Apex Resultsviewer

Javascript notation

Most programming languages

```
int myVariable = 5;
string myVariable = “blabla”;
ObjectType myVariable = new ObjectType();
```

Javascript

```
var myVariable = whateveryouwant;
```

```
myVariable = Object;
myVariable.myNumber = 5;
myVariable[“myNumber”] = 5;
myVariable.myString = “bla”;
myVariable.myArray = [];
myVariable.myArray[0] = “a”;
myVariable.myArray[1] = “b”;
```

$\longrightarrow$

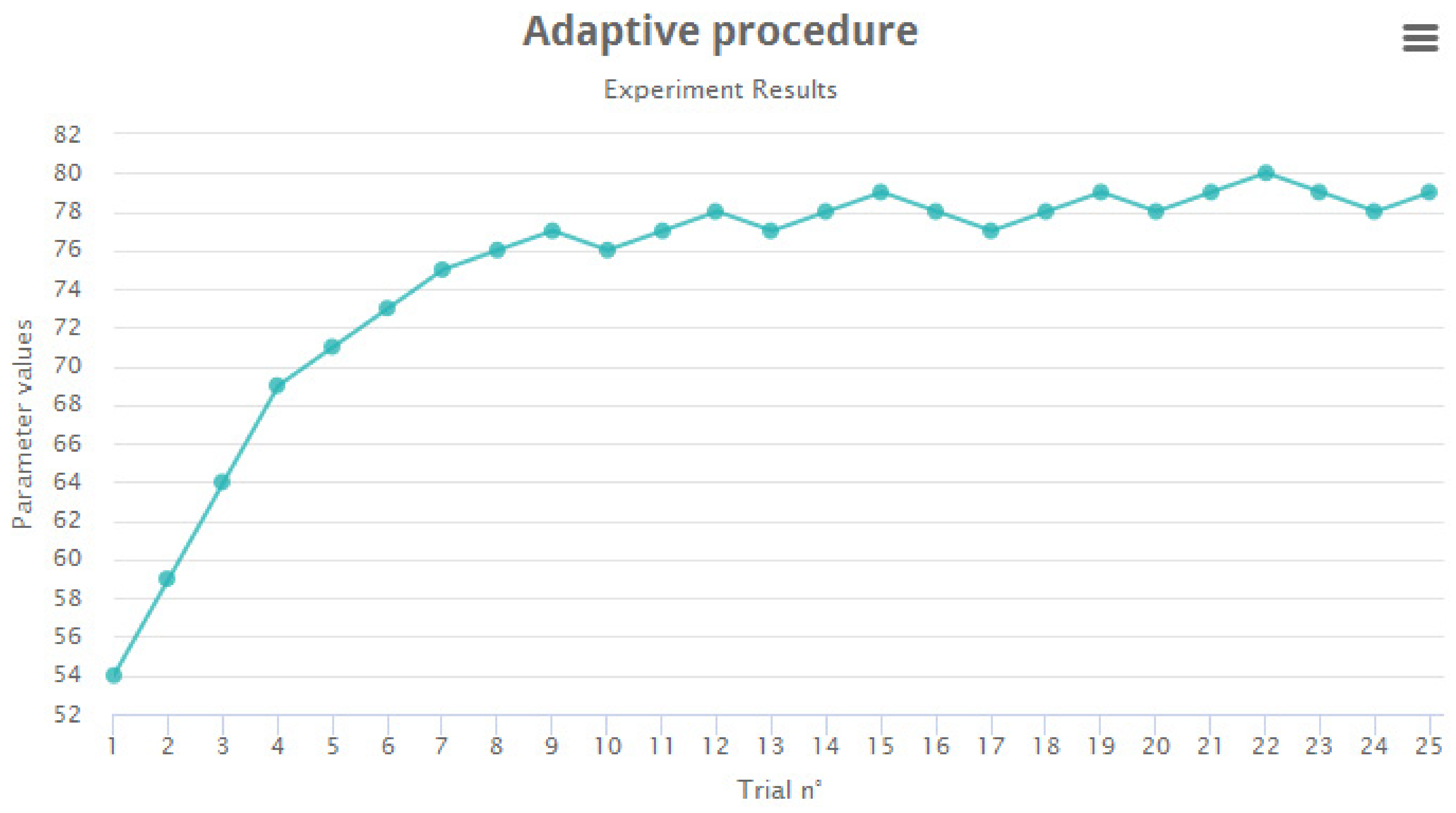
```
myVariable =
{
  “myNumber”: 5,
  myString: “bla”,
  myArray: [“a”,”b”,”c”]
};
```

3 built-in plot-types

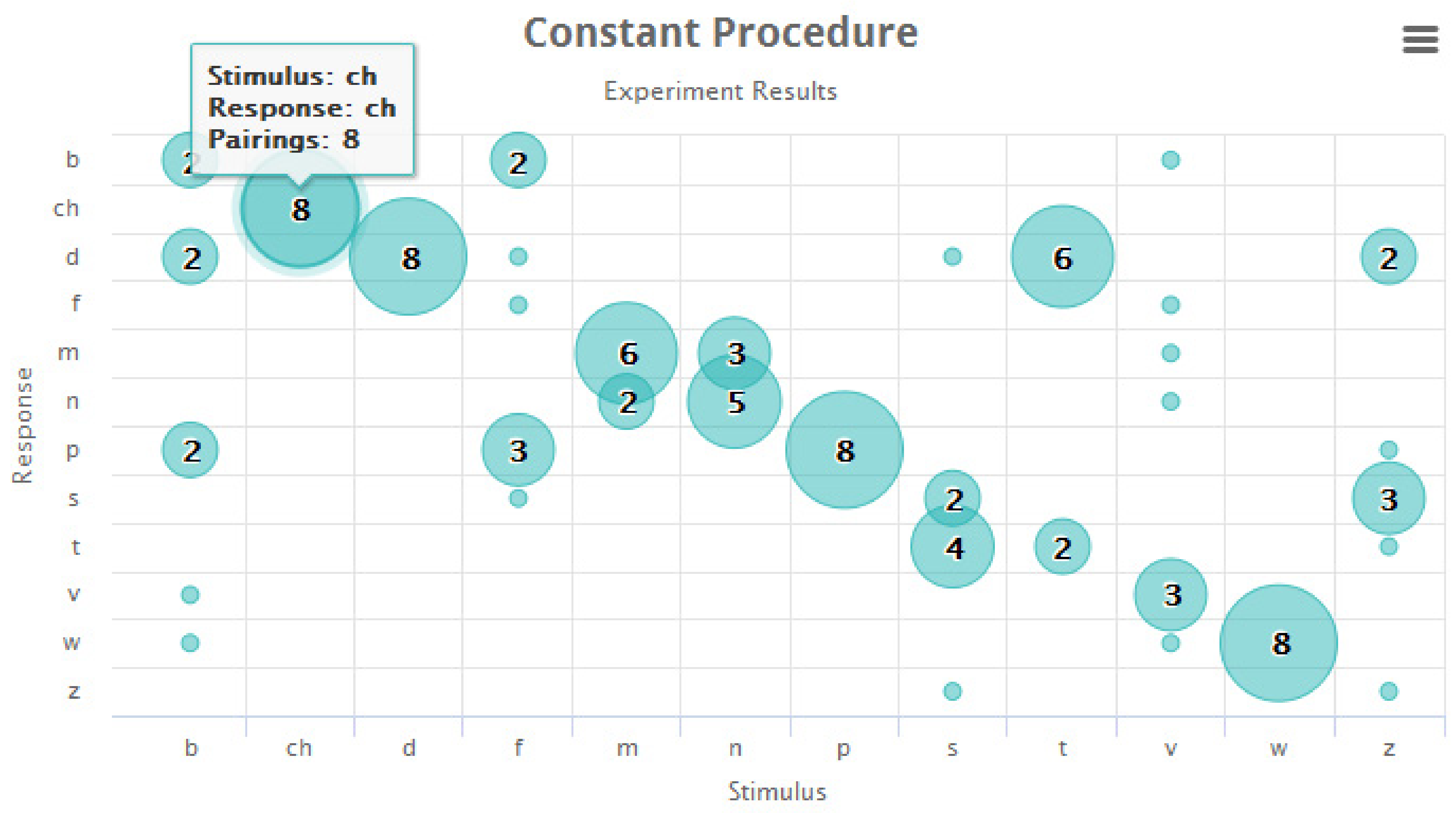
Line

Matrix

Polar



Parameter values:	54, 59, 64, 69, 71, 73, 75, 76, 77, 76, 77, 78, 77, 78, 79, 78, 77, 78, 79, 78, 79, 80, 79, 78, 79
Last value:	79
Reversals:	77, 76, 78, 77, 79, 77, 79, 78, 80, 78
Mean (std) last 6 reversals:	78.5 (±0.957)
Mean (std) last 6 trials:	78.8333 (±0.687)



Correct % per stimulus

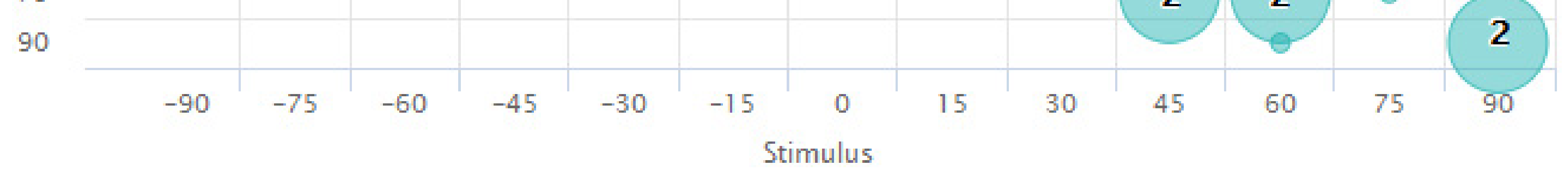
B	Ch	D	F	M	N	P	S	T	V	10
25.00%	100.00%	100.00%	12.50%	75.00%	62.50%	100.00%	25.00%	25.00%	37.50%	10

Total percentage correct: 56.2500%

	b	ch	d	f	m	n	p	s	t	v	w	z
b	2	0	0	2	0	0	0	0	0	1	0	0

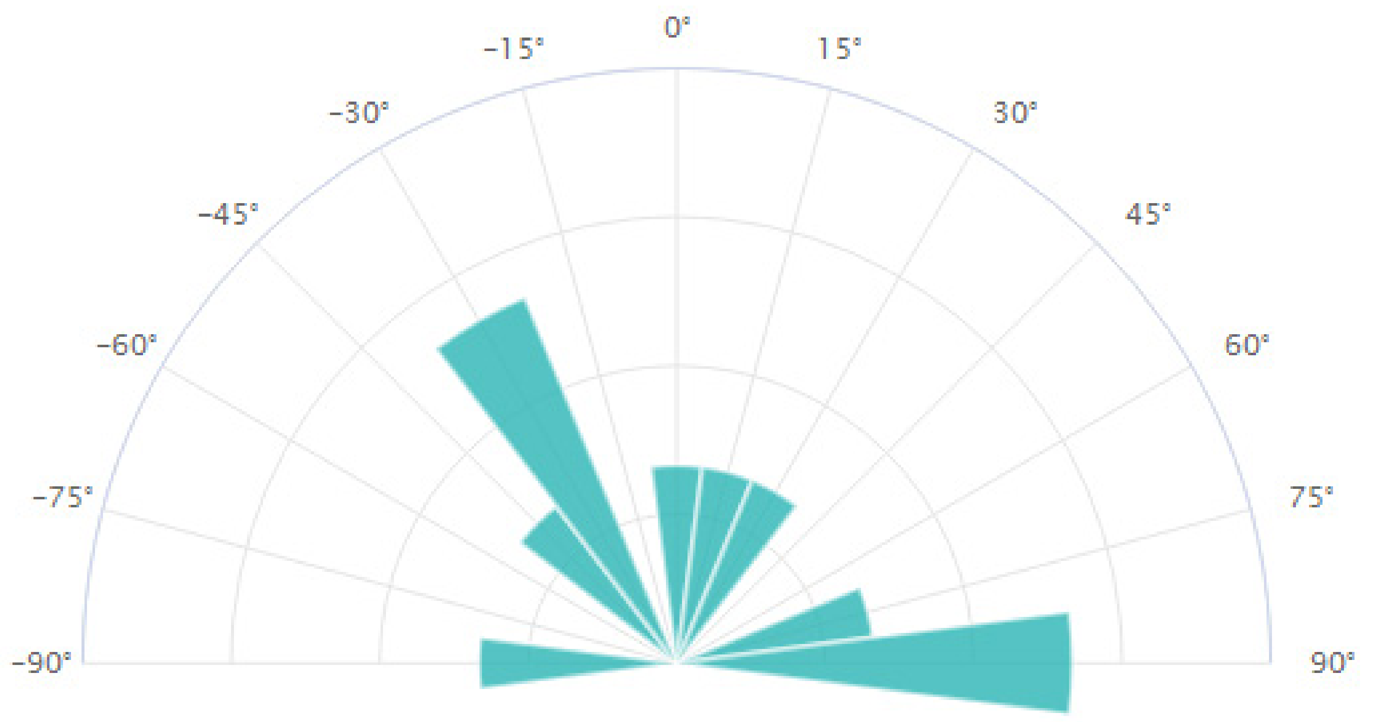
ch	0	8	0	0	0	0	0	0	0	0	0	0
d	2	0	8	1	0	0	0	1	6	0	0	2
f	0	0	0	1	0	0	0	0	0	1	0	0
m	0	0	0	0	6	3	0	0	0	1	0	0
n	0	0	0	0	2	5	0	0	0	1	0	0
p	2	0	0	3	0	0	8	0	0	0	0	1
s	0	0	0	1	0	0	0	2	0	0	0	3
t	0	0	0	0	0	0	0	4	2	0	0	1
v	1	0	0	0	0	0	0	0	0	3	0	0
w	1	0	0	0	0	0	0	0	0	1	8	0
z	0	0	0	0	0	0	0	1	0	0	0	1

Trial	Stimulus	Answer	Correct
1	z	t	Incorrect
2	b	d	Incorrect
3	p	p	Correct
4	f	d	Incorrect
5	z	s	Incorrect
6	m	m	Correct
7	w	w	Correct



Localization

% correct per direction



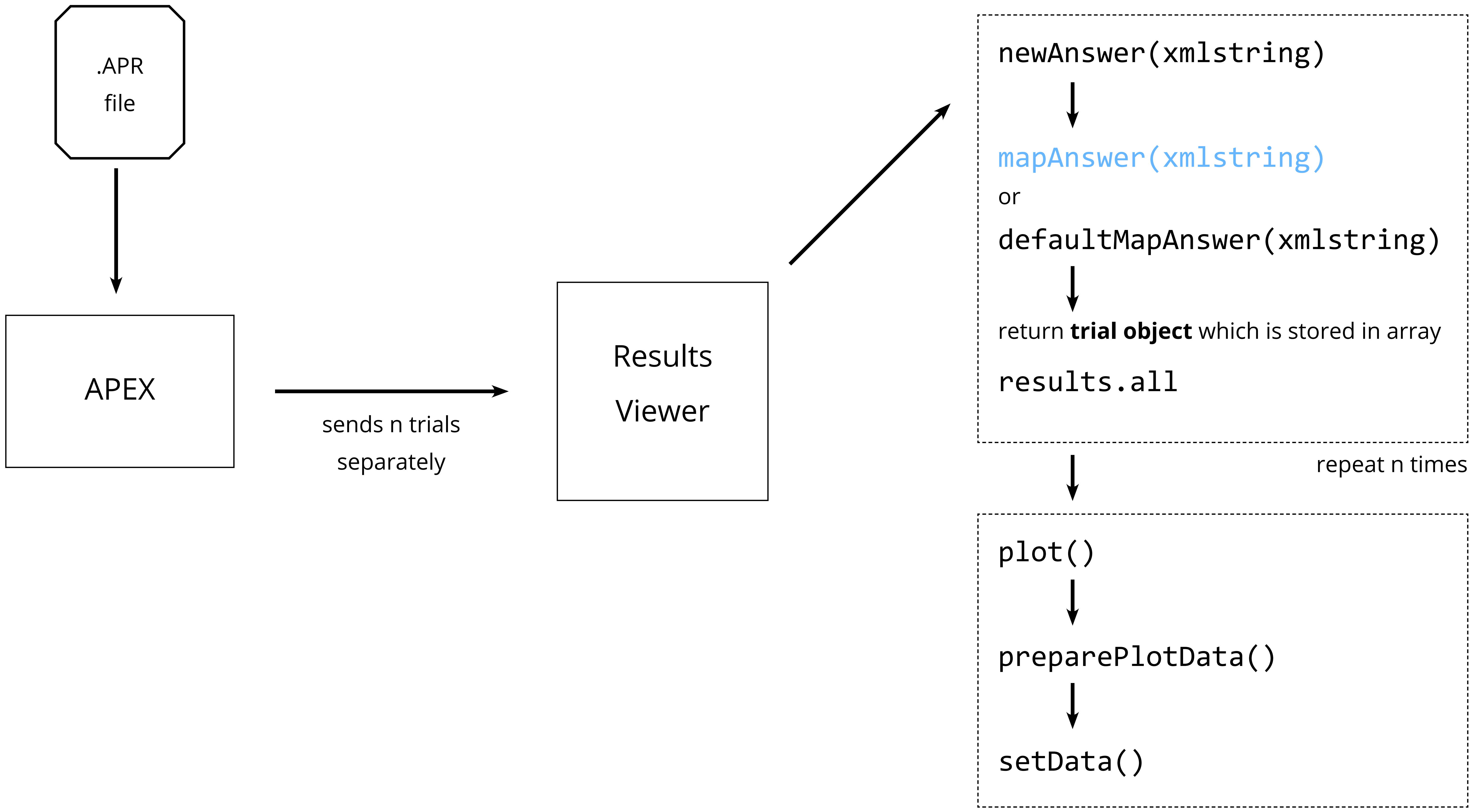
Correct % per stimulus

-90	-75	-60	-45	-30	-15	0	15	30	45	60	
33.33%	0.00%	0.00%	33.33%	66.67%	0.00%	33.33%	33.33%	33.33%	0.00%	0.00%	33

Total percentage correct: 25.6410%

-90	-75	-60	-45	-30	-15	0	15	30	45	60	75	90

Overview of Resultsviewer internals



Overview of Resultsviewer internals

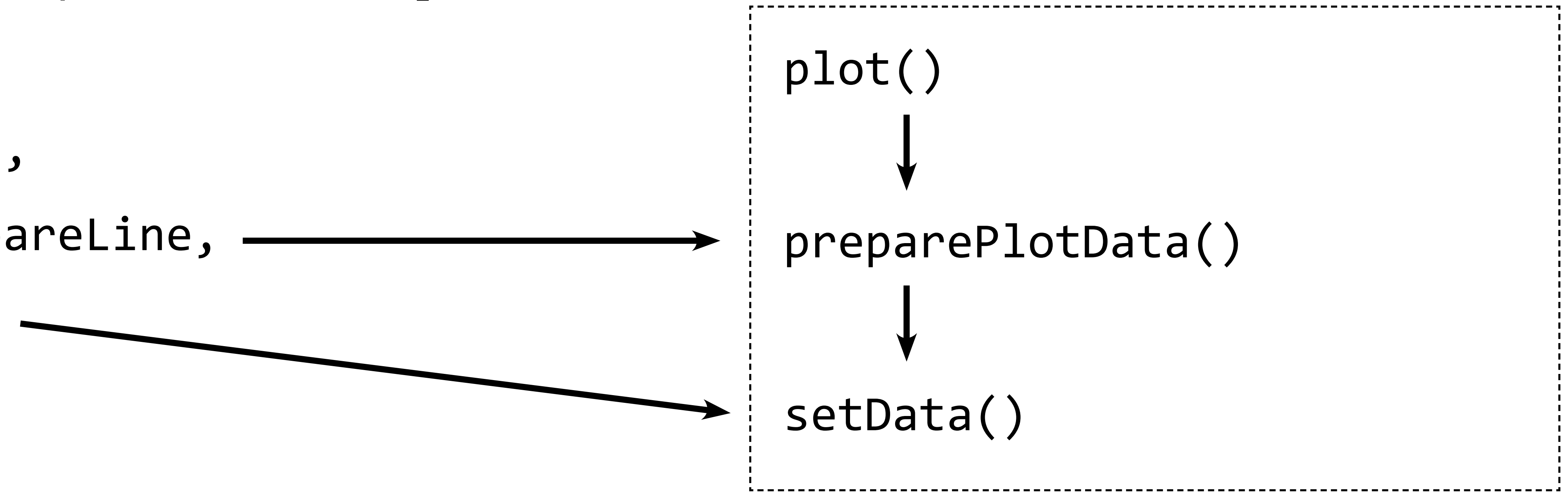
```
plots = {};


plots.types = [“line”, “matrix”, “polar”,”text”];
plots.line   = { data: [],
                 highChart: true,
                 prepare:   prepareLine,
                 setData:   null
               };
```

Overview of Resultsviewer internals

```
plots = {};


plots.types = [“line”, “matrix”, “polar”,”text”];
plots.line   = { data: [],
                 highChart: true,
                 prepare:   prepareLine,
                 setData:   null
               };
```



Prepared plot data

prepare function generates data object

Line

```
plots.line.data =
[{
   values: [],
   reversals: [],
   meanrevs: number,
   meanrevstd: number,
   meantrials: number,
   meantrialstd: number
}]
```

Matrix

```
plots.matrix.data =
[{
   values: [ { x: xlabel ref,
               y: ylabel ref,
               z: x&y pairings
           } ],
   percentages: n,
   xlabels: [],
   ylabels: [],
   raw: [][]
}]
```

Polar

```
plots.polar.data  =
[{
   values: []
}]
```

Setdata

Is only required if plot needs to be reconfigured/redrawn based on data

Matrix

```
function setDataMatrix(target, data, index)
{
    //  x & y axis labels update
    target.xAxis[0].update({
        categories: data.xlabels,
        ceiling: data.xlabels.length - 1
    });
    target.yAxis[0].update({
        categories: data.ylabels,
        max: data.ylabels.length - 1,
        ceiling: data.ylabels.length - 1
    });
    return true;
}
```

Customizing with config object

Global

```
global =
{
   answers: [“pot”,”pan”,”kan”,”kom”],
   stimuli: [],
   removefromanswer: [“word”],
   multiprocedure: true/false
}
```

Line

```
line =
{
   show: true/false,
   parametername: “gain”,
   parameterunit: “dB”,
   parameterkey: “parametervalue”,
   trialsformean: n,
   reversalsformean: n
},
```

Customizing with config object

Matrix

```
matrix:
{
   show: true/false,
   removefrommatrix: [“up”,”down”]
},
```

Polar

```
polar:
{
   show: true/false,
   mindegree: n,
   maxdegree: n
},
```

How to customize

Inside .apx file

```
<results>
   <page>resultsviewer.html</page>
   <resultparameters>
      <parameter name=”line_reversalsformean”>6</parameter>
   </resultparameters>
</results>
```

Format:

name=”*plottype_parametername*”

for arrays such as global_stimuli

use , or | to input multiple values

e.g. pot,pan,kan,kom

How to customize

Custom resultsviewer

```
<results>
   <page>my-resultsviewer.html</page>
</results>
```

my-resultsviewer.html

```
...
<script>
   config.global.stimuli = [“-90”,”-45”,”0”,”45”,”90”]; //force list of possible stimuli
   config.polar.show = true; //force polar plot to show
</script>
...
```

How to customize

In global config javascript file

`Apex/resultsviewer/resultsviewer-config.js`

> Shows all possible configuration options

> Will be applied to EVERY experiment/resultsviewer

> Not recommended to edit this unless it’s a specialized Apex install for one experiment

How to customize with js hooks

These are all shown in resultsviewer-config.js

> define these in your custom resultsviewer.html <script> tags

`mapAnswer(xmlstring)`

For custom XML answer-structures (when using pluginprocedures and such)

`resultsFilter(t)`

Filters out Trial objects based on certain criteria (function returns true to keep, false to remove)

`lineDataFilter(t)`

Same but specifically for Line plots

`matrixDataFilter(t)`

Same

`convertParameterValue(v)`

Automatically convert parametervalue with a formula

How to customize with special buttons

These are all shown in resultsviewer-config.js

> define these in your custom resultsviewer.html <script> tags

```
customButtons.push({
   name: “”,
   label: “”,
   replot: true/false,
   behavior: function()
   {
      //code
   }
});
```

Example

```
customButtons.push({
   name: “btn1”,
   label: “Add series”,
   replot: true,
   behavior: function()
   {
      var target = getChart(“line”);
      if (typeof target === “null”)
         alert(“Enable line first”);

      target.addSeries( { data: [1,2,3,4,5,6] } );
      target.redraw();
   }
});
```

Advanced: Adding a new plottype

Highcharts.com

pick a plot, give it a name and add it to the plots object, eg “scatter”

```
plots.types.push(“scatter”);
plots.scatter =
{
   data: [],
   highChart: true,
   prepare: function,
   setData: function,
   chartConfig: {}
};
```