définis plus haut, une carte choroplèthe.

De nouvelles variables sont créées pour remplir l'objet SVG et lui donner des attributs et caractéristiques graphiques. Sept classes ont été déterminées à l'avance selon leur répartition sur un scalogramme. Les couleurs d'affichage sont gérées grâce à une boucle conditionnelle sur ces classes. Ces paramètres sont stockés dans des variables globales pour pouvoir être réutilisés dans le dessin de la légende et des histogrammes.

Afin de donner un style particulier aux entités « pays », il est nécessaire d'ajouter dans le fichier style.css la classe correspondante, avec un contour et une épaisseur spécifiques.

Les pays hors de la zone d'étude

Comme précédemment, la fonction d3.json() est appelée avec pour argument cette fois-ci, le fichier GeoJSON contenant les données géographiques des pays non traités : Pays_hors_zone.geojson. Ceux-ci sont situés à proximité de la zone d'étude et permettent de resituer le contexte de celle-ci.

Deux nouvelles variables sont créées pour remplir l'objet SVG et lui donner son aspect visuel. Afin de donner un style particulier aux entités non traitées, il est nécessaire d'ajouter dans le fichier style.css la classe correspondante (pays_hors_zone) avec un contour et une épaisseur spécifiques.

La carte en symboles proportionnels

Une fonction javascript issue de D3 dessine les cercles en SVG sur les points *centroïdes* des pays. La surface de ces cercles est fonction de la valeur des dépenses moyennes par touriste à la date de sélection. Le dimensionnement de ces cercles est classiquement défini par une règle de proportionnalité à la valeur du symbole le plus grand.

Les contours et l'affichage des infobulles au survol de la carte

Cette carte a pour vocation d'être dynamique et interactive pour l'utilisateur. Dans cette optique, il semblait important d'ajouter des infobulles au survol des entités « pays » avec certaines informations telles que les valeurs exactes d'évolution du nombre de touristes et de dépenses. Ces infobulles peuvent être créées grâce à la bibliothèque JavaScript complémentaire d3-tip.

Une nouvelle variable est ainsi créée pour mettre en place ces infobulles avec le contenu voulu grâce aux champs du fichier Pays_donnees.geojson. Afin de pouvoir distinguer le pays sélectionné sur la carte, il est nécessaire d'ajouter dans le fichier style.css la rubrique correspondante (pays:hover) avec un contours noir plus épais que s'il n'était pas survolé par le curseur de la souris. De même, pour les pays hors zone, un style est également créé (pays_hors_zone:hover) afin que le survol de l'entité soit possible. Les infobulles comportent dans ce cas le nom du pays et la mention « n/a » pour les données statistiques.

En pratique, malheureusement, lorsque la souris survole une entité, le changement d'épaisseur des contours n'est pas égal sur tout le pourtour du pays car il est dessiné dans l'ordre de définition des pays dans le fichier des données cartographiques (une frontière épaisse peut donc être partiellement masquée par un pays dessiné ultérieurement). Pour y remédier, il est nécessaire de créer un masque transparent disposé au-dessus des pays, qui est visible uniquement au survol et qui permettra d'avoir une belle frontière d'épaisseur égale sur l'intégralité du contour.

La variable infobulle est donc créée par cette nouvelle fonction. Les fonctions « mouseover » et « mouseout », qui gèrent les évènements dus au mouvement de la souris, sont appelées sur le nouvel objet masque. Une fonction de réaction à l'évènement « click » est aussi créée afin que, lors de la sélection d'un pays, les informations générales et l'histogramme soient mis à jour (cf. parties suivantes). Des instructions conditionnelles permettent de gérer l'affichage pour les cas d'absence de données. La variable infobulle des pays hors zone est intégrée de la même manière sur l'objet correspondant. La variable globale pays_courant est affectée de l'identifiant du pays cliqué. Le clic sur un pays signifie qu'un pays précédemment sélectionné doit revenir à son affichage normal, et la page doit être mise à jour. L'instruction remove() masque l'élément SVG concerné. Ensuite, l'histogramme est redessiné et les informations générales affichées sont mises à jour à partir du nouveau pays sélectionné.

Les légendes

La légende des données de dépenses moyennes annuelles prend la forme de cercles proportionnels pour trois valeurs représentatives des données. Des objets cercles sont créés, leurs rayons sont proportionnels à la valeur de la donnée (de la même manière que sur la carte). Une étiquette est placée sur les cercles. Ensuite, les objets cercles sont mis en place sur la page, ainsi que les textes correspondant en légende.

La légende de l'évolution du nombre de touristes est construite grâce à une boucle « for » sur la base des variables tableaux des couleurs et des bornes de classes choisies en début du script Date.js. Elle est constituée de rectangles de largeur égale à l'étendue de classe et d'un axe présentant les bornes de ces classes. Le cas de l'absence de données est figuré par un autre rectangle, gris, portant la mention « n/a ».

Enfin, l'échelle graphique est créée grâce à l'utilisation de fonctions présentes dans la bibliothèque d3.js. Une variable *x* est déclarée par la fonction d3.scale.linear(). Sa longueur sur le navigateur est gérée par la fonction .range(). Cette longueur (en pixels) est obtenue par une mesure sur la carte sur une ligne horizontale dont la distance est connue pour avoir une échelle correcte. Une autre fonction, .domain(), permet d'associer la valeur à représenter (ici des km). Pour finir, au travers d'un .append, nous affectons cette barre à l'objet SVG de la légende.

Le graphique

Le script Histo_pays.js déclare la fonction drawHisto() qui utilise les données statistiques globales. Cette fonction est déclenchée lors de l'évènement d'un clic sur un pays (cf. plus haut). Le graphique déjà existant (représentant les données du pays sélectionné précédemment) est supprimé et la fonction drawHisto() est relancée. Lors du changement de la date (dans le script Date.js, cf. plus bas), la fonction est également relancée afin de faire figurer lisiblement la date courante sur le graphique, pour une lecture des informations plus claire. Une variante de celui-ci, Histo_continent.js, permet d'afficher le graphique des données du continent entier. Les données sont filtrées sur l'Amérique du Sud et la modification de ce graphique n'est déclenchée que par le changement de date.

La première étape de la fonction de dessin de l'histogramme est de filtrer le jeu de données sur la variable « pays_courant » c'est à dire le pays qui a été sélectionné par l'utilisateur, par le biais de l'instruction

date.filter(). Ensuite plusieurs variables sont assignées : les valeurs de marges et de dimension de l'élément d'une part et des valeurs issues des données d'autre part (tels que les *maxima* de valeurs). Cette étape permet de fixer ces constantes pour la suite du script.

L'élément SVG est déclaré, tous les éléments du script seront ajoutés à celui-ci par l'instruction append(). L'accès aux données se fait par les instructions de la bibliothèque d3.data(données).enter(). Chaque fonction possède des attributs (coordonnées, dimension, couleur) qui seront affectés au cas par cas. Une attention particulière doit être portée aux situations variables (telles que des valeurs positives ou négatives ou l'absence de données) par le biais d'instructions conditionnelles (if – else).

Trois axes sont créés : l'axe des abscisses présente les dates de la série temporelle, le premier axe des ordonnées correspond aux valeurs d'évolution du nombre de touristes et le second aux valeurs de dépense moyenne. La création des axes se fait grâce à l'objet d3.scale.linear() en fixant le domaine de valeur et la portée de l'affichage. Ces axes sont ensuite ajoutés à l'élément SVG pour l'affichage.

Les histogrammes sont formés d'une série d'objets rectangulaires dont les propriétés (largeur, hauteur, abscisse et ordonnée) dépendent des constantes déclarées et des valeurs du jeu de données. Par souci d'harmonie, on utilise les mêmes couleurs pour les barres des histogrammes que pour les pays sur la carte choroplèthe. Les valeurs à la date courante sont affichées sur les barres correspondantes.

La courbe est un objet d3.svg.line() constitué par une interpolation linéaire des points du jeu de données. Les coordonnées de ces points sont fixées en fonction des valeurs à représenter. Les données sont filtrées auparavant pour ne conserver que les valeurs non nulles afin que celles-ci n'apparaissent pas sous la forme de points de valeur 0 dollar. Des points sont placés sur la courbe afin de figurer les enregistrements. De la même manière que pour les histogrammes, la valeur de la dépense pour la date en cours est placée sous les points (en gérant là aussi les cas d'absence de donnée).

Pour ajouter de l'interactivité au graphique, des informations contextuelles sont placées au survol de l'histogramme par la souris. Les valeurs de l'évolution, de la dépense moyenne ainsi que la date sont stockées dans des variables de type d3-tip() issues de la bibliothèque d3-tip.js. Ces variables sont appelées par le biais de l'instruction svg.call(). Afin de gérer l'apparition de plusieurs informations simultanément, des fonctions regroupant les instruction show() et hide() de la bibliothèque sont créées. Enfin des rectangles transparents qui occupent toute la hauteur du graphique sont créés. Les instructions on('mouseover') et on('mouseout') sont placées sur ces rectangles pour lancer les fonctions affichant les informations apparaissant au survol de la souris, dites contextuelles. La couleur de survol est gérée par le fichier style.css. Enfin, le clic sur ces rectangles déclenche aussi le changement d'année de référence et la mise à jour de l'application entière.

La superposition d'informations dans une application de géovisualisation est un problème inhérent et récurent de ce type de technique. En effet, la position des objets et des informations dépend des données traitées et ce problème est assez difficile à gérer de manière globale.

Informations générales

La fiche d'information sur les pays est générée par le biais de deux fonctions implémentées dans le script Info.js. La fonction infoPaysInit() affiche une invitation à sélectionner un pays. Celle-ci est appelée dans la balise correspondante du fichier index.html (la division de la grille de mise en page). La fonction infoPays(), quant à elle, récupère les informations du jeu de données et les affiche. Elle est déclenchée lors du clic sur un pays (de la même manière que la création du graphique vue précédemment).

La première étape consiste à filtrer et à stocker les données sur le pays sélectionné et la date courante. Les informations contenues dans le tableau JSON sont affectées à des variables. Les images des drapeaux sont

stockées sur la machine serveur sous la forme de fichiers image et donnent lieu à une affectation particulière. En effet, le nom de ces fichiers dépend du pays sélectionné et la variable est construite sous la forme d'une balise HTML <img> qui contient le chemin d'accès à l'image et les dimensions d'affichage souhaitées.

Ensuite, une variable texte contient une instruction HTML pour placer et habiller les variables brutes précédentes. La division ( ) adéquate du fichier HTML est récupérée grâce à une instruction document.getElementById(). Le texte est affiché dans cette division grâce à la propriété innerHTML() de l'objet div. Cette instruction permet de modifier dynamiquement le contenu d'une balise HTML par une fonction JavaScript.

Création d'une barre de temps, synchronisation avec la carte et le graphique

Lors des parties précédentes nous nous sommes intéressés aux différentes représentations graphiques de notre application et à leur construction. Il s'agit maintenant d'axer notre explication autour de la liaison entre ces représentations et la mise en place d'une dimension temporelle. Pour cela, une simple barre de curseur verticale a été créée en utilisant les possibilités du langage HTML, dans le fichier index.html (élément). L'interaction avec cette barre se réalise grâce à la bibliothèque dragit (js) qui étend les fonctions de base du HTML. Le script Date.js contient les fonctions qui créent la connexion entre l'objet HTML et le reste de l'application.

La première étape est donc de créer cette barre illustrant notre série temporelle. Nous avons choisi une représentation simple, codée directement en HTML dans notre fichier index.html. En effet, la balise de type *range* génère directement une barre avec un curseur, lisible par tous les navigateurs internet.

Les attributs suivant la déclaration du type *range* correspondent aux caractéristiques de la barre temporelle. Notre série allant de 1996 à 2013, nous avons donc 17 valeurs à retranscrire. L'attribut *step* correspond à 1 car il est nécessaire que notre série s'arrête sur chaque date.

La fonction init() déclenche l'affichage au chargement de la page HTML. Elle affecte donc à la variable globale dateCourante la date initiale. La fonction qui génère la carte est déclenchée sur cette date et le graphique sur le continent entier également. Ensuite, la bibliothèque dragit est initialisée sur la division du fichier HTML contenant la barre de temps. Une instruction déclenche la fonction update() lors de l'évènement d'un clic avec le paramètre correspondant à l'endroit du clic (c'est-à-dire à la date sélectionnée sur la barre).

Cette fonction update() appelle la fonction displayYear() sur la valeur enregistrée lors du clic. C'est dans cette dernière fonction que sont placées les instructions qui mettront à jour l'affichage. La date sélectionnée est affectée à la variable globale dateCourante. Ainsi le graphique par pays, le graphique sur le continent entier, la carte et les informations sont mises à jour. Cela se produit toujours de la même manière : les éléments SVG identifiés par un code spécifique sont supprimés par l'instruction remove(), puis les fonctions sont déclenchées avec la nouvelle valeur de la date. Des instructions conditionnelles permettent de ne pas modifier l'affichage des informations et du graphique si aucun pays n'est sélectionné.

Conclusion

La géovisualisation avec D3 permet de mettre en valeur des données géospatiales grâce à de nombreuses fonctions et de larges possibilités de représentation. Le dynamisme est le grand principe de cette bibliothèque qui allie ainsi utilité et ergonomie de l'application créée.

Dans cette application, un exemple type a été choisi, en étant conscients de l'importance de pouvoir traiter de

nombreux cas de figures pour permettre aux futurs utilisateurs de se réapproprier le code avec leurs propres données (choroplèthe, cercles proportionnels, légendes, échelle, diagramme, infobulles, gestion du temps, de données nulles ou manquantes, ...). Ainsi il est possible assez simplement de réutiliser l'application sur une emprise géographique différente et avec d'autres données. Pour cela, il suffit de modifier dans les scripts les appels aux fichiers de données et de paramétrer quelques éléments (centrage et échelle de la projection, couleurs, classes, titres, dimensions) accessibles à chaque fois en début de script ou dans le fichier CSS. En enlevant des parties du code, il est également simple de ne faire apparaître que certains éléments (seulement une choroplèthe, pas de graphique sur le continent, des informations générales simplifiées...).

Cet exemple présente toutefois des limites. Suivant la vitesse de la connexion internet, l'affichage de nouveaux objets peut prendre quelques fractions de seconde. Il serait possible de contourner ce problème grâce au système de transitions géré par d3, qui adoucirait les métamorphoses graphiques de l'application. L'ergonomie de la barre de temps peut également être améliorée.

D'autre part, au début du projet, nous voulions faire apparaitre les différents sites touristiques classés par l'UNESCO en fonction des dates, comme un des phénomènes potentiellement explicatifs de l'évolution du tourisme en Amérique du sud. Il aurait également été possible d'ajouter d'autres phénomènes explicatifs tels que des crises économiques ou des ruptures politiques majeures dans certains pays sur la période d'étude, sur une frise chronologique parallèle aux histogrammes et courbes. Cependant, faute de temps et pour éviter la surcharge d'informations, ces pistes ont été laissées de côté.

Enfin, sur cette application, il manque l'information sur la provenance des touristes. Cet élément apporterait une lecture plus riche de la problématique du tourisme en introduisant la question des flux. Leur représentation sous D3 est possible quoique plus complexe à mettre en place, comme en témoigne cet article paru dans *M@ppemonde* (n°89, 2008). Une des problématiques principales réside alors dans la difficulté de se procurer des données exploitables, mais quelques exemples existent (Aéroport, Réfugiés, cf. la D3 gallery).

Bibliographie

BERTIN J. (1967). « Sémiologie graphique ». Mouton/Gauthier-Villars, 431 p. ISBN 978-2713212772

CHEYLAN J.-P. (2007). « Les processus spatio-temporels : quelques notions et concepts préalables à leur représentation ». *Mappemonde*, n°87

DEWAR M. (2012). *Getting Started with D3*. O'Reilly, 70 p. ISBN 978-1449328795

KADDOURI L. (2008). « Réflexion sur la sémiologie graphique animée des flux ». *Mappemonde*, n°89

MURRAY S . (2013). *Interactive Data Visualization for the Web*. O'Reilly, 274 p. ISBN 978-1449339739

SILL□RE G., ROBERT S. (2007). « Les cartes animées ». *Mappemonde*, n°86

Webographie

Manuel sur le SVG : http://www.w3.org/TR/SVG/expanded-toc.html

Tutoriel complet sur D3 : http://animateddata.co.uk/articles/

Tutoriel sur la représentation graphique statistique avec D3 : https://ljegou.github.io/d3sigma/

Tutoriel pour apprendre à créer des animations avec D3 :

http://blog.visual.ly/creating-animations-and-transitions-with-d3-js/

Tutoriel pour apprendre à faire une carte choroplèthe grâce à D3 : http://blog.m0le.net/2014/11/26/autopsie-dune-dataviz-8-1-un-geojson-complet-et-leger/

Tutoriel pour apprendre les bases de D3 : http://victoralexandre.fr/demo_d3.html

Tutoriel pour la création d'histogrammes : https://github.com/mbostock/d3/wiki/Histogram-Layout

Tutoriel pour la création d'un graphe en courbe : http://bl.ocks.org/mbostock/3883245

Tutoriel sur la fonction *tooltip* : http://bl.ocks.org/Caged/6476579

Tutoriels sur les cartes en cercles proportionnels : http://bost.ocks.org/mike/bubble-map/