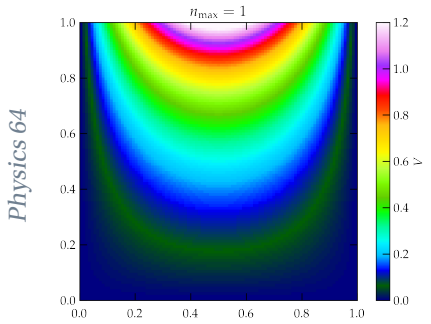
# Problem Set 9 — Solution

**Monday, 6 April 2026**




**Problem 1 – Hyperspheres (10 points)** The surface area and volume of an $n$-dimensional sphere of radius $r$ may be expressed

$$A = S_n r^{n-1} \qquad V = C_n r^n$$

In three dimensions, we know that the area and volume of a "normal" sphere are given by $4\pi r^2$ and $\frac{4}{3}\pi r^3$, respectively, so that $S_3 = 4\pi$ and $C_3 = \frac{4}{3}\pi$. We now seek to develop expressions for arbitrary dimension $n$.

(a) Express $S_n$ in terms of $C_n$.

(b) To evaluate $S_n$ we use the same trick we used to evaluate the Gaussian integral

$$\int_{-\infty}^{\infty} e^{-\alpha x^2}\,\mathrm{d}x = \sqrt{\frac{\pi}{\alpha}}$$

That is, we multiply together two Gaussian integrals and express the result in polar coordinates:

$$\left\{\int_{-\infty}^{\infty} e^{-\alpha x^2}\,\mathrm{d}x\right\}\left\{\int_{-\infty}^{\infty} e^{-\alpha y^2}\,\mathrm{d}y\right\} = \int_{-\infty}^{\infty}\mathrm{d}x\int_{-\infty}^{\infty}\mathrm{d}y\, e^{-\alpha(x^2+y^2)}$$
$$= \int_0^{\infty} e^{-\alpha r^2} 2\pi r\,\mathrm{d}r = \frac{\pi}{\alpha}$$

where $S_2 = 2\pi$. Use this same method to show that

$$\int_0^{\infty} e^{-\alpha r^2} S_n r^{n-1}\,\mathrm{d}r = \left(\frac{\pi}{\alpha}\right)^{n/2}$$

(c) Solve for $C_n$ and $S_n$ in terms of the gamma function, $\Gamma(n+1) = \int_0^{\infty} x^n e^{-x}\,\mathrm{d}x$. Check your expressions for $n = 2$ and $n = 3$, and evaluate $C_6$ and $S_6$.

(d) Use a Monte Carlo method to estimate $C_6$. Make a log-log plot of the absolute value of the difference between the numerical estimate and the analytic value as a function of the number $N$ of "darts" you throw. Does the trend agree with expectations?

(a) The volume of a thin shell at radius $r$ and thickness $\mathrm{d}r$ is

$$\mathrm{d}V = nC_n r^{n-1}\,\mathrm{d}r = S_n r^{n-1}\,\mathrm{d}r \qquad \Longrightarrow \qquad \boxed{S_n = nC_n}$$

(b) Multiplying $n$ Gaussian integrals together, we have

$$\prod_{i=1}^{n}\int_{-\infty}^{\infty} e^{-\alpha x_i^2}\,\mathrm{d}x_i = \left(\frac{\pi}{\alpha}\right)^{n/2} = \int_0^{\infty} e^{-\alpha r^2} S_n r^{n-1}\,\mathrm{d}r$$

(c) Let $u = \alpha r^2$ and $\mathrm{d}u = 2\alpha r\,\mathrm{d}r$. Then

$$\left(\frac{\pi}{\alpha}\right)^{n/2} = S_n \int_0^\infty e^{-u}\left(\frac{u}{\alpha}\right)^{n/2-1}\frac{\mathrm{d}u}{2\alpha} = \frac{S_n\Gamma(n/2)}{2\alpha^{n/2}}$$

which means that

$$\boxed{S_n = \frac{2\pi^{n/2}}{\Gamma(n/2)}}$$

As a check, when $n = 2$, we get $S_2 = 2\pi/\Gamma(1) = 2\pi$ and $C_2 = S_2/2 = \pi$. Since the perimeter and area of a circle are $2\pi r$ and $\pi r^2$, so far, so good.
When $n = 3$, the formula gives $S_3 = 2\pi^{3/2}/\Gamma(3/2)$. Noting that $\Gamma(1/2) = \sqrt{\pi}$ and $\Gamma(n+1) = n\Gamma(n)$, we find that $\Gamma(3/2) = \frac{1}{2}\Gamma(1/2) = \sqrt{\pi}/2$. Hence,

$$S_3 = \frac{2\pi^{3/2}}{\sqrt{\pi}/2} = 4\pi \qquad \Longrightarrow \qquad C_3 = \frac{4\pi}{3}$$

These also agree with expectation. When $n = 6$,

$$S_6 = \frac{2\pi^3}{\Gamma(3)} = \frac{2\pi^3}{2!} = \pi^3 \qquad \Longrightarrow \qquad C_6 = \frac{\pi^3}{6} \approx 5.1677$$

(d) If we throw $N$ darts into the six-dimensional cube that runs between $-1$ to 1 in each dimension, we expect an average of $n_{\text{success}} = NC_6/2^6$ of them to land within unit distance of the origin, since the cube has volume $2^6$. That is, $p = C_6/64$, $q = 1 - p$, and $\sigma = \sqrt{Npq}$, where $\sigma$ is the standard deviation in the number of successes. From this exercise, we would estimate

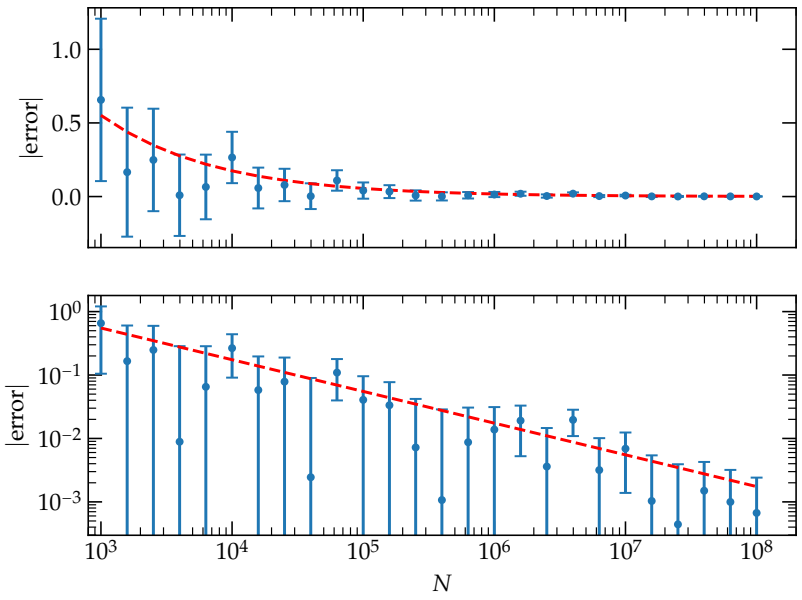
$$n_{\text{success}} = Np \pm \sqrt{Npq} \qquad \Longrightarrow \qquad p \approx \frac{n_{\text{success}}}{N} \pm \sqrt{\frac{pq}{N}}$$

That is, we expect the uncertainty in our estimate of $C_6$ to be

$$C_6 \approx 64p \pm \underbrace{64\sqrt{\frac{pq}{N}}}_{\delta C_6}$$

This uncertainty, $\delta C_6$, is shown in the dashed red line in the figure below. The discrete points were calculated with the following Python code:

```
def estimate_C6(n:int):
    r = np.linalg.norm(rng.uniform(-1,1,size=(n,6)), axis=1)
    rin = r[r <= 1]
    return 64 * len(rin) / n

trueval = np.pi**3 / 6
nv = np.logspace(3, 8, 26)
est = [estimate_C6(int(x)) for x in nv]
err = np.array(est) - trueval
fig, ax = plt.subplots()
p = np.pi**3 / (6 * 64)
q = 1-p
 = np.sqrt(p * q / nv) * 64
ax.plot(nv, np.abs(err), 'o')
ax.plot(nv, , 'r--')
ax.set_xscale('log')
ax.set_yscale('log')
ax.set_xlabel("$N$")
ax.set_ylabel(r"$|\text{error}|$");
```




**Problem 2 – Blackbody Radiation** An oven with thick, opaque walls is heated to a uniform temperature $T$. Through a tiny hole in one wall emerges light of a broad range of angular frequencies $\omega$, which is called **cavity radiation** or more commonly **blackbody radiation**. Its spectrum is given by

$$P(\omega)\,\mathrm{d}\omega = \frac{\hbar}{4\pi^2 c^2}\frac{\omega^3}{e^{\hbar\omega/k_\mathrm{B}T}-1}\,\mathrm{d}\omega$$

where $P(\omega)$ is the power per unit area per unit angular frequency. Planck used a computational trick to obtain this result in 1900, but Einstein took it seriously and used the idea that light energy came in discrete packets (which we now call photons) to explain the **photoelectric effect**. This work earned him the Nobel Prize.

(a) The spectrum above is reported as a function of angular frequency, but experiments typically measure wavelengths, not frequencies. Of course, $c = \lambda\nu = \frac{\lambda\omega}{2\pi}$. Show that the spectral distribution as a function of wavelength is

$$\mathscr{P}(\lambda) = \frac{2\pi hc^2}{\lambda^5}\frac{1}{\exp\left(hc/\lambda k_\mathrm{B}T\right)-1}$$

and confirm that it has the appropriate dimensions.

(b) Show that if one is only able to measure the long-wavelength portion of the spectrum coming from a distant star, it is impossible to determine the star's surface temperature. *Think very carefully; this is physics, not math!*

(a) We are looking for the function $\mathscr{P}(\lambda)$ so that

$$\mathscr{P}(\lambda)\,\mathrm{d}\lambda = P(\omega)\,\mathrm{d}\omega \qquad \Longrightarrow \qquad \mathscr{P}(\lambda) = P(\omega)\left|\frac{\mathrm{d}\omega}{\mathrm{d}\lambda}\right|$$

Since $\omega = 2\pi c/\lambda$, $\frac{\mathrm{d}\omega}{\mathrm{d}\lambda} = -2\pi c/\lambda^2$. Therefore,

$$\mathscr{P}(\lambda) = \frac{\hbar}{4\pi^2 c^2}\frac{(2\pi c/\lambda)^3}{e^{hc/\lambda k_\mathrm{B}T}-1}\frac{2\pi c}{\lambda^2} = \frac{2\pi hc^2}{\lambda^5}\frac{1}{e^{hc/\lambda k_\mathrm{B}T}-1}$$

(b) In the long-wavelength region of the spectrum, the argument of the exponential gets much less than 1, at which point we can use $e^x \approx 1 + x$ to get

$$\mathscr{P}(\lambda) \approx \frac{2\pi hc^2}{\lambda^5} \frac{1}{hc/\lambda k_B T} = \frac{2\pi c k_B T}{\lambda^4}$$

This function describes the power per unit area per unit wavelength that is emitted from the surface of the star, whose area and distance from us we don't necessarily know. We can therefore only compare relative amounts of light at each wavelength. Since each of these depends linearly on the temperature, there is no way to use them to infer the temperature.

**Problem 3 – Two-Dimensional Random Walk** A random walker starts at the origin and takes steps of length 1 in a random direction in the $xy$ plane. Simulate the progress of such a walker and compare your results to the expectation that the walker's typical distance from the origin after $n$ steps is $\sqrt{n}$. Use your judgment on how best to make this comparison, but please explain your choices.

Consider the walker's first step. Without fail, she ends up 1 step-length $\delta$ from the origin. Let's analyze the situation after $j$ steps. At that point, the walker is a distance $s_j$ from the origin. How far will she be after the next step? By the law of cosines,

$$s_{j+1}^2 = s_j^2 + \delta^2 - 2 s_j \delta \cos\phi$$

Taking the average over all angles $\phi$ gives

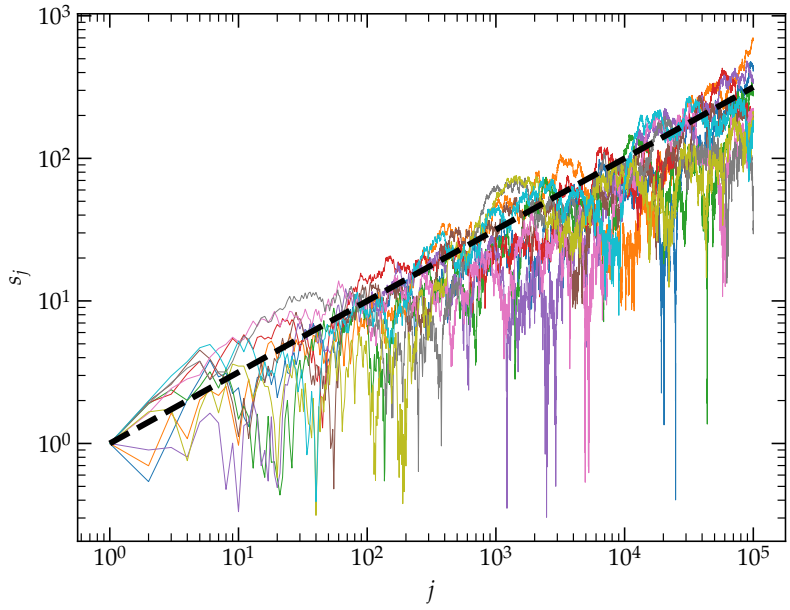
$$\left\langle s_{j+1}^2 \right\rangle = \left\langle s_j^2 \right\rangle + \delta^2 - \frac{2\delta}{2\pi} \int_0^{2\pi} \cos\phi \, \mathrm{d}\phi$$

But the integral gives zero, so

$$\left\langle s_{j+1}^2 \right\rangle = \left\langle s_j^2 \right\rangle + \delta^2 = (j+1)\delta^2 \tag{1}$$

After $n$ steps the root-mean-square distance from the origin is thus $\sqrt{n}\delta$, where $\delta$ is the length of a single step. Let's write a little Python program to check this out.

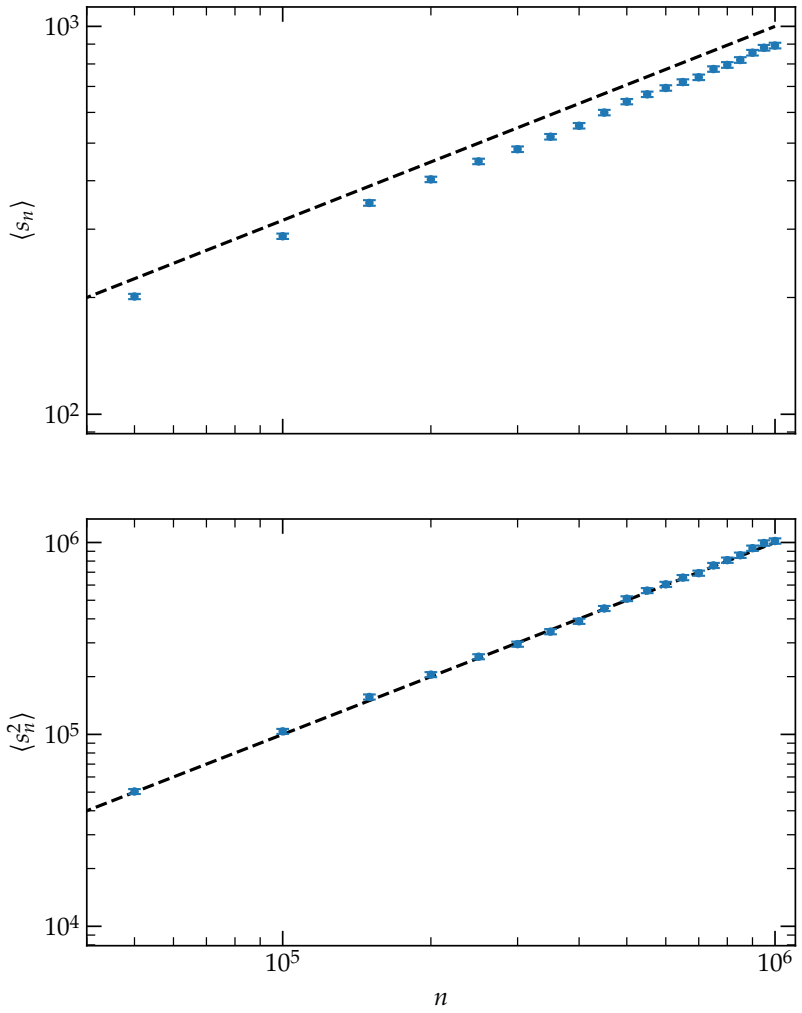
```
def walk(N:int):
    phi = rng.uniform(-np.pi, np.pi, size=N)
    z = np.exp(phi * 1j)
    pos = np.cumsum(z)
    r = np.abs(pos)
    return r

fig, ax = plt.subplots() # ncols=2, figsize=(10,4))
nsteps = 100000
n = np.arange(1, nsteps)
for _ in range(10):
    v = walk(nsteps-1)
    ax.plot(n, v, lw=0.5)
ax.plot(n, np.sqrt(n), 'k--', lw=3)
ax.set_xscale('log')
ax.set_yscale('log')
ax.set_xlabel('$j$')
ax.set_ylabel('$s_j$')
```




Distance from the origin, $s_j$, as a function of step number $j$, compared to the "expected value" $s_j = \sqrt{j}$ shown in the black dashed line. The 10 colored curves do indeed seem to dance around the expectation.

If you are unpersuaded, and concerned that the curves preferentially lie below expectations, let's calculate in a different way. Let's do a bunch of random walks of $N = 1$ million steps, and every interval of 50000 steps, store the distance from the origin. We can then average over the independent runs at each interval and compute standard errors and plot with error bars.

The following shows two different ways of analyzing the results. In the top plot, I compute statistics on the distance from the origin as a function of the number of steps, along with the naive expectation that $\langle s_n \rangle = \sqrt{n}$. However, we have shown that $\langle s_n^2 \rangle = n$, which is not the same thing! The lower plot shows the stats on the square of the distance from the origin, and you can see that it agrees quite well with expectations.




(upper panel) Distance from the origin, $s_n$, as a function of step number $n$, compared to the naive expectation $s_n = \sqrt{n}$ shown in the black dashed line. (lower panel) Mean square of the distance from the origin as a function of the number of steps $n$, which does agree with the expected value $n$ shown in the dashed black line.

```
N = 1000000
loops = 1000
interval = 50000
x = np.arange(interval-1, N, interval)
r = np.zeros((loops, len(x)))
for l in tqdm(range(loops)):
    phi = rng.uniform(-np.pi, np.pi, size=N)
    z = np.exp(phi * 1j)
    pos = np.cumsum(z)
    r[l,:] = np.abs(pos[x])N = 1000000
loops = 1000
interval = 50000
x = np.arange(interval-1, N, interval)
r = np.zeros((loops, len(x)))
for l in tqdm(range(loops)):
    phi = rng.uniform(-np.pi, np.pi, size=N)
    z = np.exp(phi * 1j)
    pos = np.cumsum(z)
    r[l,:] = np.abs(pos[x])
means = np.mean(r, axis=0)
varis = np.var(r, axis=0)
sems = np.std(r, axis=0) / np.sqrt(loops)
r2 = r**2
means2 = np.mean(r2, axis=0)
varis2 = np.var(r2, axis=0)
sems2 = np.std(r2, axis=0) / np.sqrt(loops)
fig, axs = plt.subplots(nrows=2, sharex=True)
ax = axs[0]
ax.plot([1e4, 1e6], [1e2, 1e3], 'k--')
ax.errorbar(x+1, means, yerr=sems, lw=0, fmt='.', capsize=3)
ax.set_xscale('log')
ax.set_yscale('log')
ax.set_ylabel(r"$\langle s_n \rangle$")
ax = axs[1]
ax.plot([1e4, 1e6], [1e4, 1e6], 'k--')
ax.errorbar(x+1, means2, yerr=sems2, lw=0, fmt='.', capsize=3)
ax.set_xscale('log')
ax.set_yscale('log')
ax.set_xlabel(r"$n$")
ax.set_ylabel(r"$\langle s_n^2 \rangle$")
ax.set_xlim(0.8*interval, 1.1*N)
fig.savefig('2d-random-walk2.pdf')
```