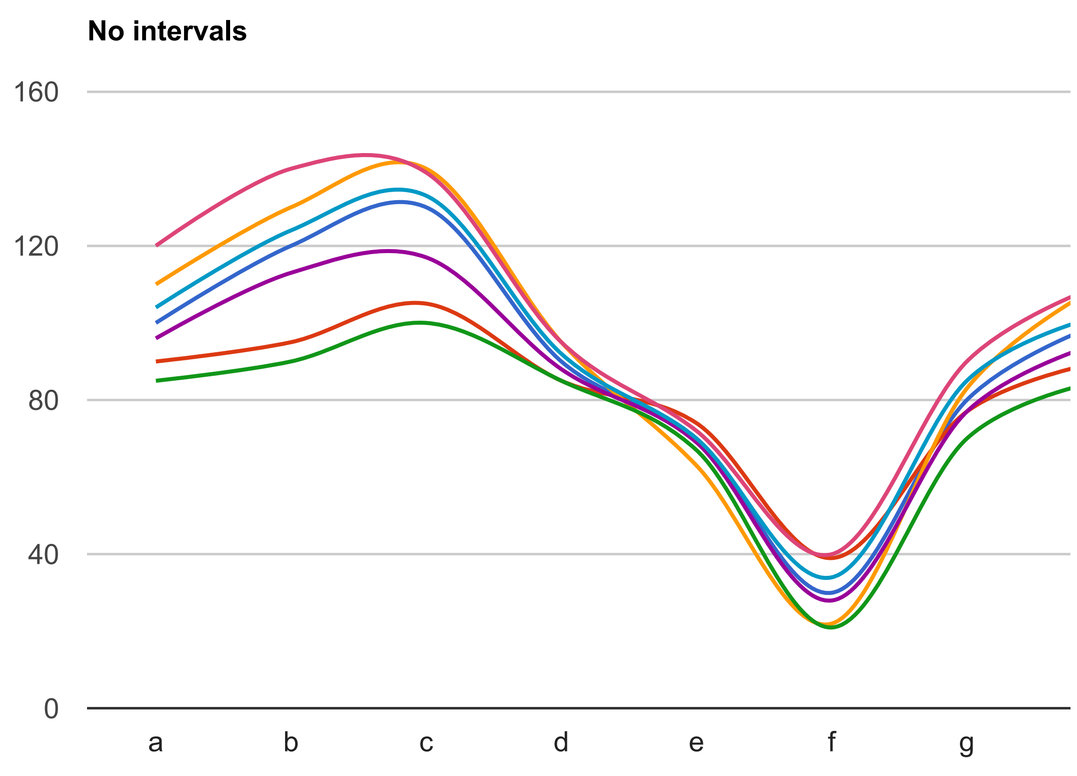
Intervals

Overview

Google Charts can display *intervals* around a series. They might be used to portray confidence intervals, minimum and maximum values around a value, percentile sampling, or anything else that requires a varying margin around a series.

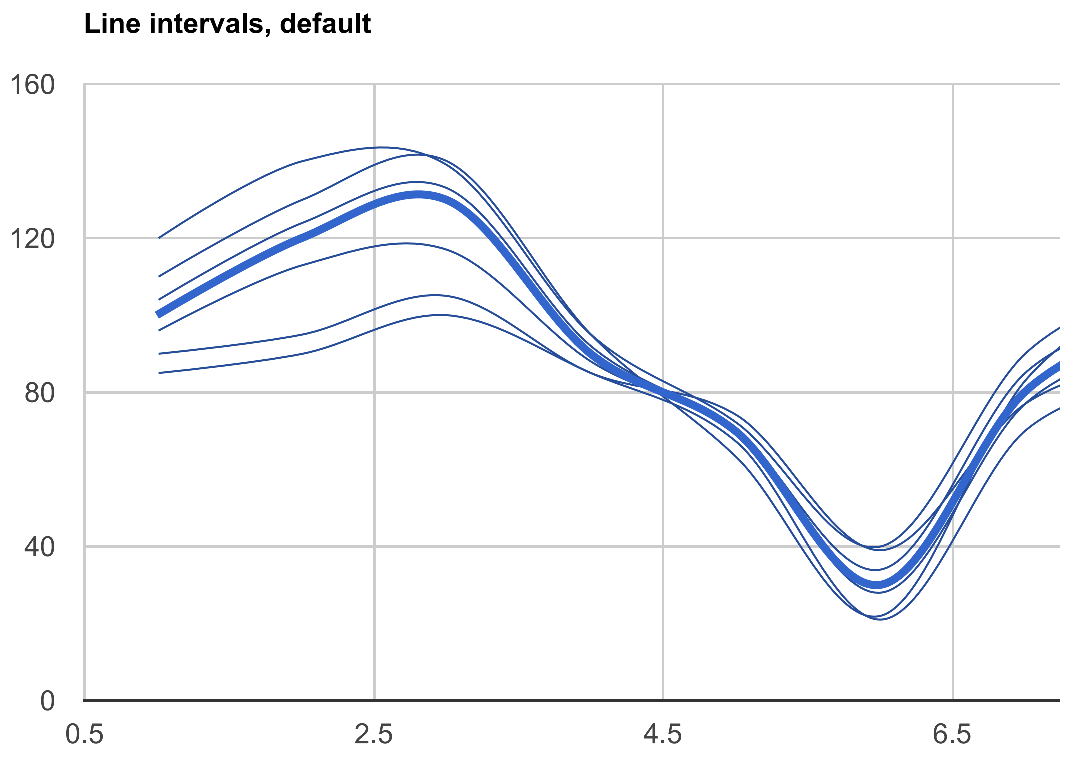
There are six styles of interval: line, bar, box, stick, point, and area. Below, you'll see examples of each. In each example, we'll use the same dataset, here shown without any intervals:



The above chart is simple: it has seven series of data, all equal in importance. In what follows, we'll assume that the first series is the primary series, and the other six are being compared to it via intervals.

Line Intervals

Line intervals are sometimes used to depict the variance of multiple experiments. In the following chart, we illustrate a primary series and intervals around it.

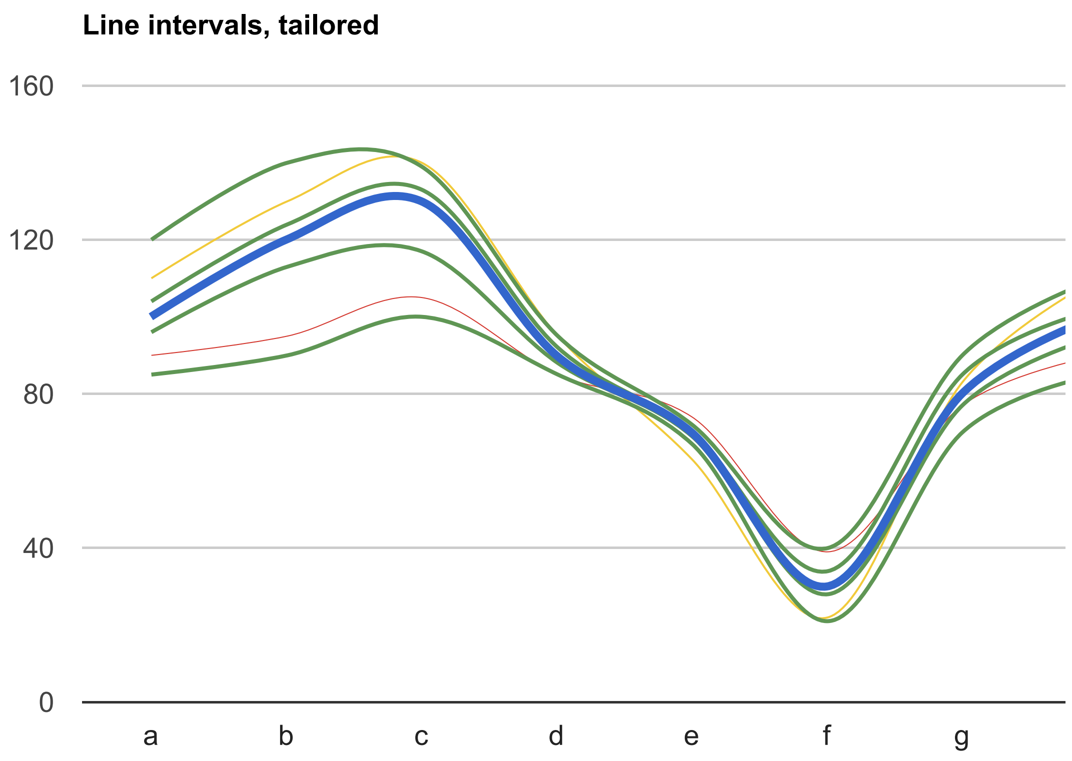


CODE IT YOURSELF ON JSFIDDLE

```
<html>
  <head>
    <script type="text/javascript" src="https://www.gstatic.com/charts/loader
    <script type="text/javascript">
      google.charts.load('current', {'packages':['corechart']});
      google.charts.setOnLoadCallback(drawChart);

      function drawChart() {
        var data = new google.visualization.DataTable();
        data.addColumn('number', 'x');
        data.addColumn('number', 'values');
        data.addColumn({id:'i0', type:'number', role:'interval'});
        data.addColumn({id:'i1', type:'number', role:'interval'});
        data.addColumn({id:'i2', type:'number', role:'interval'});
        data.addColumn({id:'i2', type:'number', role:'interval'});
        data.addColumn({id:'i2', type:'number', role:'interval'});
        data.addColumn({id:'i2', type:'number', role:'interval'});

        data.addRows([
            [1, 100, 90, 110, 85, 96, 104, 120],
            [2, 120, 95, 130, 90, 113, 124, 140],
            [3, 130, 105, 140, 100, 117, 133, 139],
            [4, 90, 85, 95, 85, 88, 92, 95],
            [5, 70, 74, 63, 67, 69, 70, 72],
            [6, 30, 39, 22, 21, 28, 34, 40],
            [7, 80, 77, 83, 70, 77, 85, 90],
            [8, 100, 90, 110, 85, 95, 102, 110]]);

        // The intervals data as narrow lines (useful for showing raw source 
        var options_lines = {
            title: 'Line intervals, default',
            curveType: 'function',
            lineWidth: 4,
            intervals: { 'style':'line' },
            legend: 'none'
        };

        var chart_lines = new google.visualization.LineChart(document.getElem
        chart_lines.draw(data, options_lines);
      }
    </script>
  </head>
  <body>
    <div id="chart_lines" style="width: 900px; height: 500px;"></div>
  </body>
</html>
```

In the data above, you can see that we attached three different identifiers to the supplementary series: `i0`, `i2`, and `i3`. We can use those to style those series differently; below, we give them different colors and thicknesses.

Note: In general, it's poor form to reuse IDs as we have above, with `i2` being used four times. It works, but we may change this behavior in the future.

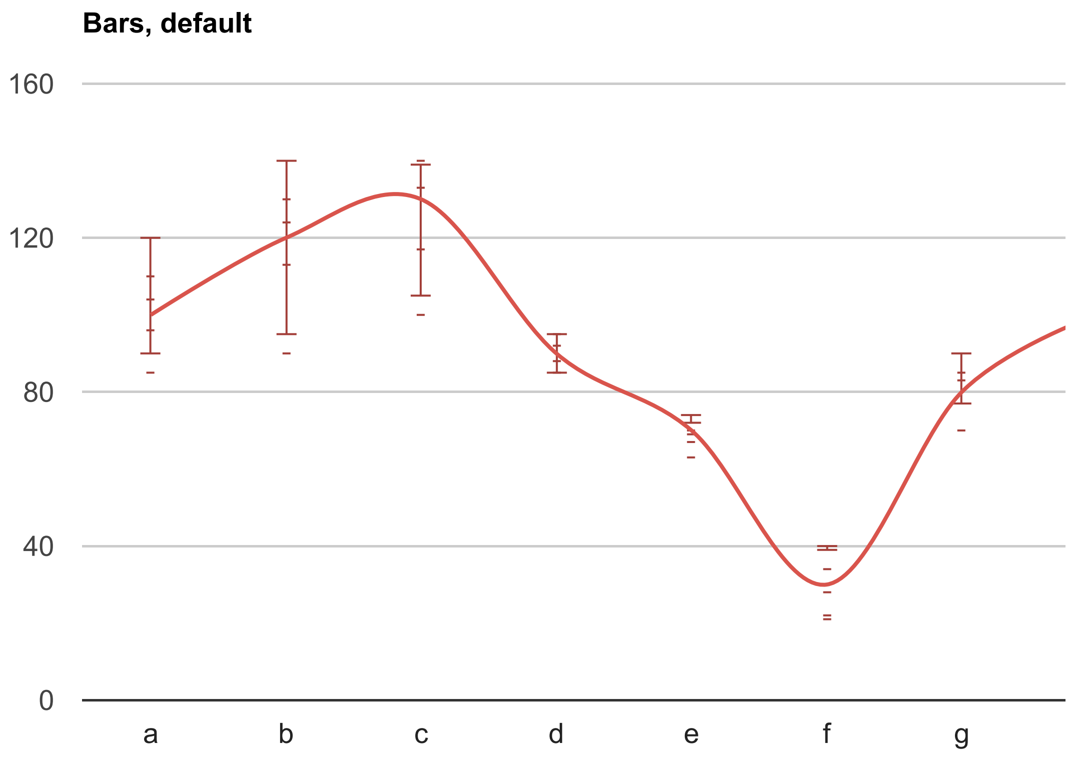


The only difference is in the options:

```
var options_lines = {
    title: 'Line intervals, tailored',
    lineWidth: 4,
    curveType:'function',
    interval: {
        'i0': { 'style':'line', 'color':'#D3362D', 'lineWidth': 0.5 },
        'i1': { 'style':'line', 'color':'#F1CA3A', 'lineWidth': 1 },
        'i2': { 'style':'line', 'color':'#5F9654', 'lineWidth': 2 },
    },
    legend: 'none',
};
```

Bar Intervals

Bar intervals create error bars around your data. The first and last columns of the interval are drawn as wide bars parallel to the domain-axis, and inner columns are drawn as shorter "ticks". A "stick" is added to join the wide bars (if these two bars have the same value then the stick is rendered as a point, unless the `pointSize` option is zero).

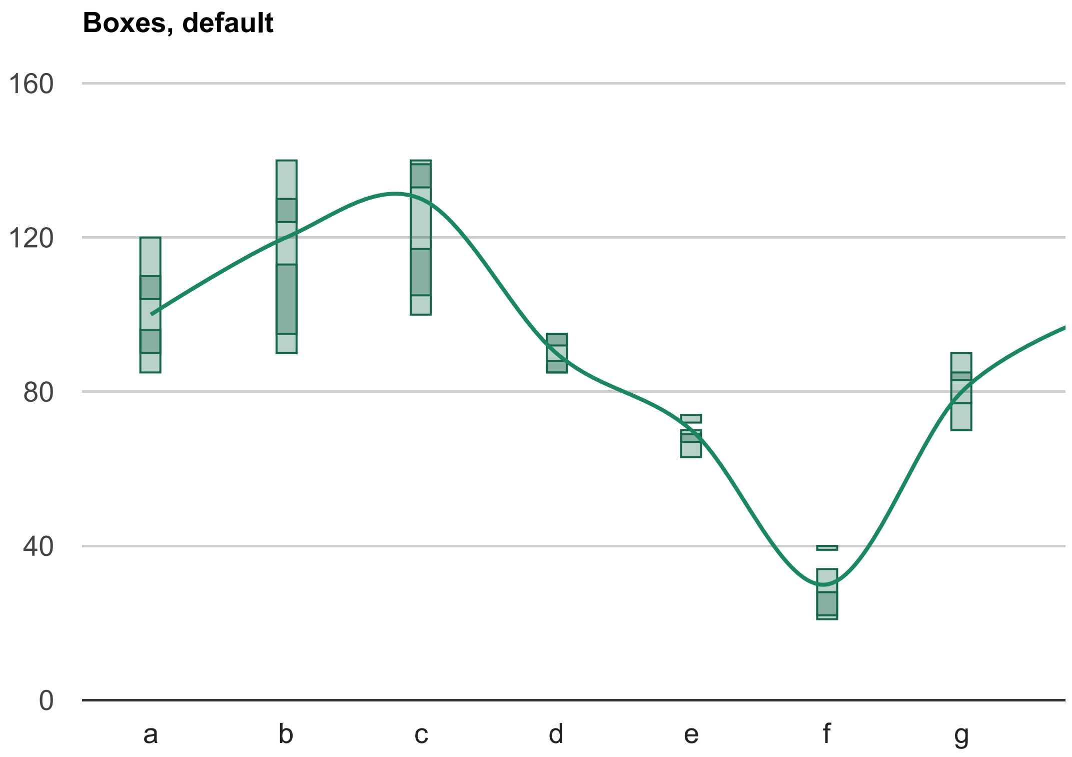


The width of the horizontal bars corresponding to the first and last columns is controlled with `intervals.barWidth`, and the width of the horizontal bars corresponding to the inner columns is controlled with `intervals.shortBarWidth`. When those are omitted, you'll get a chart like the one above with the options below:

```
var options_bars = {
    title: 'Bars, default',
    curveType: 'function',
    series: [{'color': '#D9544C'}],
    intervals: { style: 'bars' },
    legend: 'none',
};
```

Box Intervals

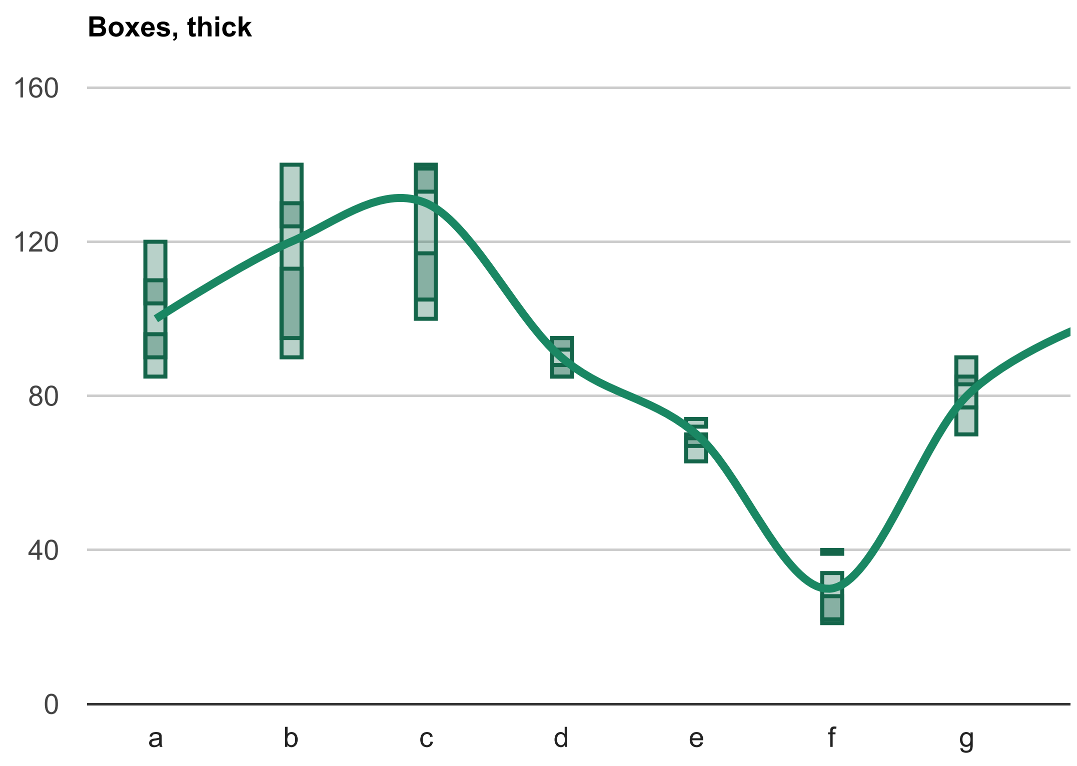
Box intervals rendered columns in your data table as a set of nested rectangles: the first and last columns form the outermost rectangle, and inner columns render as darker rectangles within their containing box.



Here's how to specify box intervals:

```
    var options = {
        series: [{'color': '#1A8763'}],
        intervals: { style: 'boxes' },
        legend: 'none',
    };
```

You can make the boxes more prominent with the `intervals.lineWidth` and `intervals.barWidth` options:

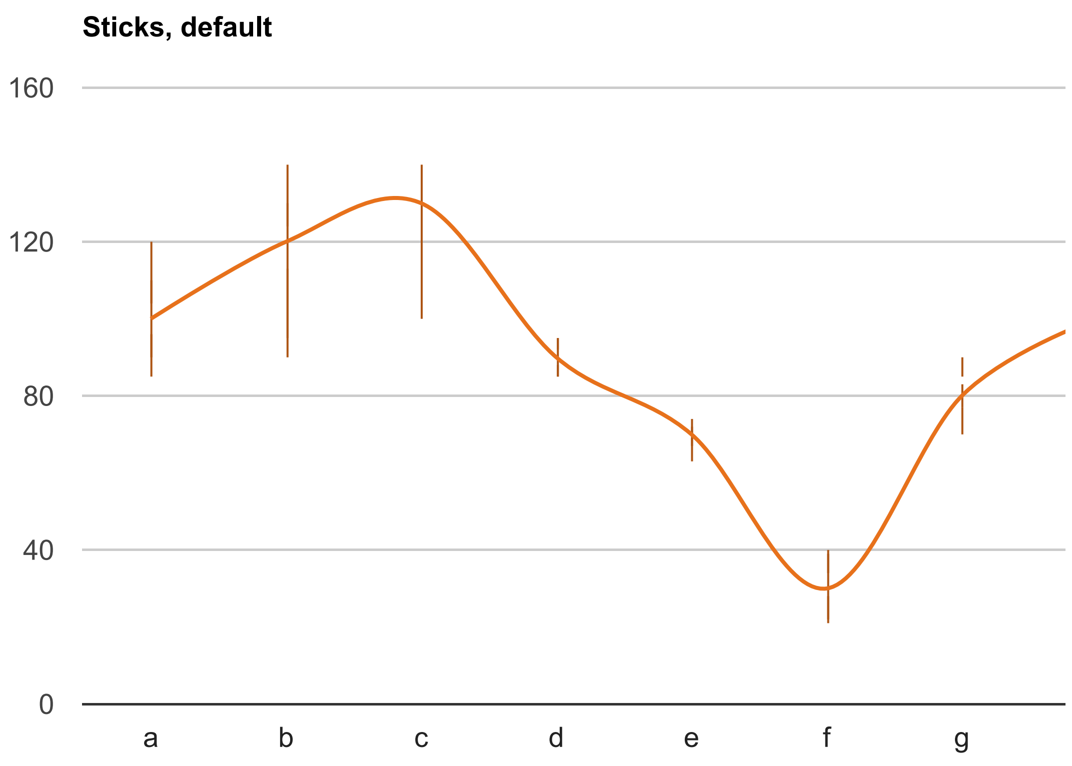


The relevant options:

```
    var options = {
        title:'Boxes, thick',
        curveType:'function',
        lineWidth: 4,
        series: [{'color': '#1A8763'}],
        intervals: { 'lineWidth':2, 'barWidth': 0.5, style: 'boxes' },
        legend: 'none',
    };
```

Stick Intervals

Stick intervals display pairs of columns as a set of sticks parallel to the target axis. A stick of zero height is rendered as a point, which can be suppressed by setting the `pointSize` option to zero.

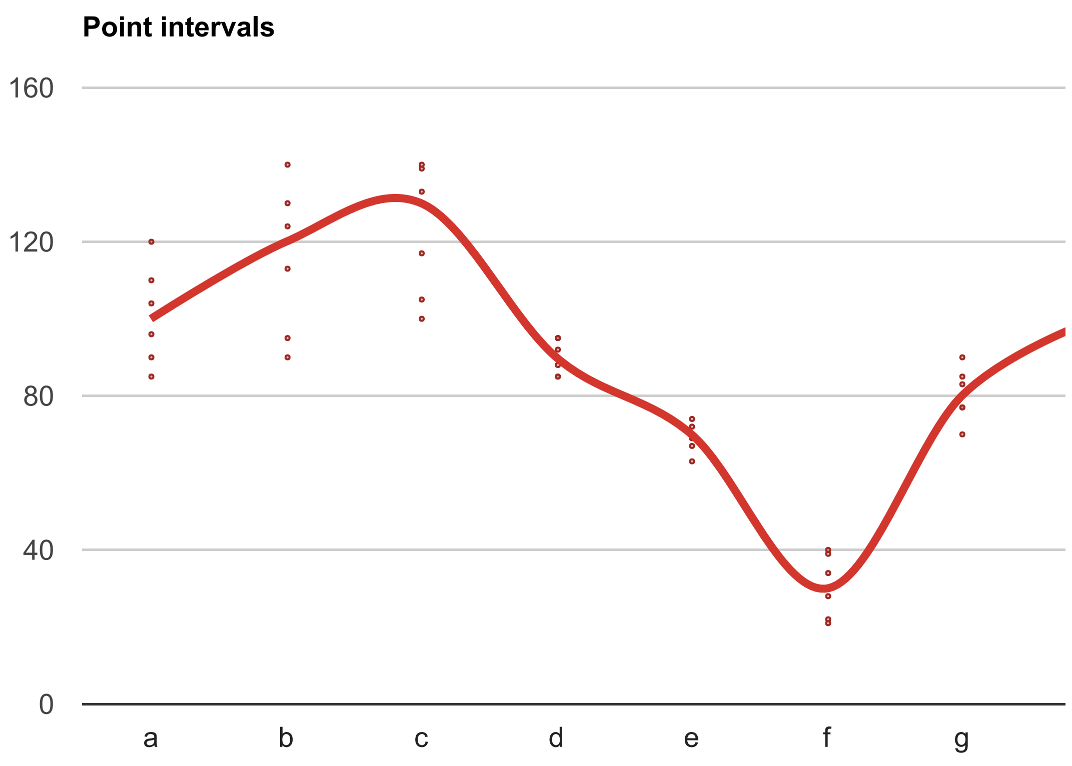


You can create these with a `style` of `'sticks'`:

```
    var options_sticks = {
        title:'Sticks, default',
        curveType:'function',
        series: [{'color': '#E7711B'}],
        intervals: { style: 'sticks' },
        legend: 'none',
    };
```

Point Intervals

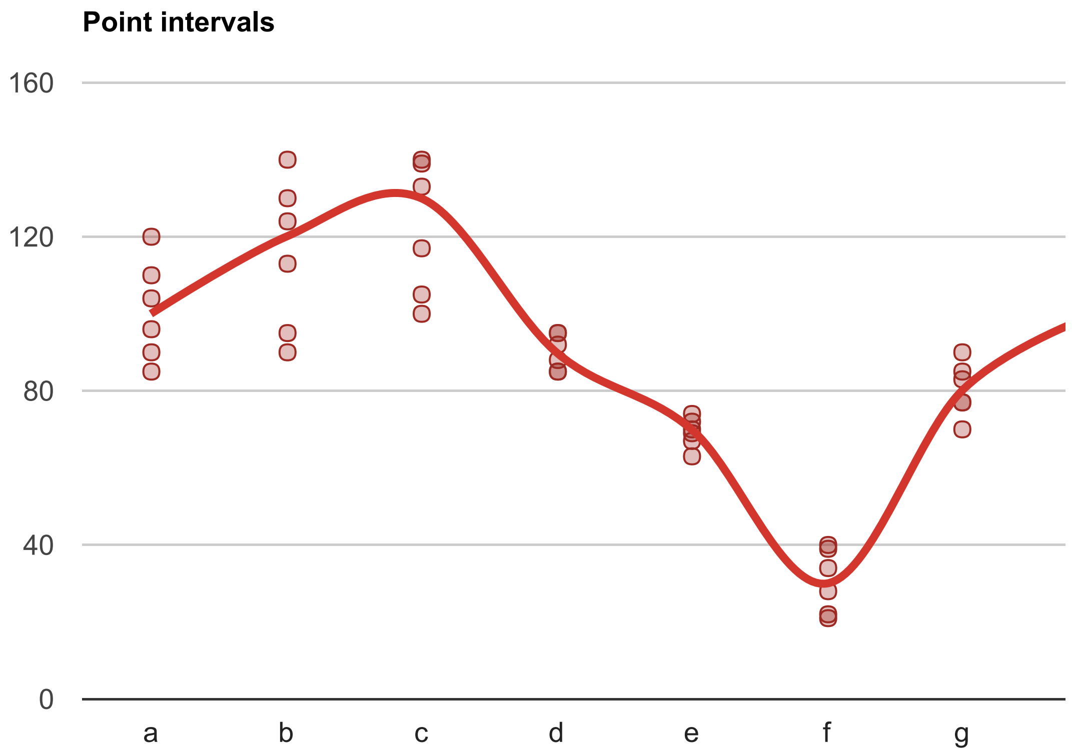
Point intervals display interval data as small circles:



The point size can be controlled with the `intervals.pointSize` option. Here, it's 2:

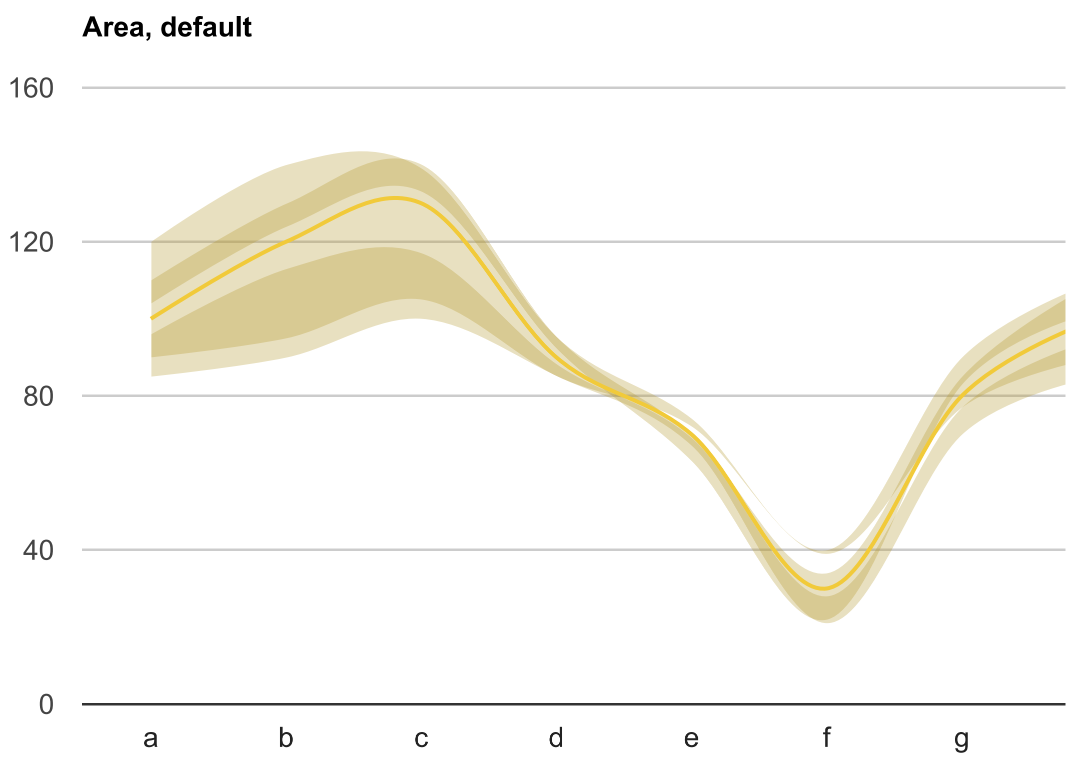
```
    var options_points = {
        title:'Points, default',
        curveType:'function',
        lineWidth: 4,
        series: [{'color': '#D3362D'}],
        intervals: { 'style':'points', pointSize: 2 },
        legend: 'none',
    };
```

Here's what it looks like at 8:



Area Intervals

An *area interval* renders interval data as a set of nested shaded areas. Nesting of pairs of columns is similar to that of box intervals, except that an even number of columns is required.



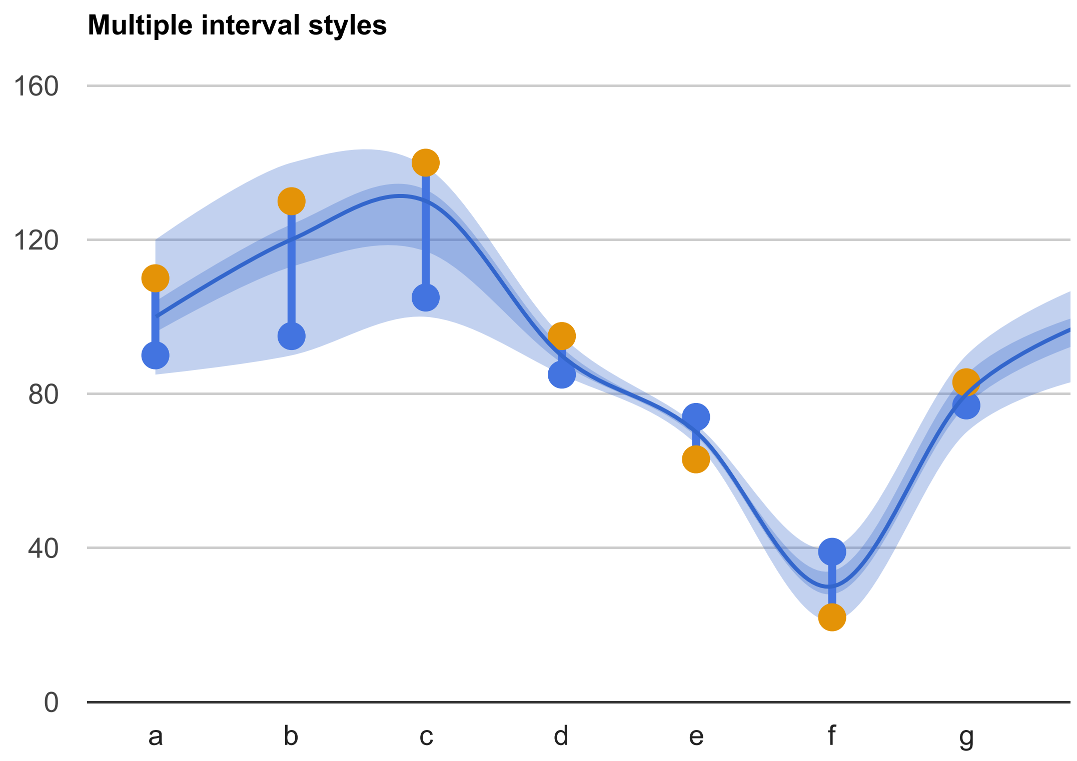
This is accomplished by setting `style` to `'area'`:

```
var options = {
    title:'Area, default',
    curveType:'function',
    series: [{'color': '#F1CA3A'}],
    intervals: { 'style':'area' },
    legend: 'none',
};
```

Combining Interval Styles

For even greater customization, you can combine intervals styles inside one chart.

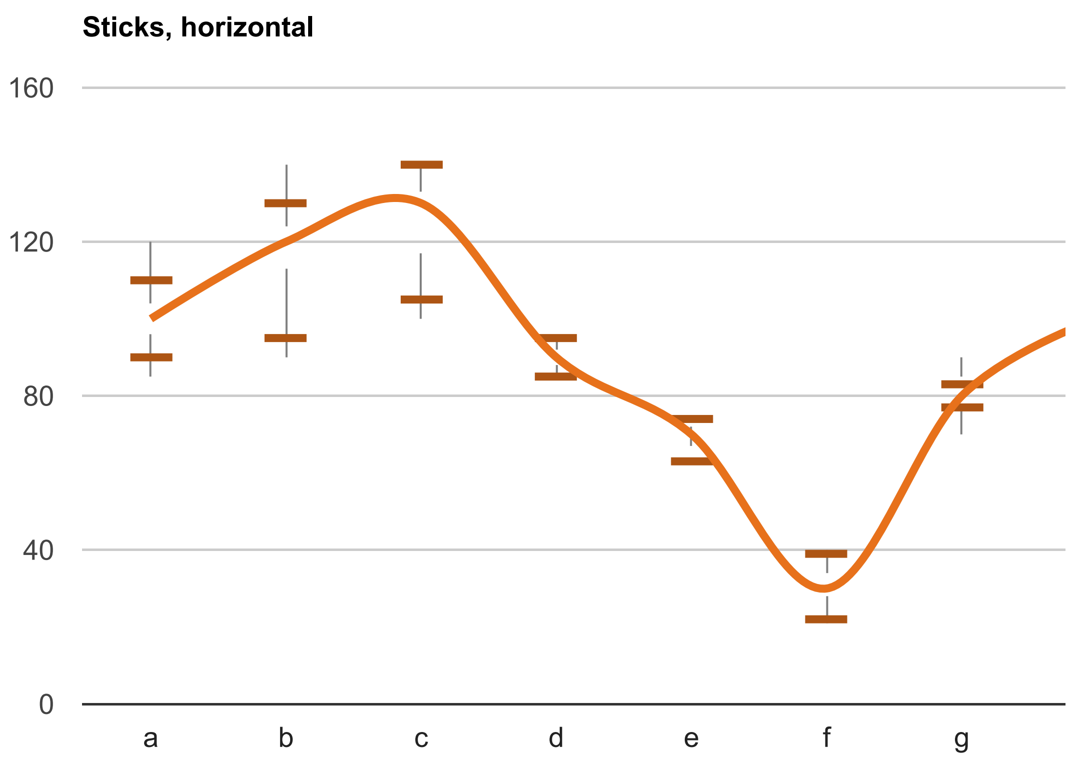
Here's a chart that combines area and bar intervals:



In the above chart, we specify a `style` of `'bars'` for intervals tagged with `i0` and `i1`, and an `'area'` style for `i2`. Then we use `pointSize` to cap the bars:

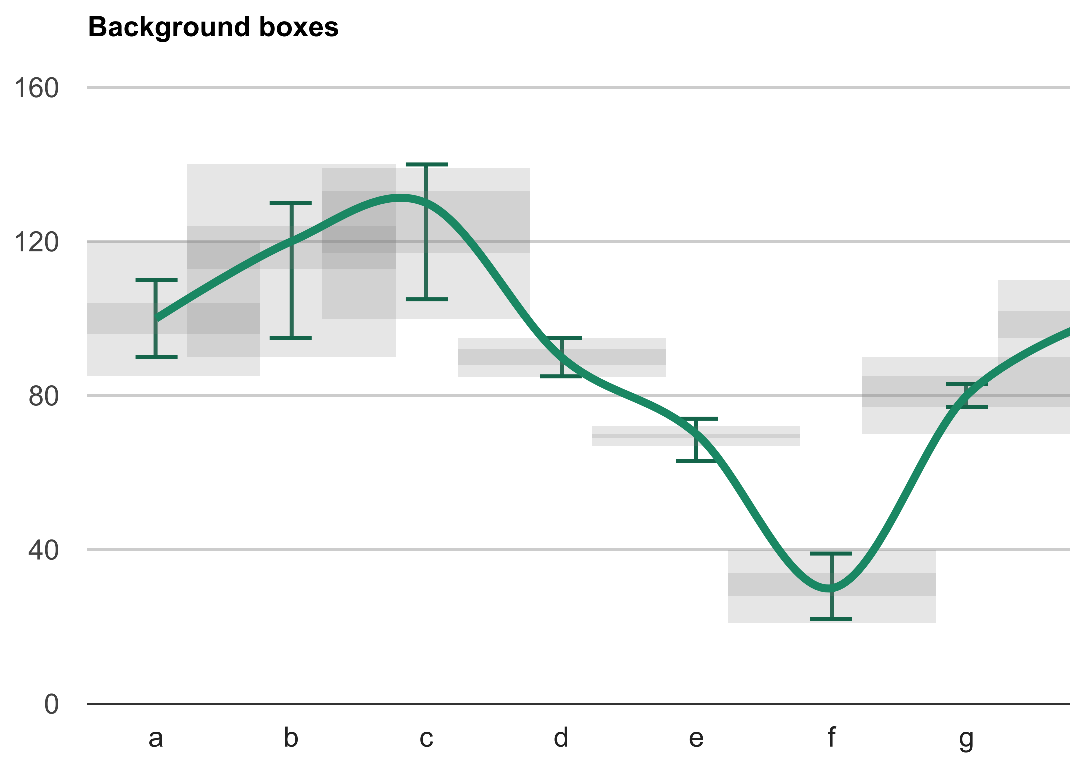
```
    var options = {
        title:'Bar/area interval chart',
        curveType:'function',
        intervals: { 'color':'series-color' },
        interval: {
            'i0': { 'color': '#4374E0', 'style':'bars', 'barWidth':0, 'lineWi
            'i1': { 'color': '#E49307', 'style':'bars', 'barWidth':0, 'lineWi
            'i2': { 'style':'area', 'curveType':'function', 'fillOpacity':0.3
        legend: 'none',
    };
```

Here's a bar interval line chart with the `i2` intervals represented as sticks:



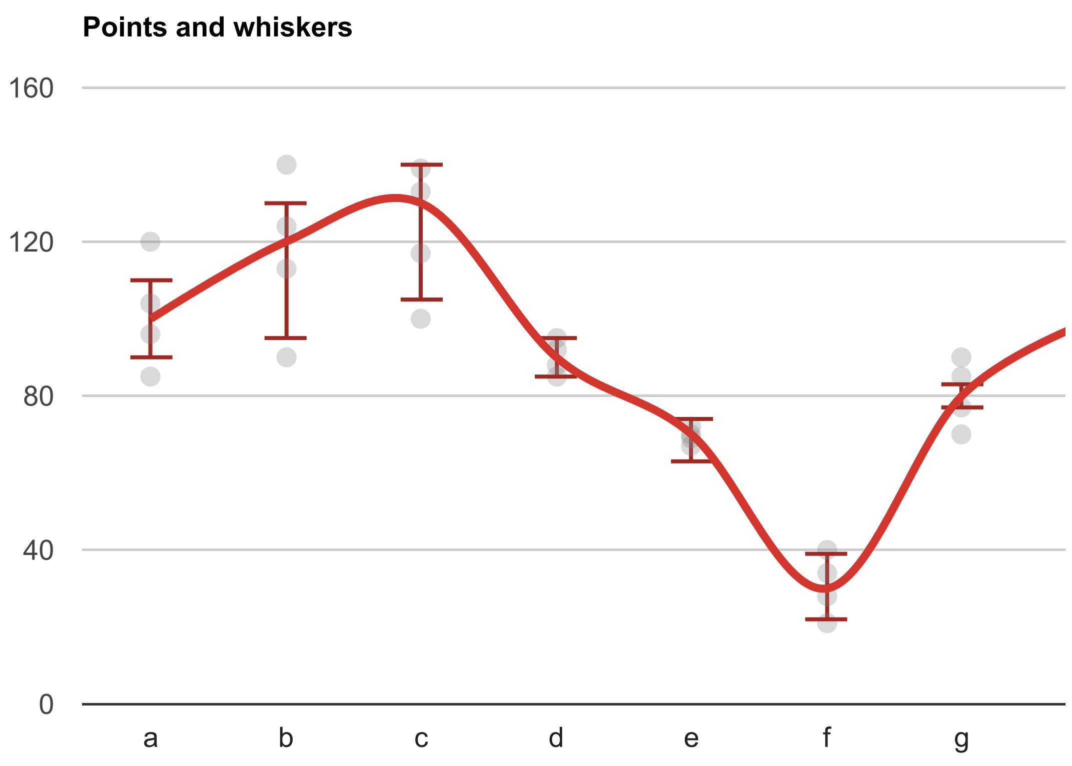
```
var options = {
    title:'Sticks, horizontal',
    curveType:'function',
    lineWidth: 4,
    series: [{'color': '#E7711B'}],
    intervals: { 'lineWidth': 4, 'barWidth': 0.5 },
    interval: {
        'i2': { 'style':'sticks', 'color':'grey', 'boxWidth': 2.5,
        'lineWidth': 1 }
    },
    legend: 'none',
};
```

Here's an interval line chart that uses low opacity boxes to place selected intervals in the background:



```
// Focus is the error bars, but boxes are visible in the background.
var options_boxes_background = {
    title:'Background boxes',
    curveType:'function',
    lineWidth: 4,
    series: [{'color': '#1A8763'}],
    intervals: { 'lineWidth':2, 'barWidth': 0.5 },
    interval: {
        'i2': { 'style':'boxes', 'color':'grey', 'boxWidth': 2.5,
        'lineWidth': 0, 'fillOpacity': 0.2 }
    },
    legend: 'none',
};
```

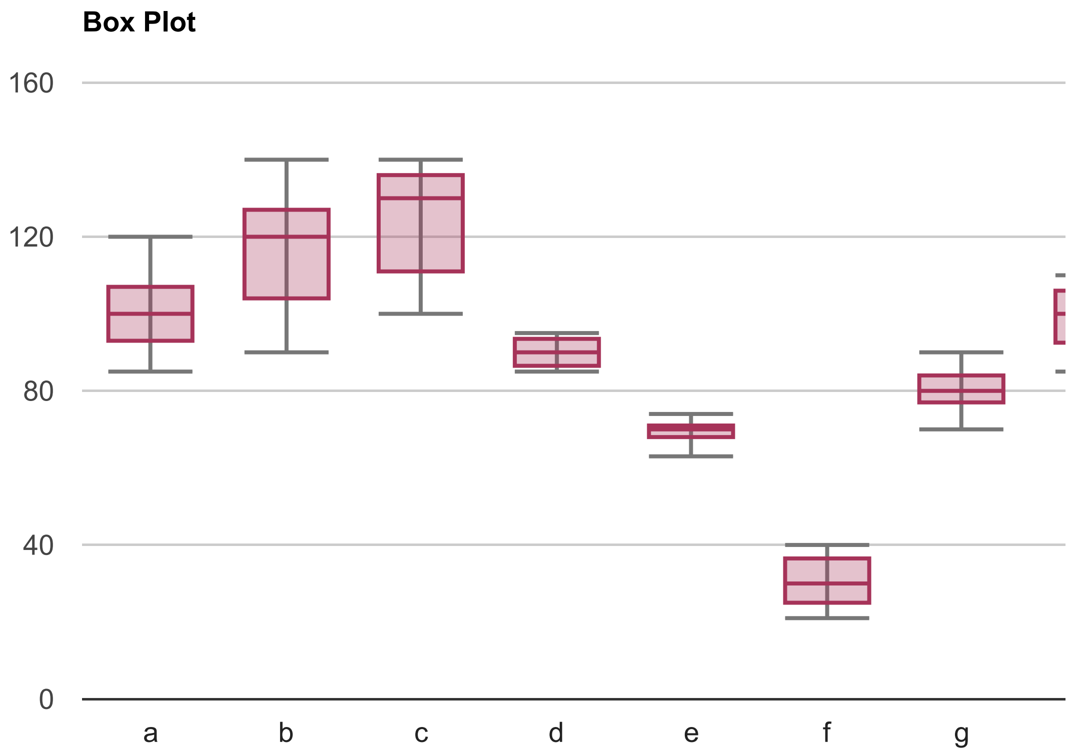
We can create a "points and whiskers" interval chart by specifying a low opacity `'points'` style for one interval along with a `boxWidth`:



```
var options = {
    title:'Points and whiskers',
    curveType:'function',
    lineWidth: 4,
    series: [{'color': '#D3362D'}],
    intervals: { 'lineWidth':2, 'barWidth': 0.5 },
    interval: {
        'i2': { 'style':'points', 'color':'grey', 'pointSize': 10,
        'lineWidth': 0, 'fillOpacity': 0.3 }
    },
    legend: 'none',
};
```

Box Plot

Finally, based on the above "points and whiskers" chart, we can use box and bar intervals to create a basic box plot chart.



CODE IT YOURSELF ON JSFIDDLE

OPTIONS FULL SCRIPT BODY

```
  google.charts.load('current', {'packages':['corechart']});
  google.charts.setOnLoadCallback(drawBoxPlot);

  function drawBoxPlot() {

    var array = [
      ['a', 100, 90, 110, 85, 96, 104, 120],
      ['b', 120, 95, 130, 90, 113, 124, 140],
      ['c', 130, 105, 140, 100, 117, 133, 139],
      ['d', 90, 85, 95, 85, 88, 92, 95],
      ['e', 70, 74, 63, 67, 69, 70, 72],
      ['f', 30, 39, 22, 21, 28, 34, 40],
      ['g', 80, 77, 83, 70, 77, 85, 90],
      ['h', 100, 90, 110, 85, 95, 102, 110]
    ];

    var data = new google.visualization.DataTable();
    data.addColumn('string', 'x');
```

```
      data.addColumn('number', 'series0');
      data.addColumn('number', 'series1');
      data.addColumn('number', 'series2');
      data.addColumn('number', 'series3');
      data.addColumn('number', 'series4');
      data.addColumn('number', 'series5');
      data.addColumn('number', 'series6');

      data.addColumn({id:'max', type:'number', role:'interval'});
      data.addColumn({id:'min', type:'number', role:'interval'});
      data.addColumn({id:'firstQuartile', type:'number', role:'interval'});
      data.addColumn({id:'median', type:'number', role:'interval'});
      data.addColumn({id:'thirdQuartile', type:'number', role:'interval'});

      data.addRows(getBoxPlotValues(array));

      /**
       * Takes an array of input data and returns an
       * array of the input data with the box plot
       * interval data appended to each row.
       */
      function getBoxPlotValues(array) {

        for (var i = 0; i < array.length; i++) {

          var arr = array[i].slice(1).sort(function (a, b) {
            return a - b;
          });

          var max = arr[arr.length - 1];
          var min = arr[0];
          var median = getMedian(arr);

          // First Quartile is the median from lowest to overall median.
          var firstQuartile = getMedian(arr.slice(0, 4));

          // Third Quartile is the median from the overall median to the hig
          var thirdQuartile = getMedian(arr.slice(3));

          array[i][8] = max;
          array[i][9] = min
          array[i][10] = firstQuartile;
          array[i][11] = median;
          array[i][12] = thirdQuartile;
        }
        return array;
      }
```

```
/*
 * Takes an array and returns
 * the median value.
 */
function getMedian(array) {
  var length = array.length;

  /* If the array is an even length the
   * median is the average of the two
   * middle-most values. Otherwise the
   * median is the middle-most value.
   */
  if (length % 2 === 0) {
    var midUpper = length / 2;
    var midLower = midUpper - 1;

    return (array[midUpper] + array[midLower]) / 2;
  } else {
    return array[Math.floor(length / 2)];
  }
}

var options = {
    title:'Box Plot',
    height: 500,
    legend: {position: 'none'},
    hAxis: {
      gridlines: {color: '#fff'}
    },
    lineWidth: 0,
    series: [{'color': '#D3362D'}],
    intervals: {
      barWidth: 1,
      boxWidth: 1,
      lineWidth: 2,
      style: 'boxes'
    },
    interval: {
      max: {
        style: 'bars',
        fillOpacity: 1,
        color: '#777'
      },
      min: {
        style: 'bars',
        fillOpacity: 1,
        color: '#777'
      }
```

```
        }
      };

      var chart = new google.visualization.LineChart(document.getElementById

      chart.draw(data, options);
    }
```