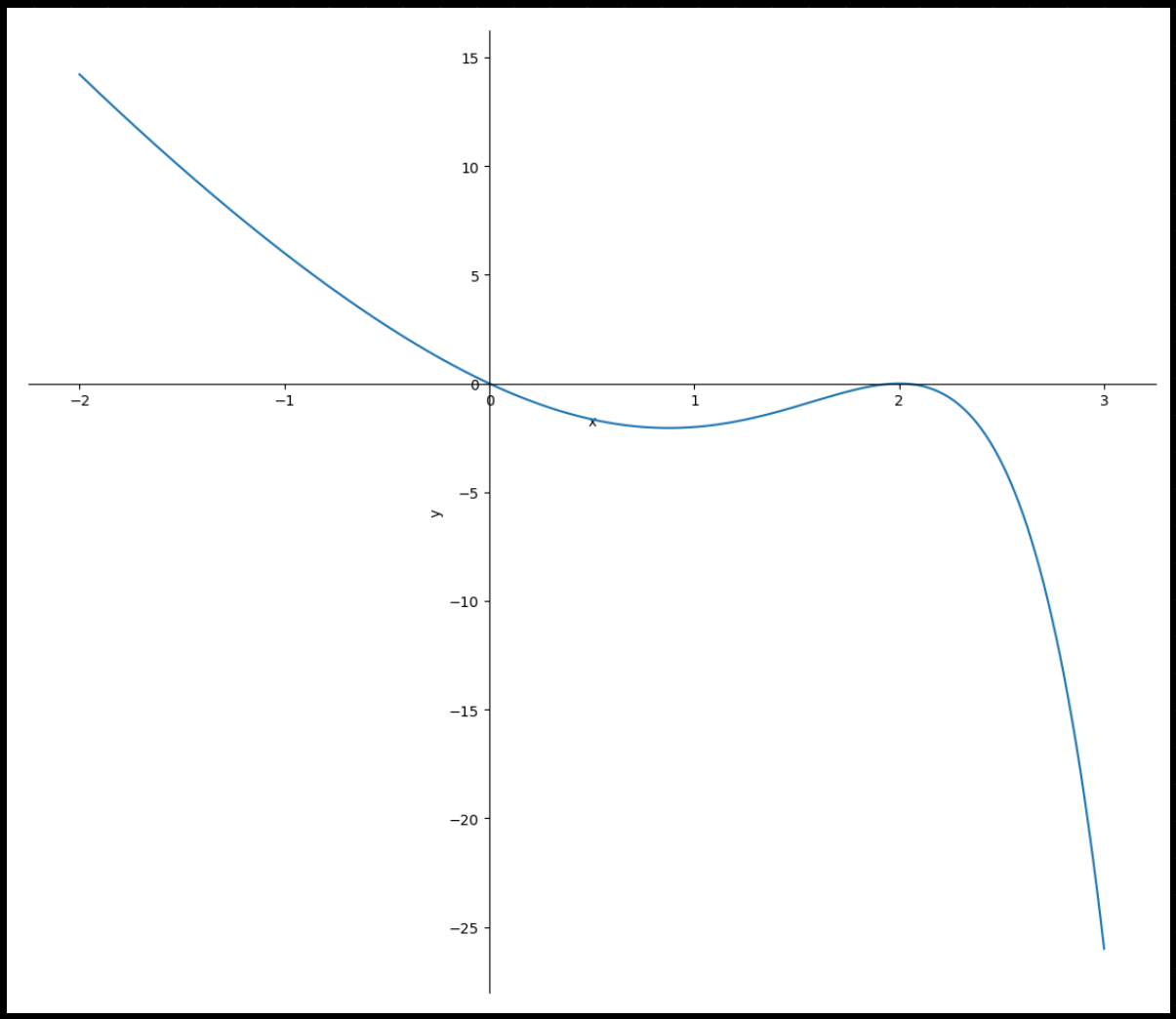
# Matte 4D – Exercise 4

**Problem 1** Consider

$$f(x) = (1-3^x)x^2 + 4(x-1)3^x + 4(1-x).$$

We wish to find the roots on $[-2,3]$, i.e. all $r$ with $f(r) = 0$.

a) The function looks like this:



b) Recall: The bisection method involves repeatedly finding the midpoint of your two nodes to get closer and closer to a root.

First iteration:

$$\frac{a+b}{2} = \frac{-2+3}{2} = \frac{1}{2}$$

$$f(1/2) = (1-\sqrt{3})\cdot\frac{1}{2^2} + 4\left(\frac{1}{2}-1\right)\sqrt{3} + 4\left(1-\frac{1}{2}\right)$$

$$= \frac{1}{4}(1-\sqrt{3}) - 2\sqrt{3} + 2 < 0$$

($1-\sqrt{3} < 0$, $-2\sqrt{3} < 0$)

and from the graph we see that $f(a) > 0$ and $f(b) < 0$, so we choose $1/2$ as a new right-endpoint.

Second iteration:

$$\frac{-2+1/2}{2} = -\frac{3}{4} \rightarrow f(-\tfrac{3}{4}) > 0$$

Third iteration:

$$\frac{-3/4+1/2}{2} = -\frac{1}{8} \rightarrow f(-\tfrac{1}{8}) > 0$$

Fourth iteration:

$$\frac{-1/8 + 1/2}{2} = \frac{3}{16} = \tilde{x}$$

Since $f(0) = 0$, the error is simply

$$\frac{3}{16} - 0 = \frac{3}{16}$$

c) Recall: $E \leq \frac{1}{2^{k+1}}(b-a)$ for bisection method

We want to solve for $k$ in

$$\frac{1}{2^{k+1}}(2+3) \leq 10^{-3}$$

$$\Leftrightarrow \quad 2^{k+1} \geq \frac{5}{10^{-3}} = 5 \cdot 10^3 = 5000$$

$$\Leftrightarrow \quad k+1 \geq \log_2 5000$$

$$k \geq \log_2 5000 - 1$$

$$\approx 11.3$$

So after approximately 12 iterations, we achieve the desired

d) Try $x = 2$:

$$f(2) = (1 - 3^2)2^2 + 4(2-1)3^2 + 4(1-2)$$
$$= -8 \cdot 4 + 4 \cdot 1 \cdot 9 + 4 \cdot (-1)$$
$$= 0$$

Thus $x = 2$ is a root.

Since both sides of the root have negative function values, the algorithm will only bounce around without converging to this root. This can be formalized via the intermediate value theorem.

e) The python implementation is essentially already given in the lecture slides, so I will do it in Uiua:

Uiua — A programming language for point-free enjoyers

Support Uiua's development | Home

```
f ← ++⊃(
  ×⊃(¯-1¯ⁿ3|ⁿ2)
| ×4×⊃(-1|¯ⁿ3)
| ×4¯-1)

Bisection ← (
  ⊃⊢⊣↠⊂⊙(÷2/+)⊸⊡-1⊙⊂⊙◌∪(
    ⊙⊂ ⊙∩(±f)
    =⊸(⊢↻)
  ) ∪((±f)÷2+)|
)                tol          a b
⍢(Bisection|>1e-3 ∨f÷2+) ¯2 3
```

```
0.000732421875
¯0.00048828125
```

b/a ⇒ we approach 0

Run

Note: Uiua is not yet stable

Uiua 0.17.0-rc.2

**Problem 2** Consider

$$f(x) = x^3 + x - 1 = 0.$$

a) $f(-1) = (-1)^3 - 1 - 1 = -3 < 0,$

$f(1) = 1^3 + 1 - 1 = 1 > 0$

Since $f \in C[-1, 1]$, from I.V.T. there must be some $r \in [-1, 1]$ with

$$f(r) = 0.$$

Additionally, $f'(x) = 3x^2 + 1 > 0 \quad \forall x \in \mathbb{R}$

So f is nondecreasing, meaning the root r is unique. □

b)

```
6.  x = x0
8.  x_new = g(x)
9.  if ~ <= 1e-6   # |E| <= tol
11. x = x_new
```

The comment addresses the fact that $f(x_0)$ did not converge in the given iteration time. Thus, the returned x-value could be radically different from x_new and does not represent the proper fix-point value of $f(x_0)$, if it exists.

To calculate the upper bound of the error we use the a posteriori error bound

$$|x_k - r| \leq L^k |x_0 - r|,$$

where $r$ is the convergent value, and $|g'(x)| \leq L < 1$ for $k$ iterations.

$$g'(x) = \left(\frac{1}{1+x^2}\right)' = \frac{-2x}{(1+x^2)^2} \leq \frac{-2x_0}{(1+x_0^2)^2} < 1,$$

So let $L = \frac{2x_0}{(1+x_0^2)^2}$, $x_0 = 1/2$, $r = 0$

then $|E| \leq (1+2^{-2})^{-2k} \cdot \frac{1}{2} = (5/4)^{-2k} \cdot \frac{1}{2}$

$$= \underline{\frac{1}{2}(4/5)^{2k}}$$

**Problem 3** Let for $a \in \mathbb{R}$

$$\phi_a(x) = (x^3 - 3a^2x - 1)^2 + (3ax^2 - a^3)^2$$

a) let $x_0 = 0.75 = 3/4$. $a = 0$:

$$\phi_0(x) = (x^3 - 1)^2$$

$$\Rightarrow \phi_0'(x) = 6x^2(x^3 - 1) = 6x^5 - 6x^2$$

first iteration:

$$\phi_0(x_0) = \left((3/4)^3 - 1\right)^2 = \left(\frac{27}{64} - 1\right)^2$$

$$= \left(-\frac{37}{64}\right)^2 = \frac{1369}{4096} \approx 0.334$$

$$\phi_0'(x_0) = 6(3/4)^5 - 6(3/4)^2$$

$$= 6 \cdot \frac{243}{1024} - 6 \cdot \frac{9}{16}$$

$$= 6\left(\frac{243 - 576}{1024}\right)$$

$$= 6\left(\frac{-233}{1024}\right)$$

$$= -\frac{699}{512} \approx -1.37$$

$$\Rightarrow x_1 = x_0 - \phi_0(x_0) : \phi_0'(x_0)$$

$$= 3/4 + \frac{1369}{4096} \cdot \frac{512}{699}$$

$$= \frac{5563}{5592} \approx 0.995$$

This is wrong

Second iteration:

$$\phi_0(x_1) = \left(\left(\frac{5563}{5592}\right)^3 - 1\right)^2 = \frac{7324850714133791881}{30577570518753817657344}$$

$$\approx 0.000240$$

So let's do it numerically instead;

Uiua *It's got um...I um...arrays* Support Uiua's development Home

```
# ? a x
Φ ← +∩ⁿ₂⊃(
  -⊃(ⁿ3|×3×⊙ⁿ₂)
| -1-⊃(×3×ⁿ2|ⁿ3•○))
Φ₀  ← Φ0
δΦ₀ ← ×6×⊃(ⁿ2|-1ⁿ3)

# ? f f' x0
Newton!! ← -⊸(÷⊃(^1)^0)
ÖNewton!!Φ₀δΦ₀1 3/4
ÖNewton!!Φ₀δΦ₀2 3/4
```

first iteration

Second

```
0.9212962962962963
0.9641055913914133
```

Run

Note: Uiua is not yet stable Uiua 0.17.0-rc.2

Uiua *It's got um...I um...arrays* Support Uiua's development Home

```
# ? a x
Φ ← +∩ⁿ₂⊃(
  -⊃(ⁿ3|×3×⊙ⁿ₂)
| -1-⊃(×3×ⁿ2|ⁿ3•○))
Φ₁ ← Φ1
# 6 |(1 + x - x^2 + 2 x^3 + x^5)
δΦ₁ ← ×6+1+++⊃(○|⁻ⁿ2|×2ⁿ3|ⁿ5)

# ? f f' x0
Newton!! ← -⊸(÷⊃(^1)^0)
ÖNewton!!Φ₁δΦ₁1 3/4
ÖNewton!!Φ₁δΦ₁2 3/4
```

$a = 1$

$\Phi_1'(x) := 6(1 + x - x^2 + 2x^3 + x^5)$

```
0.12765461328741567
¯0.2923839784989345
```

Run

Note: Uiua is not yet stable Uiua 0.17.0-rc.2

Let $F: \mathbb{R}^2 \to \mathbb{R}^2$,

$$(x,y) \mapsto (x^3 - 3xy - 1,\ 3x^2y - y^3)$$

b) Show that

$$(x^3 - 3xy^2 - 1)^2 + (3x^2y - y^3)^2 = 0 \quad : X$$

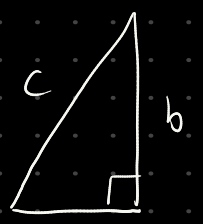
$$\Leftrightarrow x^3 - 3xy^2 - 1 = 0 \wedge 3x^2y - y^3 = 0 \quad : Y$$

i) assume $Y$, show $X \Leftarrow Y$:

$0^2 + 0^2 = 0$ Ok

ii) assume $X$, show $X \Rightarrow Y$:

by pythagorean theorem, if $c$ is to be zero, both $a$ and $b$ must be too. Ok




c) recall $J(x,y) = DF(x,y)$

$$\Rightarrow J(x,y) = \begin{bmatrix} \frac{\partial F_1}{\partial x} & \frac{\partial F_2}{\partial x} \\ \frac{\partial F_1}{\partial y} & \frac{\partial F_2}{\partial y} \end{bmatrix}$$

$$= \begin{bmatrix} 3x^2 - 3y^2 & 6xy \\ -6xy & 3x^2 - 3y^2 \end{bmatrix}, \text{ then}$$

$$\begin{bmatrix} x \\ y \end{bmatrix}_{k+1} = \begin{bmatrix} x \\ y \end{bmatrix}_k - J(x_k, y_k)^{-1} F(x_k, y_k),$$

because we can generalize newton's method through first-order taylor expansion of $F_1$ and $F_2$ s.t.

$$F_1(x, y) \approx F_1(x_k, y_k) + \frac{\partial F_1}{\partial x}(x - x_k) + \frac{\partial F_1}{\partial y}(y - y_k)$$

and similarly for $F_2$.

Analogous to the tangent lines of scalar newton's method, these are the tangent planes to our surfaces $F_1$ and $F_2$. We just established that these need to be zero, so

$$F_1(x_k, y_k) + \frac{\partial F_1}{\partial x}(x - x_k) + \frac{\partial F_1}{\partial y}(y - y_k) = 0$$

$$\wedge\ F_2(x_k, y_k) + \frac{\partial F_2}{\partial x}(x - x_k) + \frac{\partial F_2}{\partial y}(y - y_k) = 0$$

$$\Leftrightarrow F\begin{bmatrix} x_k \\ y_k \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{\partial F_1}{\partial x} & \frac{\partial F_1}{\partial y} \\ \frac{\partial F_2}{\partial x} & \frac{\partial F_2}{\partial y} \end{bmatrix}}_{\text{Jacobian}} \cdot \begin{bmatrix} x - x_k \\ y - y_k \end{bmatrix} = \vec{0}$$

$$\Leftrightarrow \begin{bmatrix} x - x_k \\ y - y_k \end{bmatrix} = -J^{-1}(x_k, y_k)\, F(x_k, y_k)$$

$$\Leftrightarrow \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}_k - J^{-1}(x_k, y_k) F(x_k, y_k)$$

where $\begin{bmatrix} x \\ y \end{bmatrix}$ is a point we can use further so $\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}_{k+1}$ □

d) $F(x,y) = 0 \Leftrightarrow F_1(x,y) = 0$

$\wedge F_2(x,y) = 0$

$$3x^2 y = y^3 \Leftrightarrow x = \pm\sqrt{\frac{1}{3} y^2}$$

$$= \pm \frac{\sqrt{3}}{3} y \wedge y \neq 0$$

$$\rightarrow \left(\pm \frac{\sqrt{3}}{3} y\right)^3 \mp \sqrt{3} y^3 = 1$$

$$\Leftrightarrow y^3 \left(\pm \left(\frac{\sqrt{3}}{3}\right)^3 \mp \sqrt{3}\right) = 1$$

$$\Leftrightarrow y = \frac{1}{\sqrt[3]{\left(\pm \left(\frac{\sqrt{3}}{3}\right)^3 \mp \sqrt{3}\right)}} = \pm \frac{\sqrt{3}}{2}$$

$$\rightarrow x = \pm\sqrt{\frac{1}{3} \frac{3}{4}} = \pm \frac{1}{2}$$

yields four candidates:

$F\left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right) \neq 0$

$F\left(\frac{-1}{2}, \frac{\sqrt{3}}{2}\right) = 0$

$F\left(\frac{1}{2}, \frac{-\sqrt{3}}{2}\right) \neq 0$

$F\left(\frac{-1}{2}, \frac{-\sqrt{3}}{2}\right) = 0$

but also $y = 0 \Rightarrow x = 1$, so check $F(1, 0) = 0$

e) Again I turn to the moon-runes of Uiua:

```
# Experimental!
~ "git: github.com/Omnikar/uiua-math" ~ MInv Mvp

# ? y x
F ← ⊃(
  ¯⊃(ⁿ3|×3×⊙ⁿ₂)
| ¯1¯⊃(×3×ⁿ2|ⁿ3•○))
# ? a x
Φ ← +∩ⁿ₂F
# ? y x
J ← ⊃[
  ⊃[×3¯∩ⁿ₂|×6×]
| ⊃[×¯6×|×3¯∩ⁿ₂]]

# ? f f' x0
Newton!! ← ¯⊸(÷⊃^1^0)
# ? x+yi
VectorNewton ← ¯⊸(/ℂ˜Mvp ⊟F°ℂ ⊶(MInv J°ℂ))

$first öVectorNewton1 ˜ℂ0.8 0.1
$second öVectorNewton2 ˜ℂ0.8 0.1
$first öVectorNewton1 ˜ℂ¯0.6 0.7
$second öVectorNewton2 ˜ℂ¯0.6 0.7
$first öVectorNewton1 ˜ℂ¯0.6 ¯0.7
$second öVectorNewton2 ˜ℂ¯0.6 ¯0.7

# Iteration
first: 1.0303747534516765-0.05956607495069O3i
second: 0.9977562079241812-0.0036508508493944 2i
first: ¯0.45997693194925027+0.8542099192618224i
second: ¯0.5015799675805404+0.8651358310710837i
first: ¯0.45997693194925027-0.8542099192618224i
second: ¯0.5015799675805404-0.8651358310710837i
```

Run

Note: Uiua is not yet stable

Uiua 0.12.0-rc.2

(0.8, 0.1)

(-0.6, 0.7)

(-0.6, -0.7)

(1, 0)

$(-\frac{1}{2}, \frac{\sqrt{3}}{2})$

$(-\frac{1}{2}, -\frac{\sqrt{3}}{2})$

I represent a vector as a complex number, so the output shows $x + y\,i$.

f) Using the above implementation (it is already in $\mathbb{R}^2$):

```
# ? f f' x0
Newton!! ← -⊸(÷⊃^1^0)
# ? x+yi
VectorNewton ← -⊸(/ƈ˜Mvp ⊟F°ƈ ⊶(MInv J°ƈ))

.öVectorNewton100 ˜ƈ0.5 0.5
≍[¯1/2 ÷2√3] ˜ƈ°ƈ

¯0.5+0.8660254037844386i
1
```

Run

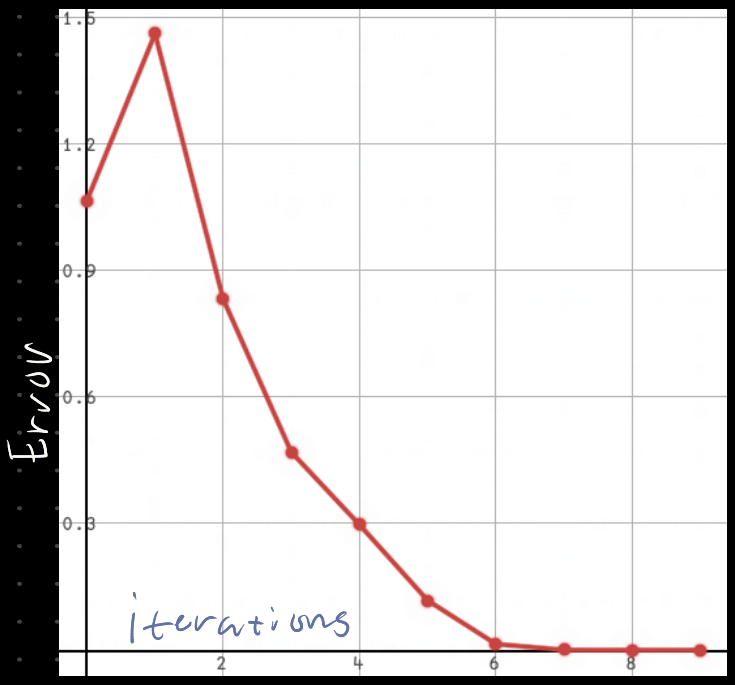
Note: Uiua is not yet stable — Uiua 0.17.0-rc.2

Check if is root

Converges to $(-\frac{1}{2}, \frac{\sqrt{3}}{2})$

```
Plot Data!(
  ↑10
  New ⧈⊟⊶(
    ⌄-ƈ÷2√3 ¯1/2 ≡(
      öVectorNewton⊙(˜ƈ0.5 0.5)
    )
  )
)
```

```
# Experimental!
~ "git: github.com/Omnikar/uiua-math" ~ MInv Mvp
~ "git: github.com/Omnikar/uiua-plot" ~ Data Plot
```

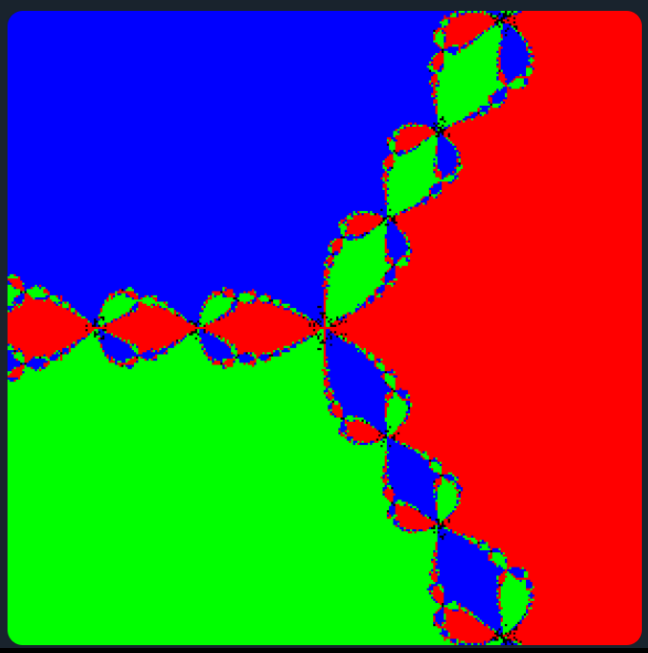



The plot shows a stable convergence, my hunch tells me it converges linearly :)

g) Here's the newton fractal:

```
20 ⍥⇌(⊂2)−2÷÷4⍜↑300
21 ≡₀(⁘⁖(⊙VectorNewton+1|<50)0)˙⊞⊂
22 <1e−3⌄−⊙¤[1 ⊂÷2√3 ¯1/2 ⊂¯÷2√3 ¯1/2]
23 ♪1_2
24 ▽₂2
25
```

Run



We can observe a few black spots because they don't converge in 50 iterations.
Note: this computation is slow because I am not terminating early (so 50 iterations per pixel).

---

That's it. I didn't get time for 1f) or 2c).
Also the fact that I didn't use python...