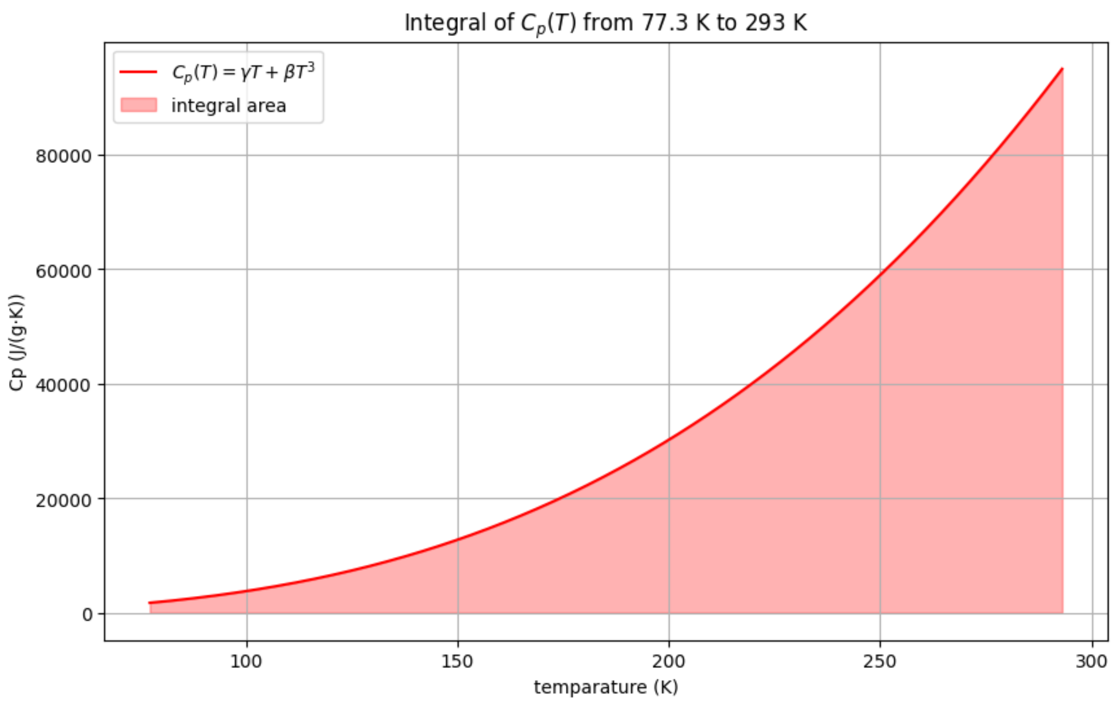
homework9

2025 年 4 月 7 日

[23]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import quad
def copper_specific_heat(T):
    gamma = 2.9
    beta = 0.24
    return gamma * T + beta * T**3
result, error = quad(copper_specific_heat, 77.3, 293)

T = np.linspace(77.3, 293, 500)
Cp = copper_specific_heat(T)

# 转换单位为 J/g*K
Cp_JgK = Cp / 63.546
plt.figure(figsize=(10, 6))
plt.plot(T, Cp_JgK, label=r'$C_p(T) = \gamma T + \beta T^3$', color='red')
plt.fill_between(T, Cp_JgK, alpha=0.3, color='red', label='integral area')
plt.xlabel('temparature (K)')
plt.ylabel('Cp (J/(g·K))')
plt.title(r'Integral of $C_p(T)$ from 77.3 K to 293 K')
plt.legend()
plt.grid(True)
plt.show()
print(f"积分结果: {result/63.546:.4f} J/(g·K)")
```



```
积分结果: 6926897.3579 J/(g·K)
```

1 需要先更新 nbformat 版本才能运行下方的代码:

[ ]:
```
from random import randint
class Die:
 """A class representing a single die."""
 def __init__(self, num_sides):
  self.num_sides = num_sides
 def roll(self):

  return randint(1, self.num_sides)

import plotly

die_1 = Die(9)
die_2 = Die(9)
die_3 = Die(9)
```

```
results = []
for roll_num in range(50000):
 result = die_1.roll() + die_2.roll()+ die_3.roll()
 results.append(result)
frequencies = []
max_result = die_1.num_sides + die_2.num_sides+ die_3.num_sides
poss_results = range(3, max_result+1)
for value in poss_results:
 frequency = results.count(value)
 frequencies.append(frequency)
title = "Results of Rolling three 9-sides-die 50,000 Times"
labels = {'x': 'Result', 'y': 'Frequency of Result'}
fig = plotly.express.bar(x=poss_results, y=frequencies, title=title,␣
 ↪labels=labels)
fig.update_layout(xaxis_dtick=1)
fig.show()
plotly.offline.plot(fig, filename='die_rolls.html', auto_open=True)
```