Übung 01

Prognosen und Exponentielles Glätten

Aufgabe 1: Zerlegung einer Zeitreihe

Eine traditionsreiche Manufaktur aus Essen möchte den Verkauf ihrer berühmten Torten besser verstehen, um Zutatenbestellungen und Personalplanung zu optimieren. Die Verkaufszahlen der letzten zwei Jahre (in Stück pro Monat) sind wie folgt:

Monat	Jahr 1	Jahr 2
Januar	80	95
Februar	75	90
März	90	105
April	110	125
Mai	130	150
Juni	150	175
Juli	160	190
August	155	180
September	120	140
Oktober	100	115
November	85	100
Dezember	100	120

Ihre Aufgaben:

1. Stellen Sie die Zeitreihe grafisch dar (eine einfache Skizze auf Papier genügt).
2. Beschreiben Sie die Hauptkomponenten (Trend, Saison, Zyklus, irreguläre Schwankungen), die Sie in den Verkaufszahlen vermuten. Wie würden Sie diese qualitativ charakterisieren? (z.B. steigender Trend, saisonale Spitzen im Sommer und zu Weihnachten).
3. Skizzieren Sie, wie Sie vorgehen würden, um die Trend- und Saisonkomponenten grob zu schätzen, basierend auf den im Skript vorgestellten Ideen.

Lösungshinweise:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

plt.rcdefaults()
plt.style.use('tableau-colorblind10')
```

```
data_task1 = {
    'Monat': ['Jan', 'Feb', 'Mär', 'Apr', 'Mai', 'Jun', 'Jul', 'Aug',
'Sep', 'Okt', 'Nov', 'Dez'] * 2,
    'Jahr': [1]*12 + [2]*12,
    'Verkauf': [80, 75, 90, 110, 130, 150, 160, 155, 120, 100, 85, 100,
                95, 90, 105, 125, 150, 175, 190, 180, 140, 115, 100, 120]
}
df_task1 = pd.DataFrame(data_task1)
df_task1['Periode'] = range(1, len(df_task1) + 1)

plt.figure(figsize=(7, 5))
plt.plot(df_task1['Periode'], df_task1['Verkauf'], marker='o',
linestyle='-')
plt.title('Verkauf Torten')
plt.xlabel('Periode (Monat)')
plt.ylabel('Verkaufte Stück')
plt.xticks(df_task1['Periode'], [f"{m}-{j}" for m,j in
zip(df_task1['Monat'], df_task1['Jahr'])], rotation=45, ha="right")
plt.grid(True)
plt.tight_layout()
plt.show()
```

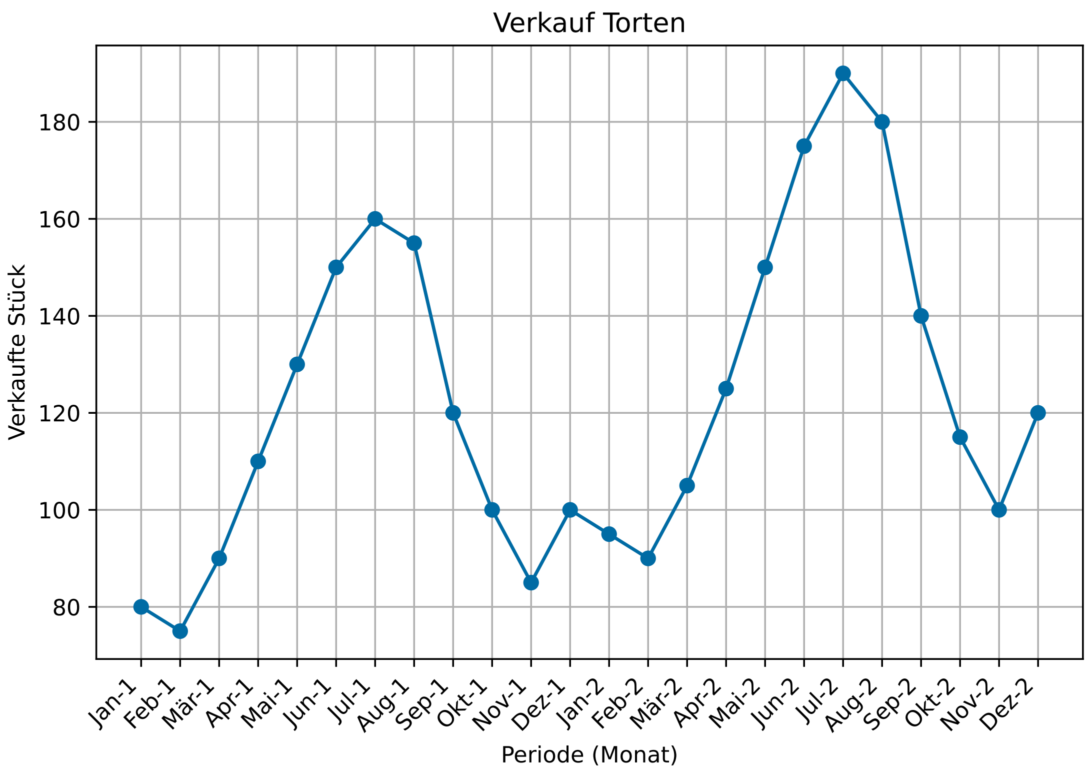


Figure 1: Verkauf Schwarzwälder Kirschtorte

Qualitative Beschreibung der Komponenten: - Trend (T): Es scheint einen ansteigenden Trend zu geben, da die Verkäufe im zweiten Jahr generell höher sind als im ersten. - Saison (S): Es gibt klare saisonale Schwankungen. Die Verkäufe sind in den Sommermonaten (Juni-August) tendenziell höher. Auch im Frühjahr (z.B. April/Mai) und im Dezember (Weihnachtsgeschäft) gibt es Spitzen. Die geringsten Verkäufe sind zu Jahresbeginn. - Zyklus (C): Mit nur zwei Jahren Daten ist es schwierig, eine mittelfristige zyklische Komponente (z.B. Konjunkturzyklen) zuverlässig zu identifizieren. - Irreguläre Schwankungen (I): Es gibt monatliche Schwankungen, die nicht perfekt durch Trend und Saison erklärt werden können und als zufällige Restschwankungen interpretiert werden.

Für eine quantitative Analyse (optional, nicht von Hand im Detail gefordert): Man könnte z.B. einen gleitenden Durchschnitt berechnen (z.B. 12-Monate MA, zentriert), um die glatte Komponente T+C zu schätzen. Anschließend könnte man die Verhältnisse der Originaldaten zum gleitenden Durchschnitt berechnen (Y/(T+C) = S*I), um saisonale Faktoren S zu isolieren und zu mitteln.

Aufgabe 2: Prognose ohne Trend

Eine 3D-Druck Firma aus Duisburg möchte die Nachfrage nach seinen 3D-Druckteilen für den nächsten Tag vorhersagen, um Überproduktion oder Engpässe zu vermeiden. Er hat die Verkaufszahlen der letzten 10 Tage notiert:

Tag	Verkaufte 3D-Druckteile (y_t)
1	120
2	125
3	115
4	122
5	118
6	128
7	123
8	119
9	126
10	124

Die Firma geht davon aus, dass der Verkauf relativ konstant ist, aber täglichen Schwankungen unterliegt (d.h. kein klarer Trend, konstantes Niveau).

Ihre Aufgaben:

1. Berechnen Sie einen gleitenden Durchschnitt der Ordnung $n = 3$ (3-Tage-Linie), um eine Prognose für Tag 11 zu erstellen (p_{11}). Der Prognosewert für Tag $t + 1$ ist der Durchschnittswert zum Zeitpunkt t.
2. Wenden Sie die exponentielle Glättung erster Ordnung an, um eine Prognose für Tag 11 zu erstellen. Verwenden Sie einen Glättungsfaktor $\alpha = 0.2$. Als Startwert für den geglätteten Wert zum Zeitpunkt $t = 0$ (Prognose für Tag 1, p_1) nehmen Sie den tatsächlichen Verkauf von Tag 1 (y_1).
3. Welche Prognose erscheint Ihnen intuitiv plausibler? Begründen Sie kurz.

4. Berechnen Sie den Prognosefehler mit der mittleren absoluten Abweichung für die letzten 4 Tage für beide Verfahren.

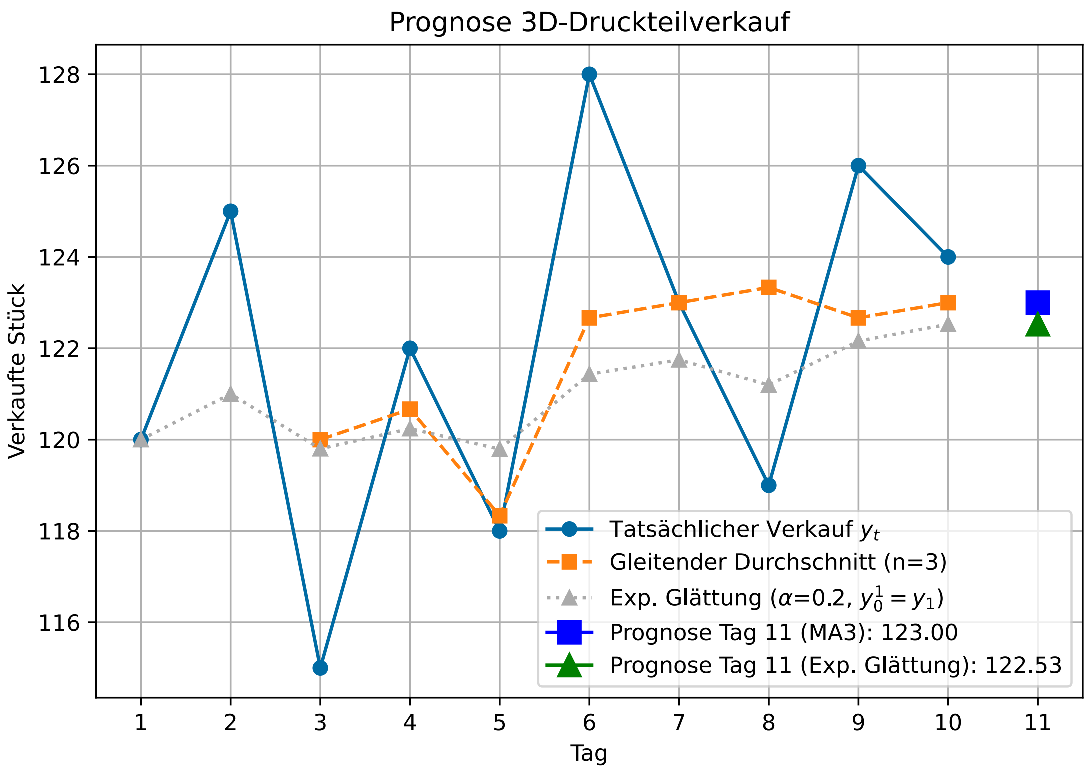
Lösungshinweise:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

plt.rcdefaults()
plt.style.use('tableau-colorblind10')

data_task2 = {'Tag': range(1, 11), 'Verkauf': [120, 125, 115, 122, 118,
128, 123, 119, 126, 124]}
df_task2 = pd.DataFrame(data_task2).set_index('Tag')

# 1. Gleitender Durchschnitt n=3
n_ma = 3
df_task2['MA3'] = df_task2['Verkauf'].rolling(window=n_ma).mean()
# Die Prognose für Tag 11 ist der MA3-Wert von Tag 10.
# MA3_10 = (y_10 + y_9 + y_8) / 3
prognose_ma_tag11 = (df_task2['Verkauf'].loc[10] +
df_task2['Verkauf'].loc[9] + df_task2['Verkauf'].loc[8]) / 3

print(f"\nTabelle mit 3-Tage gleitendem Durchschnitt:")
print(df_task2)
print(f"Prognose für Tag 11 (Gleitender Durchschnitt n=3):
{prognose_ma_tag11:.2f}")

# 2. Exponentielle Glättung erster Ordnung
alpha = 0.2
y_actual_t2 = df_task2['Verkauf'].tolist()
y_smooth_t2 = [0.0] * len(y_actual_t2)

# Initialisierung: y_0^(1) = y_1 = 120
y0_smooth_t2 = y_actual_t2[0]

# y_1^(1) = alpha * y_1 + (1-alpha) * y_0^(1)
y_smooth_t2[0] = alpha * y_actual_t2[0] + (1 - alpha) * y0_smooth_t2

for t in range(1, len(y_actual_t2)): # t_index von 1 (für y_2) bis 9 (für
y_10)
    # y_t+1^(1) = alpha * y_t+1 + (1-alpha) * y_t^(1)
    y_smooth_t2[t] = alpha * y_actual_t2[t] + (1 - alpha) *
y_smooth_t2[t-1]

df_task2['ExpGlättung (y_t^(1))'] = y_smooth_t2
prognose_exp_tag11 = y_smooth_t2[-1] # y_10^(1)

print(f"\nTabelle mit exponentieller Glättung (alpha=0.2, y_0^(1) = y_1):")
# Manuelle Berechnungsschritte für Studenten wären:
# y_0^(1) = 120
# y_1^(1) = 0.2*120 + 0.8*120 = 120
```

```
# y_2^(1) = 0.2*125 + 0.8*120 = 25 + 96 = 121
# y_3^(1) = 0.2*115 + 0.8*121 = 23 + 96.8 = 119.8
# ...
print(df_task2)
print(f"Prognose für Tag 11 (Exponentielle Glättung, alpha=0.2):
{prognose_exp_tag11:.2f}")

# Plot für Aufgabe 2
plt.figure(figsize=(7, 5))
plt.plot(df_task2.index, df_task2['Verkauf'], marker='o', linestyle='-',
label='Tatsächlicher Verkauf $y_t$')
plt.plot(df_task2.index, df_task2['MA3'], marker='s', linestyle='--',
label=f'Gleitender Durchschnitt (n={n_ma})')
plt.plot(df_task2.index, df_task2['ExpGlättung (y_t^(1))'], marker='^',
linestyle=':', label=f'Exp. Glättung ($\\alpha$={alpha}, $y_0^{(1)}=y_1$)')

# Prognosepunkte
plt.plot(11, prognose_ma_tag11, marker='s', color='blue', markersize=10,
label=f'Prognose Tag 11 (MA{n_ma}): {prognose_ma_tag11:.2f}')
plt.plot(11, prognose_exp_tag11, marker='^', color='green', markersize=10,
label=f'Prognose Tag 11 (Exp. Glättung): {prognose_exp_tag11:.2f}')

plt.title('Prognose 3D-Druckteilverkauf')
plt.xlabel('Tag')
plt.ylabel('Verkaufte Stück')
plt.xticks(list(range(1, 11)) + [11]) # Ensure Tag 11 is shown
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
Tabelle mit 3-Tage gleitendem Durchschnitt:
     Verkauf         MA3
Tag
1        120         NaN
2        125         NaN
3        115  120.000000
4        122  120.666667
5        118  118.333333
6        128  122.666667
7        123  123.000000
8        119  123.333333
9        126  122.666667
10       124  123.000000
Prognose für Tag 11 (Gleitender Durchschnitt n=3): 123.00

Tabelle mit exponentieller Glättung (alpha=0.2, y_0^(1) = y_1):
     Verkauf         MA3  ExpGlättung (y_t^(1))
Tag
1        120         NaN             120.000000
```

```
2         125         NaN            121.000000
3         115  120.000000            119.800000
4         122  120.666667            120.240000
5         118  118.333333            119.792000
6         128  122.666667            121.433600
7         123  123.000000            121.746880
8         119  123.333333            121.197504
9         126  122.666667            122.158003
10        124  123.000000            122.526403
Prognose für Tag 11 (Exponentielle Glättung, alpha=0.2): 122.53
```



3. Begründung Plausibilität: Der gleitende Durchschnitt (123.00) basiert ausschließlich auf den letzten drei Beobachtungen und gewichtet diese gleich. Die exponentielle Glättung (122.62 mit alpha=0.2) berücksichtigt alle vergangenen Beobachtungen, wobei die jüngsten ein höheres Gewicht erhalten. Ein Alpha von 0.2 bedeutet eine relativ starke Glättung, d.h. die Prognose reagiert eher träge auf neue Werte. Beide Prognosen liegen eng beieinander. Wenn man davon ausgeht, dass die letzten Tage sehr repräsentativ für die nahe Zukunft sind und es keine Ausreißer gab, könnte der MA(3) passend sein. Wenn man eine stabilere, stärker geglättete Prognose bevorzugt, die weniger von einzelnen Ausschlägen beeinflusst wird, ist die exponentielle Glättung mit niedrigem Alpha eine gute Wahl.

Aufgabe 3: Prognose mit Trend

Ein aufstrebender YouTuber hat in den letzten 8 Monaten einen stetigen Zuwachs an neuen Abonnenten verzeichnet. Er möchte die Entwicklung für die nächsten zwei Monate prognostizieren, um seine Content-Strategie anzupassen.

Monat t	Neue Abonnenten y_t
1	500
2	530
3	520
4	580
5	620
6	670
7	640
8	710

Er möchte die Methode der exponentiellen Glättung mit Trendkorrektur verwenden, wie sie im Skript vorgestellt wird. Nutzen Sie einen Glättungsfaktor $\alpha = 0.3$. Das geschätzte Niveau zum Zeitpunkt $t = 0$ ist $\hat{a}_0 = 480$ und der Trend (Steigung) ist $\hat{b}_0 = 25$.

Ihre Aufgaben:

1. Berechnen Sie die initialen Werte $y_0^{(1)}$ und $y_0^{(2)}$.
2. Berechnen Sie iterativ $y_t^{(1)}$, $y_t^{(2)}$, $\hat{a}_t$ und $\hat{b}_t$ für die Monate $t = 1$ bis $t = 8$.
3. Erstellen Sie eine Prognose für die Anzahl neuer Abonnenten für Monat 9 und Monat 10.
4. Berechnen Sie den Prognosefehler mit der mittleren quadratischen Abweichung (MSE) für Monat 7 und Monat 8.

Lösungshinweise:

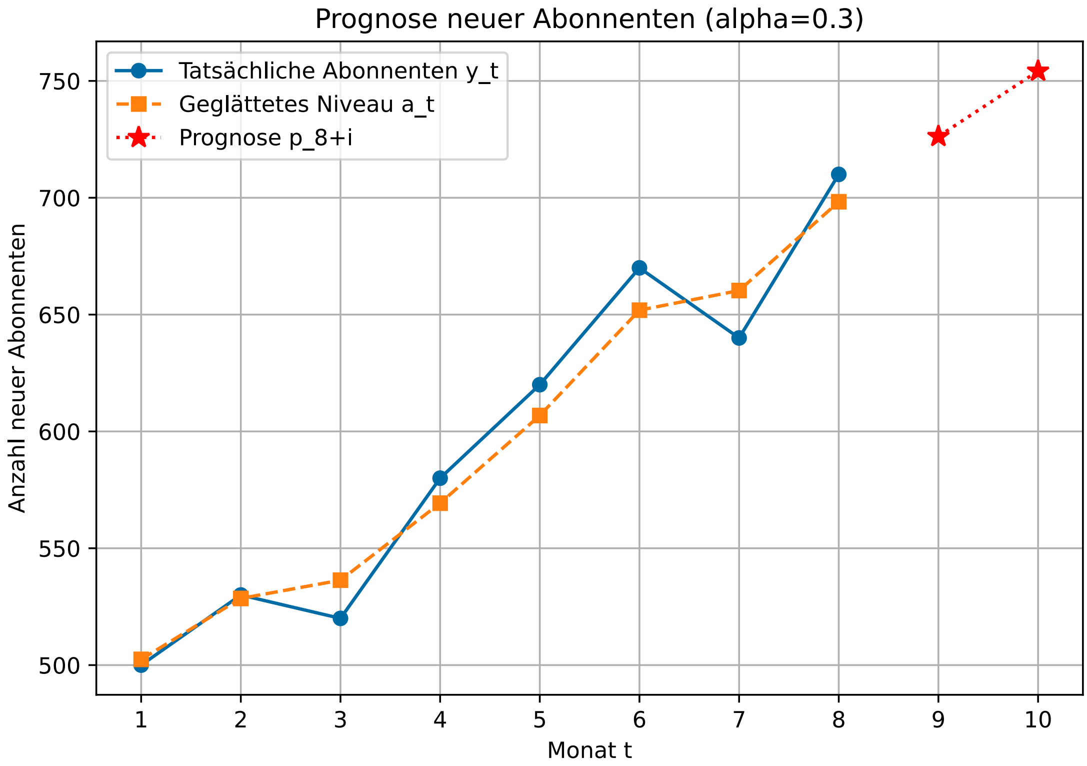
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

plt.rcdefaults()
plt.style.use('tableau-colorblind10')

data_task3 = {'Monat': range(1, 9), 'Abonnenten': [500, 530, 520, 580, 620,
670, 640, 710]}
df_task3 = pd.DataFrame(data_task3)
y_actual_t3 = df_task3['Abonnenten'].tolist()

print("Abonnentendaten TechTom:")
print(df_task3)

alpha_t3 = 0.3
a0_hat = 480
b0_hat = 25

# 1. Initialisierung y_0^(1), y_0^(2)
if alpha_t3 == 0 or alpha_t3 == 1: # Vermeide Division durch Null oder
ungültige (1-alpha)/alpha
    print("Alpha darf nicht 0 oder 1 sein für diese Initialisierung.")
```

```
    y0_1 = np.nan
    y0_2 = np.nan
else:
    y0_1 = a0_hat - b0_hat * (1 - alpha_t3) / alpha_t3
    y0_2 = a0_hat - 2 * b0_hat * (1 - alpha_t3) / alpha_t3

print(f"\nInitialisierung:")
print(f"y_0^(1) = {y0_1:.2f}")
print(f"y_0^(2) = {y0_2:.2f}")

# Arrays für die Speicherung der Werte
y1_vals = [0.0] * (len(y_actual_t3) + 1) # Index 0 für t=0
y2_vals = [0.0] * (len(y_actual_t3) + 1)
a_hat_vals = [0.0] * (len(y_actual_t3) + 1)
b_hat_vals = [0.0] * (len(y_actual_t3) + 1)

y1_vals[0] = y0_1
y2_vals[0] = y0_2

results_t3 = []
print("\n2. Iterative Berechnung für t=1 bis 8:")
print(f"t=0: y1_0={y1_vals[0]:.2f}, y2_0={y2_vals[0]:.2f},
a_hat_0={a0_hat:.2f}, b_hat_0={b0_hat:.2f} (gegeben)")


for t in range(1, len(y_actual_t3) + 1): # t from 1 to 8
    y_t_val = y_actual_t3[t-1] # y_1 ist y_actual_t3[0]

    y1_vals[t] = alpha_t3 * y_t_val + (1 - alpha_t3) * y1_vals[t-1]
    y2_vals[t] = alpha_t3 * y1_vals[t] + (1 - alpha_t3) * y2_vals[t-1]

    a_hat_vals[t] = 2 * y1_vals[t] - y2_vals[t]
    if alpha_t3 == 1: # Vermeidung Division durch Null
         b_hat_vals[t] = float('inf') if (y1_vals[t] - y2_vals[t]) != 0
else 0 # Sonderfall
    else:
        b_hat_vals[t] = (alpha_t3 / (1 - alpha_t3)) * (y1_vals[t] -
y2_vals[t])

    results_t3.append({
        't': t, 'y_t': y_t_val,
        'y1_t': y1_vals[t], 'y2_t': y2_vals[t],
        'a_hat_t': a_hat_vals[t], 'b_hat_t': b_hat_vals[t]
    })

df_results_t3 = pd.DataFrame(results_t3)

print("\nÜbersicht der Berechnungen:")
print(df_results_t3.round(2))

# 3. Prognose für Monat 9 und 10
# Basierend auf a_hat_8 und b_hat_8
```

```
a_hat_final = a_hat_vals[len(y_actual_t3)] # a_hat_8
b_hat_final = b_hat_vals[len(y_actual_t3)] # b_hat_8

prognose_monat9 = a_hat_final + b_hat_final * 1
prognose_monat10 = a_hat_final + b_hat_final * 2

print(f"\nLetzte Parameter (t=8):")
print(f"a_hat_8 = {a_hat_final:.2f}")
print(f"b_hat_8 = {b_hat_final:.2f}")
print(f"\nPrognose für Monat 9 (p_8+1): {prognose_monat9:.2f}")
print(f"Prognose für Monat 10 (p_8+2): {prognose_monat10:.2f}")

# Plot für Aufgabe 3
plt.figure(figsize=(7, 5))
# Tatsächliche Werte
plt.plot(df_task3['Monat'], df_task3['Abonnenten'], marker='o',
linestyle='-', label='Tatsächliche Abonnenten y_t')

# Geglättetes Niveau a_hat_t (ab t=1)
# df_results_t3 contains a_hat_t for t=1 to 8
plt.plot(df_results_t3['t'], df_results_t3['a_hat_t'], marker='s',
linestyle='--', label='Geglättetes Niveau a_t')

# Prognosewerte
forecast_months = [len(y_actual_t3) + 1, len(y_actual_t3) + 2]
forecast_values = [prognose_monat9, prognose_monat10]
plt.plot(forecast_months, forecast_values, marker='*', linestyle=':',
markersize=10, color='red', label='Prognose p_8+i')

# Optional: Plot der Trendlinie basierend auf finalen Parametern
# trend_line_x = np.array(range(1, len(y_actual_t3) + 3)) # Monate 1 bis 10
# trend_line_y = a_hat_final + b_hat_final * (trend_line_x -
len(y_actual_t3)) # anpassen, um bei a_hat_8 zu starten
# plt.plot(trend_line_x, trend_line_y, linestyle='-.', color='purple',
label=f'Trendlinie (ab $t=8$: $\widehat{{a}}_8 + \widehat{{b}}_8 \cdot
i$)')
# Für die Darstellung der Aufgabe ist es oft klarer, die konkreten
Prognosepunkte zu zeigen.
# Die a_hat_t Linie zeigt bereits die Entwicklung des Niveaus.

plt.title(f'Prognose neuer Abonnenten (alpha={alpha_t3})')
plt.xlabel('Monat t')
plt.ylabel('Anzahl neuer Abonnenten')
plt.xticks(list(range(1, 9)) + forecast_months)
plt.grid(True)
plt.tight_layout()
plt.legend()
plt.show()
```

```
Abonnentendaten TechTom:
   Monat  Abonnenten
```

```
0      1         500
1      2         530
2      3         520
3      4         580
4      5         620
5      6         670
6      7         640
7      8         710

Initialisierung:
y_0^(1) = 421.67
y_0^(2) = 363.33

2. Iterative Berechnung für t=1 bis 8:
t=0: y1_0=421.67, y2_0=363.33, a_hat_0=480.00, b_hat_0=25.00 (gegeben)

Übersicht der Berechnungen:
   t  y_t    y1_t    y2_t  a_hat_t  b_hat_t
0  1  500  445.17  387.88   502.45    24.55
1  2  530  470.62  412.70   528.53    24.82
2  3  520  485.43  434.52   536.34    21.82
3  4  580  513.80  458.31   569.30    23.78
4  5  620  545.66  484.51   606.81    26.21
5  6  670  582.96  514.05   651.88    29.54
6  7  640  600.07  539.86   660.29    25.81
7  8  710  633.05  567.81   698.29    27.96

Letzte Parameter (t=8):
a_hat_8 = 698.29
b_hat_8 = 27.96

Prognose für Monat 9 (p_8+1): 726.25
Prognose für Monat 10 (p_8+2): 754.21
```



Formeln:

- Initialisierung:
  - $y_0^{(1)} = \hat{a}_0 - \hat{b}_0 \cdot \frac{1-\alpha}{\alpha}$
  - $y_0^{(2)} = \hat{a}_0 - 2 \cdot \hat{b}_0 \cdot \frac{1-\alpha}{\alpha}$
- Aktualisierung der gleitenden Durchschnitte (für $t = 1, ..., 8$):
  - $y_t^{(1)} = \alpha \cdot y_t + (1-\alpha) \cdot y_{t-1}^{(1)}$
  - $y_t^{(2)} = \alpha \cdot y_t^{(1)} + (1-\alpha) \cdot y_{t-1}^{(2)}$
- Aktualisierung der Parameter der Trendgeraden (für $t = 1, ..., 8$):
  - $\hat{a}_t = 2 \cdot y_t^{(1)} - y_t^{(2)}$
  - $\hat{b}_t = \frac{\alpha}{1-\alpha} \cdot \left(y_t^{(1)} - y_t^{(2)}\right)$
- Prognosewert (basierend auf Werten von $t = 8$):
  - $p_{8+i} = \hat{a}_8 + \hat{b}_8 \cdot i$