Dion Harmon

Problem Set 2b

18.325

(4) **Non-Reversing walk**

5/17/2001

Let $v_i(\mathbf{x},t)$ be the portion of the probability of being at $\mathbf{x}$ at time t due to walks where the walker cannot go in the $\text{sign}(i)\mathbf{e}_{|i|}$ direction in the next time step. Let $J = \{-d,\dots,-1,1,\dots,d\}$, and for notational convenience let $\mathbf{e}_{-|i|} = -\mathbf{e}_{|i|}$. We wish to find the limiting distribution of $p(\mathbf{x},t) = \sum_{i\in J} v_i(\mathbf{x},t)$. If δx and δt are the space and time steps respectively, then by definition

$$v_i(\mathbf{x}, t+\delta t) = \alpha \sum_{\substack{j\in J\\ j\neq -i}} v_j(\mathbf{x}+\delta x\,\mathbf{e}_i, t) = \alpha\left[p(\mathbf{x}+\delta x\,\mathbf{e}_i, t) - v_{-i}(\mathbf{x}+\delta x\,\mathbf{e}_i, t)\right]$$

where $\alpha = 1/(2d-1)$. Let $s_i = v_{-i} + v_i$ and $r_i = v_{-i} - v_i$. To find the expansions for s_i and r_i take the sum and difference of the equations for v_{-i} and v_i. Unless otherwise noted evaluation is at $(\mathbf{x},t)$ and all summations run over all $i\in J$ such that $i>0$. The Taylor expansion of the sum yields

$$s_i + \partial_t s_i\,\delta t + \partial_{tt} s_i \frac{\delta t^2}{2} + \cdots = \alpha\left[\left(2p + \partial_{x_i x_i} p\,\delta x^2 + \cdots\right) - \left(s_i + \partial_{x_i} r_i\,\delta x + \partial_{x_i x_i} s_i \frac{\delta x^2}{2} + \cdots\right)\right].$$

The expansion of the difference yields

$$r_i + \partial_t r_i\,\delta t + \cdots = \alpha\left[\left(-2\partial_{x_i} p\,\delta x - \partial_{x_i x_i x_i} p \frac{2\,\delta x^2}{3!} + \cdots\right) + \left(r_i + \partial_{x_i} s_i\,\delta x + \partial_{x_i x_i} r_i \frac{\delta x^2}{2} + \cdots\right)\right].$$

Manipulating the sum yields

$$s_i = \frac{\alpha}{1+\alpha}\left[2p + \partial_{x_i x_i} p\,\delta x^2 - \partial_{x_i} r_i\,\delta x - \partial_{x_i x_i} s_i \frac{\delta x^2}{2}\right] - \frac{\delta t}{1+\alpha}\partial_t s_i + O(\delta t^2, \delta x^3).$$

Manipulating the difference yields

$$r_i = \frac{\alpha}{1-\alpha}\left[-2\partial_{x_i} p\,\delta x + \partial_{x_i} s_i\,\delta x + \partial_{x_i x_i} r_i \frac{\delta x^2}{2}\right] - \frac{\delta t}{1-\alpha}\partial_t r_i + O(\delta t^2, \delta x^3).$$

Now use these two equations to eliminate the lowest order terms in r_i and s_i on the right-hand sides. The difference equation:

$$\begin{aligned} r_i &= \frac{\alpha}{1-\alpha}\left[-2\partial_{x_i} p\,\delta x + \frac{\alpha}{1+\alpha}\left(2\partial_{x_i} p - \partial_{x_i x_i} r_i\,\delta x\right)\delta x + \partial_{x_i x_i} r_i \frac{\delta x^2}{2}\right] + O(\delta t^2, \delta t\delta x, \delta x^3) \\ &= -\frac{2\alpha}{1-\alpha^2}\partial_{x_i} p\,\delta x + O(\delta t^2, \delta t\delta x, \delta x^3). \end{aligned}$$

The sum equation:

$$\begin{aligned} s_i &= \frac{\alpha}{1+\alpha}\left[2p + \partial_{x_i x_i} p\,\delta x^2\left(1 + \frac{2\alpha}{1-\alpha^2} - \frac{\alpha}{1+\alpha}\right)\right] - \frac{2\alpha\,\delta t}{(1+\alpha)^2}\partial_t p + O(\delta t^2, \delta t\delta x, \delta x^3) \\ &= \frac{\alpha}{1+\alpha}\left[2p + \partial_{x_i x_i} p \frac{\delta x^2}{1-\alpha}\right] - \frac{2\alpha\,\delta t}{(1+\alpha)^2}\partial_t p + O(\delta t^2, \delta t\delta x, \delta x^3). \end{aligned}$$

By definition, $\sum s_i = \sum_{i \in J} v_i = p$. Take the sum over all positive i of the above equation for s_i:

$$p = \frac{\alpha}{1+\alpha} \sum \left[2p + \partial_{x_i x_i} p \frac{\delta x^2}{1-\alpha} \right] - \frac{2\alpha\, \delta t}{(1+a)^2} \sum \partial_t p + O(\delta t^2, \delta t\, \delta x, \delta x^3).$$

For terms independent of i, the summations amount to multiplication by $d = (1+\alpha)/2\alpha$. Thus

$$\begin{aligned} p &= p + \frac{\alpha\, \delta x^2}{1-\alpha^2} \nabla^2 p - \frac{\delta t}{1+\alpha} \partial_t p + O(\delta t^2, \delta t\, \delta x, \delta x^3) \\ \Rightarrow \quad \partial_t p &= \frac{\delta x^2}{2d\, \delta t} \left(\frac{1+\alpha}{1-\alpha} \right) \nabla^2 p + O(\delta t^2, \delta t\, \delta x, \delta x^3) \\ &= \frac{\delta x^2}{2d\, \delta t} \frac{d}{d-1} \nabla^2 p + O(\delta t^2, \delta t\, \delta x, \delta x^3). \end{aligned}$$

The equation for a standard random walk on the integer grid becomes

$$\partial_t p = \frac{\delta x^2}{2d\, \delta t} \nabla^2 p + O(\delta t^2, \delta t\, \delta x, \delta x^3)$$

in the continuum limit. Since $< X_n^2 >$ is n for the normal random walk, we should have

$$\langle X_n^2 \rangle = n \left(\frac{d}{d-1} \right)$$

for this random walk. This formula agrees very well with simulations.

(5) **Self-Trapping Walk**

The major programming problem is efficient collision detection. I considered three methods. One can store a grid of coordinate points and mark them as either traversed or free as you encounter them. This has $O(1)$ access time for each step, but requires prohibitive amounts of storage as dimension and path length grow: $O(N^d)$ where d is dimension and N is the number of steps. You might be able to do a little better, as paths do not travel in straight paths but you will still need a lot of memory.

The second method is simply storing the coordinates the path has traversed in a long list. This takes $O(N)$ storage and $O(N)$ to check for collisions at each step. This can be prohibitive for large N, especially for self-avoiding walks (I use the same structures there).

Finally consider the balanced binary search tree. Store the coordinates of the path in the tree. This takes $O(N)$ storage and $O(\log N)$ to check for collisions at each step (and to add elts). Although it takes much longer to maintain the tree than the list, the tree is much faster to search for large path lengths.

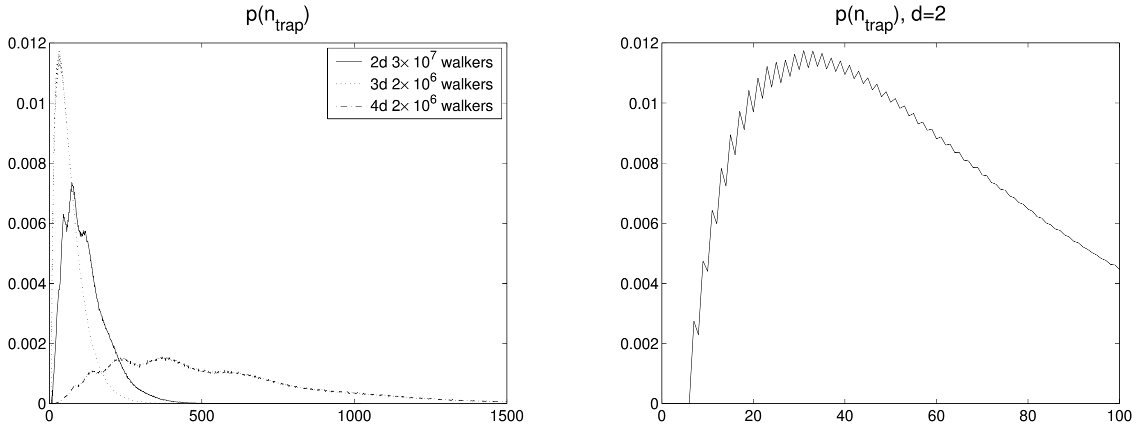
Note: All code is included separately if you are curious.

For 2d, to determine the mean, variance, and approximate distribution of n_{trap} I used 1×10^6 walkers. The mean and variance of n_{trap} appear to be around 70.8 and 2542 respectively. The longest run observed was 957 steps in length.

The code was simple to modify for 3 and 4 dimensions (I only changed one number), but I had to reduce the number of walks as the runs were usually much, much longer. For $3d$ I used 1×10^5 walkers. The mean and variance appeared to be about 3.96×10^3 and 1.38×10^7 respectively.

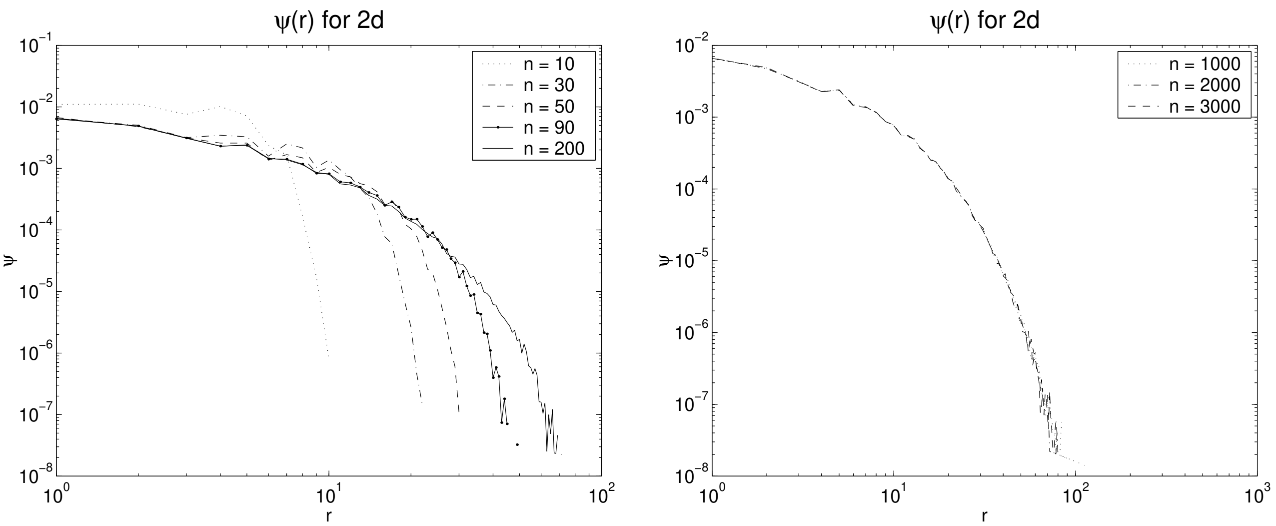
The longest observed run was 41399. The runs for $4d$ were so long, I couldn't run enough to get a reasonable mean or distribution. The runs tended to be over one million steps in length, with 9287242 as the longest observed run of 20 runs. After that the program seg-faulted, possibly due to an attempt to allocate memory when it couldn't get any more.

The plot of the distributions obtained (only defined on integer values) are plotted below for $d = 2, 3$.

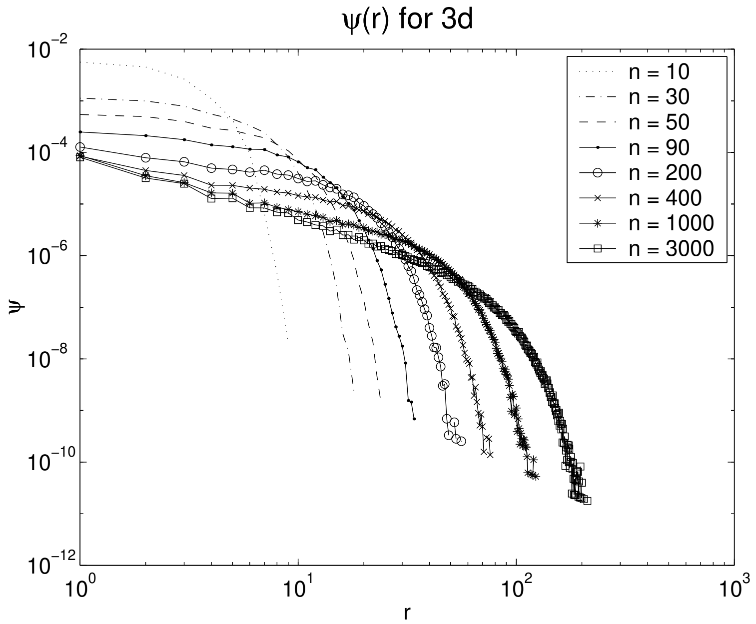


The very high frequency jitter evident in the distributions does not significantly decline with more walkers (for 2d).

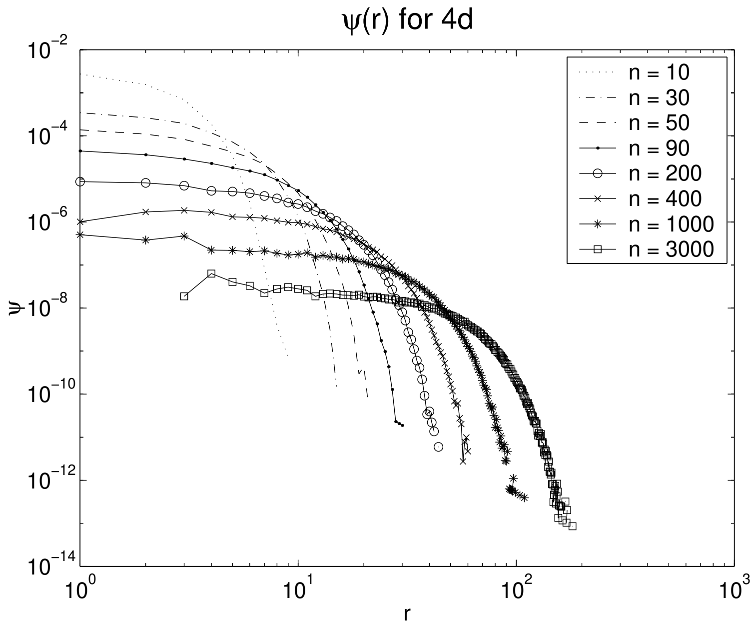
The radial distribution for a given n is discrete as one might imagine: there are only certain radii one may reach on an integer grid. This will become less evident with larger distances from the origin as the spacing becomes denser. However, I have chose bins of witdh 1.0 to produce graphs. This ensures that at least one possible radius may be found in each bin. The following plots show $\psi(r)$ where $p(r) = P(r \le r_n \le r + \Delta r) = 2r^{d-1}\pi^{d/2}\psi(r)\Delta r/\Gamma(d/2)$ where $\Delta r = 1$ and r_n is the radius (either trapped or not) of attempted length n in for a few, representative choices of n. This shows a gradual increase in diffusion for increasing n as we would expect, but once n gets sufficiently large, the walkers can't diffuse any more. The slowing of the diffustion is evident even in smaller quantites, however, as the graph of lower n shows.



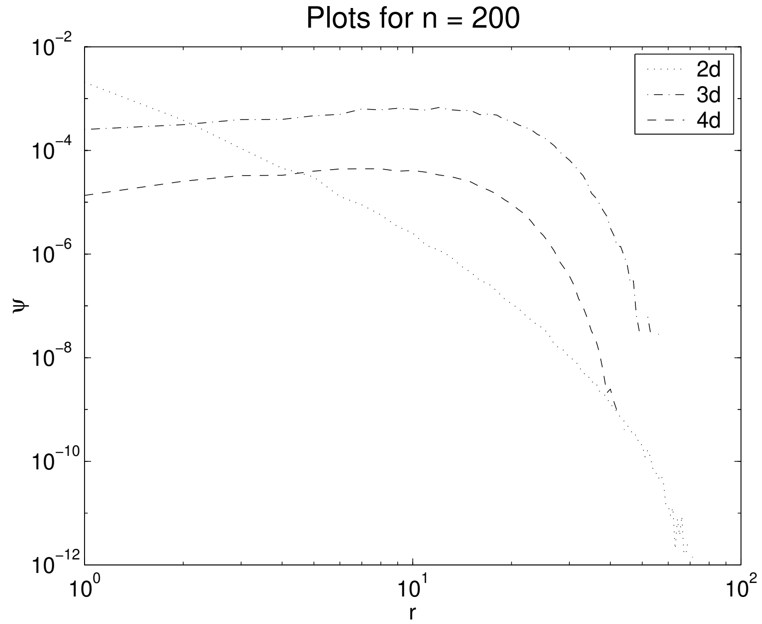
The plot for $d = 3$ exhibits similar properties, but it diffuses away from the origin more with increasing n, and requires a higher n to approach the final asymptotic distribution.



The plot for $4d$ shows much the same trend, and although it may slow down slightly for larger n, no obvious asymptote is being approached. It may become evident with larger n, but I didn't have enough time, and I doubt that it does. (I believe the intersection probability for $d = 4$ is vanishingly small.)

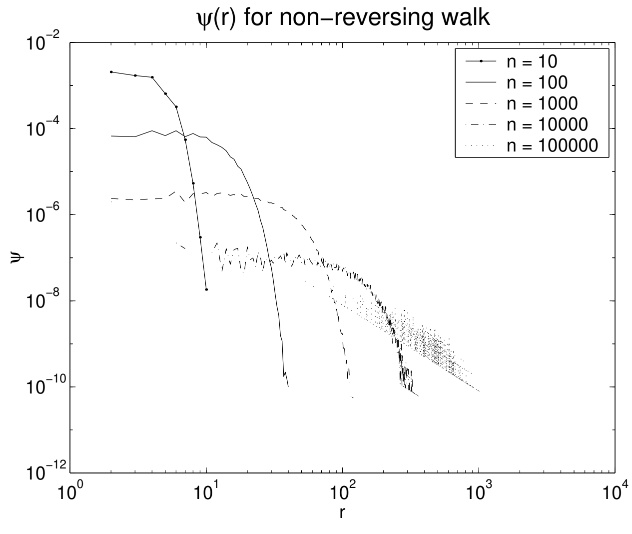


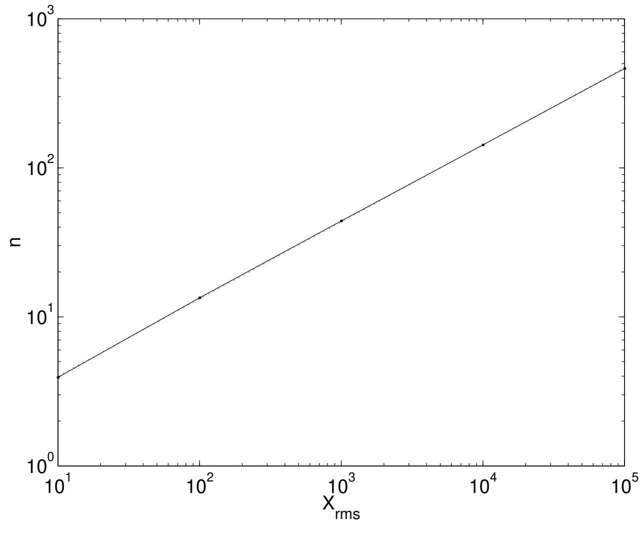
Comparing the distributions for the same n with different dimensions isn't too informative. The only clear trend is that the shapes of the curves are slightly different for different dimension, but remain very similar for a wide range of n. Here is a typical plot. The curves for higher dimension shift to the right longer for higher d as we saw earlier.



(6) **Self-avoiding walk**

The code for self-trapping walks only requires slight modifications to work for self-avoiding walks. I used the enrichment algorithm outlined on the handout from class. This algorithm seems to be a glorified yet effective method of determining how far and how often to backtrack when you encounter difficulties. Since the cost for each try got pretty large with large path lengths, the optimal s (not provable, just a pragmatic choice) was between 3000 and 4000. I only tried small p, and wound up picking $p = 10$. If I had managed to work all the bugs out of my code a bit earlier I probably could have run a few runs (1000 to 10000) of length 1×10^6. Still, the fit looks pretty good as is. The fit is shown below (dotted line—almost invisible) on the plot to the right:





The actual fit yields a relation $x_{rms} = An^{1/D_f}$ where $D_f \approx 0.5171$ and $A \approx 1.2189$. This yields a much smaller D_f than 3/5 or the recent numerical estimates. I may be doing something funny, but I'm pretty sure my simulation is correct.