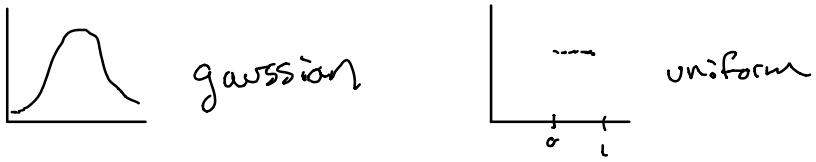
Random Number Generation

1. The random number would be the constructor / destructor. You would make the upper + lower limits private as well as the std. You would make # of points + seed public.

2. I plotted the point vs value assigned and eye-balled it.

3. both gaussians look like random gaussian distributions, uniform random dist. ie



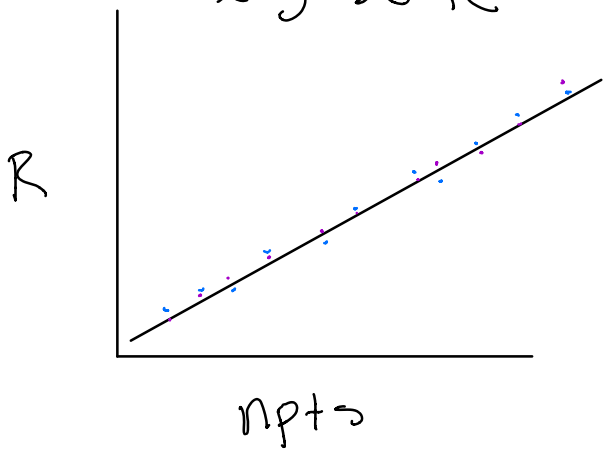
4. Yes you can do a gaussian fit w/ 100,000+ points, 10,000 follows the trend but is not a close fit. ≤1,000 is very poor

Random Walking

1. Assuming we use npts = 100 rms = R/10 where R is net average distance
2. ✓
3. ✓
4. The distance R scales as the square root of N. (Reproduction, I just ran it a bunch of times, added a loop to calculate R & looked @ the trend, I'm not sure how to do this analytically)

5. You would want to use $\sqrt{N}$ as this would be the slope of $R = \sqrt{N}\, r_{rms}$

log scale



$m = 0.5$

this makes perfect sense as $\log \sqrt{N}\, r_{rms}$
$\Rightarrow \frac{1}{2} \sqrt{N}\, r_{rms}$ w/ 1/2 as a slope vs log R

Monte Carlo Integration: Uniform & Gaussian Sampling

1. Plot the function & see where it begins to converge (if it does)

2. The slope is pretty consistently ≈ -0.5, It usually take about 1000 vectors to get consistent close results, usually $1000 < nVec < 10{,}000$ is best.

3. The more dimensions the greater the error. The scaling should be independent of dimension however I only get $N^{1/2}$ scaling consistently ~ 7 & breaks down @ $D = 8$.

4. - Integrand 1 is the same, The second returns the squared sum of gaussians.

   - the result is better because the function contains a gaussian.
   - The first method returned a zero @ $D = 100$ but the new method was able to still integrate & maintained $N^{-1/2}$ dependence on its fit.

C++ Class for a Random Walk

1. ✓

2. - A class can make the main part of the code is cleaner. There will be less variable
   - You can call the class multiple times without having to "recode" the operation
   - If you want the random walk to function in a specific way most of the "moving parts" are private.

   - Con: less flexibility if you want to change limits and/or Δx or Δy

3. • Within the class allow yourself to declare using different distributions ie gaussian vs uniform
   • create different variables & be able to choose your preffered "settings"

4. You can indeed have 2 constructors that call which algorithm you want.