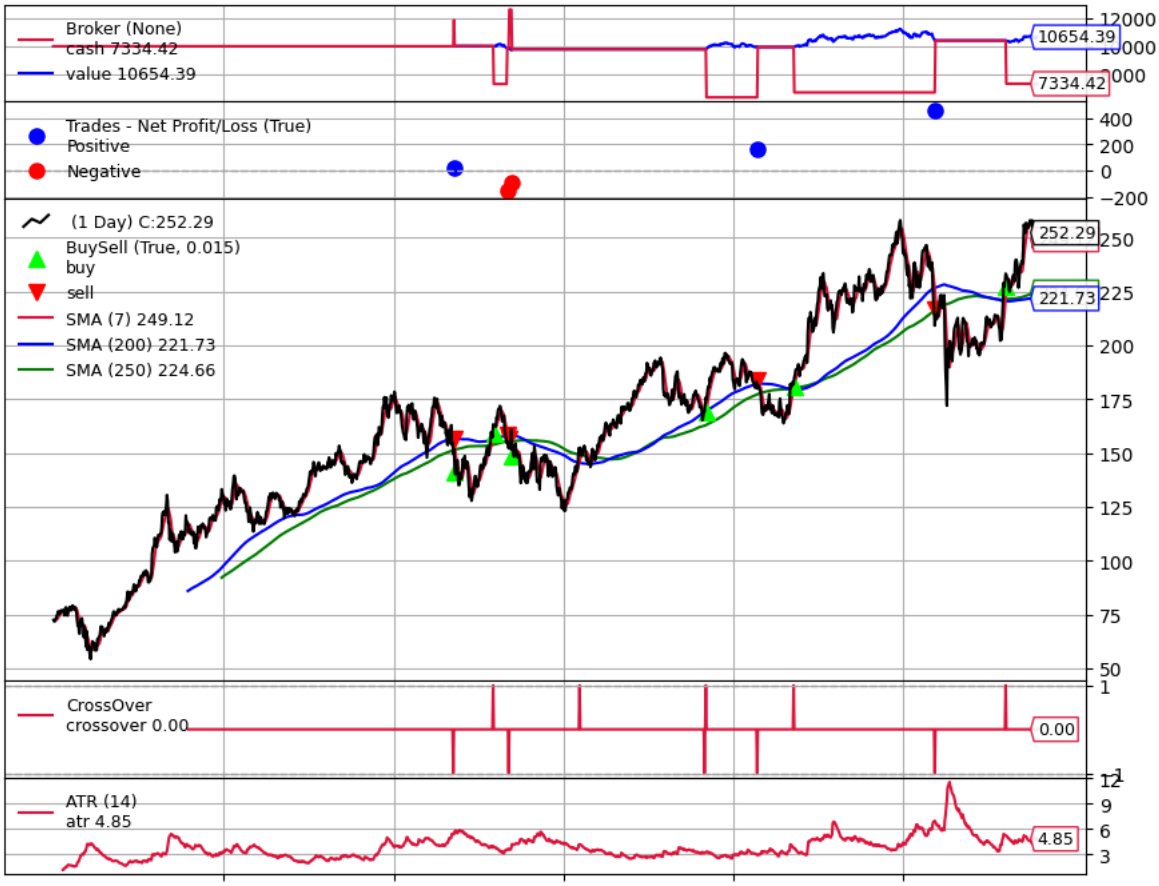
```
%matplotlib inline
!pip install backtrader --upgrade
```

```
Requirement already satisfied: backtrader in /usr/local/lib/python3.12/dist-packages (1.9.78.123)
```

```
import yfinance as yf
import backtrader as bt
import matplotlib.pyplot as plt

# Download historical data
data = yf.download('AAPL', start='2020-01-01', end='2025-10-18')
data.columns = data.columns.get_level_values(0)

# Improved Strategy: Filtered SMA Cross
class FilteredSmaCross(bt.Strategy):
    params = dict(
        fast=7,         # fast SMA period
        slow=200,         # slow SMA period
        trend=250,       # trend filter SMA period
        atr_period=14,   # ATR period
        risk_per_trade=0.01  # % of capital risk per trade
    )

    def __init__(self):
        self.sma_fast = bt.ind.SMA(period=self.p.fast)
        self.sma_slow = bt.ind.SMA(period=self.p.slow)
        self.sma_trend = bt.ind.SMA(period=self.p.trend)
        self.atr = bt.ind.ATR(period=self.p.atr_period)
        self.crossover = bt.ind.CrossOver(self.sma_fast, self.sma_slow)

    def next(self):
        cash = self.broker.getcash()
        risk = cash * self.p.risk_per_trade
        stop_dist = 1.5 * self.atr[0]  # ATR-based stop distance

        # If we have an open position
        if self.position:
            # Exit long if crossover turns negative or price falls below trend
            if self.position.size > 0 and (self.crossover < 0 or self.data.close < self.sma_tren
                self.close()
            # Exit short if crossover turns positive or price rises above trend
            elif self.position.size < 0 and (self.crossover > 0 or self.data.close > self.sma_tr
                self.close()

        # If no position, check for new trades
        else:
            # Long entry: price above trend, bullish crossover, ATR > 1
            if self.data.close > self.sma_trend * 1.005 and self.crossover > 0 and self.atr[0] :
                size = min(risk / stop_dist, cash * 0.11)
                self.buy(size=size)

            # Short entry: price below trend, bearish crossover
            elif self.data.close < self.sma_trend and self.crossover < 0:
                size = min(risk / stop_dist, cash * 0.95)
                self.sell(size=size)


# Backtesting setup
cerebro = bt.Cerebro()
cerebro.addstrategy(FilteredSmaCross)
data_feed = bt.feeds.PandasData(dataname=data)
cerebro.adddata(data_feed)
cerebro.broker.setcash(10000)
#cerebro.broker.setcommission(commission=0.001)  # optional: add small commission

# Add analyzers
cerebro.addanalyzer(bt.analyzers.SharpeRatio, _name='sharpe', timeframe=bt.TimeFrame.Days, annua
```

```
cerebro.addanalyzer(bt.analyzers.DrawDown, _name='drawdown')

# Run backtest
results = cerebro.run()
first_strategy = results[0]

# Extract analyzer results
sharpe = first_strategy.analyzers.sharpe.get_analysis()
drawdown = first_strategy.analyzers.drawdown.get_analysis()

sharpe_ratio = sharpe.get('sharperatio')
if sharpe_ratio:
    print(f"Sharpe Ratio: {sharpe_ratio:.5f}")
else:
    print("Sharpe Ratio: N/A")

print(f"Max Drawdown: {drawdown['max']['drawdown']:.5f}%")

# Plot results
%matplotlib inline
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = [9, 7]  # set figure size globally
cerebro.plot(iplot=False, volume=False)
plt.show()

```

```
/tmp/ipython-input-3543208684.py:6: FutureWarning: YF.download() has changed argument auto_adjust default to True
  data = yf.download('AAPL', start='2020-01-01', end='2025-10-18')
[*********************100%***********************]  1 of 1 completed
Sharpe Ratio: 0.04533
Max Drawdown: 8.47030%
```




```
import yfinance as yf
import backtrader as bt
import matplotlib.pyplot as plt

# Download historical data
data = yf.download('MSFT', start='2020-01-01', end='2025-10-18')
data.columns = data.columns.get_level_values(0)

```

```
# Improved Strategy: Filtered SMA Cross
class FilteredSmaCross(bt.Strategy):
    params = dict(
        fast=7,         # fast SMA period
        slow=200,         # slow SMA period
        trend=250,       # trend filter SMA period
        atr_period=14,   # ATR period
        risk_per_trade=0.01  # % of capital risk per trade
    )

    def __init__(self):
        self.sma_fast = bt.ind.SMA(period=self.p.fast)
        self.sma_slow = bt.ind.SMA(period=self.p.slow)
        self.sma_trend = bt.ind.SMA(period=self.p.trend)
        self.atr = bt.ind.ATR(period=self.p.atr_period)
        self.crossover = bt.ind.CrossOver(self.sma_fast, self.sma_slow)

    def next(self):
        cash = self.broker.getcash()
        risk = cash * self.p.risk_per_trade
        stop_dist = 1.5 * self.atr[0]  # ATR-based stop distance

        # If we have an open position
        if self.position:
            # Exit long if crossover turns negative or price falls below trend
            if self.position.size > 0 and (self.crossover < 0 or self.data.close < self.sma_tren
                self.close()
            # Exit short if crossover turns positive or price rises above trend
            elif self.position.size < 0 and (self.crossover > 0 or self.data.close > self.sma_tr
                self.close()

        # If no position, check for new trades
        else:
            # Long entry: price above trend, bullish crossover, ATR > 1
            if self.data.close > self.sma_trend * 1.005 and self.crossover > 0 and self.atr[0] >
                size = min(risk / stop_dist, cash * 0.11)
                self.buy(size=size)

            # Short entry: price below trend, bearish crossover
            elif self.data.close < self.sma_trend and self.crossover < 0:
                size = min(risk / stop_dist, cash * 0.95)
                self.sell(size=size)


# Backtesting setup
cerebro = bt.Cerebro()
cerebro.addstrategy(FilteredSmaCross)
data_feed = bt.feeds.PandasData(dataname=data)
cerebro.adddata(data_feed)
cerebro.broker.setcash(10000)
#cerebro.broker.setcommission(commission=0.001)  # optional: add small commission

# Add analyzers
cerebro.addanalyzer(bt.analyzers.SharpeRatio, _name='sharpe', timeframe=bt.TimeFrame.Days, annua
cerebro.addanalyzer(bt.analyzers.DrawDown, _name='drawdown')

# Run backtest
results = cerebro.run()
first_strategy = results[0]

# Extract analyzer results
sharpe = first_strategy.analyzers.sharpe.get_analysis()
drawdown = first_strategy.analyzers.drawdown.get_analysis()

sharpe_ratio = sharpe.get('sharperatio')
if sharpe_ratio:
    print(f"Sharpe Ratio: {sharpe_ratio:.5f}")
else:
    print("Sharpe Ratio: N/A")
```

```
    print("Sharpe Ratio: N/A")

print(f"Max Drawdown: {drawdown['max']['drawdown']:.5f}%")

# Plot results
%matplotlib inline
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = [9, 7]  # set figure size globally
cerebro.plot(iplot=False, volume=False)
plt.show()

```

```
/tmp/ipython-input-479678382.py:6: FutureWarning: YF.download() has changed argument auto_adjust default to True
  data = yf.download('MSFT', start='2020-01-01', end='2025-10-18')
[*********************100%***********************]  1 of 1 completed
Sharpe Ratio: -0.79811
Max Drawdown: 12.11707%
```