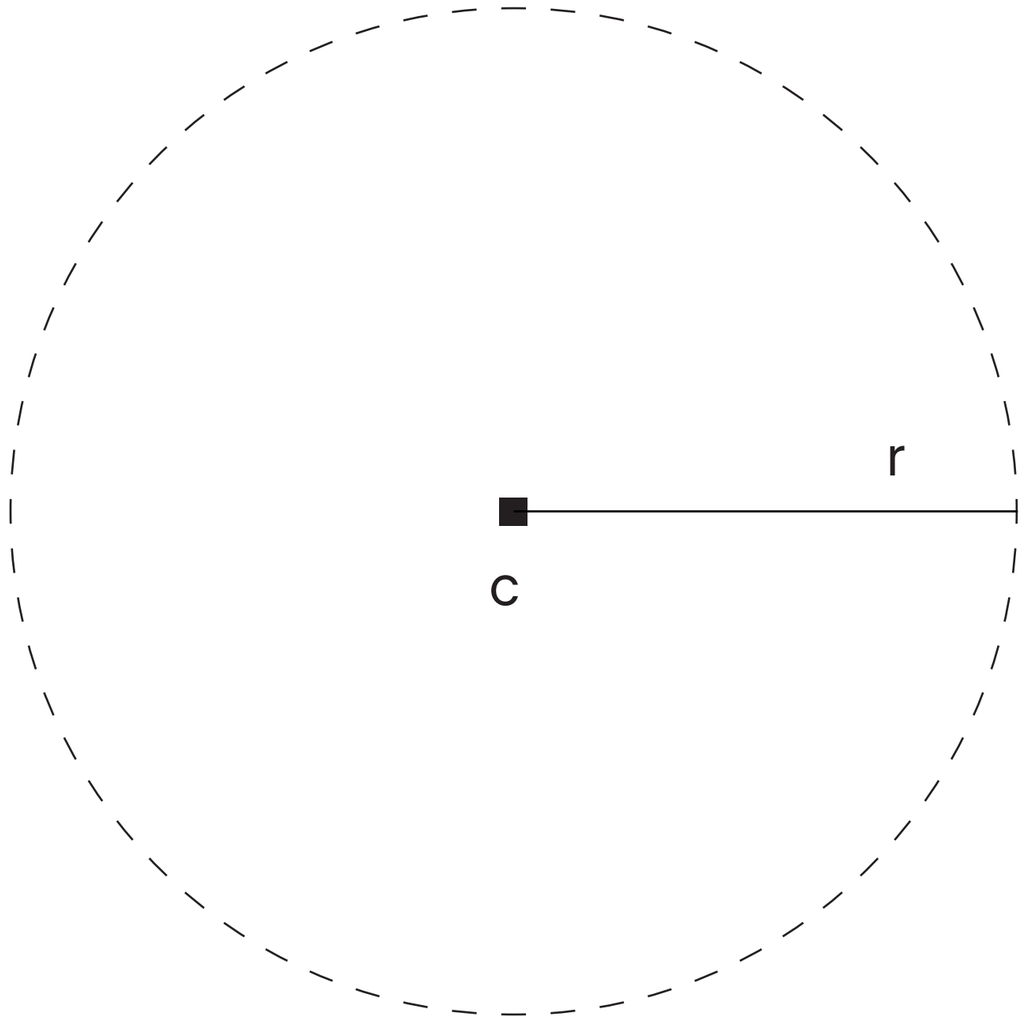
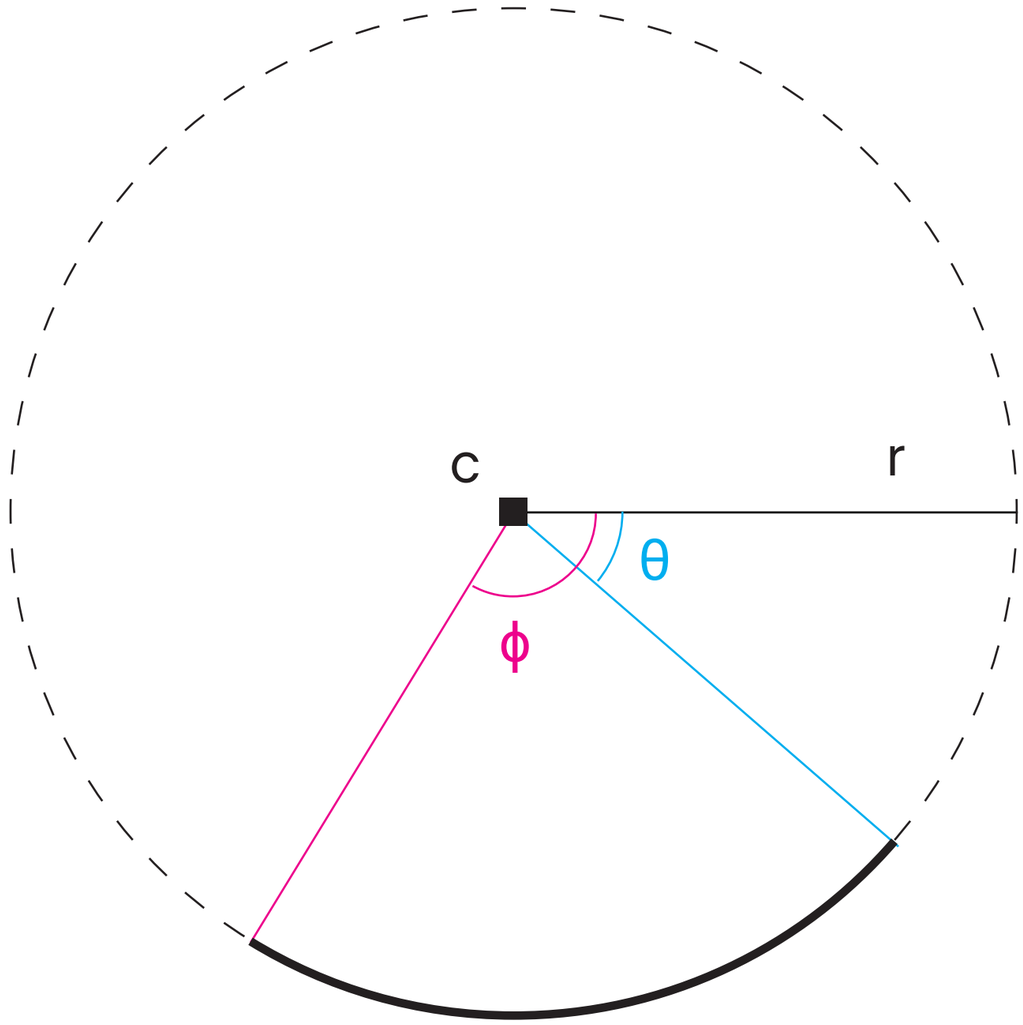
ARCS

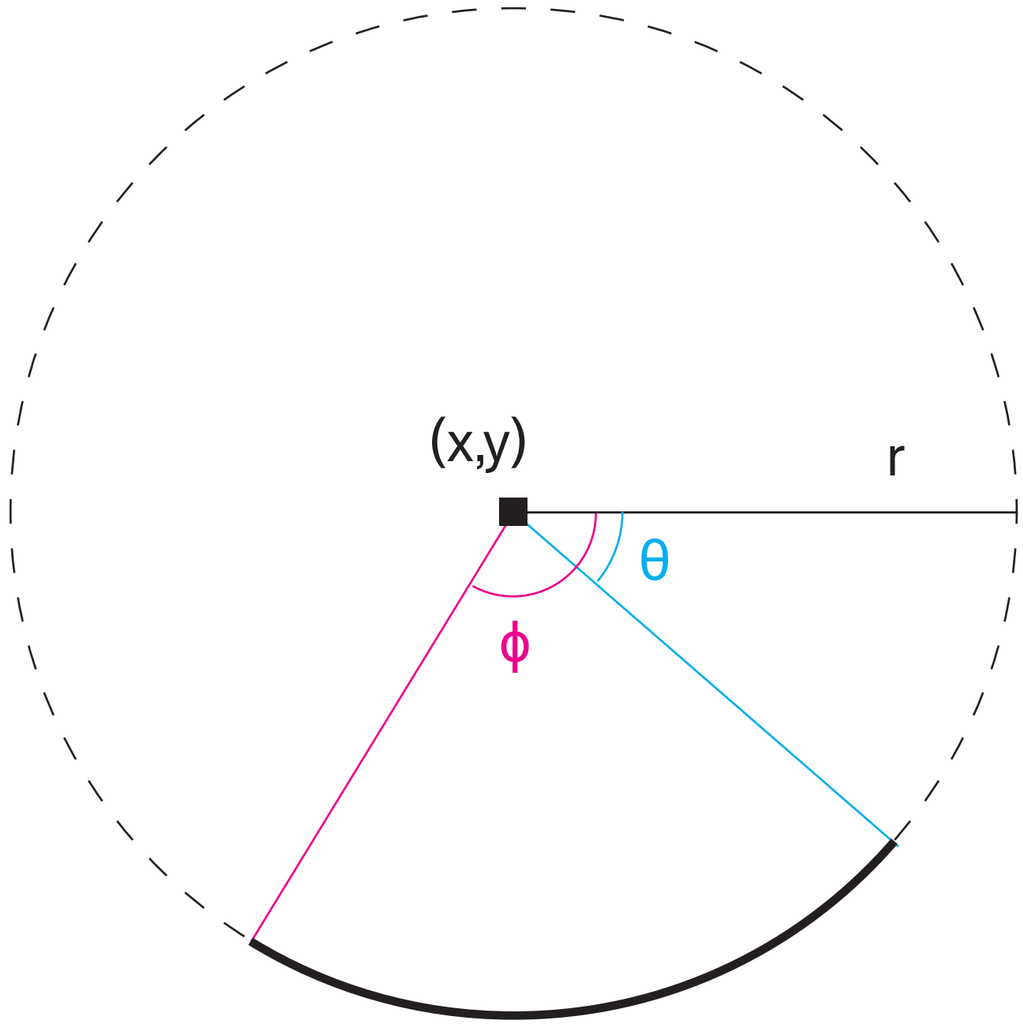
- circular segments



ARCS

- circular segments
- angles are calculated clockwise from 0° in the +x direction, unless specified otherwise

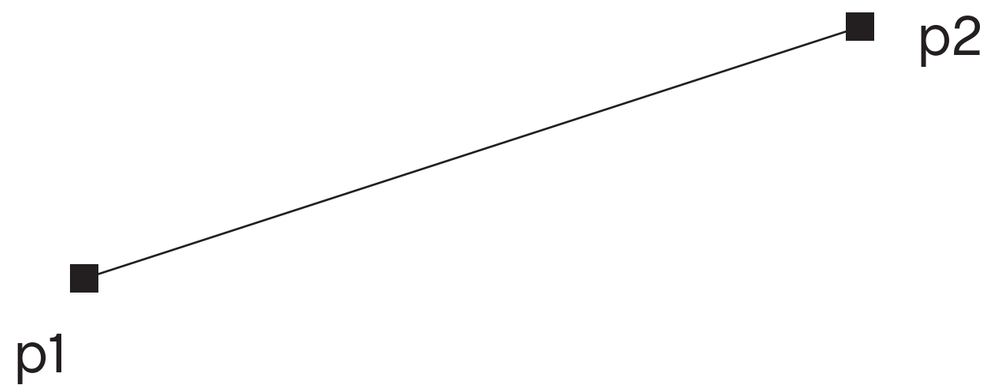





```
d3.path().arc(x, y, r, θ, ϕ[, counterclockwise])
```

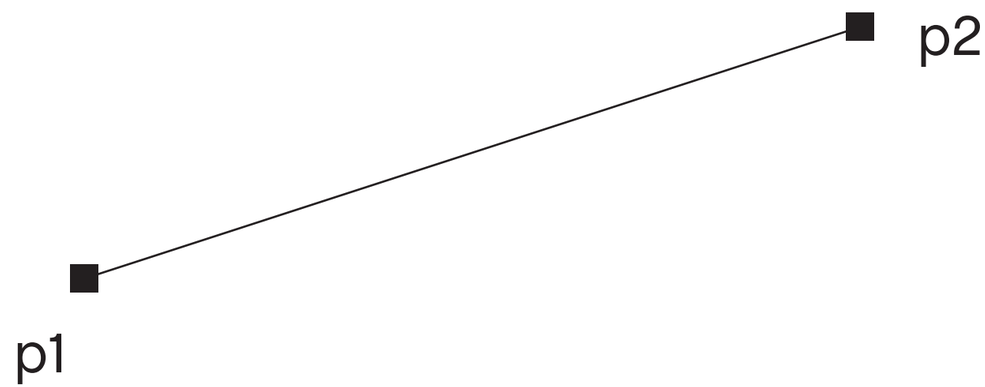
QUADRATIC CURVES

- 1 control point



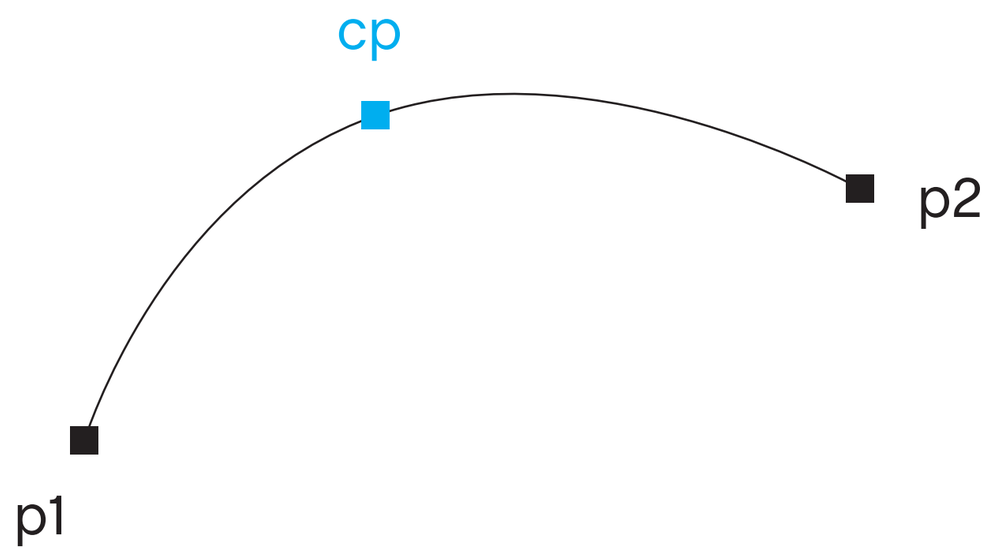
QUADRATIC CURVES

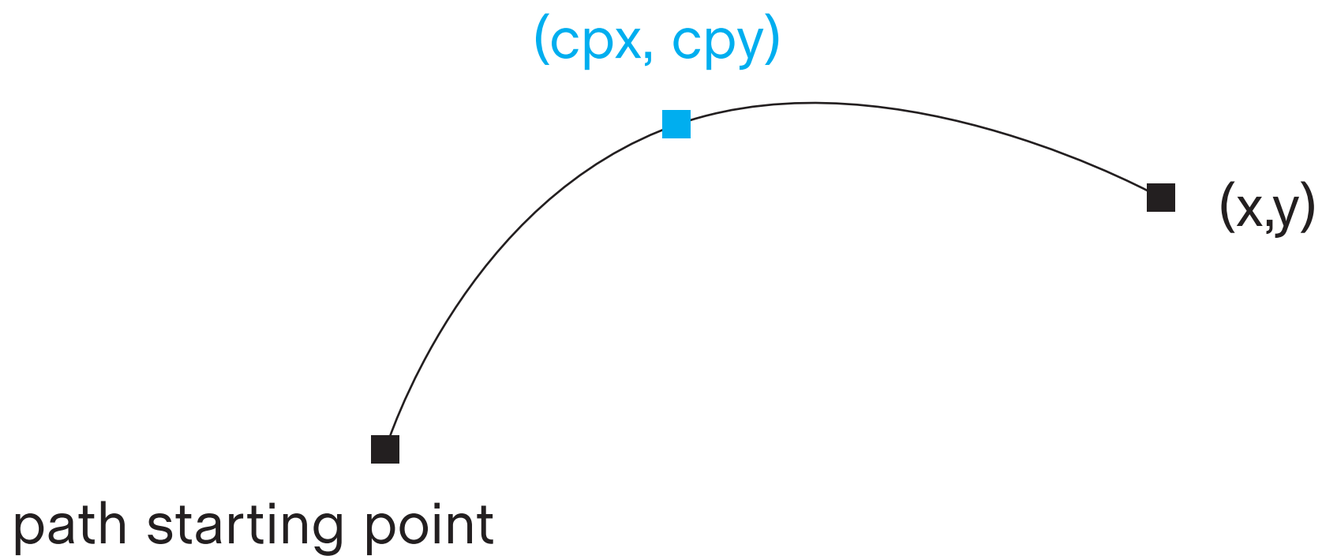
- 1 control point



QUADRATIC CURVES

- 1 control point
- point lies on the curve

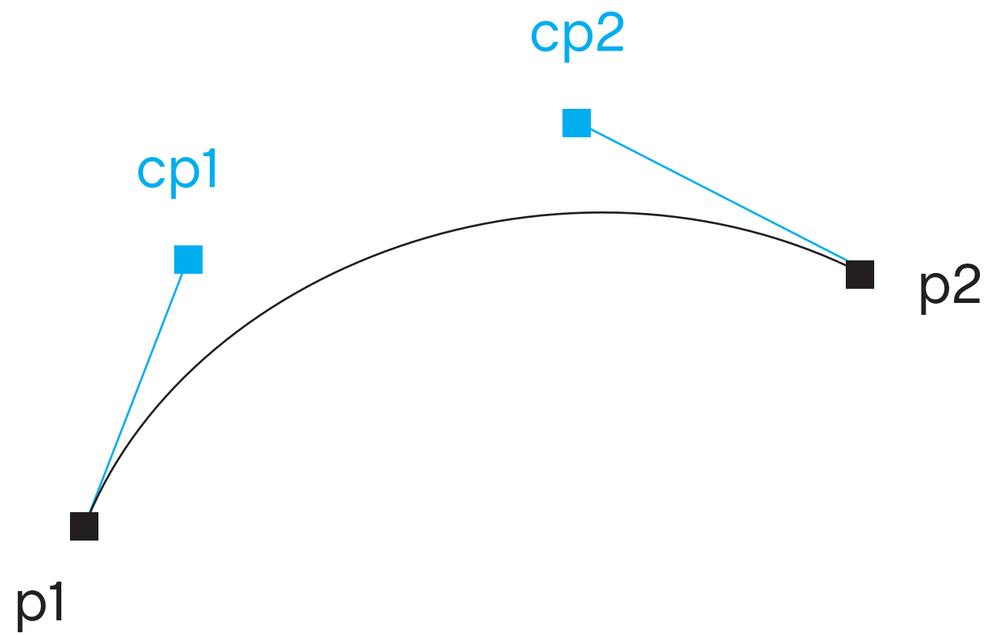





```
d3.path().quadraticCurveTo(cpx, cpy, x, y)
```

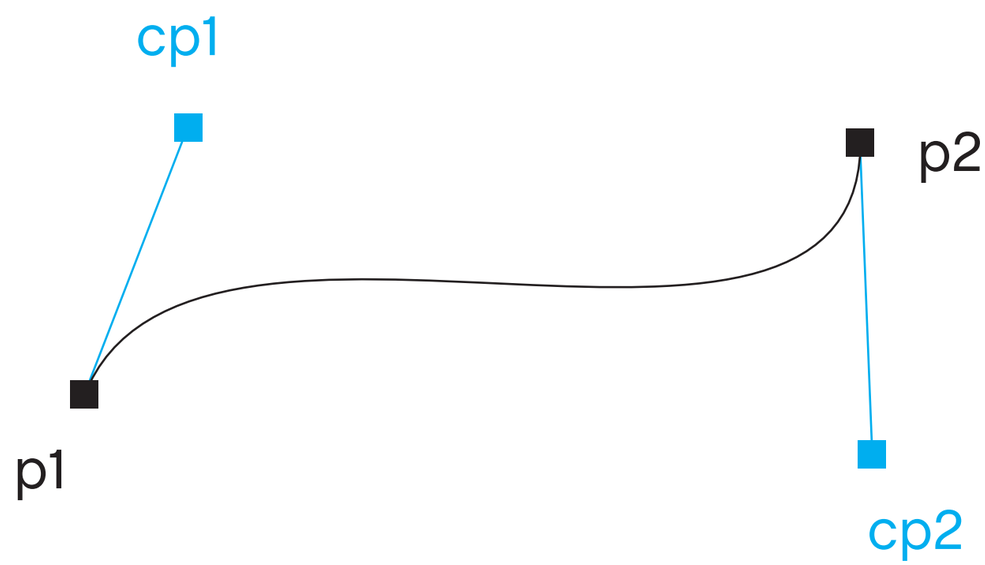
CUBIC BEZIER CURVES

- 2 control points



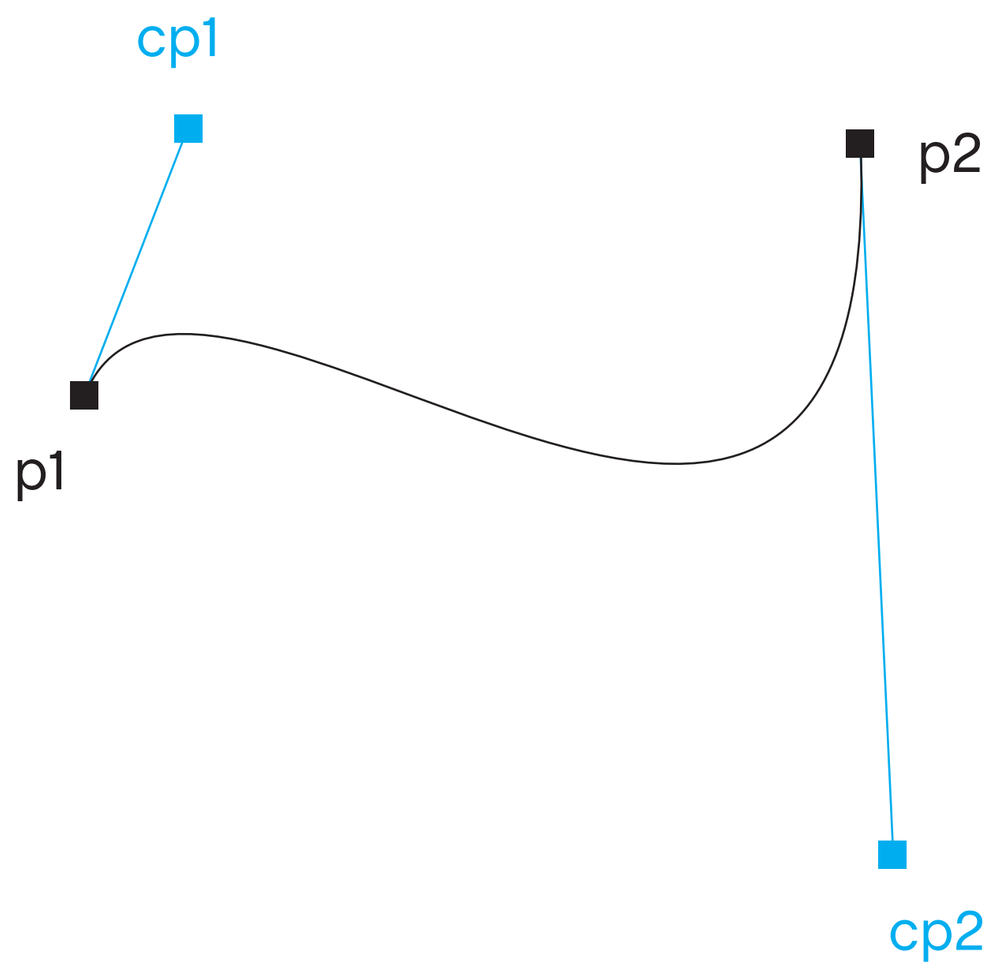
CUBIC BEZIER CURVES

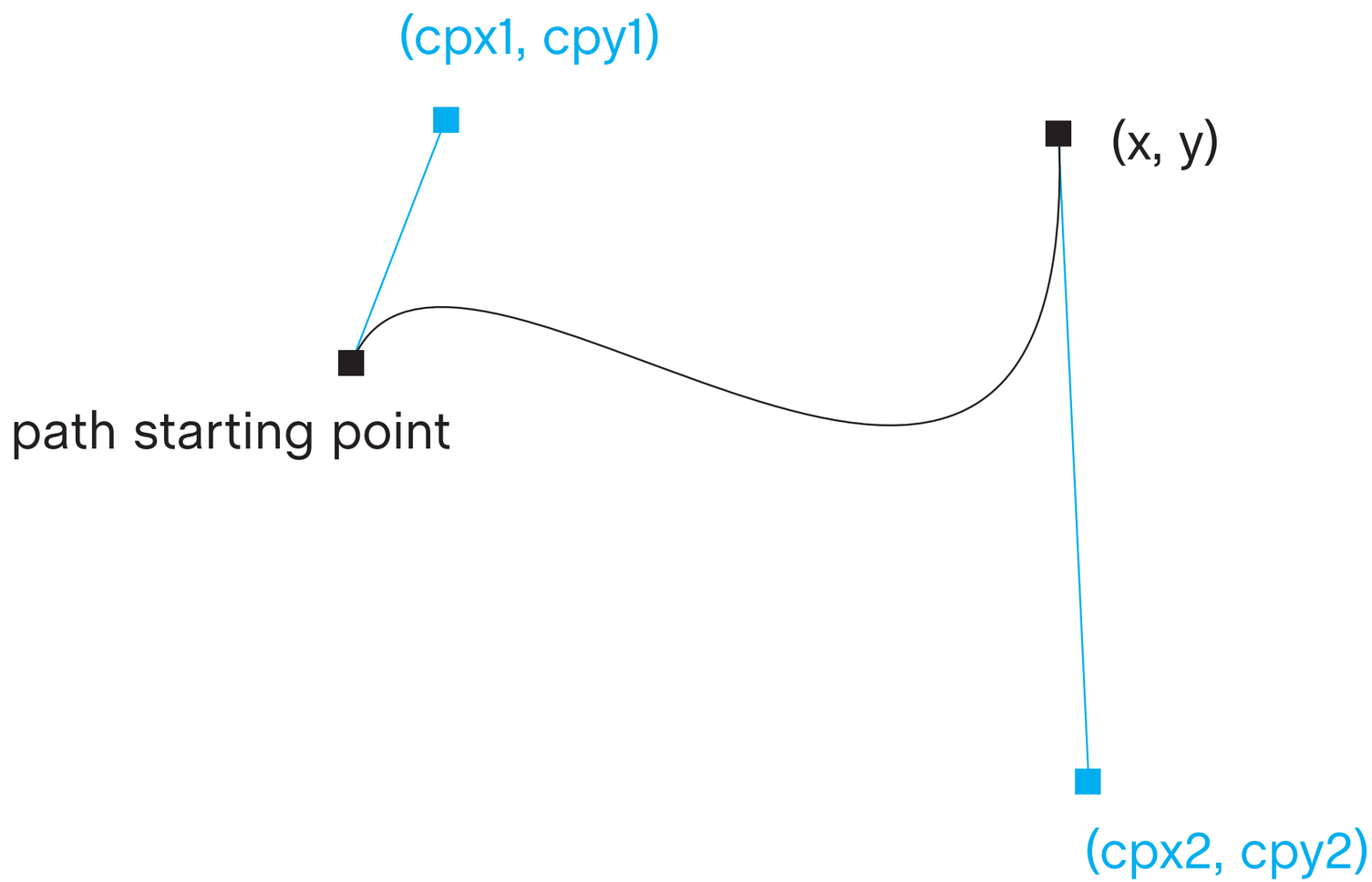
- 2 control points
- the shape of the curve is influenced by the position of the control points...



CUBIC BEZIER CURVES

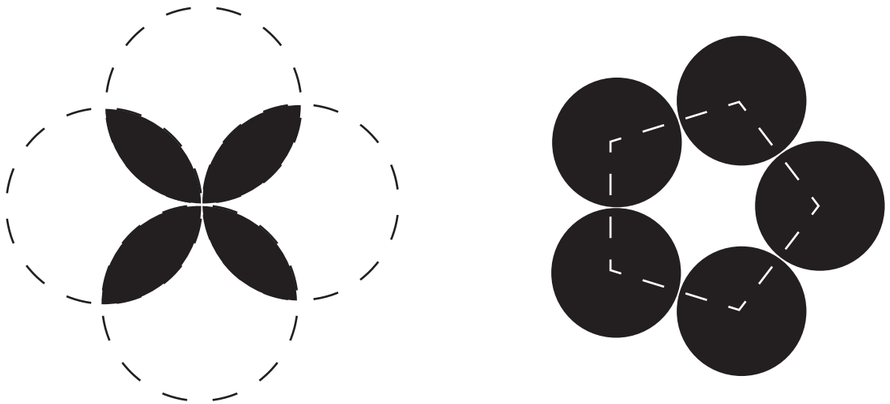
- 2 control points
- the shape of the curve is influenced by the position of the control points...
- ...as well as their distance from each other






```
d3.path().bezierCurveTo(cpx1, cpy1, cpx2, cpy2, x, y)
```

PATH EXAMPLES



Both of my shapes used the arc function.

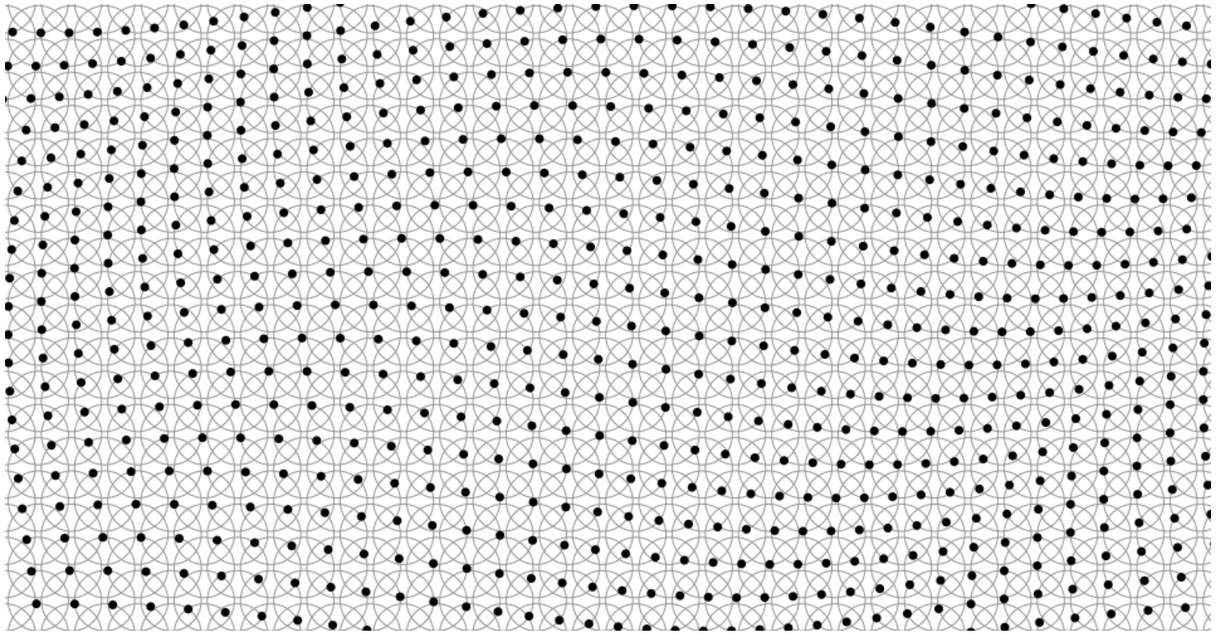
(Sorry, nothing too fancy!)

```
var r = 5;
ctx = d3.path();
ctx.arc(0, -r, r, 0, Math.PI)
ctx.arc(-r, 0, r, -Math.PI/2, Math.PI/2)
ctx.arc(0, r, r, Math.PI, 0)
ctx.arc(r, 0, r, Math.PI/2, -Math.PI/2)
return ctx.toString();
```

PATH EXAMPLES

There's a surprising amount you can do with circles... such as simulating wave motion! http://bl.ocks.org/mbostock/c66ab1426f-4b8945a7ef

If possible, try to keep the shapes simple. Let your computer handle all the math :)



ADDITIONAL LINKS

- d3 path reference
  https://github.com/d3/d3-path

- some decent d3 tutorials
  https://www.dashingd3js.com/svg-paths-and-d3js

- practicing bezier curves :3
  http://bezier.method.ac/