Application of the COM Framework to Saturn's Moon System

Martin Doina

April 25, 2025

ABSTRACT

This paper explores the application of the Continuous Oscillatory Model (COM) framework to Saturn's moon system. The COM framework, which has successfully modeled planetary spacing in both our Solar System and the TRAPPIST-1 system, is tested on Saturn's moons to determine if the same mathematical patterns apply at the satellite system scale. Using the fundamental constants LZ (1.23498) and HQS (0.235) with various phase functions, we analyze the semi-major axis distribution of Saturn's moons and compare different subsets of moons. The analysis reveals that Saturn's regular moons follow the COM pattern with moderate accuracy (R^2 = 0.564) using the hyperbolic tangent phase function, the same function that provided the best fit for the TRAPPIST-1 system. The inner regular moons (Mimas through Rhea) show particularly good agreement with the COM model, while Titan's position is significantly underestimated. These findings suggest that the COM framework captures fundamental organizing principles that operate across different astronomical scales, from planetary systems to moon systems, though with varying degrees of accuracy reflecting different formation processes.

Introduction

Saturn, the sixth planet from the Sun, hosts the most extensive moon system in our Solar System, with 83 confirmed moons. These moons vary greatly in size, composition, and orbital characteristics, and can be categorized into several groups: ring moons, co-orbital moons, regular moons, and irregular moons. The regular moons, which orbit in the equatorial plane with nearly circular orbits, are believed to have formed from the same circumplanetary disk that formed Saturn, while the irregular moons were likely captured from heliocentric orbits.

The Continuous Oscillatory Model (COM) framework has demonstrated remarkable success in modeling planetary spacing in both our Solar System and exoplanetary systems like TRAPPIST-1. This framework uses two fundamental constants: LZ (1.23498) and HQS (0.235), along with system-specific phase functions, to model the distribution of astronomical objects.

This paper explores whether the same mathematical patterns that describe planetary spacing might also apply to Saturn's moon system. If successful, this would suggest that the COM framework captures fundamental organizing principles that operate across vastly different astronomical scales.

Methodology

The COM Framework

The COM framework models astronomical distributions using the following formula:

$$a_n = a_0 \cdot \lambda^n (1 + \eta \cdot \varphi(n))$$

Where:

- a_n is the semi-major axis at position n
- a_0 is the baseline semi-major axis (innermost object)
- λ is the LZ constant (1.23498)
- η is the HQS constant (0.235)
- $\varphi(n)$ is a phase function

For this analysis, we test five different phase functions:

- Sine (Solar System style): $\varphi(n) = \sin(4\pi n)$
- Sine (Pulsar style): $\varphi(n) = \sin(2\pi n/24)$
- Cosine: $\varphi(n) = \cos(4\pi n)$
- Hyperbolic tangent (TRAPPIST-1 style): $\varphi(n) = \tanh(n/2)$
- None: $\varphi(n) = 0$ (pure exponential growth)

Saturn's Moon Data

For this analysis, we use data for 15 of Saturn's moons, including their semi-major axes and classification:

Saturn's moons used in this analysis

Moon	Semi-Major Axis (km)	Type
Pan	133,584	Ring
Atlas	137,670	Ring
Prometheus	139,380	Ring
Pandora	141,720	Ring
Epimetheus	151,422	Co-orbital
Janus	151,472	Co-orbital
Mimas	185,539	Regular
Enceladus	238,042	Regular
Tethys	294,619	Regular
Dione	377,396	Regular
Rhea	527,108	Regular
Titan	1,221,870	Regular
Hyperion	1,500,934	Irregular
Iapetus	3,560,820	Irregular
Phoebe	12,947,918	Irregular

Analysis Approach

Our analysis follows these steps:

1. Divide Saturn's moons into different subsets for analysis:
   - All Moons: All 15 moons in the dataset
   - Regular Moons: The 6 regular moons (Mimas through Titan)
   - Inner Moons: The 6 innermost moons (Pan through Janus)
   - Major Moons: The 6 largest moons (Mimas through Titan)
   - Outer Moons: The 3 outermost moons (Hyperion, Iapetus, Phoebe)
2. For each subset, apply the COM model with different phase functions
3. Evaluate model performance using:
   - R^2 score (coefficient of determination)
   - Mean Absolute Percentage Error (MAPE)
   - Individual error percentages for each moon
4. Identify the best-performing phase function for each subset
5. Use the best model to predict potential additional moons

Results

Phase Function Comparison

Table 2 shows the performance of different phase functions across various subsets of Saturn's moons.

Performance comparison of different moon subsets

Subset	Best Phase	R^2 Score	MAPE (%)
All Moons	tanh	0.352	115.72
Regular Moons	tanh	0.564	13.16
Inner Moons	sin	-351.422	67.02
Major Moons	tanh	0.564	13.16
Outer Moons	cos	-0.399	45.79

The hyperbolic tangent (tanh) phase function provides the best fit for both the Regular Moons and All Moons subsets, with the Regular Moons showing a much better fit (R^2 = 0.564) than the complete set (R^2 = 0.352). The Inner Moons and Outer Moons subsets show very poor fits, with negative R^2 scores indicating that the model performs worse than a horizontal line.

Regular Moons Analysis

Since the Regular Moons subset shows the best fit to the COM model, we analyze it in more detail. Table 3 shows the observed and predicted semi-major axes for Saturn's regular moons, along with the percentage error.

COM model performance for Saturn's regular moons

Moon	Observed (km)	Predicted (km)	Error (%)

Mimas	185,539	185,539	0.00
Enceladus	238,042	254,021	6.71
Tethys	294,619	333,626	13.24
Dione	377,396	423,811	12.30
Rhea	527,108	529,370	0.43
Titan	1,221,870	656,590	-46.26

The COM model performs very well for the inner regular moons (Mimas through Rhea), with errors below 13.3%. However, it significantly underestimates Titan's orbit, with a -46.26% error.

Visualization of Results

Figure 1 shows the comparison between observed and COM-predicted semi-major axes for all of Saturn's moons, using different phase functions.

[IMAGE: Comparison of observed and COM-predicted semi-major axes for Saturn's moons]

Figure 2 shows a detailed comparison for Saturn's regular moons using the best-fitting tanh phase function.

[IMAGE: Detailed analysis of Saturn's regular moons with the COM framework]

Predictions for Additional Moons

Based on the best-fitting model for regular moons, we can predict the positions of potential additional moons beyond the known ones. Table 4 shows these predictions.

Predictions for potential additional moons

Moon	Predicted Distance (km)
Moon-7	812,181
Moon-8	1,003,624
Moon-9	1,239,727
Moon-10	1,531,162
Moon-11	1,891,010
Moon-12	2,335,385

Figure 3 visualizes these predictions in relation to the known moons.

[IMAGE: Extended COM predictions for Saturn's moons]

Discussion

Comparison with Planetary Systems

The COM framework's performance for Saturn's moon system ($R^2 = 0.564$ for regular moons) is moderate compared to its performance for planetary systems:

- Solar System: $R^2 > 0.95$ with $\sin(4n\pi)$ phase function
- TRAPPIST-1: $R^2 = 0.889$ with $\tanh(n/2)$ phase function

This difference in performance likely reflects the different formation and evolution processes of moon systems compared to planetary systems. While both form from disks of material, moon systems are more strongly influenced by the gravitational field of their parent planet and by resonances between moons.

Phase Function Significance

It is noteworthy that the same phase function (tanh) provides the best fit for both the TRAPPIST-1 system and Saturn's regular moons. This suggests a potential connection between these systems, despite their vastly different scales and compositions.

The poor performance of the COM model for inner ring moons and outer irregular moons is consistent with our understanding of their formation. Ring moons formed through complex processes within Saturn's rings, while irregular moons were captured from heliocentric orbits rather than forming in situ.

Titan's Anomalous Position

The significant underestimation of Titan's orbit (-46.26% error) is the most notable deviation from the COM model. This suggests that Titan may have formed through a different process than the inner regular moons, or that its orbit has been significantly altered by dynamical processes not captured by the COM framework.

Titan contains more than 90% of the mass in Saturn's moon system and is the only moon with a substantial atmosphere. Its unique characteristics support the idea that it may have a different formation history than the other regular moons.

Implications for Moon Formation Theories

The moderate success of the COM framework in modeling Saturn's regular moons suggests that there may be underlying mathematical patterns in moon formation similar to those in planetary formation, but with important differences.

The clear distinction in model performance between different moon groups (regular vs. irregular) aligns with current theories about different formation mechanisms for these groups. This provides additional support for the idea that regular moons formed from the same circumplanetary disk that formed Saturn, while irregular moons were captured.

Conclusion

This analysis demonstrates that the COM framework, with its fundamental constants LZ (1.23498) and HQS (0.235), can be applied to Saturn's moon system with moderate success. The framework works best for Saturn's regular moons, achieving an R^2 score of 0.564 with the hyperbolic tangent phase function.

The same phase function (tanh) provides the best fit for both the TRAPPIST-1 exoplanetary system and Saturn's regular moons, suggesting a potential connection between these systems despite their different scales.

The varying performance of the COM model across different moon groups reflects their different formation processes, with regular moons showing the best fit to the model. The significant underestimation of Titan's orbit highlights the unique nature of this moon within Saturn's system.

These findings support the idea that the COM framework captures fundamental organizing principles that operate across different astronomical scales, from planetary systems to moon systems, though with varying degrees of accuracy reflecting different formation processes.

Future work should extend this analysis to the moon systems of other giant planets (Jupiter, Uranus, and Neptune) to determine if similar patterns exist across different satellite systems.

Acknowledgments

I would like to acknowledge the MANUS AI system for its assistance in analyzing Saturn's moon system and implementing the COM framework.

Appendix: Implementation Code

The following Python code was used to implement the COM framework for Saturn's moon system analysis:

```
"""
Saturn Moons COM Framework Analysis

This script applies the Continuous Oscillatory Model (COM) framework to analyze
the spacing of Saturn's major moons, testing whether the same mathematical patterns
that describe planetary spacing might also apply to satellite systems.

Author: Martin Doina
Date: April 25, 2025
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from scipy.stats import ks_2samp
from sklearn.metrics import r2_score

# --- Step 1: Define Saturn's major moons data ---
# Data source: NASA Planetary Fact Sheets and JPL Solar System Dynamics
saturn_moons_data = {
    'Name': [
        'Pan', 'Atlas', 'Prometheus', 'Pandora', 'Epimetheus', 'Janus',
        'Mimas', 'Enceladus', 'Tethys', 'Dione', 'Rhea', 'Titan',
        'Hyperion', 'Iapetus', 'Phoebe'
    ],
    'Semi-Major Axis (km)': [
        133584, 137670, 139380, 141720, 151422, 151472,
        185539, 238042, 294619, 377396, 527108, 1221870,
        1500934, 3560820, 12947918
    ],
    'Type': [
        'Ring', 'Ring', 'Ring', 'Ring', 'Co-orbital', 'Co-orbital',
        'Regular', 'Regular', 'Regular', 'Regular', 'Regular', 'Regular',
        'Irregular', 'Irregular', 'Irregular'
    ]
}

# Convert to DataFrame for easier manipulation
moons_df = pd.DataFrame(saturn_moons_data)

# --- Step 2: Define COM model functions ---
def com_model(a0, n, phase_func='sin', lz=1.23498, hqs=0.235):
    """
```

```
    Generate semi-major axis values using the COM framework.

    Parameters:
    - a0: Baseline semi-major axis (innermost moon)
    - n: Number of positions to generate
    - phase_func: Phase function to use ('sin', 'tanh', 'cos', or 'none')
    - lz: LZ constant (default: 1.23498)
    - hqs: HQS constant (default: 0.235)

    Returns:
    - List of semi-major axis values following the COM pattern
    """
    semi_major_axes = []

    for i in range(n):
        # Apply different phase functions
        if phase_func == 'sin':
            phase = np.sin(4 * np.pi * i)  # Similar to Solar System
        elif phase_func == 'sin2':
            phase = np.sin(2 * np.pi * i / 24)  # Similar to pulsar analysis
        elif phase_func == 'cos':
            phase = np.cos(4 * np.pi * i)
        elif phase_func == 'tanh':
            phase = np.tanh(i / 2)  # Similar to TRAPPIST-1
        elif phase_func == 'none':
            phase = 0  # No phase modulation, pure exponential growth
        else:
            raise ValueError(f"Unknown phase function: {phase_func}")

        # Apply COM formula: a0 * LZ^n * (1 + HQS * phase)
        a_n = a0 * (lz ** i) * (1 + hqs * phase)
        semi_major_axes.append(a_n)

    return semi_major_axes

def evaluate_model(observed, predicted):
    """
    Evaluate model performance using R² score and mean absolute percentage error.

    Parameters:
    - observed: Observed semi-major axis values
    - predicted: Predicted semi-major axis values

    Returns:
    - Dictionary with evaluation metrics
    """
    # Calculate R² score
    r2 = r2_score(observed, predicted)

    # Calculate mean absolute percentage error (MAPE)
    mape = np.mean(np.abs((observed - predicted) / observed)) * 100
```

```
    # Calculate individual errors
    errors = [(predicted[i] - observed[i]) / observed[i] * 100 for i in range(len(obs

    return {
        'r2': r2,
        'mape': mape,
        'errors': errors
    }

# --- Step 3: Analyze different moon subsets ---
# We'll analyze different subsets of moons to find patterns
subsets = {
    'All Moons': moons_df['Semi-Major Axis (km)'].values,
    'Regular Moons': moons_df[moons_df['Type'] == 'Regular']['Semi-Major Axis (km)'].
    'Inner Moons': moons_df.iloc[:6]['Semi-Major Axis (km)'].values,
    'Major Moons': moons_df.iloc[6:12]['Semi-Major Axis (km)'].values,
    'Outer Moons': moons_df.iloc[12:]['Semi-Major Axis (km)'].values
}

# --- Step 4: Test different phase functions on each subset ---
phase_functions = ['sin', 'sin2', 'cos', 'tanh', 'none']
results = {}

for subset_name, subset_data in subsets.items():
    subset_results = {}

    # Use the innermost moon as baseline
    a0 = subset_data[0]
    n = len(subset_data)

    for phase_func in phase_functions:
        # Generate COM predictions
        predicted = np.array(com_model(a0, n, phase_func))

        # Evaluate model
        evaluation = evaluate_model(subset_data, predicted)

        # Store results
        subset_results[phase_func] = {
            'predicted': predicted,
            'r2': evaluation['r2'],
            'mape': evaluation['mape'],
            'errors': evaluation['errors']
        }

    # Find best phase function based on R² score
    best_phase = max(phase_functions, key=lambda x: subset_results[x]['r2'])
    subset_results['best_phase'] = best_phase

    results[subset_name] = subset_results
```

```
# --- Step 5: Visualize results ---
# Create figure with subplots
fig, axs = plt.subplots(2, 2, figsize=(15, 12))
fig.suptitle('Saturn Moons: COM Framework Analysis', fontsize=16)

# Plot 1: Linear comparison of all moons
axs[0, 0].scatter(range(len(subsets['All Moons'])), subsets['All Moons'], color='red'
for phase_func in phase_functions:
    predicted = results['All Moons'][phase_func]['predicted']
    axs[0, 0].plot(range(len(predicted)), predicted, 'o-', label=f'COM ({phase_func})
axs[0, 0].set_xlabel("Moon Index")
axs[0, 0].set_ylabel("Semi-Major Axis (km)")
axs[0, 0].set_title("All Saturn Moons - Linear Scale")
axs[0, 0].legend()
axs[0, 0].grid(True, alpha=0.3)

# Plot 2: Logarithmic comparison of all moons
axs[0, 1].scatter(range(len(subsets['All Moons'])), subsets['All Moons'], color='red'
for phase_func in phase_functions:
    predicted = results['All Moons'][phase_func]['predicted']
    axs[0, 1].plot(range(len(predicted)), predicted, 'o-', label=f'COM ({phase_func})
axs[0, 1].set_xlabel("Moon Index")
axs[0, 1].set_ylabel("Semi-Major Axis (km)")
axs[0, 1].set_title("All Saturn Moons - Log Scale")
axs[0, 1].set_yscale('log')
axs[0, 1].legend()
axs[0, 1].grid(True, alpha=0.3)

# Plot 3: Regular moons comparison
axs[1, 0].scatter(range(len(subsets['Regular Moons'])), subsets['Regular Moons'], col
best_phase = results['Regular Moons']['best_phase']
predicted = results['Regular Moons'][best_phase]['predicted']
axs[1, 0].plot(range(len(predicted)), predicted, 'o-', label=f'COM ({best_phase})')
axs[1, 0].set_xlabel("Moon Index")
axs[1, 0].set_ylabel("Semi-Major Axis (km)")
axs[1, 0].set_title(f"Regular Moons - Best Phase: {best_phase}")
axs[1, 0].legend()
axs[1, 0].grid(True, alpha=0.3)

# Plot 4: Results table
axs[1, 1].axis('tight')
axs[1, 1].axis('off')
table_data = [
    ['Subset', 'Best Phase', 'R² Score', 'MAPE (%)'],
]
for subset_name in subsets.keys():
    best_phase = results[subset_name]['best_phase']
    r2 = results[subset_name][best_phase]['r2']
    mape = results[subset_name][best_phase]['mape']
    table_data.append([
        subset_name,
```

```
        best_phase,
        f"{r2:.3f}",
        f"{mape:.2f}"
    ])
table = axs[1, 1].table(cellText=table_data, loc='center', cellLoc='center')
table.auto_set_font_size(False)
table.set_fontsize(12)
table.scale(1, 1.5)
axs[1, 1].set_title("COM Framework Performance by Moon Subset")

# Add COM parameters text
param_text = (
    f"COM Framework Parameters:\n"
    f"LZ = 1.23498\n"
    f"HQS = 0.235\n"
)
fig.text(0.02, 0.02, param_text, fontsize=12, bbox=dict(facecolor='white', alpha=0.8)

plt.tight_layout(rect=[0, 0.05, 1, 0.95])
plt.savefig('saturn_moons_com_analysis.png', dpi=300)

# --- Step 6: Create detailed visualization for regular moons ---
# Regular moons are most likely to follow COM patterns due to their formation process
regular_moons = moons_df[moons_df['Type'] == 'Regular']
regular_names = regular_moons['Name'].values
regular_distances = regular_moons['Semi-Major Axis (km)'].values

best_phase = results['Regular Moons']['best_phase']
predicted = results['Regular Moons'][best_phase]['predicted']
errors = results['Regular Moons'][best_phase]['errors']

plt.figure(figsize=(12, 8))
plt.scatter(range(len(regular_distances)), regular_distances, color='red', s=100, lab
plt.plot(range(len(predicted)), predicted, 'o-', color='blue', label=f'COM ({best_pha

# Add moon names to the plot
for i, name in enumerate(regular_names):
    plt.annotate(name, (i, regular_distances[i]), textcoords="offset points",
                 xytext=(0,10), ha='center')

    # Add error percentage
    plt.annotate(f"{errors[i]:.1f}%", (i, regular_distances[i]), textcoords="offset p
                 xytext=(0,-15), ha='center', color='green')

plt.xlabel("Moon Index")
plt.ylabel("Semi-Major Axis (km)")
plt.title(f"Saturn's Regular Moons: COM Framework Analysis (R² = {results['Regular Mo
plt.grid(True, alpha=0.3)
plt.legend()
plt.tight_layout()
plt.savefig('saturn_regular_moons_com_analysis.png', dpi=300)
```

```
# --- Step 7: Print summary of findings ---
print("\n" + "="*80)
print("COM Framework Analysis of Saturn's Moons")
print("="*80)

# Print results for each subset
for subset_name in subsets.keys():
    best_phase = results[subset_name]['best_phase']
    r2 = results[subset_name][best_phase]['r2']
    mape = results[subset_name][best_phase]['mape']

    print(f"\n{subset_name}:")
    print(f"  Best phase function: {best_phase}")
    print(f"  R² Score: {r2:.3f}")
    print(f"  Mean Absolute Percentage Error: {mape:.2f}%")

    if subset_name == 'Regular Moons':
        print("\nDetailed results for Regular Moons:")
        print("-"*60)
        print(f"{'Moon':<12} {'Observed (km)':<15} {'Predicted (km)':<15} {'Error (%)
        print("-"*60)

        for i, name in enumerate(regular_names):
            observed = regular_distances[i]
            predicted_val = predicted[i]
            error = errors[i]
            print(f"{name:<12} {observed:<15.0f} {predicted_val:<15.0f} {error:<10.2f

print("="*80)

# --- Step 8: Save results to CSV ---
# Create results dataframe for regular moons
results_df = pd.DataFrame({
    'Moon': regular_names,
    'Observed_km': regular_distances,
    'COM_Predicted_km': predicted,
    'Error_Percent': errors
})
results_df.to_csv('saturn_moons_com_analysis_results.csv', index=False)
print("Results saved to 'saturn_moons_com_analysis_results.csv'")

# --- Step 9: Predict potential undiscovered moons ---
# Extend the COM model to predict additional moons beyond the known ones
best_phase = results['Regular Moons']['best_phase']
a0 = regular_distances[0]  # Use Mimas as baseline
extended_n = 12  # Predict 6 additional moons beyond the 6 known regular moons

extended_predictions = com_model(a0, extended_n, best_phase)

plt.figure(figsize=(12, 8))
plt.scatter(range(len(regular_distances)), regular_distances, color='red', s=100, lab
```

```
plt.plot(range(extended_n), extended_predictions, 'o-', label=f'COM Extended Predicti
plt.xlabel("Moon Index")
plt.ylabel("Semi-Major Axis (km)")
plt.title("Extended COM Prediction for Saturn's Moons")
plt.yscale('log')
plt.grid(True, alpha=0.3)
plt.legend()

# Add moon names to the plot
for i, name in enumerate(regular_names):
    plt.annotate(name, (i, regular_distances[i]), textcoords="offset points",
                 xytext=(0,10), ha='center')

# Add potential prediction markers
for i in range(len(regular_distances), extended_n):
    plt.axhline(y=extended_predictions[i], color='green', linestyle='--', alpha=0.3)
    plt.text(extended_n-1, extended_predictions[i], f"Predicted: {extended_predictior
             va='center', ha='right', bbox=dict(facecolor='white', alpha=0.7))

plt.tight_layout()
plt.savefig('saturn_moons_extended_prediction.png', dpi=300)

# Print predictions for potential undiscovered moons
print("\nPredictions for potential additional moons:")
print("-"*60)
print(f"{'Moon':<12} {'Predicted Distance (km)':<25}")
print("-"*60)
for i in range(len(regular_distances), extended_n):
    print(f"Moon-{i+1:<8} {extended_predictions[i]:<25.0f}")
print("="*80)
```