Eric Du — Physics 105

HW 06 — Analytic Mechanics — March 10, 2023

Collaborators

I worked with **Andrew Binder, Adarsh Iyer** and **Aren Martinian** to complete this homework.

Problem 1

A bucket of water is set spinning about its symmetry axis with angular speed Ω. Determine the shape of the water in the bucket.

Solution: Consider a small portion of mass m. The centrifugal force acting on it $F_{cf} = m\Omega^2\rho\hat{\rho}$, and the gravitational force is $F_g = -mg\hat{z}$. Therefore, using $\nabla U = -F$, we know that:

$$U = mgz - \frac{1}{2}m\Omega^2\rho^2$$

We now use the fact that the surface of the water is an equipotential, so we require U to be constant. Deriving a relation for z in terms of ρ, we get:

$$\begin{aligned} mgz &= U + \frac{m\Omega^2 r^2}{2} \\ z(r) &= \frac{2U + m\Omega^2\rho^2}{2mg} \end{aligned}$$

which means that $z \propto \rho^2$ implying that the shape of the water surface is that of a parabola. $\square$

Problem 2

A car is driving with acceleration a and instantaneous velocity v. The tires of radius r are not slipping. What point on the tire has the greatest acceleration relative to the ground? What is this acceleration?

Solution: Consider the following diagram:

[insert tikz here]

Here, we have a friction term because the wheel is rolling without slipping. In order to maximize the acceleration relative to the ground, we require that the acceleration vectors add constructively, and the location at which this occurs is shown in the diagram above. Now, consider the acceleration in the y direction:

$$a_y = F_{cf}\cos\theta - f sin\theta$$

$a_y = 0$ since the wheel isn't accelerating either up or down, therefore we have the equation:

$$F_{cf}\cos\theta = f\sin\theta$$

Solving for θ, we get:

$$\begin{aligned} tan\theta &= \frac{F_{cf}}{f} = \frac{m\Omega^2}{mrA} = \frac{\Omega^2}{A} = \frac{v^2}{rA} \\ \therefore \theta &= \tan^{-1}\left(\frac{v^2}{rA}\right) \end{aligned}$$

Here, I use A to reference the linear acceleration of the car. This acceleration exists only in the x direction now:

$$\begin{aligned} a_x &= F_{cf}\sin\theta + f\cos\theta + a \\ &= m\Omega^2 r\sin\left(\tan^{-1}\left(\frac{v^2}{rA}\right)\right) + f\cos\left(\tan^{-1}\left(\frac{v^2}{rA}\right)\right) + a \end{aligned}$$

which is also the total magnitude of the acceleration. □

Problem 3

How much greater is the acceleration due to Earth's gravity g at the equator than at the pole (assume a spherical Earth)? If the actual measured difference is $\Delta g = 52$ mm/s^2, explain this difference. How might you calculate this difference between the measured result and your calculation?

Solution: The net force on an object on Earth is $F = F_g + F_{cf} + F_{cor}$. Since we are measuring the acceleration relative to ourselves, $F_{cor} = 0$. Furthermore, an observer at the north pole experiences no Coriolis force, so therefore the gravity it experiences is simply F_g. Therefore, we have the equation:

$$m\ddot{r} = mg - m\Omega^2\rho$$
$$\therefore \ddot{r} = 9.8\text{m/s}^2 - \Omega^2 r_e$$

Crunching the numbers, we get that $\Omega^2 r_e = 0.034$ m/s^2, or 34 mm/s^2. To explain the difference between the measured result and this calculation, we have to remember that the Earth is not exactly a sphere, and its radius is larger at the equator than at the pole, so we'd expect a larger discrepancy. To calculate this difference, it suffices to look at the difference between the magnitude of the centrifugal force on a spherical Earth as opposed to an ellipsoidal one, since the mass in both scenarios is the same. □

Problem 4

Supposedly, during World War I, there was an event during which a British warship near the Falkland Islands repeatedly missed its target due to not accounting for the Coriolis force. If it fires a projectile due South at latitude 50°S at 33° elevation with a speed of 800 m/s, by how much do the shells miss their target and in what direction? Ignore air resistance.

Solution: We set up our coordinate axes as follows:

[insert tikz here]

The equations of motion for a particle which subject to the Coriolis force is:

$$\begin{aligned}
\ddot{x} &= 2\Omega(\dot{y}\cos\theta - \dot{z}\sin\theta) \\
\ddot{y} &= -2\Omega\dot{x}\cos\theta \\
\ddot{z} &= -g + 2\Omega x\sin\theta
\end{aligned}$$

These are the general equations of motion, without any approximations. From the problem statement, we know that:

$$\begin{aligned}
\dot{y} &= -800\cos(33^\circ) & \qquad \dot{z} &= 800\sin(33^\circ) - gt \\
\ddot{y} &= 0 & \qquad \ddot{z} &= -g
\end{aligned}$$

So substituting this into our equation for $\ddot{x}$, we get:

$$\ddot{x} = 2\Omega(-800\cos(33^\circ)\cos(140^\circ) - (800\sin(33^\circ) - gt)\sin(140^\circ)$$

Integrating twice, we get the equation (and also evaluating the known terms):

$$x(t) = \frac{467.82\Omega}{2}t^2 + \frac{12.6\Omega}{6}t^3$$

All that remains now is to find the time of flight t. To do this, we use our equations of projectile motion. Since the Coriolis force acts perpendicularly to the velocity of the particle at all times, then its presence does not affect the projectile's time of flight. Therefore, using standard equations for projectile motion, we know that:

$$T = \frac{2v_0\sin\theta}{g} = 88.92 \text{ seconds}$$

Plugging this time back into $x(t)$, gives:

$$x(t = 88.92) = 241.79 \text{ m}$$

which is the distance travelled east due to the Coriolis force. □

Problem 5

Show that the equation motion of a particle of mass m with spatial coordinate $\mathbf{r}$ and velocity $\mathbf{v}$ in a noninertial reference frame can be written in the form of the Euler-Lagrange equation with a potential given by

$$V = -m\boldsymbol{\Omega} \times \mathbf{r} \cdot \mathbf{v} + m\mathbf{A} \cdot \mathbf{r} - \frac{m}{2}(\boldsymbol{\Omega} \times \mathbf{r})^2$$

where $\mathbf{A}$ is the frame's linear acceleration and $\boldsymbol{\Omega}$ is its angular velocity vector.

Solution: We write out our Lagrangian:

$$\mathcal{L} = T - U = \frac{1}{2}m\mathbf{r}^2 + m(\boldsymbol{\Omega} \times \mathbf{r}) \cdot \mathbf{r} - m\mathbf{A} \cdot \mathbf{r} + \frac{m}{2}(\boldsymbol{\Omega} \times \mathbf{r})^2$$

Taking the derivative with respect to $\mathbf{r}$, we get:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{r}} = m(\dot{\mathbf{r}} \times \boldsymbol{\Omega}) - m\mathbf{A} + m\boldsymbol{\Omega}^2\mathbf{r} - m(\boldsymbol{\Omega} \cdot \mathbf{r})\boldsymbol{\Omega}$$

Now with respect to $\dot{\mathbf{r}}$:

$$\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{r}}} = m\dot{\mathbf{r}} + m(\boldsymbol{\Omega} \times \mathbf{r})$$

so:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{r}}}\right) = m\ddot{\mathbf{r}} + m(\boldsymbol{\Omega} \times \mathbf{r})$$

Putting these together, we get:

$$m\ddot{\mathbf{r}} + m(\boldsymbol{\Omega} \times \mathbf{r}) = m(\mathbf{r} \times \boldsymbol{\Omega}) - m\mathbf{A} + m\,\underbrace{\boldsymbol{\Omega}^2\mathbf{r} - m(\boldsymbol{\Omega} \cdot \mathbf{r})\boldsymbol{\Omega}}_{=m(\boldsymbol{\Omega}\times\mathbf{r})\times\boldsymbol{\Omega}}$$

$$\therefore m\ddot{\mathbf{r}} = 2m((r \times \Omega) - m\mathbf{A} + m(\boldsymbol{\Omega} \times \mathbf{r}) \times \boldsymbol{\Omega}$$

as desired. □

Problem Set 1 problems

Question 1: Fourier Series

Learning objectives

In this question you will:

- understand how a function can be expanded in a series
- see how the accuracy of an approximation improves with the number of terms

The set of square-integrable functions ($\int |f(x)|^2 dx < \infty$) on the interval $[0, \lambda]$ make up a *Hilbert space*, which is a vector space with an inner product that satisfies certain properties. In this case the space is known as $L^2([0, \lambda])$, with inner product $\langle f, g\rangle = \int_0^\lambda f(x)g(x)\,dx$.

With $k_n = n\frac{2\pi}{\lambda}$, the functions $c_n = \cos(k_n x)$ and $s_n = \sin(k_n x)$ form a **complete and orthogonal basis** of this Hilbert space, known as the *Fourier basis*. Orthogonality means

$$\langle c_m, c_n\rangle = C_n \delta_{m,n}, \tag{1}$$
$$\langle s_m, s_n\rangle = S_n \delta_{m,n}, \tag{2}$$
$$\langle c_n, s_m\rangle = 0, \tag{3}$$

and completeness means

$$\forall f \in L^2([0,\lambda]), \quad f = \sum_{n=0}^{\infty} \left(\frac{\langle f, c_n\rangle}{C_n} c_n + \frac{\langle f, s_n\rangle}{S_n} s_n \right).$$

This means that any square-integrable function $f(x)$ on a finite domain can be written as a sum of sines and cosines,

$$f(x) = \sum_{n=0}^{\infty} \big(A_n \cos(k_n x) + B_n \sin(k_n x) \big).$$

Furthermore, we are guaranteed that at all points x where f is continuous, the above sum converges (so the terms $A_n \cos(k_n x) + B_n \sin(k_n x)$ decrease sufficiently quickly). This can be extended to periodic functions in the obvious manner.

(We have formulated things in terms of x, suggesting a "spatial" variable, but of course this holds for functions of any variable. When the variable is "time", the symbols used are conventionally t, ω, and T instead of x, k, and λ. Also, $s_0 = 0$ and is implicitly excluded from the above sums.)

1a.

Show that the Fourier basis is orthogonal, and find the normalisations C_n and S_n.

Hint: $\cos(a+b) = \cos a \cos b - \sin a \sin b$.

First we can find the coefficients S_n and C_n:

$$S_n = \langle s_n, s_n \rangle = \int_0^\lambda \sin^2\left(\frac{2\pi n x}{\lambda}\right) dx$$

Recall the identity: $\sin^2(x) = \frac{1-\cos(2x)}{2}$, so therefore:

$$\begin{aligned} S_n &= \frac{1}{2}\int_0^\lambda 1 - \cos\left(\frac{4\pi n x}{\lambda}\right) dx \\ &= \frac{\lambda}{2} - \left[\sin\left(\frac{4\pi n x}{\lambda}\right)\right]_0^\lambda \\ &= \frac{\lambda}{2} \end{aligned}$$

Similarly for C_n:

$$\begin{aligned} C_n &= \int_0^\lambda \cos^2\left(\frac{2\pi n x}{\lambda}\right) dx \\ &= \frac{1}{2}\int_0^\lambda 1 + \cos\left(\frac{4\pi n x}{\lambda}\right) dx \\ &= \frac{\lambda}{2} - \left[\sin\left(\frac{4\pi n x}{\lambda}\right)\right]_0^\lambda \\ &= \frac{\lambda}{2} \end{aligned}$$

For the orthogonality condition, we show that the integral:

$$\langle s_m, s_n \rangle = \int_0^\lambda \sin\left(\frac{2\pi n x}{\lambda}\right) \sin\left(\frac{2\pi m x}{\lambda}\right) dx$$

must equal zero. To do so, notice that both sine terms are actually odd over the interval $x \in [0\lambda]$, so therefore this integral must evaluate to zero, or in other words $\langle s_m, s_n \rangle = 0$ when $n \neq m$. Obviously, if $n = m$, then the integrand becomes a $\sin^2(x)$ term, which is no longer an odd function, leading to a nonzero inner product. Furthermore, if we expand this out, we get:

$$\begin{aligned}\int_0^\lambda \sin\left(\frac{2\pi nx}{\lambda}\right)\sin\left(\frac{2\pi mx}{\lambda}\right)dx &= \int_0^\lambda \cos\left(\frac{2\pi nx}{\lambda}\right)\cos\left(\frac{2\pi mx}{\lambda}\right)dx - \int_0^\lambda \cos\Bigg(\frac{2}{} \\ &= \langle c_m, c_n\rangle - \frac{\lambda}{2\pi(m+n)}\sin\left(\frac{2\pi(m+n)x}{\lambda}\right)\Bigg|_0^\lambda \\ &= \langle c_m, c_n\rangle\end{aligned}$$

And since $\langle s_m, s_n\rangle = 0$ when $n \neq m$, then this necessarily implies that $\langle c_m, c_n\rangle$ follow the same property. Therefore, as a concluding statement, we write:

$$\langle s_m, s_n\rangle = S_n\delta_{m,n}$$
$$\langle c_n, c_n\rangle = C_n\delta_{m,n}$$

where $\delta_{m,n}$ represents the Kronecker delta, where $\delta_{m,n} = 1$ when $m = n$ and 0 otherwise.

The last orthogonality condition asks us to show that $\langle c_m, s_n\rangle = 0$ for all m, n. As an integral:

$$\langle c_m, s_n\rangle = \int_0^\lambda \cos\left(\frac{2\pi mx}{\lambda}\right)\sin\left(\frac{2\pi nx}{\lambda}\right)dx$$

Notice that the integrand is again odd over the interval $x \in [0, \lambda]$, and since they are different functions then $m = n$ is no longer a special case where we get an even function. Therefore, this integral evaluates to 0 regardless of the choice for m, n.

1b.

Find the Fourier coefficients A_n and B_n of a square function, which is a common periodic signal:

$$f(x) = \begin{cases} f_0 & \text{if } a \leq x \leq b \\ f_1 & \text{else} \end{cases},$$

for $0 \leq a \leq b \leq \lambda$.

Hint: adding a constant to a function only changes the A_0 term.

To compute the fourier transform, we compute:

$$A_n = \int_0^\lambda f(x)\sin(k_n x) = \int_0^a f_1\sin(k_n x) + \int_a^b f_0\sin(k_n x) + \int_b^\lambda f_1\sin(k_n x)$$
$$B_n = \int_0^\lambda f(x)\cos(k_n x) = \int_0^a f_1\cos(k_n x) + \int_a^b f_0\cos(k_n x) + \int_b^\lambda f_1\cos(k_n x)$$

Now, using the relation that $k_n = \frac{2\pi n}{\lambda}$, we can plug these two integrals into WolframAlpha, which gives:

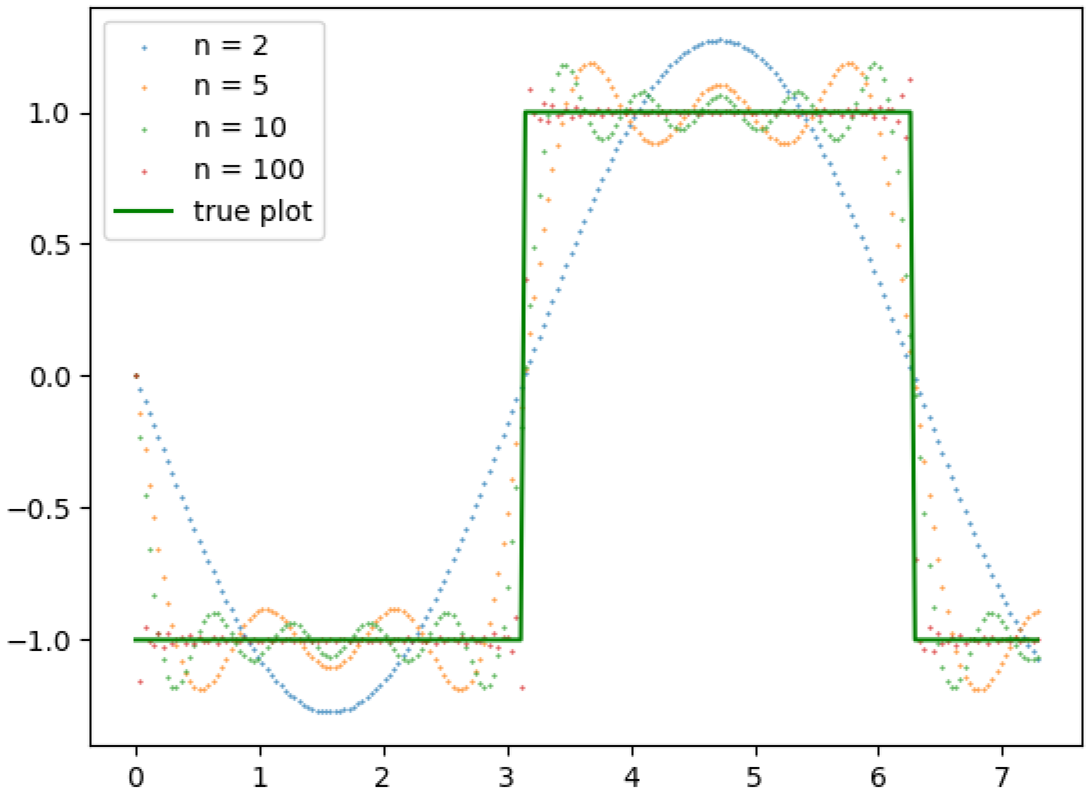
$$A_n = \frac{1}{n\pi}\left[f_1 \sin\left(\frac{2\,pina}{\lambda}\right) + f_0 \sin\left(\frac{2\pi nb}{\lambda}\right) - f_0 \sin\left(\frac{2\pi na}{\lambda}\right) - f_1 \sin\left(\frac{2\pi nb}{\lambda}\right)\right]$$
$$B_n = \frac{1}{n\pi}\left[f_1 \cos\left(\frac{2\pi na}{\lambda}\right) - f_0 \cos\left(\frac{2\pi nb}{\lambda}\right) + f_0 \cos\left(\frac{2\pi na}{\lambda}\right) + f_1 \cos\left(\frac{2\pi nb}{\lambda}\right)\right]$$

1c.

While we have exact equality in the $n \to \infty$ limit, with finite resources we may only compute a finite number of terms. Plot the true signal, on top of the approximations with 2, 5, 10, and 100 terms. Set $a = \frac{\lambda}{2}$, $b = \lambda$, $\lambda = 2\pi$, and $f_0 = -f_1 = 1$.

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [8]:
```
lamb = 2*np.pi
a = lamb/2
b = lamb
f0 = 1
f1 = -f0

def compute_square(k):
    #k denotes the number of fourier coefficents we're taking
    An = []
    Bn = []
    kn = [2 * n * np.pi/lamb for n in range(1, k+1)]
    for n in range(1, k+1):
        An.append(1/ (n * np.pi) * (f1*np.sin(kn[n-1]*a) + f0 *np.sin(kn[n-1
        Bn.append(1 /(n * np.pi) * (np.cos(kn[n-1]*a) - f0 * np.cos(kn[n-1]
    f_n = lambda x: np.sum([An[i] * np.cos(kn[i] * x) + Bn[i] * np.sin(kn[i]
    return f_n
square_2 = compute_square(2)
square_5 = compute_square(5)
square_10 = compute_square(10)
square_100 = compute_square(100)

x_values = np.linspace(0, lamb+1, 200)

def square(x):
    if a <= x <= b:
        return f0
    else:
        return f1

vfunc_sq = np.vectorize(square)
true_square = [vfunc_sq(x_values[i]) for i in range(len(x_values))]

plt.scatter(x_values, [square_2(x_values[i]) for i in range(len(x_values))],
plt.scatter(x_values, [square_5(x_values[i]) for i in range(len(x_values))],
plt.scatter(x_values, [square_10(x_values[i]) for i in range(len(x_values))]
plt.scatter(x_values, [square_100(x_values[i]) for i in range(len(x_values))
plt.plot(x_values, true_square, c = 'green')
plt.legend(["n = 2", "n = 5", 'n = 10', "n = 100", 'true plot'])
plt.show()
```



1d.

Now compute the Fourier series representation of the triangular function,

$$f(x) = \begin{cases} x & \text{if } 0 \le x \le \frac{\lambda}{4} \\ \frac{\lambda}{2} - x & \text{if } \frac{\lambda}{4} \le x \le \frac{3\lambda}{4} \\ x - \lambda & \text{if } \frac{3\lambda}{4} \le x \le \lambda \end{cases},$$

periodically continued.

Hint: Use integration by parts.

As plotted below, the triangular function is an odd function over the interval. Therefore, when we compute the integral:

$$\int_0^\lambda f(x)\sin(k_n x)dx$$

we are computing the integral of an odd function over a symmetric interval (symmetric about $\lambda/2$), so therefore all the sine terms vanish. Computing the cosine terms means evaluating:

$$B_n = \int_0^{\lambda/4} x\cos(k_n x)dx + \int_{\lambda/4}^{3\lambda/4} \left(\frac{\lambda}{2} - x\right)\cos(k_n x)dx + \int_{3\lambda/4}^{\lambda} (x-\lambda)\cos(k_n x)dx$$

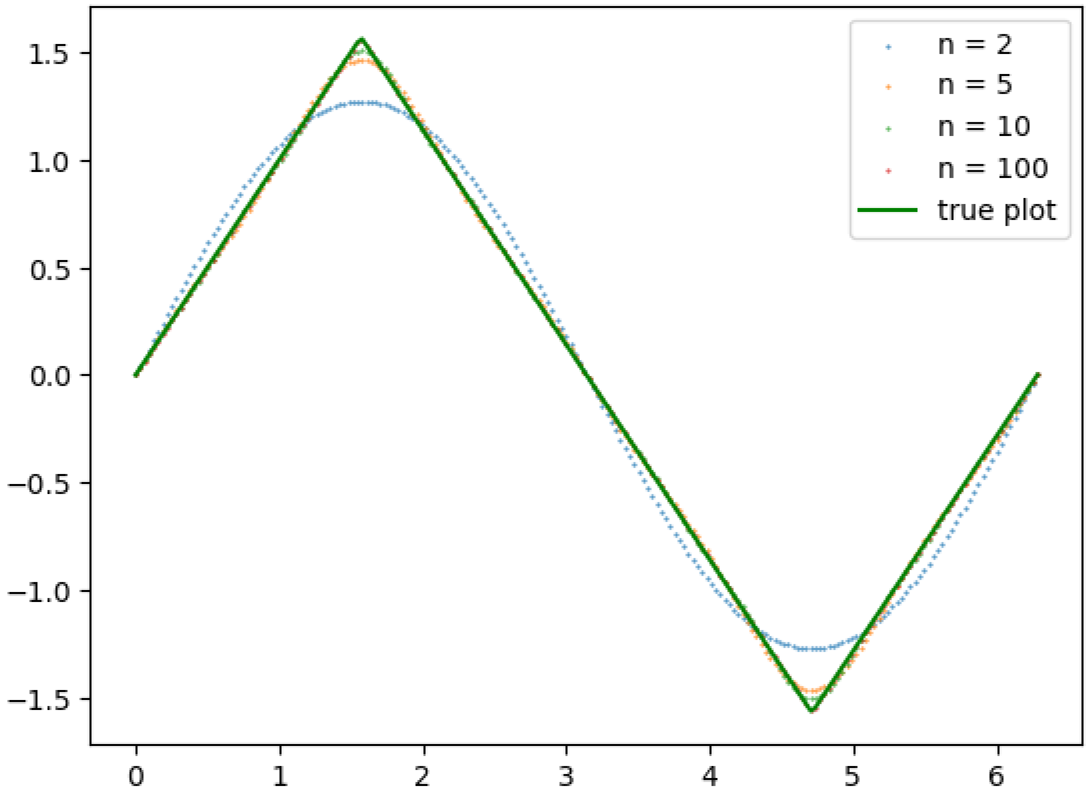
Plugging this into WolframAlpha gives:

$$B_n = \frac{\lambda^2 \cos(\pi n)\left(\pi n \cos\left(\frac{\pi n}{2}\right) - 2\sin\left(\frac{\pi n}{2}\right)\right)}{4\pi^2 n^2} - \frac{\lambda^2\left(2\sin\left(\frac{3\pi n}{2}\right) + \pi n \cos\left(\frac{3\pi n}{2}\right)\right)}{8\pi^2 n^2} - \frac{\lambda^2\left(\pi n \cos\left(\frac{\pi n}{2}\right) - 2\sin\left(\frac{\pi n}{2}\right)\right)}{8\pi^2 n^2}$$

1e.

Plot the true signal for the triangular function, on top of the approximations with 2, 5, 10, and 100 terms. Again, set $\lambda = 2\pi$.

In [4]:
```
#Write your answer here

lamb = 2 * np.pi
def compute_triangle(k):
    Bn = []
    kn = [2 * n * np.pi/lamb for n in range(1, k+1)]
    for n in range(1, k+1):
        Bn.append(2/lamb * (lamb**2 * np.cos(np.pi*n) * (np.pi * n * np.cos(
                  - lamb**2 * (2 * np.sin(3 * np.pi * n/2) + np.pi * n * np.c
                  - lamb**2 * (np.pi * n * np.cos(np.pi * n/2) - 2 * np.sin(n
    fn = lambda x: np.sum([Bn[i] * np.sin(kn[i] * x) for i in range(len(Bn))
    return fn

triangle_2 = compute_triangle(2)
triangle_5 = compute_triangle(5)
triangle_10 = compute_triangle(10)
triangle_100 = compute_triangle(100)

x_values = np.linspace(0, lamb, 200)

def triangle(x):
    if 0 <= x <=lamb/4:
        return x
    elif lamb/4 < x <= 3 * lamb/4:
        return lamb/2 - x
    else:
        return x - lamb

vfunc_tri = np.vectorize(triangle)
true_triangle = [vfunc_tri(x_values[i]) for i in range(len(x_values))]

plt.scatter(x_values, [triangle_2(x_values[i]) for i in range(len(x_values))
plt.scatter(x_values, [triangle_5(x_values[i]) for i in range(len(x_values))
plt.scatter(x_values, [triangle_10(x_values[i]) for i in range(len(x_values)
plt.scatter(x_values, [triangle_100(x_values[i]) for i in range(len(x_values
plt.plot(x_values, true_triangle, c = 'green')
plt.legend(["n = 2", "n = 5", 'n = 10', "n = 100", 'true plot'])
plt.show()
```



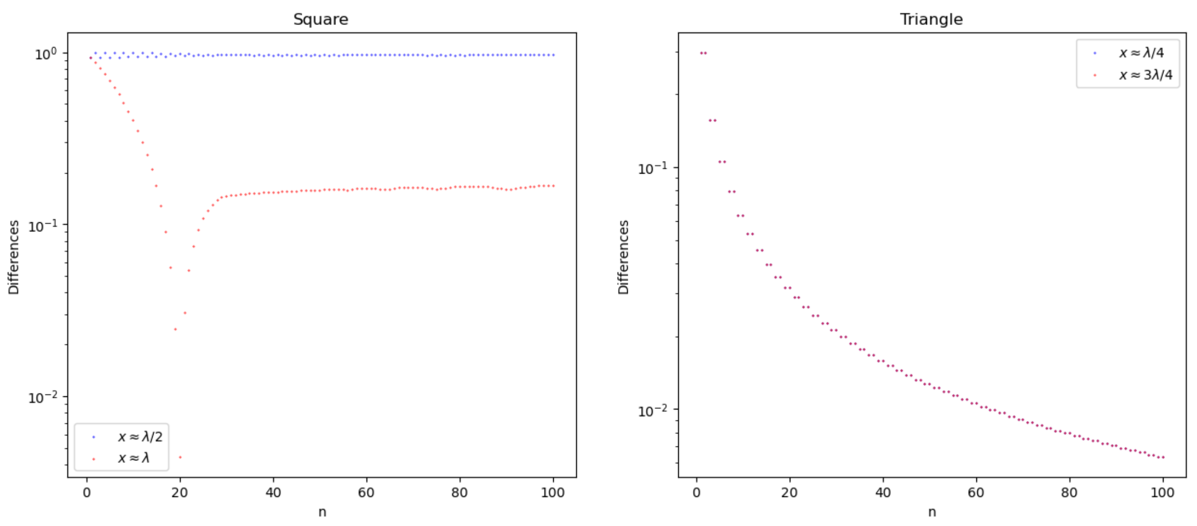
1f.

For each of the functions whose Fourier series you have computed, estimate the absolute value of the overshoot or undershoot at the kink or discontinuity caused by the Gibbs phenomenon as a function of the number of terms in the series. Plot your results on the sample plot for each $n \leq 100$, with the errors in a log-scale.

Hint: if you used a fine enough resolution for your plotting array ($k_n \Delta x \ll 1$), just find the sampled overshoot by e.g. comparing the maximum/minimum sampled value to the true max/min.

In [5]:
```
# Square wave
delta = 0.1

diffs_sq1 = []
diffs_sq2 = []

diffs_tri1 = []
diffs_tri2 = []
for n in range(1, 101):
    fourier_sq = compute_square(n)
    fourier_tri = compute_triangle(n)
    #10 points within this interval is a fine enough resolution

    #first discontinuity - Square wave
    x_values = np.linspace(lamb/2 - 0.1, lamb/2 + 0.1, 30)
    diffs_sq1.append(np.abs(np.max([fourier_sq(x_values[i]) for i in range(l
                        np.max([vfunc_sq(x_values[i]) for i in range(len(x_v

    # second discontinuity - Square wave
    x_values = np.linspace(lamb - 0.1, lamb + 0.1, 30)
    diffs_sq2.append(np.abs(np.max([fourier_sq(x_values[i]) for i in range(l
                        np.max([vfunc_sq(x_values[i]) for i in range(len(x_v

    #first discontinuity - Triangle Wave
    x_values = np.linspace(lamb/4 - delta, lamb/4 + delta, 75)
    diffs_tri1.append(np.abs(np.max([fourier_tri(x_values[i]) for i in range
                        np.max([vfunc_tri(x_values[i]) for i in range(len(x_

    #second discontinuity - Triangle Wave
    x_values = np.linspace(3*lamb/4 - delta, 3 * lamb/4 + delta, 75)
    diffs_tri2.append(np.abs(np.min([fourier_tri(x_values[i]) for i in range
                        np.min([vfunc_tri(x_values[i]) for i in range(len(x_
```

In [6]:
```
n = np.linspace(1, 100, 100)

fig, axs = plt.subplots(1, 2)
fig.set_figheight(6)
fig.set_figwidth(15)
axs[0].scatter(n, diffs_sq1, s = 0.2, c = 'b')
axs[0].scatter(n, diffs_sq2, s = 0.2, c = 'r')
axs[0].legend([r'$x \approx \lambda/2$', r'$x \approx \lambda$'])
# axs[1].title('Overshoot/Undershoot for Square Wave')
axs[0].set_xlabel('n')
axs[0].set_ylabel('Differences')
axs[0].set_title('Square')
axs[0].set_yscale('log')

axs[1].scatter(n, diffs_tri1, s = 0.2, c = 'b')
axs[1].scatter(n, diffs_tri2, s = 0.2, c = 'r')
axs[1].legend([r'$x \approx \lambda/4$', r'$x \approx 3\lambda/4$'])
axs[1].set_ylabel('Differences')
axs[1].set_xlabel('n')
axs[1].set_yscale('log')
axs[1].set_title('Triangle')
# axs[2].title('Overshoot/Undershoot for Triangle Wave')
plt.show()
```



Note that the plot on the left actually makes sense - I'm taking the absolute value of the differences, so for small values of n, we know that the Fourier transform significantly undershoots the square wave.

1g.

(Completely Optional---Not worth any points) A function on a continuous interval consists of a value at each point in an uncountable domain, so has an uncountable number of degrees of freedom. However, its Fourier series has a countable number of degrees of freedom. How do we resolve this (apparent) paradox?

(Hint: one of the defining properties of a Hilbert space is *nondegeneracy*: $||f|| = \langle f, f\rangle = 0 \implies f = 0$.)

Write your answer here

On notation

When working with functions that map to $\mathbb{C}$ instead of $\mathbb{R}$, the inner product $\langle f, g\rangle = \int f^*(x)g(x)\,dx$ is used. In $\mathbb{C}$, the Fourier basis is $f_n(x) = e^{ik_n x}$, with complex coefficients C_n, for $-\infty < n < \infty$. For real functions in this convention, we have $C_n = C^*_{-n} = \frac{1}{2}(A_n - iB_n)$.

Complex numbers are preferred particularly in quantum mechanics. It is common in this context to use *Dirac notation*, in which elements of a Hilbert space are represented by objects like $|f\rangle$, known as *kets*. It is common to define a *dual* Hilbert space H^* as the set of continuous linear maps from the Hilbert space H to $\mathbb{R}$ (or $\mathbb{C}$). The *Riesz representation theorem* guarantees that each element of the dual space can be respresented by an element of the Hilbert space, which acts on other elements by the inner product , i.e. $\forall g \in H^*,\ \exists f \in H \text{ s. t. } \forall h \in H,\ g(h) = \langle f, h\rangle$. Thus, in Dirac notation, elements of the dual space are denoted using *bras*, $\langle f|$, so that the inner product can be denoted by *brackets* $\langle f|g\rangle$.

An equivalent way to state the completeness of an orthogonal basis $|f_n\rangle$ is that the projection operator into this basis $\sum_n \frac{1}{\langle f_n|f_n\rangle} |f_n\rangle\langle f_n|$ is the identity operator 1. For the Fourier basis, borrowing Dirac notation, we have

$$1 = \sum_{n=0}^{\infty} \left(\frac{1}{S_n} |s_n\rangle\langle s_n| + \frac{1}{C_n} |c_n\rangle\langle c_n| \right).$$

Several complete, orthogonal bases exist for a variety of spaces (like the full real line, the sphere, etc.) and are useful in various areas of physics, including:

- Legendre polynomials
- Bessel functions
- Hermite polynomials
- Chebyshev polynomials

These functions can be derived as solutions to common differential equations, which become "independent" in the appropriate basis. Then a solution is often found by decomposing boundary conditions into a sum over these independent "modes".