Trading Indicators Handbook

June 15, 2023

SMA Trading indicator

Contents

About

The Simple Moving Average (SMA) is a technical indicator that calculates the **average price** over a specific number of periods, and it moves along as new data is added, making it a "moving average".

Calculating

Formula

SMA = (Sum of price data for the last N periods) / N

For example, a 5-period SMA would sum up the last 5 closing prices and divide it by 5 to find the average. It is called a 'moving' average because as new prices become available, the oldest prices are dropped and the average recalculates.

Minute	Open	High	Low	Close	SMA
1	$10.0	$11.0	$9.5	$10.0	-
2	$10.1	$12.1	$10.0	$12.0	-
3	$12.2	$15.2	$12.0	$15.0	-
4	$15.1	$15.1	$13.9	$14.0	-
5	$14.1	$16.1	$14.0	$16.0	$13.4

Using our given market data and specifically the closing prices, here's how the 5-minute SMA would be calculated after Minute 5:

1. Minute 1-5 close prices: `$10.0, $12.0, $15.0, $14.0, $16.0`
2. SMA = `(10.0 + 12.0 + 15.0 + 14.0 + 16.0) / 5 = $13.4`

The 5-minute SMA after the 5th minute would be $13.4.

Pros and Cons

Pros:

- The SMA is simple and easy to calculate and understand.
- It smooths out price fluctuations and helps to filter out the "noise" of the market.
- It's useful for identifying trend directions over the specified period.

Cons:

- The SMA is a lagging indicator, meaning it's based on past prices and tends to be slow to respond to recent price changes.
- Because it equally weighs all data points, it might not accurately reflect recent changes in the market.
- It might give false signals in volatile markets because it doesn't adapt to market changes as quickly as some other indicators.

Example of signals

Traders often use two SMAs: a **short-term** one and a **long-term** one. When the **short-term** SMA crosses **above** the long-term SMA, it's considered a bullish (**buy**) signal. When it crosses **below**, it's a bearish (**sell**) signal.

True Positive:

In an uptrending market, the short-term SMA might cross above the long-term SMA, correctly indicating a continuing upward trend and a good time to buy.

False Positive:

Let's say the market is in a sideways trend (prices fluctuate within a narrow range). A brief price spike could cause the short-term SMA to cross above the long-term SMA, generating a buy signal. However, this could be misleading as the overall trend hasn't changed.

Use in Real Trading

In real trading, SMA can be paired with other indicators for better results. For instance, a trader might use SMA in conjunction with the **Relative Strength Index (RSI)**. The RSI could help confirm whether the market is overbought or oversold when the SMAs cross.

Python Implementation

```python
# Importing Required Libraries
import pandas as pd
import yfinance as yf
import matplotlib.pyplot as plt

# Define the Ticker Symbol
tickerSymbol = 'GOOG'

# Get the data on this ticker
tickerData = yf.Ticker(tickerSymbol)

# Get the historical prices for the specified period
tickerDf = tickerData.history(period='1d', start='2021-1-1', end='2023-6-15')

# Calculate Simple Moving Averages
tickerDf['SMA30'] = tickerDf['Close'].rolling(window=30).mean()
tickerDf['SMA100'] = tickerDf['Close'].rolling(window=100).mean()

# Plot the chart
plt.figure(figsize=(12.2, 4.5)) #width = 12.2in, height = 4.5
plt.plot(tickerDf.index, tickerDf['SMA30'], label='SMA30', color = 'red')
plt.plot(tickerDf.index, tickerDf['SMA100'], label='SMA100', color = 'blue')
plt.plot(tickerDf.index, tickerDf['Close'], label='GOOG Close', color = 'gray')

plt.legend(loc='upper left')
plt.show()
```

sma.png

EMA - Exponential Moving Average

About

Unlike the SMA, the EMA gives more weight to recent data, making it quicker to respond to price changes.

Calculating

Formula

EMA = (Close - Previous EMA) * Multiplier + Previous EMA
Where:

- **Close**: the closing price for a given period (Price today/now). Here close prices is for example
- **N**: the period of the EMA.
- **Multiplier**: `2 / (N + 1)`
- **Previous EMA** is the EMA of the previous period.

The EMA for the first period is just the **Close price**.
But for subsequent periods, it's calculated as follows:

- `EMA = (Close - Previous EMA) * Multiplier + Previous EMA`
- `EMA[i] = (Close[i] - EMA[i-1]) * 2/(N+1) + EMA[i-1]`

However, the EMA's calculation is slightly more complex for the initial period because there is **no previous EMA**. In this case, we use the SMA as the first EMA:

```
EMA(first period) = SMA
```

Example:
Let's calculate a 5-minute EMA at Minute 6 with the following market data:

Minute	Open	High	Low	Close	EMA
1	$10.0	$11.0	$9.5	$10.0	-
2	$10.1	$12.1	$10.0	$12.0	-
3	$12.2	$15.2	$12.0	$15.0	-
4	$15.1	$15.1	$13.9	$14.0	-
5	$14.1	$16.1	$14.0	$16.0	13.4(SMA)
6	$16.1	$16.1	$14.9	$15.0	15.67
7	$15.1	$17.1	$15.0	$17.0	⋯
8	$17.1	$17.1	$15.9	$16.0	⋯
9	$16.1	$18.1	$16.0	$18.0	⋯

**Can be calculated from first price*
Here, `N = 5`, so `Multiplier = 2 / (5 + 1) = 0.33`.

- `SMA(5) = (10.0 + 12.0 + 15.0 + 14.0 + 16.0) / 5 = 13.4`
- This value becomes the first EMA (`EMA[5]`). Now, to calculate the EMA for the **6th** minute, we use the EMA formula:

```
EMA[6] = (Close[6] - EMA[5]) * multiplier + EMA[5]
       = (15.0 - 13.4) * 0.33 + 13.4
       = 14.13
```

Pros and Cons

Pros:

- More responsive: By giving more weight to recent prices, the EMA can adapt faster to price changes.
- Combines trend and momentum: The EMA not only captures the overall trend but also shows the asset's momentum.
- Often used for High frequency trading.

Cons:

- More prone to false signals: The sensitivity of the EMA can sometimes lead to false signals, especially in volatile markets.
- Complex calculation: Compared to the SMA, the EMA's calculation is slightly more complex, especially for longer periods.

Example of signals

Like the SMA, traders often use two EMAs: a short-term one and a long-term one. When the **short-term** EMA **crosses above** the long-term EMA, it's a bullish (**buy**) signal, and when it crosses below, it's a bearish (sell) signal.

True Positive:

In minute 7, the short EMA crosses above the long EMA, which is a buy signal. The price then goes up, confirming this was a correct signal.

In a stable uptrend, the short-term EMA might cross above the long-term EMA, correctly suggesting that it's a good time to enter a long position.

False Positive:

In minute 11, the short EMA dips below the long EMA, suggesting a sell signal. However, the price increases in the next minute, making this a false signal.

In a volatile market, the price might swing up and down sharply, causing the short-term EMA to cross the long-term EMA back and forth, generating multiple buy and sell signals that could be misleading.

Use in Real Trading

In real trading, EMA can be used in combination with other indicators such as **MACD** (Moving Average Convergence Divergence) or **Bollinger Bands**.

For instance, a trader might look for the short-term EMA to cross above the long-term EMA and the MACD to cross above its signal line as a confirmation for a long position.

Python Implementation

```
    # Importing Required Libraries
    import pandas as pd
    import yfinance as yf
    import matplotlib.pyplot as plt

    # Define the Ticker Symbol
    tickerSymbol = 'BAC'

    # Get the data on this ticker
    tickerData = yf.Ticker(tickerSymbol)

    # Get the historical prices for the specified period
    tickerDf = tickerData.history(period='1d', start='2021-4-1', end='2022-10-30')

    # Calculate Exponential Moving Averages
    tickerDf['EMA12'] = tickerDf['Close'].ewm(span=12, adjust=False).mean()
    tickerDf['EMA26'] = tickerDf['Close'].ewm(span=26, adjust=False).mean()

    # Plot the chart
    plt.figure(figsize=(12.2, 4.5)) #width = 12.2in, height = 4.5
    plt.plot(tickerDf.index, tickerDf['EMA12'], label='EMA12', color = 'red')
    plt.plot(tickerDf.index, tickerDf['EMA26'], label='EMA26', color = 'blue')
    plt.plot(tickerDf.index, tickerDf['Close'], label='BAC Close', color = 'gray')

    plt.title('Exponential Moving Averages of '+tickerSymbol)
    plt.xlabel('Date',fontsize=15)
    plt.ylabel('Price USD ($)',fontsize=15)
    plt.legend(loc='upper left')
    plt.show()
```

ema.png

MACD - Moving Average Convergence Divergence

About

The Moving Average Convergence Divergence (MACD) is a versatile trading indicator used in various forms of trading, including high-frequency trading (HFT).

MACD is a trend-following momentum indicator that shows the relationship between two moving averages of a security's price.

Calculating

Formula

The MACD is calculated by subtracting the **26-period** Exponential Moving Average (**EMA**) from the **12-period EMA**. A **9-day EMA** of the MACD, called the "signal line", is then plotted on top of the MACD, functioning as a trigger for buy and sell signals.

Here's the formula for the MACD:

```
    MACD = 12-Period EMA − 26-Period EMA
```

To calculate the signal line, you take the 9-period EMA of the MACD.

```
    Signal Line = 9-Period EMA of MACD
```

Example:
Since MACD requires 26 periods to start, we'll calculate from the 26th minute:

Minute	Open	High	Low	Close	EMA12	EMA26	EMA9 (Signal)	MACD
1	$10.0	$11.0	$9.5	$10.0	-	-	-	-
2	$10.1	$12.1	$10.0	$12.0	-	-	-	-

Minute	Open	High	Low	Close	EMA12	EMA26	EMA9 (Signal)	MACD
3	$12.2	$15.2	$12.0	$15.0	-	-	-	-
4	$15.1	$15.1	$13.9	$14.0	-	-	-	-
5	$14.1	$16.1	$14.0	$16.0	-	-	-	-
6	$16.1	$16.1	$14.9	$15.0	-	-	-	-
7	$15.1	$17.1	$15.0	$17.0	-	-	-	-
8	$17.1	$17.1	$15.9	$16.0	-	-	-	-
9	$16.1	$18.1	$16.0	$18.0	-	-	Starts here	-
10	$18.1	$18.1	$16.9	$17.0	-	-		-
11	$17.1	$19.1	$17.0	$19.0	-	-		-
12	$19.1	$19.1	$17.9	$18.0	Starts here	-		-
13	$18.1	$20.1	$18.0	$20.0		-		-
14	$20.1	$21.1	$20.0	$21.0		-		-
15	$21.1	$21.1	$19.9	$20.0		-		-
16	$20.1	$22.1	$20.0	$21.0		-		-
17	$21.1	$23.1	$21.0	$22.0		-		-
18	$22.1	$24.1	$22.0	$23.0		-		-
19	$23.1	$25.1	$23.0	$24.0		-		-
20	$24.1	$26.1	$24.0	$25.0		-		-
21	$25.1	$27.1	$25.0	$26.0		-		-
22	$26.1	$28.1	$26.0	$27.0		-		-
23	$27.1	$29.1	$27.0	$28.0		-		-
24	$28.1	$30.1	$28.0	$29.0		-		-
25	$29.1	$31.1	$29.0	$30.0		-		-
26	$30.1	$32.1	$30.0	$31.0		Starts here		Starts here
27	$31.1	$33.1	$31.0	$32.0				
28	$32.1	$34.1	$32.0	$33.0				
29	$33.1	$35.1	$33.0	$34.0				
30	$34.1	$36.1	$34.0	$35.0				

In python it can be calculated in the following way:

```python
import pandas as pd

# Assuming 'df' is your DataFrame and 'Close' is the column with closing prices
df['EMA12'] = df['Close'].ewm(span=12, adjust=False).mean()
df['EMA26'] = df['Close'].ewm(span=26, adjust=False).mean()
df['MACD'] = df['EMA12'] - df['EMA26']
df['Signal'] = df['MACD'].ewm(span=9, adjust=False).mean()
```

Pros and Cons

Pros:

- Trend Identification: MACD can identify the **start of a trend**, providing good entry points.
- Signal Line Crossovers: The MACD's signal line can provide clear buy and sell signals.

Cons:

- False Signals: Like any indicator, MACD can produce **false signals**, particularly in **volatile markets**.
- Lag: MACD can sometimes lag behind the market because it's a trend-following indicator.
- It may not work well in ranging (non-trending) markets, where price movements can be random.

Example of signals

True Positive:

A true positive in MACD is a situation where the MACD line crosses **above the signal line** and the **price goes up**, indicating a strong bullish signal, or the MACD line crosses **below the signal line** and the **price goes down**, indicating a strong bearish signal.

These signals can help traders decide when to enter or exit trades.

False Positive:
A false positive in MACD is typically a situation where the MACD line crosses above the signal line (potential buy signal), but the price does not go up, or the MACD line crosses below the signal line (potential sell signal), but the price does not go down. It's important to confirm MACD signals with other indicators or patterns to avoid false positives.

Use in Real Trading

In a real trading scenario, traders often use MACD in conjunction with other indicators to confirm signals and minimize risks. For example, a trader may use the Relative Strength Index (**RSI**) together with MACD.

If MACD gives a buy signal (MACD line crosses above the signal line), and RSI is below 30 (indicating an oversold condition), the trader may consider this as a strong buy signal.

Python Implementation

```
    # Importing Required Libraries
    import pandas as pd
    import yfinance as yf
    import matplotlib.pyplot as plt

    # Define the Ticker Symbol
    tickerSymbol = 'AAPL'

    # Get the data on this ticker
    tickerData = yf.Ticker(tickerSymbol)

    # Get the historical prices for the specified period
    tickerDf = tickerData.history(period='1d', start='2021-1-1', end='2023-6-15')

    # Calculate MACD
    # Short term EMA
    ShortEMA = tickerDf.Close.ewm(span=12, adjust=False).mean()

    # Long term EMA
    LongEMA = tickerDf.Close.ewm(span=26, adjust=False).mean()

    # Calculate MACD line
    MACD = ShortEMA - LongEMA

    # Calculate Signal Line
    signal = MACD.ewm(span=9, adjust=False).mean()

    # Add MACD and signal line to the data frame
    tickerDf['MACD'] = MACD
    tickerDf['Signal Line'] = signal

    # Plot the chart
    plt.figure(figsize=(12.2, 4.5)) #width = 12.2in, height = 4.5
    plt.plot(tickerDf.index, MACD, label='AAPL MACD', color = 'red')
    plt.plot(tickerDf.index, signal, label='Signal Line', color='blue')
    plt.legend(loc='upper left')
    plt.show()
```

macd.png

RSI - Relative Strength Index

About

The Relative Strength Index (RSI) is a momentum oscillator that measures the speed and change of price movements. Developed by J. Welles Wilder, the RSI is a very popular indicator that is used primarily to identify **overbought** and **oversold** conditions in a market.

Calculating

The RSI is calculated using the following steps:

1. Calculate the average gain and the average loss over the specified period (usually 14 periods).
2. Compute the relative strength (RS) which is the ratio of average gain to average loss.
3. The RSI is then calculated using the formula: RSI = 100 - (100 / (1 + RS))

Formula

RSI = 100 - (100 / (1 + RS))
Where:

- **RS**: (Relative Strength) = Average Gain / Average Loss

Pros and Cons

Pros:

- RSI is a versatile indicator that can be used in trending or ranging markets.
- It helps identify potential reversals, overbought/oversold conditions, and divergence.

Cons:

- During strong trends, the RSI may remain overbought or oversold for extended periods.
- False signals can occur, especially during choppy market conditions.

Example of signals

Minute	Open	High	Low	Close
1	$10.0	$11.0	$9.5	$10.0
2	$10.1	$12.1	$10.0	$12.0
3	$12.2	$15.2	$12.0	$15.0
4	$15.1	$15.1	$13.9	$14.0
5	$14.1	$16.1	$14.0	$16.0
6	$16.1	$16.1	$14.9	$15.0
7	$15.1	$17.1	$15.0	$17.0
8	$17.1	$17.1	$15.9	$16.0
9	$16.1	$18.1	$16.0	$18.0
10	$18.1	$18.1	$16.9	$17.0
11	$17.1	$19.1	$17.0	$19.0
12	$19.1	$19.1	$17.9	$18.0
13	$18.1	$20.1	$18.0	$20.0
14	$20.1	$21.1	$20.0	$21.0
15	$21.1	$21.1	$19.9	$20.0
16	$20.1	$22.1	$20.0	$21.0
17	$21.1	$23.1	$21.0	$22.0
18	$22.1	$24.1	$22.0	$23.0
19	$23.1	$25.1	$23.0	$24.0
20	$24.1	$26.1	$24.0	$25.0
21	$25.1	$27.1	$25.0	$26.0

For the following examples, let's assume the RSI settings are set to the standard **14 periods**.

True Positive:

In the 6th minute, the price drops from $16 to $15. The RSI dips below the 30 level, indicating an oversold condition. In the next minute, the price increases to $17, marking a successful buy signal.

False Positive:

In the 21th minute, the price jumps from $25 to $26. The RSI exceeds 70, implying an overbought condition. However, the price continues to increase in the next minutes, rendering this a false sell signal.

Use in Real Trading

In real trading, RSI is often used in conjunction with other indicators to increase the robustness of the signals.

For instance, traders might look for price action patterns (like support/resistance breaks) or confirmations from other indicators before acting on an RSI signal.

Python Implementation

```
# Importing Required Libraries
import pandas as pd
import yfinance as yf
import matplotlib.pyplot as plt

# Define the Ticker Symbol
tickerSymbol = 'MSFT'

# Get the data on this ticker
tickerData = yf.Ticker(tickerSymbol)

# Get the historical prices for the specified period
tickerDf = tickerData.history(period='1d', start='2020-1-1', end='2023-1-25')

# Calculate RSI
delta = tickerDf['Close'].diff(1)
delta = delta.dropna()
up = delta.copy()
down = delta.copy()
up[up < 0] = 0
down[down > 0] = 0
average_gain = up.rolling(window=14).mean()
average_loss = abs(down.rolling(window=14).mean())
rs = average_gain / average_loss
RSI = 100.0 - (100.0 / (1.0 + rs))

# Add RSI to the data frame
tickerDf['RSI'] = RSI

# Plot the chart
plt.figure(figsize=(12.2, 4.5)) #width = 12.2in, height = 4.5
plt.plot(tickerDf.index, tickerDf['RSI'], label='RSI', color = 'red')
plt.axhline(0, linestyle='--', alpha = 0.5, color = 'gray')
plt.axhline(10, linestyle='--', alpha = 0.5, color = 'orange')
plt.axhline(20, linestyle='--', alpha = 0.5, color = 'green')
plt.axhline(30, linestyle='--', alpha = 0.5, color = 'red')
plt.axhline(70, linestyle='--', alpha = 0.5, color = 'red')
plt.axhline(80, linestyle='--', alpha = 0.5, color = 'green')
plt.axhline(90, linestyle='--', alpha = 0.5, color = 'orange')
plt.axhline(100, linestyle='--', alpha = 0.5, color = 'gray')
plt.title('RSI of '+tickerSymbol)
plt.xlabel('Date',fontsize=15)
plt.ylabel('RSI',fontsize=15)
plt.legend(loc='upper left')
plt.show()
```

rsi.png