The Ising Model 100 x 100

A two-dimensional Ising model. See *Thermal Physics*, by Daniel V. Schroeder, pg. 349.

This version uses Compile[ ] for the inner functions of the main loop -- that led to more than a factor of 2 speedup.

In[3]:=
```
Clear["Global`*"]
```

In[4]:=
```
initialize[L_] := Table[RandomChoice[{-1, 1}], {x, 1, L}, {y, 1, L}];
neighbors = Compile[{{i, _Integer}, {j, _Integer}, {L, _Integer}},
   {
    {If[i > 1, i - 1, L], j}, (*left *)
    {If[i < L, i + 1, 1], j}, (* right *)
    {i, If[j > 1, j - 1, L]}, (* top *)
    {i, If[j < L, j + 1, 1]}
   }];
eij = Compile[{{s, _Integer, 2}, {i, _Integer}, {j, _Integer}, {L, _Integer}},
   -s[[i, j]] Sum[s[[x[[1]], x[[2]]]], {x, neighbors[i, j, L]}]];
ΔU = Compile[{{s, _Integer, 2}, {i, _Integer}, {j, _Integer},
    {L, _Integer}}, -2 * eij[s, i, j, L]];
etot[s_, L_] := N[(1/2) Sum[eij[s, i, j, L], {i, 1, L}, {j, 1, L}]] / L^2
(* Total energy per spin *)
Mtot[s_, L_] :=
 N[Sum[Sum[s[[i, j]], {i, 1, L}], {j, 1, L}]] / L^2 (* Net magnetic moment per spin *)
```

The plain ‘run’ version of the function operates on the global ‘s’ array. This is useful for Dynamic[] to show the development. It can not be used in a ParallelTable.

In[10]:=
```
run[T_, Nsteps_, L_] := Module[{i, j, Ediff},
  Do[i = RandomInteger[{1, L}]; j = RandomInteger[{1, L}];
   Ediff = ΔU[s, i, j, L];
   If[Ediff ≤ 0,
    s[[i, j]] = - s[[i, j]];,
    If[RandomReal[1.0] < Exp[-Ediff/T],
     s[[i, j]] = -s[[i, j]];
    ]
   ];, {n, 1, Nsteps}];
  s(* Return the new array *)
 ]
```

The ‘prun’ version of the function operates on a local copy of the global ‘s’ array. This is not useful for Dynamic[] to show the development, but it *can* be used in a ParallelTable.

In[11]:=
```
prun[s0_, T_, Nsteps_, L_] := Module[{s, i, j, Ediff},
  s = s0; (* Argument tensor is passed in read-only! *)
  PrintTemporary[DateString[], " Starting run T = ", T];
  Do[i = RandomInteger[{1, L}]; j = RandomInteger[{1, L}];
   Ediff = ΔU[s, i, j, L];
   If[Ediff ≤ 0,
    s[[i, j]] = - s[[i, j]],
    If[RandomReal[1.0] < Exp[-Ediff/T],
     s[[i, j]] = -s[[i, j]]
    ]
   ];, {n, 1, Nsteps}];
  PrintTemporary[DateString[], " Ending run T = ", T];
  s(* Return the new array *)
 ]
```

Utility Functions

In[12]:=
```
show[s_, T_] :=
 ArrayPlot[s, MaxPlotPoints → Infinity, ColorRules → {-1 → Blue, 1 → Yellow},
  PlotLabel → T, LabelStyle → Larger, ImageSize → Large]
showSmall[s_] := ArrayPlot[s, MaxPlotPoints → Infinity,
  ColorRules → {-1 → Blue, 1 → Yellow}, , LabelStyle → Larger, ImageSize → Large]
```

Saving results

In[14]:=
```
tofname[T_, suffix_] :=
 ToString[StringForm["Ising-2D-T``.``", NumberForm[T, {3, 2}], suffix]]
saveImages[spins_, Ts_] :=
 Do[Export[tofname[Ts[[i]], "png"], show[spins[[i]], Ts[[i]]], {i, 1, Length[spins]}]

saveWork[Ts_, spins_, Evst_, MvsT_] := Module[{},
  Export["Ising-2D-EvsT.txt", EvsT, "Table"];
  Export["Ising-2D-MvsT.txt", MvsT, "Table"];
  Do[{s = spins[[i]]; T = Ts[[i]]; Save[tofname[T, "m"], T]; Save[tofname[T, "m"], s];},
   {i, 1, Length[spins]}];
  Do[Export[tofname[Ts[[i]], "png"], show[spins[[i]], Ts[[i]]], {i, 1, Length[spins]}]
 ]
```

Initialization

In[17]:=
```
L = 100; Nsteps = 100 L^2;
s = initialize[L];
```

This picture is pretty to watch, but almost doubles the calculation time. Delete it if you want to get done more quickly.

In[ ]:=
```
Dynamic[show[s, T]]
```

First, look at high temperature. The net energy and magnetization (per site) are quite small.

In[ ]:=
```
s = initialize[L];
```

In[ ]:=
```
T = 5.0; run[T, Nsteps, L];
etot[s, L]
Mtot[s, L]
```

Next, look at relaxation at low temperature. The net energy and magnetization (per site) are quite

significant.

In[ ]:=
```
T = 0.02; run[T, Nsteps, L];
etot[s, L]
Mtot[s, L]
```

Initialize arrays to scan a range of temperatures. For conditions close to the critical point, we need to take more steps. Use the same initial condition for all temperatures.

In[19]:=
```
Ts =    {0.5, 1.0, 2.0, 2.1,  2.2,  2.27, 2.5,  3.0, 4.0};
steps = {10,   20,  40,  400, 800,  400,  200,  40,  10} * Nsteps ;
s0 = initialize[L];
spins = Table[s0, {i, 1, Length[Ts]}];
```

This took about 2 hours on my system (4 cores) for a first pass for a 100x100 array. For conditions close to the critical point, we need to take more steps.

In[ ]:=
```
AbsoluteTiming[ spins = ParallelTable[ prun[spins[[i]], Ts[[i]], steps[[i]], L],
    {i, 1, Length[Ts]}, Method → "FinestGrained"];]
```

In[ ]:=
```
EvsT = Table[{Ts[[i]], etot[spins[[i]], L]}, {i, 1, Length[spins]}];
MvsT = Table[{Ts[[i]], Abs[Mtot[spins[[i]], L]]}, {i, 1, Length[spins]}];
```

In[ ]:=
```
ListPlot[EvsT, LabelStyle → Larger,
 FrameLabel → {"T", "E"}, Frame → True, ImageSize → Large]
ListPlot[MvsT, LabelStyle → Larger,
 FrameLabel → {"T", "M"}, Frame → True, ImageSize → Large]
```

In[ ]:=
```
Table[show[spins[[i]], Ts[[i]]], {i, 1, Length[spins]}]
```

In[23]:=
```
SetDirectory[NotebookDirectory[]]
```

Out[23]=
```
/home/faculty/doughera/335/2022/lectures-dev/Ch08
```

In[ ]:=
```
Save["Ising-2D-results.m", spins]
Save["Ising-2D-results.m", Ts]
saveImages[spins, Ts];
```

In[24]:=
```
Get["Ising-2D-results.m"](* Pick up here if interrupted. *);
```

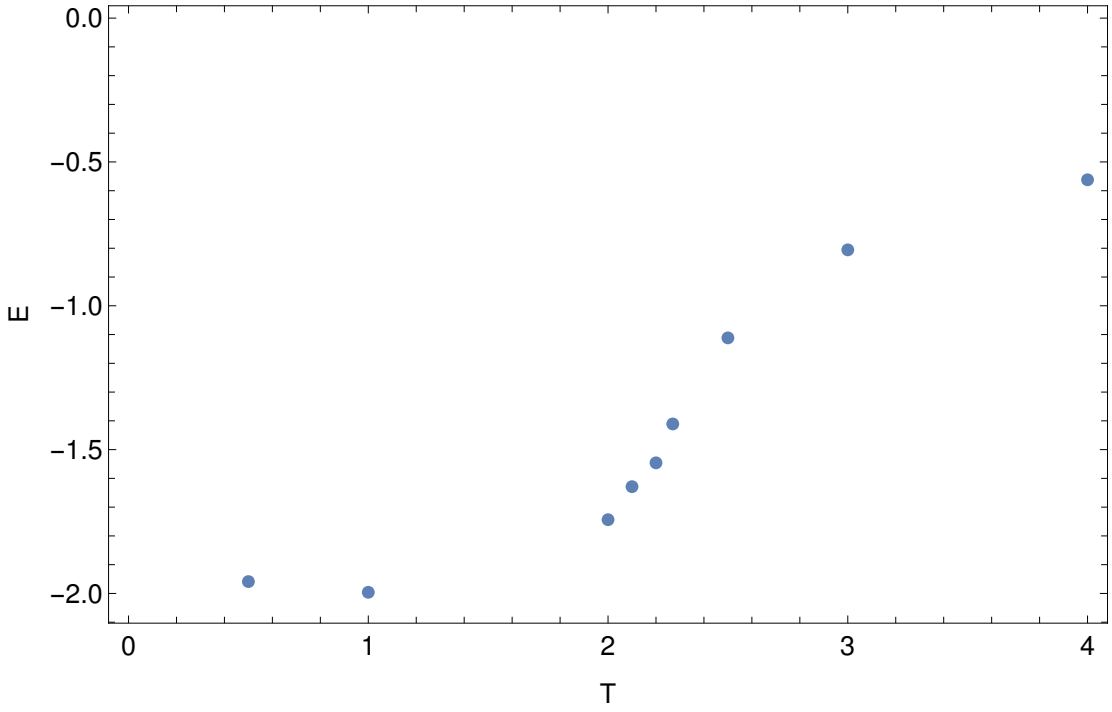
Do further refinements in smallish chunks so we can abort and still save some progress.

In[26]:=
```
niter = 10;
Do[{
  AbsoluteTiming[
   spins = ParallelTable[ prun[spins[[i]], Ts[[i]], steps[[i]], L], {i, 1, Length[Ts]},
       Method → "FinestGrained"];],
  If[FileExistsQ["Ising-2D-results.m"], RenameFile[
    "Ising-2D-results.m", "Ising-2D-results-old.m", OverwriteTarget → True]];
  Save["Ising-2D-results.m", spins];
  Save["Ising-2D-results.m", Ts];
  Print[StringForm["Iteration `` of `` `` ", j, niter, DateString[]],
   ProgressIndicator[j, {0, niter}]]
 }, {j, 1, niter}]
```

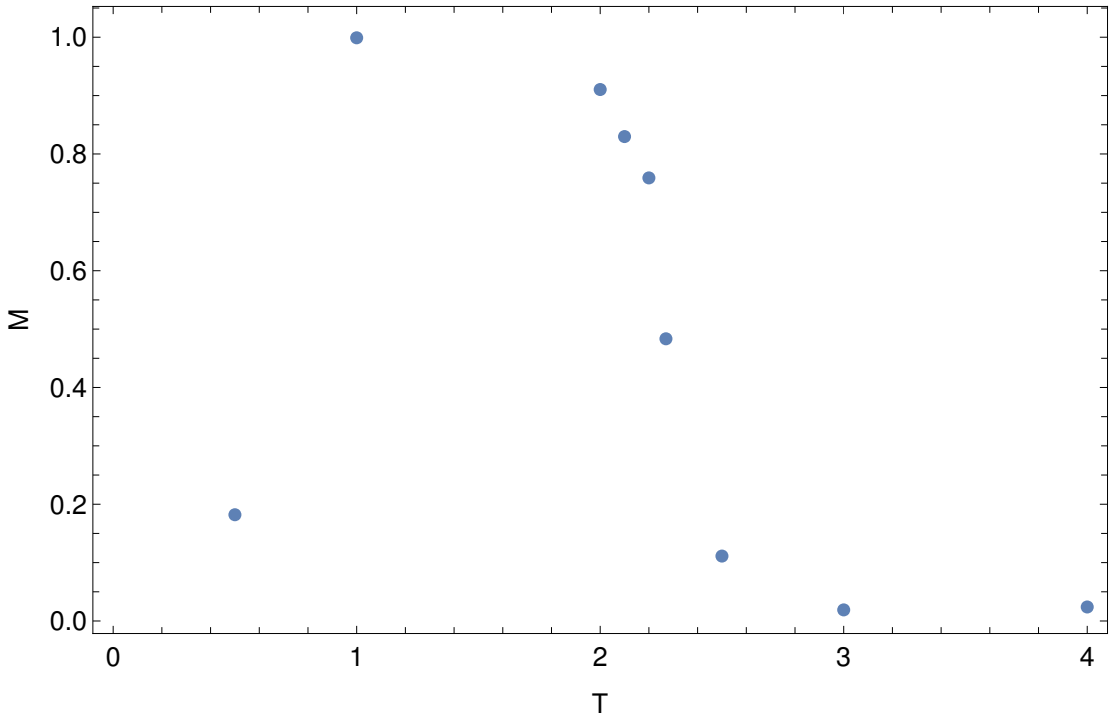
In[28]:=
```
EvsT = Table[{Ts[[i]], etot[spins[[i]], L]}, {i, 1, Length[spins]}];
MvsT = Table[{Ts[[i]], Abs[Mtot[spins[[i]], L]]}, {i, 1, Length[spins]}];
```

In[30]:=
```
ListPlot[EvsT, LabelStyle → Larger,
 FrameLabel → {"T", "E"}, Frame → True, ImageSize → Large]
ListPlot[MvsT, LabelStyle → Larger,
 FrameLabel → {"T", "M"}, Frame → True, ImageSize → Large]
Table[show[spins[[i]], Ts[[i]]], {i, 1, Length[spins]}]
```

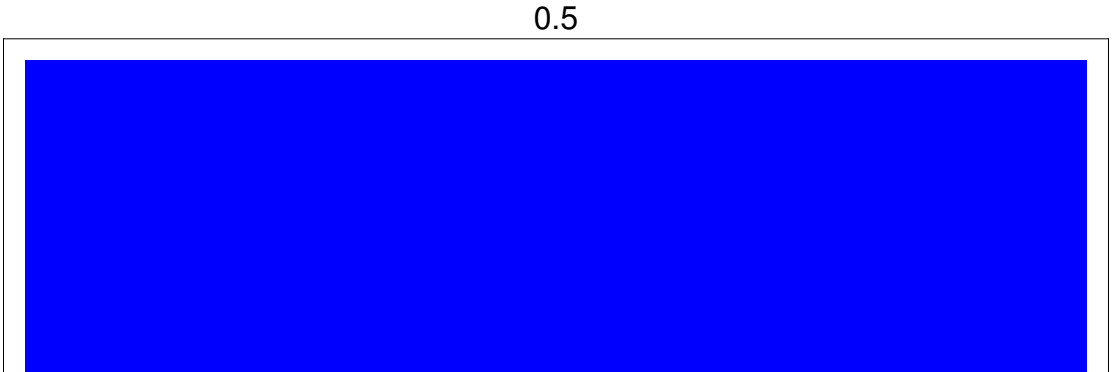
Out[30]=



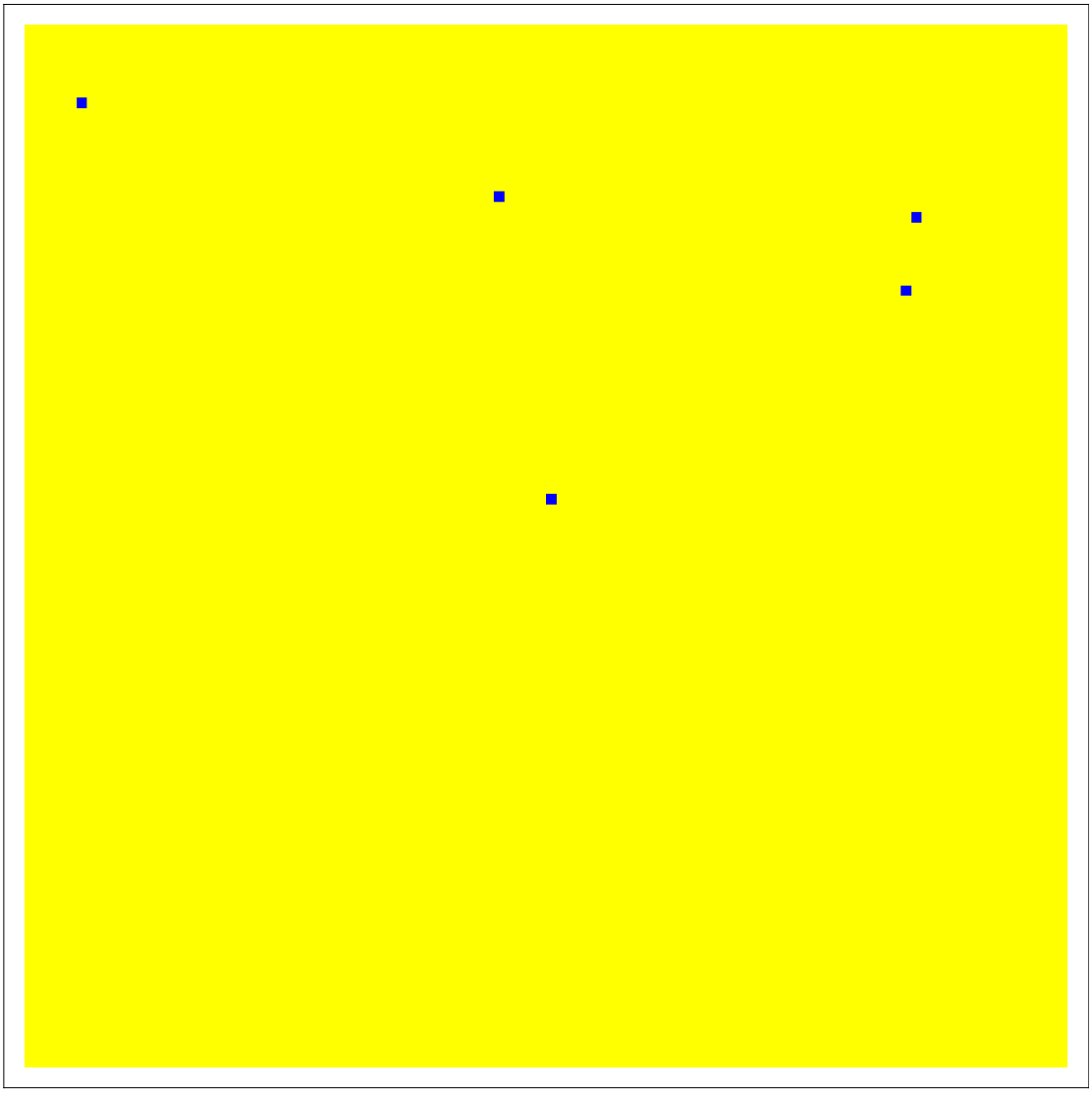
Out[31]=



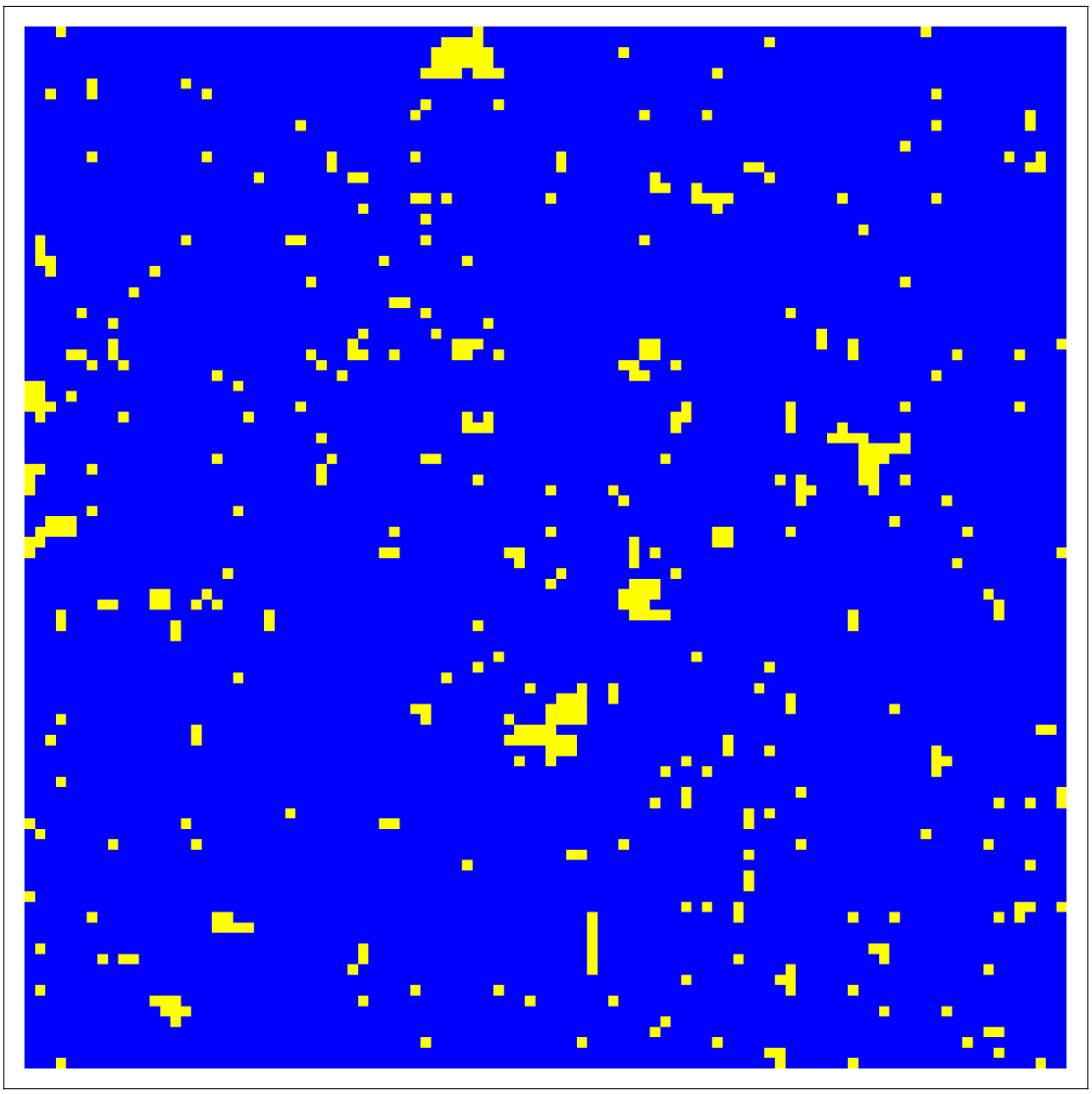
Out[32]=



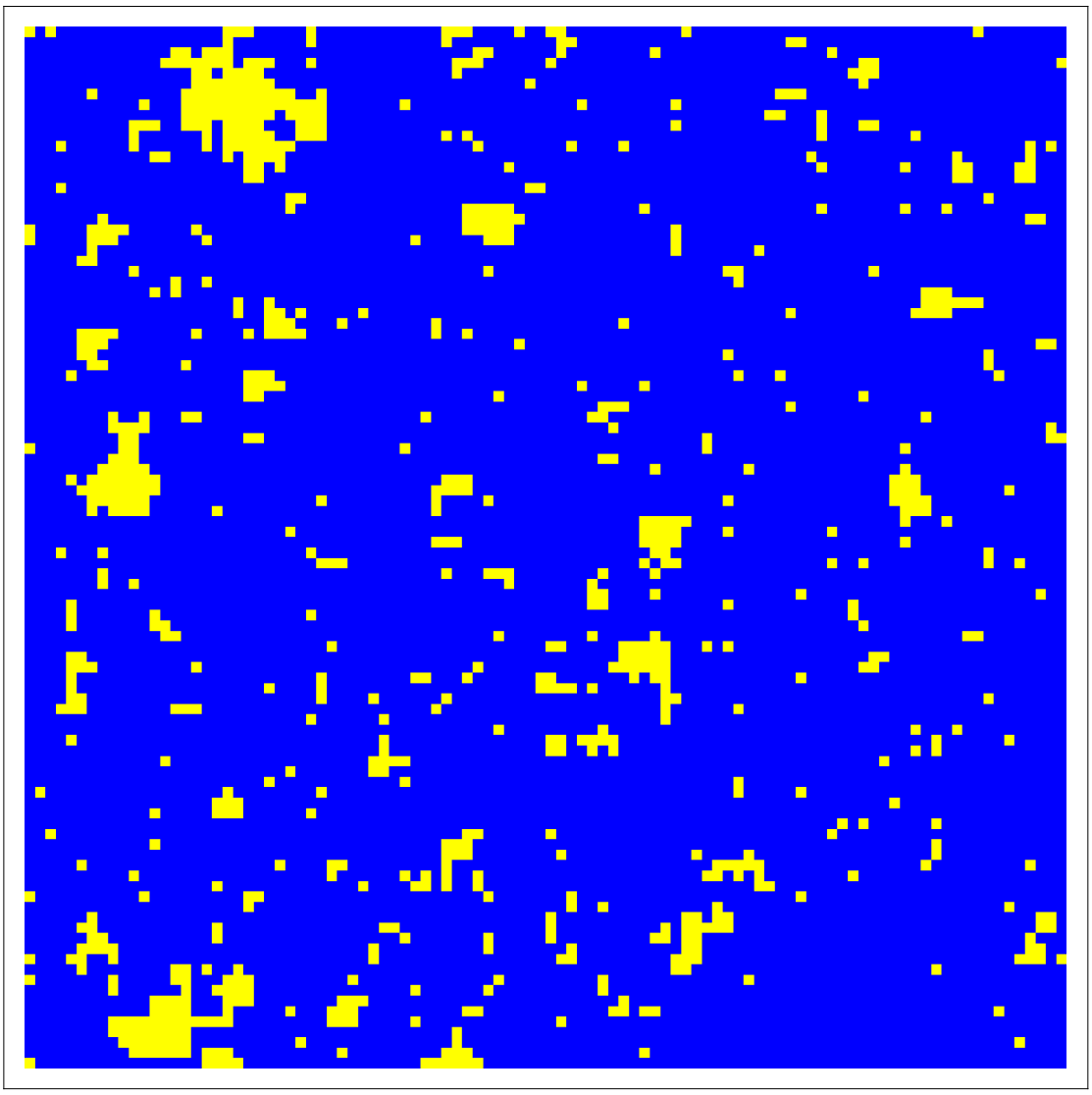
1.



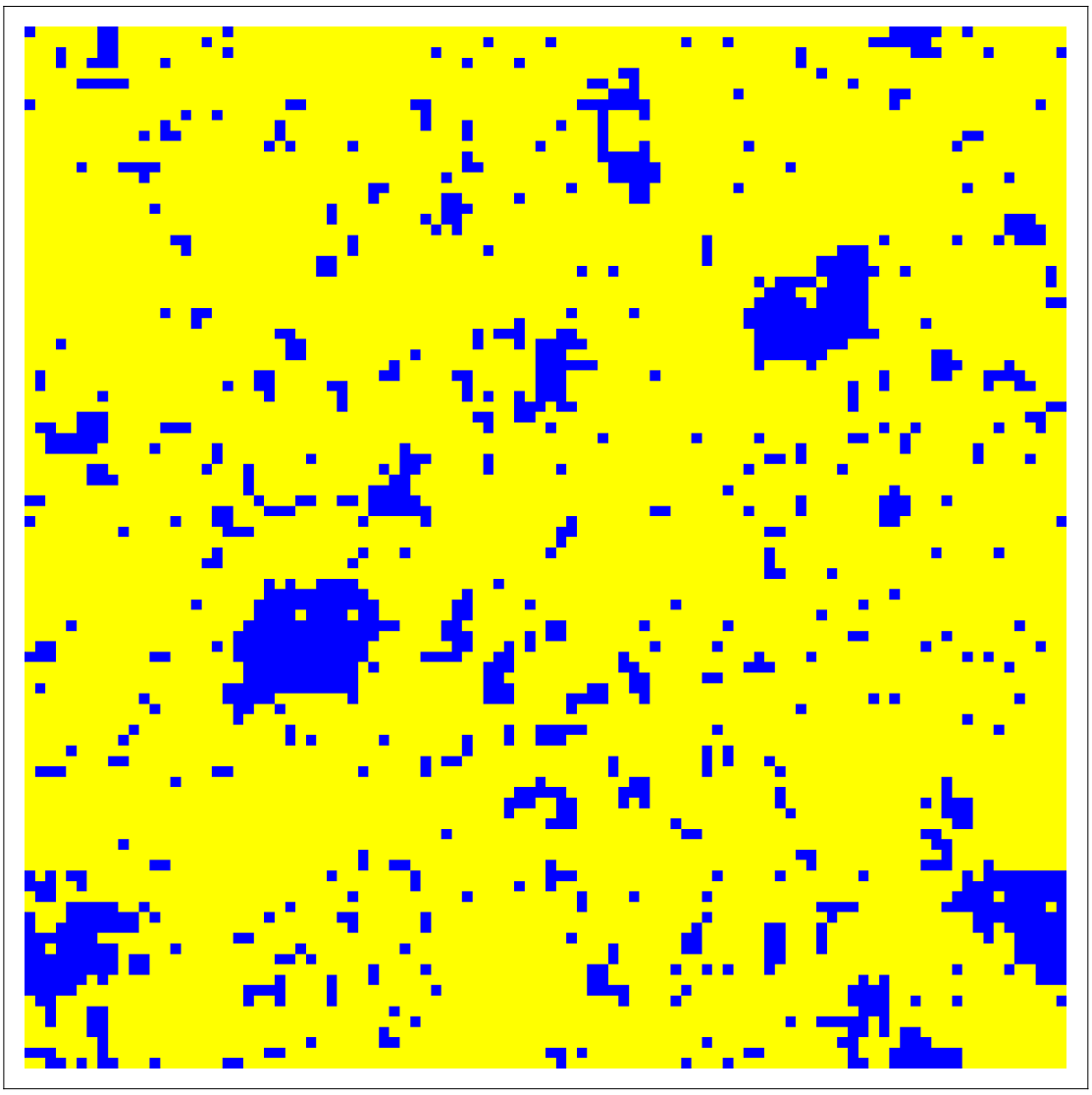
2.



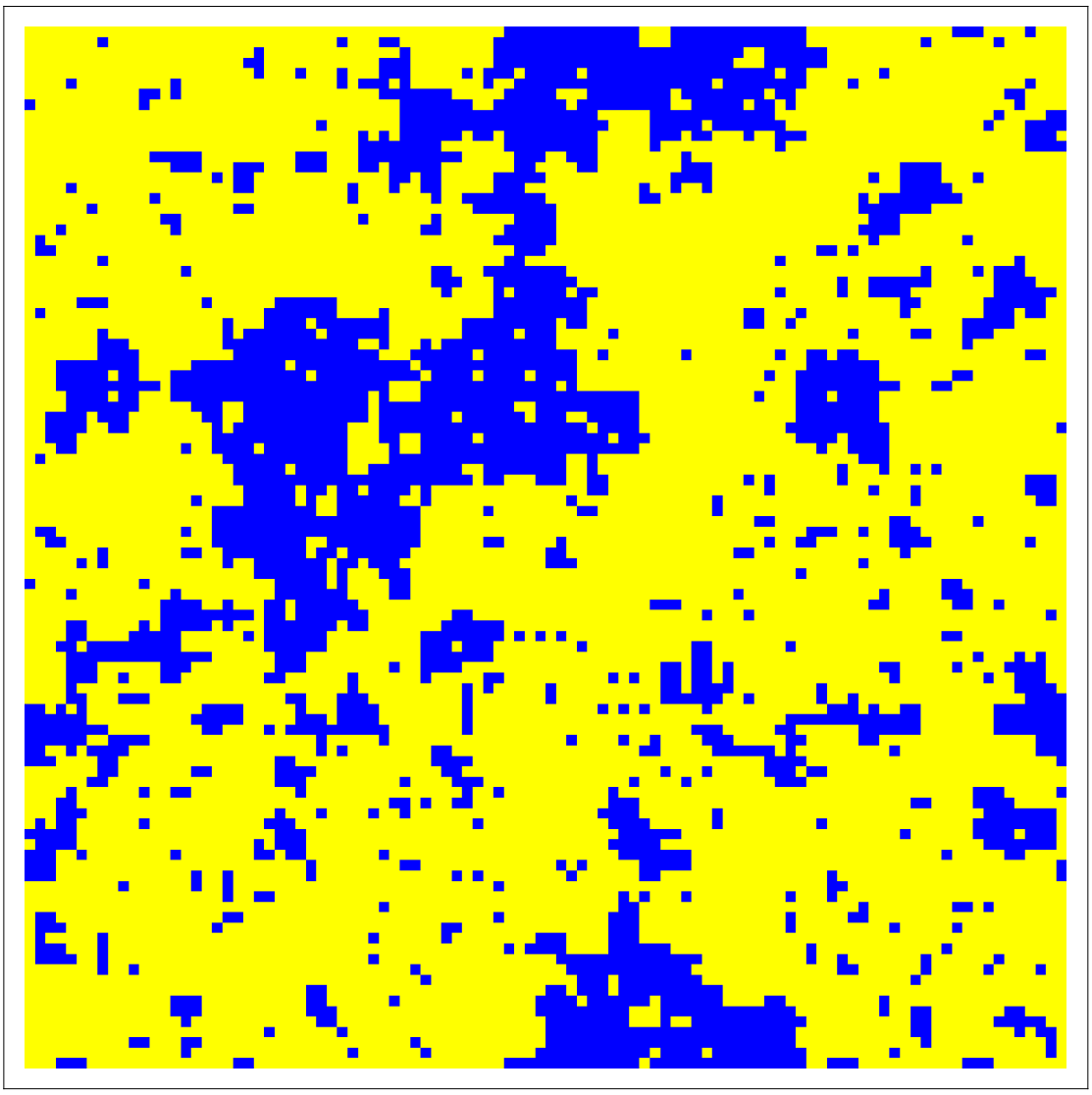
2.1



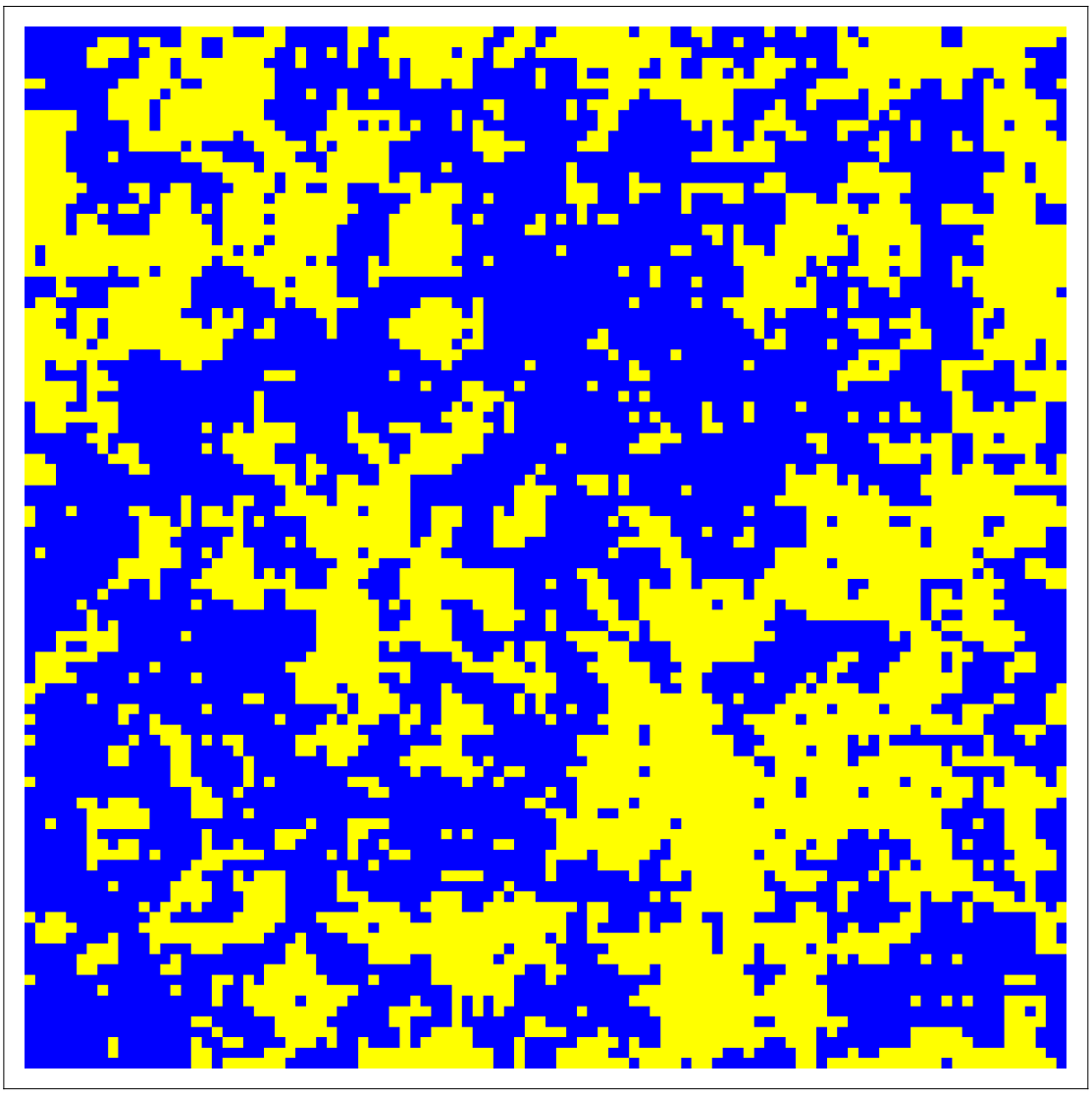
2.2


,

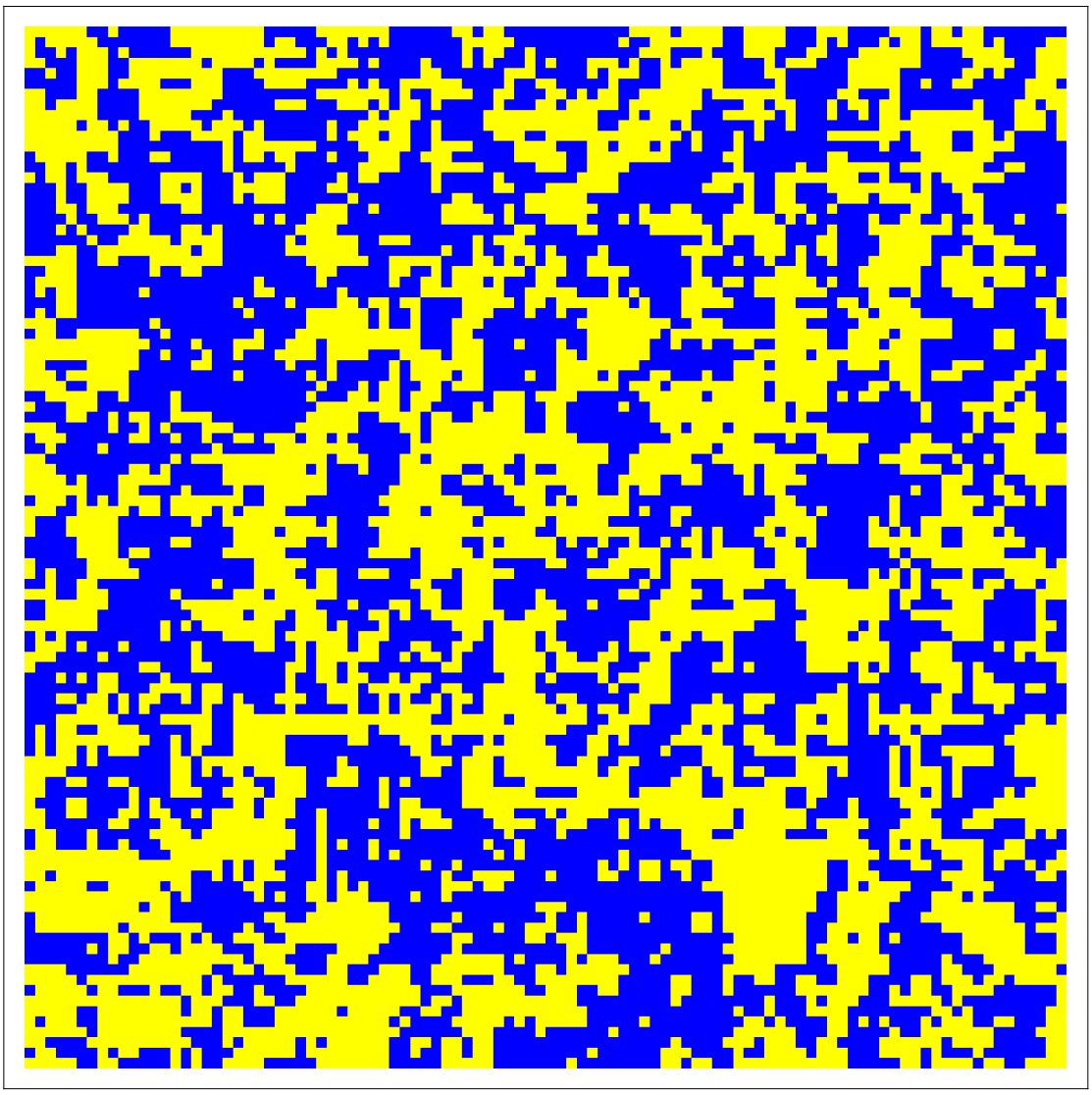
2.27



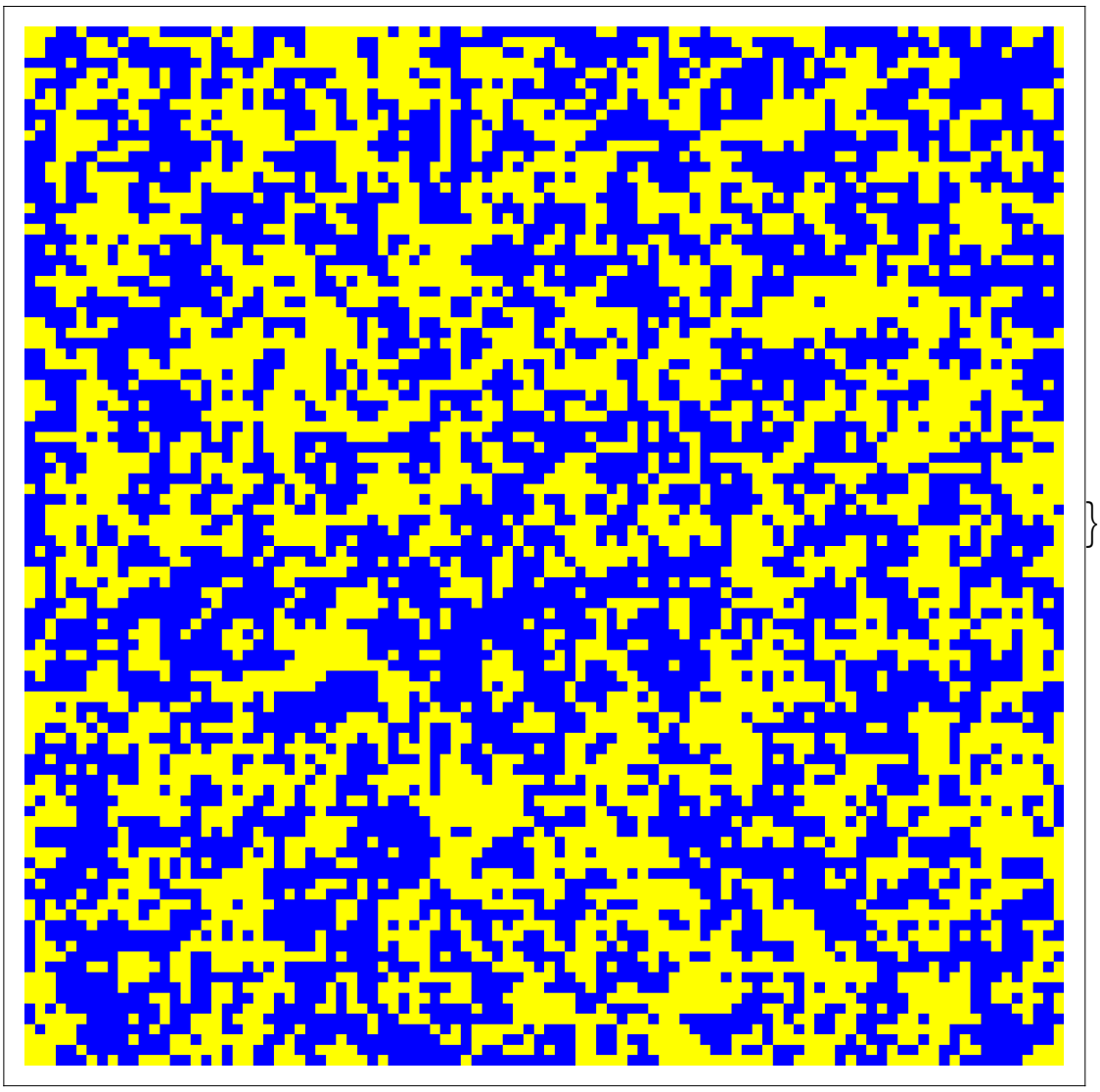
2.5



3.


,

4.



In[36]:=
```
Save["Ising-2D-results.m", spins]
Save["Ising-2D-results.m", Ts]
saveImages[spins, Ts];
```