# Credit Trading Strategies: Full Toolkit

## 1. Bloomberg Commands and Functions

### Historical Time Series (Excel)

* Z-Spread: =BDH("US9128285M81 Corp","Z\_SPREAD","1/1/2023","12/31/2023","Per=DAILY")
* CDS Spread: =BDH("CDS\_USA\_5Y\_CORP\_SPREAD","PX\_LAST","1/1/2023","12/31/2023")

### Static Data (Excel)

* OAS Spread: =BDP("US9128285M81 Corp","OAS\_SPREAD")
* Modified Duration: =BDP("US9128285M81 Corp","MOD\_DUR\_ADJ")
* DV01: =BDP("US9128285M81 Corp","DV01")

### Common Bloomberg Functions

* YAS <GO> – Yield and spread analysis for bonds
* CDSW <GO> – CDS analytics and PV01
* RV <GO> – Relative value comparison
* GCDS <GO> – CDS curve builder
* SRCH <GO> – Bond screening
* CSDR <GO> – CDS relative value analysis
* SWPM <GO> – Swaption vol surface and trade analysis

## 2. Python Code Snippets for Strategy Backtesting

### Z-Spread Mean Reversion

import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
  
# Historical Z-spread time series  
z\_spreads = pd.Series([...]) # Insert Bloomberg data here  
rolling\_mean = z\_spreads.rolling(window=60).mean()  
rolling\_std = z\_spreads.rolling(window=60).std()  
z\_score = (z\_spreads - rolling\_mean) / rolling\_std  
  
# Signal: Buy when z\_score < -2, Sell when z\_score > 2  
buy\_signals = z\_score < -2  
sell\_signals = z\_score > 2  
  
plt.plot(z\_spreads, label="Z-Spread")  
plt.plot(rolling\_mean, label="Mean")  
plt.fill\_between(z\_spreads.index, rolling\_mean - 2\*rolling\_std, rolling\_mean + 2\*rolling\_std, color='grey', alpha=0.3)  
plt.legend()  
plt.title("Z-Spread Mean Reversion")  
plt.show()

### Relative Value Pair Trade

import pandas as pd  
import matplotlib.pyplot as plt  
  
# Spread between Bond A and Bond B  
bond\_a = pd.Series([...]) # Z-spread for bond A  
bond\_b = pd.Series([...]) # Z-spread for bond B  
spread = bond\_a - bond\_b  
spread\_mean = spread.rolling(window=60).mean()  
spread\_std = spread.rolling(window=60).std()  
z\_score = (spread - spread\_mean) / spread\_std  
  
# Signal: Long bond A / Short bond B when z\_score < -2, reverse when > 2  
plt.plot(spread, label='Spread')  
plt.plot(spread\_mean, label='Mean')  
plt.fill\_between(spread.index, spread\_mean - 2\*spread\_std, spread\_mean + 2\*spread\_std, color='lightgrey', alpha=0.5)  
plt.legend()  
plt.title("Bond Spread Z-Score")  
plt.show()

### CDS-Bond Basis Trade

# CDS spread vs Bond Z-spread  
import pandas as pd  
import matplotlib.pyplot as plt  
  
cds\_spread = pd.Series([...])  
bond\_zspread = pd.Series([...])  
basis = cds\_spread - bond\_zspread  
  
# Strategy: Buy bond and buy CDS when basis is significantly negative  
plt.plot(basis, label='CDS - Z-Spread')  
plt.axhline(0, linestyle='--', color='grey')  
plt.title("CDS-Bond Basis")  
plt.legend()  
plt.show()

### Credit Curve Steepener/Flattener

# Credit curve: 1Y, 5Y, 10Y CDS spreads  
import pandas as pd  
import matplotlib.pyplot as plt  
  
cds\_1y = pd.Series([...])  
cds\_5y = pd.Series([...])  
cds\_10y = pd.Series([...])  
  
# Flattener: Long 10Y CDS, Short 1Y CDS if curve steep  
spread\_1y\_10y = cds\_10y - cds\_1y  
plt.plot(spread\_1y\_10y, label="10Y - 1Y CDS Spread")  
plt.title("CDS Curve Flattener")  
plt.legend()  
plt.show()

### Volatility Trading via Swaptions

# Placeholder: This strategy would involve comparing implied vol (Bloomberg) vs realized vol (computed)  
# Implied vol is not typically scraped, must be manually loaded or accessed via Bloomberg API  
import pandas as pd  
import matplotlib.pyplot as plt  
  
realized\_vol = pd.Series([...]) # e.g., rolling std of 10Y rates  
implied\_vol = pd.Series([...]) # manually pulled or input  
  
plt.plot(implied\_vol, label='Implied Vol')  
plt.plot(realized\_vol, label='Realized Vol')  
plt.title("Implied vs Realized Volatility")  
plt.legend()  
plt.show()  
  
# Sell vol when implied >> realized

### Regression-Based Relative Value Model

import pandas as pd  
import statsmodels.api as sm  
  
# Independent variables: VIX, 10Y yield  
X = pd.DataFrame({  
 'VIX': [...],  
 '10Y\_Yield': [...],  
})  
X = sm.add\_constant(X)  
  
# Dependent variable: Bond spread  
y = pd.Series([...]) # e.g., OAS of bond  
  
# Fit model  
model = sm.OLS(y, X).fit()  
predicted = model.predict(X)  
residual = y - predicted  
z\_score = (residual - residual.mean()) / residual.std()  
  
# Signal: Buy if z < -2, Sell if z > 2  
print(model.summary())