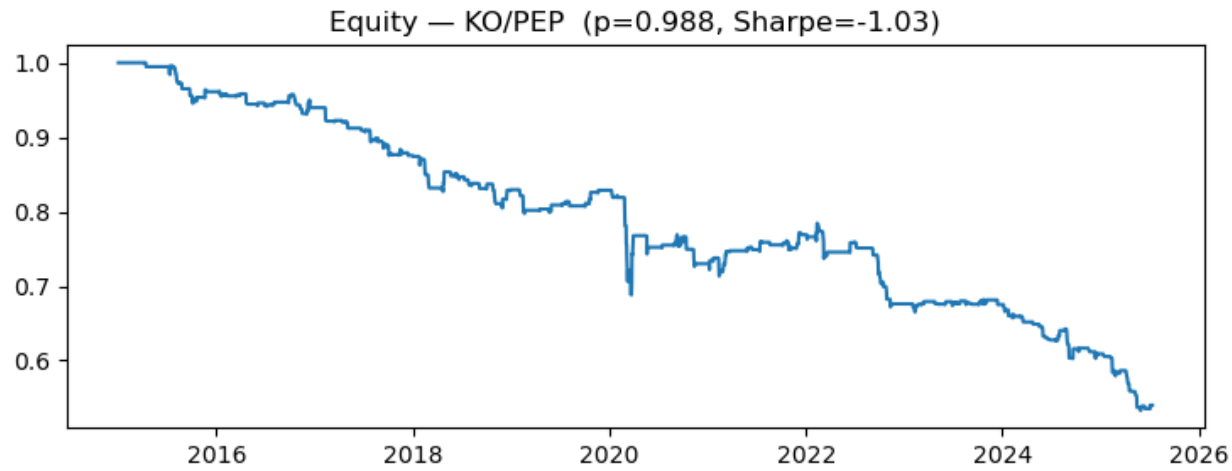
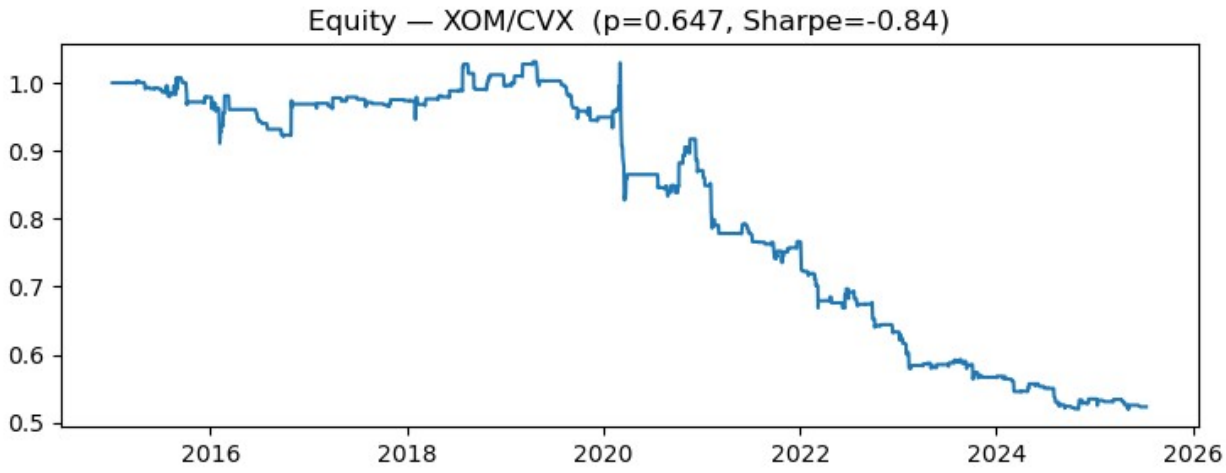
```
import warnings, numpy as np, pandas as pd, matplotlib.pyplot as plt,
itertools
from datetime import datetime
warnings.filterwarnings("ignore", category=FutureWarning)

try:
    import yfinance as yf
    from statsmodels.tsa.stattools import coint
except ImportError as e:
    raise SystemExit("□ Missing libs → pip install numpy pandas
matplotlib yfinance statsmodels") from e

#price loader

def _adj(df, tkr):
    if isinstance(df.columns, pd.MultiIndex):
        for f in ("Adj Close", "Close"):
            key = (f, tkr)
            if key in df: return df[key].astype(float)
    else:
        for f in ("Adj Close", "Close"):
            if f in df: return df[f].astype(float)
    raise KeyError(f"No price for {tkr}")

def get_px(tkr, start, end):
    df = yf.download(tkr, start=start, end=end, progress=False,
auto_adjust=False)
    s  = _adj(df, tkr).ffill(); s.name = tkr; return s


# Kalman β



def kalman_beta(y, x, d=1e-4, R=1e-3):
    n, beta = len(y), np.zeros(len(y)); bp, P = 0., 1.; Q = d/(1-d)
    for i in range(n):
        y_i, x_i = float(y.iat[i]), float(x.iat[i])
        Pp, betap = P+Q, bp; S = x_i*x_i*Pp + R; K = Pp*x_i/S
        b = betap + K*(y_i - x_i*betap); P = Pp - K*x_i*Pp
        beta[i] = b; bp = b
    return pd.Series(beta, index=y.index)


# Strategy helpers


WIN, EZ, XZ, TC = 60, 1.5, 0.5, 0.0005
```

```
def roll_z(sp):
    return ((sp - sp.rolling(WIN).mean().shift()) /
sp.rolling(WIN).std().shift()).fillna(0)

def pos_series(z):
    p = pd.Series(0, z.index); state=0
    for i,zv in enumerate(z):
        if state==0:
            if zv<-EZ: state=1
            elif zv>EZ: state=-1
        elif state==1 and zv>=-XZ: state=0
        elif state==-1 and zv<=XZ: state=0
        p.iat[i]=state
    return p

def returns(y,x,beta,pos):
    ry, rx = np.log(y).diff().fillna(0), np.log(x).diff().fillna(0)
    gross  = pos.shift().fillna(0)*(ry - beta*rx)
    cost   = pos.diff().abs().fillna(0)*TC*2
    return gross - cost

def summary(r, ppyr=252):
    cum = (1+r).prod()-1; ann = r.mean()*ppyr;
vol=r.std()*np.sqrt(ppyr)
    return cum, ann, vol, ann/vol if vol else np.nan

#PAIRS analysied
CANDIDATE_PAIRS = [
    ("KO","PEP"), ("XOM","CVX"),
    ("V","MA"), ("SPY","IVV"), ("VOO","SPY"), ("GLD","IAU"),
    ("UL","UN"), ("BHP","RIO"), ("MSFT","AAPL"), ("JPM","BAC"),
    ("DIS","CMCSA"), ("MMM","HON"), ("CAT","DE")
]

start, end = "2015-01-01", datetime.today().strftime("%Y-%m-%d")
results = []

for y_tkr, x_tkr in CANDIDATE_PAIRS:
    try:
        y, x = get_px(y_tkr, start, end), get_px(x_tkr, start, end)
    except Exception as e:
        print(f"Skip {y_tkr}/{x_tkr}: {e}"); continue
    if len(y)<500 or len(x)<500: print(f"Skip {y_tkr}/{x_tkr}: too
little data"); continue

    pval = coint(y,x)[1]
    beta = kalman_beta(y,x)
    z    = roll_z(y - beta*x)
    pos  = pos_series(z)
    ret  = returns(y,x,beta,pos)
```

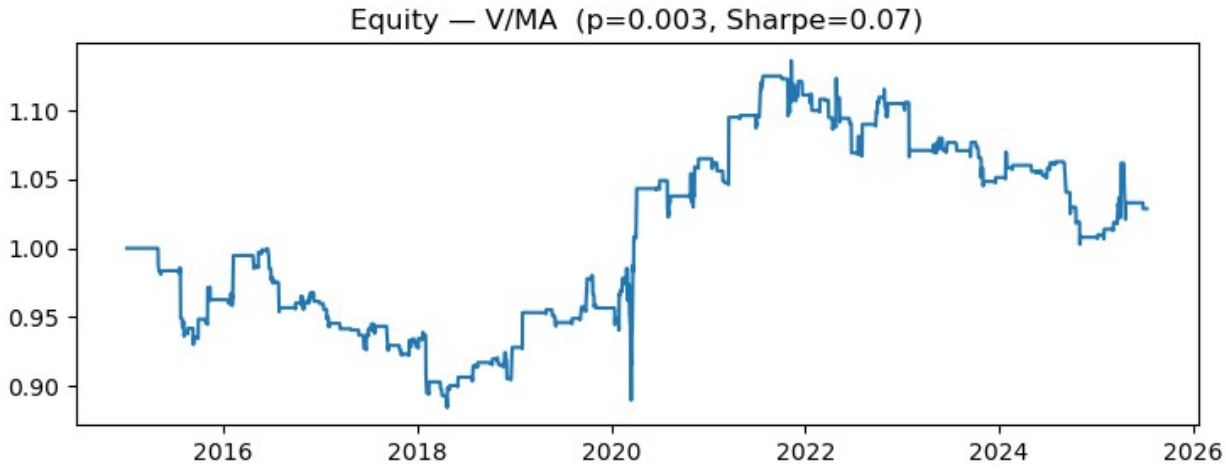
```
    cum, ann, vol, sh = summary(ret)
    results.append({"Pair":f"{y_tkr}/{x_tkr}", "p":pval, "Cum%":cum,
"Ann%":ann, "Vol%":vol, "Sharpe":sh})

    # plot & save
    plt.figure(figsize=(7.5,3))
    plt.plot((1+ret).cumprod())
    plt.title(f"Equity — {y_tkr}/{x_tkr}  (p={pval:.3f},
Sharpe={sh:.2f})")
    plt.tight_layout(); fn=f"equity_{y_tkr}_{x_tkr}.png";
plt.savefig(fn,dpi=150); plt.show()
    print("Saved", fn)

#result leaderboard

df = (pd.DataFrame(results)
        .assign(**{c:lambda d: d[c]*100 for c in ["Cum%","Ann%","Vol
%"]})
        .sort_values("Sharpe", ascending=False))

print("\nPerformance summary  (2015-today, $-neutral, 5 bps)")
if not df.empty:
    print(df.to_string(index=False, formatters={"p":lambda
v:f"{v:.4f}",
                                                "Cum%":lambda
v:f"{v:6.1f}%",
                                                "Ann%":lambda
v:f"{v:6.1f}%",
                                                "Vol%":lambda
v:f"{v:6.1f}%",
                                                "Sharpe":lambda
v:f"{v:6.2f}"}))
else:
    print("No pairs processed successfully.")
```



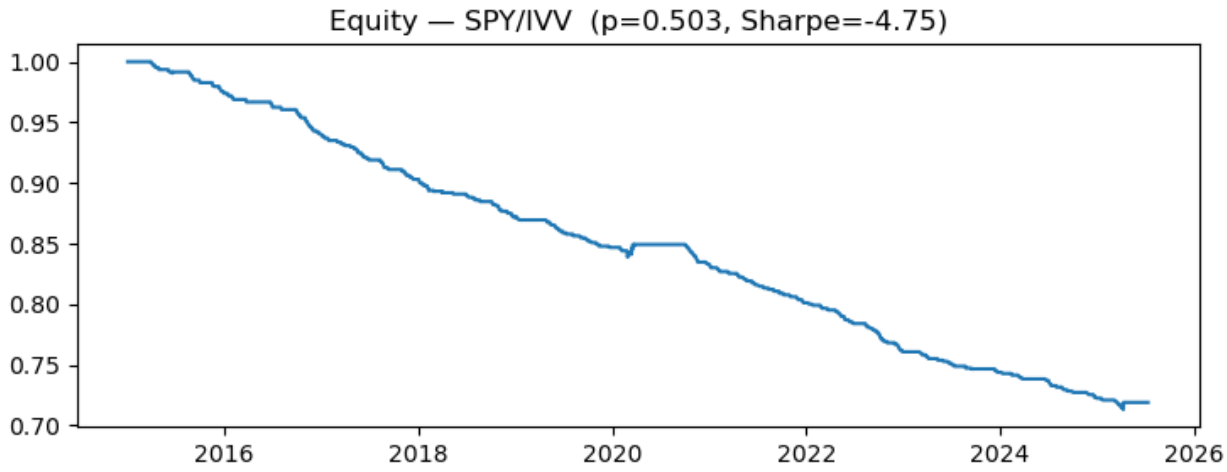
```
Saved equity_KO_PEP.png
```



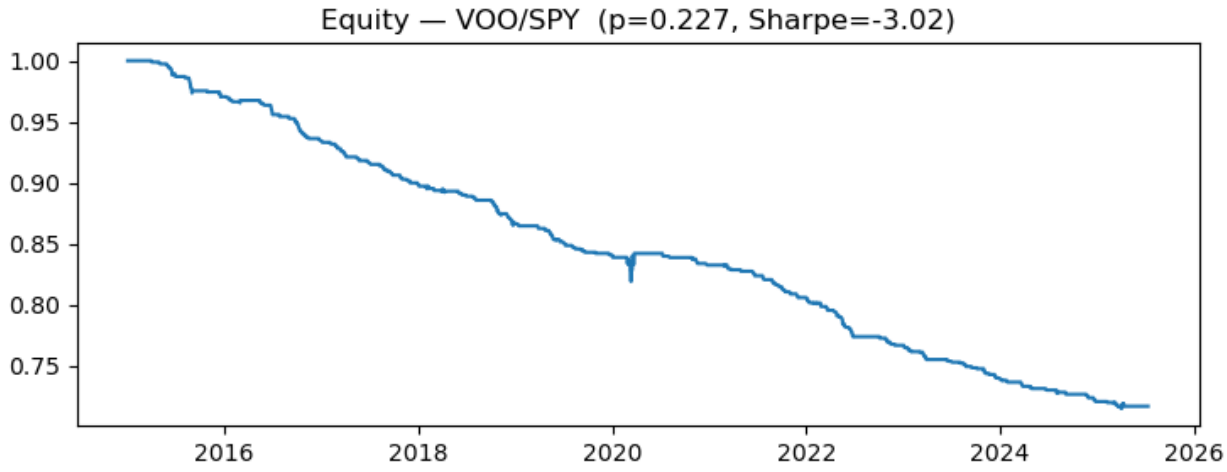

```
Saved equity_XOM_CVX.png
```



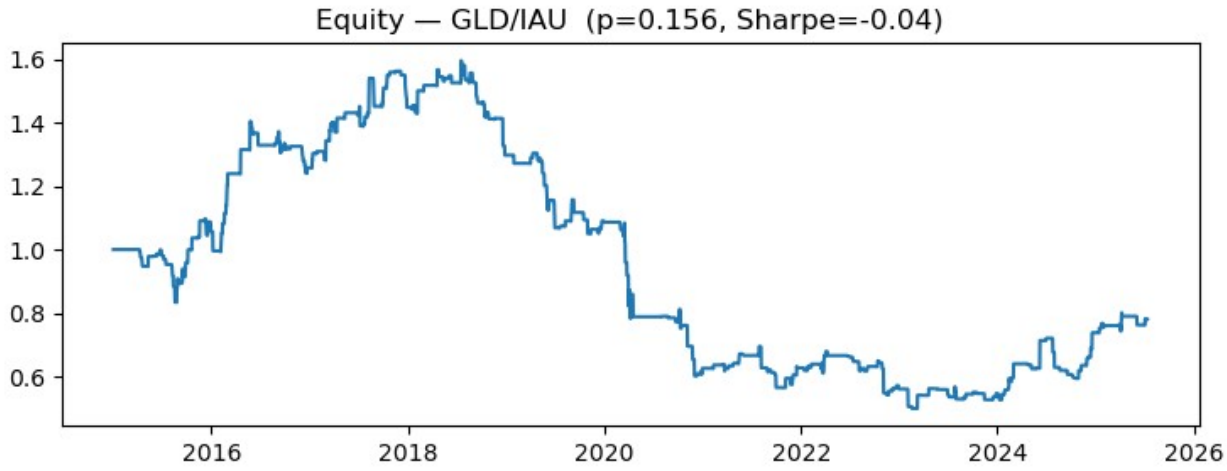

```
Saved equity_V_MA.png
```



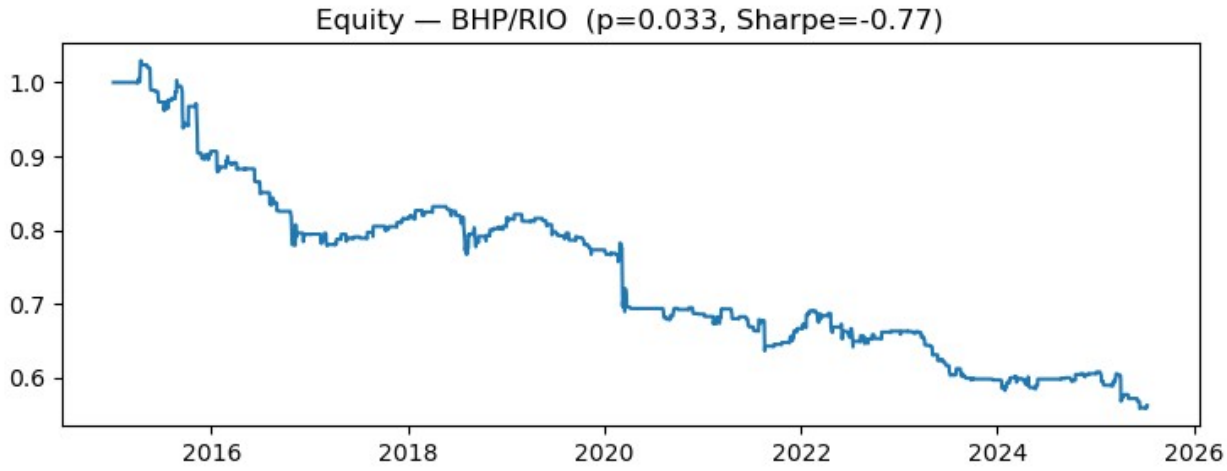

```
Saved equity_SPY_IVV.png
```



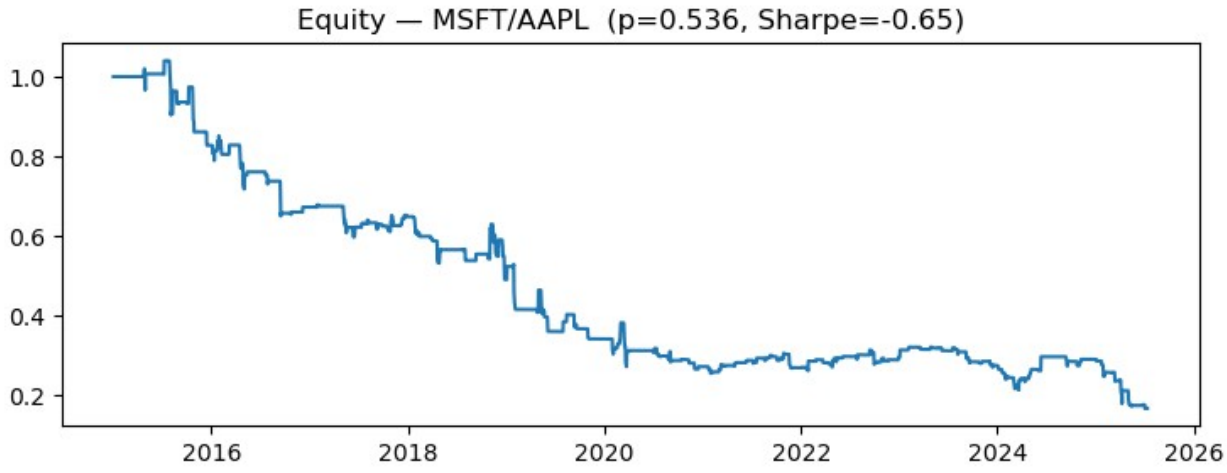

```
Saved equity_VOO_SPY.png
```



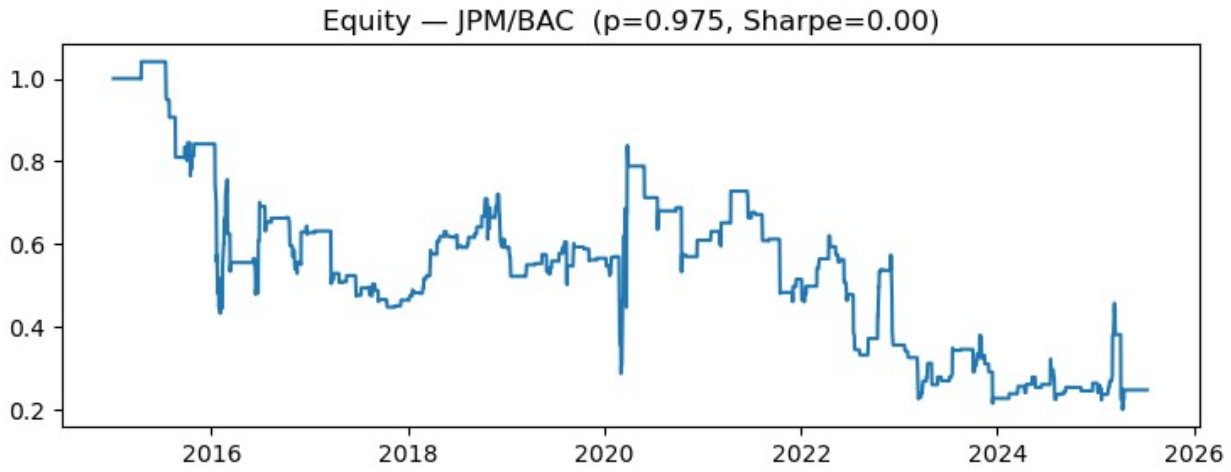
```
Saved equity_GLD_IAU.png


1 Failed download:
['UN']: YFTzMissingError('possibly delisted; no timezone found')

Skip UL/UN: too little data
```



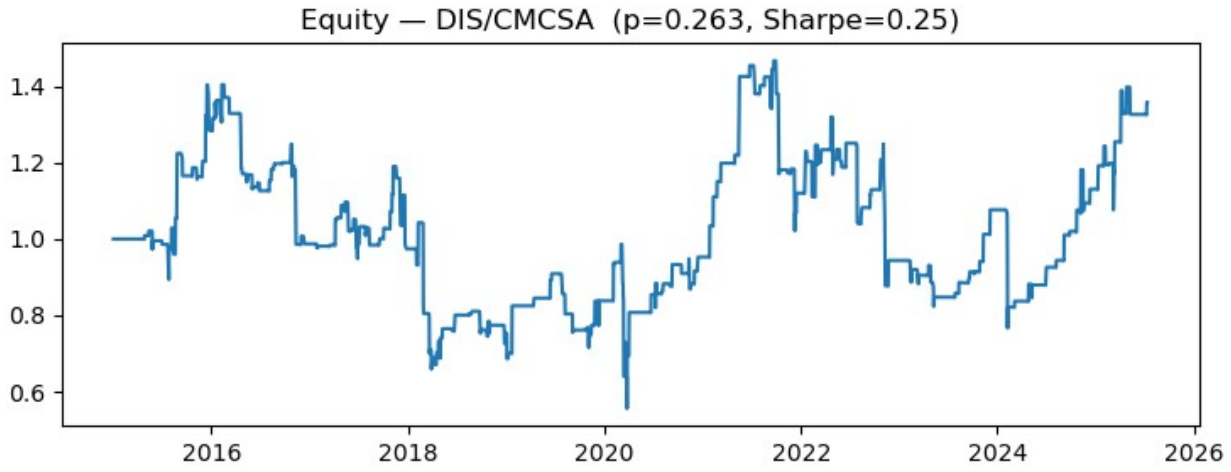

```
Saved equity_BHP_RIO.png
```



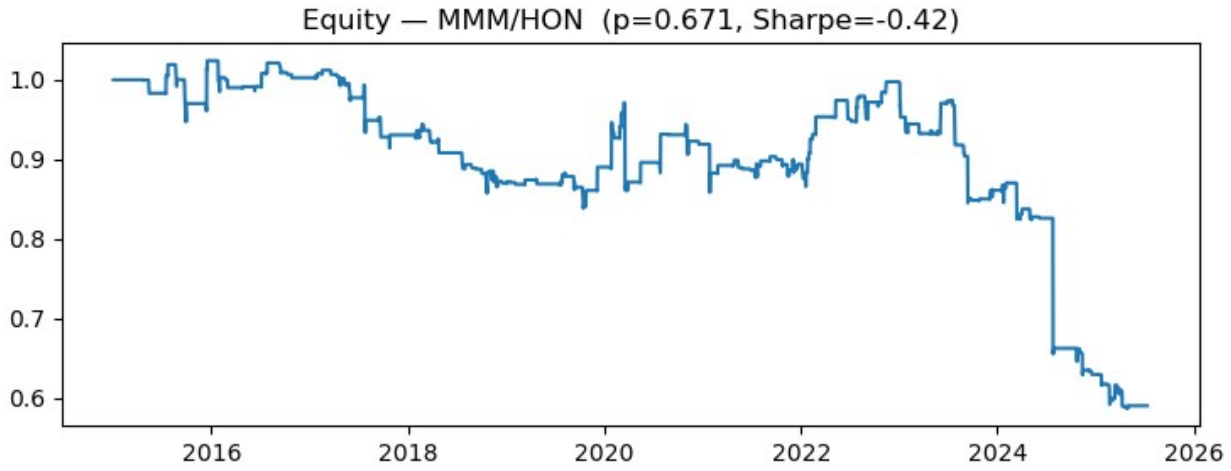

```
Saved equity_MSFT_AAPL.png
```



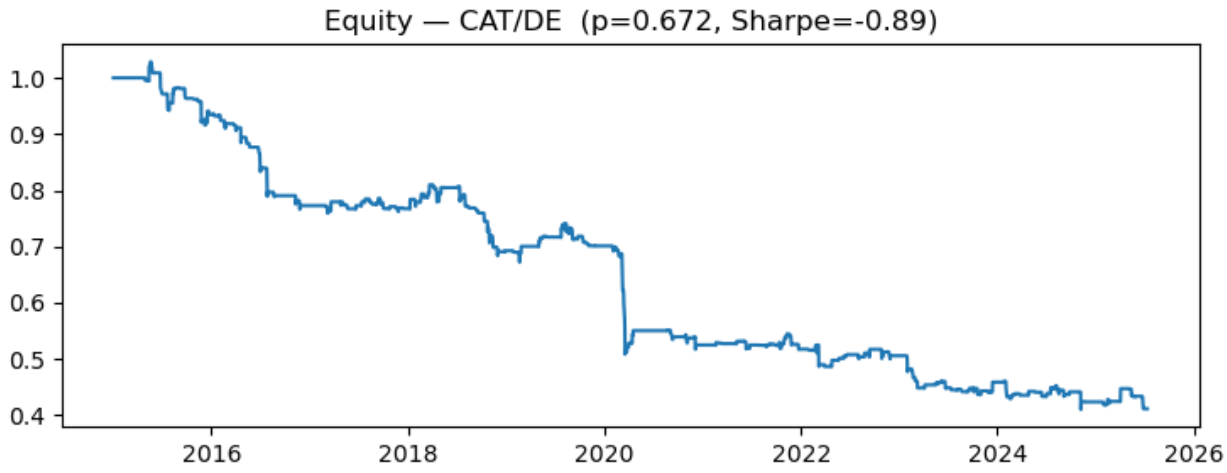

```
Saved equity_JPM_BAC.png
```




```
Saved equity_DIS_CMCSA.png
```



```
Saved equity_MMM_HON.png
```




```
Saved equity_CAT_DE.png

Performance summary  (2015-today, $-neutral, 5 bps)
     Pair      p    Cum%    Ann%    Vol% Sharpe
DIS/CMCSA 0.2626   30.8%   30.8%   30.8%   0.25
     V/MA 0.0034    5.9%    5.9%    5.9%   0.07
  JPM/BAC 0.9749   51.9%   51.9%   51.9%   0.00
  GLD/IAU 0.1564   17.9%   17.9%   17.9%  -0.04
  MMM/HON 0.6706   10.5%   10.5%   10.5%  -0.42
MSFT/AAPL 0.5359   22.4%   22.4%   22.4%  -0.65
  BHP/RIO 0.0333    6.8%    6.8%    6.8%  -0.77
  XOM/CVX 0.6473    7.0%    7.0%    7.0%  -0.84
   CAT/DE 0.6725    9.0%    9.0%    9.0%  -0.89
   KO/PEP 0.9883    5.5%    5.5%    5.5%  -1.03
  VOO/SPY 0.2271    1.1%    1.1%    1.1%  -3.02
  SPY/IVV 0.5033    0.7%    0.7%    0.7%  -4.75
```