CODE:ME

UCZYMY PROGRAMOWANIA

{256}

TYDZIEŃ PROGRAMISTY

Szymon Moliński
CODE:ME

Wrzesień 2022

CODE:ME
UCZYMY PROGRAMOWANIA

{256}
TYDZIEŃ PROGRAMISTY

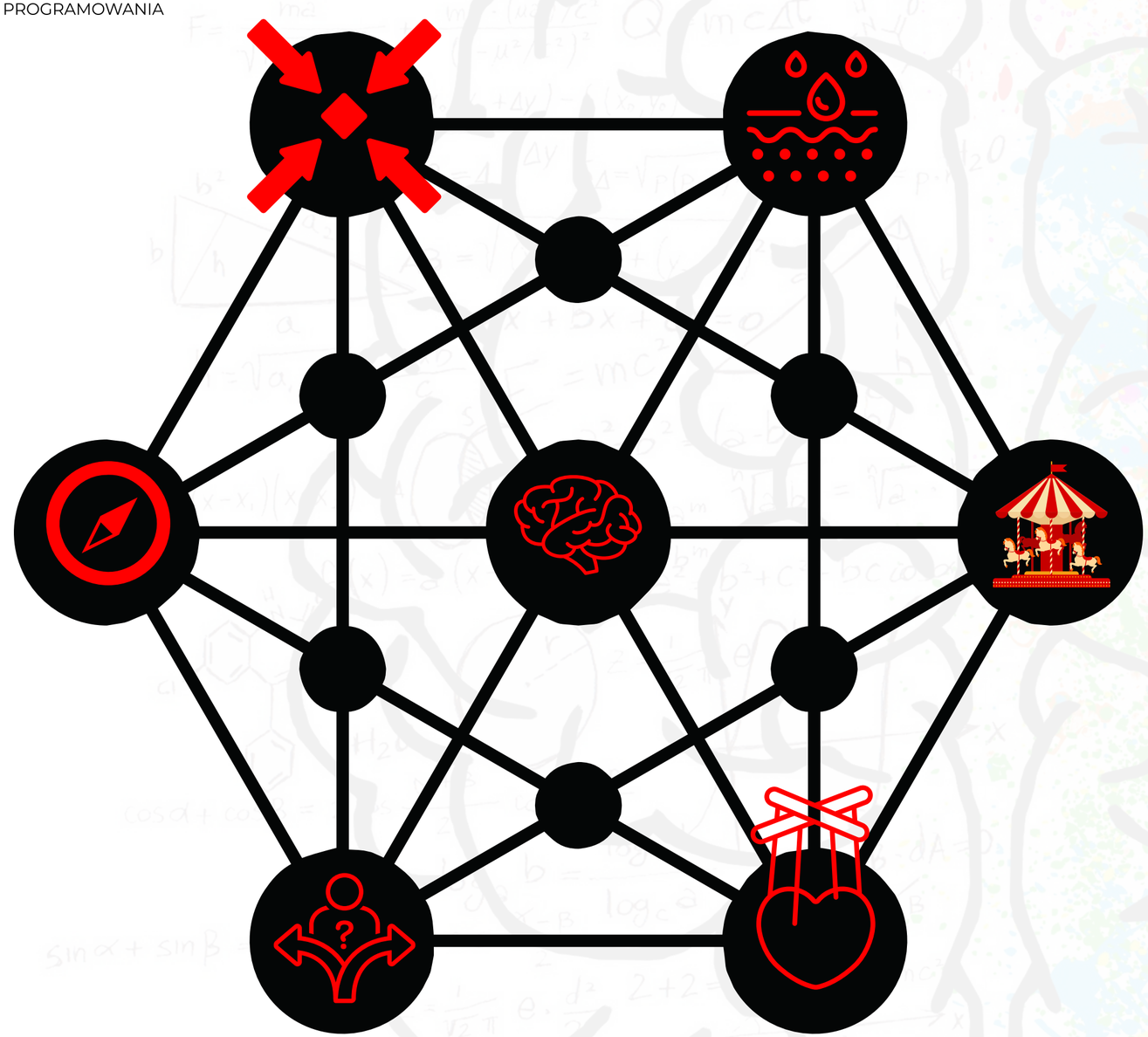
Wizualizacja danych

OD RAPORTU DO USŁUGI

Szymon Moliński
CODE:ME

Wrzesień 11111100110

Spis Treści



Cele wizualizacji danych

Funkcjonalność

Jaki jest cel wizualizacji?

Eksploracja danych

Znaleźć to, co ukryte

Decyzje

Data-driven Organization

Edukacja

Wzmacnianie procesu nauki

Perswazja

Próba wpłynięcia na decyzje

Sztuka

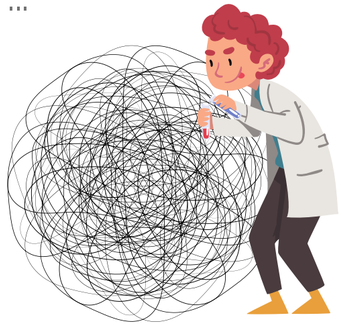
Piękno w danych?

CODE:ME
UCZYMY PROGRAMOWANIA

Adresaci

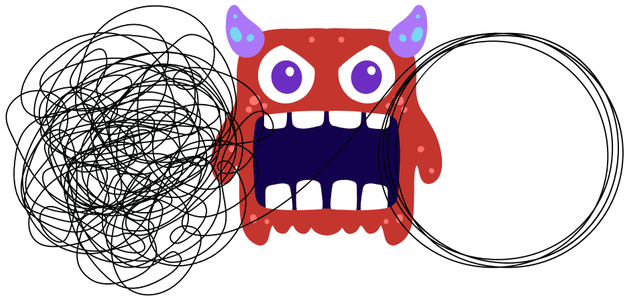
Ja

...



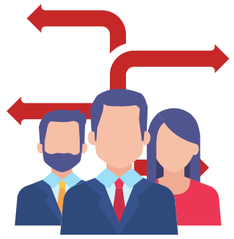
Eksperci

...



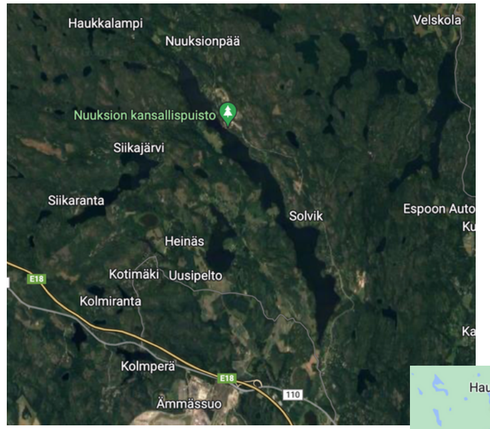
Decydenci

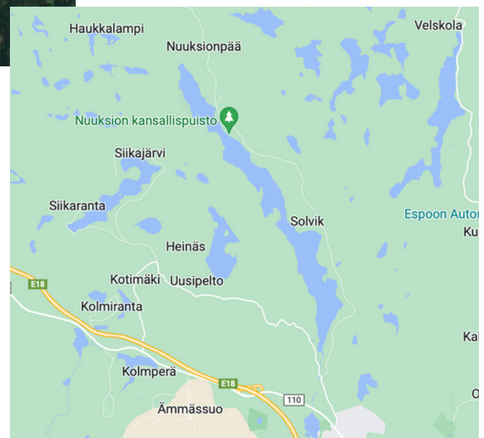
...



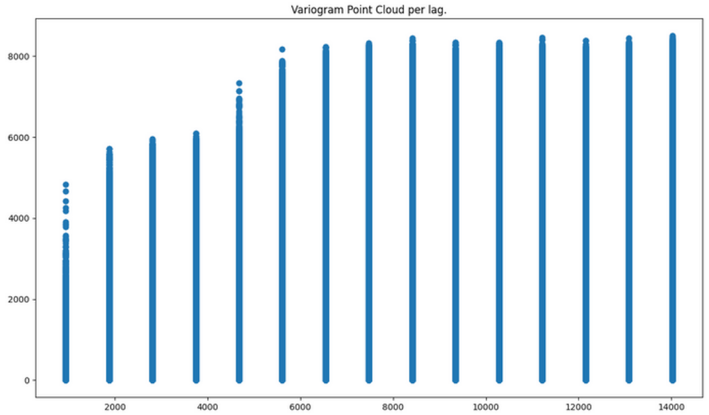
Społeczeństwo

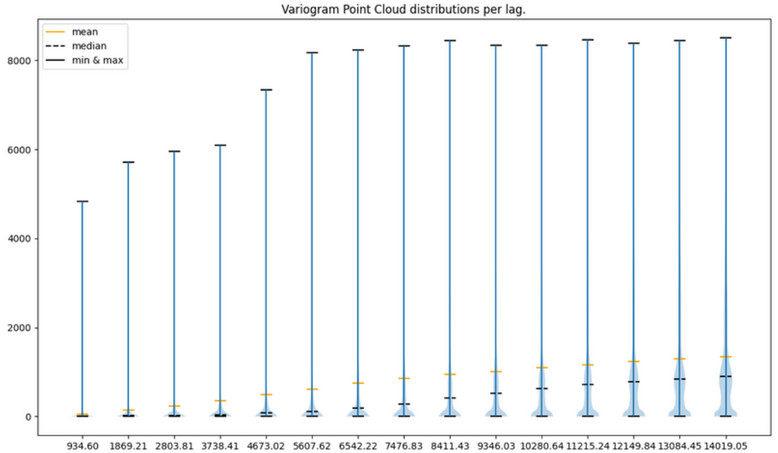
...

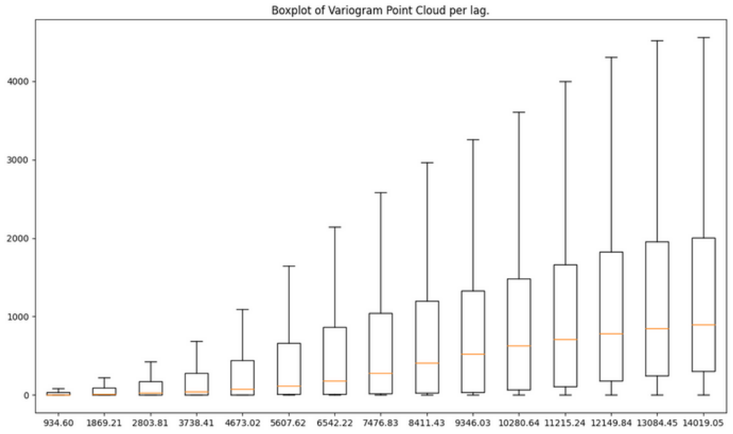




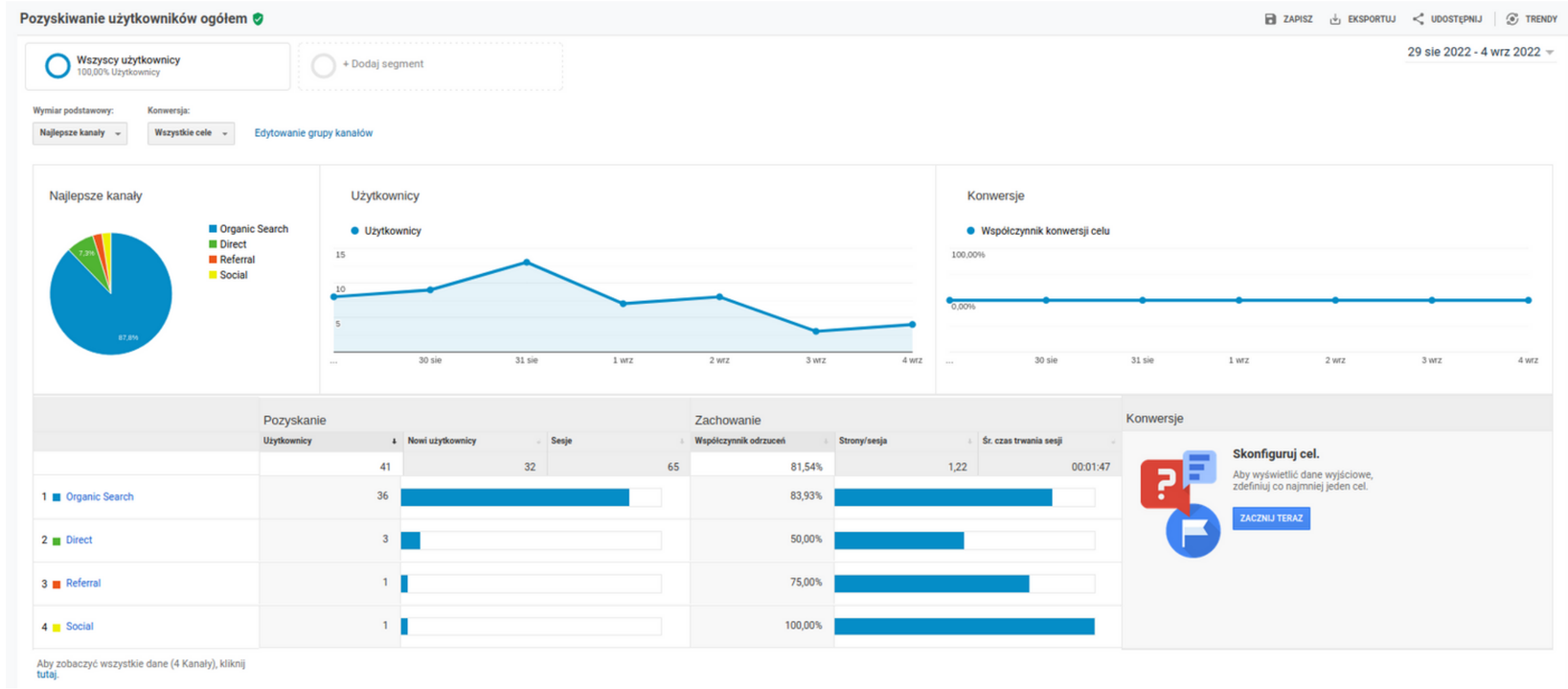
Eksploracja danych







Raporty biznesowe



Aplikacje (1/2)

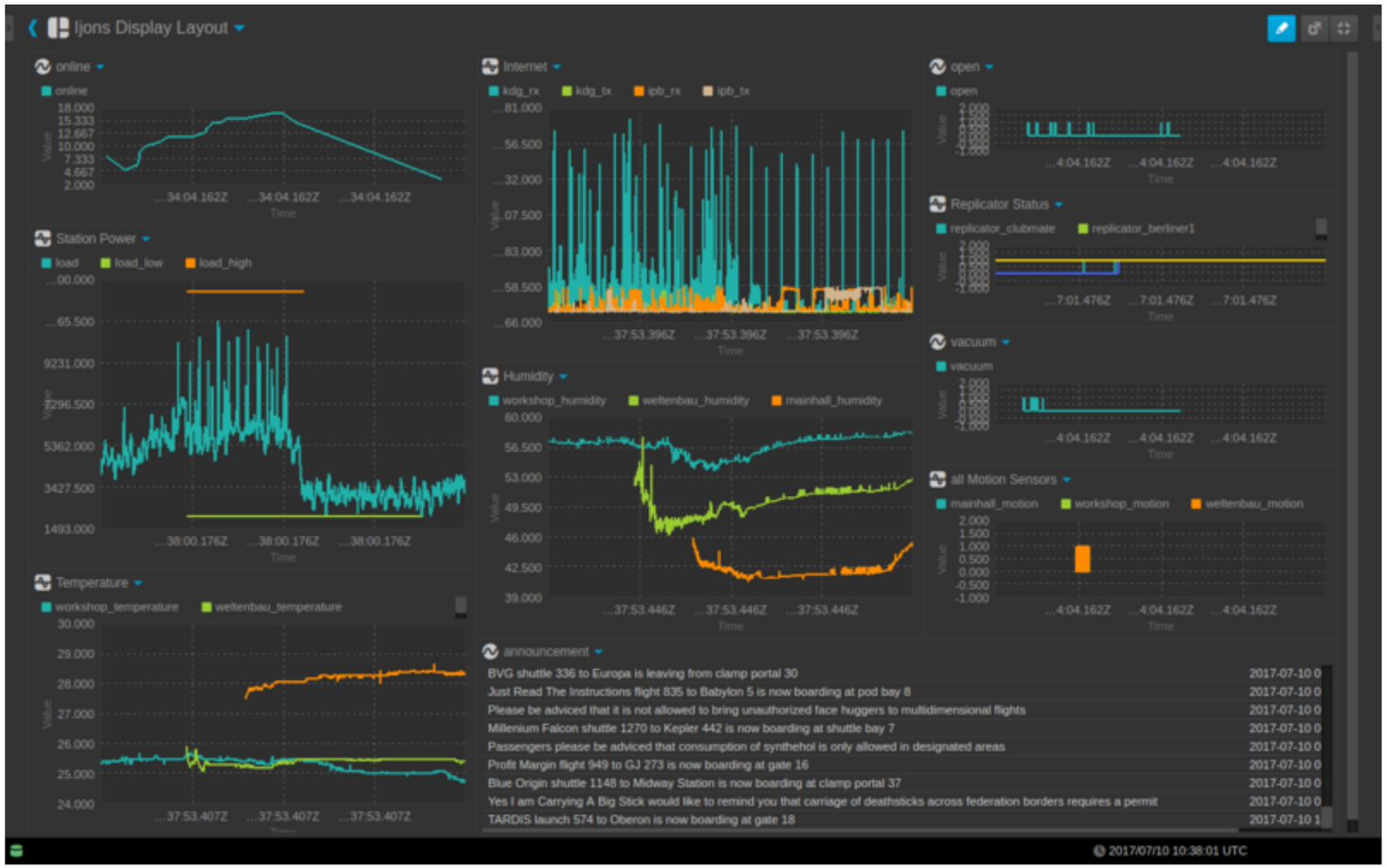
React

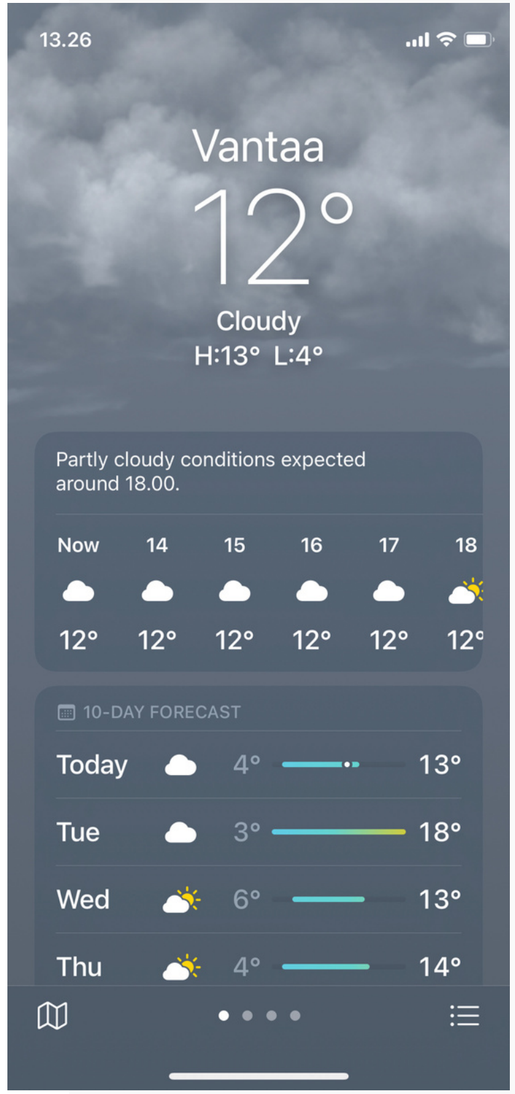
JS

Java

https://coronavirus.jhu.edu/map.html

Aplikacje (2/2)

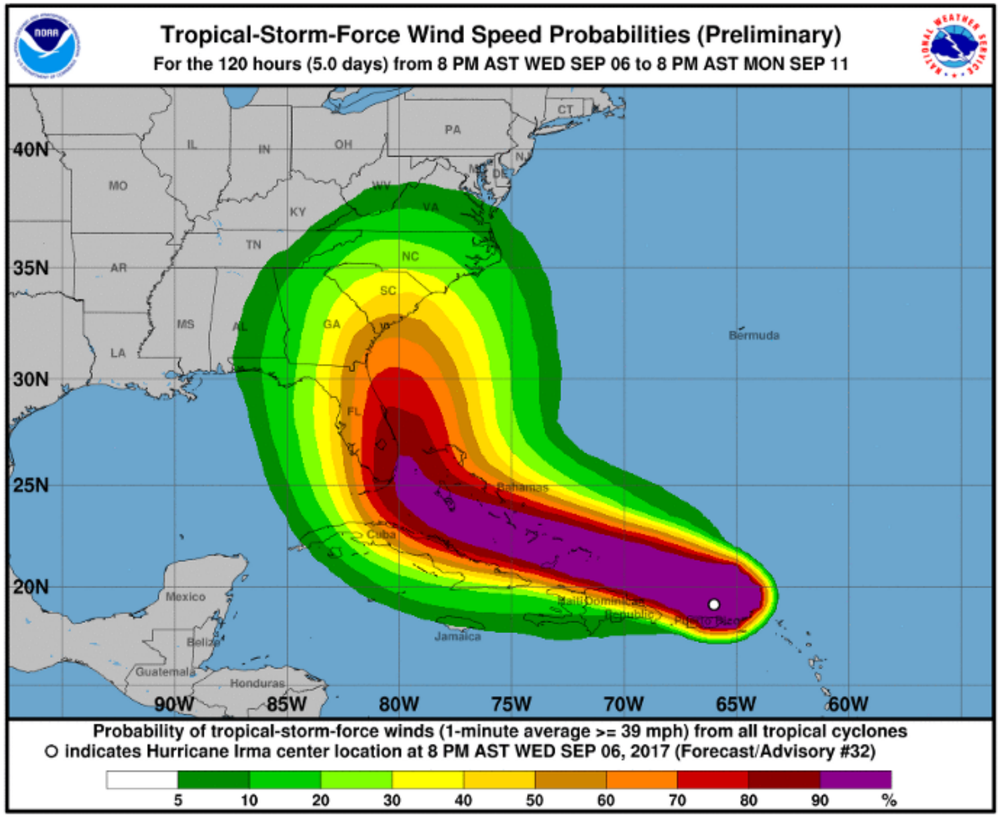


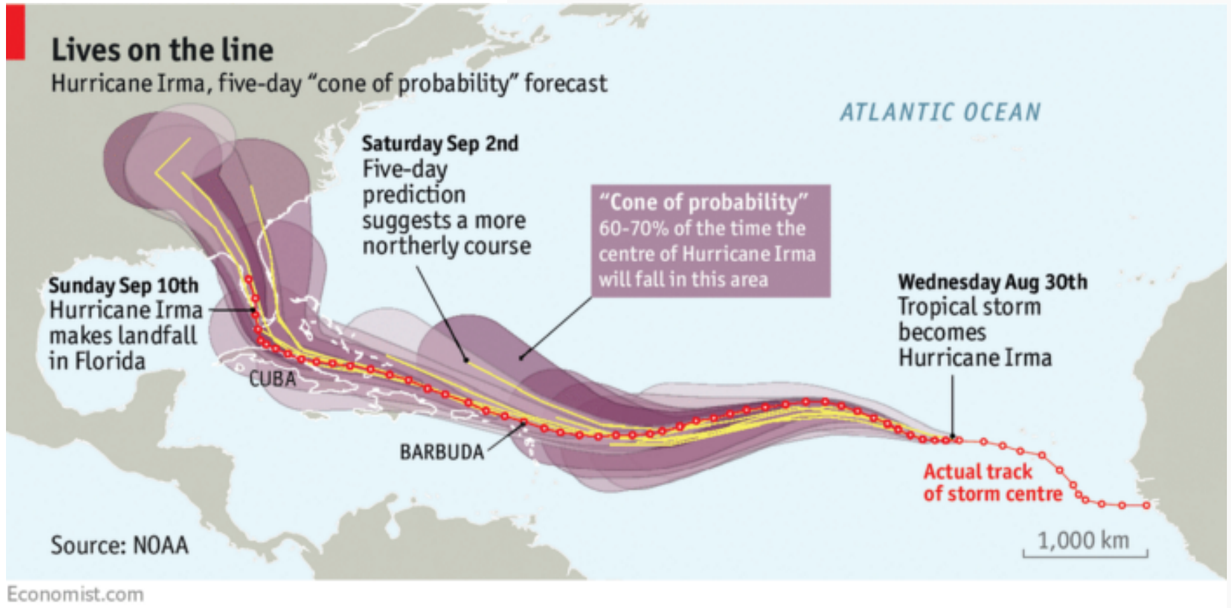


https://bergie.iki.fi/blog/nasa-openmct-iot-dashboard/

Czasopisma: The Economist

The Economist





https://medium.economist.com/the-science-of-forecasting-is-getting-better-probably-883abc1cd155

Dodatkowe materiały:
https://medium.economist.com/tagged/data-visualization

e World of Flowers

LAZONED WITH BEAUTY, this floral map ows the origins of 117 of man's favorite s. As people began to move from one part of orld to another, they carried plants with them. ers, conquerors, and adventurers returned to omelands with flowers from far-off places. sts carried seeds and bulbs to the New World. have done so well in their adopted regions that eginnings are seldom remembered. Holland's a native of Turkey; the "French" marigold d in Europe with the return of the conquista- om Mexico. To trace these blossoms to their , GEOGRAPHIC artist Ned Seidler consulted ldred E. Mathias, Professor of Botany at the sity of California at Los Angeles.

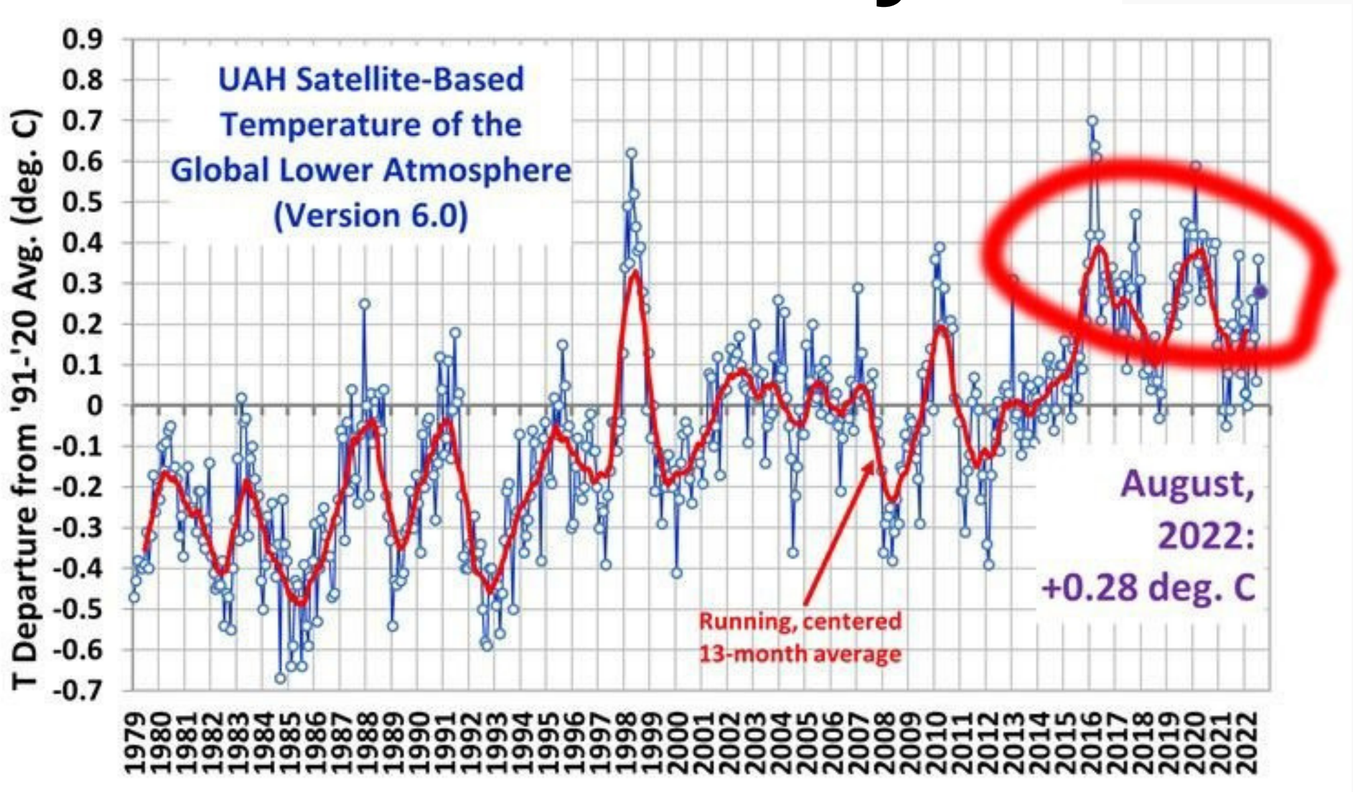
NATIONAL GEOGRAPHIC

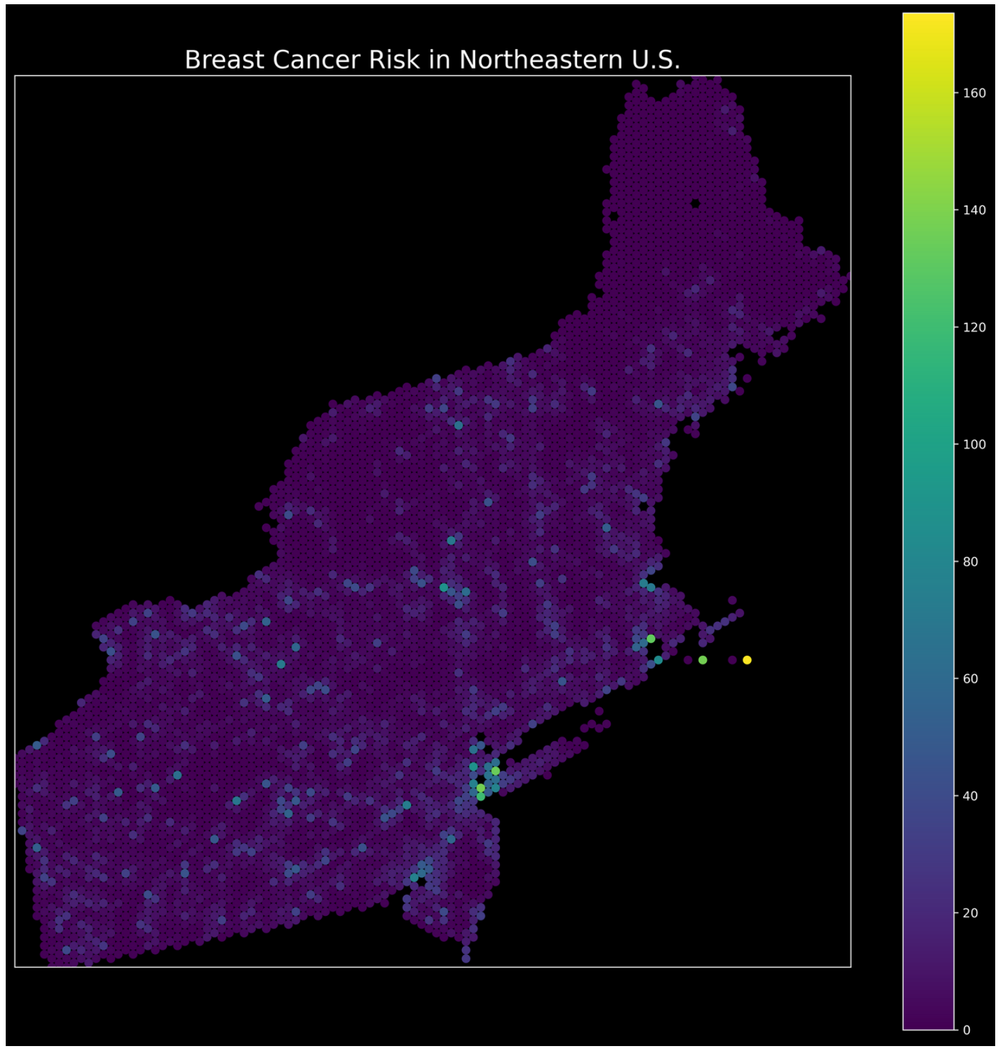
Czasopisma: Interaktywne źródła

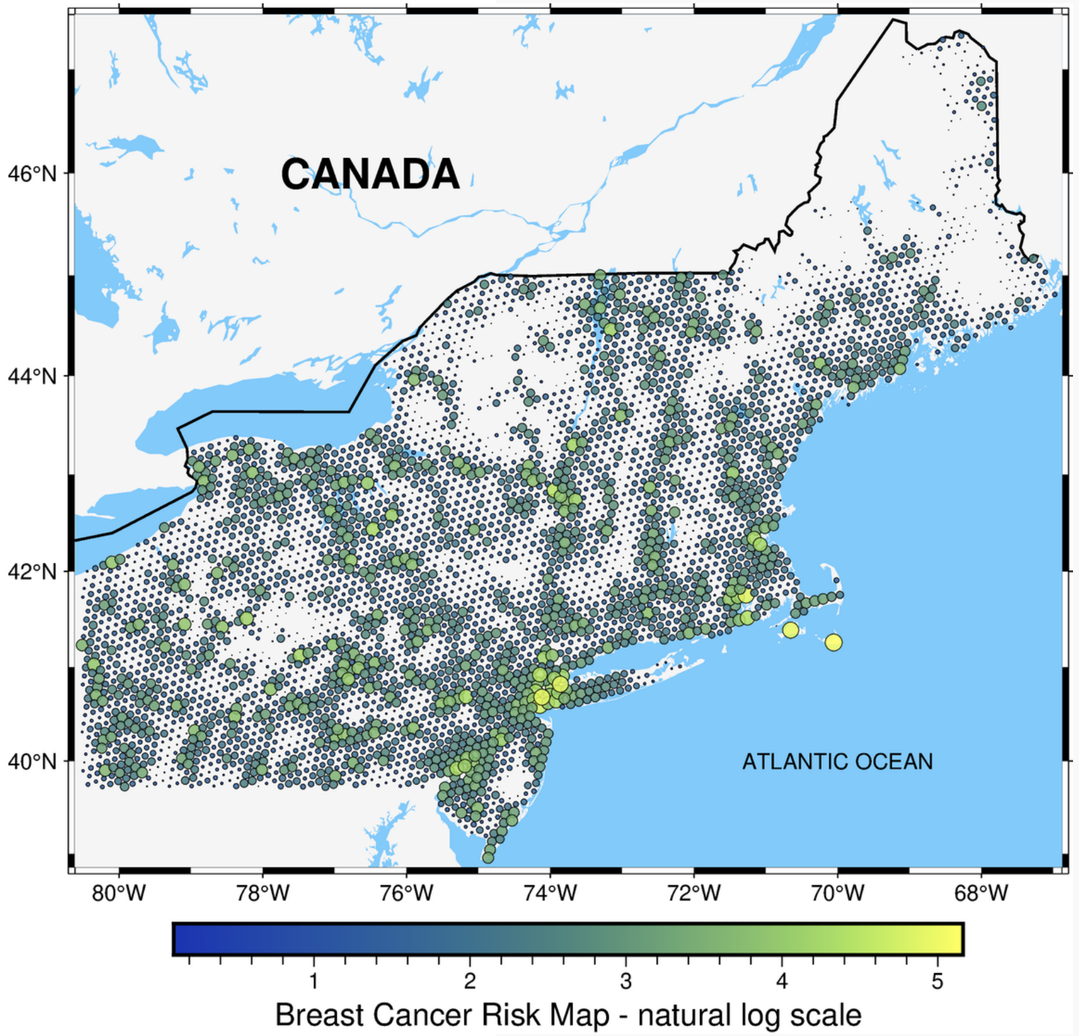
https://govdna.sudox.nl

Publikacje naukowe

https://vciba.springeropen.com/articles/10.1186/s42492-021-00092-y



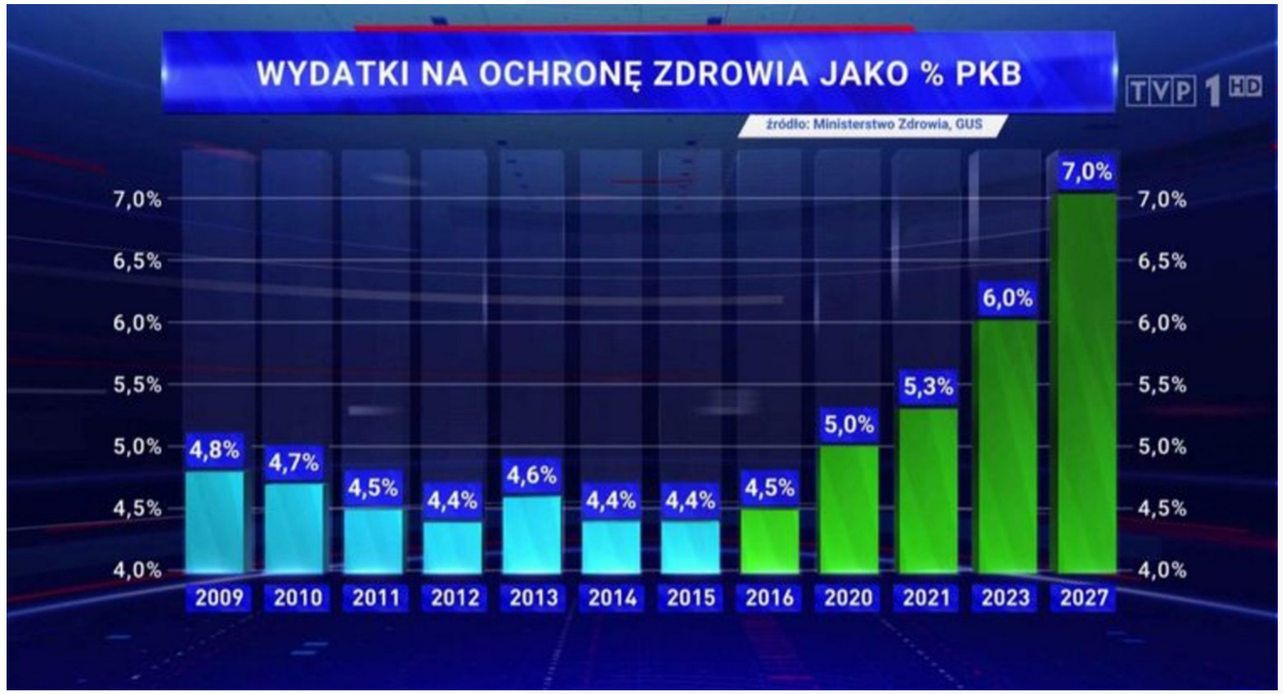

Breast Cancer Risk in Northeastern U.S.
160
140
120
100
80
60
40
20
0


CANADA
ATLANTIC OCEAN
46°N
44°N
42°N
40°N
80°W
78°W
76°W
74°W
72°W
70°W
68°W
1
2
3
4
5
Breast Cancer Risk Map - natural log scale

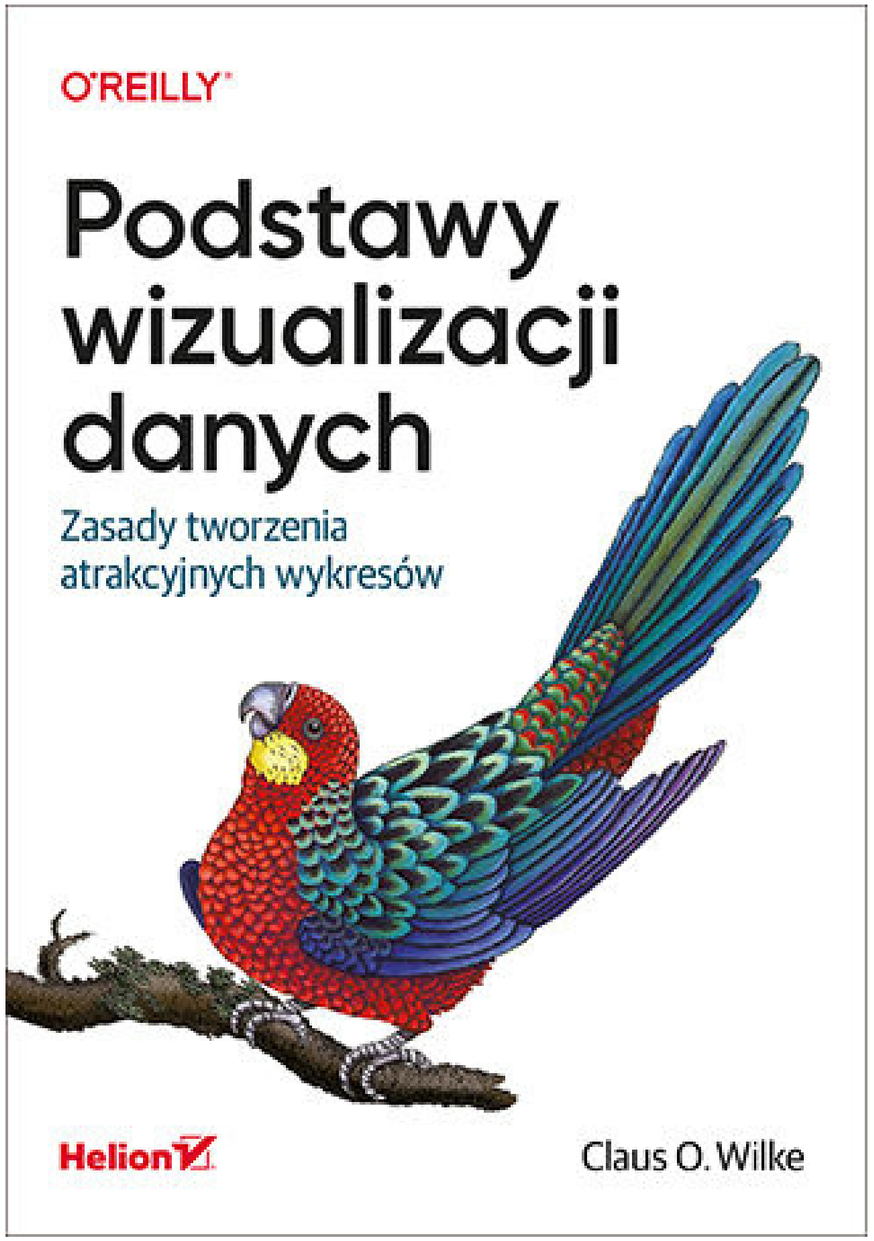
"Najtrudniejszy" wykres

TVN

TVP



https://janinadaily.com/statystyk-damian-umilowany-w-modzajto-i-wizualizacja-danych/


O'REILLY
Podstawy wizualizacji danych
Zasady tworzenia atrakcyjnych wykresów
Helion
Claus O. Wilke

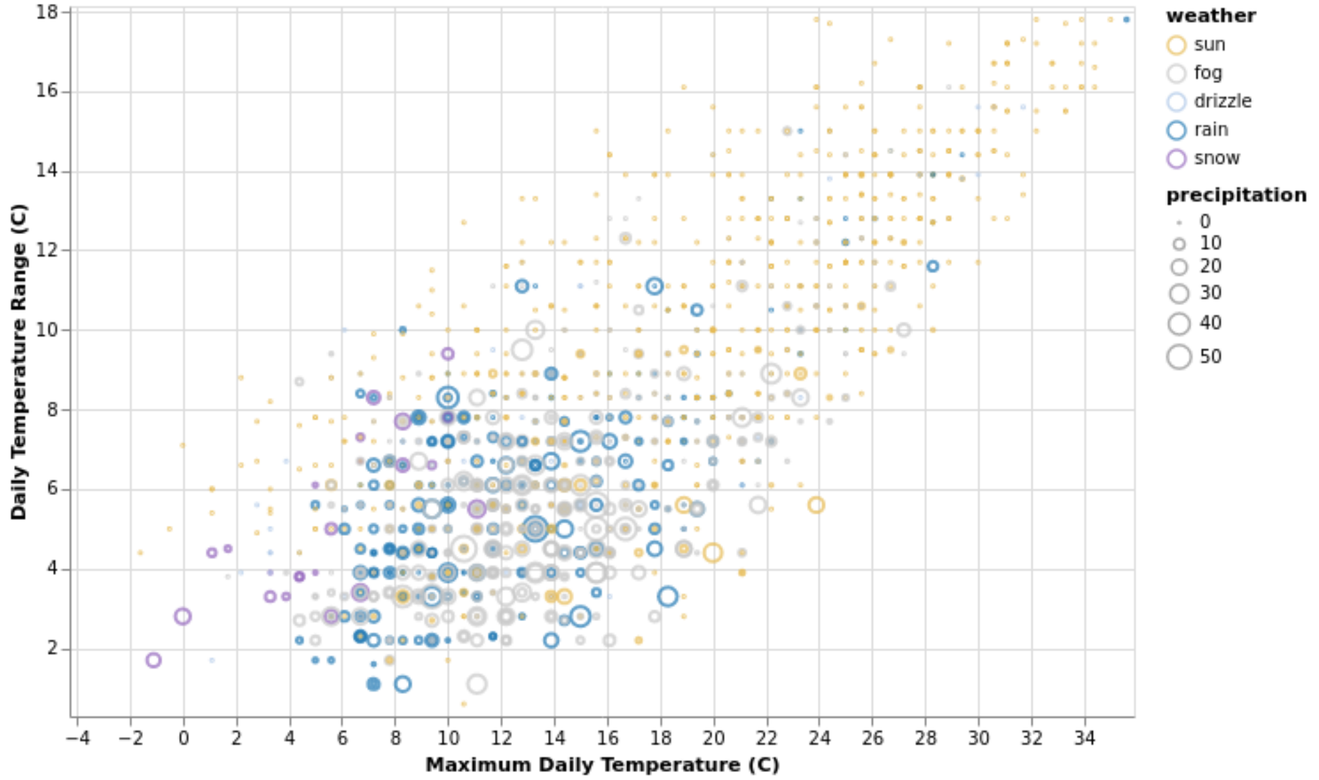
Python

(O czym sobie nie powiemy?)

Python pozwala na oskryptowanie większości programów do grafiki komputerowej, więc de facto może być używany do tworzenia zaawansowanych grafik jak i sztuki.

Altair

```
alt.Chart(df).mark_point().encode(
    alt.X('temp_max', title='Maximum Daily Temperature (C)'),
    alt.Y('temp_range:Q', title='Daily Temperature Range (C)'),
    alt.Color('weather', scale=scale),
    alt.Size('precipitation', scale=alt.Scale(range=[1, 200]))
).transform_calculate(
    "temp_range", "datum.temp_max - datum.temp_min"
).properties(
    width=600,
    height=400
).interactive()
```

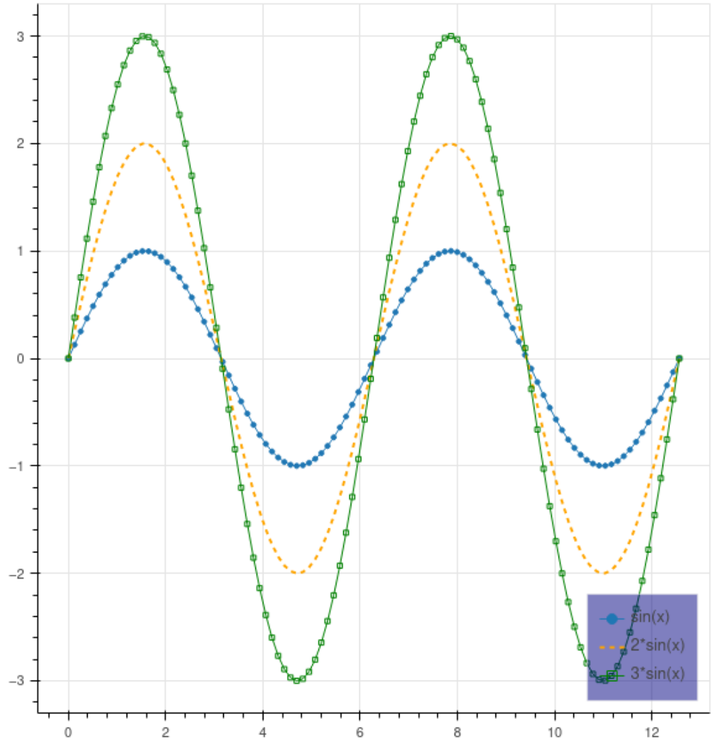


Komentarz

Zaawansowane narzędzie wizualizacji danych z backenedem napisanym w JavaScript (Vega, Vega-Lite). Tworzy piękne wykresy przy niewielu linijkach kodu ale wymaga silnika JS do pracy.

Bokeh

```python
import numpy as np

from bokeh.plotting import figure, output_file, show

x = np.linspace(0, 4*np.pi, 100)
y = np.sin(x)

output_file("legend_background.html")

p = figure()

p.circle(x, y, legend_label="sin(x)")
p.line(x, y, legend_label="sin(x)")

p.line(x, 2*y, legend_label="2*sin(x)",
       line_dash=[4, 4], line_color="orange", line_width=2)

p.square(x, 3*y, legend_label="3*sin(x)", fill_color=None, line_color="green")
p.line(x, 3*y, legend_label="3*sin(x)", line_color="green")

#  3*sin(x) curve should be under this legend at initial viewing, so
# we can see that the legend is transparent
p.legend.location = "bottom_right"
p.legend.background_fill_color = "navy"
p.legend.background_fill_alpha = 0.5

show(p)
```

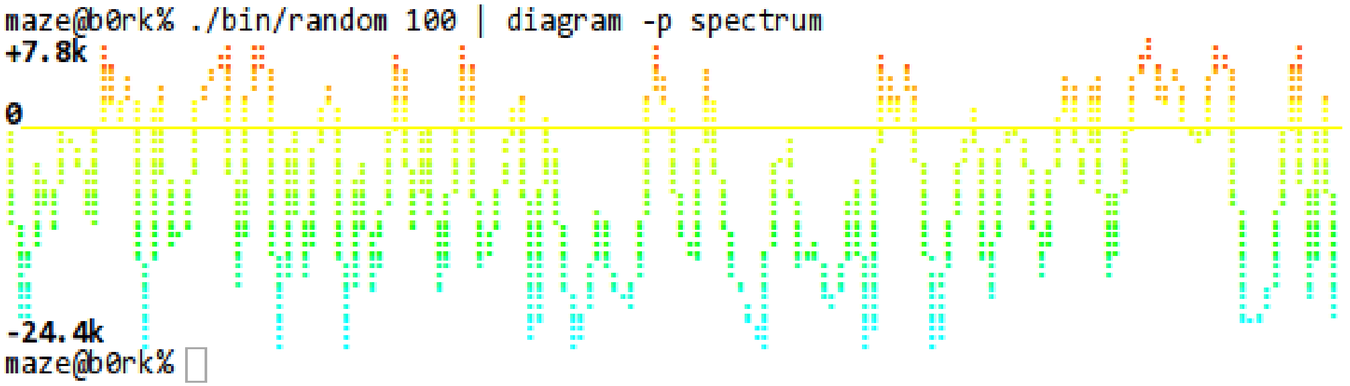


Komentarz

Zaawansowane narzędzie wizualizacji danych z backenedem napisanym w JavaScipt. **Ma duże wsparcie do tworzenia aplikacji webowych, np.: w połączeniu z Flaskiem.**

diagram

```
from diagram import DGWrapper
gram = DGWrapper(data=[points, values])
gram.show()
```

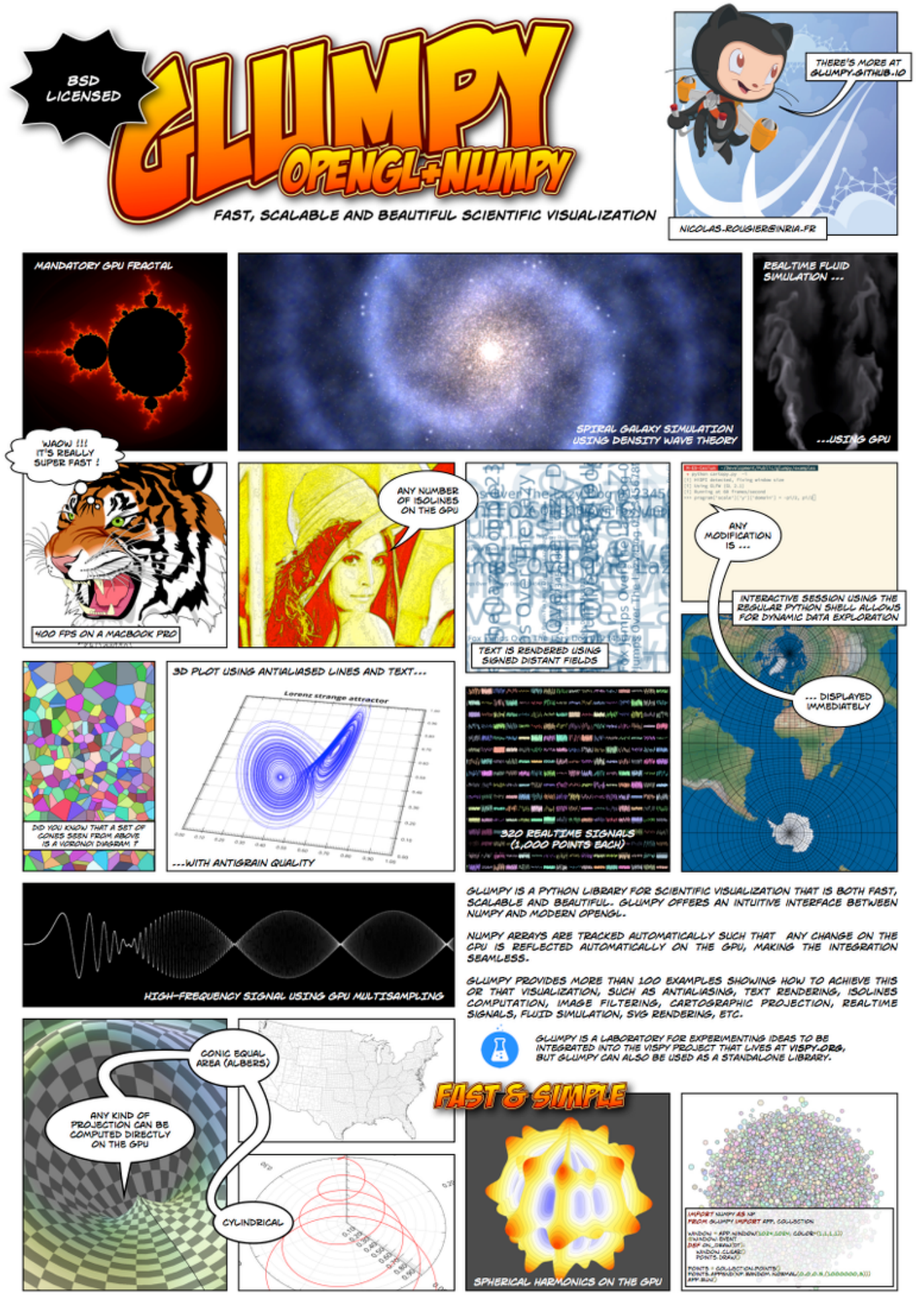


Komentarz

Z oficjalnej dokumentacji: Text mode diagrams using UTF-8 characters and fancy colors (using Python). Link do repo:

https://github.com/tehmaze/diagram

glumpy (vispy)



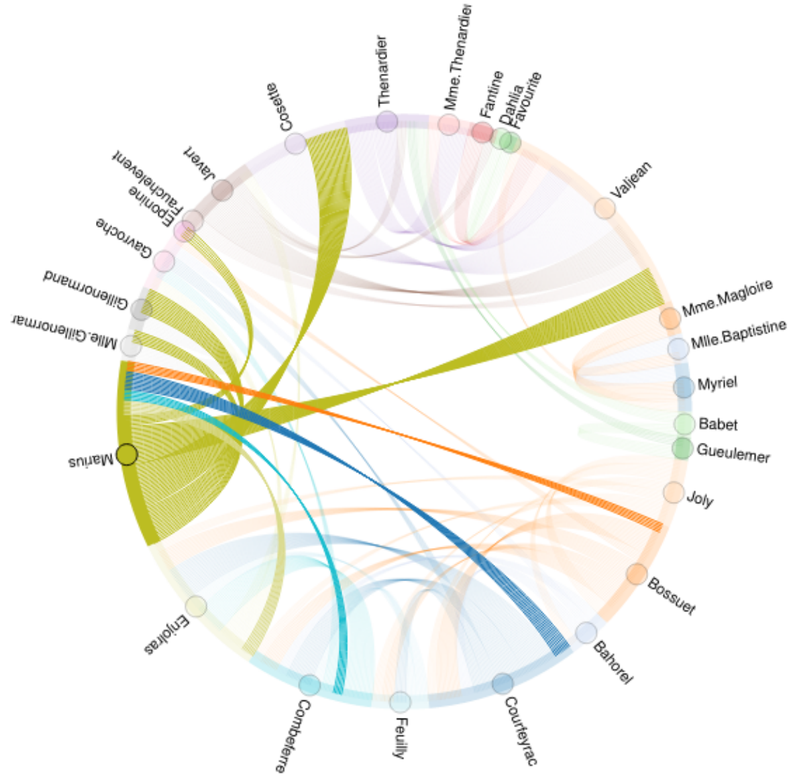
Komentarz

Narzędzie do symulacji fizycznych i rysowania wielkich zbiorów danych, czysty Python.

+ Vispy - to samo co glumpy, tylko interaktywne wizualizacje.

HoloViews

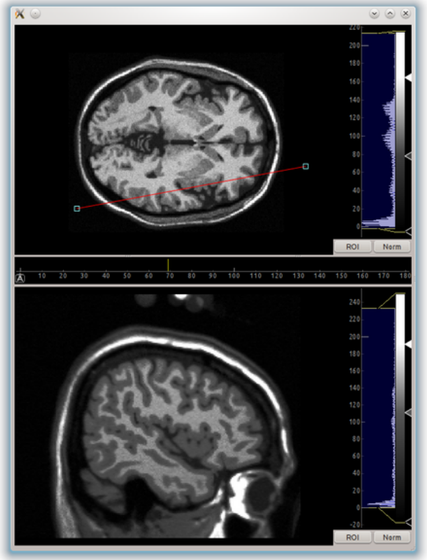
```python
import pandas as pd
import holoviews as hv
from holoviews import opts, dim
from bokeh.sampledata.les_mis import data

hv.extension('bokeh')
hv.output(size=200)

links = pd.DataFrame(data['links'])

nodes = hv.Dataset(pd.DataFrame(data['nodes']), 'index')
nodes.data.head()

chord = hv.Chord((links, nodes)).select(value=(5, None))
chord.opts(
    opts.Chord(cmap='Category20', edge_cmap='Category20', edge_color=dim('source').str(),
               labels='name', node_color=dim('index').str()))
```

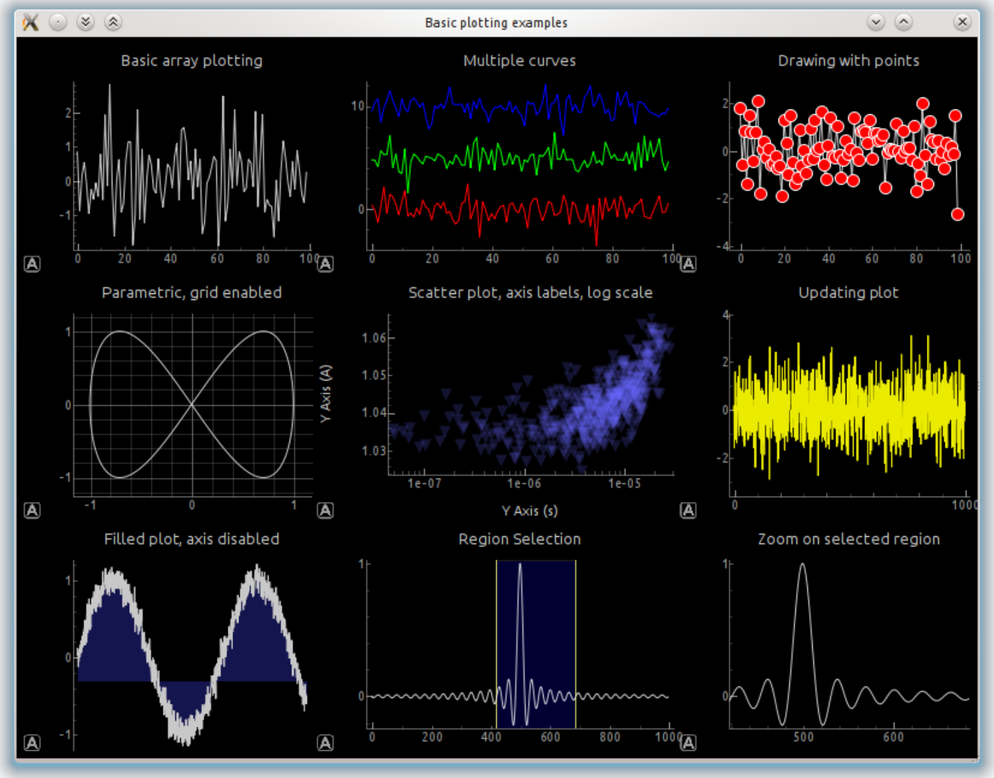


Komentarz

Narzędzie wykorzystujące Bokeh albo Matplotlib, ułatwiające tworzenie ładnych, interaktywnych wizualizacji danych.

PyQTGraph

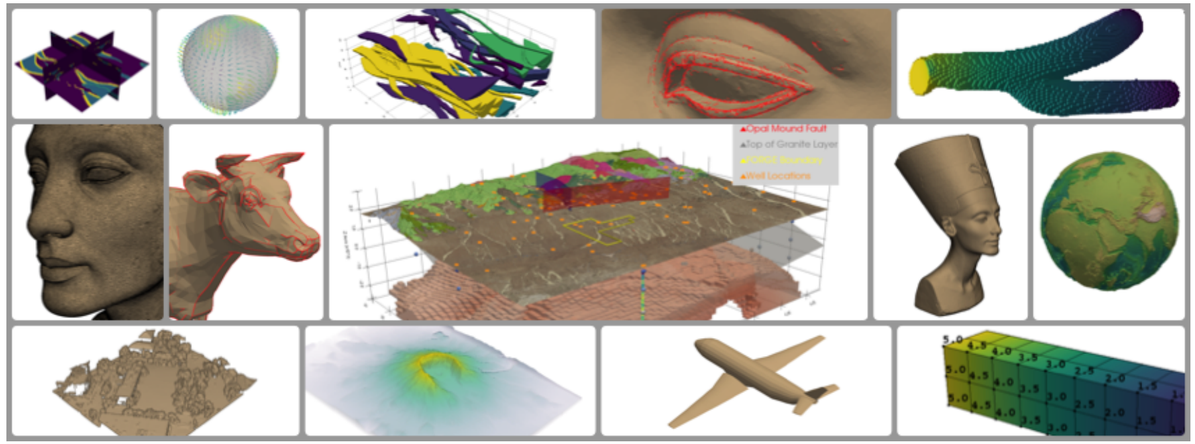




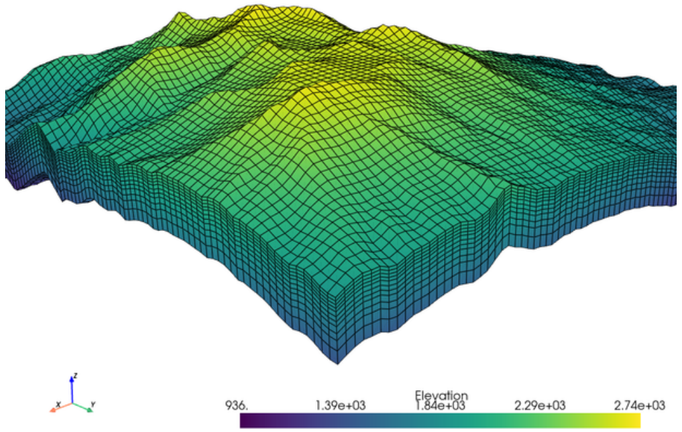
Komentarz

Jeśli budujesz programy z GUI na platformy desktopowe, albo jeśli tworzysz programy w Qt, wtedy ta biblioteka jest niezastąpiona. (Integracja Matplotlib z Qt to katorga, ostrzegam!)

PyVista



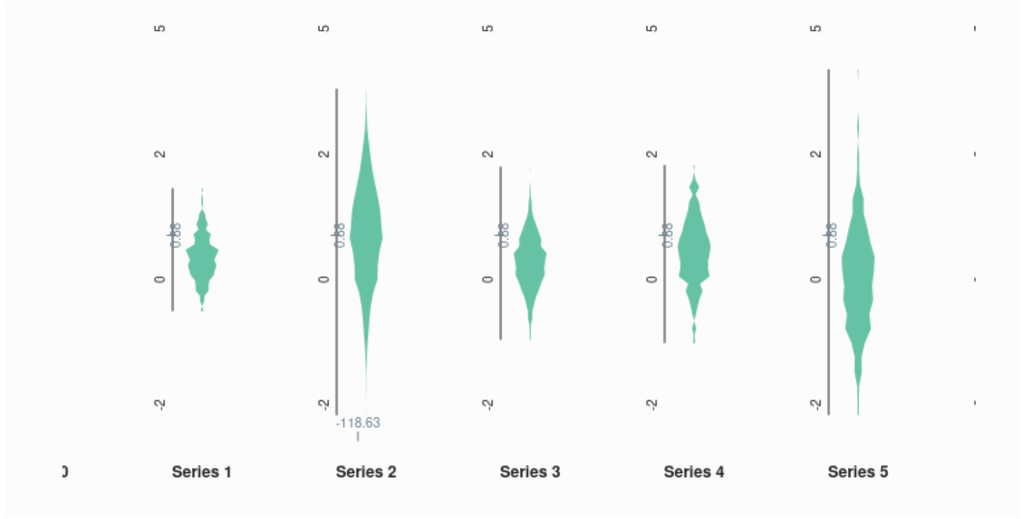

```python
import numpy as np

import pyvista as pv
from pyvista import examples

dem = examples.download_crater_topo()

subset = dem.extract_subset((500, 900, 400, 800, 0, 0), (5, 5, 1))

terrain = subset.warp_by_scalar()

z_cells = np.array([25] * 5 + [35] * 3 + [50] * 2 + [75, 100])

xx = np.repeat(terrain.x, len(z_cells), axis=-1)
yy = np.repeat(terrain.y, len(z_cells), axis=-1)
zz = np.repeat(terrain.z, len(z_cells), axis=-1) - np.cumsum(z_cells).reshape((1, 1, -1))

mesh = pv.StructuredGrid(xx, yy, zz)
mesh["Elevation"] = zz.ravel(order="F")

cpos = [
    (1826736.796308761, 5655837.275274233, 4676.8405505181745),
    (1821866.179851998, 5649248.765538796, 943.0995128226014),
    (-0.2797856225380979, -0.27966946337594883, 0.9184252809434081),
]

mesh.plot(show_edges=True, lighting=False, cpos=cpos)
```

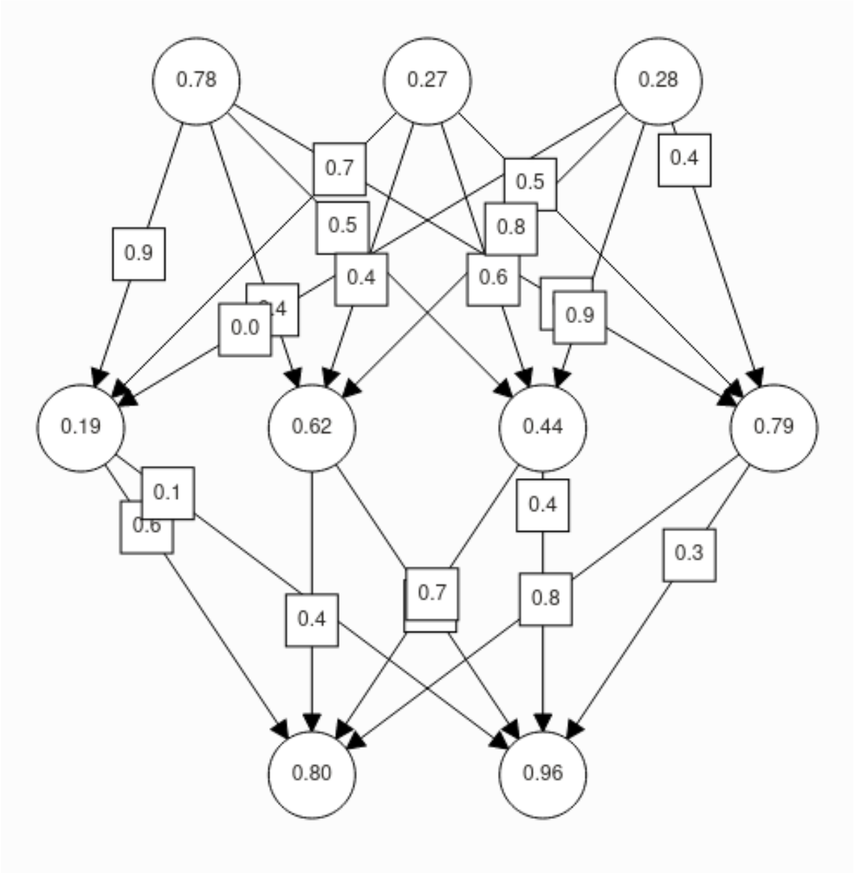


Komentarz

Niezbędnik dla każdego badacza danych, który pracuje z danymi wolumetrycznymi.

toyplot

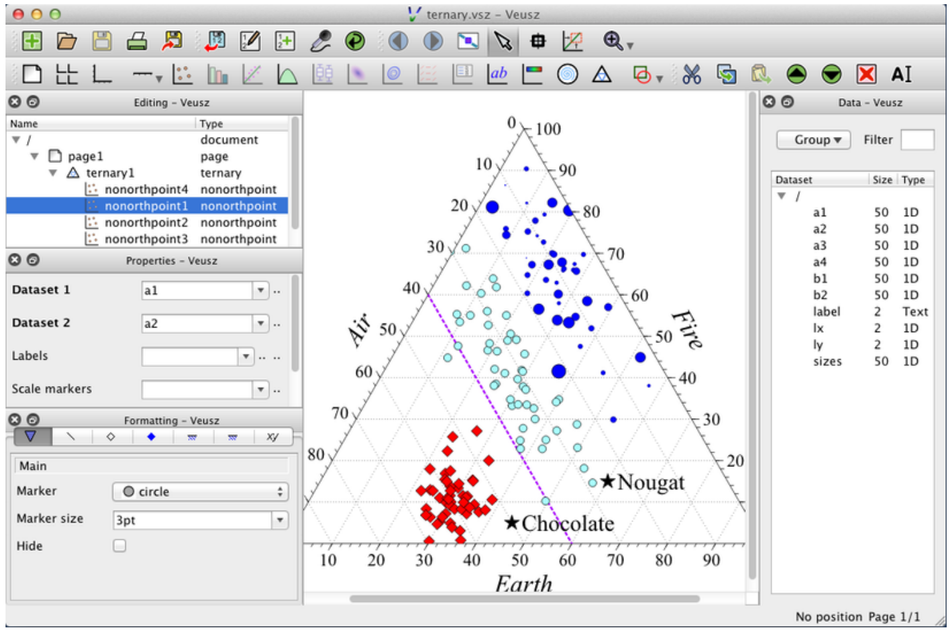


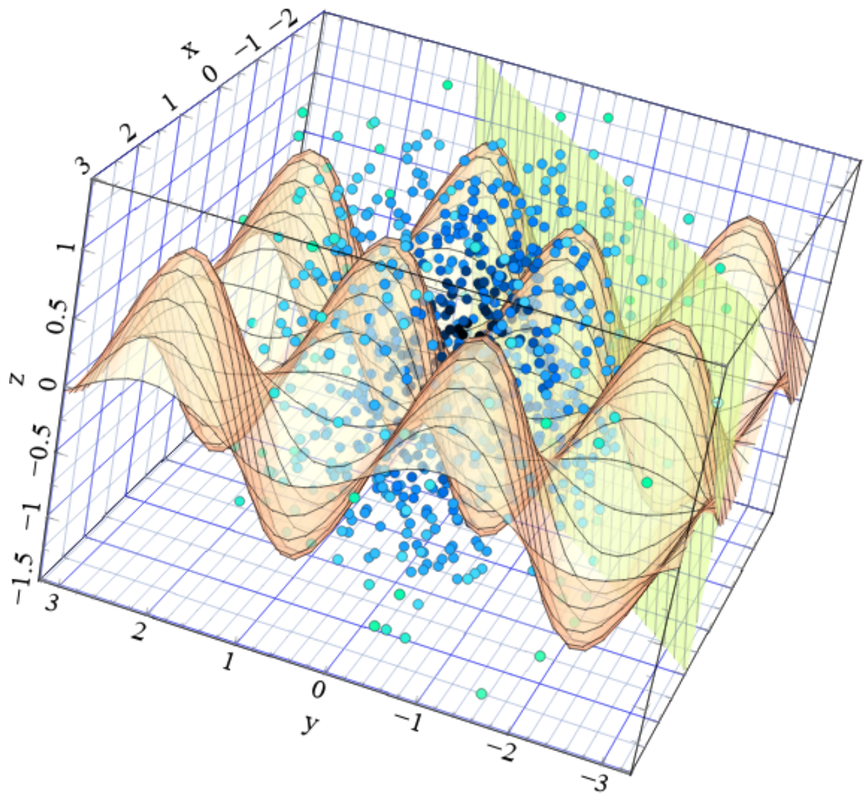


Komentarz

Czyste i niewielkie wizualizacje dla elektronicznych publikacji (svg).

Veusz





Komentarz

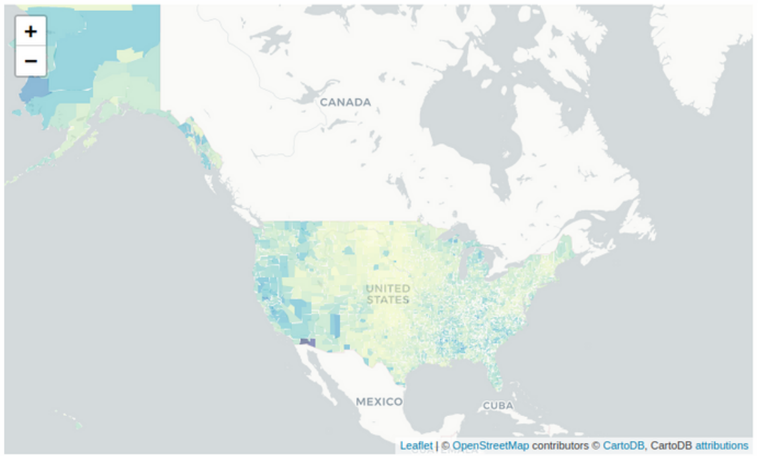
Narzędzie do przygotowywania obrazów do publikacji.

Folium

```
""" flask_example.py

    Required packages:
    - flask
    - folium

    Usage:

    Start the flask server by running:

        $ python flask_example.py

    And then head to http://127.0.0.1:5000/ in your browser to see the map displayed

"""

from flask import Flask

import folium

app = Flask(__name__)


@app.route('/')
def index():
    start_coords = (46.9540700, 142.7360300)
    folium_map = folium.Map(location=start_coords, zoom_start=14)
    return folium_map._repr_html_()


if __name__ == '__main__':
    app.run(debug=True)
```

[16]:
```
colorscale = branca.colormap.linear.YlGnBu_09.scale(0, 30)

employed_series = df.set_index("FIPS_Code")["Unemployment_rate_2011"]


def style_function(feature):
    employed = employed_series.get(int(feature["id"][-5:]), None)
    return {
        "fillOpacity": 0.5,
        "weight": 0,
        "fillColor": "#black" if employed is None else colorscale(employed),
    }


m = folium.Map(location=[48, -102], tiles="cartodbpositron", zoom_start=3)

folium.TopoJson(
    json.loads(requests.get(county_geo).text),
    "objects.us_counties_20m",
    style_function=style_function,
).add_to(m)

m
```

[16]:



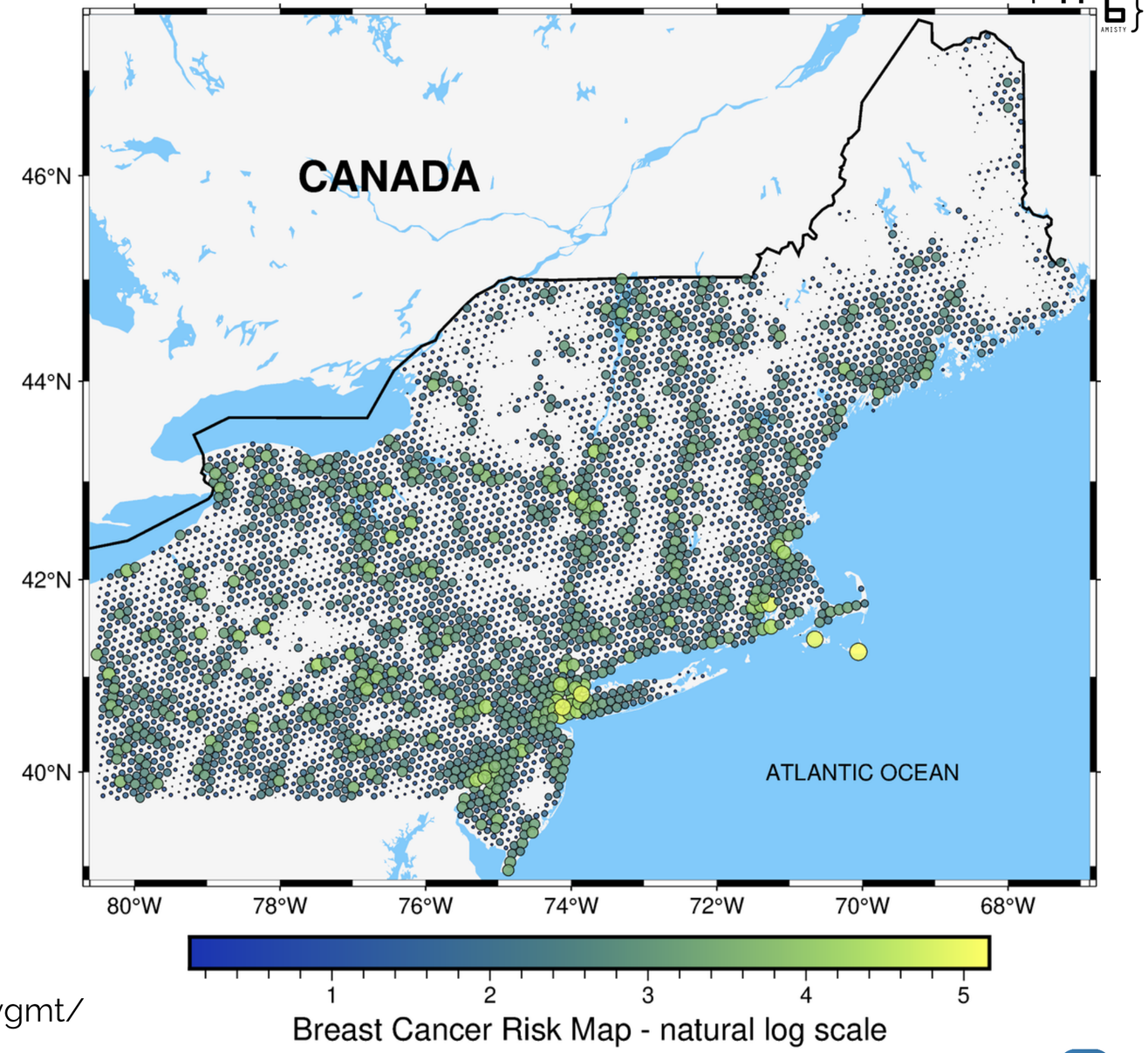
[17]:
```
colorscale = branca.colormap.linear.PuRd_09.scale(0, 100000)

employed_series = df.set_index("FIPS_Code")["Median_Household_Income_2011"].dropna()


def style_function(feature):
```

Komentarz

Narzędzie do publikacji map online, Pythonowa nakładka na bibliotekę Leaflet.js

pyGMT

```python
fig = pygmt.Figure()
pygmt.makecpt(cmap="imola", series=[transformed.min(), transformed.max()])
fig.basemap(region=region, projection="M15c", frame=True)
fig.coast(land="#F5F5F5", water="#82CAFA", borders='1/1p')
fig.text(text="CANADA", x=-76.5, y=46, font="18p,Helvetica-Bold")
fig.text(text="ATLANTIC OCEAN", x=-70, y=40)
fig.plot(
    x=data2.geometry.x,
    y=data2.geometry.y,
    size=transformed / 20,
    style="cc",
    color=transformed,
    cmap=True,
    pen="black",
    transparency=15
)
fig.colorbar(frame='af+l"Breast Cancer Risk Map - natural log scale"')
fig.show()
```



Komentarz

Narzędzie do tworzenia pięknych map pod druk.

https://ml-gis-service.com/index.php/2022/01/06/data-science-leave-geopandas-and-create-beautiful-map-with-pygmt/

Wielka Trójca

matplotlib

seaborn

plotly | Graphing Libraries

JS

Komentarz

Najpopularniejsze i najważniejsze narzędzia graficzne w Pythonie.

JavaScript

(O czym sobie nie powiemy?)

O urządzeniach mobilnych (wielki, wciąż otwarty rynek na rozwiązania bazujące na JS i pozwalające na wyświetlanie ładnych i informacyjnych grafik) i o responsywności w kontekście wizualizacji danych.

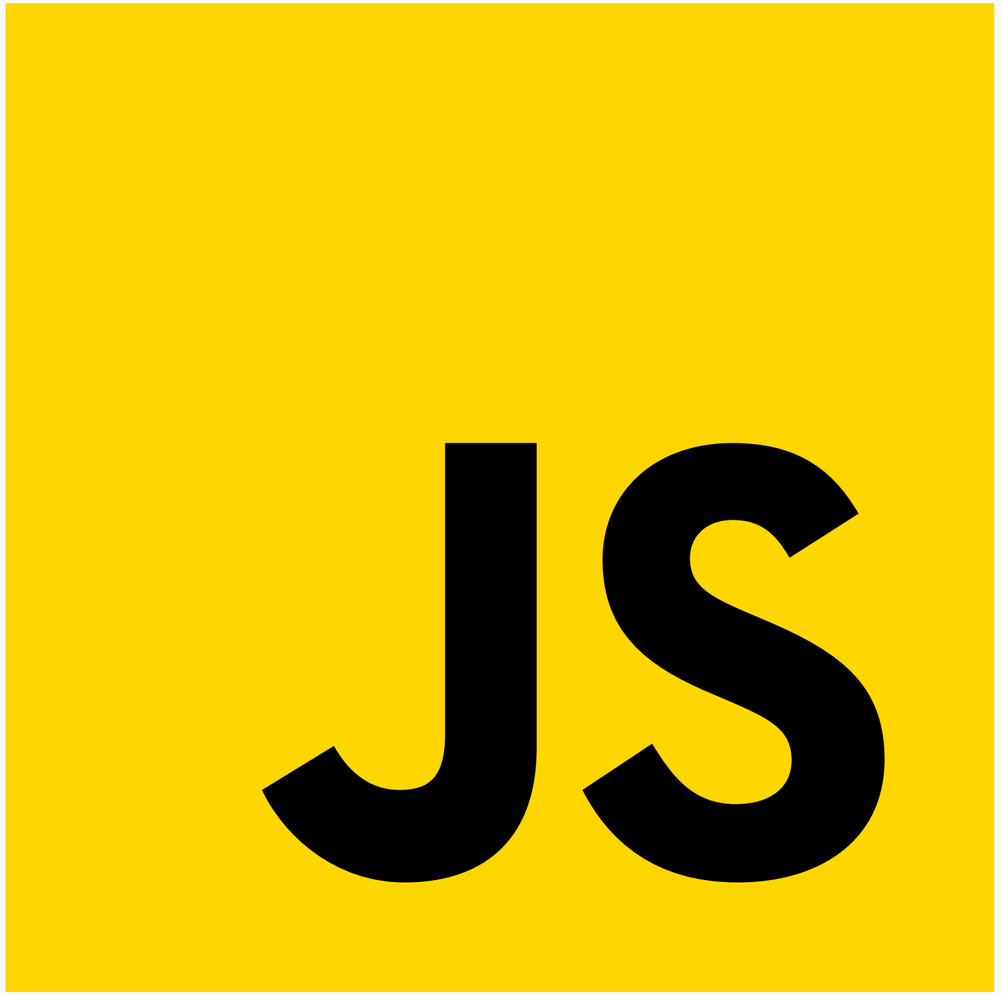
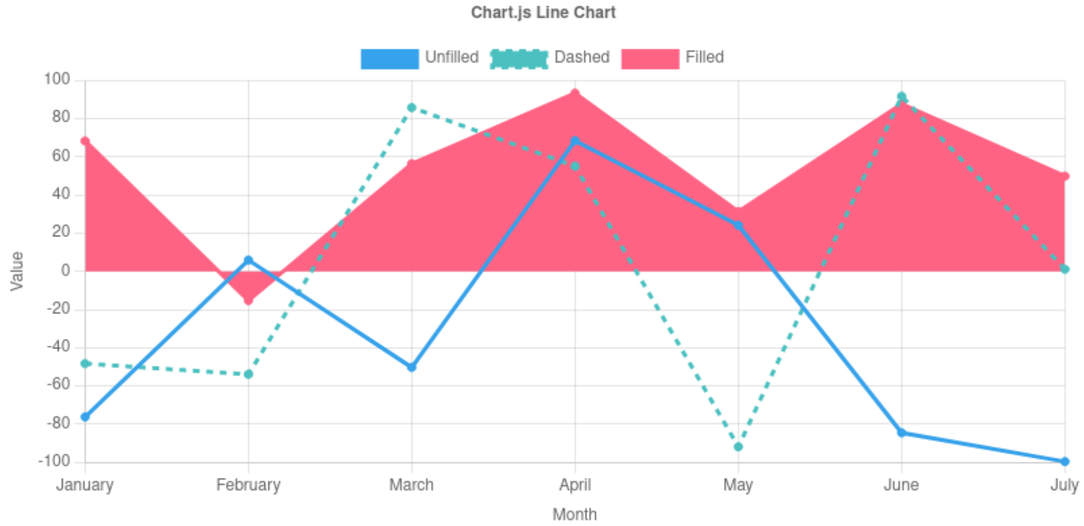


Chart.js

```
const config = {
  type: 'line',
  data: data,
  options: {
    responsive: true,
    plugins: {
      title: {
        display: true,
        text: 'Chart.js Line Chart'
      },
    },
    interaction: {
      mode: 'index',
      intersect: false
    },
    scales: {
      x: {
        display: true,
        title: {
          display: true,
          text: 'Month'
        }
      },
      y: {
        display: true,
        title: {
          display: true,
          text: 'Value'
        }
      }
    }
  },
};
```

```
const DATA_COUNT = 7;
const NUMBER_CFG = {count: DATA_COUNT, min: -100, max: 100};

const labels = Utils.months({count: DATA_COUNT});
const data = {
  labels: labels,
  datasets: [
    {
      label: 'Unfilled',
      fill: false,
      backgroundColor: Utils.CHART_COLORS.blue,
      borderColor: Utils.CHART_COLORS.blue,
      data: Utils.numbers(NUMBER_CFG),
    }, {
      label: 'Dashed',
      fill: false,
      backgroundColor: Utils.CHART_COLORS.green,
      borderColor: Utils.CHART_COLORS.green,
      borderDash: [5, 5],
      data: Utils.numbers(NUMBER_CFG),
    }, {
      label: 'Filled',
      backgroundColor: Utils.CHART_COLORS.red,
      borderColor: Utils.CHART_COLORS.red,
      data: Utils.numbers(NUMBER_CFG),
      fill: true,
    }
  ]
};
```

Line Styling



Komentarz

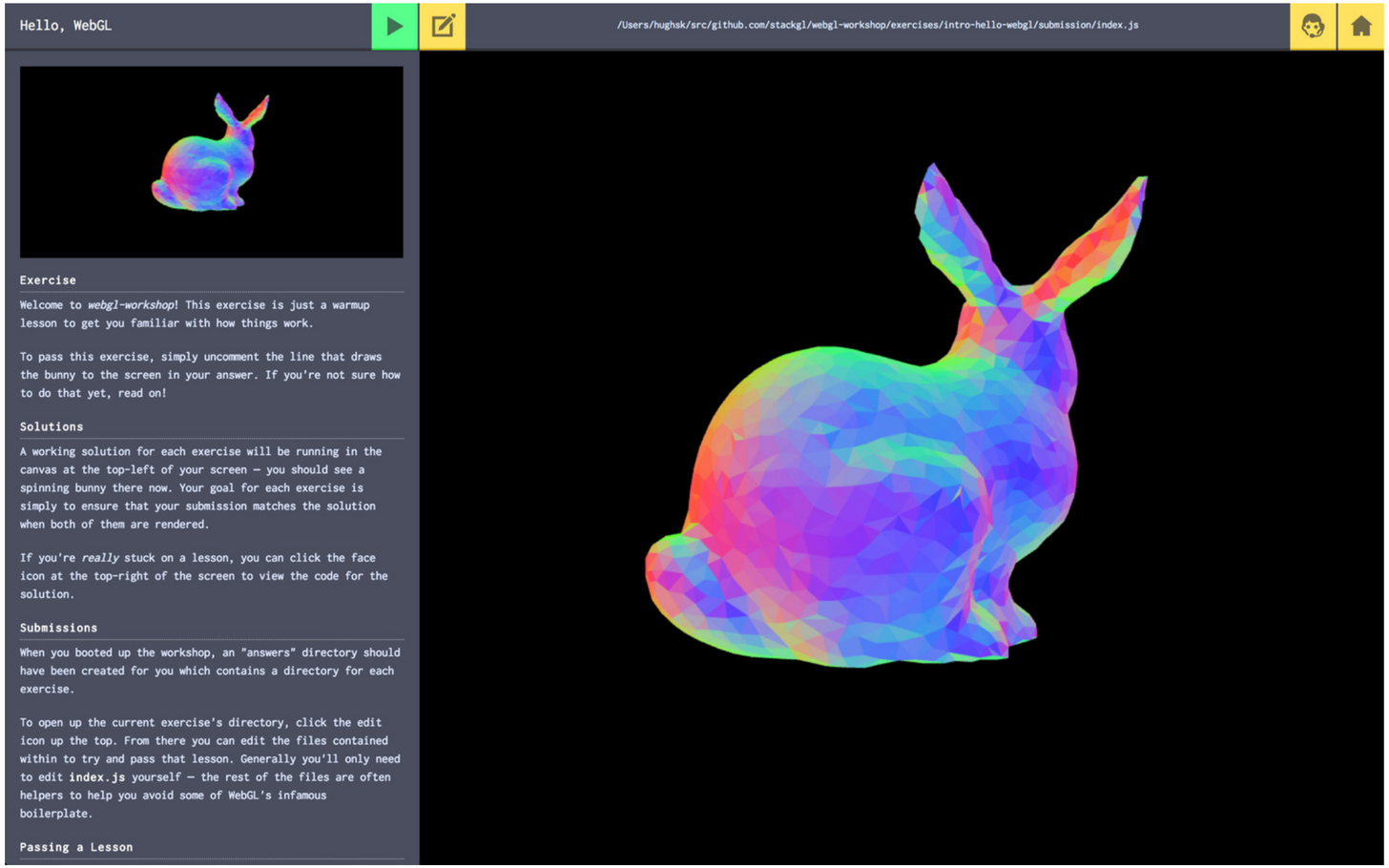
Bardzo szybkie narzędzie do renderowania grafik i wykresów w HTML5.

D3.js

Komentarz

Najpopularniejsza biblioteka do wizualizacji danych na stronach internetowych, zawiera bardzo dużo funkcjonalności, renderuje wykresy w postaci SVG, HTML i CSS, pozwala na tworzenie prawdziwie interaktywnych i customowych wizualizacji. https://observablehq.com/@d3/gallery

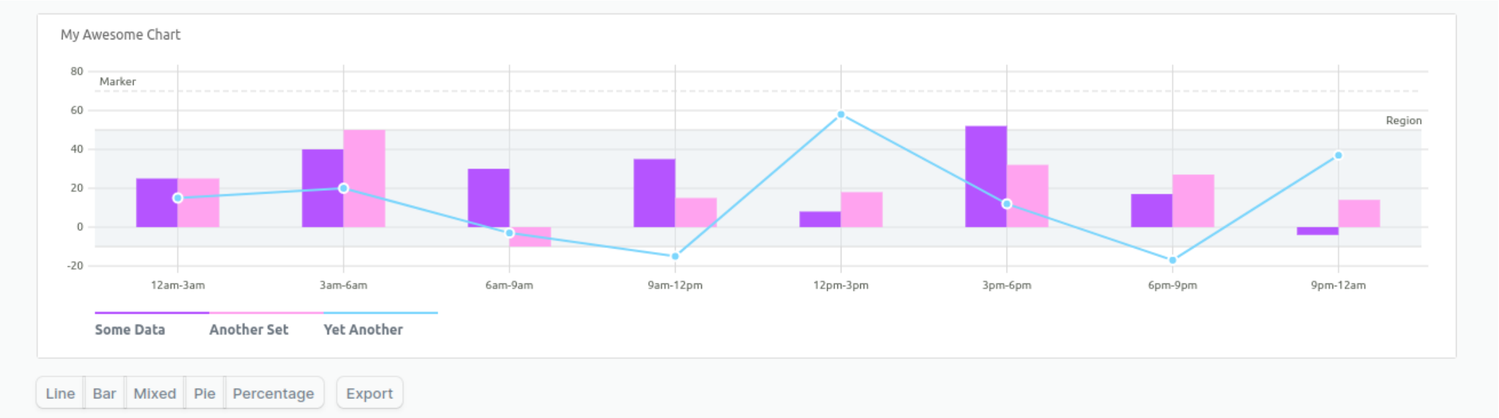
stack.gl



Komentarz

Renderowanie grafiki 3D w przeglądarce, tworzenie pięknych animacji i wizualizacja danych.

Frappe Charts




```
let chart = new frappe.Chart( "#frost-chart", { // or DOM element
  data: {
  labels: ["12am-3am", "3am-6am", "6am-9am", "9am-12pm",
      "12pm-3pm", "3pm-6pm", "6pm-9pm", "9pm-12am"],

  datasets: [
      {
          name: "Some Data", chartType: 'bar',
          values: [25, 40, 30, 35, 8, 52, 17, -4]
      },
      {
          name: "Another Set", chartType: 'bar',
          values: [25, 50, -10, 15, 18, 32, 27, 14]
      },
      {
          name: "Yet Another", chartType: 'line',
          values: [15, 20, -3, -15, 58, 12, -17, 37]
      }
  ],

  yMarkers: [{ label: "Marker", value: 70,
      options: { labelPos: 'left' }}],
  yRegions: [{ label: "Region", start: -10, end: 50,
      options: { labelPos: 'right' }}]
  },

  title: "My Awesome Chart",
  type: 'axis-mixed', // or 'bar', 'line', 'pie', 'percentage'
  height: 300,
  colors: ['purple', '#ffa3ef', 'light-blue'],

  tooltipOptions: {
      formatTooltipX: d => (d + '').toUpperCase(),
      formatTooltipY: d => d + ' pts',
  }
});

chart.export();
```

Komentarz

Wykresy w SVG z minimalną liczbą zależności.

JS

GraphicsJS

HTML

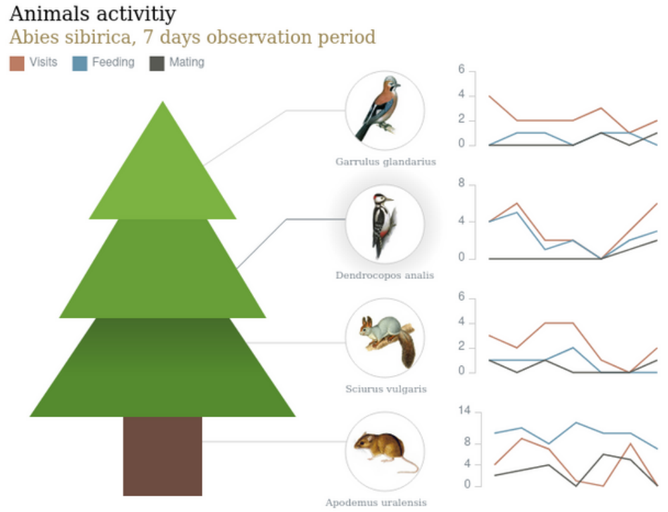
```html
<div id="container"></div>
```

CSS

```css
html,
body,
#container {
  width: 100%;
  height: 100%;
  margin: 0;
  padding: 0;
}
```

JavaScript

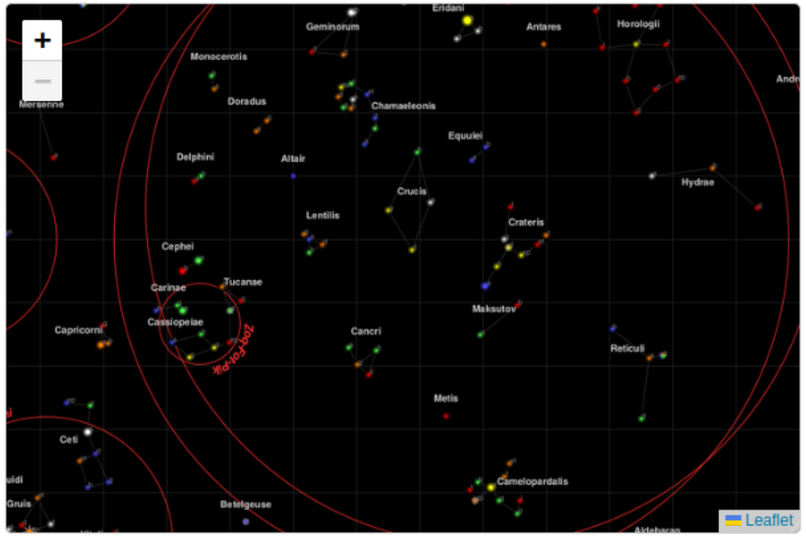
```javascript
anychart.onDocumentReady(function () {
  // create stage
  var stage = acgraph.create('container');

  // set colors for sample
  var textColor = '#7C868E';
  var lineColor = '#CECECE';
  var palette = anychart.palettes
    .distinctColors()
    .items(['#bC7C63', '#6594AD', '#585953']);

  // layer for the text elements
  var layerText = stage.layer();

  // legend on the chart
  var chartLegend = createLegend([
    { index: 0, text: 'Visits', iconFill: '#bC7C63' },
    { index: 1, text: 'Feeding', iconFill: '#6594AD' },
    { index: 2, text: 'Mating', iconFill: '#585953' }
  ]);

  chartLegend.container(layerText).draw();

  // create main title
  layerText.text(20, 20, 'Animals activitiy', { fontSize: 20 });
  // create subtitle
  layerText.text(20, 45, 'Abies sibirica, 7 days observation period', {
    fontSize: 18,
    color: '#998552',
    opacity: 20
  });

  // jay, woodpecker, squirrel, mouse
  var dataSet = [
    ['1', 4, 0, 0, 4, 4, 0, 3, 1, 1, 4, 10, 2],
    ['2', 2, 1, 0, 6, 5, 0, 2, 1, 0, 9, 11, 3],
```

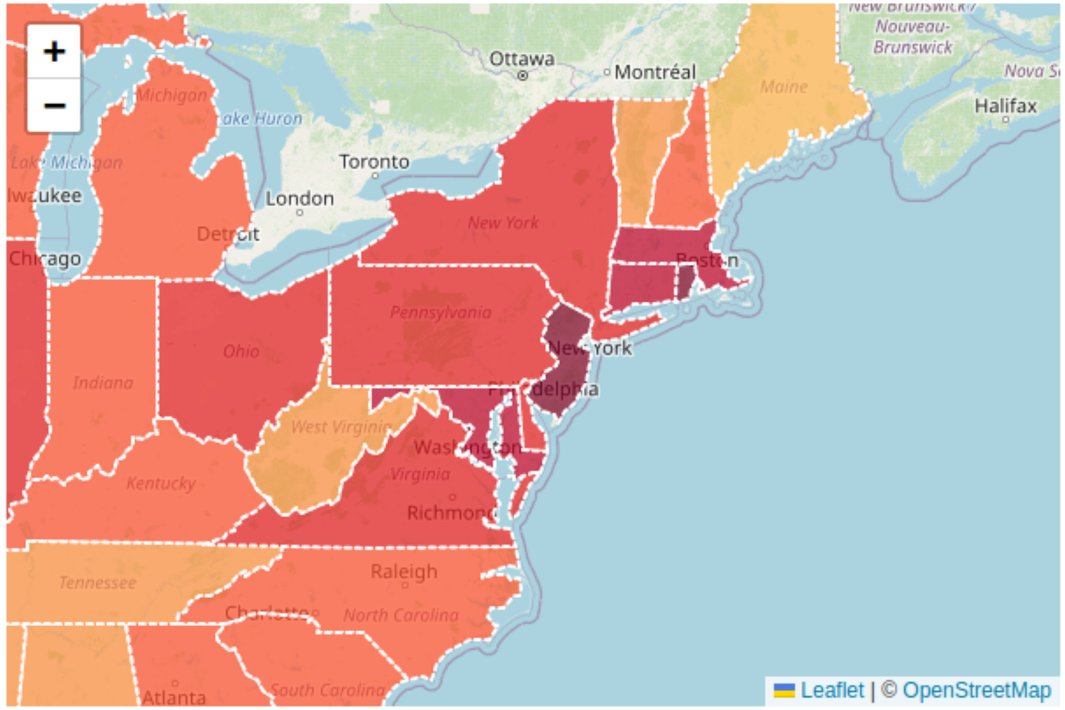


Komentarz

Biblioteka graficzna operująca na SVG/VML, pozwala na tworzenie zaawansowanych grafik na stronie internetowej.

Leafletjs



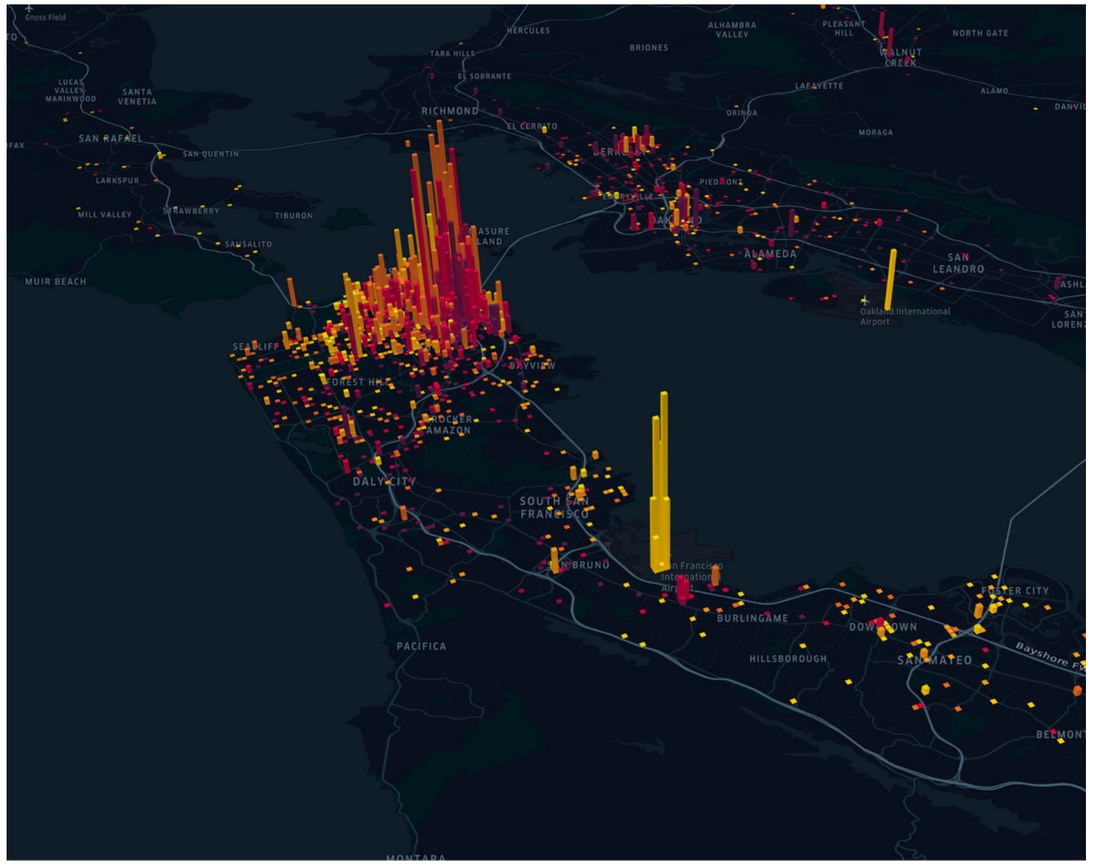


Komentarz

Najbardziej popularna i lekka biblioteka do tworzenia map.

JS

Kepler.gl

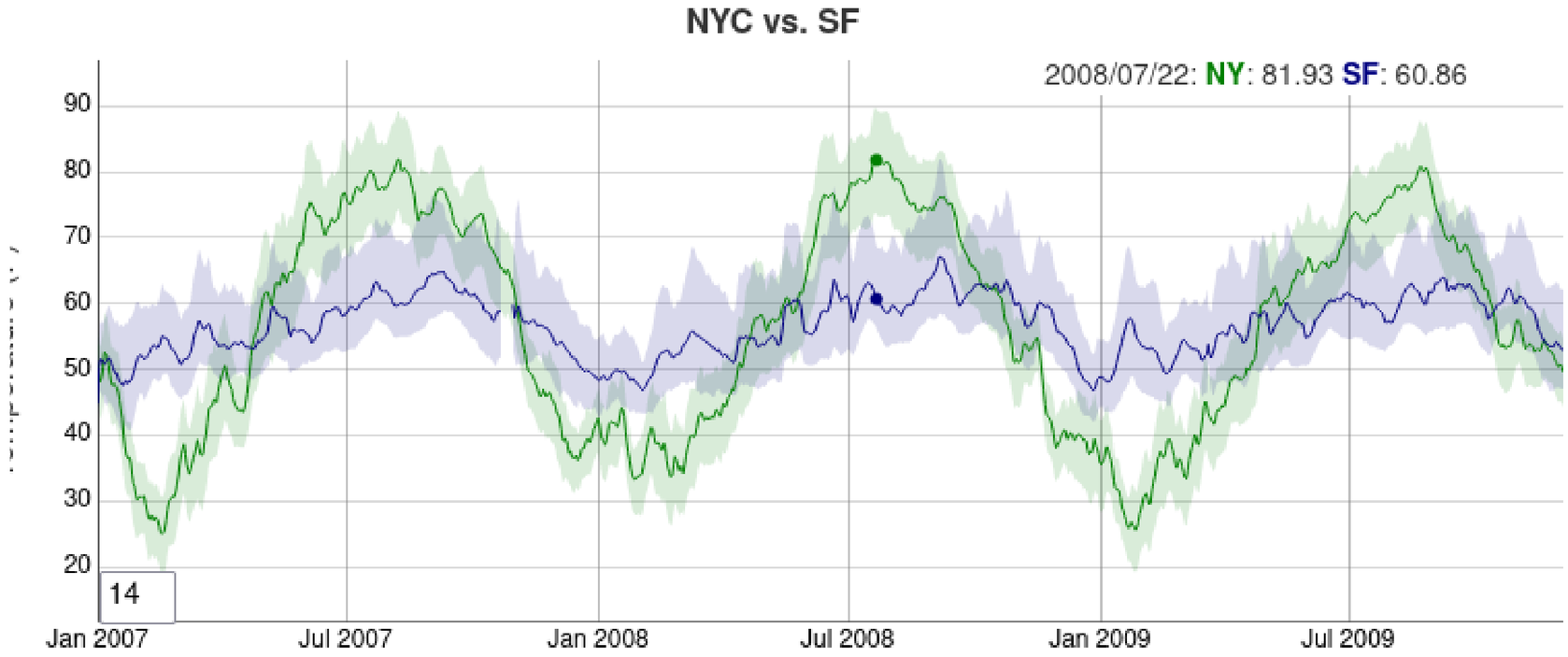


Komentarz

Biblioteka pozwalająca na tworzenie map składających się z olbrzymich ilości danych, wyświetlanie danych 3D, wbudowanie map w aplikacje. Wymagana dobra znajomość React i Redux.

Dygraphs

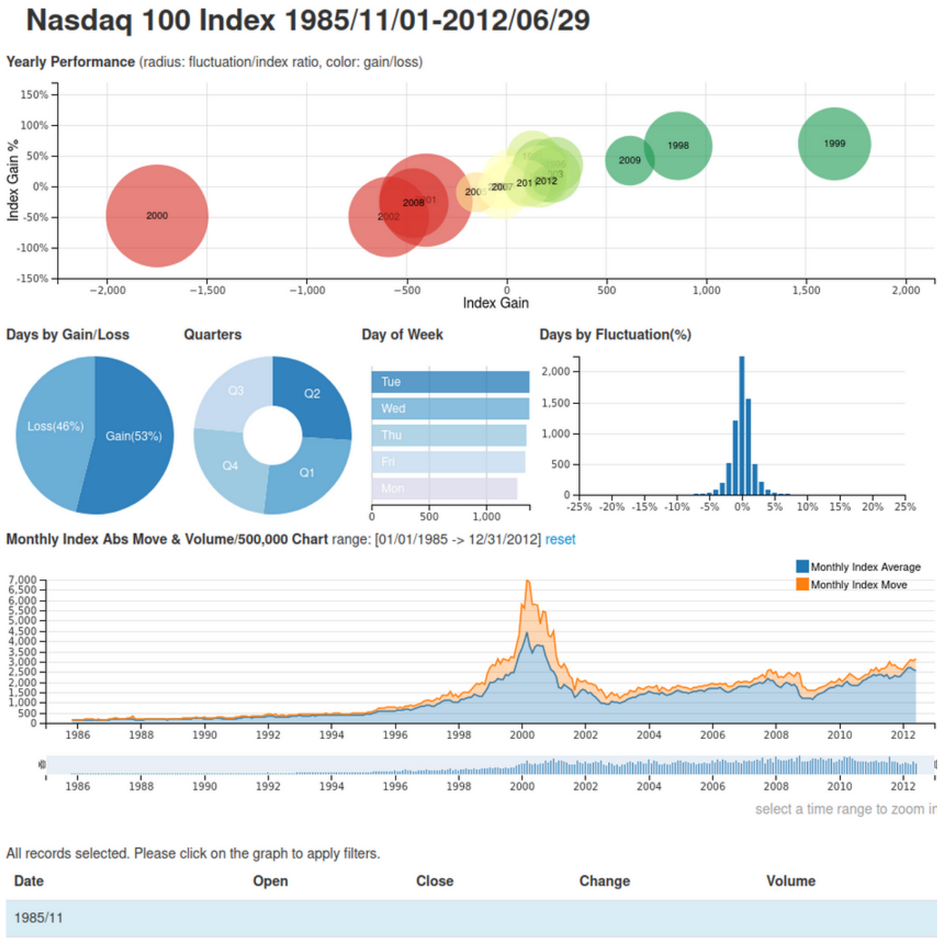
```
new Dygraph(div, "ny-vs-sf.txt", {
  legend: 'always',
  title: 'NYC vs. SF',
  showRoller: true,
  rollPeriod: 14,
  customBars: true,
  ylabel: 'Temperature (F)',
});
```



Komentarz

Niewielka biblioteka do tworzenia interaktywnych wykresów.

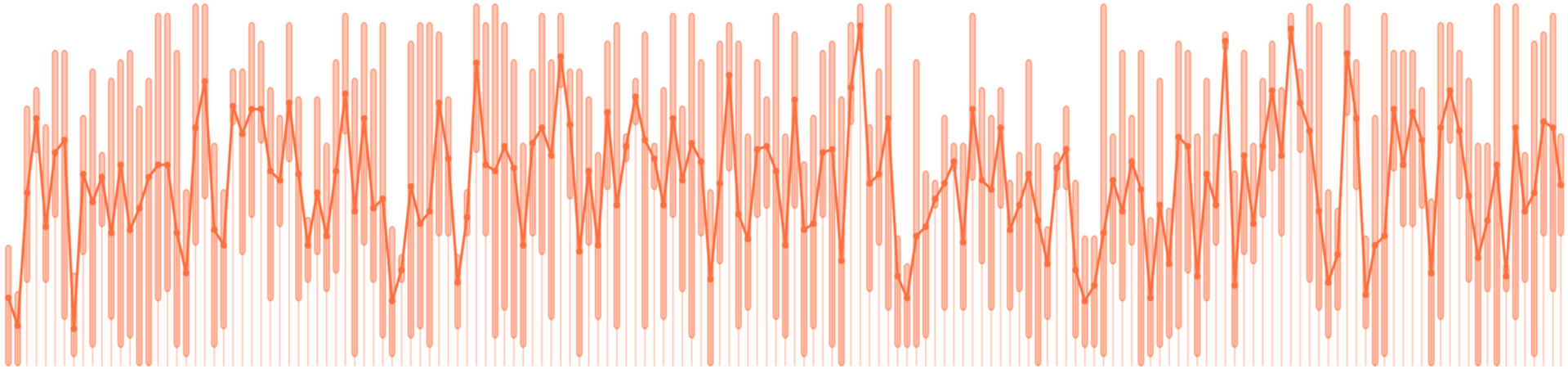
dc.js



Komentarz

Zaawansowane narzędzie, którego backend to D3.js. Pozwala na tworzenie wielowymiarowych platform wizualizacji i analizy danych, gdzie dynamicznie filtrowane jest wiele wykresów na raz!

lit-line

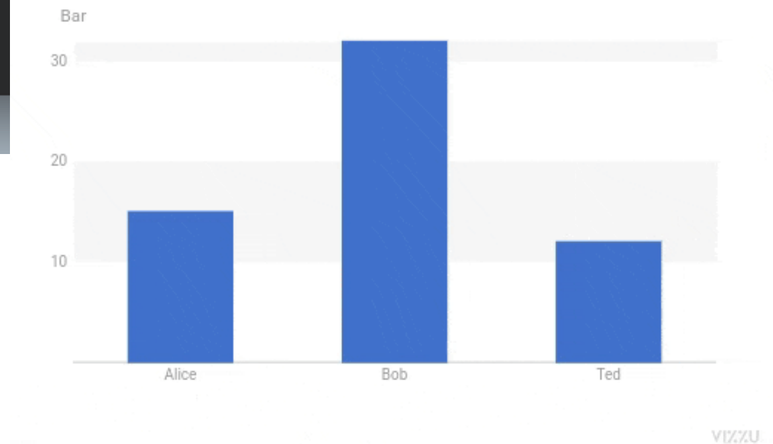


Komentarz

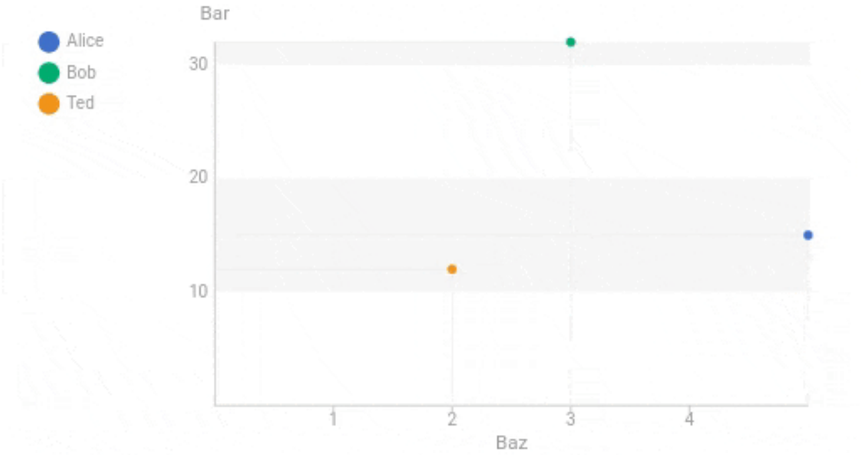
Bardzo lekka paczka (6 kb jako gzip), która służy tworzeniu wykresów liniowych.

VIZZU

```
import Vizzu from 'https://cdn.jsdelivr.net/npm/vizzu@latest/dist/vizzu.min.js';

let data = {
  series: [
    { name: 'Foo', values: ['Alice', 'Bob', 'Ted'] },
    { name: 'Bar', values: [15, 32, 12] },
    { name: 'Baz', values: [5, 3, 2] }
  ]
};

let chart = new Vizzu('myVizzu', { data });

chart.animate({
  x: 'Foo',
  y: 'Bar'
};
```




```
chart.animate({
  color: 'Foo',
  x: 'Baz',
  geometry: 'circle'
});
```

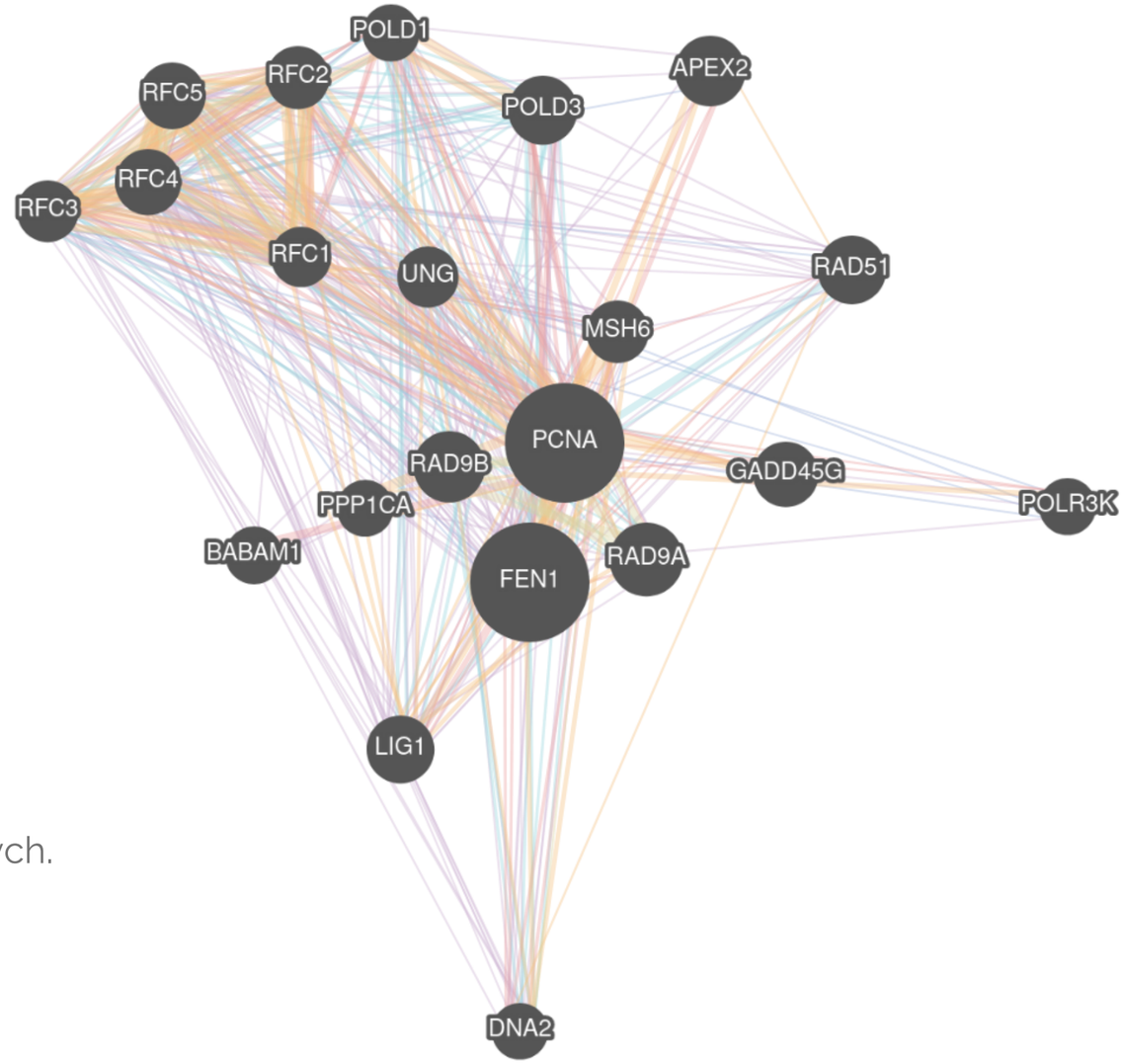


Komentarz

Tworzenie data-stories, płynne przejścia między różnymi typami wykresów.

JS

cytoscape.js

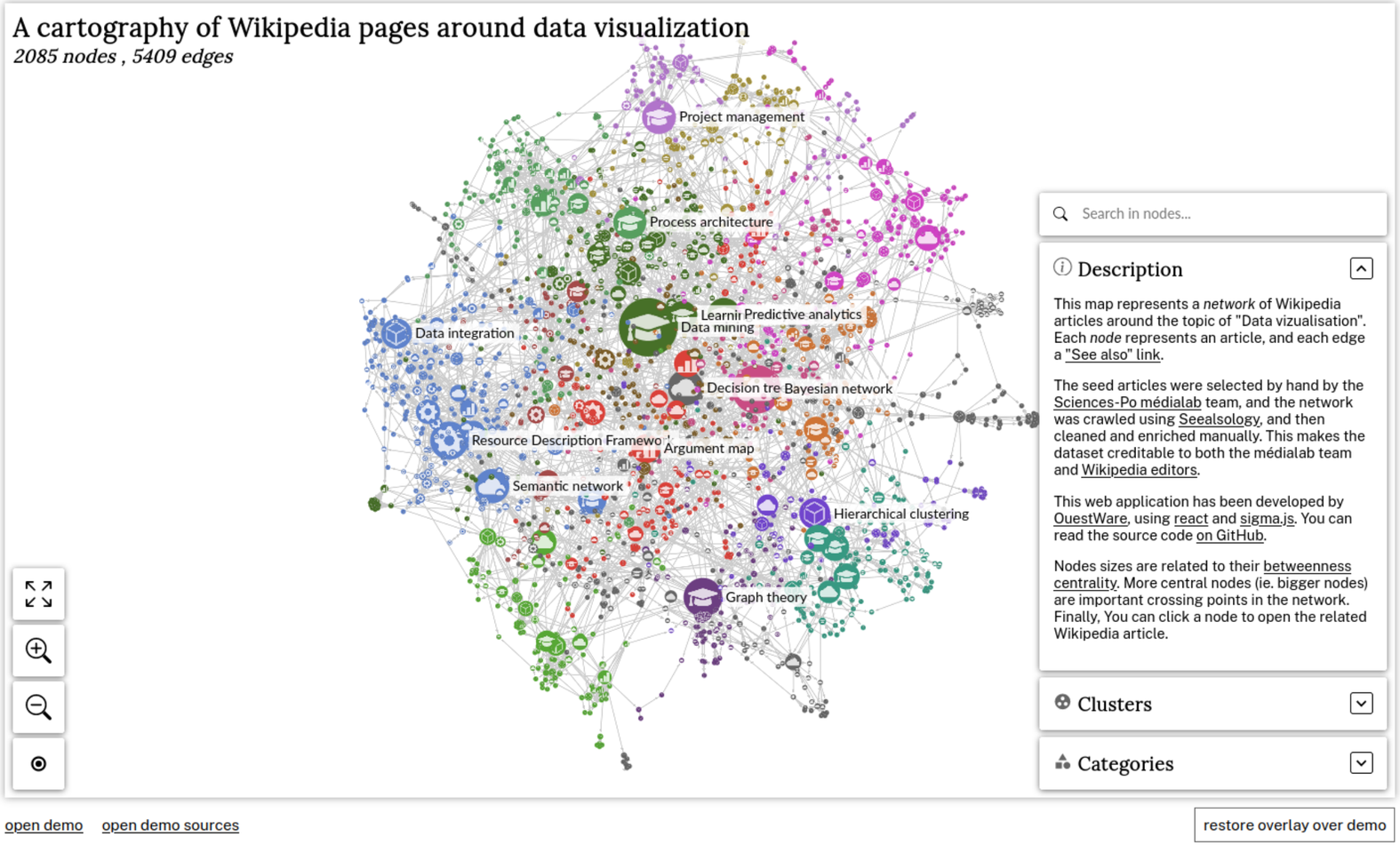


Komentarz

Biblioteka do wizualizacji danych grafowych i sieciowych.

JS

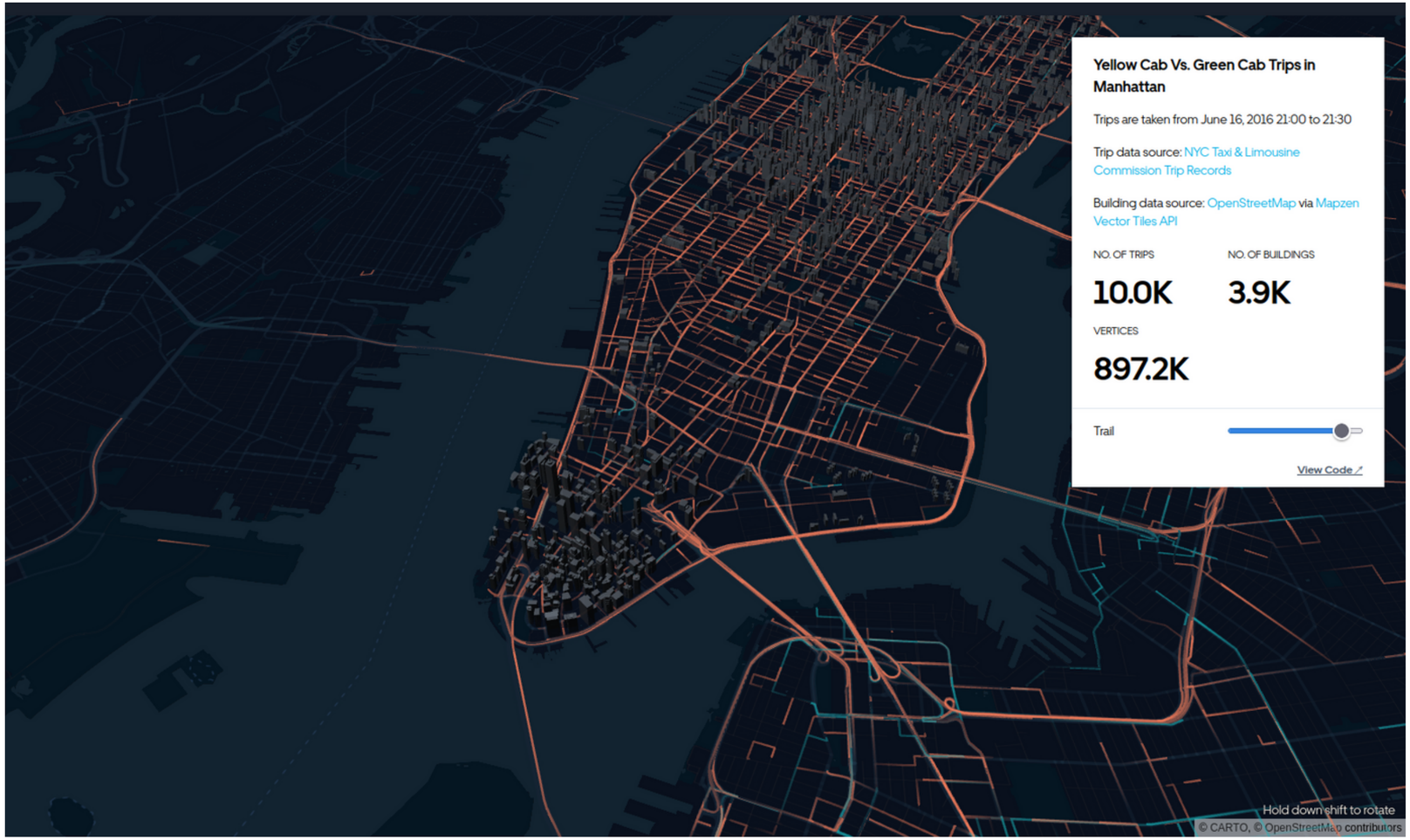
sigma.js



Komentarz

Biblioteka do wizualizacji danych grafowych i sieciowych. Pozwala również na interakcję z wierzchołkami grafów.

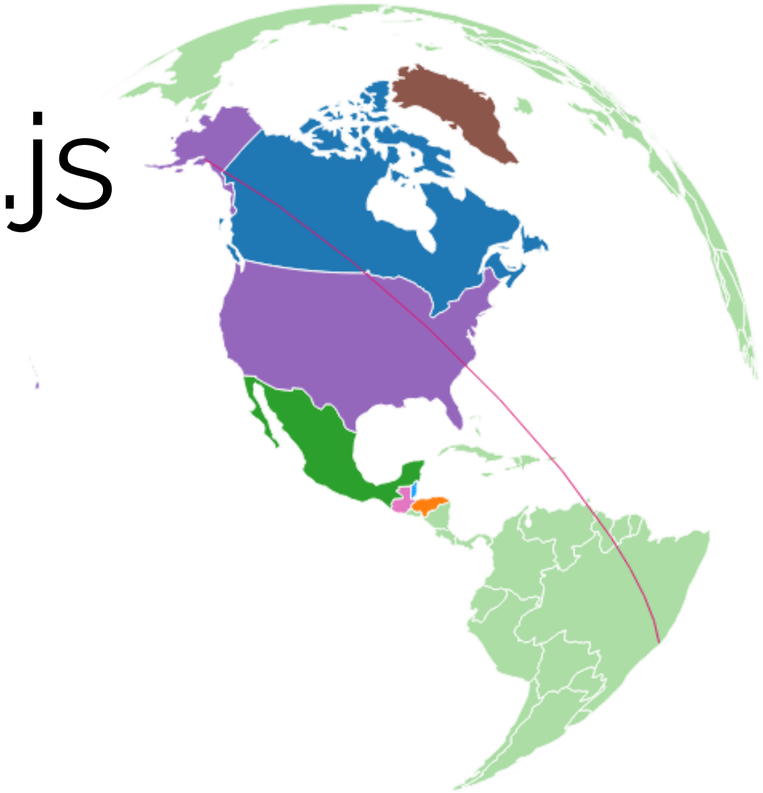
Deck.gl



Komentarz

Dynamiczne wizualizacje dla zbiorów Big Data - transport, bioinformatyka, geoinformatyka. Ma otwartą licencję!

datamaps.js



```
var map = new Datamap({
        scope: 'world',
        element: document.getElementById('projection_map'),
        projection: 'orthographic',
        fills: {
          defaultFill: "#ABDDA4",
          gt50: colors(Math.random( ) * 20),
          eq50: colors(Math.random( ) * 20),
          lt25: colors(Math.random( ) * 10),
          gt75: colors(Math.random( ) * 200),
          lt50: colors(Math.random( ) * 20),
          eq0: colors(Math.random( ) * 1),
          pink: '#0fa0fa',
          gt500: colors(Math.random( ) * 1)
        },
        projectionConfig: {
          rotation: [97,-30]
        },
        data: {
          'USA': {fillKey: 'lt50' },
          'MEX': {fillKey: 'lt25' },
          'CAN': {fillKey: 'gt50' },
          'GTM': {fillKey: 'gt500'},
          'HND': {fillKey: 'eq50' },
          'BLZ': {fillKey: 'pink' },
          'GRL': {fillKey: 'eq0' },
          'CAN': {fillKey: 'gt50' }
        }
      });

      map.graticule();

      map.arc([{
        origin: {
          latitude: 61,
          longitude: -149
        },
        destination: {
          latitude: -22,
          longitude: -43
        }
      }], {
        greatArc: true,
        animationSpeed: 2000
      });
```

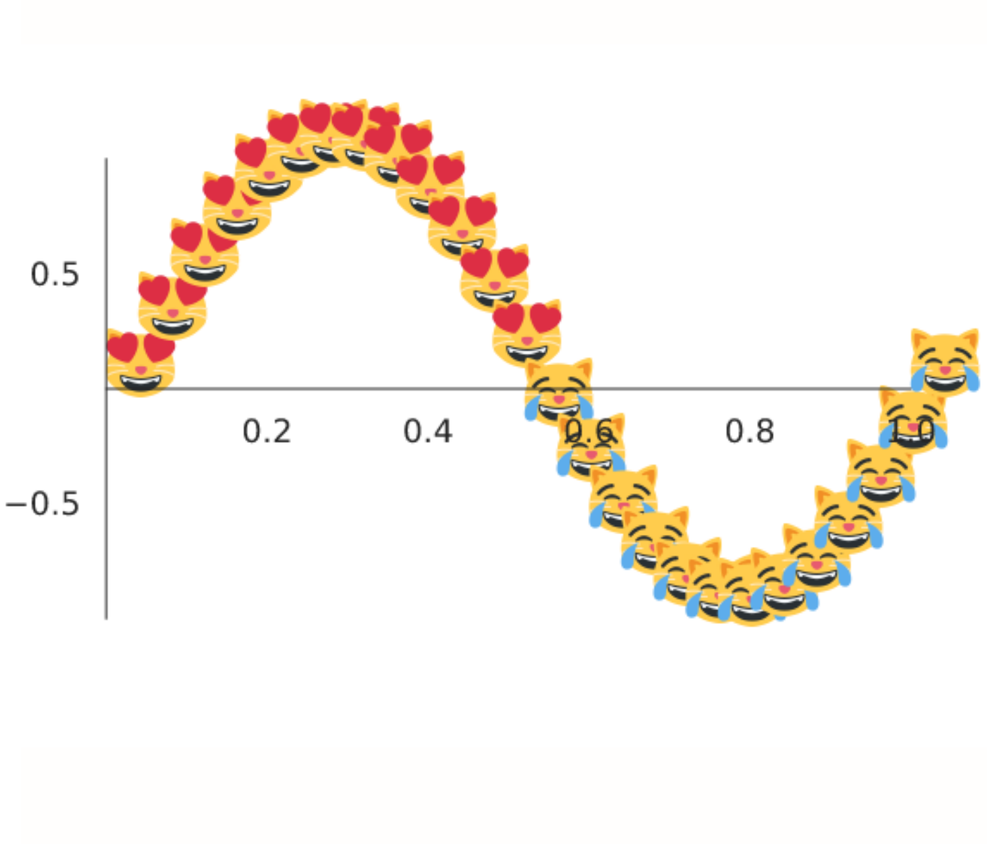
Komentarz

Biblioteka z backendem D3.js do wizulizacji danych geograficznych.

JS

Victory

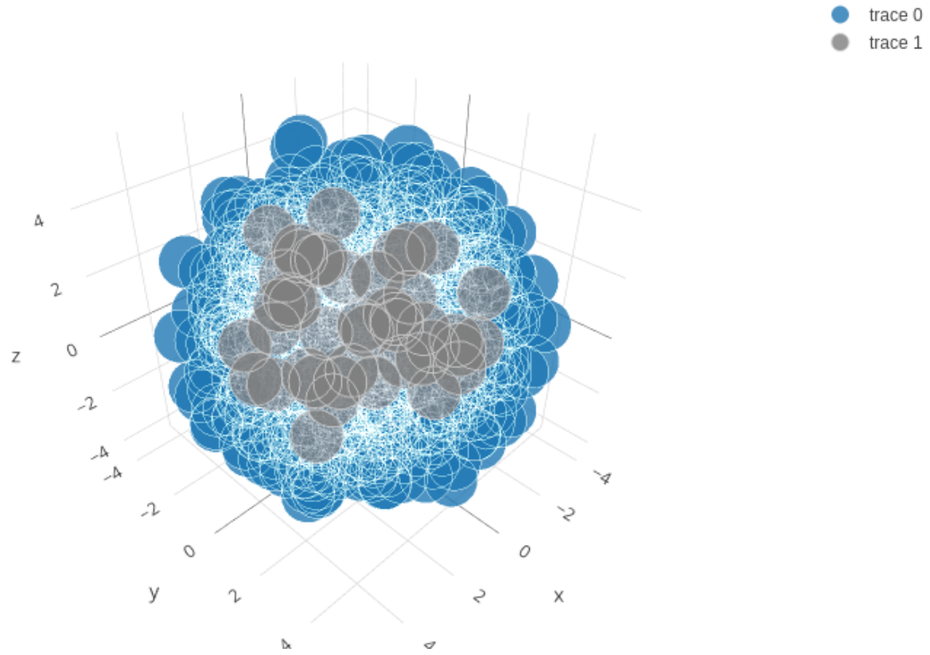
```
class CatPoint extends React.Component {
  render() {
    const {x, y, datum} = this.props;
    const cat = datum._y >= 0 ? "😻" : "😹";
    return (
      <text x={x} y={y} fontSize={30}>
        {cat}
      </text>
    );
  }
}

class App extends React.Component {
  render() {
    return (
      <VictoryChart>
        <VictoryScatter
          y={(d) =>
            Math.sin(2 * Math.PI * d.x)
          }
          samples={25}
          dataComponent={<CatPoint/>}
        />
      </VictoryChart>
    );
  }
}
ReactDOM.render(<App/>, mountNode);
```



Komentarz

Biblioteka do wizualizacji danych, oparta na komponentach React.

plotly | Graphing Libraries




```
d3.csv('https://raw.githubusercontent.com/plotly/datasets/master/3d-scatter.csv', function(err, rows){
function unpack(rows, key) {
    return rows.map(function(row)
    { return row[key]; });}

var trace1 = {
    x:unpack(rows, 'x1'), y: unpack(rows, 'y1'), z: unpack(rows, 'z1'),
    mode: 'markers',
    marker: {
        size: 12,
        line: {
        color: 'rgba(217, 217, 217, 0.14)',
        width: 0.5},
        opacity: 0.8},
    type: 'scatter3d'
};

var trace2 = {
    x:unpack(rows, 'x2'), y: unpack(rows, 'y2'), z: unpack(rows, 'z2'),
    mode: 'markers',
    marker: {
        color: 'rgb(127, 127, 127)',
        size: 12,
        symbol: 'circle',
        line: {
        color: 'rgb(204, 204, 204)',
        width: 1},
        opacity: 0.8},
    type: 'scatter3d'};

var data = [trace1, trace2];
var layout = {margin: {
    l: 0,
    r: 0,
    b: 0,
    t: 0
  }};
Plotly.newPlot('myDiv', data, layout);
});
```

Komentarz

Jedna z najważniejszych bibliotek napędzających cały ekosystem Data Science przez interaktywne wykresy.

Narzędzia, których nie wymieniliśmy ⚠

Komentarz

W sieci można znaleźć również inne paczki, których ze względu bezpieczeństwa nie wymieniliśmy w prezentacji. Są to paczki utrzymywane w Chinach, głównie przez firmę Alibaba:

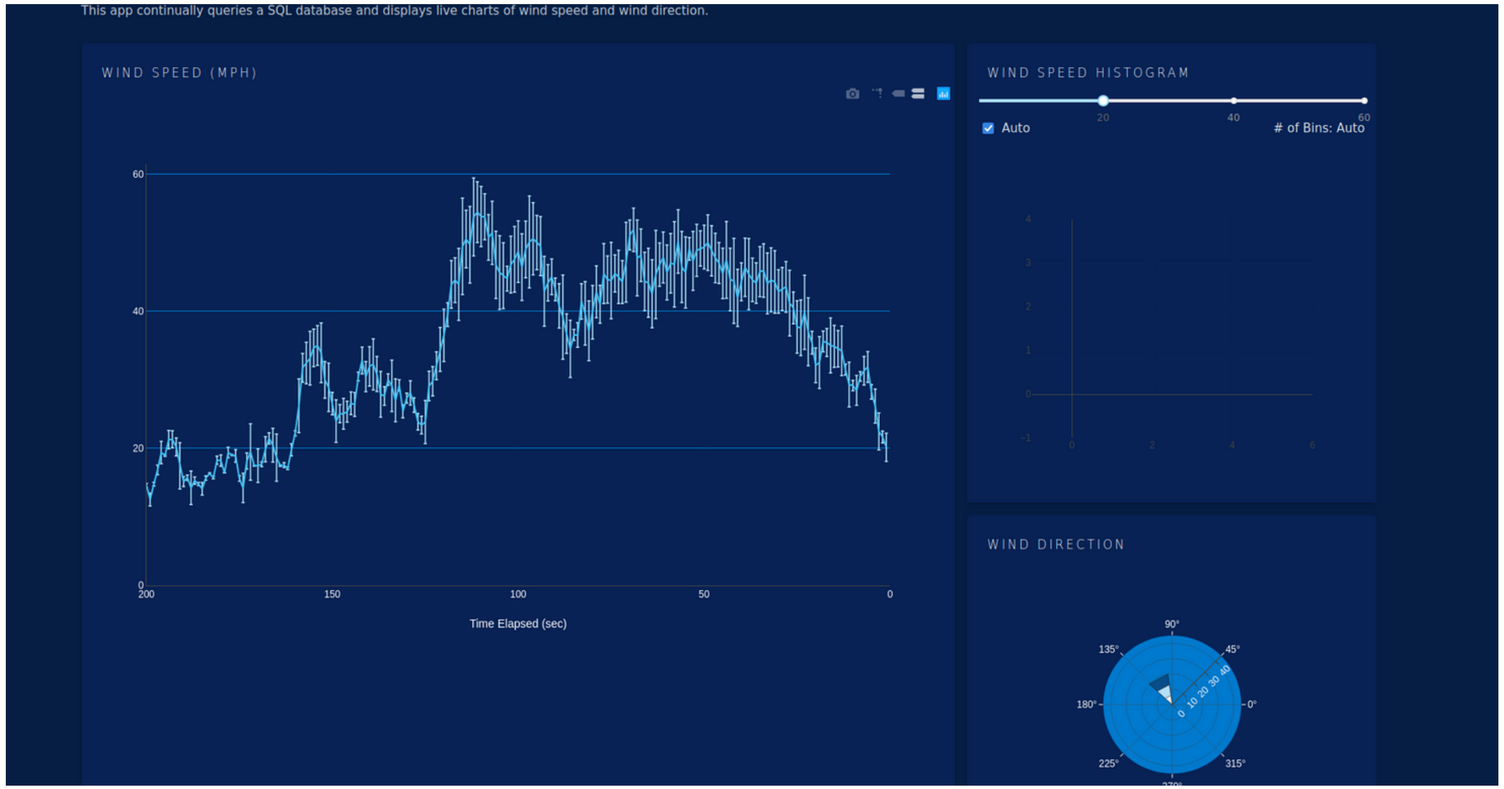
- G2,
- BizCharts,
- ReCharts.

Pozostałe

(O czym sobie nie powiemy?)

O wszystkich programach do projektowania grafiki w wersji Desktop jak i webowej!

DASH (Plotly)

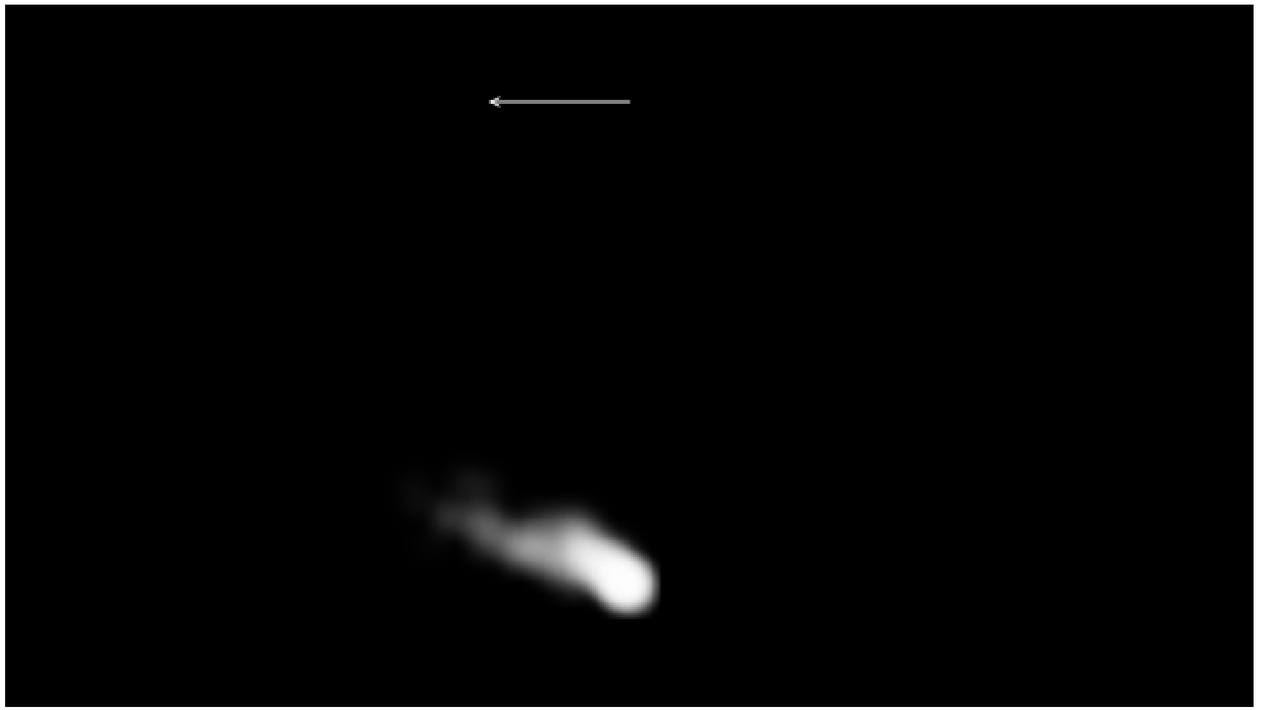


Komentarz

Dash pozwala na tworzenie aplikacji opartych na danych, które można następnie publikować w sieci.

Processing

```
/**
 * Smoke Particle System
 * by Daniel Shiffman.
 *
 * A basic smoke effect using a particle system. Each particle
 * is rendered as an alpha masked image.
 */

ParticleSystem ps;

void setup() {
  size(640, 360);
  PImage img = loadImage("texture.png");
  ps = new ParticleSystem(0, new PVector(width/2, height-60), img);
}

void draw() {
  background(0);

  // Calculate a "wind" force based on mouse horizontal position
  float dx = map(mouseX, 0, width, -0.2, 0.2);
  PVector wind = new PVector(dx, 0);
  ps.applyForce(wind);
  ps.run();
  for (int i = 0; i < 2; i++) {
    ps.addParticle();
  }

  // Draw an arrow representing the wind force
  drawVector(wind, new PVector(width/2, 50, 0), 500);
}

// Renders a vector object 'v' as an arrow and a position 'loc'
void drawVector(PVector v, PVector loc, float scayl) {
  pushMatrix();
  float arrowsize = 4;
  // Translate to position to render vector
  translate(loc.x, loc.y);
  stroke(255);
  // Call vector heading function to get direction (note that pointing up is a heading of 0) and rotate
  rotate(v.heading());
  // Calculate length of vector & scale it to be bigger or smaller if necessary
  float len = v.mag()*scayl;
  // Draw three lines to make an arrow (draw pointing up since we've rotate to the proper direction)
  line(0, 0, len, 0);
  line(len, 0, len-arrowsize, +arrowsize/2)
  line(len, 0, len-arrowsize, -arrowsize/2);
  popMatrix();
}
```



Komentarz

Potężne narzedzie służące wizualizacji danych, nauce programowanie, sztuce, a nawet interakcji ze światem fizycznym!

Musisz spróbować!

ttps://processing.org/examples/smokeparticlesystem.htm

Pytania?

Dziękuję!

Kontakt

TT: twitter.com/**SimonMolinsky**
Li: www.linkedin.com/in/**szymonmolinskipl**
Github: https://github.com/**SimonMolinsky**