On the Monte Carlo simulation of anisotropic Newtonian gravitation

S. Halayka*

Thursday 14^{th} August, 2025 17:44

Abstract

This paper contains a short introduction to anisotropic Newtonian gravitation.

1 Introduction

First see [1] for a short tutorial for C++ programmers on isotropic Newtonian gravitation. In [1] we build an isotropic gravitational field through the use of pseudorandomly generated field lines.

Here we introduce a method for generating anisotropic gravitational fields through the use of spherical linear interpolation.

Anisotropic gravitational fields provide extra acceleration, which is useful for modeling the flat rotation curves in galactic dynamics.

2 Brute force: field line intersection density gradient

Regarding the holographic principle [2, 3], the number of gravitational field lines for a black hole is related to the event horizon area A:

$$n = \frac{Akc^3}{4G\hbar \log 2}. \tag{1}$$

For a Schwarzschild black hole in particular [4], the event horizon radius r_s is

$$r_s = \sqrt{\frac{A}{4\pi}} = \sqrt{\frac{nG\hbar \log 2}{kc^3\pi}}, \tag{2}$$

and the mass of the Schwarzschild black hole is

$$M = \frac{c^2 r_s}{2G} = \sqrt{\frac{nc\hbar \log 2}{4Gk\pi}}. \tag{3}$$

Where the β function is the integer field line collision count, R is the distance from the black hole centre, and ϵ is some small number:

$$\alpha = \frac{\beta(R+\epsilon) - \beta(R)}{\epsilon}. \tag{4}$$

*sjhalayka@gmail.com

The gradient strength, where r is the receiver radius, is:

$$g = \frac{-\alpha}{r^2}. \tag{5}$$

Below is a bunch of code.

The full code can be found at

`https://github.com/sjhalayka/numerical_newtonian_gravity`

```
vector_3 random_unit_vector(void)
{
	const real_type z = dis(generator) * 2.0 - 1.0;
	const real_type a = dis(generator) * 2.0 * pi;

	const real_type r = sqrt(1.0f - z * z);
	const real_type x = r * cos(a);
	const real_type y = r * sin(a);

	return vector_3(x, y, z).normalize();
}

vector_3 slerp(vector_3 s0, vector_3 s1, const real_type t)
{
	vector_3 s0_norm = s0;
	s0_norm.normalize();

	vector_3 s1_norm = s1;
	s1_norm.normalize();

	const real_type cos_angle = s0_norm.dot(s1_norm);
	const real_type angle = acos(cos_angle);

	const real_type p0_factor = sin((1 - t) * angle) / sin(angle);
	const real_type p1_factor = sin(t * angle) / sin(angle);

	return s0 * p0_factor + s1 * p1_factor;
}

bool circle_intersect(
	const vector_3 normal,
	const real_type circle_location,
	const real_type circle_radius)
{
	vector_3 outline_dir(
		circle_location,
		circle_radius,
		0);

	outline_dir.normalize();

	static const vector_3 v(1, 0, 0);

	const real_type d = outline_dir.dot(v);

	if (d <= 0)
		return false;
```

```
        const real_type d_ = normal.dot(v);

        if (d_ <= d)
                return false;

        return true;
}
```

```
// beta function
long long signed int get_intersecting_line_count_integer(
        const long long signed int n,
        const vector_3 sphere_location,
        const real_type sphere_radius,
        const real_type D)
{
        const real_type disk_like = 3.0 - D;

        long long signed int count = 0;

        generator.seed(static_cast<unsigned>(0));

        for (long long signed int j = 0; j < n; j++)
        {
                const vector_3 p = random_unit_vector();

                vector_3 p_disk = p;
                p_disk.y = 0;
                p_disk.normalize();

                const vector_3 normal = slerp(p, p_disk, disk_like);

                if (circle_intersect(normal, sphere_location.x, sphere_radius))
                        count++;
        }

        return count;
}
```

The Newtonian gravitational variables are acceleration:

$$a_N = \frac{GM}{R^2} = \sqrt{\frac{nGc\hbar \log 2}{4k\pi R^4}}, \tag{6}$$

circular orbit speed:

$$v_N = \sqrt{a_N R}, \tag{7}$$

and gradient strength:

$$g_N = \frac{a_N k 2\pi M}{Rc\hbar \log 2}. \tag{8}$$

The flat rotation curve variables are:

$$v_{\text{flat}} = x v_N, \tag{9}$$

where $x = 2$ for example.

$$a_{\text{flat}} = \frac{v_{\text{flat}}^2}{R} = \frac{gRc\hbar \log 2}{k2\pi M}. \tag{10}$$

Since

$$a_{\text{flat}} \propto g \tag{11}$$

and

$$a_{\text{ratio}} = \frac{a_{\text{flat}}}{a_N}, \tag{12}$$

$$g_{\text{ratio}} = \frac{g}{g_N}, \tag{13}$$

to then find D, look for where $g_{\text{ratio}} \geq a_{\text{ratio}}$, starting from $D = 3$, marching toward $D = 2$.

References

[1] Halayka. Newtonian gravitation from scratch, for C++ programmers. (2024)

[2] ‘t Hooft. Dimensional reduction in quantum gravity. (1993)

[3] Susskind. The World as a Hologram. (1994)

[4] Misner et al. Gravitation. (1970)

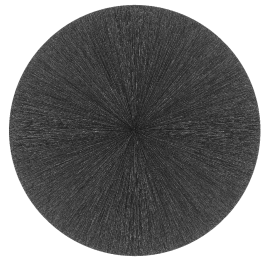

Figure 1: Where $D = 3$, as viewed from the side. The field lines are isotropic, spherical.

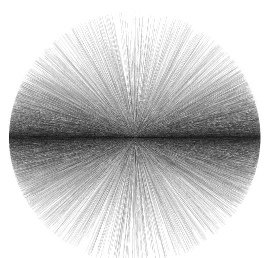

Figure 2: Where $D = 2.1$, as viewed from the side. The field lines are increasingly anisotropic.

Figure 3: Where $D = 2.001$, as viewed from the side. The field lines are anisotropic, disk-like.

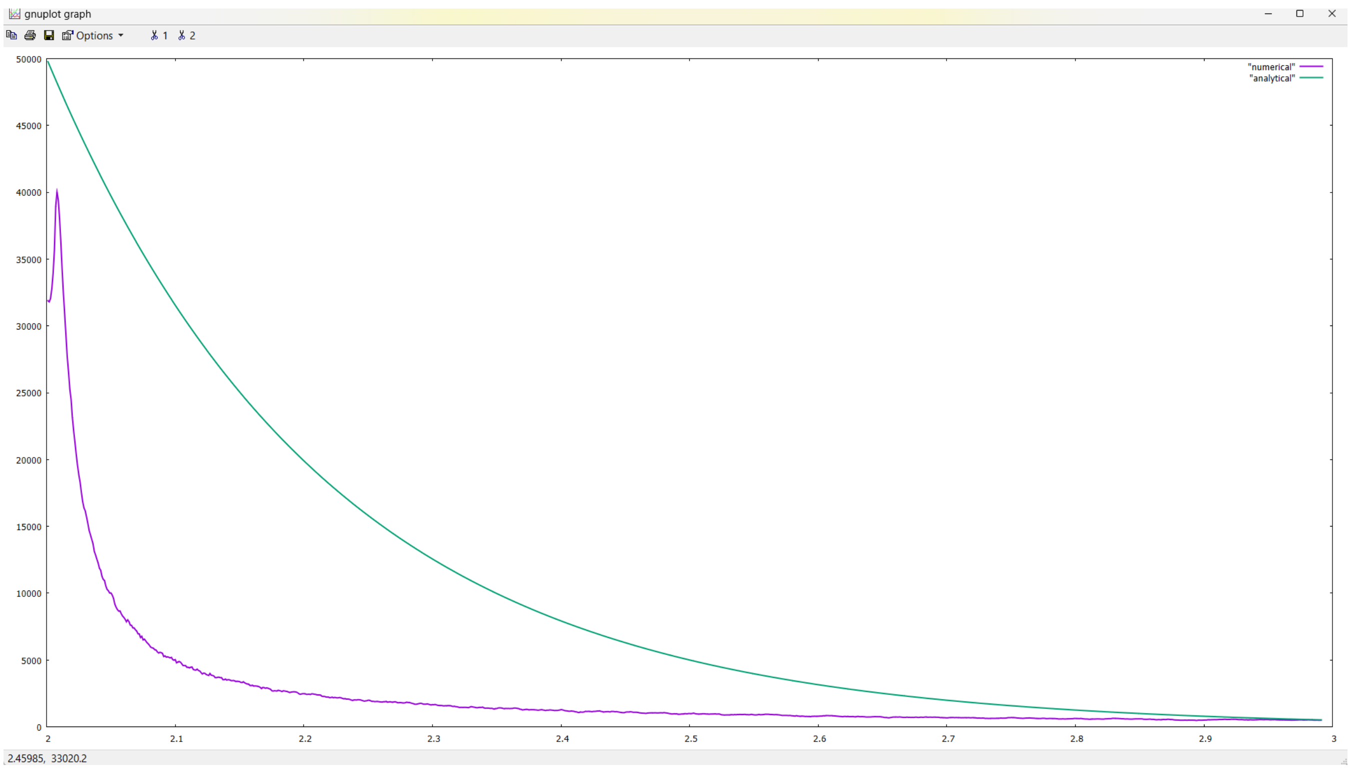

Figure 4: $R = 100$, $r = 1$, $n = 10^8$, $\epsilon = 1$. The analytical plot is generated by the formula $y = n/(2R^D)$.

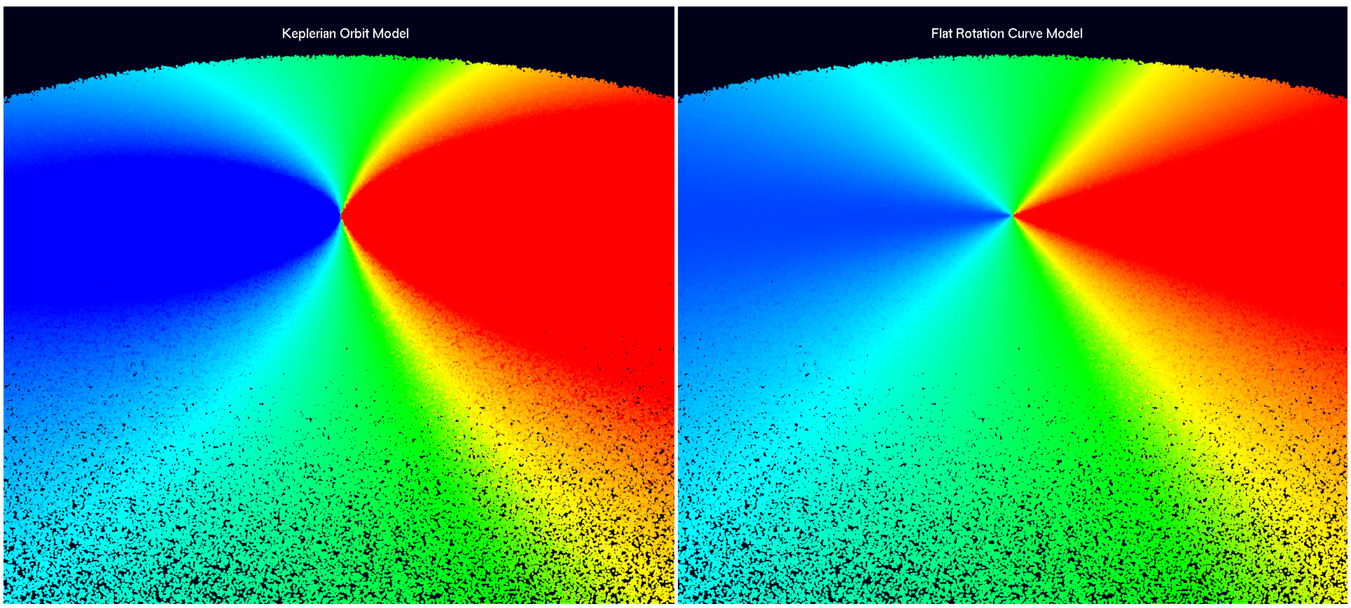

Figure 5: Visualization of the relativistic Doppler effect for 10^6 stars. The Newtonian orbit is on the left, and the flat rotation curve orbit is on the right. The redshift of the wavelength indicates stars moving away from the camera, and blueshift of the wavelength indicates stars moving toward the camera. Thus, the stars in the galaxy are orbiting counterclockwise. On the left, the wavelength is dependent on angle and distance from the galactic centre. On the right, the wavelength is dependent only on angle, which means that there is a constant orbit speed that is independent of the distance from the galactic centre. This is exactly what Vera Rubin discovered in the disks of the galaxies that she observed, and so dark matter was posited.