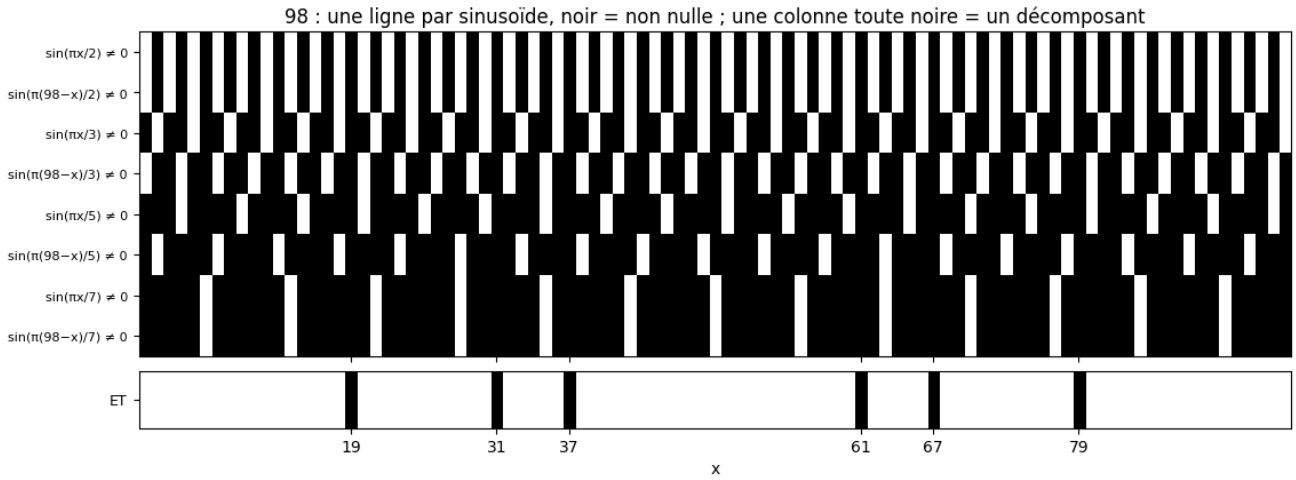
Grandes orgues, Denise Vella-Chemla pilotant claude, 11 octobre 2026

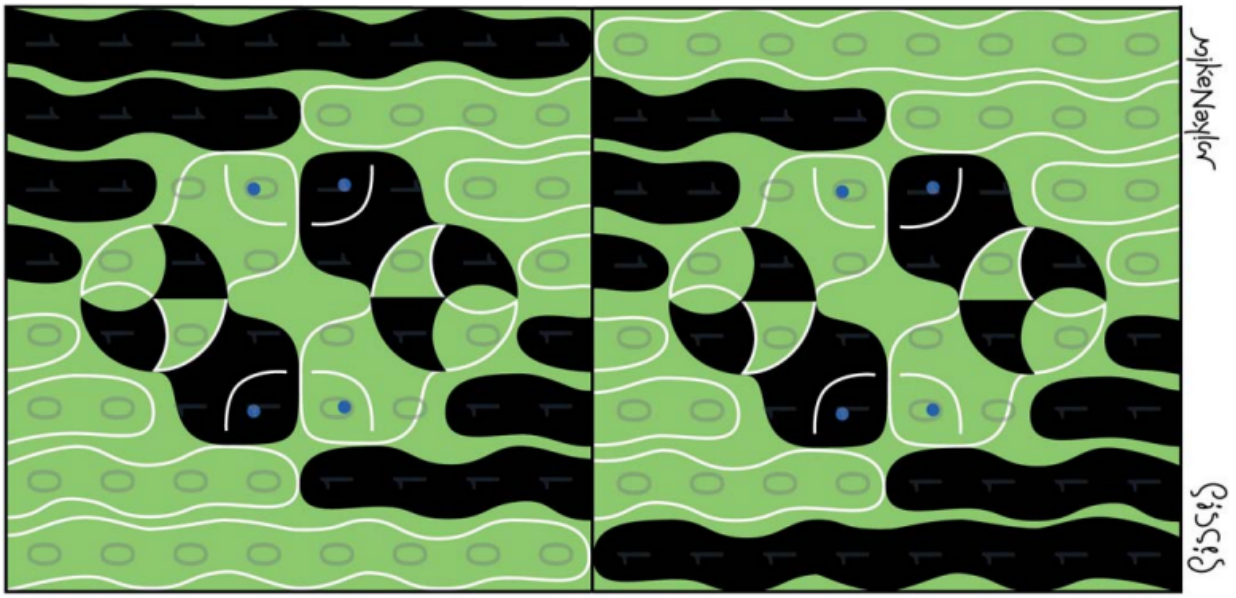
En m'amusant avec une ia à lui faire programmer des graphiques anciens trouvés dans un article d'un magazine de vulgarisation anglais qui s'appelle Eureka, j'ai voulu utiliser des visualisations similaires à base de pixels, comme je les utilisais dans mes grilles de booléens, il y a très longtemps.

Et l'ia m'a fourni une visualisation que je trouve vraiment jolie : elle fait penser à des tuyaux d'orgues.



Les lignes de pixels montrent les annulations ou non annulation des sinus $(\sin \pi x/p)$ pour p un nombre premier quelconque compris entre 2 et $\sqrt{n}$ (l'exemple est le cas $n = 98$; nota : π désigne bien le nombre ici et pas la fonction de décompte des nombres premiers). Là, au-dessous les unes des autres, on voit la ligne de $\sin(\pi x/2)$ et en dessous celle de $\sin(\pi(98-x)/2$, puis celle de $\sin(\pi x/3)$ et en-dessous celle de $\sin(\pi(98-x)/3)$, puis celle de $\sin(\pi x/5)$ et en-dessous celle de $\sin(\pi(98-x)/5)$ et enfin celle de $\sin(\pi x/7)$ et en-dessous celle de $\sin(\pi(98-x)/7)$.

Dans la même veine et dans le même magazine, j'avais trouvé un dessin d'artiste qui s'appelait *Poissons* et qui montre tous les mots de 16 lettres qu'on peut former sur l'alphabet $\{0, 1\}$. Tout ceci est stylisé et décoré pour que l'esprit ne s'ennuie pas à voir des suites de 0 et 1 toutes fades (enfin, fades pour certains, épicées pour d'autres, "des goûts et des couleurs, il ne faut pas discuter").



Pisces, by Mike Naylor

Aujourd'hui, bonne pêche, donc !

```
#!/usr/bin/env python3
"""
Les decomposants de 98, lisibles a l'oeil : pixels noirs/blancs, ET logique,
affichage immediat.

Colonne = x (2..96).  Pour chaque module p premier <= sqrt(98) (2, 3, 5, 7),
deux lignes :
    sin(pi*x/p)      != 0   (noir si non nul)
    sin(pi*(98-x)/p) != 0   (noir si non nul)
Derniere ligne (en bas, separee) : ET logique de toutes les lignes du dessus.
Une colonne entierement noire est un decomposant : x et 98-x passent
tous les modules.
"""
import math
import numpy as np
import matplotlib.pyplot as plt

N = 98
EPS = 1e-9
P = [p for p in range(2, math.isqrt(N) + 1) if all(p \% d for d in range(2, p))]
# 2, 3, 5, 7
x = np.arange(2, N - 1)
# colonnes 2..96
```

```
lignes, noms = [], []
for p in P:
    lignes.append(np.abs(np.sin(np.pi * x / p)) > EPS)
    noms.append(f"sin( x /{p})      0")
    lignes.append(np.abs(np.sin(np.pi * (N - x) / p)) > EPS)
    noms.append(f"sin(  (98  x  )/{p})      0")
M = np.array(lignes)          \# (8, 95) booleens : True = noir
ET = M.all(axis=0)            \# ET logique de toutes les lignes

decomposants = [int(v) for v in x[ET]]
print("colonnes entierement noires :", decomposants)
assert decomposants == [19, 31, 37, 61, 67, 79]

fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(13, 4.5), sharex=True,
                gridspec_kw={"height_ratios": [len(lignes), 1.4],
                "hspace": 0.08})
ext = [x[0] - 0.5, x[-1] + 0.5, 0, 1]
ax1.imshow(M, cmap="gray_r", interpolation="nearest", aspect="auto",
           extent=[ext[0], ext[1], len(lignes), 0], vmin=0, vmax=1)
ax1.set_yticks(np.arange(len(lignes)) + 0.5)
ax1.set_yticklabels(noms, fontsize=8)
ax1.set_title("98 : noir = non nulle, col.toute noire = decomp. G.")
ax2.imshow(ET[None, :], cmap="gray_r", interpolation="nearest",
              aspect="auto", extent=[ext[0], ext[1], 1, 0],
              vmin=0, vmax=1)
ax2.set_yticks([0.5]); ax2.set_yticklabels(["ET"], fontsize=9)
ax2.set_xticks(decomposants)
ax2.set_xlabel("x")
plt.show()
```