

In [2]: ## Niraj Neupane

```
# pip install yfinance pandas numpy matplotlib scipy

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import yfinance as yf

# -----
# 1) Download market proxies
# -----
START = "2020-01-01"
tickers = {
    "SPY": "SPY",      # equity
    "VIX": "^VIX",     # vol
    "TLT": "TLT",      # long rates proxy
    "HYG": "HYG",      # high yield
    "LQD": "LQD"       # investment grade
}

px = yf.download(list(tickers.values()), start=START, auto_adjust=True)[["Close"]]
px.columns = list(tickers.keys())
px = px.dropna()

rets = px.pct_change().dropna()

# Loss series (positive = worse)
loss = pd.DataFrame(index=rets.index)
loss["Equity_Loss"] = -rets["SPY"]
loss["Rates_Loss"] = -rets["TLT"]
loss["Credit_SpreadProxy_Loss"] = -(rets["HYG"] - rets["LQD"]) # HY underperforms
loss = loss.dropna()

# -----
# 2) Entropy tilting functions (Skoglund-style)
# -----
def tilted_weights(y, theta):
    y = np.asarray(y, dtype=float)
    z = theta * y
    z = z - np.max(z)           # numerical stability
    w = np.exp(z)
    return w / w.sum()

def tilted_expectation(y, theta):
    w = tilted_weights(y, theta)
    return float((w * y).sum())

# -----
# 3) Compute tilted Loss vs theta
# -----
theta_grid = np.linspace(0, 10, 51)
series = "Credit_SpreadProxy_Loss"      # main paper-aligned proxy (stress Losses)
y = loss[series].values
```

```

tilted_curve = np.array([tilted_expectation(y, th) for th in theta_grid])
base_mean = y.mean()
tail_99 = np.quantile(y, 0.99)

# -----
# 4) Stress window co-moves (empirical validation)
# -----
worst10 = loss["Equity_Loss"].sort_values(ascending=False).head(10).index
stress_comove = loss.loc[worst10].mean().sort_values(ascending=False)

# -----
# 5) Rolling correlation (fixed, no xs)
# -----
window = 63
roll_corr_tlt = rets["SPY"].rolling(window).corr(rets["TLT"])
roll_corr_hyg = rets["SPY"].rolling(window).corr(rets["HYG"])
roll_corr_lqd = rets["SPY"].rolling(window).corr(rets["LQD"])

# -----
# 6) CHARTS (what you need)
# -----


# Chart A: Base vs Tail vs Entropy-Tilted Expected Loss (core paper validation)
plt.figure()
plt.plot(theta_grid, tilted_curve, marker="o", markersize=3, label="Entropy-tilted")
plt.axhline(base_mean, linestyle="--", label="Base mean loss")
plt.axhline(tail_99, linestyle="--", label="Historical 99% loss")
plt.title(f"Entropy-Tilted Expected Loss vs Theta - {series}")
plt.xlabel("Theta (robustness / entropy budget)")
plt.ylabel("Loss")
plt.legend()
plt.show()

# Chart B: Loss distribution (shows tail mass the tilt will emphasize)
plt.figure()
plt.hist(y, bins=80, density=True)
plt.title(f"Loss Distribution - {series}")
plt.xlabel("Loss (positive = worse)")
plt.ylabel("Density")
plt.show()

# Chart C: Stress co-moves table as a bar chart (State Street MRM-friendly)
plt.figure()
stress_comove.plot(kind="bar")
plt.title("Average Losses During Worst 10 Equity Stress Days (Co-moves)")
plt.xlabel("Risk proxy")
plt.ylabel("Average loss")
plt.show()

# Chart D: Rolling correlations vs SPY (regime shift / robustness narrative)
plt.figure()
plt.plot(roll_corr_tlt.index, roll_corr_tlt, label="TLT vs SPY")
plt.plot(roll_corr_hyg.index, roll_corr_hyg, label="HYG vs SPY")
plt.plot(roll_corr_lqd.index, roll_corr_lqd, label="LQD vs SPY")
plt.title(f"Rolling {window}d Correlation vs SPY (Regime Shifts)")

```

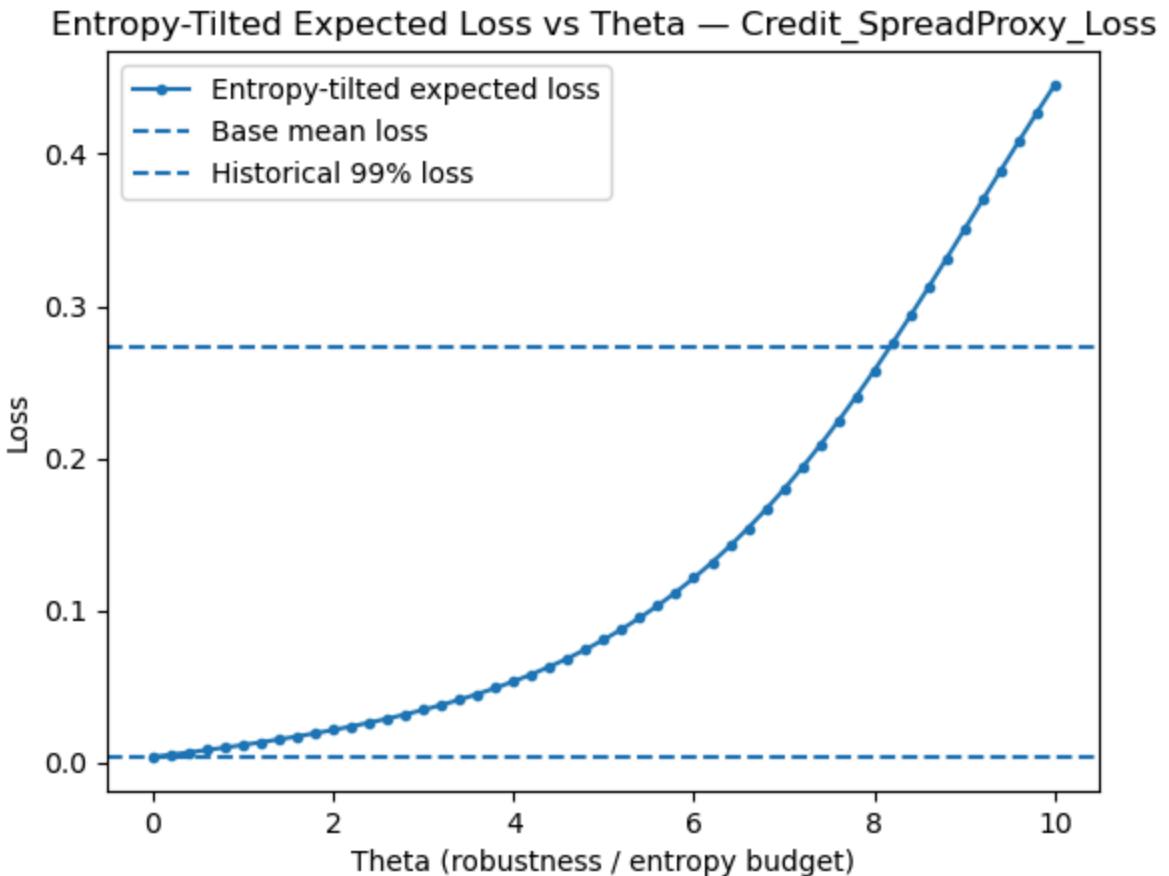
```

plt.xlabel("Date")
plt.ylabel("Correlation")
plt.legend()
plt.show()

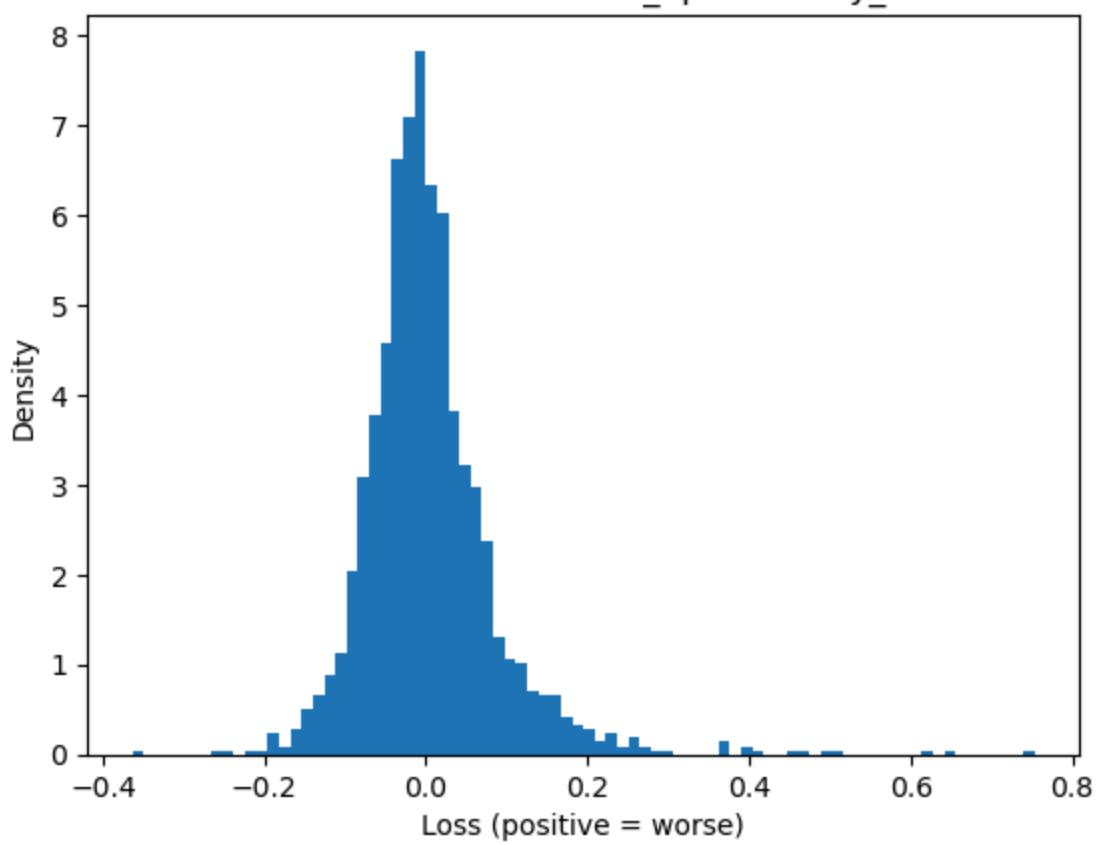
# Print key numbers for your memo later
print("Base mean loss:", round(base_mean, 6))
print("Historical 99% loss:", round(tail_99, 6))
print("Tilted loss at theta=6:", round(tilted_expectation(y, 6.0), 6))
print("\nStress co-moves (avg losses on worst 10 SPY stress days):\n", stress_comov

```

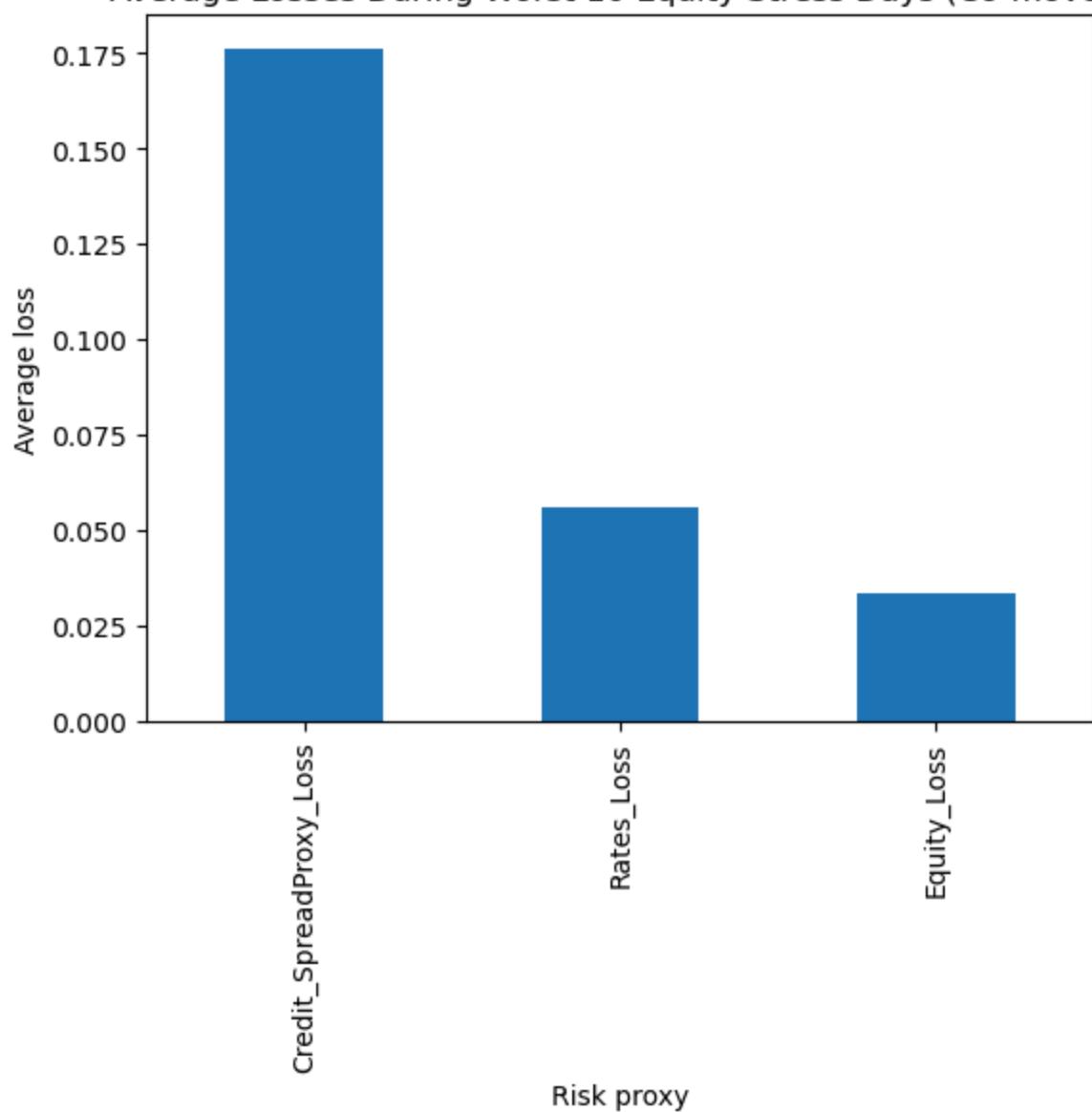
[*****100*****] 5 of 5 completed

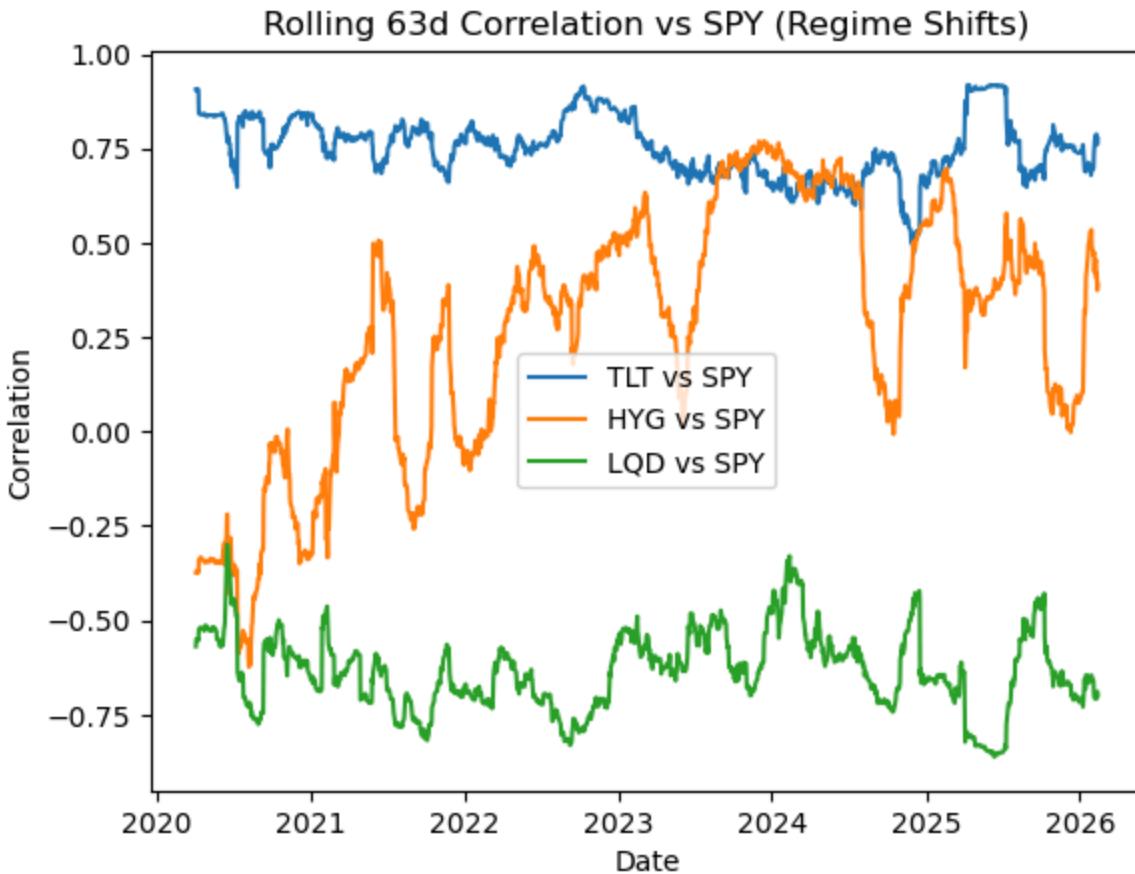


Loss Distribution — Credit_SpreadProxy_Loss



Average Losses During Worst 10 Equity Stress Days (Co-moves)





Base mean loss: 0.0037

Historical 99% loss: 0.272934

Tilted loss at theta=6: 0.121342

Stress co-moves (avg losses on worst 10 SPY stress days):

Credit_SpreadProxy_Loss 0.176042

Rates_Loss 0.055890

Equity_Loss 0.033630

dtype: float64

```
In [3]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

series = "Credit_SpreadProxy_Loss" # focus series
y = loss[series].dropna().values

theta_grid = np.linspace(0, 15, 76) # 0..15
tilted_curve = np.array([tilted_expectation(y, th) for th in theta_grid])

# Numerical derivative (fragility)
dL = np.gradient(tilted_curve, theta_grid)

plt.figure()
plt.plot(theta_grid, tilted_curve)
plt.title(f"Entropy-Tilted Expected Loss vs Theta (0-15) - {series}")
plt.xlabel("Theta")
plt.ylabel("Tilted expected loss")
plt.show()
```

```

plt.figure()
plt.plot(theta_grid, dL)
plt.title(f"Frailty:  $d(\text{Tilted Loss})/d\Theta - \{\text{series}\}$ ")
plt.xlabel("Theta")
plt.ylabel("Marginal sensitivity")
plt.show()

print("Tilted loss @theta=6:", tilted_expectation(y, 6.0))
print("Tilted loss @theta=10:", tilted_expectation(y, 10.0))
print("Max  $d\text{Loss}/d\Theta$ :", dL.max())

alpha = 0.99
y = loss[series].dropna().values

VaR = np.quantile(y, alpha)
CVaR = y[y >= VaR].mean() # Expected shortfall beyond VaR

theta_list = [2, 4, 6, 8, 10]
entropy_losses = {th: tilted_expectation(y, th) for th in theta_list}

print(f"Base mean loss: {y.mean():.6f}")
print(f"VaR {int(alpha*100)}%: {VaR:.6f}")
print(f"CVaR {int(alpha*100)}%: {CVaR:.6f}")
print("Entropy losses:", {k: round(v,6) for k,v in entropy_losses.items()})

# Chart: compare on one plot
plt.figure()
plt.axhline(y.mean(), linestyle="--", label="Base mean")
plt.axhline(VaR, linestyle="--", label=f"VaR {int(alpha*100)}%")
plt.axhline(CVaR, linestyle="--", label=f"CVaR {int(alpha*100)}%")
plt.plot(list(entropy_losses.keys()), list(entropy_losses.values()), marker="o", label="Entropy vs VaR vs CVaR - {series}")
plt.title(f"Entropy vs VaR vs CVaR - {series}")
plt.xlabel("Theta")
plt.ylabel("Loss")
plt.legend()
plt.show()

def subsample_stats(loss_df, series, start, end, theta=6.0, alpha=0.99):
    sub = loss_df.loc[start:end, series].dropna().values
    if len(sub) < 50:
        return None
    VaR = np.quantile(sub, alpha)
    CVaR = sub[sub >= VaR].mean()
    return {
        "start": start,
        "end": end,
        "n": len(sub),
        "mean": sub.mean(),
        "VaR99": VaR,
        "CVaR99": CVaR,
        f"Entropy(theta={theta})": tilted_expectation(sub, theta)
    }

periods = [
    ("2020-01-01", "2021-12-31"),
]

```

```

        ("2022-01-01", "2023-12-31"),
        ("2024-01-01", "2025-12-31"),
    ]

rows = []
for s,e in periods:
    out = subsample_stats(loss, series, s, e, theta=6.0)
    if out:
        rows.append(out)

stab = pd.DataFrame(rows)
stab

plt.figure()
plt.plot(stab["start"], stab["VaR99"], marker="o", label="VaR99")
plt.plot(stab["start"], stab["CVaR99"], marker="o", label="CVaR99")
plt.plot(stab["start"], stab["Entropy(theta=6.0)"], marker="o", label="Entropy (the
plt.title(f"Stability Across Regimes - {series}")
plt.xlabel("Period start")
plt.ylabel("Loss")
plt.legend()
plt.show()

lookback_days = 252*2 # 2 years
horizon_days = 21      # ~1 month
theta_bt = 6.0

s = loss[series].dropna()

dates = s.index
bt_rows = []

for i in range(lookback_days, len(s) - horizon_days):
    train = s.iloc[i-lookback_days:i].values
    test = s.iloc[i:i+horizon_days].values

    pred = tilted_expectation(train, theta_bt)           # robust expected Loss esti
    realized = np.max(test)                            # realized worst Loss over
    bt_rows.append([dates[i], pred, realized])

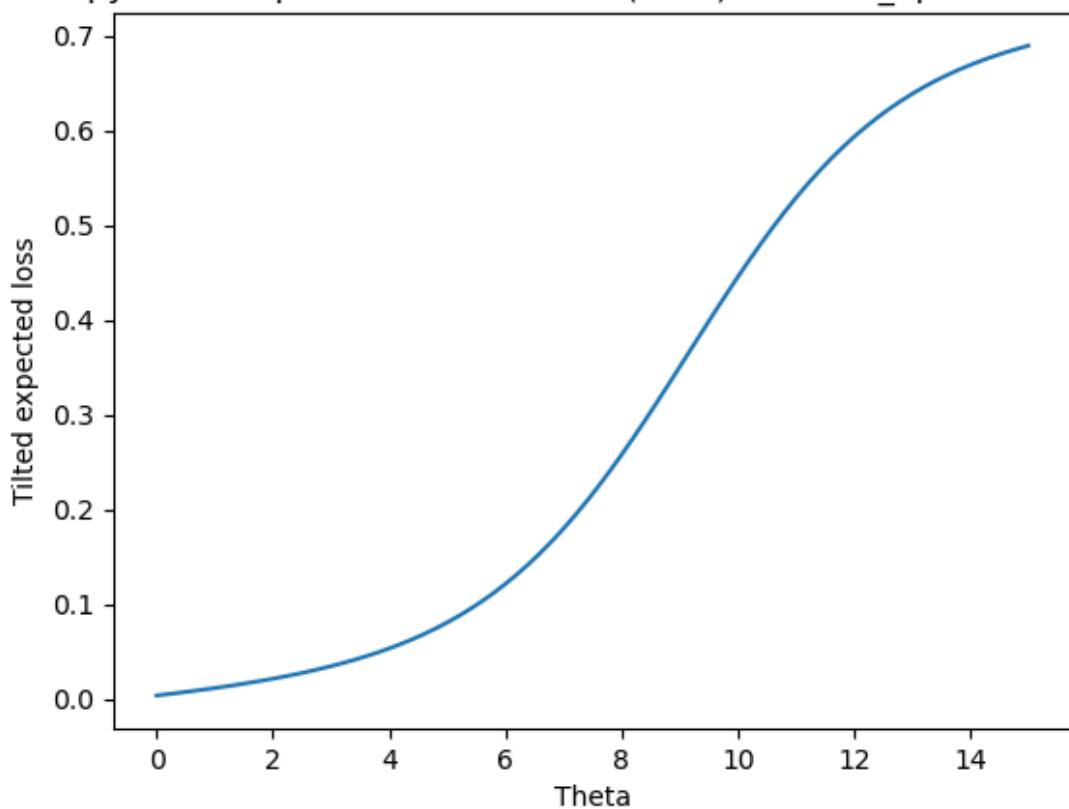
bt = pd.DataFrame(bt_rows, columns=["date", "pred_entropy", "realized_next_month_max"])

plt.figure()
bt[["pred_entropy", "realized_next_month_max"]].rolling(63).mean().plot(title="Backt
plt.xlabel("Date")
plt.ylabel("Loss")
plt.show()

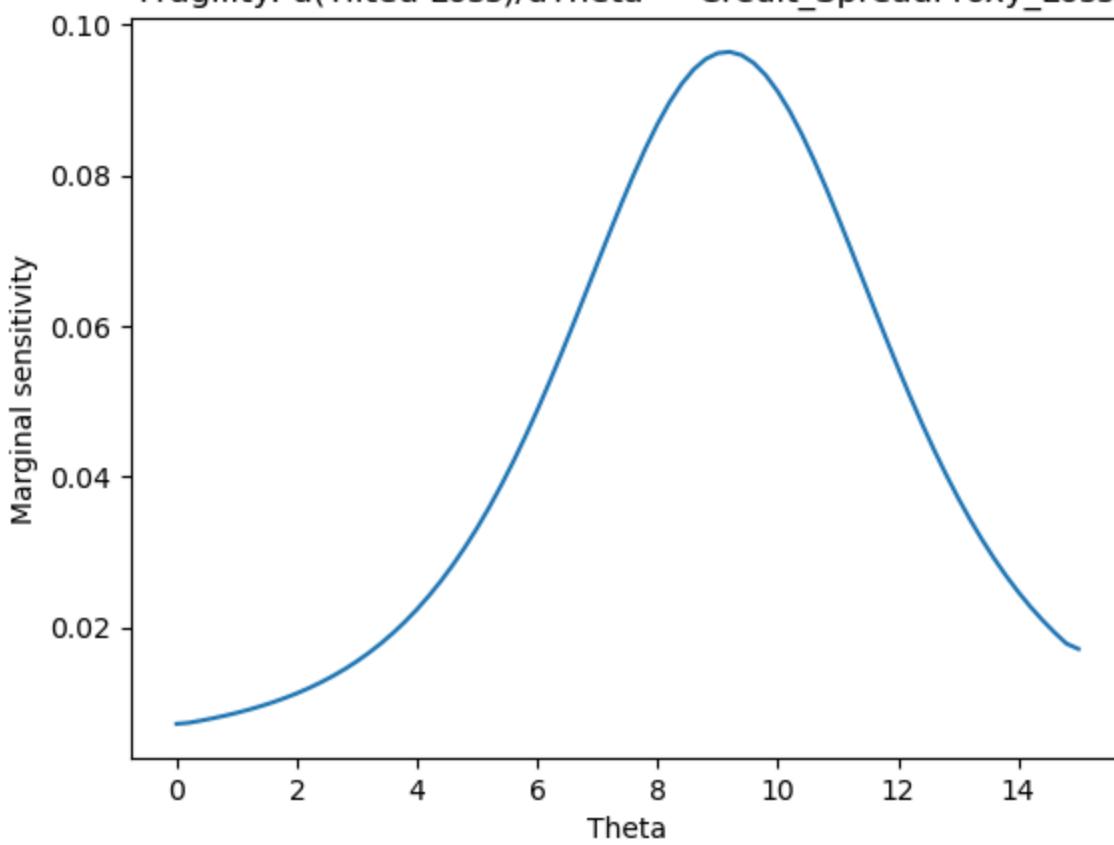
# Simple hit-rate: how often pred >= realized?
hit_rate = (bt["pred_entropy"] >= bt["realized_next_month_max"]).mean()
print("Conservatism hit-rate (pred >= realized):", round(hit_rate, 3))

```

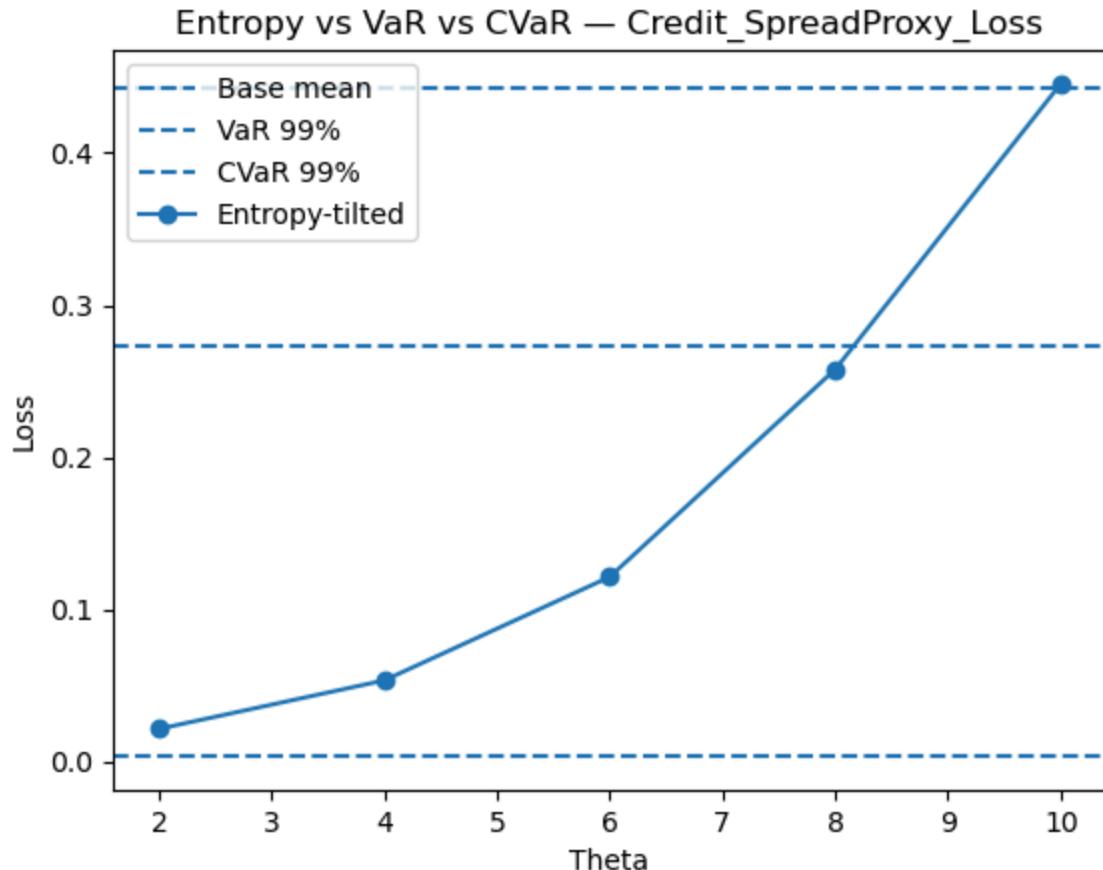
Entropy-Tilted Expected Loss vs Theta (0-15) — Credit_SpreadProxy_Loss

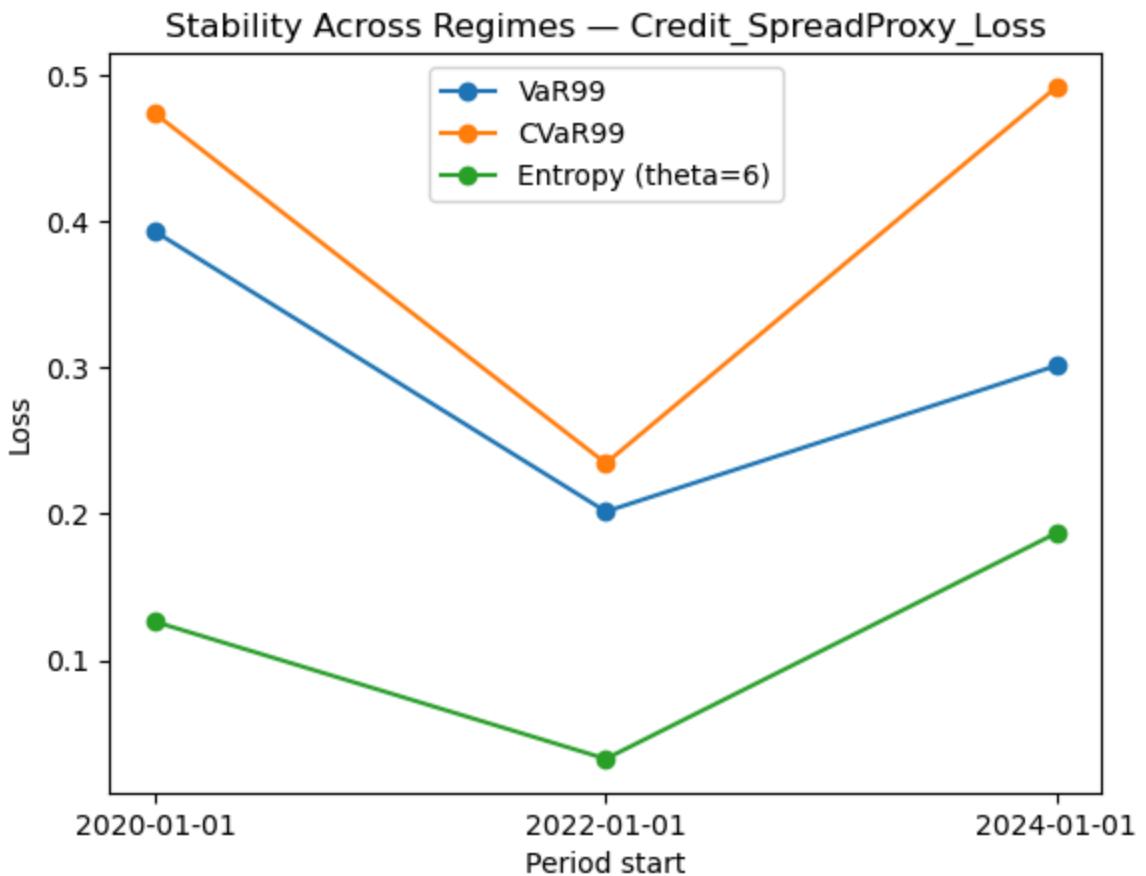


Fragility: $d(\text{Tilted Loss})/d\Theta$ — Credit_SpreadProxy_Loss

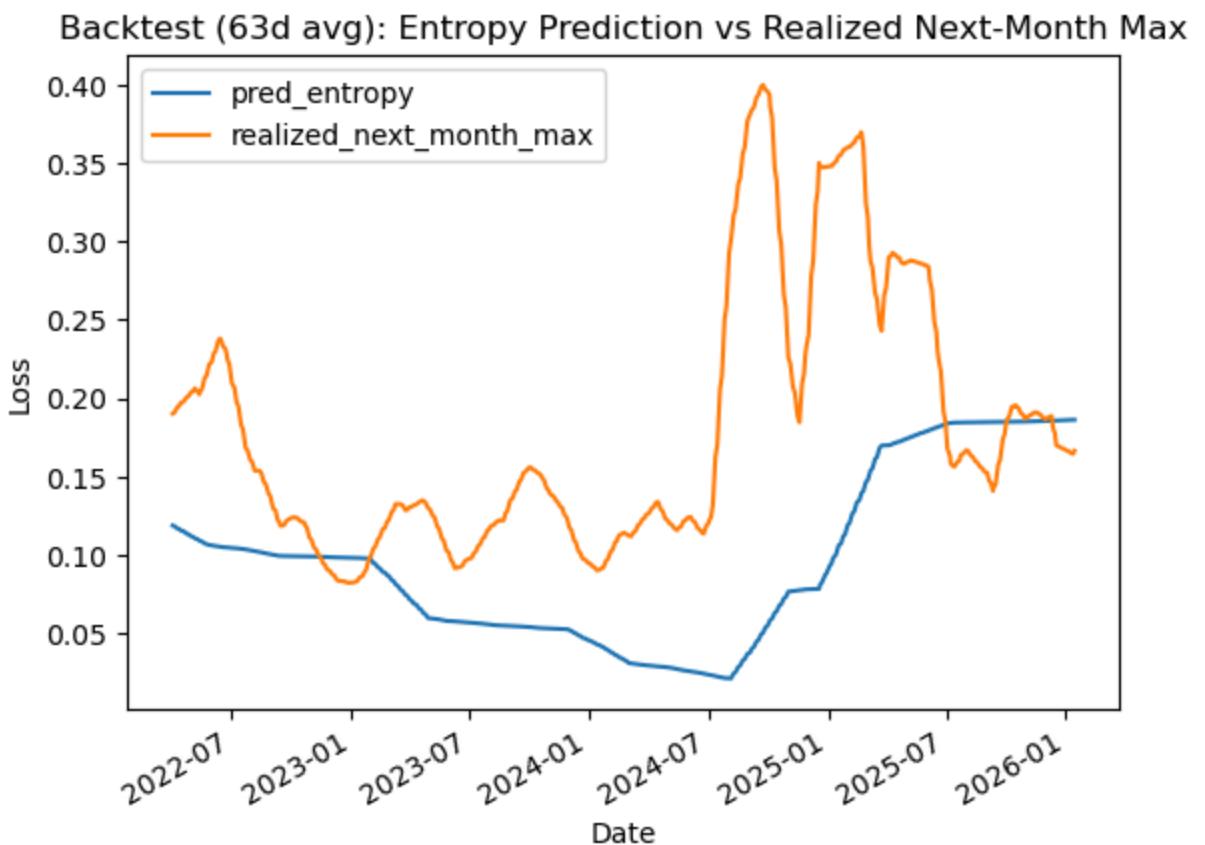


```
Tilted loss @theta=6: 0.12134165048235501
Tilted loss @theta=10: 0.4453961322972741
Max dLoss/dTheta: 0.09636680909666118
Base mean loss: 0.003700
VaR 99%: 0.272934
CVaR 99%: 0.442888
Entropy losses: {2: 0.021391, 4: 0.0533, 6: 0.121342, 8: 0.257755, 10: 0.445396}
```





<Figure size 640x480 with 0 Axes>



Conservatism hit-rate (pred \geq realized): 0.237