## Gene Expression Algorithm

```
import mattle fundom base Expension Algorithm
FUNCS = {"1":2, ".":2, """:2, "/":2, "Sn':1, "coa":1
 APPLY . L
      "4": Lambda ab:ab, "-": Lambda abiab,
       " lambda a, b a b, /": lambda a, b i a i f abs(b) e1e-12
       else a/b, "sir lambo a math sin(a),
        cos: lambon asmath. cos(a)
        "exp": lambia a math-exp (moults), min(50, a)),
         log't lambda a imallilog (obs (a)*1)
 TERMS : ["K" " " "]
def wit chrom (head, tout).
    9 - Tranden . chose (list (fours) + TERNS) for in range Chead ]
    9+= (random chave (seems) for in range (till)
     c= Estandom-unform(-1,2) for his in ecumenate (4) if s== 2)
  Teture ("g' + g, "e" + c)
def devote (9):
   Seq = (1) ned = 1; (10)
  while need > 0 and interesting;
     Sym = 901; seq-opperd(laym, i)); need -1
  the sym to functioned to functional toyont
    return Sug
def eval-enor (sequipme) s
   Spos = 59[1]
   I s in FUNCS.
     args : [] ; siet
     for a range (funcs 63):
```

```
riseval ever (equinc); args append (-)
     they return carrier to Charges of the
      Except : return (front (" 11") ))
    return (will see "a" else coget Look (), (w)
det sate-eval (elija):
   thy: up - etual expr (excess (th () 13), (), x, () (()())
   except return fleat ("ins")
   return a 11 material manife (a) else front ("not")
der - se (And ):
  for my or mig (x,y).
     d=sale-eval (this)-4
     If not mate is infinite (4): return let?
   Peturn s/lin (x)
def mutate (ch, how is talk, pm + 0.05, pc + 0.17);
    queschial chilicity in body
    tor I in range (a):
        it random random ( ) com :
         9 (1) = Conton: choice (Light FORES) + TERMS) -
       If Lanear due TERNS)
         it g(5) == "C' = L'0] + (sudom - uniform (-2, 2)
         elil the citel col
    to can hat colo
         of random . Partial I cpc ; c [1] + Fardom . goust (0,0-1)
det crossour (0,4) =
  n=lan (6897); cut = random randot (4 mai)
  det make (quiph):
      3 = pa [q"] [cut] = pb [4] [w+:]; == {}
      for is in enmerale (a) i
         H see "C" coll=[pol" o"] if i cout the pol ["c"])
          - get (1, tombon runiform (-2,2))
```

```
return { "g": g, "c": c}
      return make (a,b), make (b,a)
                              ( and fault) reason & Eguns
def select (pop, fit, k=3):
    i= min (randon: sample (range (len (pop)), k), key = lambda j: fizy
    return ("g": pop[]["g"][:], "c": dict (pap[i]["o"])}
def evolve (fitarget = (-3,3), pts = 64, popn=100, head=10, gens=100.
                               ("Tw") traft mudar stgars
          (x: 0.7) Seed = None):
    if seed: random: seed (seed) as ( ) distribution is a mortal
    tail = head (max (FUNCS, values ()) - 1)+1
    X = [target [0] + (target [1] - target [0])"i /pts-1) for i in range (pts)
    y = [f(x) \text{ for } x \text{ in } X]
    Pop = [init - chrom (head, tail) for in range (popn)]
    best, fit = float ("inf"), None" : (1) aliministic stars for it
    for gen in range (gens):
         fits = [mse (ind, x, y) for ind in pop]
         b=min (range (popn), key = lambda i=fita[i])
         if fits (b) < best: best; fit = fits (b)
         {"9": pop [b] ["g"] [:], "c": dict (pop [b] ["c"])}
         while len (new) < popn: mg > ( ) mobiles modernes si
              pl = select, (pop, fits); p2 = select (pop, fits)
              c) c2 = crossover (pl,p2) if random. random () < cx
              else (p1, p2)
              mutate (ci, head, tail); mutate (cz, head, tail)
              new + = [a, cz] [00 /06 :0 01 5 7110
         POP = new
    return fit, best "best" best)
   B_name . = : ". main " . :
     f=lambda a: x * 3_0.5 * x+math.sin(x)
     print (Best exros: ", exx)
    point (Poeds: ", [safe eval (best, x) for x in [2,-1,0;1,7])
                                  is in enumerale (a) :
     ( L'o Day sels toon it L'o Day s Libo " o"
```

```
Code:
print("Shreya Raj 1BM23CS317")
import math, random
# --- Function set ---
FUNCS = {"+":2,"-":2,"*":2,"/":2,"sin":1,"cos":1,"exp":1,"log":1}
APPLY = {
 "+":lambda a,b:a+b, "-":lambda a,b:a-b, "*":lambda a,b:a*b,
  "/":lambda a,b:a if abs(b)<1e-12 else a/b,
  "sin":lambda a:math.sin(a), "cos":lambda a:math.cos(a),
  "exp":lambda a:math.exp(max(-50,min(50,a))),
  "log":lambda a:math.log(abs(a)+1)
}
TERMS=["x","C"]
def init_chrom(head,tail):
  g=[random.choice(list(FUNCS)+TERMS) for in range(head)]
 g+=[random.choice(TERMS) for _ in range(tail)]
  c={i:random.uniform(-2,2) for i,s in enumerate(g) if s=="C"}
  return {"g":g,"c":c}
def decode(g):
  seq=[];need=1;i=0
 while need>0 and i<len(g):
   sym=g[i];seq.append((sym,i));need-=1
   if sym in FUNCS: need+=FUNCS[sym]
```

```
i+=1
  return seq
def eval_expr(seq,i,x,c):
  s,pos=seq[i]
  if s in FUNCS:
    args=[];j=i+1
   for _ in range(FUNCS[s]):
     v,j=eval_expr(seq,j,x,c);args.append(v)
    try:return (APPLY[s](*args),j)
    except: return (float("inf"),j)
  return (x if s=="x" else c.get(pos,0),i+1)
def safe_eval(ch,x):
  try:v,_=eval_expr(decode(ch["g"]),0,x,ch["c"])
  except:return float("inf")
  return v if math.isfinite(v) else float("inf")
def mse(ch,X,Y):
  s=0
 for x,y in zip(X,Y):
    d=safe_eval(ch,x)-y
   if not math.isfinite(d): return 1e12
    s+=d*d
  return s/len(X)
```

```
def mutate(ch,head,tail,pm=0.05,pc=0.1):
 g,c=ch["g"],ch["c"];n=len(g)
 for i in range(n):
   if random.random()<pm:</pre>
     g[i]=random.choice((list(FUNCS)+TERMS) if i<head else TERMS)
     if g[i]=="C":c[i]=random.uniform(-2,2)
     elif i in c:del c[i]
 for i in list(c):
   if random.random()<pc:c[i]+=random.gauss(0,0.1)
def crossover(a,b):
 n=len(a["g"]);cut=random.randint(1,n-1)
 def make(pa,pb):
   g=pa["g"][:cut]+pb["g"][cut:];c={}
   for i,s in enumerate(g):
     if s=="C":c[i]=(pa["c"] if i<cut else pb["c"]).get(i,random.uniform(-2,2))
   return {"g":g,"c":c}
 return make(a,b),make(b,a)
def select(pop,fit,k=3):
 i=min(random.sample(range(len(pop)),k),key=lambda j:fit[j])
 return {"g":pop[i]["g"][:],"c":dict(pop[i]["c"])}
def evolve(f,target=(-3,3),pts=64,popn=100,head=10,gens=100,cx=0.7,seed=None):
 if seed:random.seed(seed)
 tail=head*(max(FUNCS.values())-1)+1
```

```
X=[target[0]+(target[1]-target[0])*i/(pts-1) for i in range(pts)]
 Y=[f(x) \text{ for } x \text{ in } X]
  pop=[init_chrom(head,tail) for _ in range(popn)]
  best,fit=float("inf"),None
 for gen in range(gens):
   fits=[mse(ind,X,Y) for ind in pop]
    b=min(range(popn),key=lambda i:fits[i])
   if fits[b]<best:best,fit=fits[b],{"g":pop[b]["g"][:],"c":dict(pop[b]["c"])}</pre>
    new=[fit]
   while len(new)<popn:
     p1=select(pop,fits);p2=select(pop,fits)
     c1,c2=crossover(p1,p2) if random.random()<cx else (p1,p2)
     mutate(c1,head,tail);mutate(c2,head,tail)
     new+=[c1,c2]
    pop=new
   if (gen+1)%20==0:print("Gen",gen+1,"Best",best)
  return fit, best
if __name__=="__main__":
 f=lambda x:x**3-0.5*x+math.sin(x)
  best,err=evolve(f,gens=100)
  print("Best error:",err)
  print("Preds:",[safe_eval(best,x) for x in [-2,-1,0,1,2]])
```

## Output:

```
Shreya Raj 1BM23CS317

Gen 20 Best 0.2711388144567344

Gen 40 Best 0.08276309290483551

Gen 60 Best 0.02638736211355352

Gen 80 Best 0.0038234107968136122

Gen 100 Best 0.0035796711491496228

Best error: 0.0035796711491496228

Preds: [-7.947649551427884, -1.2578413596603832, 0.0, 1.2578413596603832, 7.947649551427884]
```