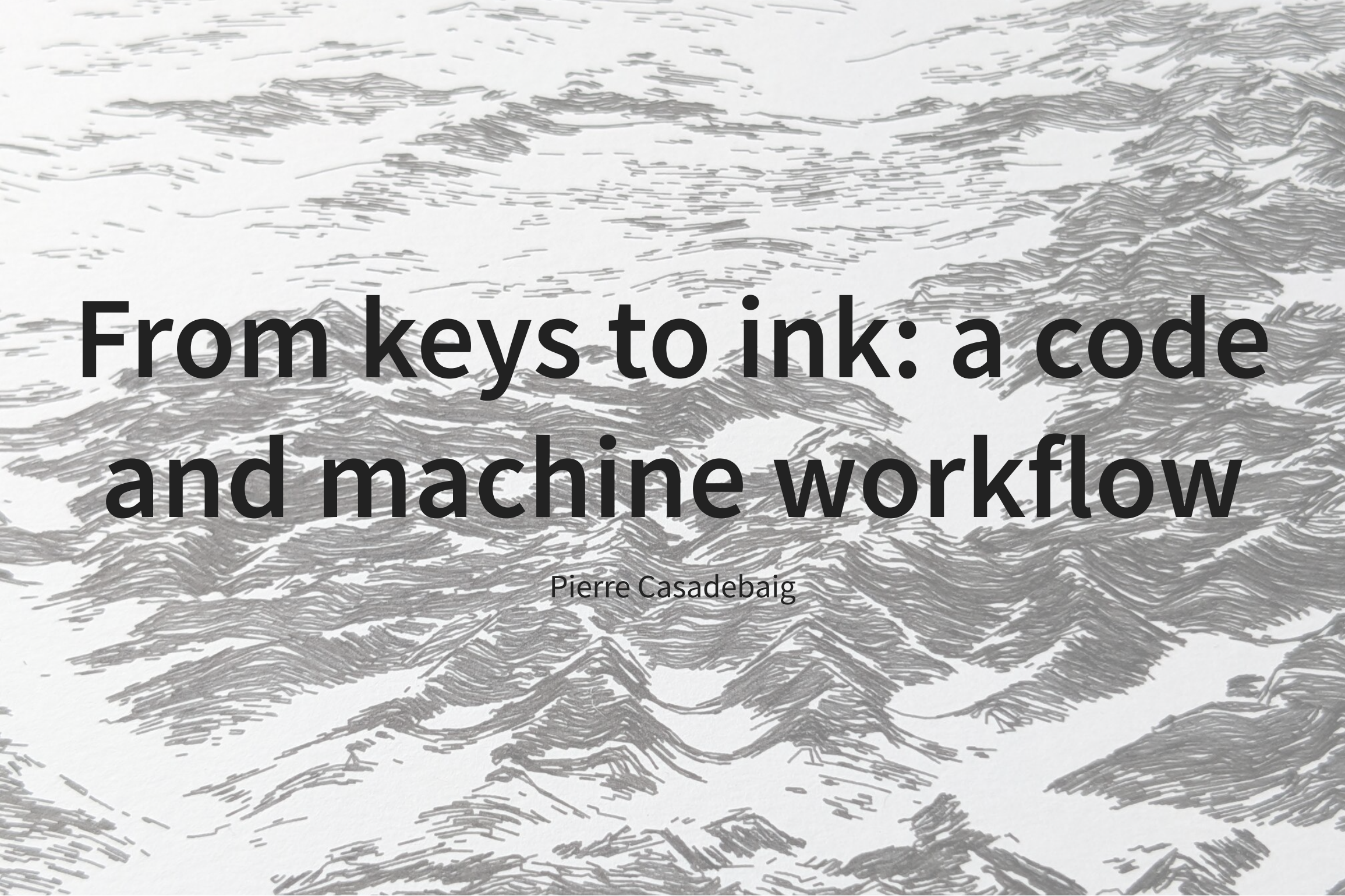
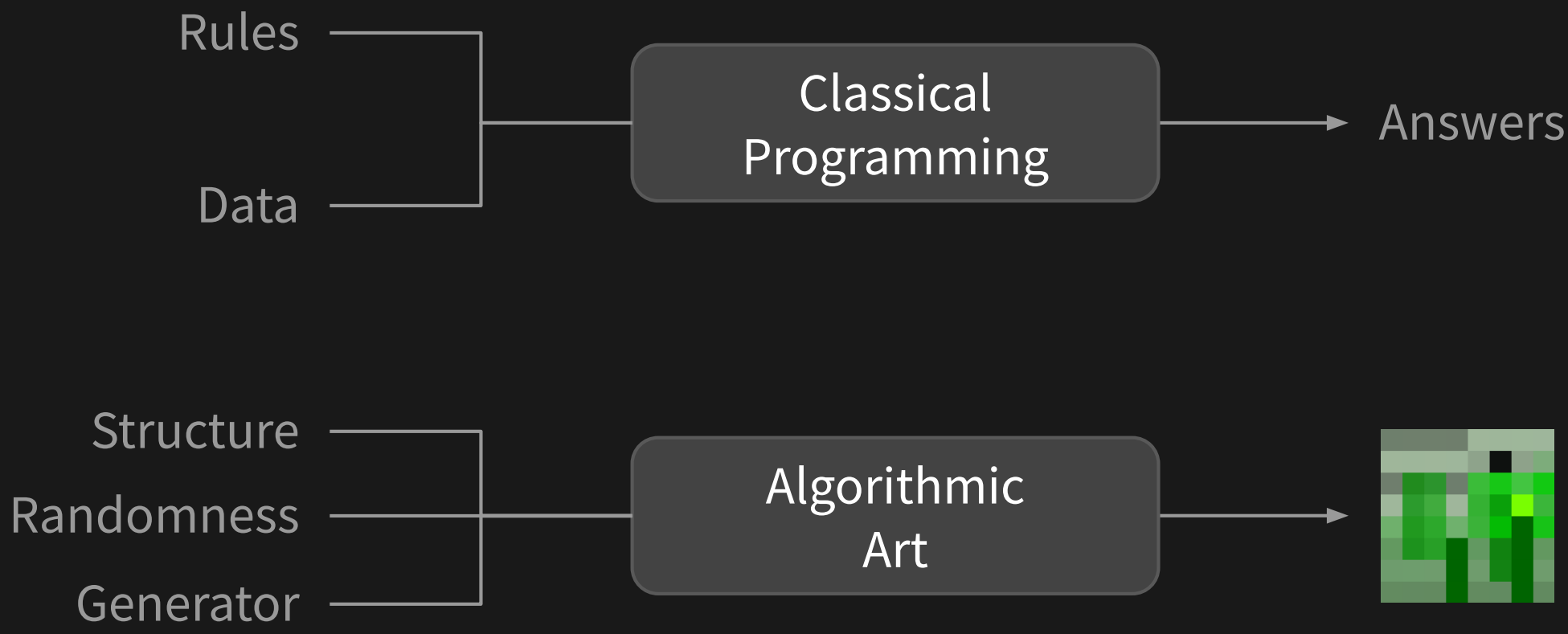


From keys to ink: a code and machine workflow

Pierre Casadebaig

An illustration of generative art

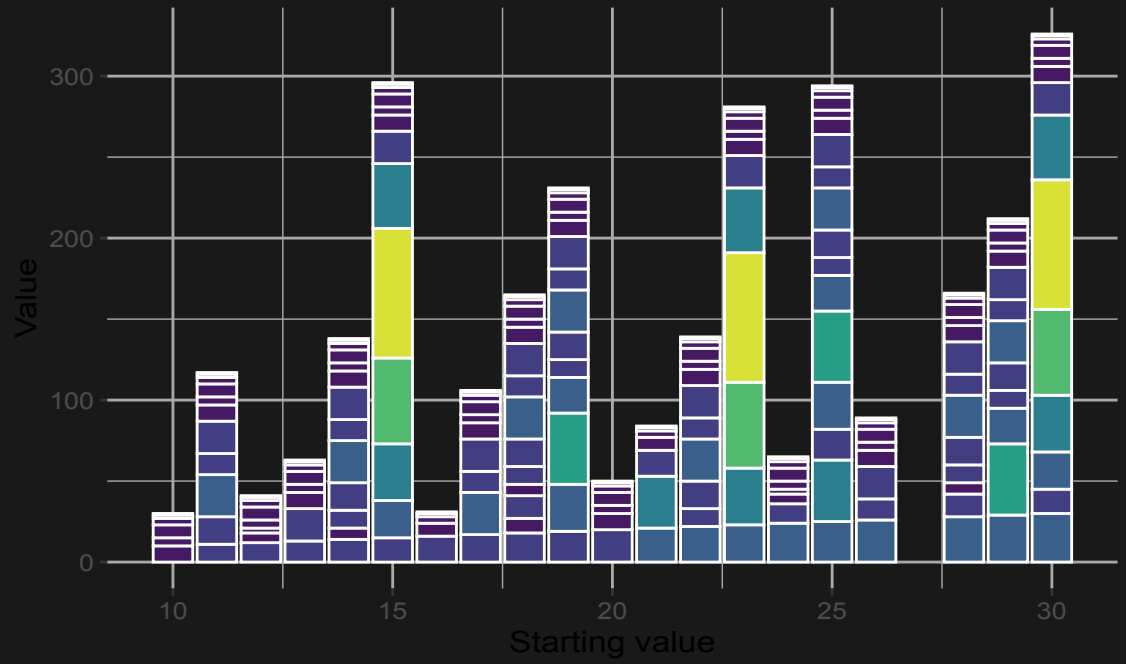


loosely adapted from Chollet2018

A simple rule for stem-like structures

$$n+1 = \begin{cases} \frac{n}{2} & \text{if } even \\ 3n+1 & \text{if } odd \end{cases}$$

Collatz sequence: e.g. 10 → 5 → 16 → 8 → 4 → 2 → 1



Shiffman2019, Miller2020

From numbers to geometry

structure

- number value → lengths
- number parity → angles

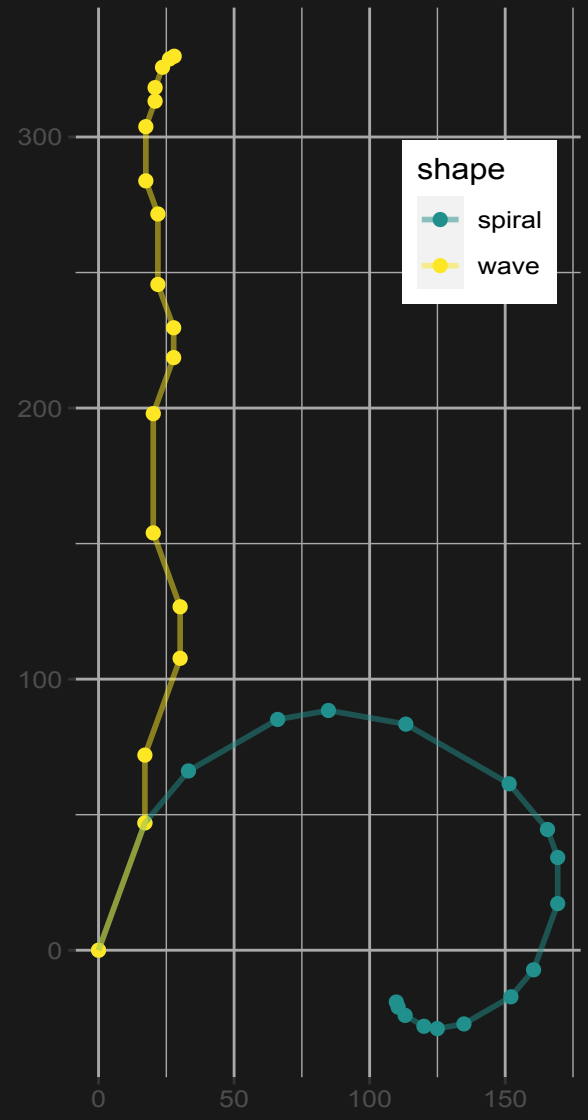
```
# recurse on Collatz function
seq_collatz <- function(i, max = 1000) {
  accumulate(1:max, ~ collatz(.), .init = i)
}

# compute sequence for i = 50
seq_collatz(50) |> head()
```

```
[1] 50 25 38 19 29 44
```

```
# create a table with xy coordinates
gen_leaf(i = 50, a = 20, shape = "spiral") |> head()
```

```
# A tibble: 6 × 8
      n     s length angle     x     y  xend  yend
  <int> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1     1    50     50    20   0     0    17.1  47.0
2     2    25     25    40  17.1  47.0  33.2  66.1
3     3    38     38    60  33.2  66.1  66.1  85.1
4     4    19     19    80  66.1  85.1  84.8  88.4
5     5    29     29   100  84.8  88.4 113.   83.4
6     6    44     44   120 113.   83.4 151.   61.4
```



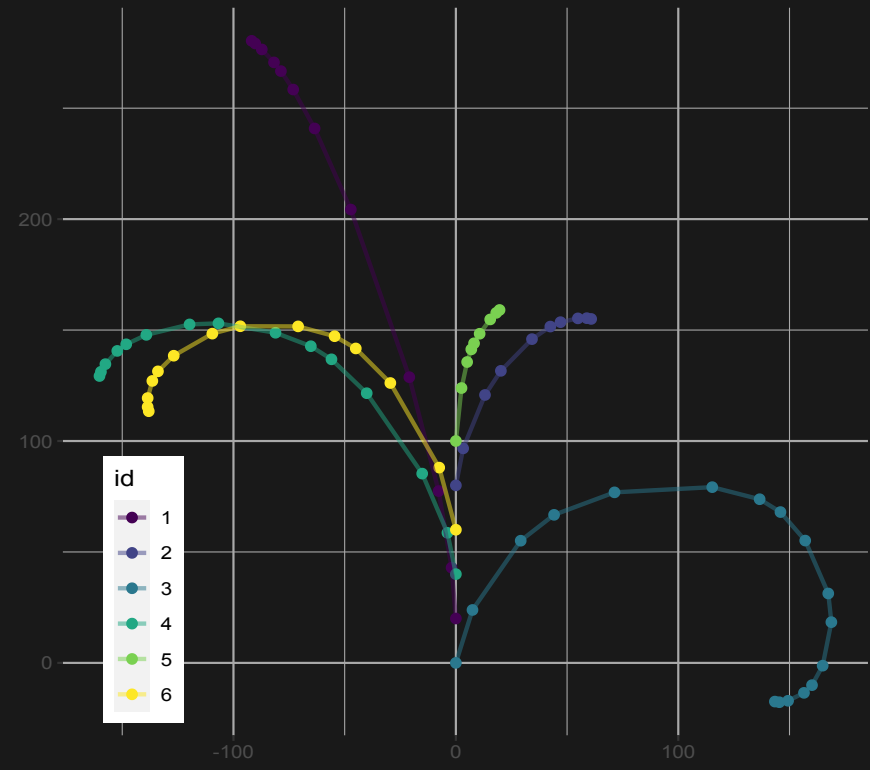
Iterate before high energy costs!

> **structure** → shifting shapes
> **randomness** → starting value, angle

```
 1  n = 6
 2
 3  # iterate leaf generator function
 4  tibble(
 5    id = seq_len(n),
 6    i = runif(n, 10, 30) |> as.integer(),
 7    a = runif(n, -20, 20)) |>
 8    mutate(
 9      path = map2(i, a, ~ gen_leaf(i = ..1, a = ..2))
10    )
```

```
# A tibble: 6 × 4
     id     i      a path
  <int> <int>  <dbl> <list>
1     1    15  17.8  <tibble [13 × 8]>
2     2    17   6.43 <tibble [10 × 8]>
3     3    21   5.16 <tibble [7 × 8]>
4     4    28 -17.5  <tibble [14 × 8]>
5     5    14 -11.8  <tibble [13 × 8]>
6     6    27 -12.9  <tibble [71 × 8]>
```

```
 1  # create a node generator function
 2  data_node <- gen_node(
 3    n, imin = 10, imax = 30, amin = -20, amax = 20,
 4    shift = c(0, 20), method = "path")
```

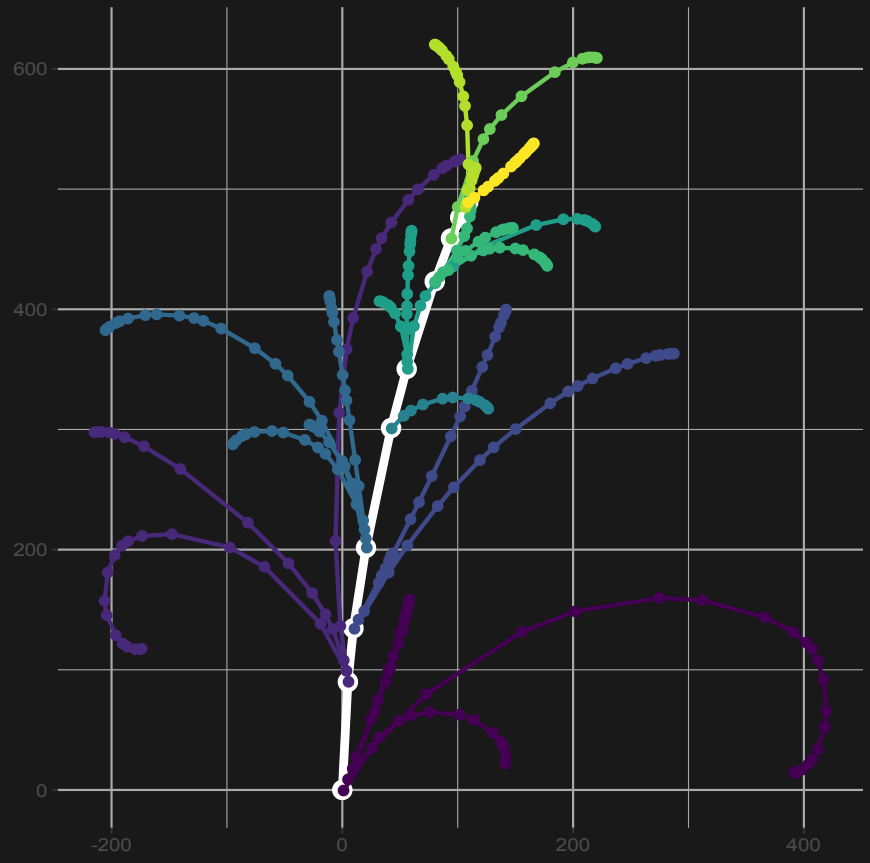


(Re)iteration is a thing in botany

> **structure** → node angle (phyllotaxy)
> **randomness** → enough

```
# merge stem and nodes attributes
left_join(topology_stem, topology_node, by = "node") |>
  mutate(
    data = pmap(
      list(leaf, amin, amax, scale),
      ~ gen_node(n = ..1, amin = ..2, amax = ..3, scale =
    )) |>
  select(-data)
```

```
# A tibble: 8 × 8
   node     x0    y0       a  leaf   amin  amax scale
  <int>  <dbl> <dbl>   <dbl> <dbl>  <dbl> <dbl> <dbl>
1     1  0       0    0.406      5 -20    20    1
2     2  0.523  15.0 -0.357      4 -17.9  17.9  0.886
3     3  2.13   37.9  0.415      4 -15.7  15.7  0.771
4     4  5.79   72.7 -0.211      3 -13.6  13.6  0.657
5     5 13.2   125.   0.524      3 -11.4  11.4  0.543
6     6 27.1   204.  -0.107      2  -9.29  9.29 0.429
7     7 35.4   243.   0.565      2  -7.14  7.14 0.314
8     8 40.2   263.  -0.0491     2  -5     5    0.2
```



[Hallé1970, Barthélémy2007]

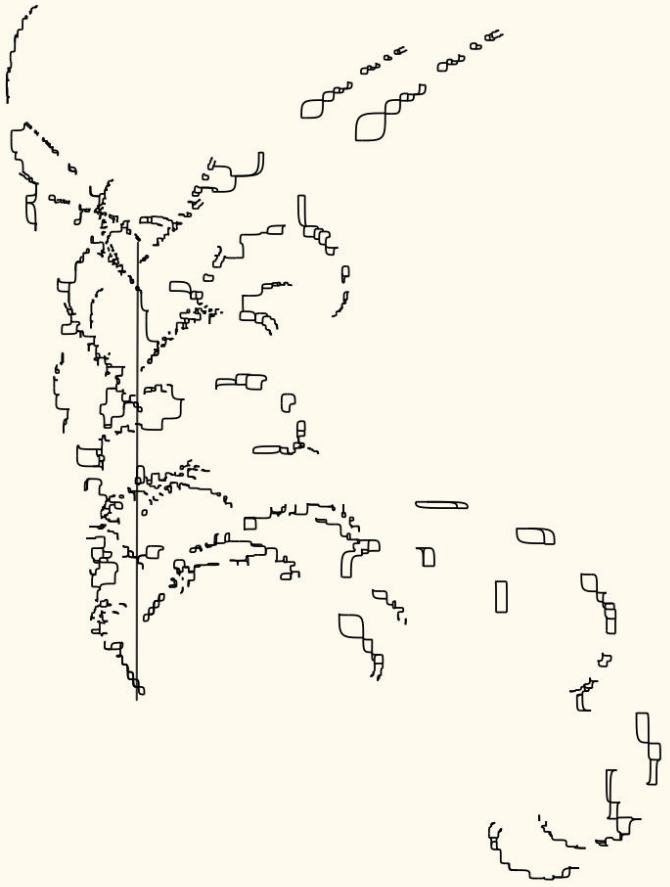
Same data, different visualization

Polygons

an efficient way to add realism.

Graphs

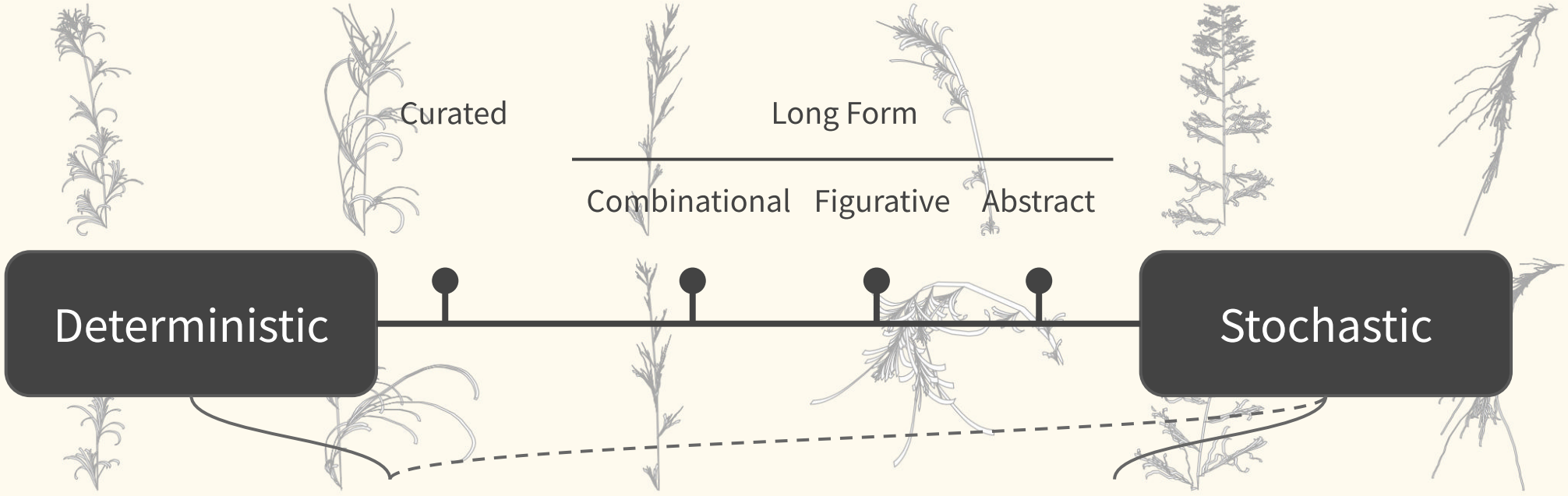
thinking in relations rather than individuals ?



Technically, the separation of data from its representation is well implemented in dataviz languages (Wilkinson2011,

Wickham2010).

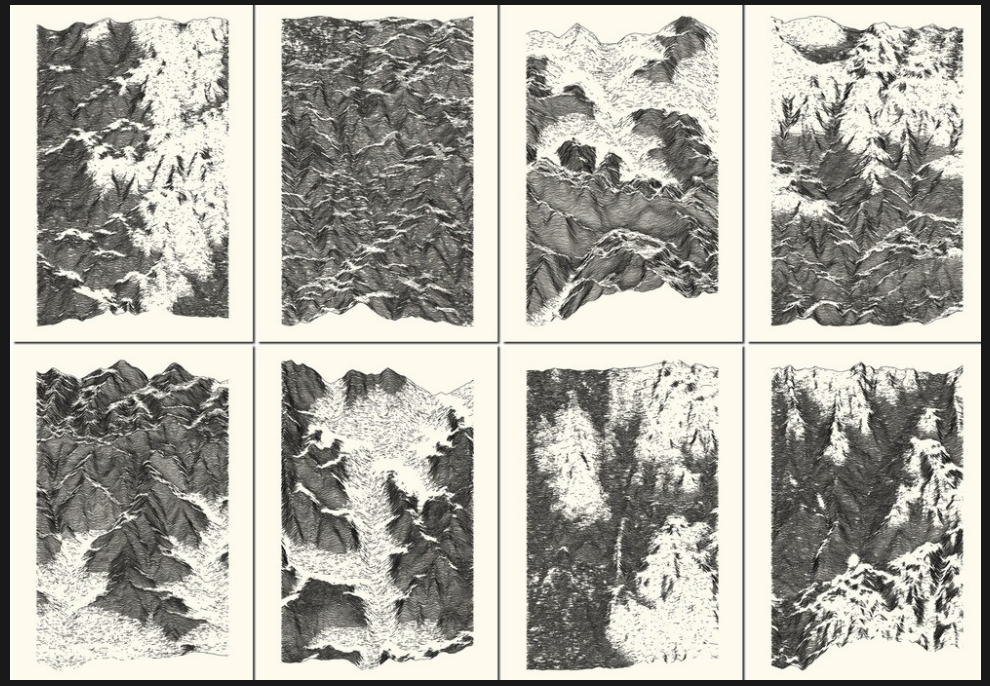
Calibrating the output space

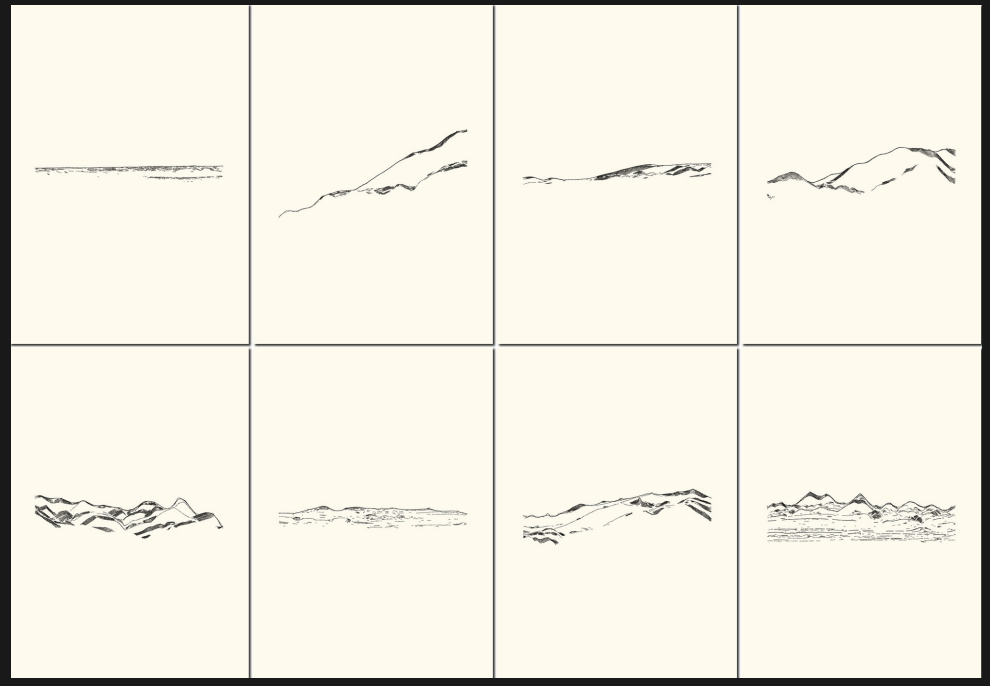


Giving away a level of control to the generative system (Anyene2022).

Improve the Gem:Garbage ratio[1]

- Self-learning (observation, trial and error, systematic analysis)
- Machine-learning (predict how features would impact aesthetics, optimization)
- How to involve peoples in this calibration process ? (Roux2022)

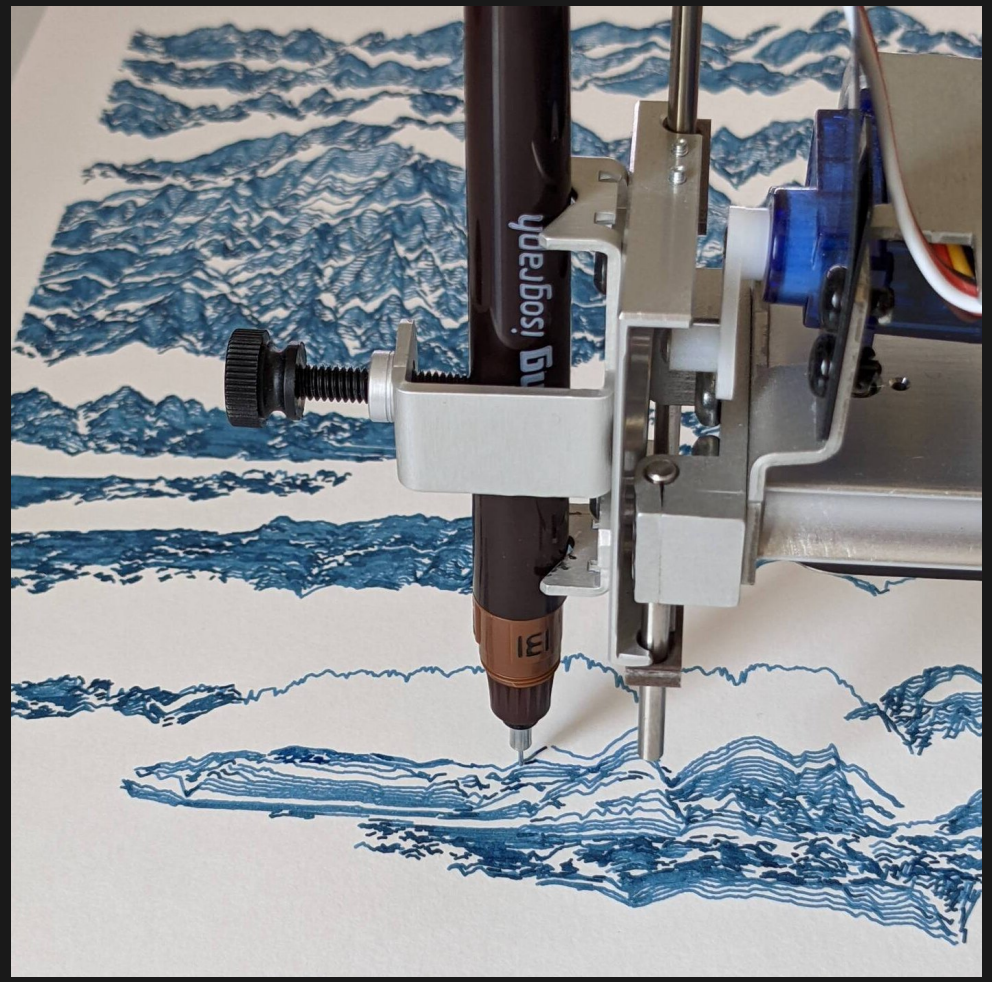




[1]Hobbs2021

Getting back to paper

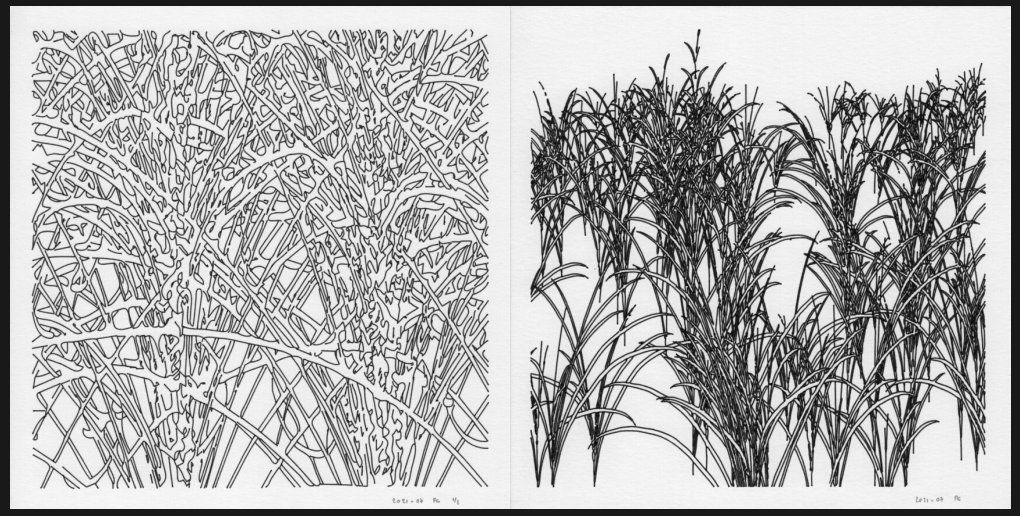
- not printing, design your work around it:
  - Catt, geometry
  - Haber, acrylic paint
  - He, watercolor
  - Zancan, lines
- translate your work into a vector image
  - XY precise movements
  - only up / down state (no pressure)

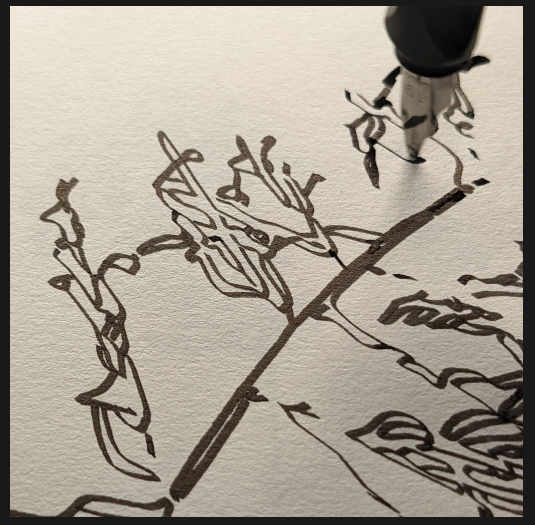


Drawings from the “stem” system

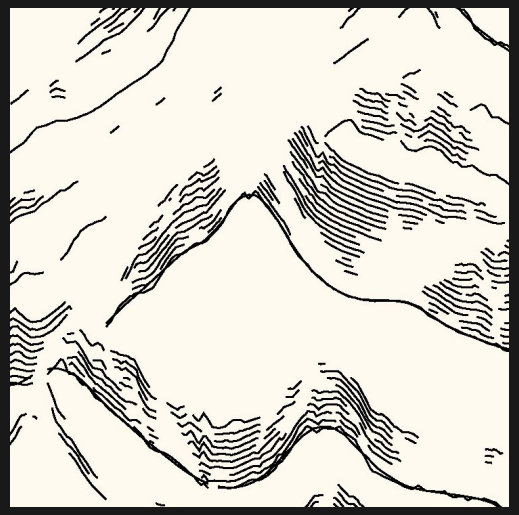
Hiding shapes by overlaying won’t help

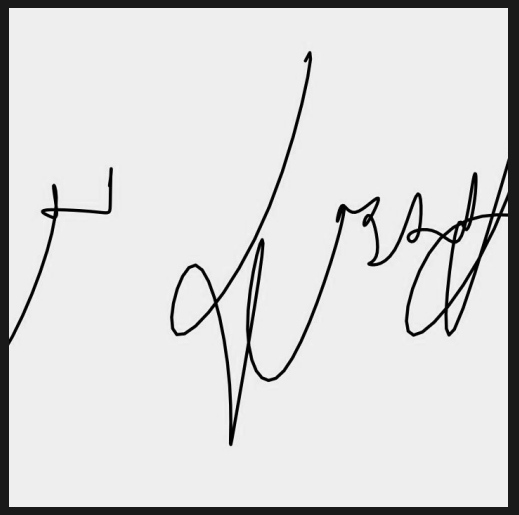
- code (more control)
- geometry libs
- post-processing (e.g. vpype)

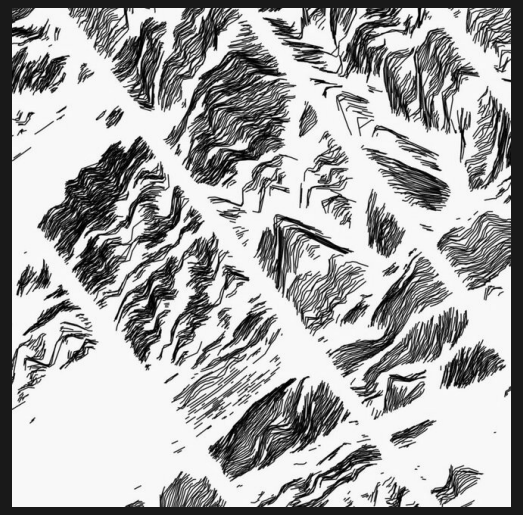


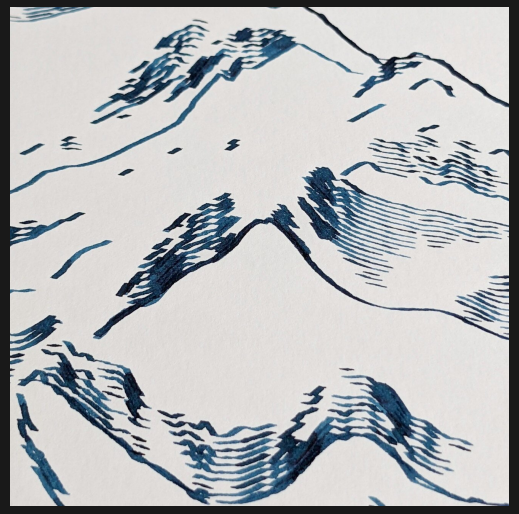


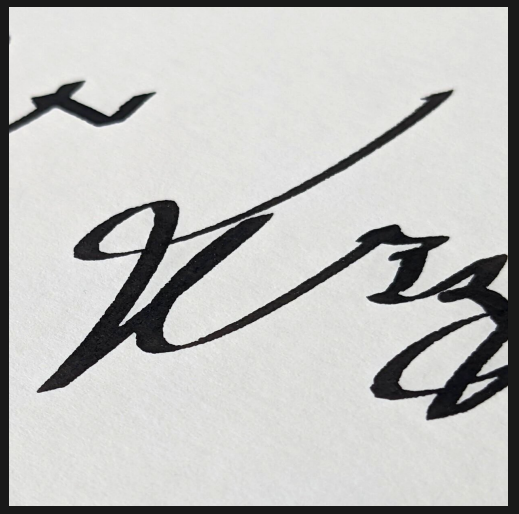
Code a line, get a shape.

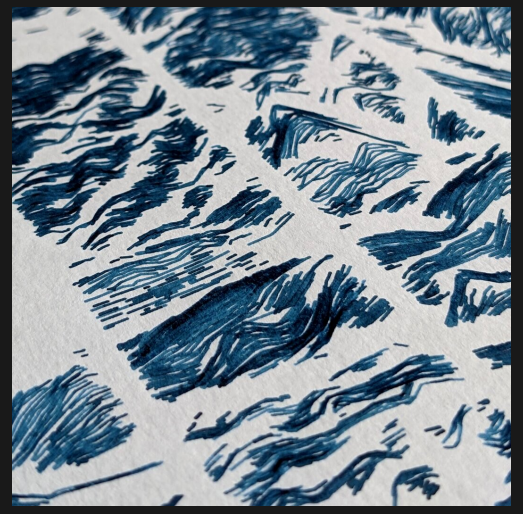












Thank you for your attention!

#rtistry!
Sharla Gelfand, Meghan Harris, Georgios Karamanis, Danielle Navarro, Ijeamaka Anyene Fumagalli, Thomas Lin Pedersen, Antonio Sánchez Chinchón, George Savva, Jacquie Tran, Claus Wilke

Starting with R?
Ijeamaka's intro Anatomy of generative art and Danielle's workshop Art from code are perfect.

It's a complex and fast-evolving crypto-art world, take the time to look at other algorithmic art communities not always into NFTs.

R code from @geokaramanis