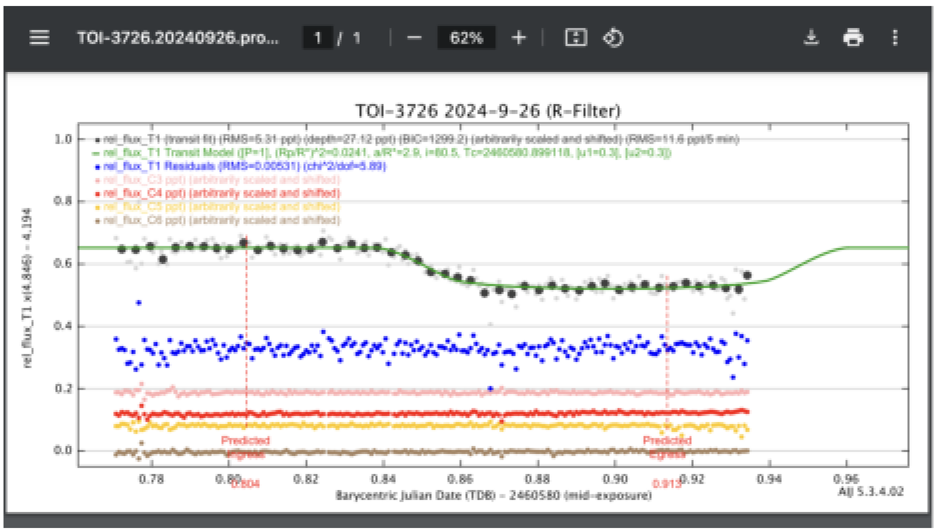
TOI-3726 Light Curves and Fits

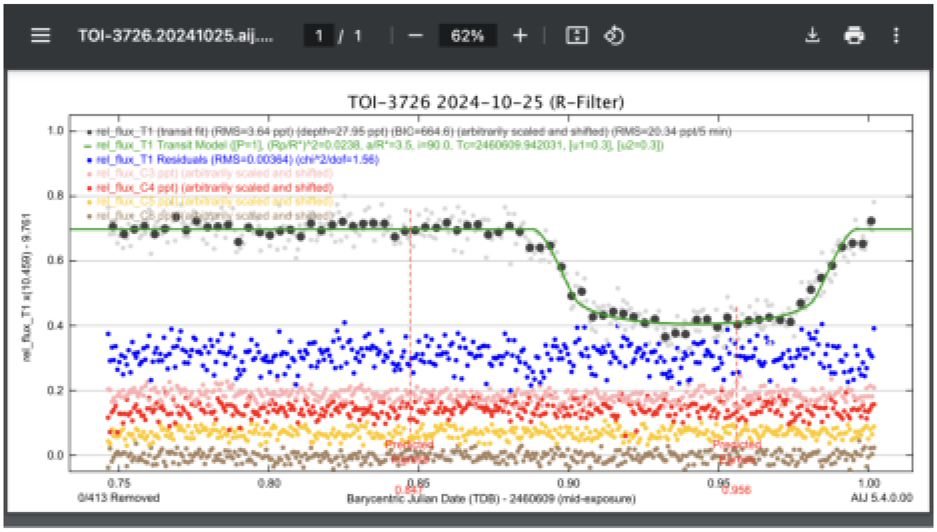
This is a working notebook of TOI-3726 data analysis.

- GitHub link
- Swarthmore Planet Finder Prediction on TOI-3726

In [48]:
```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
image = mpimg.imread("./20240926/light_curve_20240926.PNG")
plt.figure(figsize=(6, 4))
plt.imshow(image)
plt.axis("off")
plt.show()
```



In [49]:
```
image = mpimg.imread("./20241025/light_curve_20241025.PNG")
plt.figure(figsize=(6, 4))
plt.imshow(image)
plt.axis("off")
plt.show()
```



Analysis

Transit Timing Variation (TTV)

Period Analysis

1. Period Based on the Observations on 09/26/2024 and 10/25/2024.
    - The below calculation indicates the period of the transits remains as reported by ExoMast.

In [3]:
```python
# Tc(transit center timing)
Tc_20241025 = 2460609.941884418
Tc_20240926 = 2460580.899117944
period_exo_mast = 4.8404431

periods = (Tc_20241025 - Tc_20240926) / period_exo_mast
print(f"The number of periods between the two transit centers is: {periods}"
```

```
The number of periods between the two transit centers is: 6.000022286014968
```

2. The Amount of T_c (Transit Center) Shift Compared with Swathmore Report

In [4]:
```python
# For 09/26/2024
shift_09262024 = 0.899117944 - 0.8590
print(f"09-26-2024 shift: {shift_09262024}")

# For 10/25/2024
shift_10252024 = 0.941884418 - 0.9016
print(f"10-25-2024 shift: {shift_10252024}")
```

```
09-26-2024 shift: 0.040117944000000016
10-25-2024 shift: 0.040284418000000044
```

3. Compare TESS Sector 59 and 73 Light Curves

- For TOI-3726, TESS collected 2 sectors of data: 59 and 73. Somehow, there are two sets of data for sector 59.
- With the base BJD of 2457000, sector 59 is between (2910 - 2937 BJD) and sector 73 is between (3285 - 3313 BJD)
- The current ExoMast report was based on the data of sector 73(see thedownloaded report), even though the data of sector 59 is in no less quality.
- The flux of both sectors are individually flattened and folded with the reported transit period (4.8404431 BJD). It seems to indicate the transit center has shifted to a later time in sector 73 compared with sector 59. The amount of shift is close to the amount between the current ground follow-up observations and the sector 73. See the below comparison.
- However, I don't know how to fit the light curves and the shift is a rough estimate. screenshot.

Compare the TOIs in TESS Sector 59 and 73 to Identify TTVs

We also observed a significant shift in the TOI-3726 trasit timing between the ground follow-up observations (2024-09-25, 2024-10-25) and the TESS sector 73. We also discovered a similar significant shift of the transit timing in TOI-3726 between sector 59 and 73 in the earlier study. We also observed a similar significant shift in the trasit timeing between 2024-09-25 and the TESS sector 73.

There are a few theories about the nature of the shift:

1. A genuine TTV caused by other nearby planetary objects in the same star star system.
2. A TESS clock drift between different sectors.
3. An accumulation of the inaccuracy in the calculated period.

We can explore the the theory #2 with the current TESS data with the following procedure:

1. A few TOIs are selected randomly from ExoFOP-TESS, so that these candidates all have been observed in Sector 59 and 73.
2. The light curves are constructed using the python "lightkurve" libary, then folded and binned using their ExoMast reported transit-periods and epoch-times.
3. The light curves from sector 59 and 73 are graphed, so that the shift of the folded transits can be compared.

Theory #2: TESS Clock Drift

- Hypotheses
  - If this theory were true, we should observe simiar shifts in all the candidate TOIs. Such shifts should be in the same direction and by the same shift amount.
  - Otherewise, we should observe shifts of different directions and amounts. Or no significant shifts at all, as TTVs have been observed rarely.
- Findings
  - Compare the TOIs in TESS Sector 59 and 73 to Identify TTVs
  - As shown in the above notebook, we discovered that significant transit-timing shifts are present in almost all candidate TOIs! Such shifts are of different directions and amounts. This discovery indicates the following two conclusions:
    - There is no TESS clock drift between sector 59 and sector 73.
    - TTVs are not as rare as they have been believed so.

Theory #3 Calculation Error Accumulation

- LATTE Report
  - LATTE command `python -m LATTE --tic "122695048" --north --sector 59,73`
- TESS Reports
  - Sector 59
    - The DVM report for Sector 59 does not appear in ExoMast website directly. I found it when running LATTE against the Sector 59 data. In the LATTE report(at the bottom of pag 1), there is a link pointing to the Sector 59 DVM report.
  - Sector 73
  - Model Prediction | Sector | Period (days) | Epoch (BTJD) | |--------|----------------------|-----------------------| | 59 | 4.84169 ± 0.00044 | 2912.8314 ± 0.0011 | | 73 | 4.84044 ± 0.00041 | 3290.4324 ± 0.0010 | | Diff % | 0.026% | |
- Error Estimation
  - Based on the reported periods of the two sectors, the cumulative drifts caused by the the max error and the average error are 0.066 and 0.033 BJDs between Sector 59 and 73.
  - The actual estimated drift over the time between Sector 59 and 73 is ~0.04 BJD.
- Conclusion
  - It appears the actual drift is within the reasonable interval of the period errors. Based on this comparison, we cannot rule out that the shift has been caused by the calculated period not being precise enough.

Explore TTV Possibility

Estimate Semi-major Axis of the Hypothetical New Planet X

```
- Assuming planet-X is in the same inclination plane as TOI-
3726.01, for X not to produce a transit, its semi-major axis
a_X needs to satisfy:
```

$$a_X \ge \frac{R*}{cos(\angle inclination)}$$

```
- Given the two observations on 20240926 and 20241025, we
have the below calculation.
```

In [ ]:
```
from math import cos, pi, sin, tan
# Note the R* = 1.05 R_SUN is from https://exo.mast.stsci.edu/exomast_planet
ground_observation = [
    {
        "obs": "20240926",
        "a/R*": 2.854413994,
        "inclination_degree":80.466677487,
        "R*": 1.05,
        "color": "blue",
        "Tc": 2460580.899117944,
        "predicted": {
            "offset": 2460580,
            "ingress": 0.8045,
            "Tc": 0.8590,
            "egress": 0.9135,
        },
    },
    {
        "obs": "20241025",
        "a/R*": 2.719625175,
        "inclination_degree": 76.245721043,
        "R*": 1.05,
        "color": "green",
        "Tc": 2460609.941884418,
        "predicted": {
            "offset": 2460609,
            "ingress": 0.8471,
            "Tc": 0.9016,
            "egress": 0.9561,
        },
    },
    {
        "obs": "20241227",
        "a/R*": 2.496465454,
        "inclination_degree": 72.975816460,
        "R*": 1.05,
        "color": "green",
        "Tc": 2460672.878957585,
        "predicted": {
            "offset": 2460672,
            "ingress": 0.7729,
            "Tc": 0.8274,
            "egress": 0.8819,
```

```
        },
    },
]
```

20241227 Observation Data Analysis

At this point, we have done three observations on 20240926, 20241025 and 20241227. The data have been fit with AstroImageJ and the fitted Tc are compared with the predicted transit center of the Transit Finder of the Swarthmore College.

- The shift in min is as below:

Date	Shift (min)
2024-09-26	57.77
2024-10-25	58.01
2024-12-27	74.24

- The Tc shifts of 20240926 and 20241025 are quite close to each other. However, the shift on 20241227 was much more signifiant of about 16 min.

In [ ]:
```
for entry in ground_observation:
    obs = entry["obs"]
    tc = entry["Tc"]
    predicted_tc = entry["predicted"]["offset"] + entry["predicted"]["Tc"]
    diff = (tc - predicted_tc) * 24 * 60
    print(f"{obs}: {diff:.2f} min")
```

In [10]:
```
tc_end = 2460672 + (0.7729 + 0.8819) / 2.0
tc_begin_73 =  2457000 +  3290.4324
tc_begin_59 = 2457000 + 2912.8314
```

In [13]:
```
tc_begin_59 = 2457000 + 2912.8314
tc_begin_73 =  2457000 +  3290.4324
tc_20241025 = 2460609.941884418
tc_20241227 = 2460672.878957585
```

In [35]:
```
# BJD converter: https://ssd.jpl.nasa.gov/tools/jdc/#/jd
tc_list = [
    {
        "name": "tc_sec_59",
        "tc_val": 2459912.8314,
        "n": 0,
        "sidereal_date": "2022-11-29 07:57:13",
        "period": 4.84169,
        "error": 0.00044,
        "source": "ExoMast sector 59 report.",
    },
    {
        "name": "tc_sec_73",
```

```
        "tc_val": 2460290.4324,
        "n": 78,
        "sidereal_date": "2023-12-11 22:22:39",
        "period": 4.84044,
        "error": 0.00041,
        "source": "ExoMast sector 73 report",
    },
    {
        "name": "tc_20240926",
        "tc_val": 2460580.899117944,
        "n": 138,
        "sidereal_date": "2024-09-27 09:34:44",
        "source": "AstroImageJ model fit.",
    },
    {
        "name": "tc_20241025",
        "tc_val": 2460609.941884418,
        "n": 144,
        "sidereal_date": "2024-10-26 10:36:19",
        "source": "AstroImageJ model fit.",
    },
    {
        "name": "tc_20241227",
        "tc_val": 2460672.878957585,
        "n": 157,
        "sidereal_date": "2024-12-28 09:05:42",
        "source": "AstroImageJ model fit.",
    },
    {
        "name": "tc_20250205",
        "tc_val": 2460711.610065628,
        "n": 165,
        "sidereal_date": "2025-02-05 02:38:30",
        "source": "AstroImageJ model fit.",
    },
]
```


In [56]:

```
# Calculate the best average period using the first sector 59 transit and th
# The elapsed time spans 2.081 years, which is a good average covering nearl
best_period = (tc_list[4]["tc_val"] - tc_list[0]["tc_val"]) / 157
best_period
```

Out[56]: 4.841067245763958

In [57]:

```
# Calculate the ground observation based period using the 20241025 and 20241
ground_obs_period = (tc_list[4]["tc_val"] - tc_list[3]["tc_val"]) / 13
ground_obs_period
```

Out[57]: 4.841313320534447

In [36]:

```
from datetime import date
from math import sin, pi

C_IN_KM_PER_SEC = 299792.458
EARTH_AVG_OBITAL_RADIUS_KM = 1.496e8
```

```
EARTH_SIDEREAL_PERIOD = 365.2564
DAYS_IN_2024 = 366

d_2022_autumn_equinox = date(2022, 9, 22)
d_2023_autumn_equinox = date(2023, 9, 23)
d_2024_autumn_equinox = date(2024, 9, 22)

theta_3726 = (5 * 60 * 60  + 9 * 60 + 11) / (24 * 60 * 60) * 2 * pi # The ar

d_20240926 = date(2024, 9, 26)
d_20241025 = date(2024, 10, 25)
d_20241227 = date(2024, 12, 27)

def calc_avg_velocity(d1, d2, d_autumn_equnox, offset_angle, days_in_year=EA
    """Calculate the average vertical velocity between d1 and d2 when the Ea
    alpha = (d1 - d_autumn_equnox).days / days_in_year * 2 * pi + offset_ang
    beta = (d2 - d_autumn_equnox).days / days_in_year * 2 * pi + offset_angl
    days_diff = (d2 - d1).days
    vertical_displacement = (sin(beta) - sin(alpha)) * radius
    vertical_velocity_km_sec = vertical_displacement / days_diff / 24 / 60 /
    return vertical_velocity_km_sec

def doppler_period(p_original, v=C_IN_KM_PER_SEC, v_source=0, v_observer=0):
    """Calculate the doppler effect on a original period.
    p_original: The original period when the source and the observer are rel
    v: The speed of the wave in the medium (e.g., speed of sound in air or s
    v_source: the velocity of the source relative to the medium (positive if
    v_observer: the velocity of the observer relative to the medium (positiv
    """
    return p_original * (v - v_source) / (v + v_observer)
```

In [ ]:
```
d_list = [
    (d_20240926, d_20241025, d_2024_autumn_equinox),
    (d_20241025, d_20241227, d_2024_autumn_equinox),
    (date(2022, 11, 29), date(2022, 12, 24), d_2022_autumn_equinox),
    (date(2023, 12, 11), date(2024, 1, 4), d_2023_autumn_equinox),
]
print(best_period)

for d1, d2, d_autum_equinox in d_list:
    v_observer = calc_avg_velocity(d1, d2, d_2024_autumn_equinox, (0.5 * pi
    p_theo_observer = doppler_period(best_period, v_observer=v_observer)
    print(f"{d1} - {d2}: v_earth: {v_observer:.3} km/sec,  \t p_theo_obs: {p
```

Explor the Romer Effect and the Doppler Effect

Basic Data

Name	TC Value	N	Sidereal Date	Period	Error	Source
tc_sec_59	2459912.8314	0	2022-11-29	4.84169	0.00044	ExoMast sector 59

Name	TC Value	N	Sidereal Date	Period	Error	Source
			07:57:13			report.
tc_sec_73	2460290.4324	78	2023-12-11 22:22:39	4.84044	0.00041	ExoMast sector 73 report
tc_20241025	2460609.941884418	144	2024-10-26 10:36:19	-	-	AstroImageJ model fit.
tc_20241227	2460672.878957585	157	2024-12-28 09:05:42	-	-	AstroImageJ model fit.

- The sidereal dates were converted using https://ssd.jpl.nasa.gov/tools/jdc/#/jd
- The sector 59 and sector 73 TCs and periods were taken from the ExoMast reports.

The Romer Effect and the Doppler Effect

- When the Earth revolves around the Sun, it generates a sinusoildal fluctuation on the observed exoplanet transit period which is known as the Romer effect. Fundamentally, it is caused by the Doppler effect.
- During the 2024 fall-winter time, the Earth is moving toward TOI-3726 as it revolves around the Sun. This movement creates a small blue-shift due to the Doppler effect.
- Such blue-shift causes the observed period to be shorter than the actually host star transit period.
- The host star transit period is approximated by the average of the observed period between the first transit observed in sector 59 and the ground observed transit on 20241227 (where the full transit was captured).

Calculation

- `p_avg` : the average period
  - The t_c of the first transit occurred in sector 59 had a value of `2459912.8314` .
  - The t_c of the model fit transit on 20241227 was at `2460672.878957585` .
  - These two t_c(s) have 157 periods in between, which covers 2.08 years. This close-to 2-year time span works well in averaging the sinusoidal Romer effect.
  - `p_avg = (2460672.878957585 - 2459912.8314) / 157 = 4.841067245763958`
- `p_real_obs_20241025_20241227` : the real observed period between 20241025 to 20241227 (13 periods)
  - `p_real_obs_20241025_20241227 = (2460672.878957585 - 2460609.941884418) / 13 = 4.841313320534447`

- `p_theo_obs_xxx` : the calculated period at different times with the corrections from the Doppler Effect

```
2024-09-26
2024-09-26 - 2024-10-25: v_earth: 25.3 km/sec,
p_theo_obs: 4.840659002620029
2024-10-25 - 2024-12-27: v_earth: 6.73 km/sec,
p_theo_obs: 4.840958642089622
2022-11-29 - 2022-12-24: v_earth: -0.807 km/sec,
p_theo_obs: 4.84108028484455
2023-12-11 - 2024-01-04: v_earth: -6.48 km/sec,
p_theo_obs: 4.841171810075927
```

- Note, each 0.0001 BJE is 8.64 seconds

Analysis

- The observations and the Earth Movement diagram
- By comparing the calculated result, we found that the `p_real_obs_20241025_20241227` is greater than the `p_avg` by 21.26 sec.

Discussion

- The above difference is small. Is it significant?
- The p_avg is quite accurate, as it is an average of 157 periods.
- The AstroImageJ does not provide an error for the calculated values. I wonder how accurate the data is?

In [ ]:
```python
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
tc_df = pd.DataFrame(tc_list)
tc_df['pred_59'] = (tc_df['n'] - tc_df['n'].iloc[0])* tc_df['period'].iloc[0
tc_df['pred_73'] = (tc_df['n'] - tc_df['n'].iloc[1])* tc_df['period'].iloc[1
tc_df['pred_59_diff'] = tc_df['tc_val'] - tc_df['pred_59']
tc_df['pred_73_diff'] = tc_df['tc_val'] - tc_df['pred_73']
tc_df['rel_bjd_59'] = tc_df['tc_val'] - tc_df['tc_val'].iloc[0]

tc_df
```

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

def extend(X, Y, x_min, x_max):
    slope = (Y[1] - Y[0]) / (X[1] - X[0])  # (y2 - y1) / (x2 - x1)
    intercept = Y[0] - slope * X[0]  # y = mx + b -> b = y1 - m*x1

    # Extend the x range beyond the endpoints
    x_extended = np.linspace(x_min, x_max, 2)  # Extend range by 2 units on
    y_extended = slope * x_extended + intercept  # Calculate y values for th
```

```
    return x_extended, y_extended
```

In [ ]:
```
from sklearn.metrics import r2_score

# Fit using sec_59, sec_73, 20241025, 20241227 data
X = tc_df[['n']].iloc[[i for i in range(len(tc_df)) if i in[0, 1, 3, 4]]]  #
y = tc_df['tc_val'].iloc[[i for i in range(len(tc_df)) if i in[0, 1, 3, 4]]]

# Fit using sec_59, sec_73, 20241227 data
#X = tc_df[['n']].iloc[[i for i in range(len(tc_df)) if i not in[2, 3]]]  #
#y = tc_df['tc_val'].iloc[[i for i in range(len(tc_df)) if i not in[2, 3]]]

# Fit using sec_59, sec_73 data
#X = tc_df[['n']].iloc[:2]  # Independent variable (must be 2D for scikit-le
#y = tc_df['tc_val'].iloc[:2]    # Dependent variable

# Fit using sec_59, 20241227 data
#X = tc_df[['n']].iloc[0::4]  # Independent variable (must be 2D for scikit-
#y = tc_df['tc_val'].iloc[0::4]    # Dependent variable

model = LinearRegression()
model.fit(X, y)
# Coefficients and intercept
slope = model.coef_[0]  # The slope (m)
intercept = model.intercept_  # The intercept (b)

fitted_slope = slope

# Predict using the fitted model
tc_df['y_pred'] = model.predict(tc_df[['n']])


# The diff between the actual and the model
tc_df['diff'] = tc_df['tc_val'] - tc_df['y_pred']

# Output results
print(f"Slope (m): {slope}")
print(f"Intercept (b): {intercept}")
print("r^2:", r2_score(tc_df['tc_val'], tc_df['y_pred']))

print("Fitted DataFrame:")
# print(tc_df)

plt.plot(tc_df['n'], tc_df['diff'] * 24 * 60, marker='o', linestyle='-', col
plt.xlabel('Period Num')  # Optional: Label for the x-axis
plt.ylabel(f"Tc_obs - Tc_pred (min)")  # Optional: Label for the y-axis
```

The Accuracy of the Calculated Periods of Sector 59 and 73

- When we plot the predicated Tc of the sector 59 and 73 against the observed Tc, we notice a linear trend in each case. The trends indicate that the sector 59 prediction overestimates the real period, while the sector 73 underestimates the real period. Note that the Tc shifts we see in our AIJ analysis are caused the result of the sector 73 prediction, which is also what the Swarthmore Transit Finder uses.

- It's not surprising the results of both TESS sectors are not accurate. The TESS sector predictions have been based on a short observation of less than six successive periods. When the observation range is extended to two years, a small inaccuracy in the period can be accumulated to a significant shift.

A More Accurate Estimated Period - P_accurate

- Given our current longer horizon observation data, we can estimate this by fitting a linear model.
- When choosing what data to include in the fitting, we should exlcude the incomplete 20240926 transit observation.
- With Sector 59, 73, 20241025, and 20241227 data, the fitted result is as below:
```
Slope (m): 4.841059536739922
Intercept (b): 2459912.8307693945
r^2: 0.9999999999712603
```
- Because this fitted time range covers very close to 2 whole years (2.08 years), this period is an average period over the sinugolidal Romer effect due to the period fluctuation caused by the Earth revolving aroud the Sun.

Sinusoidal Romer Effect Analysis

- With the above `P_accurate`, we can apply it to the Sinusoidal Romer Effect analysis describe in the previous section. This `P_accurate` is slightly different from the estimated period using only the sector 59 and 20241227 data, but it would arrive to the same colusion as before.

In [50]:
```
from matplotlib.ticker import FuncFormatter

def custom_format(y, pos):
    return f'{y:.2f}'  # Format to 2 decimal places
scalar_uint = "hour"
scalar = 24
model_59 = LinearRegression()
model_59.fit(tc_df[['rel_bjd_59']], tc_df['pred_59_diff'])
y_pred_59 = model_59.predict(tc_df[['rel_bjd_59']])

model_73 = LinearRegression()
model_73.fit(tc_df[['rel_bjd_59']], tc_df['pred_73_diff'])
y_pred_73 = model_73.predict(tc_df[['rel_bjd_59']])

model_fitted = LinearRegression()
model_fitted.fit(tc_df[['rel_bjd_59']], tc_df['diff'])
y_pred_fitted = model_fitted.predict(tc_df[['rel_bjd_59']])
```

```
fig, axes = plt.subplots(2, 1, figsize=(5, 5), sharex="col", gridspec_kw={'h

axes[0].set_title("Linear Drift of Observed Tc vs Prediction Caused by\nthe 
axes[0].scatter(tc_df['rel_bjd_59'], tc_df['pred_59_diff'] * scalar, label='
axes[0].scatter(tc_df['rel_bjd_59'], tc_df['pred_73_diff'] * scalar, label='
axes[0].plot(tc_df['rel_bjd_59'], y_pred_59 * scalar)
axes[0].plot(tc_df['rel_bjd_59'], y_pred_73 * scalar)
axes[0].set_ylim(-2.5, 2.5)
axes[0].set_ylabel(f"Tc_obs - Tc_pred ({scalar_uint})")


axes[1].plot(tc_df['rel_bjd_59'], y_pred_fitted * scalar, c='green')
axes[1].scatter(tc_df['rel_bjd_59'], tc_df['diff'] * scalar, c='green', labe
axes[1].set_ylim(-0.1, 0.1)
axes[1].set_ylabel(f"Tc_obs - Tc_pred ({scalar_uint})")
axes[1].set_xlabel("BJD_obs - BJD_sec_59")

for ax in axes:
    ax.yaxis.set_major_formatter(FuncFormatter(custom_format))
    ax.grid()
    ax.legend(loc='upper left')

plt.tight_layout()
```