# Talks and Workshops

## MoMath Workshop

> id: bridges

> title: The Bridges of Königsberg

> section: momath

> color: "#A7208A"

Can you draw a path that crosses every bridge once, but \_not more than once\_, without entering the

water? You can start and end on any area of land.

x-tabbox.full-width

.tab

h3 Map 1#[span.check.incorrect(when="bridge-0")]

x-solved

include ../graph-theory/svg/bridges-1.svg

button.btn Clear

button.btn.right(hidden) Skip

.tab

h3 Map 2#[span.check(when="bridge-1")]

x-solved

include ../graph-theory/svg/bridges-2.svg

button.btn Clear

button.btn.right(hidden) Skip

.tab

h3 Map 3#[span.check(when="bridge-2")]

x-solved

include ../graph-theory/svg/bridges-3.svg

button.btn Clear

button.btn.right(hidden) Skip

.tab

h3 Map 4 #[span.check.incorrect(when="bridge-3")]

x-solved

include ../graph-theory/svg/bridges-4.svg

button.btn Clear

button.btn.right(hidden) Skip

---

> id: utilities

> title: Three Utilities Puzzle

Can you connect each of these utility companies to each of the houses, without any of the lines

intersecting?

.box.no-padding

include ../graph-theory/svg/utilities.svg

button.btn Clear

---

> id: planarity

> title: Planarity Game

::: .box.blue

#### Planarity

x-solved

svg#planarity(viewBox="0 0 720 360")

This is a planar graph, but the ${n}{n|7|5,20,1} vertices have been scrambled up. Rearrange the

vertices so that none of the edges overlap.

p.btn-row: button.btn New Random Graph

:::

---

> id: maps-1

> title: Map Colouring

How many colours do you need for these maps, if adjacent countries or states cannot have the same

colour?

.four-colour-icons

for i in [1, 2, 3, 4, 5, 6, 7]

.four-colour-icon(tabindex=0)

x-tabbox.four-colours.full-width

.tab

h3 United States #[span.check(when="map-0")]

x-solved

.colour-count(style="margin-bottom: -32px") #[span 0] colours used

include ../graph-theory/svg/colours-1.svg

button.btn.clear Clear

.tab

h3 South America #[span.check(when="map-1")]

x-solved

.colour-count #[span 0] colours used

include ../graph-theory/svg/colours-2.svg

button.btn.clear Clear

.tab

h3 Germany #[span.check(when="map-2")]

x-solved

.colour-count #[span 0] colours used

include ../graph-theory/svg/colours-3.svg

button.btn.clear Clear

.tab

h3 England #[span.check(when="map-3")]

x-solved

.colour-count #[span 0] colours used

include ../graph-theory/svg/colours-4.svg

button.btn.clear Clear

---

> id: salesman-4

> title: Travelling Salesperson Map

Try rearranging the cities on this map, and watch how the shortest path between them changes. You

can remove cities by tapping them, and you can add cities by clicking anywhere on the map (up to 8):

figure: .tsm

svg(width=760 height=480 viewBox="0 0 760 480")

---

## NCTM 2021

> id: maps-v2

> title: Map Colouring

> section: mathsconf

.four-colour-icons

for i in [1, 2, 3, 4, 5, 6, 7]

.four-colour-icon(tabindex=0)

x-tabbox.four-colours.full-width

.tab

h3 United States #[span.check(when="map-0")]

x-solved

.colour-count(style="margin-bottom: -32px") #[span 0] colours used

include ../graph-theory/svg/colours-1.svg

button.btn.clear Clear

.tab

h3 South America #[span.check(when="map-1")]

x-solved

.colour-count #[span 0] colours used

include ../graph-theory/svg/colours-2.svg

button.btn.clear Clear

.tab

h3 Germany #[span.check(when="map-2")]

x-solved

.colour-count #[span 0] colours used

include ../graph-theory/svg/colours-3.svg

button.btn.clear Clear

.tab

h3 England #[span.check(when="map-3")]

x-solved

.colour-count #[span 0] colours used

include ../graph-theory/svg/colours-4.svg

button.btn.clear Clear

---

> title: Three Body Problem

> id: three-bodies

What happens when three plants orbit around each other in space?

figure: x-geopad.simulation.r(width=480 height=480)

canvas(width=960 height=960)

svg

circle.large.move.red(name="a")

circle.large.move.blue(name="b")

circle.large.move.green(name="c")

path.thin(x="segment(a, a.translate(va))" arrows="end")

path.thin(x="segment(b, b.translate(vb))" arrows="end")

path.thin(x="segment(c, c.translate(vc))" arrows="end")

x-play-toggle

button.icon-btn.restore: x-icon(name="restart")

---

> id: bridges-v2

> title: The Bridges of Königsberg

Can you draw a path that crosses every bridge once, but \_not more than once\_, without entering the

water? You can start and end on any area of land.

x-tabbox.full-width

.tab

h3 Map 1#[span.check.incorrect(when="bridge-0")]

x-solved

include ../graph-theory/svg/bridges-1.svg

button.btn Clear

button.btn.right(hidden) Skip

.tab

h3 Map 2#[span.check(when="bridge-1")]

x-solved

include ../graph-theory/svg/bridges-2.svg

button.btn Clear

button.btn.right(hidden) Skip

.tab

h3 Map 3#[span.check(when="bridge-2")]

x-solved

include ../graph-theory/svg/bridges-3.svg

button.btn Clear

button.btn.right(hidden) Skip

.tab

h3 Map 4 #[span.check.incorrect(when="bridge-3")]

x-solved

include ../graph-theory/svg/bridges-4.svg

button.btn Clear

button.btn.right(hidden) Skip

---

> id: utilities-v2

> title: Three Utilities Puzzle

Can you connect each of these utility companies to each of the houses, without any of the lines

intersecting?

.box.no-padding

include ../graph-theory/svg/utilities.svg

button.btn Clear

---

> title: Julia Sets

> id: julia2

In this diagram, we highlight all points `pill(x\_0,"yellow","x0")` on the complex plane, for which

the recursive sequence `pill(x\_n,"yellow") = pill(x\_(n-1),"yellow")^2 + pill(c,"red","c")` is

bounded (it doesn't diverge).

x-geopad(width=720 height=480 x-axis="-1.8,1.8,1" y-axis="-1.2,1.2,1" axes grid padding=8 projections="no" style="margin-bottom: 24px" label-suffix=",i" axis-names="Real, Imaginary")

canvas(width=1440 height=960)

svg

circle.move.yellow(name="x0" x="point(0.5,0.5)" target="x0")

circle.move.red(name="c" x="point(0,0)" target="c")

circle.yellow.transparent(name="x1" x="iterate(x0,c)" target="x1")

circle.yellow.transparent(name="x2" x="iterate(x1,c)" target="x2")

circle.yellow.transparent(name="x3" x="iterate(x2,c)" target="x3")

path.yellow(x="spiral(x0,c)")

.geo-legend

.formula.md `pill(x\_n,"yellow") = pill(x\_(n-1),"yellow")^2 + pill(var("complex(c)"),"red","c")`

.sequence

.md `pill(x\_0,"yellow", "x0") = var("complex(x0)")`

.md `pill(x\_1,"yellow", "x1") = var("complex(x1)")`

.md `pill(x\_2,"yellow", "x2") = var("complex(x2)")`

.md `pill(x\_3,"yellow", "x3") = var("complex(x3)")`

div

span.vdots …

strong.var.m-blue(:show="converges" data-display="inline") Bounded!

strong.var(:show="!converges" data-display="inline") Diverges!

x-slideshow(hidden): .legend-box: div

---

> id: mandel-paint

> title: The Mandelbrot Set

For Julia sets, we chose a fixed value for `pill(c,"red","c")`, and then changed the position of

`pill(x\_0,"yellow","x0")` to colour the plane. Now let’s fix the value of

`pill(x\_0 = 0,"yellow","x0")`, and instead change the value of `pill(c,"red","c")`:

figure: x-geopad.no-background(width=720 height=480 x-axis="-2.1,1.1,1" y-axis="-1.1,1.1,1" axes grid padding=8 projections="no" label-suffix=",i" axis-names="Real, Imaginary")

img(src="/content/fractals/images/mandelbrot.png" data-bounds="1,0.5,-1,-1.5")

canvas(width=1440 height=960 style="opacity: 1")

svg

circle.move.red.pulsate(name="c" cx=0 cy=0 target="c")

circle.yellow.transparent(name="x0" x="point(0,0)" target="x0")

circle.yellow.transparent(name="x1" x="iterate(x0,c)" target="x1")

circle.yellow.transparent(name="x2" x="iterate(x1,c)" target="x2")

circle.yellow.transparent(name="x3" x="iterate(x2,c)" target="x3")

path.yellow(x="spiral(x0,c)")

.geo-legend

.formula.md `pill(x\_n,"yellow") = pill(x\_(n-1),"yellow")^2 + pill(var("complex(c)"),"red","c")`

.sequence

.md `pill(x\_0,"yellow", "x0") = var("complex(x0)")`

.md `pill(x\_1,"yellow", "x1") = var("complex(x1)")`

.md `pill(x\_2,"yellow", "x2") = var("complex(x2)")`

.md `pill(x\_3,"yellow", "x3") = var("complex(x3)")`

div

span.vdots …

strong.var.m-blue(:show="converges" data-display="inline") Bounded!

strong.var(:show="!converges" data-display="inline") Diverges!

---

> id: mandel-zoom

> title: Mandelbrot Zoom

Like all fractals, we can “zoom into” the Mandelbrot set forever, finding new patterns at every

scale. Here you can zoom into a part of the Mandelbrot set called the \_\_Seahorse valley\_\_:

.mandel-frame

- i = 1;

while i <= 27

img(src="/content/fractals/images/mandel/mandel-" + i + ".jpg" width=760 height=500)

- i += 1;

.scale.var Scale: ${pow(scale)}

x-slider(steps=27 continuous speed=0.1 :bind="scale")

---

> id: fractal-builder

> title: The Chaos Game

The \_\_chaos game\_\_ is a way to generate fractals using a simple rule. You start with a point,

repeatedly pick a random vertex of a polygon, and then mark the midpoint of the line from your

original point to that vertex. Then you continue from that new point. What shapes can you make?

include ../fractals/components/chaos-game

---

> id: circles

> title: Apollonian Gasket

> section: alex

## Alex Kontorovich

Can you place all smaller circles inside the big one?

svg.circles(width=600 height=600 viewBox="0 0 320 320")

circle.outer(cx=160 cy=160 r=108.07)