markovPaint

from tables to pixels

a project by Agustin Ramos Anzorena
triggered at C7

isAbout

MarkovPaint...

... is about tables, grids and pixels.

... is about transforming a perfectly logical arrangement of numbers, into a perfectly coloured canvas of digital art.

... is about creating a unique probability-based artistic finger-print.

... is about showing that numbers can be fun and beautifull too.

MarkovPaint is a little experiment that wishes to interpret results coming out of mathematical processes and translate them into visual graphics, shifting our approach and interpretation of the numbers.

They are mental no more.

Stripped from logic, they are percieved visually, providing totally different sensations.

Contents

wasBorn

I recently started working at Continente 7, a consulting company that specializes in working with numbers and complexity to run scenario simulations for sales forecasts, among other services.
Every friday, it's "compulsory" for the employees to find a project or research on a topic we like, where we will be able to "squeeze our brains" and come out with results (such as a specific algorithm or a methodology) that could be used for later application on company projects.

I jumped into a project that intended to create a word/phrase generator, with the ability to learn and improve it's generation.
Research on text generation techniques begun. Among others, I found Disassociated Press and started to code on that. The outcomes were pretty interesting, but I didn't really see a lot of future for advance application.
Other techniques I research were the Dadaist Cut and Paste, Neural Networks, and Probabilistic Markov Chains methods.

The objective was to use one of these techniques for graphical purposes.

wasThought

MarkovPaint is the result of the investigation and application of Markov Chains to generate custom graphics.

MarkovPaint is a Computer Program.
The user can type his name (or any other phrase) which will be translated into a visual representation, with various graphical functions.

Representations will be as unique as the probability the characters have of appearing in the order they are typed in.

This happens because the characters are plotted and ran through Markov Chain Matrices.
The process spits out some values for each character, which are then interpreted as coordinates for 2 dimensional drawing, with a 3rd dimension translated as a Depth Map, and used to control various properties of the drawing.

Add some very weak randomness to this, and no representation will ever be identical, but at the same time will follow a very strict set of rules.

These rules are initially set by analizing the RAE Dictionary (The Dictionary of the Royal Spanish Academy), and evolve as the characters typed are read.

isRendered

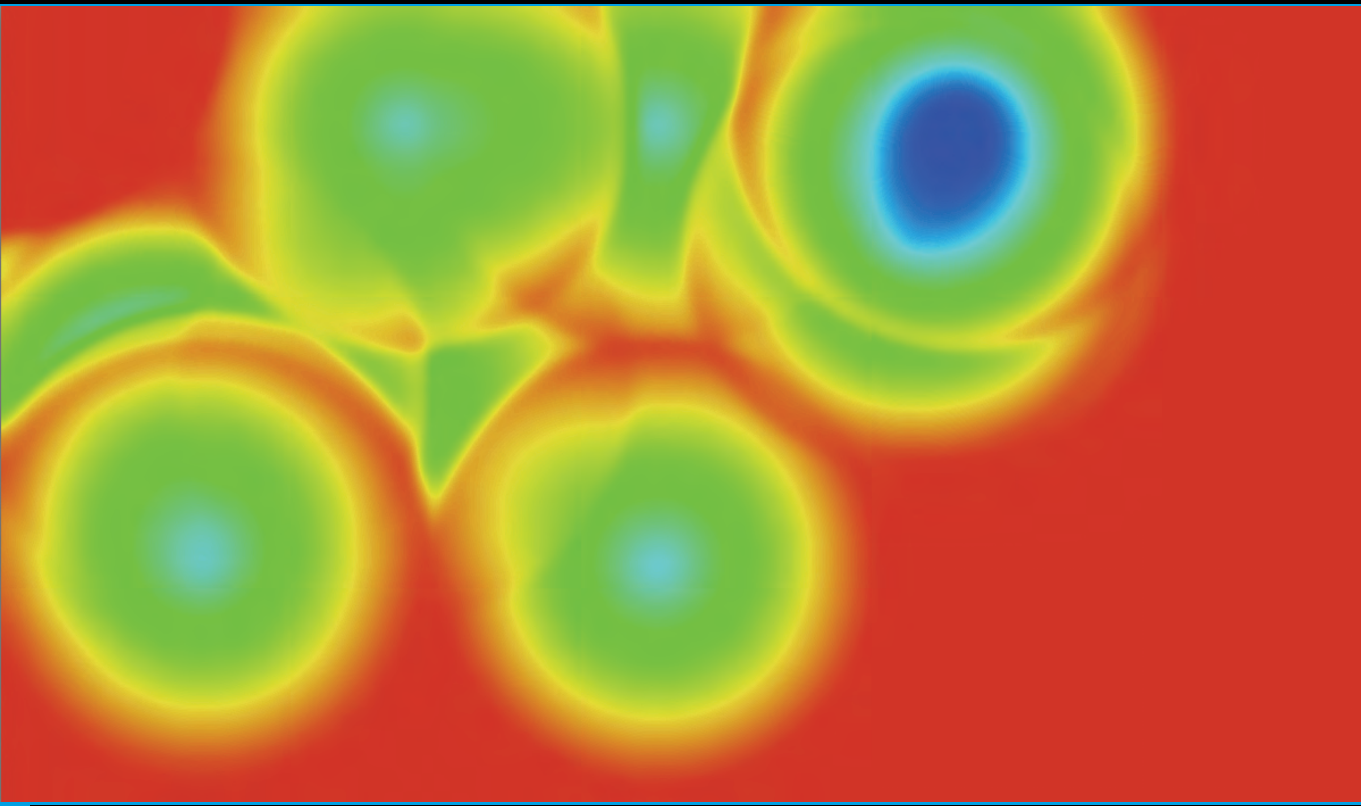
The following gallery is a showcase of some of the outputs generated for the input:

"Continente Siete"

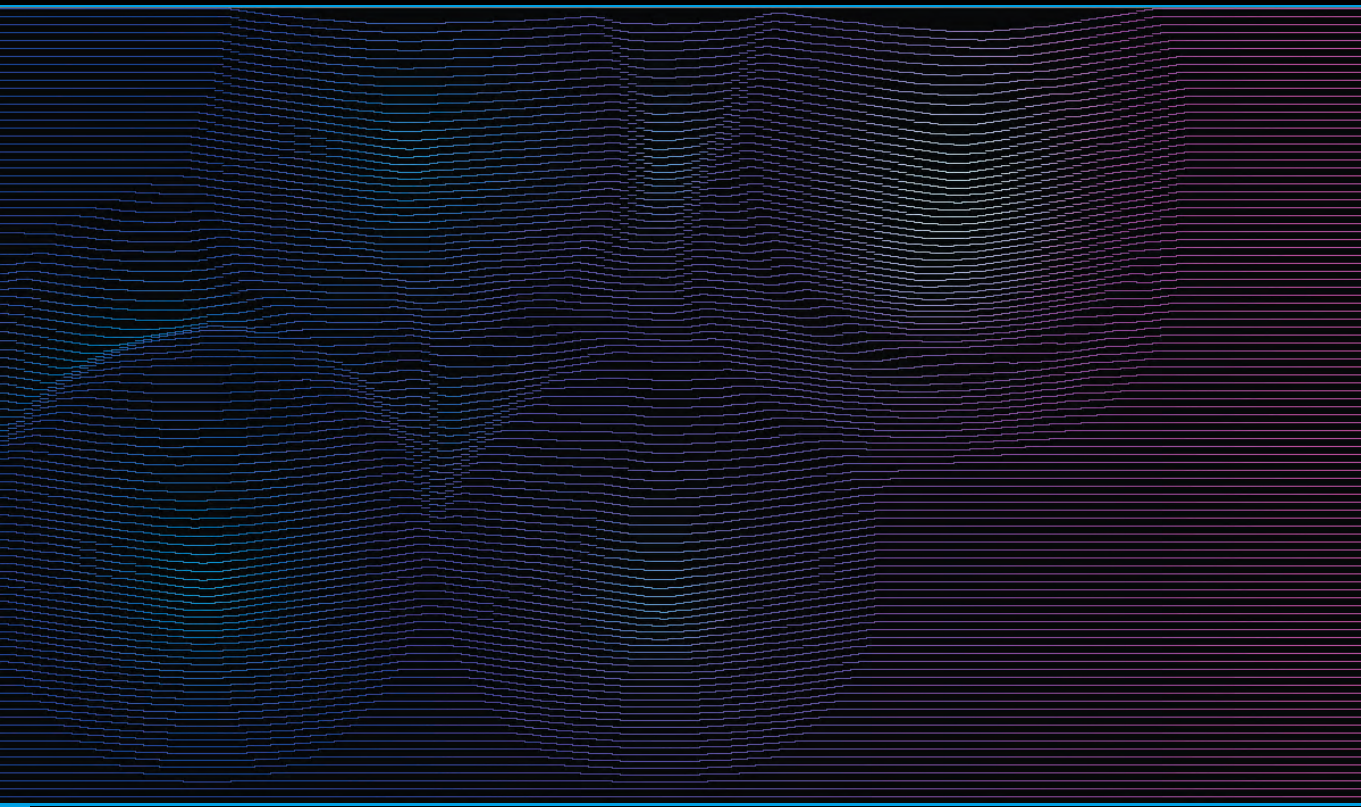
The amount of outputs can be infinite, because the amount of graphical function that may interpret the values from the matrices can be infinite too. This simply depends on the creativity of the coder.

As you will see, some graphical representations clearly give away the underlying coordinates for the input, and others appear to have no connection at all between the math and the drawing.

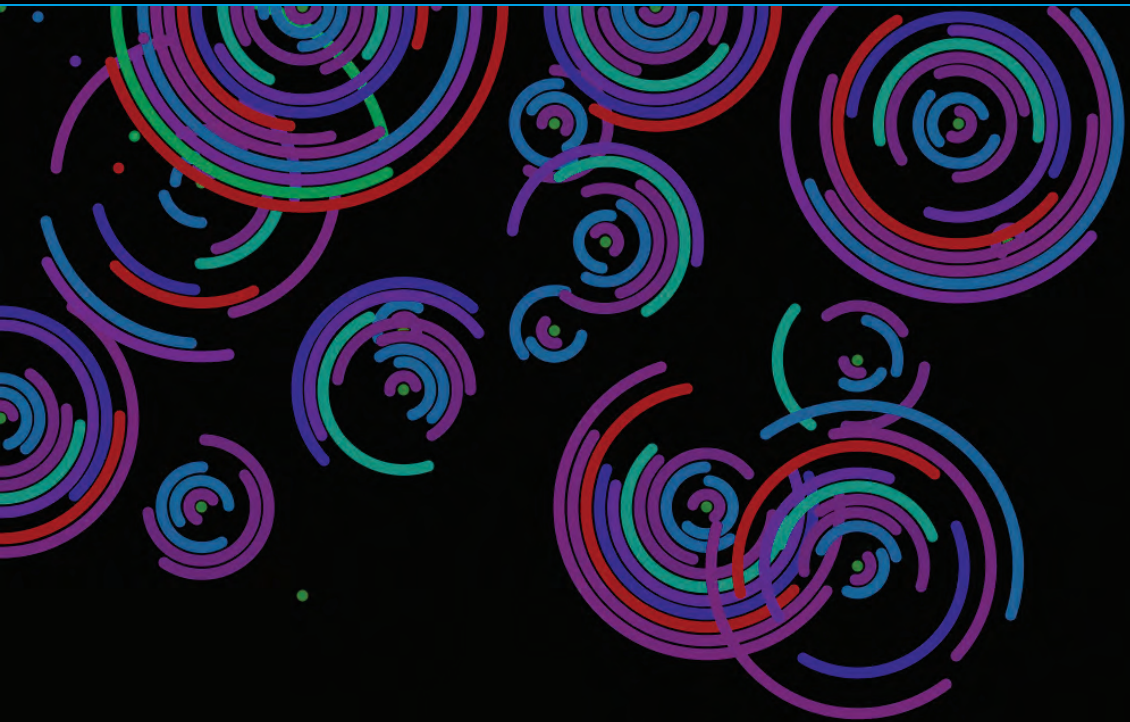
All graphics are coded and programmed in Processing



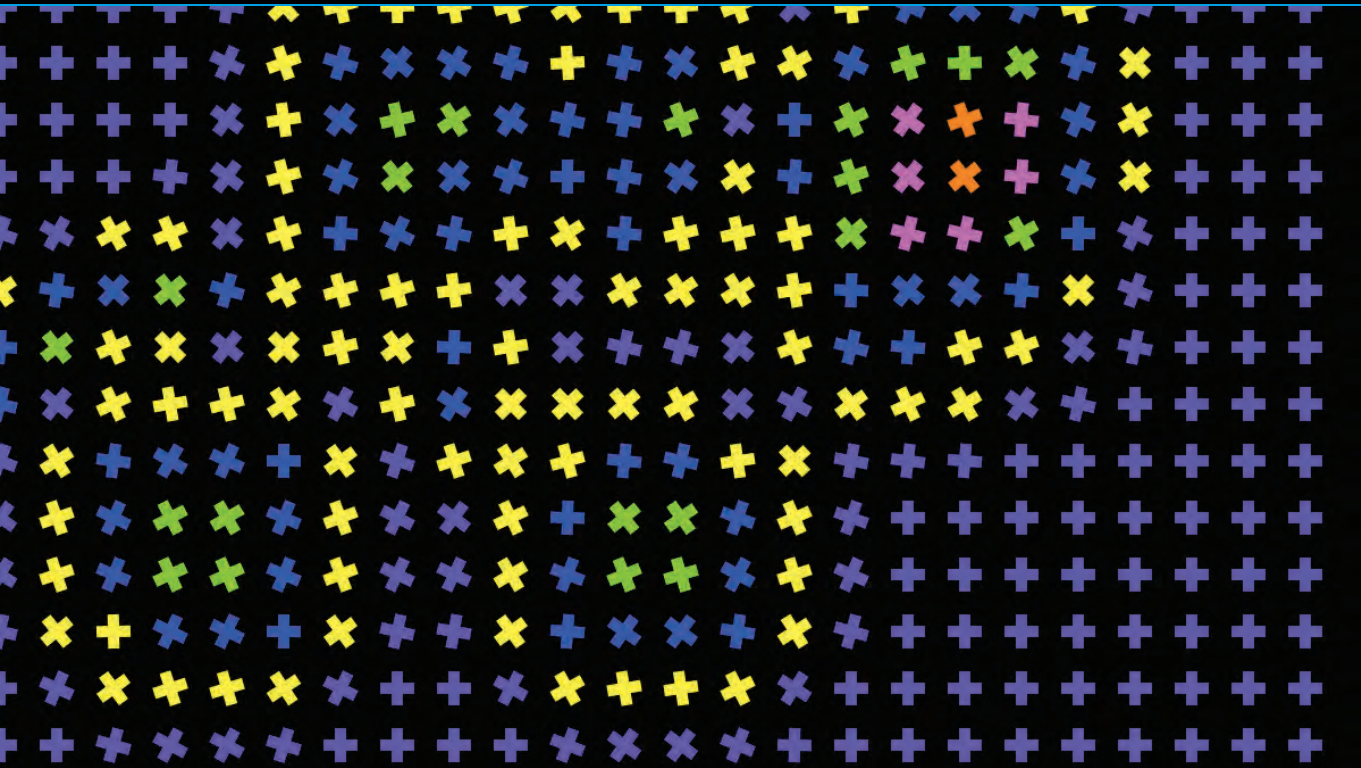
Thermal Vision
Values mapped to Hue Spectrum



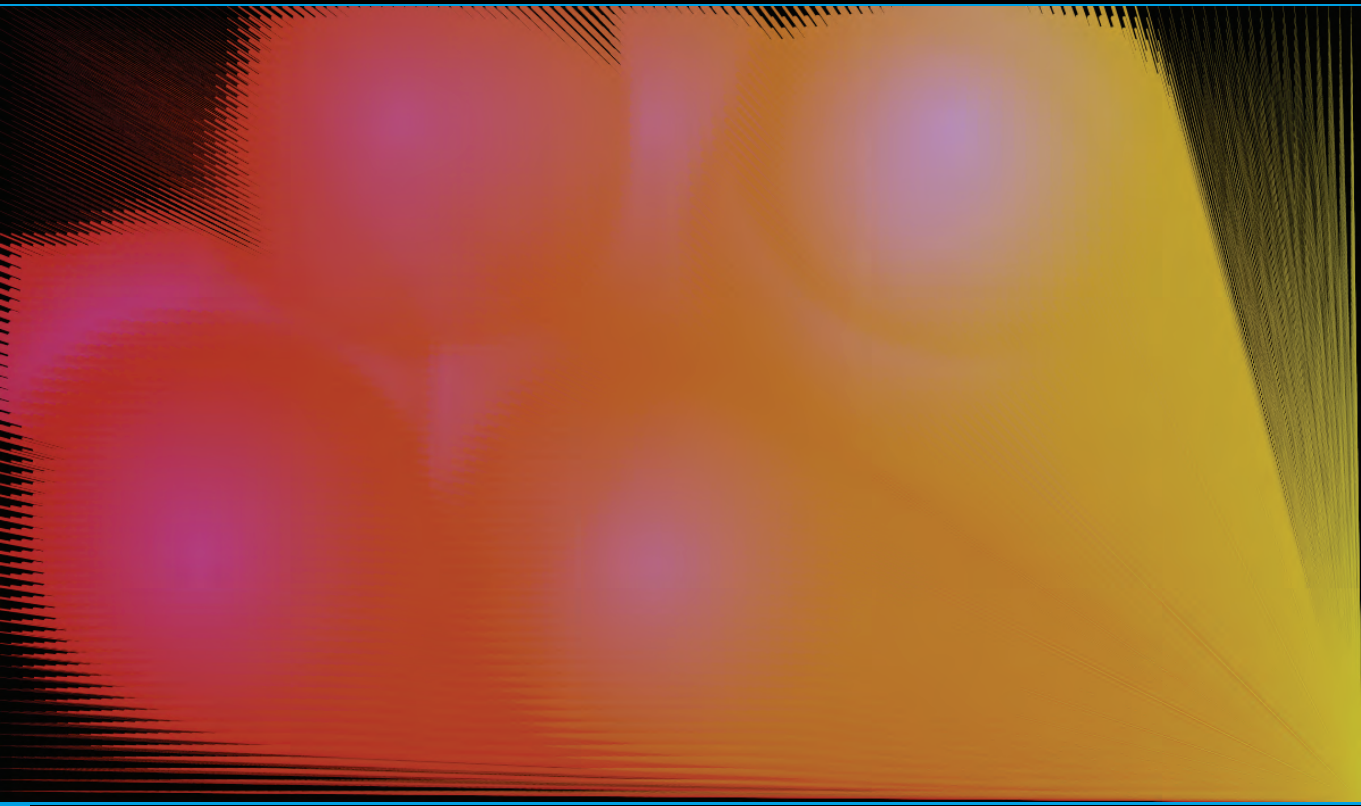
80´s MainFrame
Vertical pixel displacement



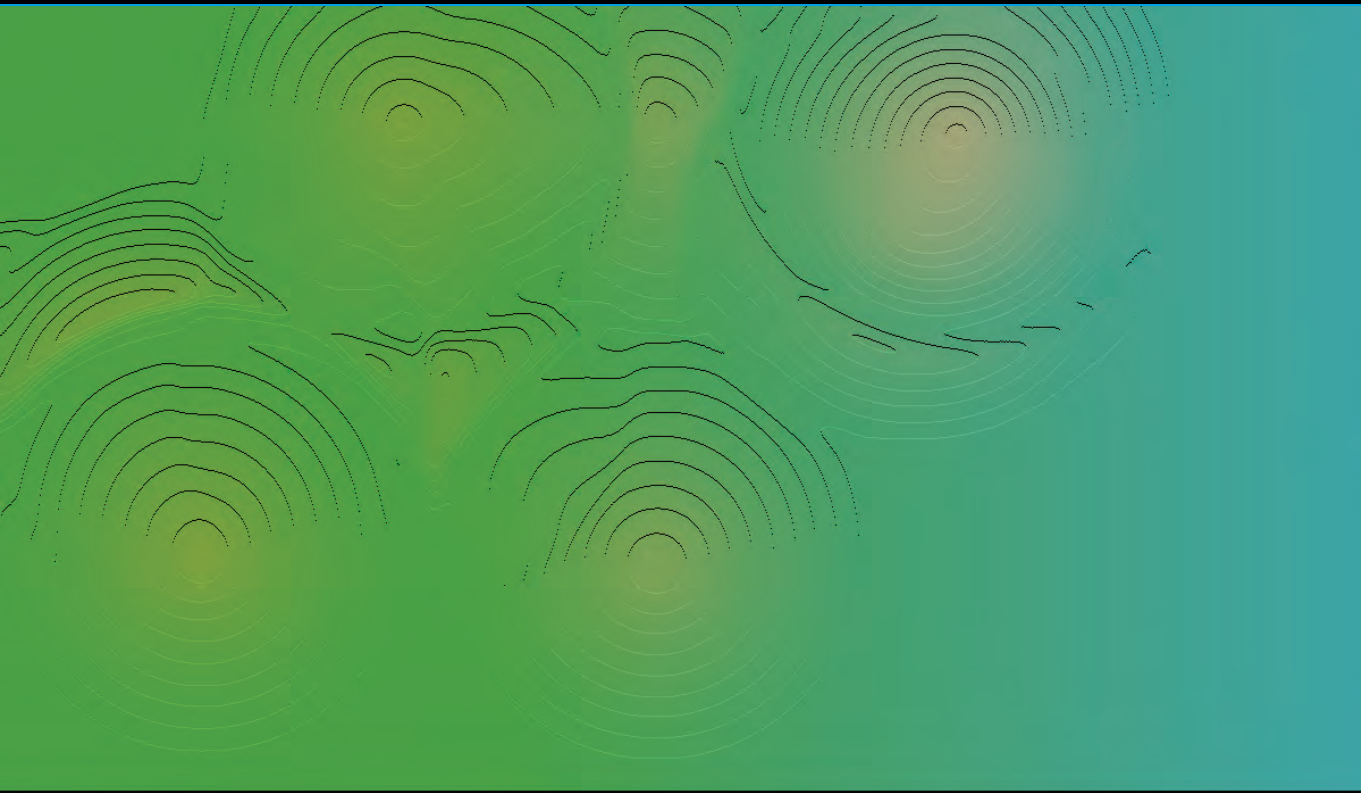
Flowers on a Pond
Arc count increments, randomly rotating. Colors mapped to Value.



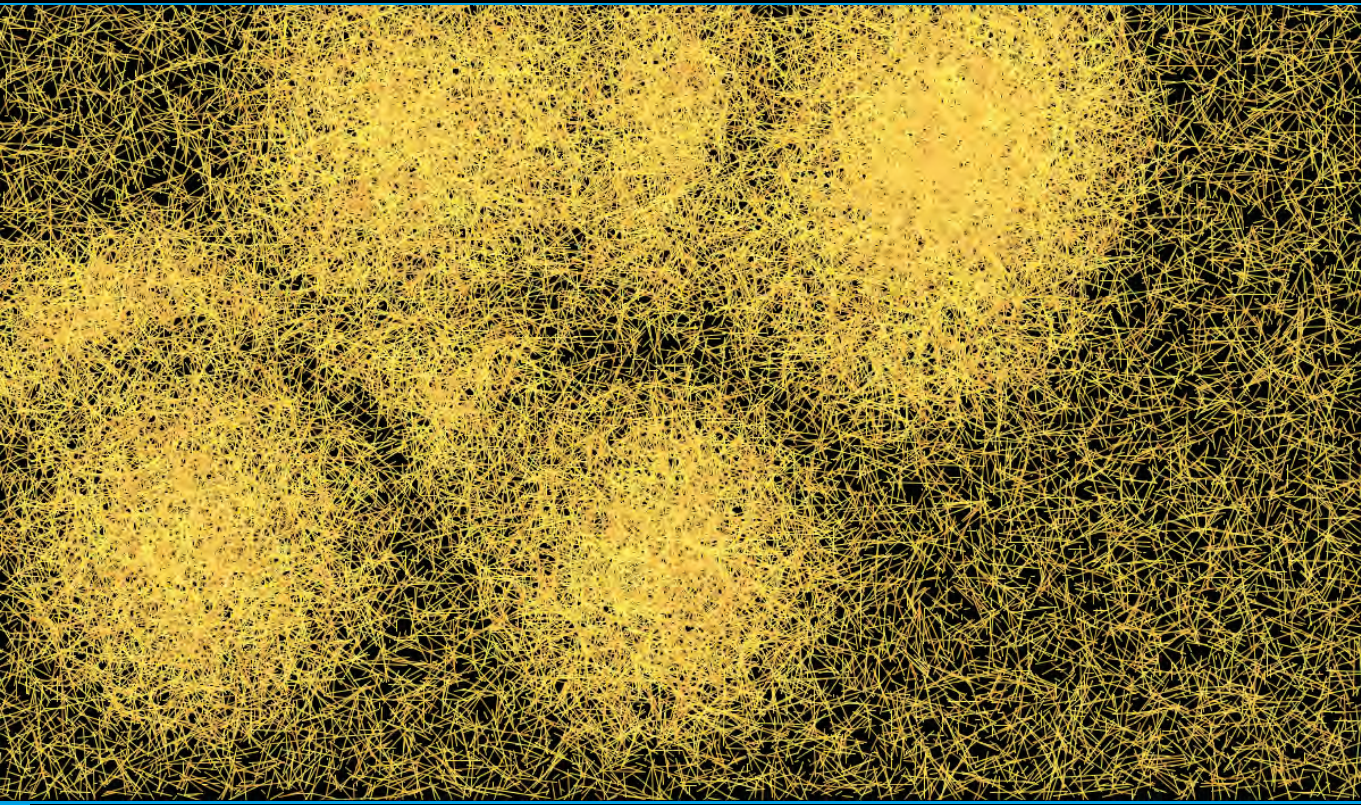
Kid's Game
Hue and Rotation on a Module



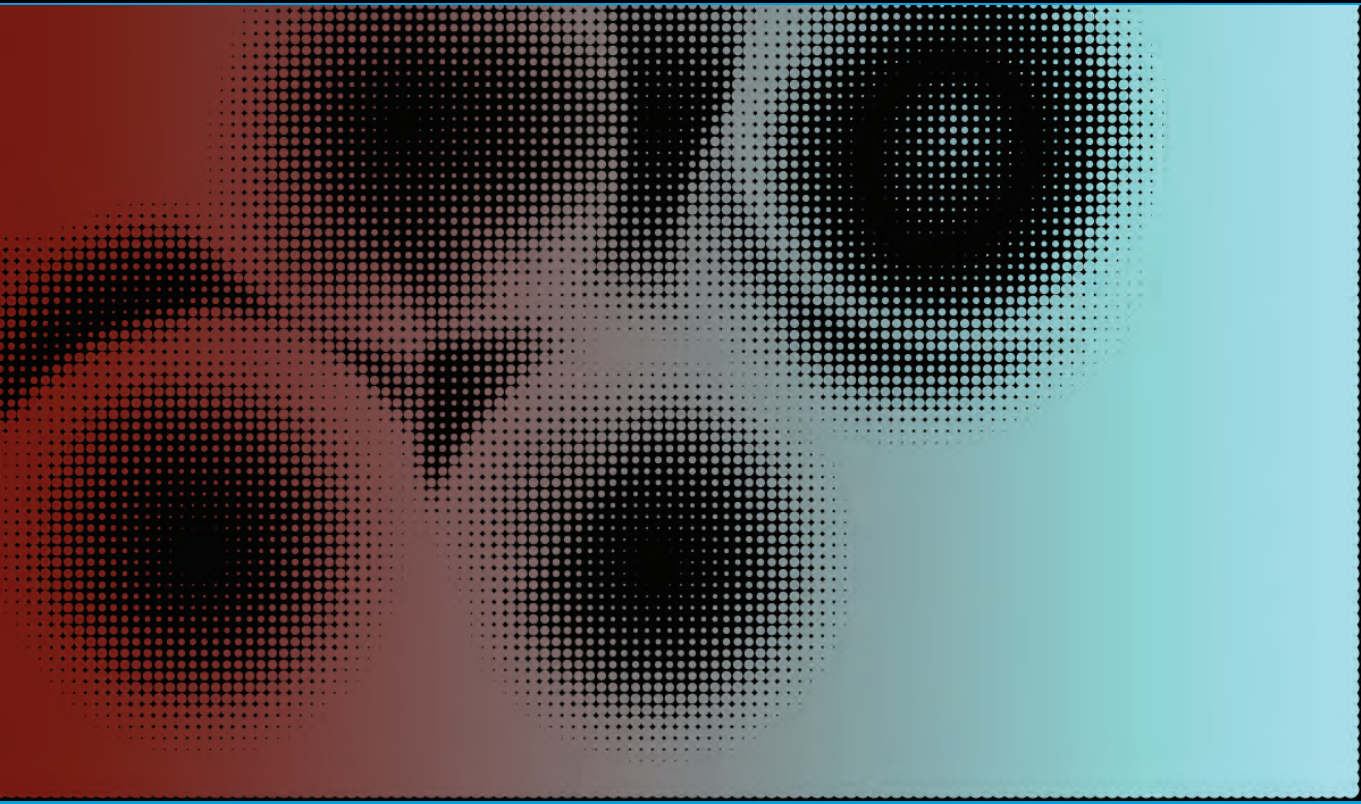
SunBurst
Hue and line weight traced from an origin



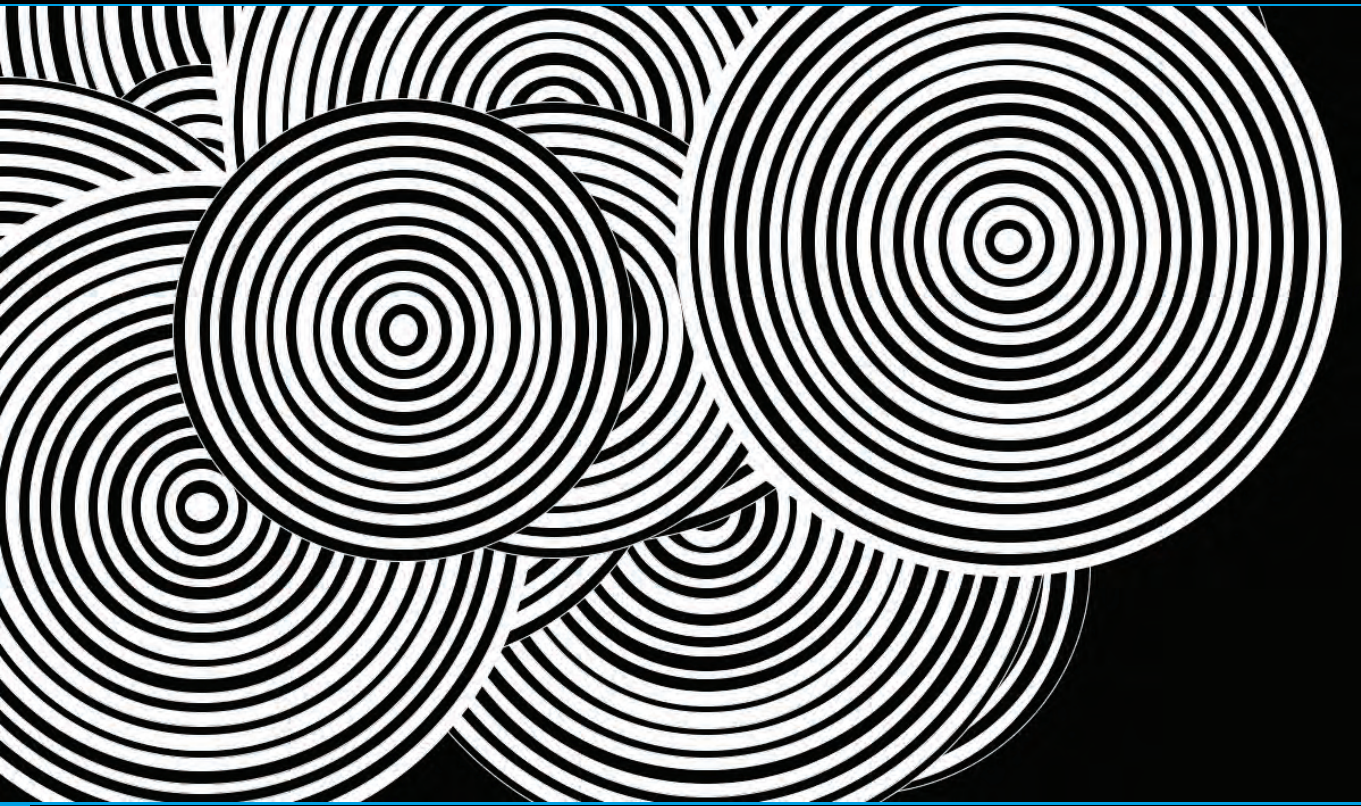
Countour Landscape
Vertical and Horizontal pixel displacement



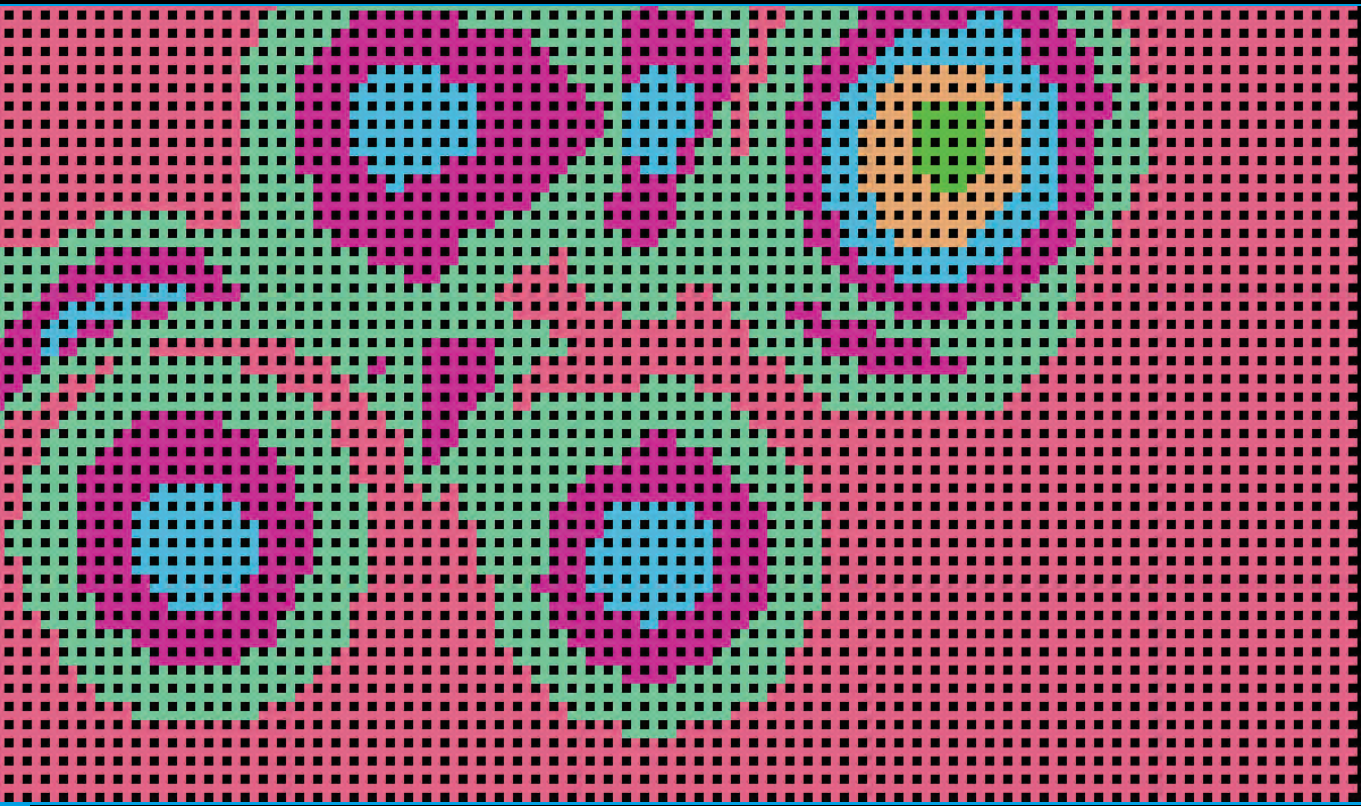
Needle in a HayStack
Variation and Concentration of hay



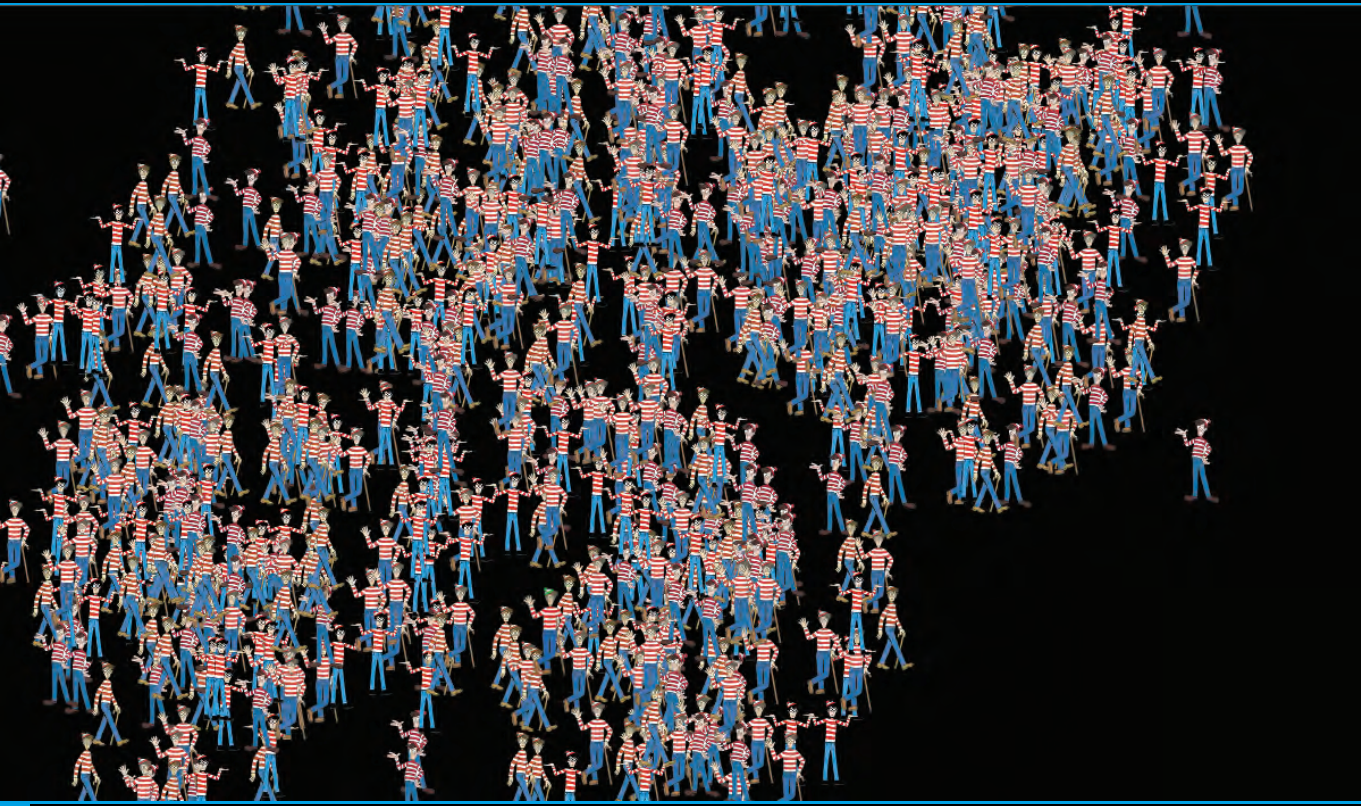
Moire
Circle diameter changes over a grid



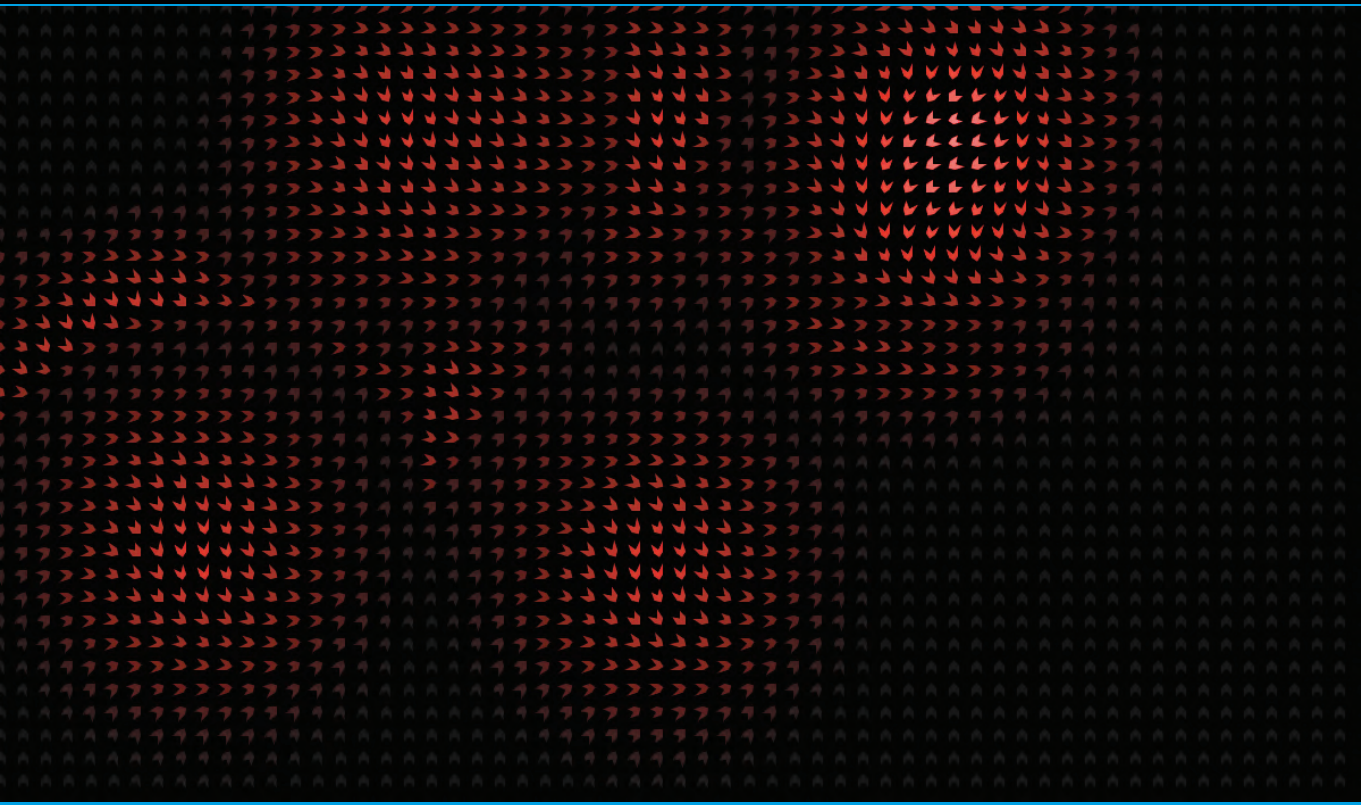
Op Art
Minor diameter variation over nodes



Knitting Pattern
Grid of crosses overlapping

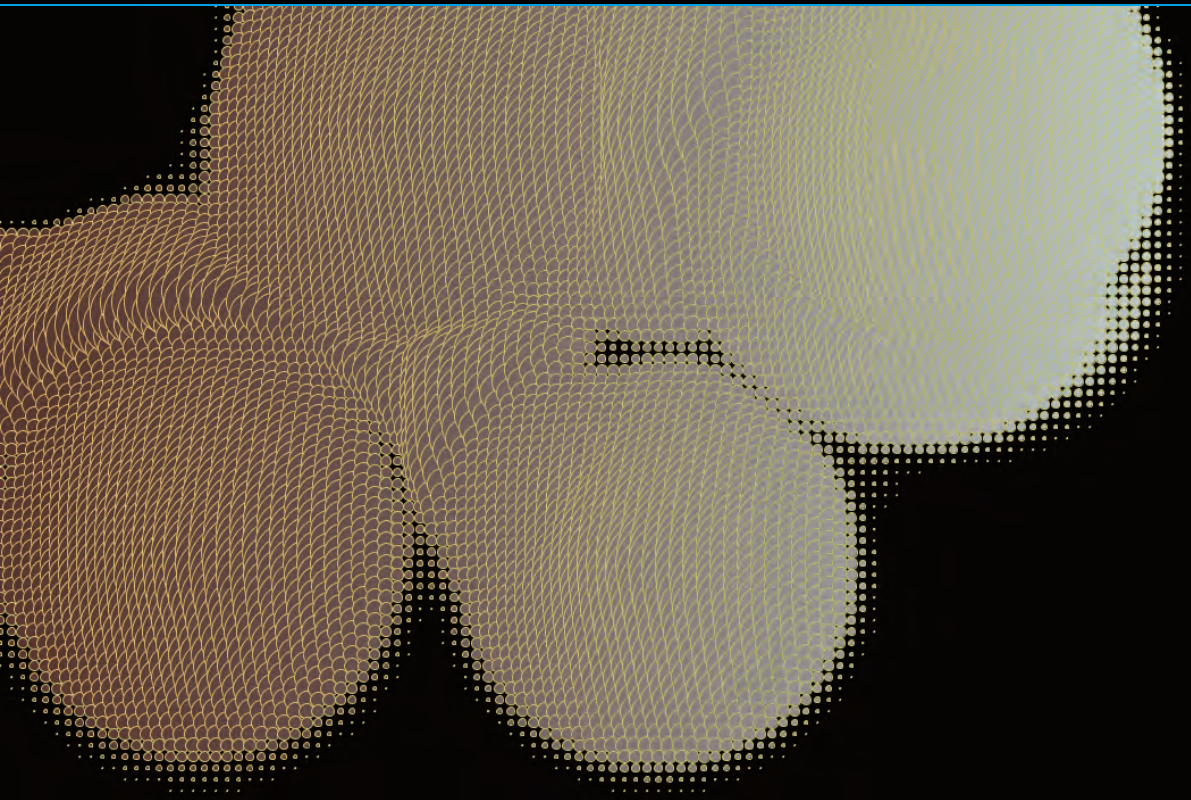


Where's Waldo?
Concentration of Waldos around nodes. (The real Waldo lost his hat, but got a brand new green one!!)



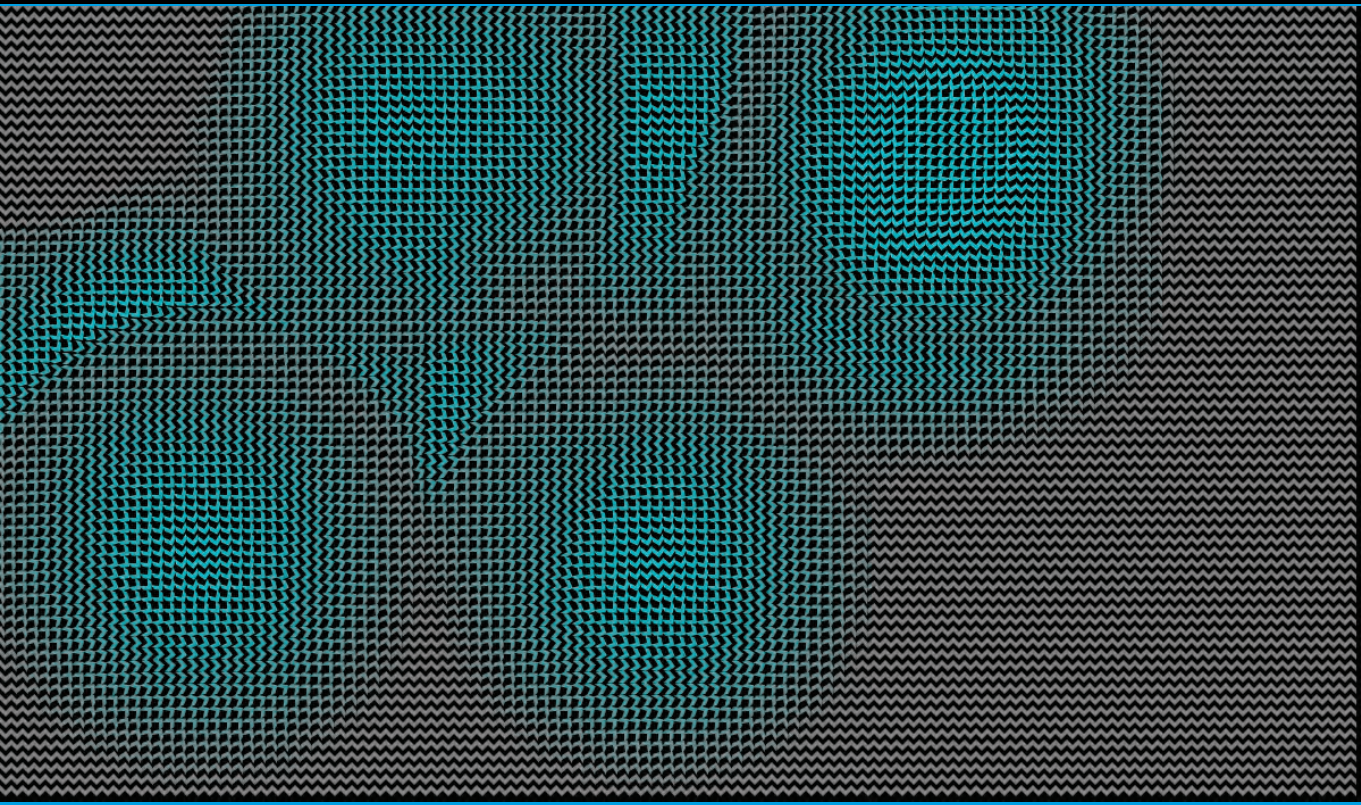
Magnetism
Saturation and Rotation on a Directional Module

Mondrian would be proud !!
Nodes control various properties of the Rectangle

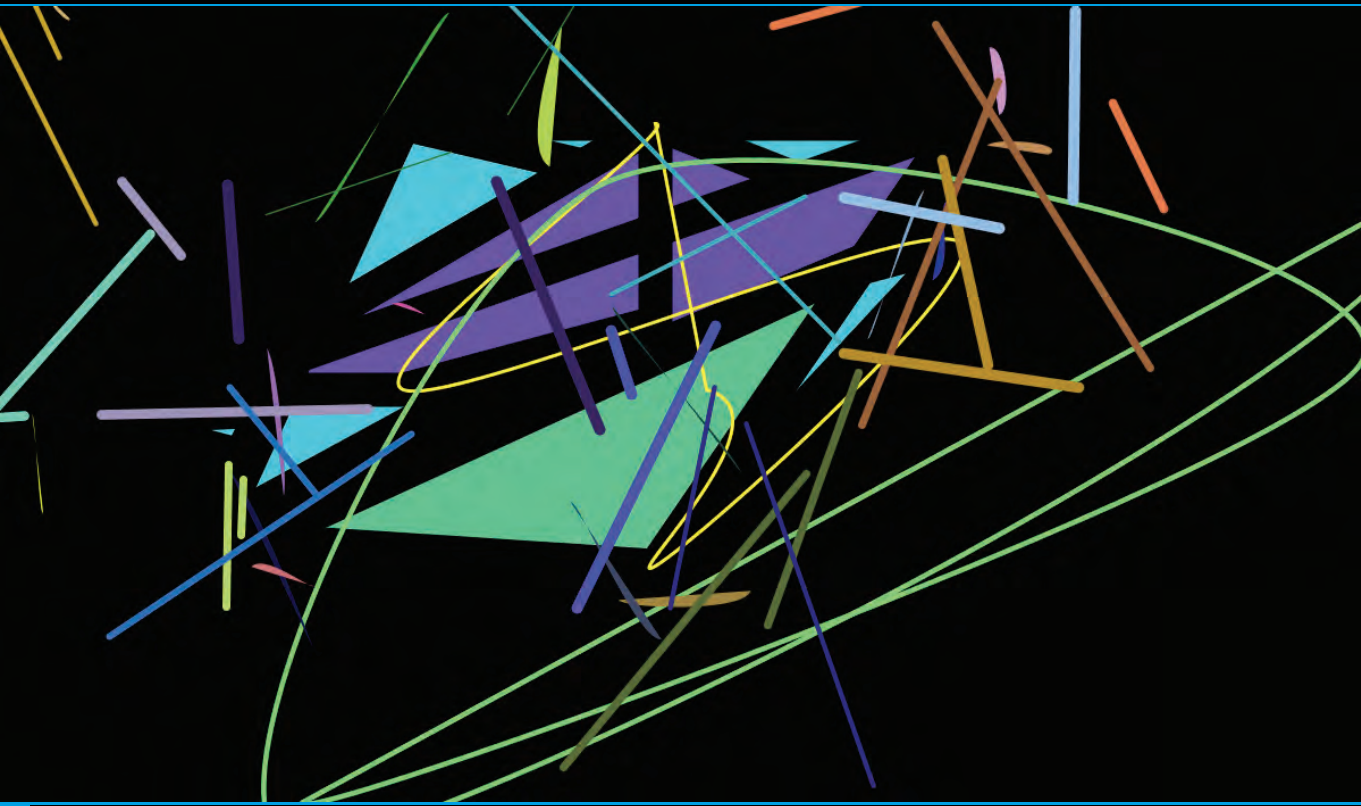


Turbulent Clouds
Overlapping circles create distortion

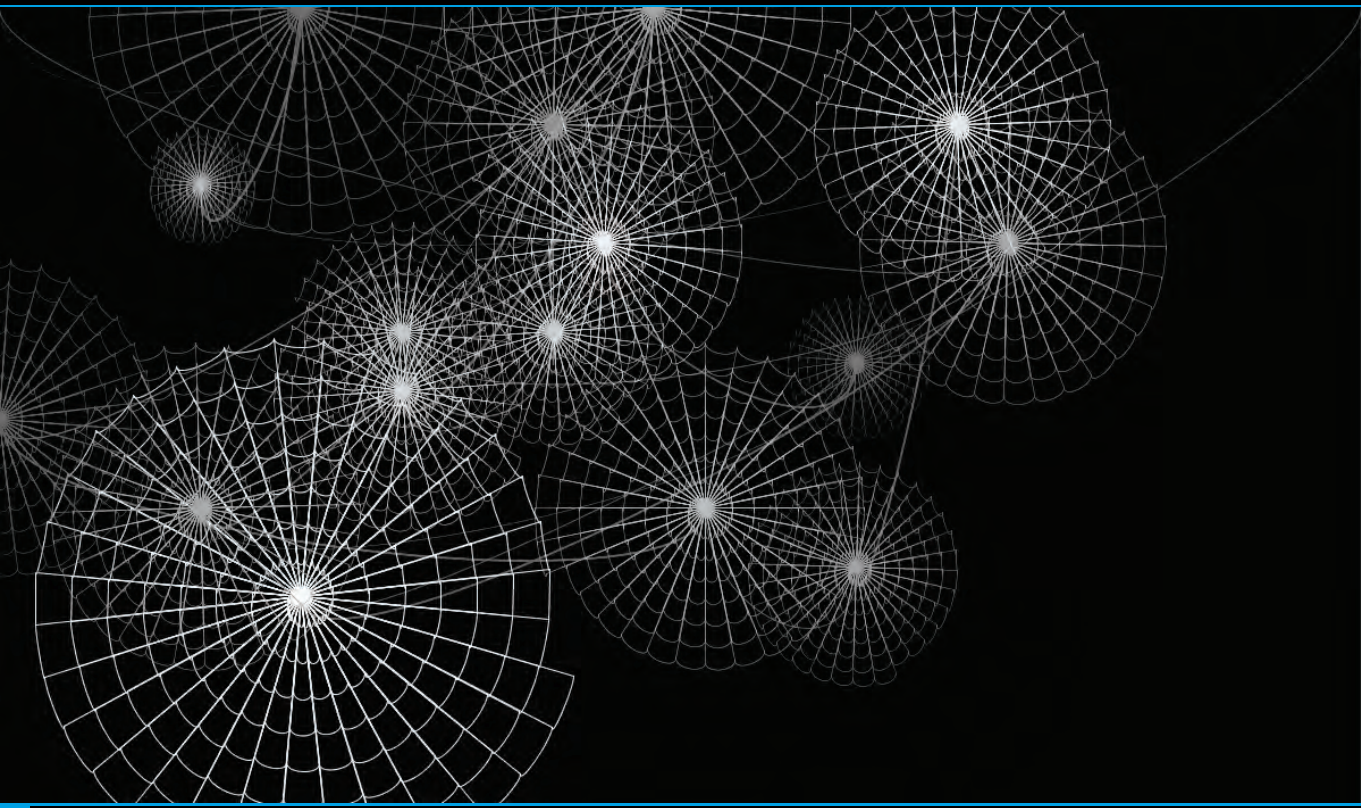
PaperCut
Rotating and Overlapping rectangles



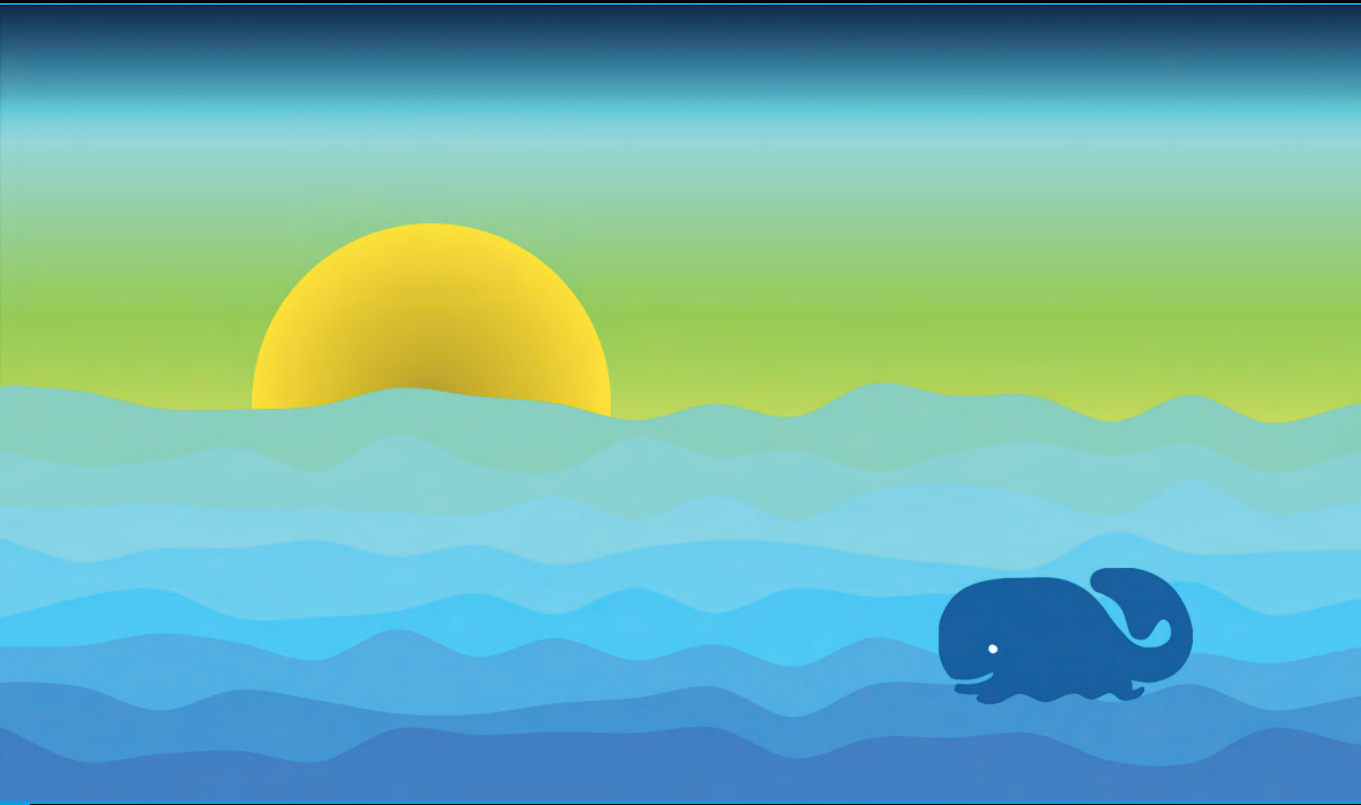
Vertigo
Directional modules on a tight grid



Untitled Number One
Curves and Planes interact wildly



SpiderWebs
Spirals and Lightness

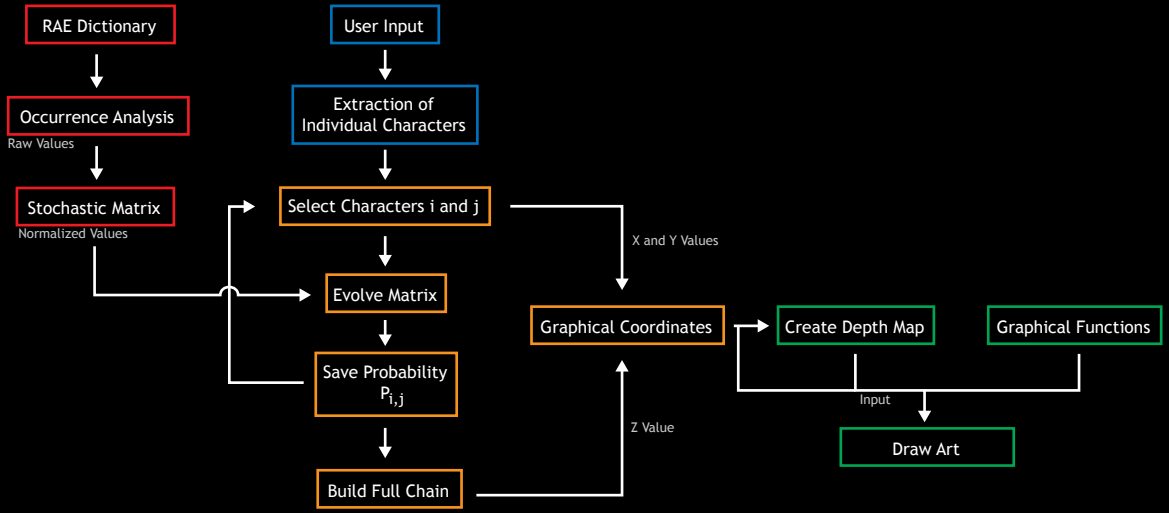


Sunset Horizon
Low Nodes control water tide, High Nodes control sun properties. The whale is just cute.

Theory

This section will explain the theory behind the processes involved.

First, the good old flowchart:



Analyzing the RAE Dictionary

The first step of the process is determining the initial probability of the system, or technically speaking, creating the **Transition or Stochastic Matrix.**

This means plotting a two-entry table where both axis contain the same letters to be analyzed, and the values inside the table tell us the probability of the letter at the horizontal axis to appear after the letter at the vertical axis.

The **RAE Dictionary** (The Dictionary of the Royal Spanish Academy) was used as a **universal source** for the creation of the table.
The program reads the whole dictionary and examines the appearence of a letter after other letter, filling the chart with integer values. The rows of one column show the number of times the letter at the row appears after the letter at the column. The sum of the entire column is the total number of occurrences of the letter in the dictionary.

In reality, the only way to determine the real probabilities of letter occurrences in the language is to analyze every single book ever written and continue analyzing the future ones.

This is impractical for two reasons:
1 - It's practically impossible, specially taking into account very old writings that are not digitalized, and publisher rights that can restrict access to texts.
2 - The probabilities will tend to stabilize as we included more and more books. Contrast will increase.

It seemed perfectly logical to analyze a dictionary, since it included all the possible words that could be used in any other text. But this does not take into account the frequency of the words.
To my surprise, and after analyzing other books, like Don Quijote de la Mancha, the Validation process proved that this approach was enough and worked.

The output of this process is the following table:

0.000

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	ñ
a	229	3378	9714	5479	2841	1520	3902	2321	6883	1726	50	8599	5569	6180	420	4326	0	12684	3929	9871	2102	1628	14	113	485	3199	777
b	3545	3	2	0	750	0	1	1	1239	0	0	287	1780	0	1240	0	0	648	166	8	723	0	0	0	1	3	0
c	6755	25	301	1	3541	0	1	1	6556	2	2	557	2	3854	2196	63	0	1273	2746	3	1435	0	0	133	3	101	0
d	10644	53	8	2	1731	0	12	0	3481	0	0	410	0	2877	890	1	0	1220	55	3	966	0	0	0	0	4	0
e	260	1442	3749	6905	127	1080	943	1412	3061	1068	27	5619	5154	3173	188	3624	0	8351	3559	9877	3275	1876	8	67	228	34	202
f	1124	5	0	0	609	7	0	0	1117	0	0	226	0	733	669	0	0	212	317	3	226	0	0	5	0	1	0
g	2008	3	0	20	1387	1	2	0	1460	0	0	367	0	1752	1657	0	0	789	215	2	384	0	0	0	0	98	0
h	409	0	4755	13	135	1	3	0	66	0	0	70	0	118	169	0	0	22	118	7	30	0	2	21	0	2	0
i	546	1949	7373	3897	475	2088	1373	1782	9	423	59	5265	4217	3483	272	2686	0	9095	3375	6493	2308	1889	11	342	27	47	187
j	1373	31	0	22	844	0	0	0	419	0	0	34	1	281	459	0	0	65	9	0	329	0	0	0	0	0	0
k	18	0	9	1	7	1	1	2	14	0	0	4	0	5	11	0	0	8	8	0	4	0	0	0	0	0	0
l	9483	1915	756	5	3562	543	336	1	4930	1	2	3867	4	106	4270	1130	0	262	208	83	2442	0	0	1	1	2	0
m	5969	10	9	35	2789	0	70	6	2474	0	0	738	8	252	2578	0	0	1019	1547	31	1101	0	0	4	4	60	0
n	9867	11	73	4	12940	2	278	1	7999	0	1	20	101	83	11067	12	0	921	117	18	1485	1	2	0	0	102	0
o	138	1948	9738	7872	1698	1340	1960	1514	6838	940	9	4704	4024	3814	90	2489	0	8633	3444	7021	178	1337	3	93	250	1516	776
p	2442	4	1	0	1124	0	0	1	1127	0	0	200	1912	2	1437	4	0	322	1588	4	536	0	0	169	0	18	0
q	486	0	2	10	305	0	0	0	350	0	0	75	1	323	333	0	0	220	368	0	97	0	0	7	0	29	0
r	19812	1969	1245	992	12465	840	1638	5	2275	0	9	8	1	173	9004	2077	0	3708	29	4970	3278	0	1	0	1	3	0
s	3682	197	7	17	8672	1	2	0	5523	0	0	222	2	1831	3975	168	0	1140	18	5	1301	0	1	0	1	1	0
t	4897	27	1153	0	3639	28	3	6	3227	0	0	750	5	10295	2396	389	0	1675	5800	9	1083	0	2	230	1	7	0
u	866	1011	2683	1436	322	579	2406	1005	109	596	8	1153	977	573	24	1110	3103	1539	1412	1851	5	123	0	34	184	317	74
v	1091	14	0	38	684	0	0	0	1449	0	0	291	1	401	456	0	0	359	102	2	86	0	0	1	0	0	0
w	6	0	0	2	2	0	0	0	1	0	1	0	0	0	4	0	0	4	1	2	1	0	0	1	0	0	0
x	128	0	0	0	866	0	0	0	54	0	0	0	0	0	156	0	0	2	0	0	29	0	0	0	0	0	0
y	582	16	1	11	121	0	0	1	18	0	1	3	0	30	205	1	0	6	13	1	102	0	0	1	0	0	0
z	1535	0	3	2	735	0	4	0	1813	1	0	89	0	423	281	0	0	151	6	14	248	0	0	0	0	9	0
ñ	707	0	0	0	688	0	0	0	218	0	0	0	0	0	140	0	0	0	1	0	184	0	0	0	0	0	0

Table of Raw Data
Showing total number of ocurrences of letter at Y-axis after letter at X-axis

Markov Chains

With the Raw Values, columns are **normalized.**
They will add up to 1, and are called **Probability Vectors.**

This normalized table is the **Original Stochastic matrix** that sets the rules for all the letters.
It means that if we were to run the process for 2 letters only, this table will tell us.

But what if we had more than 2 letters in the series?
Words form a chain of letters.

A Markov Chain is a process that consists of a finite or countable number of states and some known probabilities Pi,j, where Pi,j is the probability of moving from state i to state j.

In our case:
- The finite number of states is the amount of letters in the alphabet.
- The known probabilities are the output of the RAE Dictionary analysis, the Stochastic Matrix
- i is the letter before j.
- j is the actual letter.
- Pi,j is the probability of j ocurring after i.

The Stochastic Matrix plots j at the horizontal axis, i at the vertical axis, and Pi,j inside the table.
Note: Due to ease of coding, this plotting does not follow the English Mathematics convention that uses the rows as the Probability Vectors. In this work, columns sum to one, and not rows.

The output of this process is the following Matrix:

0.000

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	ñ
a	0.003	0.241	0.234	0.205	0.045	0.189	0.302	0.288	0.110	0.363	0.296	0.256	0.234	0.152	0.009	0.239	0.000	0.233	0.135	0.245	0.088	0.238	0.318	0.092	0.409	0.576	0.385
b	0.040	0.000	0.000	0.000	0.012	0.000	0.000	0.000	0.020	0.000	0.000	0.009	0.075	0.000	0.028	0.000	0.000	0.012	0.006	0.000	0.030	0.000	0.000	0.000	0.001	0.001	0.000
c	0.076	0.002	0.007	0.000	0.056	0.000	0.000	0.000	0.105	0.000	0.012	0.017	0.000	0.095	0.049	0.003	0.000	0.023	0.094	0.000	0.060	0.000	0.000	0.109	0.003	0.018	0.000
d	0.120	0.004	0.000	0.000	0.027	0.000	0.001	0.000	0.056	0.000	0.000	0.012	0.000	0.071	0.020	0.000	0.000	0.022	0.002	0.000	0.040	0.000	0.000	0.000	0.000	0.001	0.000
e	0.003	0.103	0.090	0.258	0.002	0.134	0.073	0.175	0.049	0.225	0.160	0.167	0.217	0.078	0.004	0.200	0.000	0.154	0.122	0.245	0.137	0.274	0.182	0.055	0.192	0.006	0.100
f	0.013	0.000	0.000	0.000	0.010	0.001	0.000	0.000	0.018	0.000	0.000	0.007	0.000	0.018	0.015	0.000	0.000	0.004	0.011	0.000	0.009	0.000	0.000	0.004	0.000	0.000	0.000
g	0.023	0.000	0.000	0.001	0.022	0.000	0.000	0.000	0.023	0.000	0.000	0.011	0.000	0.043	0.037	0.000	0.000	0.015	0.007	0.000	0.016	0.000	0.000	0.000	0.000	0.018	0.000
h	0.005	0.000	0.114	0.000	0.002	0.000	0.000	0.000	0.001	0.000	0.000	0.002	0.000	0.003	0.004	0.000	0.000	0.000	0.004	0.000	0.001	0.000	0.045	0.017	0.000	0.000	0.000
i	0.006	0.139	0.177	0.146	0.008	0.260	0.106	0.221	0.000	0.089	0.349	0.157	0.177	0.085	0.006	0.149	0.000	0.167	0.116	0.161	0.096	0.276	0.250	0.280	0.023	0.008	0.093
j	0.015	0.002	0.000	0.001	0.013	0.000	0.000	0.000	0.007	0.000	0.000	0.001	0.000	0.007	0.010	0.000	0.000	0.001	0.000	0.000	0.014	0.000	0.000	0.000	0.000	0.000	0.000
k	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
l	0.107	0.137	0.018	0.000	0.056	0.068	0.026	0.000	0.079	0.000	0.012	0.115	0.000	0.003	0.096	0.062	0.000	0.005	0.007	0.002	0.102	0.000	0.000	0.001	0.001	0.000	0.000
m	0.067	0.001	0.000	0.001	0.044	0.000	0.005	0.001	0.039	0.000	0.000	0.022	0.000	0.006	0.058	0.000	0.000	0.019	0.053	0.001	0.046	0.000	0.000	0.003	0.003	0.011	0.000
n	0.111	0.001	0.002	0.000	0.205	0.000	0.021	0.000	0.128	0.000	0.006	0.001	0.004	0.002	0.248	0.001	0.000	0.017	0.004	0.000	0.062	0.000	0.045	0.000	0.000	0.018	0.000
o	0.002	0.139	0.234	0.294	0.027	0.167	0.152	0.188	0.109	0.198	0.053	0.140	0.169	0.094	0.002	0.138	0.000	0.159	0.118	0.174	0.007	0.195	0.068	0.076	0.211	0.273	0.385
p	0.028	0.000	0.000	0.000	0.018	0.000	0.000	0.000	0.018	0.000	0.000	0.006	0.080	0.000	0.032	0.000	0.000	0.006	0.054	0.000	0.022	0.000	0.000	0.138	0.000	0.003	0.000
q	0.005	0.000	0.000	0.000	0.005	0.000	0.000	0.000	0.006	0.000	0.000	0.002	0.000	0.008	0.007	0.000	0.000	0.004	0.013	0.000	0.004	0.000	0.000	0.006	0.000	0.005	0.000
r	0.224	0.141	0.030	0.037	0.198	0.105	0.127	0.001	0.036	0.000	0.053	0.000	0.000	0.004	0.202	0.115	0.000	0.068	0.001	0.123	0.137	0.000	0.023	0.000	0.001	0.001	0.000
s	0.042	0.014	0.000	0.001	0.138	0.000	0.000	0.000	0.088	0.000	0.000	0.007	0.000	0.045	0.089	0.009	0.000	0.021	0.001	0.000	0.054	0.000	0.023	0.000	0.001	0.000	0.000
t	0.055	0.002	0.028	0.000	0.058	0.003	0.000	0.001	0.051	0.000	0.000	0.022	0.000	0.253	0.054	0.022	0.000	0.031	0.199	0.000	0.045	0.000	0.045	0.188	0.001	0.001	0.000
u	0.010	0.072	0.065	0.054	0.005	0.072	0.186	0.125	0.002	0.125	0.047	0.034	0.041	0.014	0.001	0.061	1.000	0.028	0.048	0.046	0.000	0.018	0.000	0.028	0.155	0.057	0.037
v	0.012	0.001	0.000	0.001	0.011	0.000	0.000	0.000	0.023	0.000	0.000	0.009	0.000	0.010	0.010	0.000	0.000	0.007	0.003	0.000	0.004	0.000	0.000	0.001	0.000	0.000	0.000
w	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.006	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.000	0.000	0.000
x	0.001	0.000	0.000	0.000	0.014	0.000	0.000	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.003	0.000	0.000	0.000	0.000	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.000
y	0.007	0.001	0.000	0.000	0.002	0.000	0.000	0.000	0.000	0.000	0.006	0.000	0.000	0.001	0.005	0.000	0.000	0.000	0.000	0.000	0.004	0.000	0.000	0.001	0.000	0.000	0.000
z	0.017	0.000	0.000	0.000	0.012	0.000	0.000	0.000	0.029	0.000	0.000	0.003	0.000	0.010	0.006	0.000	0.000	0.003	0.000	0.000	0.010	0.000	0.000	0.000	0.000	0.002	0.000
ñ	0.008	0.000	0.000	0.000	0.011	0.000	0.000	0.000	0.003	0.000	0.000	0.000	0.000	0.000	0.003	0.000	0.000	0.000	0.000	0.000	0.008	0.000	0.000	0.000	0.000	0.000	0.000

Table of Normalized Data

Stochastic Matrix of letter ocurrences in the RAE Dictionary

Validation

I was filled with great satisfaction when Julia Picabea, the mathematician at Continente 7, who did a work on cryptography, referred me to a table at Wikipedia that showed the most frequently occurring letters in many languages, including Spanish.

Right away, I started to code functions that will enable me to **visually distinguish Pi,j above a certain threshold.**
My guess was that, after running the Chain for more than 2 letters and **for enough letters**, the probabilities will tend to contrast and stabilize: letters with higher occurrences will raise the probabilities of all the other letters to pair with it, and the ones with lower occurrences will sink.

And indeed, after enough runs (evolutions), I could see a clear distinction of higher probabilities, which I then compared to the table at Wikipedia.
The numbers matched. Letters e,a,o,i,s,r,n appeared with the most frequency at both, over 6%.

This validated the text analysis and the later normalization to Probability Vectors.

Letter	French [6]	German [7]	Spanish [8]	Portuguese [9]	Esperanto [10]
e	14.715%	17.40%	13.68%	12.57%	8.99%
a	7.636%	6.51%	12.53%	14.63%	12.12%
o	5.378%	2.51%	8.68%	10.73%	8.78%
s	7.948%	7.27%	7.98%	7.81%	6.09%
r	6.553%	7.00%	6.87%	6.53%	5.91%
n	7.095%	9.78%	6.71%	5.05%	7.96%
i	7.529%	7.55%	6.25%	6.18%	10.01%
d	3.669%	5.08%	5.86%	4.99%	3.04%
l	5.456%	3.44%	4.97%	2.78%	6.14%
c	3.260%	3.06%	4.68%	3.88%	0.78%
t	7.244%	6.15%	4.63%	4.74%	5.27%
u	6.311%	4.35%	3.93%	4.63%	3.18%
m	2.968%	2.53%	3.15%	4.74%	2.99%
p	3.021%	0.79%	2.51%	2.52%	2.74%

The following tables with **Visual Guides** show:

1. Original Stochastic Matrix.
2. Matrix after a couple of evolutions. Notice how vowels rule!

Reds show mapped values from 0 to Maximum Probability calculated.
Greens show values over a specified threshold.

0.300

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	ñ
a	0.003	0.241	0.234	0.205	0.045	0.189	0.[illegible]	0.288	0.110	0.[illegible]	0.296	0.256	0.234	0.152	0.009	0.239	0.000	0.233	0.135	0.245	0.088	0.238	0.[illegible]	0.092	0.[illegible]	0.[illegible]	0.[illegible]
b	0.040	0.000	0.000	0.000	0.012	0.000	0.000	0.000	0.020	0.000	0.000	0.009	0.075	0.000	0.028	0.000	0.000	0.012	0.006	0.000	0.030	0.000	0.000	0.000	0.001	0.001	0.000
c	0.076	0.002	0.007	0.000	0.056	0.000	0.000	0.000	0.105	0.000	0.012	0.017	0.000	0.095	0.049	0.003	0.000	0.023	0.094	0.000	0.060	0.000	0.000	0.109	0.003	0.018	0.000
d	0.120	0.004	0.000	0.000	0.027	0.000	0.001	0.000	0.056	0.000	0.000	0.012	0.000	0.071	0.020	0.000	0.000	0.022	0.002	0.000	0.040	0.000	0.000	0.000	0.000	0.001	0.000
e	0.003	0.103	0.090	0.258	0.002	0.134	0.073	0.175	0.049	0.225	0.160	0.167	0.217	0.078	0.004	0.200	0.000	0.154	0.122	0.245	0.137	0.274	0.182	0.055	0.192	0.006	0.100
f	0.013	0.000	0.000	0.000	0.010	0.001	0.000	0.000	0.018	0.000	0.000	0.007	0.000	0.018	0.015	0.000	0.000	0.004	0.011	0.000	0.009	0.000	0.000	0.004	0.000	0.000	0.000
g	0.023	0.000	0.000	0.001	0.022	0.000	0.000	0.000	0.023	0.000	0.000	0.011	0.000	0.043	0.037	0.000	0.000	0.015	0.007	0.000	0.016	0.000	0.000	0.000	0.000	0.018	0.000
h	0.005	0.000	0.114	0.000	0.002	0.000	0.000	0.000	0.001	0.000	0.000	0.002	0.000	0.003	0.004	0.000	0.000	0.000	0.004	0.000	0.001	0.000	0.045	0.017	0.000	0.000	0.000
i	0.006	0.139	0.177	0.146	0.008	0.260	0.106	0.221	0.000	0.089	0.[illegible]	0.157	0.177	0.085	0.006	0.149	0.000	0.167	0.116	0.161	0.096	0.276	0.250	0.280	0.023	0.008	0.093
j	0.015	0.002	0.000	0.001	0.013	0.000	0.000	0.000	0.007	0.000	0.000	0.001	0.000	0.007	0.010	0.000	0.000	0.001	0.000	0.000	0.014	0.000	0.000	0.000	0.000	0.000	0.000
k	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
l	0.107	0.137	0.018	0.000	0.056	0.068	0.026	0.000	0.079	0.000	0.012	0.115	0.000	0.003	0.096	0.062	0.000	0.005	0.007	0.002	0.102	0.000	0.000	0.001	0.001	0.000	0.000
m	0.067	0.001	0.000	0.001	0.044	0.000	0.005	0.001	0.039	0.000	0.000	0.022	0.000	0.006	0.058	0.000	0.000	0.019	0.053	0.001	0.046	0.000	0.000	0.003	0.003	0.011	0.000
n	0.111	0.001	0.002	0.000	0.205	0.000	0.021	0.000	0.128	0.000	0.006	0.001	0.004	0.002	0.248	0.001	0.000	0.017	0.004	0.000	0.062	0.000	0.045	0.000	0.000	0.018	0.000
o	0.002	0.139	0.234	0.294	0.027	0.167	0.152	0.188	0.109	0.198	0.053	0.140	0.169	0.094	0.002	0.138	0.000	0.159	0.118	0.174	0.007	0.195	0.068	0.076	0.211	0.273	0.[illegible]
p	0.028	0.000	0.000	0.000	0.018	0.000	0.000	0.000	0.018	0.000	0.000	0.006	0.080	0.000	0.032	0.000	0.000	0.006	0.054	0.000	0.022	0.000	0.000	0.138	0.000	0.003	0.000
q	0.005	0.000	0.000	0.000	0.005	0.000	0.000	0.000	0.006	0.000	0.000	0.002	0.000	0.008	0.007	0.000	0.000	0.004	0.013	0.000	0.004	0.000	0.000	0.006	0.000	0.005	0.000
r	0.224	0.141	0.030	0.037	0.198	0.105	0.127	0.001	0.036	0.000	0.053	0.000	0.000	0.004	0.202	0.115	0.000	0.068	0.001	0.123	0.137	0.000	0.023	0.000	0.001	0.001	0.000
s	0.042	0.014	0.000	0.001	0.138	0.000	0.000	0.000	0.088	0.000	0.000	0.007	0.000	0.045	0.089	0.009	0.000	0.021	0.001	0.000	0.054	0.000	0.023	0.000	0.001	0.000	0.000
t	0.055	0.002	0.028	0.000	0.058	0.003	0.000	0.001	0.051	0.000	0.000	0.022	0.000	0.253	0.054	0.022	0.000	0.031	0.199	0.000	0.045	0.000	0.045	0.188	0.001	0.001	0.000
u	0.010	0.072	0.065	0.054	0.005	0.072	0.186	0.125	0.002	0.125	0.047	0.034	0.041	0.014	0.001	0.061	0.[illegible]	0.028	0.048	0.046	0.000	0.018	0.000	0.028	0.155	0.057	0.037
v	0.012	0.001	0.000	0.001	0.011	0.000	0.000	0.000	0.023	0.000	0.000	0.009	0.000	0.010	0.010	0.000	0.000	0.007	0.003	0.000	0.004	0.000	0.000	0.001	0.000	0.000	0.000
w	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.006	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.000	0.000	0.000
x	0.001	0.000	0.000	0.000	0.014	0.000	0.000	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.003	0.000	0.000	0.000	0.000	0.000	0.001	0.000	0.000	0.000	0.000	0.000	0.000
y	0.007	0.001	0.000	0.000	0.002	0.000	0.000	0.000	0.000	0.000	0.006	0.000	0.000	0.001	0.005	0.000	0.000	0.000	0.000	0.000	0.004	0.000	0.000	0.001	0.000	0.000	0.000
z	0.017	0.000	0.000	0.000	0.012	0.000	0.000	0.000	0.029	0.000	0.000	0.003	0.000	0.010	0.006	0.000	0.000	0.003	0.000	0.000	0.010	0.000	0.000	0.000	0.000	0.002	0.000
ñ	0.008	0.000	0.000	0.000	0.011	0.000	0.000	0.000	0.003	0.000	0.000	0.000	0.000	0.000	0.003	0.000	0.000	0.000	0.000	0.000	0.008	0.000	0.000	0.000	0.000	0.000	0.000

Table of Normalized Data

Stochastic Matrix (no evolutions) - Highlighting values over 30 %

0.060

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	ñ
a	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140	0.140
b	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016	0.016
c	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047	0.047
d	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036
e	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080	0.080
f	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009
g	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017
h	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007	0.007
i	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063
j	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006	0.006
k	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
l	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052	0.052
m	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030	0.030
n	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076	0.076
o	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103	0.103
p	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017	0.017
q	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004
r	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063	0.063
s	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043	0.043
t	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059	0.059
u	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032	0.032
v	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008
w	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
x	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002
y	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002	0.002
z	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008
ñ	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003

Table of Evolved Normalized Data

Matrix after a couple of Evolutions - Highlighting values over 6 %. A reason for letter "S" not passing the bar might be that there are no plurals in dictionaries, therefore less ocurrences.

Markov Chain Evolution

To understand how we can calculate probabilities after a certain number of events, let´s evaluate a process for a simple Stochastic matrix of 2 states:

	i \ j	1 A	2 B
1	A	0.25	0.5
2	B	0.75	0.5
		1.0	1.0

To find Pi,j (Probability of j, happening after i), after 2 iterations 'N' (the second "roll of dice") we need to cover all the possible combinations to get there. This means multiplying the last matrix obtained (at N = 2 it is the same as the original) by the original stochastic matrix.
So to fill the entire table with the new probabilities, we have to:

ORIGINAL MATRIX EVOLVED MATRIX

$P_{A,A} = (P_{1,1} \times P_{1,1}) + (P_{1,2} \times P_{2,1}) = (0.25 \times 0{,}25) + (0.75 \times 0.5) = 0{,}4375$
$P_{A,B} = (P_{1,1} \times P_{1,2}) + (P_{1,2} \times P_{2,2}) = (0.25 \times 0{,}75) + (0.75 \times 0.5) = 0{,}5625$
$P_{B,A} = (P_{2,1} \times P_{1,1}) + (P_{2,2} \times P_{2,1}) = (0.5 \times 0{,}25) + (0.5 \times 0.5) = 0{,}375$
$P_{B,B} = (P_{2,2} \times P_{2,2}) + (P_{2,1} \times P_{1,2}) = (0.5 \times 0{,}5) + (0.5 \times 0.75) = 0{,}625$

The Evolved Matrix, for **N = 2**, is:

	i \ j	1 A	2 B
1	A	0.4375	0.375
2	B	0.5625	0.625
		1.0	1.0

If we are to find out the Evolved Matrix after 3 iterations, we would have to multiply the N = 2 Matrix by the Original Stochastic Matrix. For 4 iterations: N=3 x Original.
And so on, always multiplying N-1 by Original.
And repeat the process every time we want to evolve the matrix.

Algorithm to Code

Coding this will be translating it into a **couple of nested loops.** (Pseudo/Code in Processing(Java)).

```
char[] letters = letters in the alphabet;
float[][] originalMatrix = RAE Dictionary analysis;
float[][] evolvedMatrix = originalMatrix;
int matrixLength = letters.length();

for (nIteration) { // NUMBER OF EVOLUTIONS
  float[][] tempEvolvedMatrix = new float[matrixLength][matrixLength];
  for (int i = 0; i < matrixLength; i++) {
    for (int j = 0; j < matrixLength; j++) {
      float P = 0;
      for (int r = 0; r < matrixLength; r++) {
        P += originalMatrix[i][r] * evolvedMatrix[r][j];
      }
      tempEvolvedMatrix[i][j] = P;
    }
  }
  arraycopy(tempEvolvedMatrix, evolvedMatrix);
}
```

There are 4 iterating loops:
1 - A Global matrix evolution iterator
2 - A run through all i´s
3 - A run through all j´s
4 - A loop to iterate through all posible combinations of state, that will add up to find Pi,j.

The variable P captures Pi,j, which is then stored in a temporal Evolved matrix, which at the end is copied to the final Evolved Matrix. This is done this way because the Evolved Matrix should not be refreshed until all P´s have been calculated, since the fourth loop will search back to the matrix to find the values to work with.

```
for (int r = 0; r < matrixLength; r++) {
    P += originalMatrix[i][r] * evolvedMatrix[r][j];
}
```

responds to iterating $P_{i,j} \mathrel{+}= A_{i,r} \times B_{r,j}$ on all possible states/ways from i to j:

$P_{AA} = (P_{1,1} \times P_{1,1}) + (P_{1,2} \times P_{2,1}) + (P_{1,3} \times P_{3,1}) + (P_{1,4} \times P_{4,1}) +$ and so on...

iteration r = 0 r = 1 r = 2 r = 3

nIteration is set by the number of characters the user inputs. Therefore each letter probability will be related to the probability of each previous letter through the evolution.
For the word "Afro", the letter "o" will have the resulting probability of evolving the matrix 3 times, for the pairs "Af", "fr", "ro".

Values of Dimension

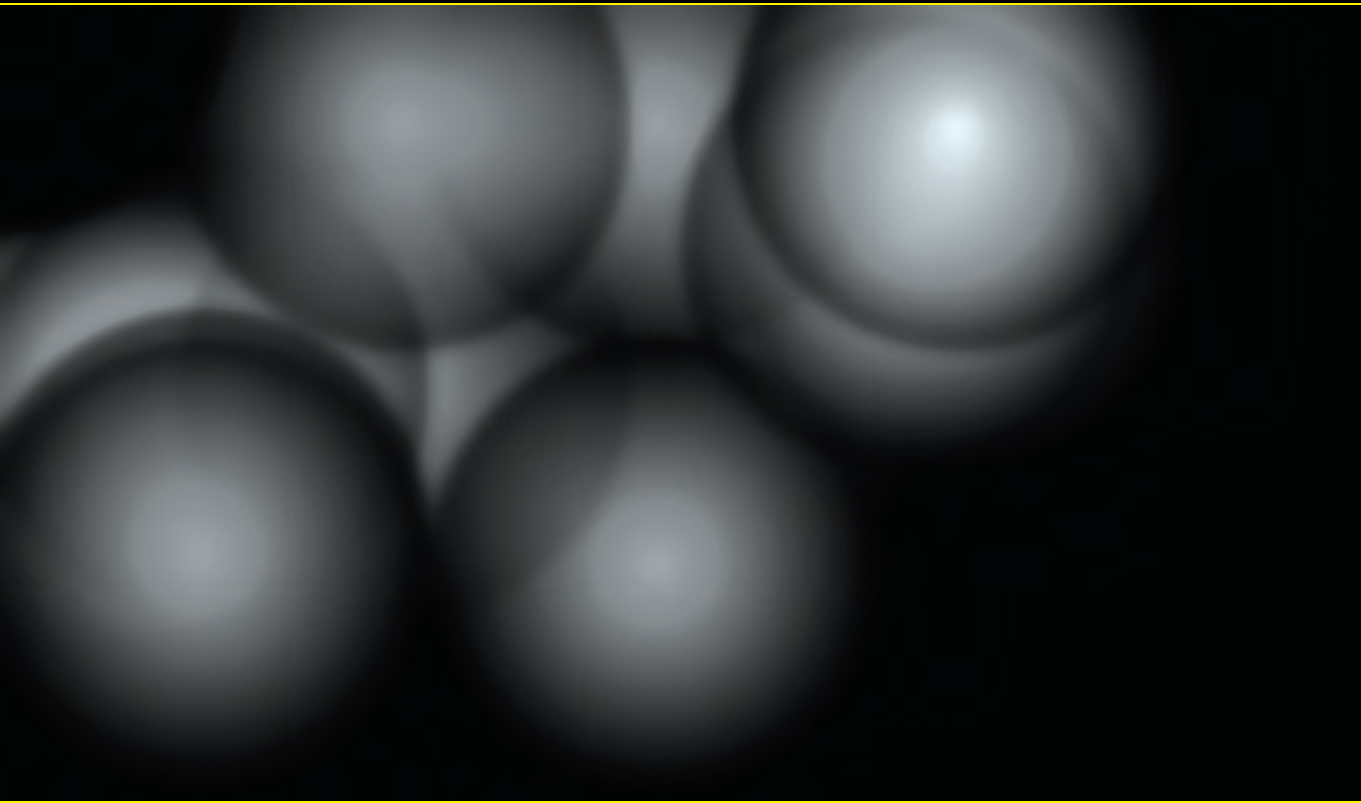
Done with the hardcore maths, I was able to extract values to use in a **Cartesian Coordinates System.**

The X and Y coordinates are simply extracted from the position of the pair of letter in the matrix, and the Z coordinate is the Pi,j.

For instance: **Pa,z** will have coordinates:
X = 0
Y = Maximum (screen height)
Z = $P_{a,z}$

With these coordinates, a Depth Map image can be created.
This will be a greyscale bitmap where the "pikes of the mountains", placed at X-Y, have a height of Z represented by mapping Z to the grey values and sloping down the mountain within a chosen radius. Pair of letters with higher probabilities will define a higher mountain (a whiter spot).

The result is a grey canvas where the intersections between mountains create some turbulence that will give the later graphical functions some delight.



Depth Map
Think of it as an aerial view of mountains with ground level at Black, and peak at White

Creative Graphics

Armed with an array of **single node coordinates (the mountains)**, and a **bitmap that represents the interaction of these nodes across the canvas**, the graphical functions can have a great degree of freedom.

The coder has access not only to a certain node in particular, but to a specific pixel in the screen that contains data about surrounding nodes and how they fit together.

Most of the renders shown in the gallery are based on an invisible grid traced upon the canvas that reads on the Depth Map.
Some grids are wide, some are as close as 1 pixel.

By going through the process described, the MarkovPaint project **effectively transform** the rigid maths of probability calculation to something as creative and artistic as a canvas full of colors and shapes.

Any coder can take the array of node data and the Depth Map and come up with totally different graphical representation, which will be, at the same time, totally different in relation to the initial word input, and will also have a third degree of freedom if the text analyzed is other.

Feel free to paint your own canvas of your own Markov Chain.

Agustin Ramos Anzorena