Cellular Potts Model

f(x)

In[62]:=
```
(* cross and moore neighbour indices within bounds of the canvas *)
crossneighbours[{p_, q_}] := DeleteCases[{
    {p, q - 1},
    {p, q + 1},
    {p - 1, q},
    {p + 1, q}}, {OrderlessPatternSequence[x_ /; x ≤ 0 || x > n, _]}, {1}];

mooreneighbours[{p_, q_}] := DeleteCases[{
    {p, q - 1},
    {p, q + 1},
    {p - 1, q - 1},
    {p - 1, q},
    {p - 1, q + 1},
    {p + 1, q - 1},
    {p + 1, q},
    {p + 1, q + 1}}, {OrderlessPatternSequence[x_ /; x ≤ 0 || x > n, _]}, {1}];
```

In[64]:=
```
(* boundary points of all cells together *)
boundarymerger[assoc_] := DeleteDuplicates[Cases[Normal@assoc, {__Integer}.., {-2}]]
```

In[65]:=
```
(* local adhesive energy *)
Clear@localAdhesionEnergyCalc;
localAdhesionEnergyCalc[{x_, y_}] := Block[
   {neigh = crossneighbours[{x, y}], elem, C = {{0, 0, 0}, {0, 0, 0}, {0, 0, 0}}, kronec = {{0, 0, 0}, {0, 0, 0}, {0, 0, 0}}, cellid},
   elem = Extract[A, neigh];
   (* counts *)
   With[{ind = A[[x, y]]},
    Do[
     If[j == ind,
      C[[j + 1, ind + 1]] += 1,
      C[[ind + 1, j + 1]] += 1
     ]
     , {j, elem}]
   ];

   (* same cell or dissimilar matrix *)
   With[{cell = pixToCellAssoc[{x, y}]},
    Do[
     cellid = A[[j]];
     If[cellid == cell,
      (*neighbour belongs to the same cell*)
      kronec[[cellid + 1, cell + 1]] += 1 - KroneckerDelta[cell, cellid],
      (*neighbour from a different cell*)
      kronec[[cell + 1, cellid + 1]] += 1 - KroneckerDelta[cell, cellid]
     ]
     , {j, neigh}]
   ];
   Total[{{1, 1, 1}, {1, 0.5, 0.5}, {1, 0.5, 0.5}} kronec C J, 2]
  ];
```

In[67]:=
```
(* total adhesive energy for all cells *)
Clear@globalAdhesionEnergy;
globalAdhesionEnergy[assoc_] :=
 Total[localAdhesionEnergyCalc /@ boundarymerger[assoc]]
```

In[69]:=
```
(* energy because of cell area and perimeter away from rest params *)
Clear@springEnergy;
springEnergy[{celltype_, {_, area_, peri_}}] := Block[{k = Kparam[celltype]},
   k[ka] ((area - a0)^2) + k[kp] ((peri - p0)^2)
  ];
```

In[71]:=
```
(* total energy of the system *)
Clear@globalTotalEnergy;
globalTotalEnergy[assoc_] :=
 Total[Map[springEnergy, assoc]] + globalAdhesionEnergy[assoc]
```

In[73]:=
```
Clear@fun; Clear@xx;
fun = Function[x,
   Which[Length[xx = Union@Extract[A, mooreneighbours@x]] > 1, x,
    Length[xx] == 1 && (Complement[xx, A[[x]]]) ≠ {}, x,
    True, Nothing]
  ];
```

params

In[75]:=
```
n = 100; (* canvas size *)
```

In[76]:=
```
(* adhesion strength *)
j00 = 0; j11 = 6; j22 = 6;
j01 = j10 = 6;
j02 = j20 = 6;
j12 = j21 = 16;

J = ( j00  j01  j02
      j10  j11  j12
      j20  j21  j22 );
```

In[81]:=
```
a0 = 100; p0 = 1; (* rest parameters *)
Kparam = <|1 → <|ka → 1, kp → 2|>, 2 → <|ka → 1, kp → 2|>|>; (*stifness params*)
T = 20;(*temperatures*)
```

In[191]:=
```
iter = 1000; (* number of iterations *)
```

In[85]:=
```
A = ConstantArray[0, {n, n}]; (* empty canvas *)
```

CPM lattice

In[86]:=
```
shift = {30, 30};
pos = shift + # & /@ Position[DiskMatrix[20], 1];
Scan[(A[[Sequence @@ #]] = 1) &, pos]
```

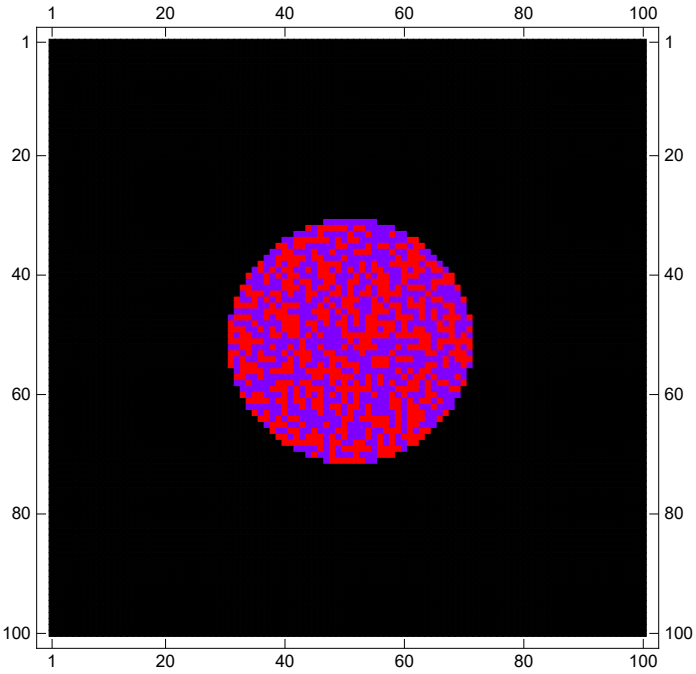
In[89]:=
```
(*
Table[shift={35+i,40+j};
  pos=shift+#&/@Position[DiskMatrix[2],1];
  Scan[(A[[Sequence@@#]]=2)&,pos],{i,Range[1,28,8]},{j,Range[1,20,8]}];
*)
```

In[90]:=
```
saltpepper = Array[RandomChoice[{1, 2}] &, {n, n}];
```

In[91]:=
```
A = A * saltpepper;
```

In[92]:=
```
pltinitial =
 plt = MatrixPlot[A, ColorFunction → Hue, ColorRules → {0 → Black}, ImageSize → Medium]
```

Out[92]=



initial properties

In[192]:=
```
components = MapAt[
   Values@ComponentMeasurements[MorphologicalComponents[#, CornerNeighbors → False],
      {"Mask", "Area", "PerimeterLength"}] &,
   ComponentMeasurements[A, "Mask"], {All, 2}];
assoc = <|
   MapIndexed[First[#2] → {#[[1]], #1[[2]]} &, Flatten@Map[Thread, components]]|>;
assocT = MapAt[Map[fun]@* (Position[ImageData@MorphologicalPerimeter@Image[#], 1] &),
   assoc, {All, 2, 1}];
pixToCellAssoc = AssociationMap[Thread[#[[2, 2, 1]] → First[#]] &, assocT];
pixToTypeAssoc = AssociationMap[Thread[#[[2, 2, 1]] → #[[2, 1]]] &, assocT];
totalE = globalTotalEnergy[assocT];
```

simulate CPM

In[99]:=
```
ls = CreateDataStructure["DynamicArray"]
```

Out[99]=
```
DataStructure[ Type:DynamicArray
               Length:0 ]
```

In[198]:=
```
Off[General::munfl];
Block[{t, prevEnergy = totalE, randpt, prevtype, opts, newcelltype,
```

```
 components, newtotalE, T = T, componentsprev = components, assocprev = assoc,
 assocTprev = assocT, assoc = assoc, assocT = assocT, p, newlatticesite, neighbours,
 cellnum, pixToCellAssoc = pixToCellAssoc, pixToTypeAssoc = pixToTypeAssoc},
t = 0;
Monitor[
 While[t < iter,
  (* pick a random boundary pt and get its 4-neighbours for mutation *)
  p = assocT[RandomChoice@*Range@*Length@assocT][[2, 1]];
  If[p ≠ {},
   randpt = RandomChoice[p]; (*random boundary pt*)
   prevtype = A[[Sequence @@ randpt]];
   cellnum = pixToCellAssoc[randpt];
   neighbours = crossneighbours[randpt];
   opts = Position[Lookup[pixToCellAssoc, neighbours] /. _Missing → 0,
     Except[cellnum], {1}, Heads → False];
   (*Pick[neighbours,KroneckerDelta@@@Thread[{pixToCellAssoc[randpt],
        Lookup[pixToCellAssoc,neighbours]/. _Missing→0}],0];*)
   If[opts ≠ {},
    newlatticesite = RandomChoice@Extract[neighbours, opts];
    newcelltype = Extract[A, newlatticesite];
    A[[Sequence @@ randpt]] = newcelltype;

    (* compute local energy change *)
    components = MapAt[
      Values@ComponentMeasurements[MorphologicalComponents[#,
          CornerNeighbors → False], {"Mask", "Area", "PerimeterLength"}] &,
      ComponentMeasurements[A, "Mask"], {All, 2}];
    assoc = <|MapIndexed[
       First[#2] → {#[[1]], #1[[2]]} &, Flatten@Map[Thread, components]]|>;
    assocT = MapAt[Map[fun]@*(Position[ImageData@MorphologicalPerimeter@Image[#],
          1] &), assoc, {All, 2, 1}];
    newtotalE = globalTotalEnergy[assocT];
    pixToCellAssoc = AssociationMap[Thread[#[[2, 2, 1]] → First[#]] &, assocT];
    (*pixToTypeAssoc=AssociationMap[Thread[#[[2,2,1]]→ #[[2,1]]]&,assocT];*)
    (* acceptance, rejection step *)
    If[newtotalE < prevEnergy,
     (*accept it*)
     ls["Append", newtotalE];
     prevEnergy = newtotalE,
     If[RandomReal[1] < Exp[-(newtotalE - prevEnergy) / T],
      (*accept it by probability*)
      ls["Append", newtotalE];
      prevEnergy = newtotalE,
      (*return to the previous state*)
      A[[Sequence @@ randpt]] = prevtype;
      components = componentsprev;
      assoc = assocprev;
      assocT = assocTprev;
      pixToCellAssoc = AssociationMap[Thread[#[[2, 2, 1]] → First[#]] &, assocT];
      (*pixToTypeAssoc=AssociationMap[Thread[#[[2,2,1]]→ #[[2,1]]]&,assocT];*)
      ls["Append", prevEnergy];
```

```
        ]
       ];
      ];
      componentsprev = components;
      assocprev = assoc;
      assocTprev = assocT;
      plt =
       MatrixPlot[A, ColorFunction → Hue, ColorRules → {0 → Black}, ImageSize → Medium];
      (*Export["C:\\Users\\aliha\\Desktop\\save\\"<>ToString[t]<>".jpg",plt];*)
      t += 1
     ]
    ], {t, plt}]
  ];
On[General::munfl];
```

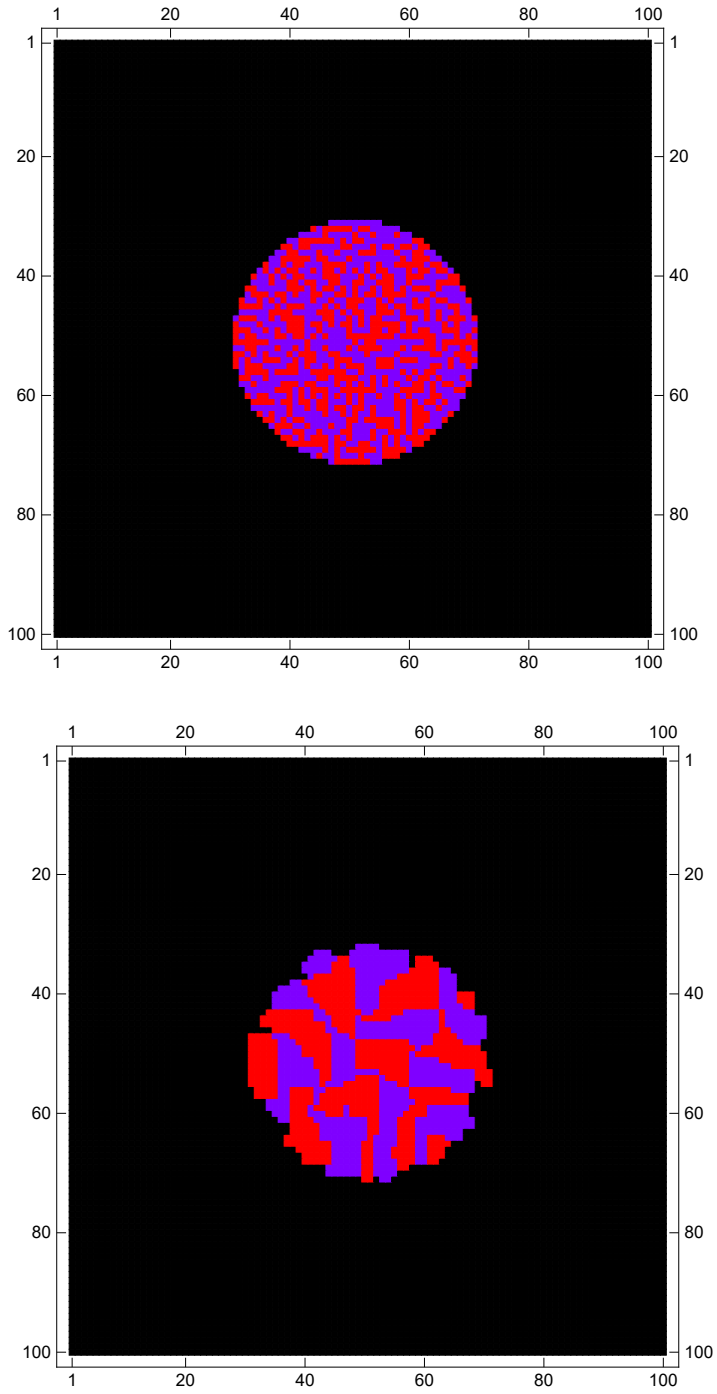
results

In[201]:=
```
ls["Length"]
```

Out[201]=
```
14670
```

In[202]:=
```
Row[{pltinitial, plt}]
```

Out[202]=



In[203]:=
```
ListLinePlot[Normal@ls, PlotStyle → {Thick, Black}, Filling → Axis,
 AxesStyle → Directive[Black, Bold, 12],
 AxesLabel → {"Iteration", "Energy"}, ImageSize → 512, PlotRange → All]
```

Out[203]=