**Random walk, self-avoiding random walk**

IMPLICIT NONE
integer saw
integer i,j,is,weight
integer io,jo
integer ne,nemax,nt,ntmax,vmax
double precision rnd,rnds,r2,t,wnow
parameter(saw=1) ! saw=0, random walk, saw=1, self-avoiding walk
parameter(ntmax=100) ! maximum number of time steps
parameter(nemax=100000) ! number of walks in ensemble
parameter(vmax=100) ! max size for visit matrix
double precision r2a(ntmax) ! accumulated average of r**2
double precision wtot(ntmax) ! accumulated "weights" at each step
integer visit(-vmax:vmax,-vmax:vmax) ! keep track of visited sites
Random number generator

c initialize random number generator
    call RANDOM_SEED

50     call RANDOM_NUMBER(rnd)
    call RANDOM_NUMBER(rnds)

One random number (rnds) can be used to decide on +/- Step

Other random number (rnd) can be used to decide whether we move walker in x,y, or z direction
Average $r^{**2}$ for many random walks

do ne=1,nemax ! Nemax realizations of random walk

do nt=1,ntmax

...
Random walk, self-avoiding random walk in 3D

\[ \langle r^2 \rangle \sim N^{6/5} \]

SAW not exactly right! What is wrong?
Each path should be equally likely...

Path 1
\[ P_1 = \frac{1}{4} \times \frac{1}{3} \times \frac{1}{3} \times \frac{1}{3} \]

Path 2
\[ P_2 = \frac{1}{4} \times \frac{1}{3} \times \frac{1}{3} \times \frac{1}{2} \]

\[ P_2 > P_1 \]

In fact, \( P_2 = \frac{3}{2} P_1 \).

This somewhat surprising result shows that some paths will be overrepresented in a random ensemble due to self-intersecting trajectories. The disallowed red path skews ensemble.
We could throw away entire paths...

• If a self-intersecting step is chosen at random, throw away entire path and start over

• Correct statistics… terrible sampling…

• For long enough paths, we hardly ever avoid one self-intersecting step…

• We can apply an “enumeration” technique of Giordano

• Another approach is to weight trajectories
Enumeration a la Giordano… consider 2D SAW

- The array dir(n) selects the direction for the nth step
- Predefine length we are searching (ntmax=20)
- Do project in two-dimensions
Enumeration a la Giordano… consider 2D SAW

• Sample all paths for some nmax (e.g. ntmax=20)

• For nmax=20, \( \sim 10^9 \) paths!

• Hard to go much further
Outline of approach...

Start with \( n=1, \ dir(1)=1 \) for the first step. Set \( \text{visit}(0,0)=1 \).

For each site \( n \) we are at...

1. Check if we have tested each direction... \( \text{dir}(n)=1,2,3,4 \)
   
   If yes, then backtrack \( n=n-1 \), set visit for site to 0 (unvisited)
   
   If no, check and see if the next site is unvisited

When we backtrack, we will consider the next direction from the \( n-1 \) site

Otherwise, if next site unvisited, go to it and mark it as visited, also increment \( \text{dir}(n) \)

If next site is visited, go to next direction (increment \( \text{dir}(n) \)) and again go back to step 1 to see if each direction searched
How do we proceed? When do we end?

- Each path that reaches desired limit is included in averages
- When we backtrack to n=0, we are finished (all paths searched)

For example… if dir(n)=1 searches “up”, and ntmax=3, we
First sample a path of all up arrows and set dir(1)=2, dir(2)=2

Next path… after backtrack…

This is for dir(3)=2… accept it then set dir(3)=3…
Continuing along...

After this step, dir(3)=3, corresponding to a “downward” step which revisits a site… so increment dir(2)=4.

We accept this one and increment dir(2)…. But then dir(2)=5, so we are done with this “family” of paths, so we backtrack….
And more...

Last step before backtrack...

Since \( \text{dir}(2) = 2 \), we must consider all paths that have an “up” then a “right” step… start with the path at the left which is for \( \text{dir}(3) = 1 \)… accept and set \( \text{dir}(3) = 2 \).
Another approach…

• Random paths with appropriate weights…

• Weight path by factors 4-possible paths

Path 1

Weight factor 3

Path 2

Weight factor 2
Results for 100,000 random walks, with and without weights for $N=100$ steps...

- Conclusion is that weights approach agrees with $\nu=3/4$
- Can extend to larger walks than enumeration
Effect of step size... 10,000 and 100,000 random paths to compare statistics...

• Statistics reasonable even for $10^4$... Giordano does $10^9$ for only a 20 step SAW!!
Weight factors in my code…

50 weight=4-visit(i+1,j)-visit(i-1,j)-visit(i,j+1)-visit(i,j-1)

wnow=(1.0d0/3.0d0)*wnow*weight
wtot(nt)=wtot(nt)+wnow

weight= 1, 2, 3 depending on how many paths exist

More possible paths give a higher weight to chosen path

Total weight of path is product of factors for each step

Weight=0 used in case we have a dead end.