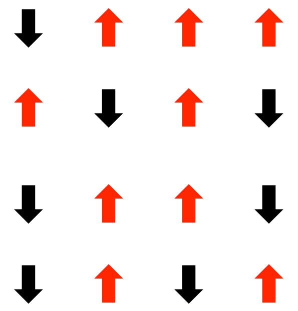
Laboratory: Worm Algorithm for the Ising model:

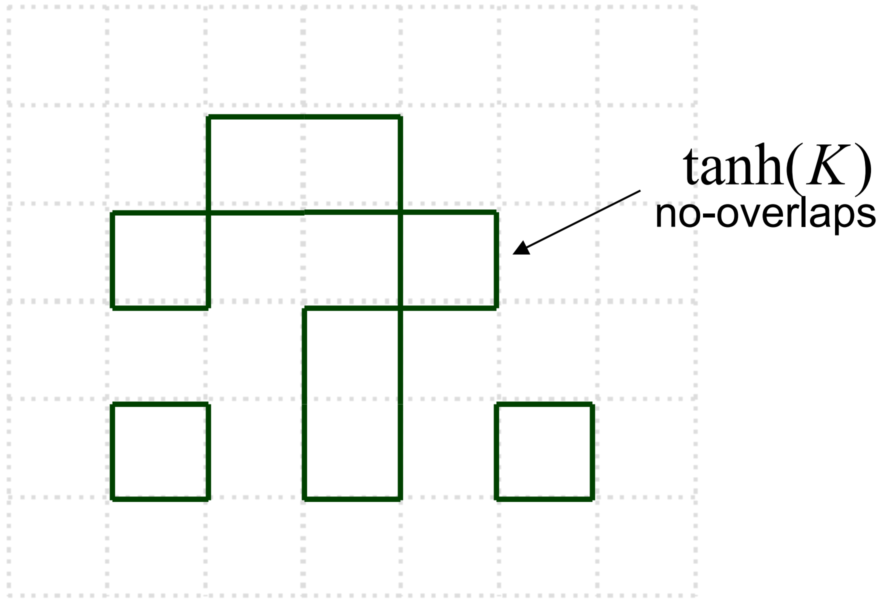
$$-H/T = K\sum_{\langle ij\rangle}\sigma_i\sigma_j \quad \text{with} \quad \sigma_i = \pm 1 \text{ on a square lattice}$$



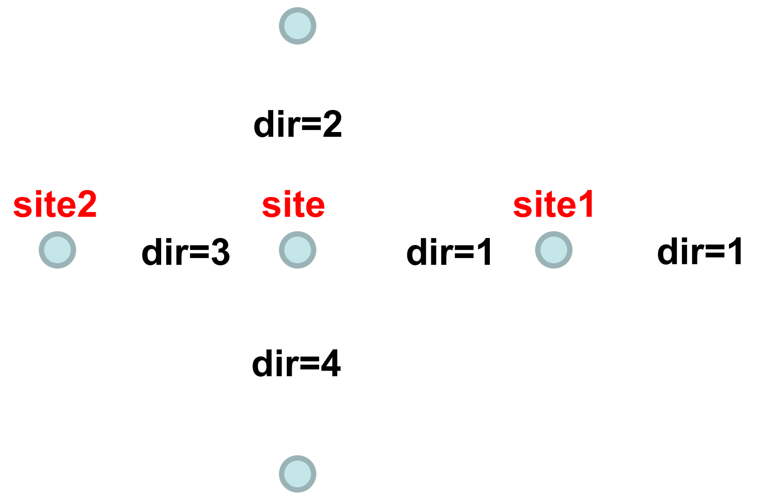
$$Z = \sum_{\{\sigma_i\}}\left(\prod_{\langle ij\rangle} e^{K\sigma_i\sigma_j}\right)$$

$$\Big\Downarrow$$

$$Z/\cosh^{dN}(K) = \sum_{\{N_b=0,1\}}^{loops}\left(\prod_{b=\langle ij\rangle}\tanh^{N_b}(K)\right)$$



Link the lattice with periodic boundary conditions



create an array **nn(site,dir)** which returns the nearest neighbor of **site** in the direction **dir**

for example: **site1= nn(site,1)** and **site= nn(site1,1+dim)**

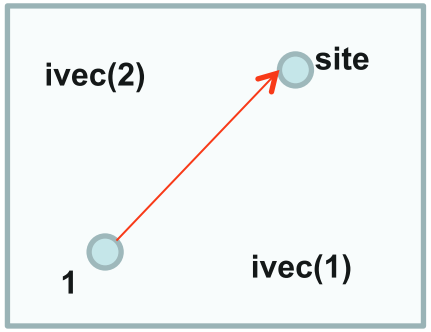
Periodic boundary conditions mean that: **site2= nn(site1,1)** and **site1= nn(site2,1+dim)**

```
integer, parameter :: L=10, dim=2, N=L**dim, dd=2*dim
double precision, parameter ::    JT=0.3d0
double precision :: ratio
integer :: nn(N,dd)
integer :: site,site1
integer :: ivec(dim), i

 ratio=tanh(JT)

DO  site=1,N
                site1=site-1
                DO i=dim,1,-1
                ivec(i)=site1 / L**(i-1)
                site1=site1-ivec(i)*L**(i-1)
                ENDDO
DO i=1,dim
site1=site+L**(i-1)
IF(ivec(i)==L-1) site1=site1-L**i
nn(site,i)=site1
nn(site1,i+dim)=site
ENDDO
ENDDO

READ*,  site, i
Print*, nn(site,i)
END                              !    compile with   ifort
```



Now, we need to initialize the configuration and counters for Z,G,M2,E

Configuration is kept in the array bond(site,dim), the sum of all bond numbers is NB, do not forget Ira and Masha and the distance between them IM(dim), dist

And, of course, we need a random number generator.

```
integer :: bond(N,dim), NB, Ira, Masha, IM(dim), dist
double precision :: Z, G(N), M2, E

double precision :: ugen(97) , c7, cd, cm
integer   :: ij, kl, j, i97, j97

Z=1.d-15
M2=0.d0
E=0.d0
G=0.d0
bond=0
NB=0
Ira=1
Masha=1
IM=0

ij=4836
kl=2748
call SRANMAR(ij,kl)
```

Collect all definitions in one place

serve rndm() generator

"partition function"

magnetization squared

energy

correlation function

number of bond lines

rndm() initialization

We are all set to do the simulation. Just one update!

```
double precision :: step

step=0.d0
DO
step=step+1
call UPDATE

     IF(step>N*1000.d0) THEN

IF(Ira==Masha) THEN
Z=Z+1.d0
E=E+NB
ENDIF

G(dist)=G(dist)+1.d0
M2=M2+1

     ENDIF

IF(MOD(step,1.d7)==0) THEN
PRINT*,  E/Z, M2/Z
ENDIF

ENDDO

 contains
 Subroutine UPDATE

END
```

thermolize!

collect statistics

Z configuration

G configuration; G(0)=Z

print something

The update itself!

```
SUBROUTINE UPDATE
integer :: k, k1, site1, n, nnew

IF(Ira==Masha) THEN
Ira=rndm()*N+1 ;   Masha=Ira   ;  ENDIF

k=rndm()*dd+1 ;  site1=nn(Ira,k)

IF(k>dim) THEN ;  k1=k-dim ;
                  n=bond(site1,k1)
           ELSE ; n=bond(Ira,k)  ;   ENDIF

IF(n==0 .AND. rndm()>ratio) RETURN

nnew=MOD(n+1,2)
IF(k>dim) THEN ; bond(site1,k1)=nnew
IM(k1)=IM(k1)-1  ; IF(IM(k1)<0) IM(k1)=L-1
           ELSE  ; bond(Ira,k)=nnew
IM(k)=IM(k)+1     ; IF(IM(k)==L) IM(k)=0
ENDIF
Ira=site1

dist=1 ;
DO i=1,dim; dist=dist+IM(i)*L**(i-1); ENDDO
NB=NB-n+nnew

END SUBROUTINE UPDATE
```

} If in Z configuration move Ira and Masha elsewhere at random

direction

} bond number

accept/reject

} configuration change

} counters

Formally we are done!

We can now enjoy it and study critical phenomena.
Let' s fix JT=0.44069 (critical point), and calculate susceptibility (second printout of M2/Z for different system sizes

L	$\langle M^2 \rangle$
4.	12.184
8.	41.434
16.	139.56
32.	470.
64.	1578
128.	5290.

The data I have

Plot your results as log(y) *vs* log(x) to observe a nice power –law scaling (given by the slope of your nearly straight-line curve).

Prediction of the Onsager solution: $$\chi = \langle M^2 \rangle \propto L^{1.75}$$

Random number generator is separate!

```
 DOUBLE PRECISION FUNCTION RNDM()

  DOUBLE PRECISION, parameter :: r1=1.d-15, r2=1.d-15
  DOUBLE PRECISION :: RVAL

  RVAL = UGEN(I97) - UGEN(J97)
  IF ( RVAL < 0.D0 ) RVAL = RVAL + 1.0
  UGEN(I97) = RVAL
  I97=I97-1
  IF (I97 == 0) I97 =97
  J97=J97-1
  IF (J97 == 0) J97=97
  C7=C7-CD
  IF (C7 < 0.D0 ) C7=C7+CM
  RVAL = RVAL - C7
  IF ( RVAL .LT. 0.D0 ) RVAL = RVAL + 1.0
  RNDM = max(RVAL-r1,r2)

  END FUNCTION RNDM
```

generator itself

```
SUBROUTINE SRANMAR(IJ,KL)
 INTEGER IRAND
 INTEGER :: i7, k7, l7, j7,ii7, jj7, m7, ij, kl
 DOUBLE PRECISION :: S7, T7
 I7 = MOD(IJ/177, 177) + 2
 J7 = MOD(IJ, 177) + 2
 K7 = MOD(KL/169, 178) + 1
 L7 = MOD(KL, 169)
 DO  ii7 = 1, 97
 S7 = 0.D0
 T7 = 0.5D0
                 DO  jj7 = 1, 24
                 M7 = MOD(MOD(I7*J7, 179)*K7, 179)
                 I7 = J7
                J7 = K7
                K7 = M7
                L7 = MOD(53*L7+1, 169)
                IF (MOD(L7*M7, 64) > 32) S7 = S7 + T7
                T7 = 0.5D0 * T7
                ENDDO
 UGEN(ii7) = S7
 ENDDO
 C7 = 362436.D0 / 16777216.D0
 CD = 7654321.D0 / 16777216.D0
 CM = 16777213.D0 /16777216.D0
 I97 = 97
J97 = 33
RETURN
END subroutine SRANMAR
```

call it to initialize rndm()