

data-vis\helper-functions.js

```
1 // -----
2 // Data processing helper functions.
3 // -----
4 function sum(data) {
5     var total = 0;
6
7     // Ensure that data contains numbers and not strings.
8     data = stringsToNumbers(data);
9
10    for (let i = 0; i < data.length; i++) {
11        total = total + data[i];
12    }
13
14    return total;
15 }
16
17 function mean(data) {
18     var total = sum(data);
19
20     return total / data.length;
21 }
22
23 function sliceRowNumbers (row, start=0, end) {
24     var rowData = [];
25
26     if (!end) {
27         // Parse all values until the end of the row.
28         end = row.arr.length;
29     }
30
31     for (i = start; i < end; i++) {
32         rowData.push(row.getNum(i));
33     }
34
35     return rowData;
36 }
37
38 function stringsToNumbers (array) {
39     return array.map(Number);
40 }
41
42 // -----
43 // Plotting helper functions
44 // -----
45
46 //----- START NEW CODE -----
47 -//
48 function drawAxis(layout, colour=0, threeAxis=false) {
49     stroke(color(colour));
50     strokeWeight(1);
51
52     if (threeAxis) {
```

```
52 // x-axis
53 line(layout.leftMargin,
54     layout.bottomMargin,
55     layout.rightMargin,
56     layout.bottomMargin);
57
58 // y-axis
59 line(layout.leftMargin,
60     layout.topMargin,
61     layout.leftMargin,
62     layout.bottomMargin);
63
64 // y2-axis
65 line(layout.rightMargin,
66     layout.topMargin,
67     layout.rightMargin,
68     layout.bottomMargin);
69
70 //----- END NEW CODE -----//
71
72 }
73 else {
74     // x-axis
75     line(layout.leftMargin,
76         layout.bottomMargin,
77         layout.rightMargin,
78         layout.bottomMargin);
79
80     // y-axis
81     line(layout.leftMargin,
82         layout.topMargin,
83         layout.leftMargin,
84         layout.bottomMargin);
85 }
86 }
87
88 function drawAxisLabels(xLabel, y1Label, y2Label="", layout) {
89     fill(0);
90     noStroke();
91     textAlign('center', 'center');
92     textFont(robotoFont);
93     textSize(16);
94
95     // Draw x-axis label.
96     text(xLabel,
97         (layout.plotWidth() / 2) + layout.leftMargin,
98         layout.bottomMargin + (layout.marginSize * 0.25));
99
100    // Draw y-axis label.
101    push();
102    translate(layout.leftMargin - (layout.marginSize * 0.3),
103        layout.bottomMargin / 1.45);
104    rotate(- PI / 2);
105    text(y1Label, 0, 0);
```

```
106     pop();
107
108     if (y2Label !== "") {
109         // Draw y2-axis label.
110         push();
111         translate(layout.rightMargin - (layout.marginSize * 0.3),
112                 layout.bottomMargin / 1.45);
113         rotate(- PI / 2);
114         text(y2Label, 0, 80);
115         pop();
116     }
117 }
118
119 function drawYAxisTickLabels(min, max, layout, mapFunction,
120                             decimalPlaces) {
121     // Map function must be passed with .bind(this).
122     var range = max - min;
123     var yTickStep = range / layout.numYTickLabels;
124
125     fill(0);
126     noStroke();
127     textAlign('right', 'center');
128     textFont(robotoFont);
129     textSize(16);
130
131     // Draw all axis tick labels and grid lines.
132     for (i = 0; i <= layout.numYTickLabels; i++) {
133         var value = min + (i * yTickStep);
134         var y = mapFunction(value);
135
136         // Add tick label.
137         text(value.toFixed(decimalPlaces),
138              layout.leftMargin - layout.pad,
139              y);
140
141         if (layout.grid) {
142             // Add grid line.
143             stroke(200);
144             strokeWeight(1);
145             line(layout.leftMargin, y, layout.rightMargin, y);
146         }
147     }
148 }
149
150 //----- START NEW CODE -----
151 -//
152 function drawY2AxisTickLabels(min, max, layout, mapFunction,
153                               decimalPlaces) {
154     // Map function must be passed with .bind(this).
155     var range = max - min;
156     var yTickStep = range / layout.numYTickLabels;
157
158     fill(0);
159     noStroke();
```

```
159     textAlign('right', 'center');
160     textFont(robotoFont);
161     textSize(16);
162
163     // Draw all axis tick labels
164     for (i = 0; i <= layout.numYTickLabels; i++) {
165         var value = min + (i * yTickStep);
166         var y = mapFunction(value);
167
168         // Add tick label.
169         text(value.toFixed(decimalPlaces),
170             layout.rightMargin + 4 * layout.pad,
171             y);
172     }
173 }
174 //----- END NEW CODE -----//
175
176 function drawXAxisTickLabel(value, layout, mapFunction) {
177     // Map function must be passed with .bind(this).
178     var x = mapFunction(value);
179
180     fill(0);
181     noStroke();
182     textAlign('center', 'center');
183     textFont(robotoFont);
184
185     // Add tick label.
186     text(value,
187         x,
188         layout.bottomMargin + layout.marginSize / 8);
189
190     if (layout.grid) {
191         // Add grid line.
192         stroke(220);
193         strokeWeight(1);
194         line(x,
195             layout.topMargin,
196             x,
197             layout.bottomMargin);
198     }
199 }
200
201 //----- START NEW CODE -----
202 -//
203 function draw2Legend(xPos, yPos, labelText1, labelText2) {
204     textFont(robotoFont);
205     textAlign(LEFT, CENTER);
206     textSize(16);
207     noFill();
208     noStroke();
209
210     fill('#C8102E');
211     rect(xPos, yPos, 15, 15, 3);
212     fill(0);
```

```
212     text(labelText1, xPos + 25, yPos + 6);
213
214     fill('#002147');
215     rect(xPos + 120, yPos, 15, 15, 3);
216     fill(0);
217     text(labelText2, xPos + 145, yPos + 6);
218 }
219
220 function drawLegend(xPos, yPos, labels) {
221     // Space per block
222     let spacing = 145;
223     let boxSize = 15;
224
225     // Font
226     textFont(robotoFont);
227     textAlign(LEFT, CENTER);
228     textSize(16);
229     noStroke();
230
231     for (let i = 0; i < labels.length; i++) {
232         //
233         let currentX = xPos + i * spacing;
234
235         //
236         fill(labels[i].color);
237         rect(currentX, yPos, boxSize, boxSize, 3);
238
239         //
240         fill(0);
241         text(labels[i].text, currentX + boxSize + 10, yPos + boxSize / 2);
242     }
243 }
244 //----- END NEW CODE -----//
245
```