DRAFT 13.1 - Badger's Law; The Golden Spiral

Generated 2025-06-08 - Plain ASCII, ready for replication

1. Purpose

Enable any reviewer to reproduce the Spiral Penalty experiments, from concept through running code.

2. Official working name

Badger's Law; The Golden Spiral (nickname: The Golden Spiral) - keep this exact naming for citation.

3. Concept recap

- Add a small spiral tension term to classical gravity.
- Lambda controls strength (0 = Newton, larger = spiral).
- k sets pitch of logarithmic spiral.
- Goal: test stability, explore artistic and toy-gravity uses.

4. Metaphor origins

Ball point pen micro helix
Breath vortex rings
GPS satellite ground track spiral

5. Core mathematics

Spiral Penalty Lagrangian:
L = 0.5 m (r_dot^2 + r^2 theta_dot^2) - G M m / r + 0.5 lambda m (r_dot - k r theta_dot)^2
Triangle-perimeter lemma (legacy):
V(t) = a sum|e^(i w_j t) - e^(i w_k t)| <= 3 sqrt(3) a for n=3 bodies.

6. Colab Cell A - Single Body Spiral Orbit

Copy into Colab:

```
import numpy as np, matplotlib.pyplot as plt
G,M,k,dt,steps = 1.0,1.0,0.15,0.00025,40000
def step(x,y,vx,vy,lmb):
    r3=(x*x+y*y)**1.5
    ax=-G*M*x/r3 + lmb*(-k*vy - vx)
    ay=-G*M*y/r3 + lmb*( k*vx - vy)
    vx+=0.5*dt*ax; vy+=0.5*dt*ay
    x+=dt*vx; y+=dt*vy
    r3=(x*x+y*y)**1.5
    ax=-G*M*x/r3 + lmb*(-k*vy - vx)
    ay=-G*M*y/r3 + lmb*( k*vx - vy)
    vx+=0.5*dt*ax; vy+=0.5*dt*ay
    return x,y,vx,vy
def run(lmb):
    x,y = 1.0,0.0; vx,vy = 0.0,1.0; xs,ys=[],[]
    for _ in range(steps):
        xs.append(x); ys.append(y)
        x,y,vx,vy = step(x,y,vx,vy,lmb)
    plt.figure(figsize=(4,4)); plt.plot(xs,ys,'.',ms=1)
    plt.gca().set_aspect('equal'); plt.title('lambda='+str(lmb)); plt.show()
run(0.05)
```

7. Colab Cell B - Radius Drift Scan

Paste next to obtain radius drift curve:

```
steps_per_orbit = int(1/dt); orbits=10; steps=steps_per_orbit*orbits
def final_r(lmb):
    x,y = 1.0,0.0; vx,vy = 0.0,1.0
    for _ in range(steps):
        x,y,vx,vy = step(x,y,vx,vy,lmb)
    return (x*x+y*y)**0.5
lam_vals = np.linspace(0.0,0.20,11)
r_end = [final_r(l) for l in lam_vals]
plt.plot(lam_vals,r_end,'o-'); plt.xlabel('lambda'); plt.ylabel('radius after 10 orbits'); plt.show()
```

End of document - all ASCII

8. Expected results