

1. Introduction

InvestIQ-v1 est une infrastructure modulaire de backtesting en Python, conçue pour analyser et tester des stratégies systématiques.

La version 1 propose un moteur fonctionnel complet (data, stratégie, exécution, journalisation), prêt à être utilisé ou étendu.

2. Structure du code

```
src/
|
└── backtest_engine/          # moteur de simulation
└── strategy_engine/         # interface stratégie + exemple
└── historical_data_engine/   # récup. données historiques
└── export_engine/            # export résultats / métriques
└── config/                  # fichiers de configuration
└── utilities/               # outils généraux
└── Main.py                   # point d'entrée
```

3. Installation (Windows)

Ce guide permet d'installer et d'exécuter *InvestIQ-v1* dans un environnement isolé et reproductible.

3.1 Cloner le dépôt

Ouvrir PowerShell puis exécuter :

```
git clone https://github.com/ManuelBayon/InvestIQ-v1.git
cd InvestIQ-v1
```

3.2 Créer un environnement virtuel

Créer un environnement dédié dans le dossier `InvestIQ-v1` :

```
python -m venv .venv
```

3.3 Activer l'environnement virtuel

```
.venv\Scripts\Activate.ps1
```

La ligne de commande doit afficher un préfixe (.venv) , par exemple :

```
(.venv) PS C:\Users\Manuel\Documents\...\src>
```

3.4 Installer les dépendances

```
pip install -r requirements.txt
```

4. Configuration Interactive Brokers (TWS)

Cette section explique comment installer et configurer **Trader Workstation (TWS)** pour permettre à InvestIQ-v1 de communiquer avec Interactive Brokers (en mode simulé ou réel).

4.1. Pourquoi une configuration TWS propre ?

TWS mélange dans le même dossier :

- les exécutables (`tws.exe` , `ibgateway.exe`)
- **les paramètres utilisateurs** (`jts.ini` , fichiers XML, caches)
- les logs

Pour éviter les conflits, assurer la reproductibilité et permettre à InvestIQ-v1 de charger une configuration propre, **il est recommandé d'installer TWS dans un dossier dédié**, différent du dossier `C:\Jts` (qui est le dossier de configuration utilisé par IB par défaut).

4.2 Installation de TWS (Trader WorkStation)

Étape 1 - Télécharger TWS

Télécharger TarderWorkstation (TWS) sur le site officiel d'Interactive Brokers :

👉 <https://www.interactivebrokers.ie/en/trading/trading-platforms.php>

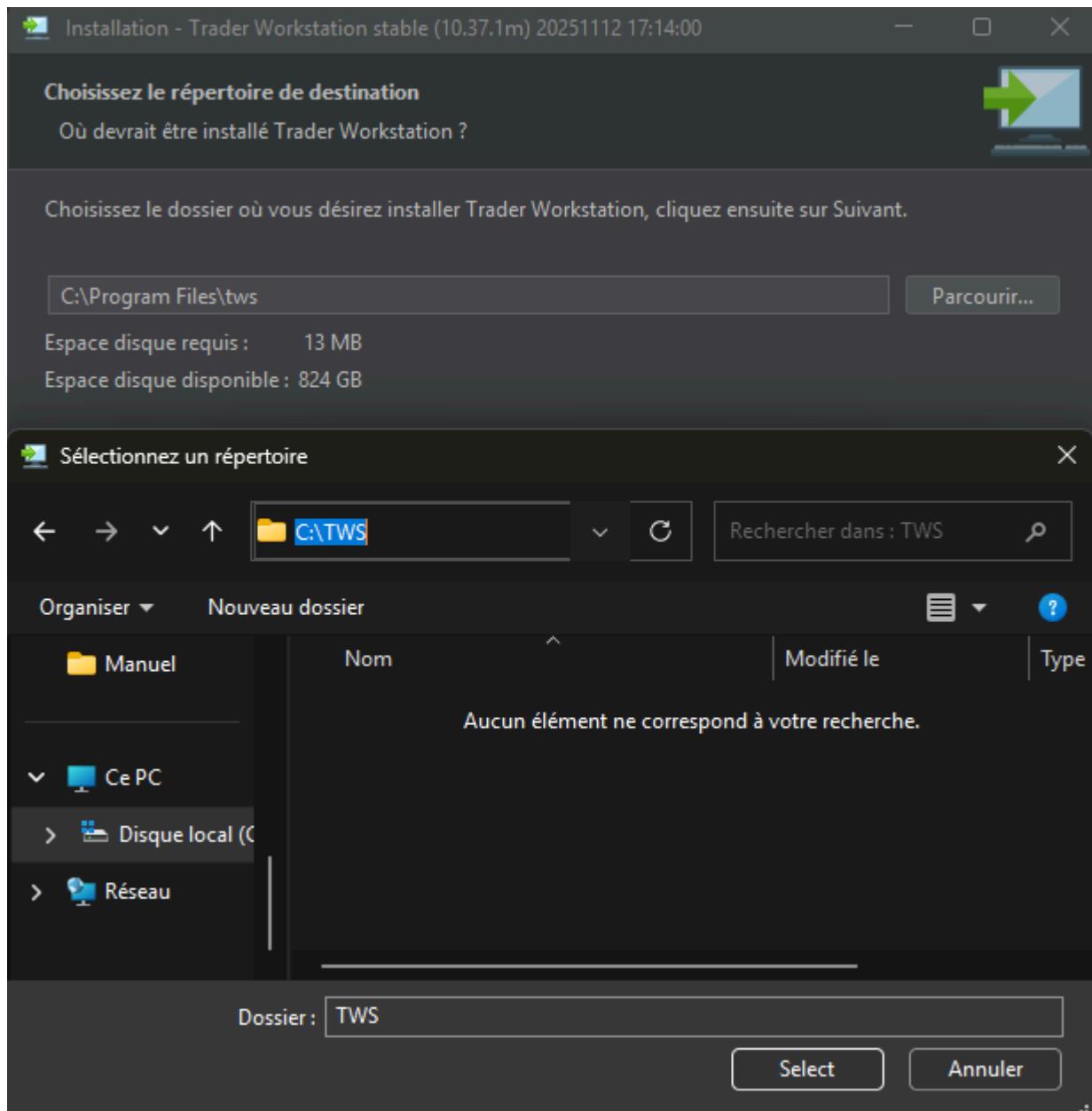
Deux versions existent :

- **TWS (recommandé)**
- **TWS Latest** (plus fréquent en mise à jour)

Étape 2 - Créer un dossier d'installation propre

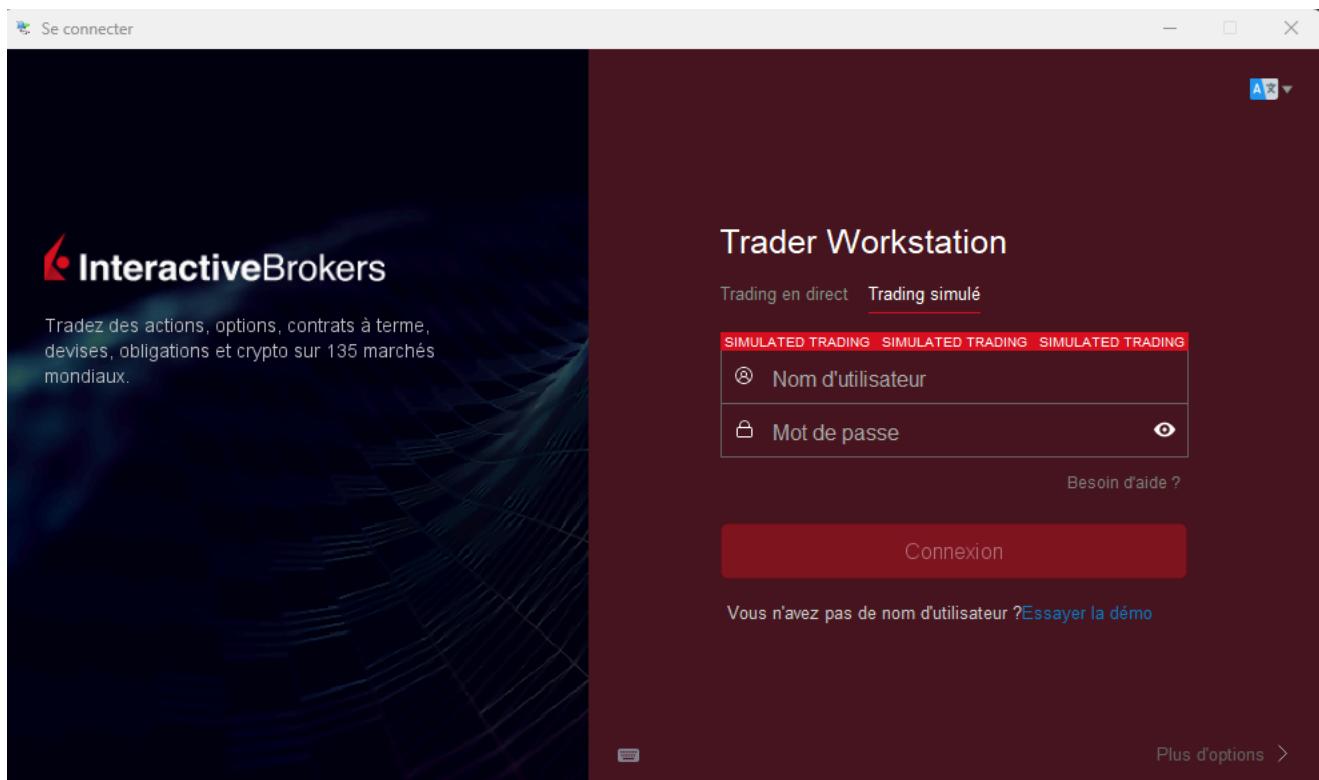
⚠️ Attention — par défaut, l'installateur de TWS choisit le dossier C:\Jts .

1. Créer un dossier C:\TWS
2. Lors de l'installation, sélectionner ce dossier comme destination.



4.3 Première connexion (Trading Simulé)

1. Lancer TWS
2. Se connecter en Trading Simulé
3. Attendre que l'interface se charge complètement

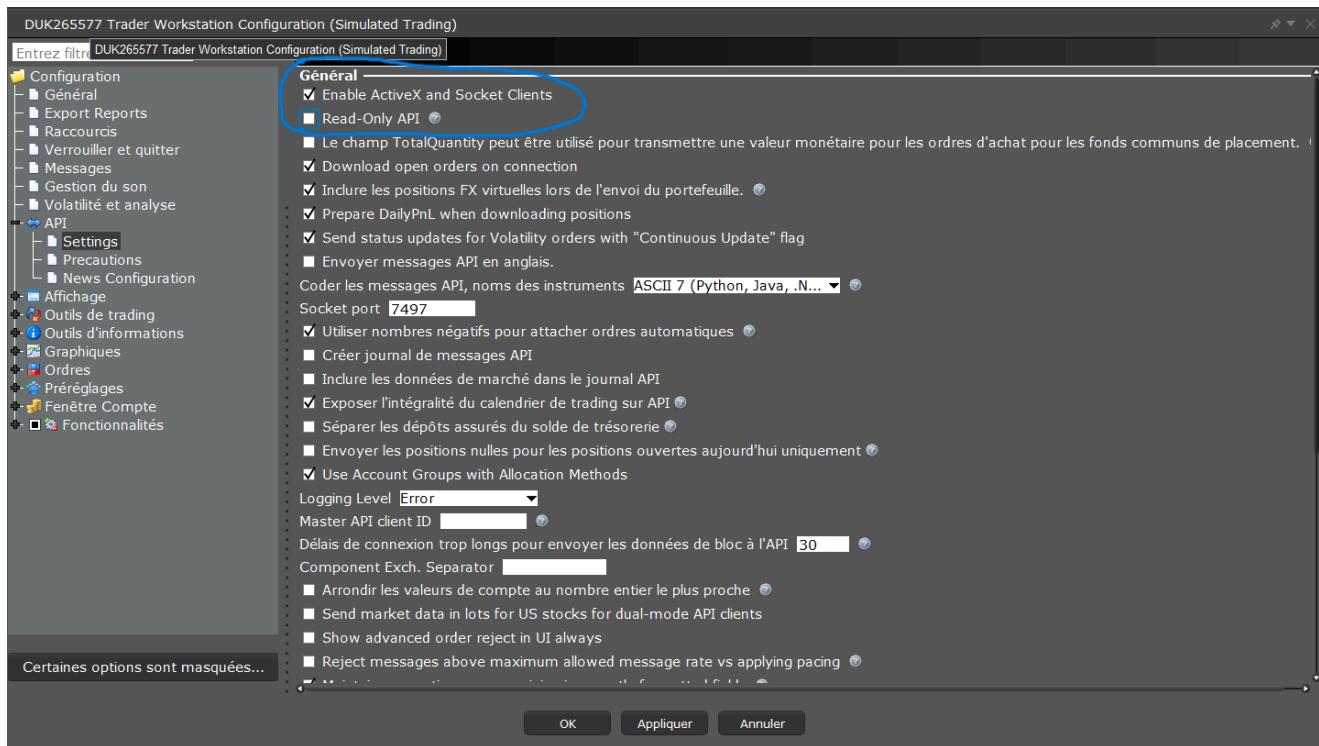


4.4. Configuration API requise

Dans TWS pour trouver les paramètres de l'API suivre les instructions suivantes :

- **Fichier** → *Configuration Générale*
- **API** → *Settings*

1. Cocher : **Enable ActiveX and Socket Clients**
2. Décocher : **Read-Only API**
3. Vérifier le port : **7497** (compte simulé)



⚠ Ne PAS fermer TWS

5. Création d'une stratégie personnalisée

Pour créer une nouvelle stratégie dans **InvestIQ-v1**, rendez-vous dans :

```
./src/strategy_engine/
```

Vous y trouverez la classe abstraite `AbstractStrategy`, qui définit l'interface que toutes les stratégies doivent respecter.

Une stratégie doit **hériter** de `AbstractStrategy` et **implémenter** la méthode suivante :

```
generate_signals(self, data: pd.DataFrame) -> pd.DataFrame
```

Cette méthode reçoit les données de marché (OHLC + timestamp) et doit retourner un `DataFrame` contenant au minimum :

- `timestamp`
- `close` (ou autre prix utilisé)
- `target_position` (position cible souhaitée, utilisée par le moteur FIFO)

5.1 Exemple complet pas à pas

Étape 1 — Créer un fichier `exemple.py`

Dans :

```
./src/engine
```

Créer un fichier nommé comme vous le souhaitez, par exemple :

```
exemple.py
```

Étape 2 : Créer la classe de stratégie

Voici le squelette minimal d'une stratégie :

```
from strategy_engine.AbstractStrategy import AbstractStrategy
import pandas as pd

class MaStrategie(AbstractStrategy):

    def __init__(self,
                 param_1: int,
                 param_2: float = 0.025):
        """
        Exemple d'initialisation de paramètres de stratégie.
        param_1 : entier (ex : période courte)
        param_2 : flottant (ex : seuil)
        """
        self.param_1 = param_1
        self.param_2 = param_2
```

Étape 3 : Implémenter la méthode `generate_signals`

La signature doit être strictement :

```
def generate_signals(self, data: pd.DataFrame) -> pd.DataFrame:
```

Et vous devez retourner un `DataFrame` contenant **au minimum** :

```
[ "timestamp", "close", "target_position" ]
```

Note

Le moteur utilise par défaut `close` pour le calcul du PnL.
Voir ci-dessous pour changer le type de prix.

Étape 3 (bis): Modifier le type de prix utilisé pour le PnL

Pour utiliser `open`, `high`, `low` ou tout autre prix :

1. Aller dans :

```
./src/backtest_engine/portfolio/portfolio.py
```

2. Trouver la méthode `generate_and_apply_fifo_operations_from_signals`

- 3.Modifier :

```
price = row.close
```

en :

```
price = row.open # ou row.high / row.low
```

Étape 4 : Exemple final : stratégie `BollingerMeanReversion`

Objectif :

- Acheter quand le prix touche la bande basse.
- Vendre quand il touche la bande haute.
- Rester neutre au milieu.

```
from strategy_engine.AbstractStrategy import AbstractStrategy
import pandas as pd
```

```
class BollingerMeanReversionStrategy(AbstractStrategy):
```

```
    def __init__(
```

```

        self,
        window: int = 20,
        num_std: float = 2.0
    ):

        """
        Stratégie de retour à la moyenne basée sur les bandes de Bollinger.
        - window : taille de la fenêtre mobile
        - num_std : nombre d'écart-types pour les bandes
        """
        self.window = window
        self.num_std = num_std

    def generate_signals(self, data: pd.DataFrame) -> pd.DataFrame:
        df = data.copy()

        # Bande centrale : moyenne mobile
        df["middle"] = df["close"].rolling(self.window).mean()

        # Écart-type
        df["std"] = df["close"].rolling(self.window).std()

        # Bandes de Bollinger
        df["upper"] = df["middle"] + self.num_std * df["std"]
        df["lower"] = df["middle"] - self.num_std * df["std"]

        # Initialisation de la position cible
        df["target_position"] = 0

        # Règles :
        df.loc[df["close"] < df["lower"], "target_position"] = +1      #
acheter
        df.loc[df["close"] > df["upper"], "target_position"] = -1      # vendre

        # Retour au format attendu
        return df[["timestamp", "close", "target_position"]]

```

6. Démarrage rapide (Quick Start)

5.1 Lancement de l'application

Aller dans le répertoire `src` du projet `InvestIQ-v1` et exécuter la commande suivante:

```
python Main.py
```

Le moteur va :

- initialiser les moteurs (backtest, data, export),
- se connecter à Interactive Brokers (simulé ou réel selon configuration),

- exécuter la stratégie par défaut,
 - produire un fichier Excel dans `InvestIQ-v1\Backtest Logs\output.xlsx` dans lequel sont répertorié l'ensemble des positions exécutés par la stratégie.
 - produire un fichier de logs pour l'ensemble des moteurs du projet dans `InvestIQ-v1\Engine Logs\output.log`

5.2 Exemple d'utilisation

Logs console :

```
PS C:\Users\Manuel\Documents\tests\InvestIQ-v1> python Main.py
2025-12-03 16:55:53.027 Backtest InvestIQ.BacktestingEngine.Bootstrap | Initializing backtest engine...
2025-12-03 16:55:53.031 Backtest InvestIQ.BacktestingEngine.Bootstrap | Transition rules registered: 0 / 9
2025-12-03 16:55:53.034 Backtest InvestIQ.BacktestingEngine.Bootstrap | Transition strategies registered: 11 / 11
2025-12-03 16:55:53.036 Backtest InvestIQ.BacktestingEngine.Bootstrap | FIFO resolve strategies registered: 4 / 4
2025-12-03 16:55:53.037 Backtest InvestIQ.BacktestingEngine.Bootstrap | Round robin strategies registered: 5 / 5
2025-12-03 16:55:53.039 Backtest InvestIQ.BacktestingEngine.Bootstrap | FIFO execution strategies registered: 2 / 2
2025-12-03 16:55:53.039 Backtest InvestIQ.ExportEngine.Bootstrap | Backtest engine initialized successfully.
2025-12-03 16:55:53.040 Backtest InvestIQ.ExportEngine.Bootstrap | Initializing export engine...
2025-12-03 16:55:53.042 Backtest InvestIQ.ExportEngine.Bootstrap | Backtest export bindings: 1 / 1 registered.
2025-12-03 16:55:53.042 Backtest InvestIQ.ExportEngine.Bootstrap | Export engine initialized successfully.
2025-12-03 16:55:53.043 Backtest InvestIQ.TWS.Connection | Initializing historical data engine...
2025-12-03 16:55:53.043 Backtest InvestIQ.TWS.Connection | Connected to IB...
2025-12-03 16:55:53.450 Backtest InvestIQ.TWS.Connection | Connected to IB...
2025-12-03 16:55:53.451 Backtest InvestIQ.HistoricalData.Engine | Historical data engine initialized successfully.
2025-12-03 16:55:53.451 Backtest InvestIQ.InteractiveBroker.DataSource | Requesting Historical data...
2025-12-03 16:55:53.451 Backtest InvestIQ.InteractiveBroker.DataSource | Historical data request successful.
2025-12-03 16:55:53.809 Backtest InvestIQ.TWS.Connection | Disconnecting from IB...
2025-12-03 16:55:53.890 Backtest InvestIQ.TWS.Connection | Disconnected from IB
2025-12-03 16:55:53.899 Backtest InvestIQ.TransitionEngine state=FLAT event=GO_FLAT current=0.00 target=0.00 rule=NO_Operation strategy=NO_Operation transition=NO_OP actions=0 fifo_ops=0
2025-12-03 16:55:53.906 Backtest InvestIQ.TransitionEngine state=FLAT event=GO_LONG current=0.00 target=1.00 rule=OpenLongGrowth strategy=OpenLong transition=OPEN_LONG actions=1 fifo_ops=1
2025-12-03 16:55:53.913 Backtest InvestIQ.TransitionEngine state=LONG event=GO_LONG current=1.00 target=1.00 rule=AdjustShortFromLong strategy=NoOperation transition=NO_OP actions=0 fifo_ops=0
2025-12-03 16:55:53.913 Backtest InvestIQ.TransitionEngine state=LONG event=GO_SHORT current=1.00 target=1.00 rule=ReversalToLong strategy=ReversalToLong transition=REVERSAL_TO_LONG actions=2 fifo_ops=2
2025-12-03 16:55:53.913 Backtest InvestIQ.TransitionEngine state=SHORT event=GO_SHORT current=1.00 target=1.00 rule=AdjustShortFromShort strategy=NoOperation transition=NO_OP actions=0 fifo_ops=0
2025-12-03 16:55:53.916 Backtest InvestIQ.TransitionEngine state=LONG event=GO_LONG current=1.00 target=1.00 rule=AdjustLongFromShort strategy=NoOperation transition=NO_OP actions=2 fifo_ops=2
2025-12-03 16:55:53.916 Backtest InvestIQ.TransitionEngine state=LONG event=GO_SHORT current=1.00 target=1.00 rule=ReversalToShort transition=REVERSAL_TO_SHORT actions=2 fifo_ops=2
2025-12-03 16:55:53.917 Backtest InvestIQ.TransitionEngine state=LONG event=GO_SHORT current=1.00 target=1.00 rule=NoOperation transition=NO_OP actions=0 fifo_ops=0
2025-12-03 16:55:53.917 Backtest InvestIQ.TransitionEngine state=LONG event=GO_SHORT current=1.00 target=1.00 rule=ReversalToShortFromLong strategy=ReversalToShort transition=REVERSAL_TO_SHORT actions=2 fifo_ops=2
2025-12-03 16:55:53.918 Backtest InvestIQ.TransitionEngine state=LONG event=GO_SHORT current=1.00 target=1.00 rule=NoOperation transition=NO_OP actions=0 fifo_ops=0
2025-12-03 16:55:53.918 Backtest InvestIQ.TransitionEngine state=LONG event=GO_SHORT current=1.00 target=1.00 rule=ReversalToShortFromShort strategy=NoOperation transition=NO_OP actions=0 fifo_ops=0
2025-12-03 16:55:53.925 Backtest InvestIQ.TransitionEngine state=LONG event=GO_LONG current=1.00 target=1.00 rule=AdjustLongFromLong strategy=NoOperation transition=NO_OP actions=0 fifo_ops=0
2025-12-03 16:55:53.925 Backtest InvestIQ.TransitionEngine state=LONG event=GO_SHORT current=1.00 target=1.00 rule=ReversalToShortFromLong strategy=NoOperation transition=NO_OP actions=0 fifo_ops=0
2025-12-03 16:55:53.929 Backtest InvestIQ.TransitionEngine state=LONG event=GO_SHORT current=1.00 target=1.00 rule=ReversalToShortFromShort strategy=NoOperation transition=NO_OP actions=0 fifo_ops=0
2025-12-03 16:55:53.932 Backtest InvestIQ.TransitionEngine state=SHORT event=GO_SHORT current=1.00 target=1.00 rule=AdjustShortFromLong strategy=NoOperation transition=NO_OP actions=0 fifo_ops=0
2025-12-03 16:55:53.932 Backtest InvestIQ.TransitionEngine state=SHORT event=GO_SHORT current=1.00 target=1.00 rule=ReversalToLong transition=REVERSAL_TO_LONG actions=2 fifo_ops=2
2025-12-03 16:55:53.932 Backtest InvestIQ.TransitionEngine state=SHORT event=GO_SHORT current=1.00 target=1.00 rule=AdjustShortFromShort strategy=NoOperation transition=NO_OP actions=0 fifo_ops=0
2025-12-03 16:55:53.934 Backtest InvestIQ.BatchWriter Engine.WriterCore | Starting writer core and opening sink.
2025-12-03 16:55:53.934 Backtest InvestIQ.BatchWriter Engine.WriterCore | Writer core started.
2025-12-03 16:55:53.934 Backtest InvestIQ.BatchWriter Engine.WriterCore | Writer core started.
2025-12-03 16:55:53.934 Backtest InvestIQ.BatchWriter Engine.WriterCore | ExcelWriterCore starting (allocating buffer).
2025-12-03 16:55:53.934 Backtest InvestIQ.BatchWriter Engine.WriterCore | ExcelWriterCore starting (allocating buffer).
2025-12-03 16:55:53.935 Backtest InvestIQ.BatchWriter Engine.WriterCore | Dataframe successfully encoded to Excel bytes.
2025-12-03 16:55:53.966 Backtest InvestIQ.BatchWriter Engine.WriterCore | Dataframe successfully encoded to Excel bytes.
2025-12-03 16:55:53.966 Backtest InvestIQ.BatchWriter Engine.WriterCore | Dataframe successfully encoded to Excel bytes.
2025-12-03 16:55:53.967 Backtest InvestIQ.BatchWriter Engine.WriterCore | WriterCoreWriter encoded data.
2025-12-03 16:55:53.967 Backtest InvestIQ.BatchWriter Engine.WriterCore | WriterCoreWriter successfully encoded data.
2025-12-03 16:55:53.967 Backtest InvestIQ.Engine.Sink | Wrote 6880 bytes to sink.
2025-12-03 16:55:53.967 Backtest InvestIQ.BatchWriter Engine.WriterCore | Sink wrote successfully.
2025-12-03 16:55:53.967 Backtest InvestIQ.BatchWriter Engine.WriterCore | Sink successfully wrote encoded data.
2025-12-03 16:55:53.967 Backtest InvestIQ.BatchWriter Engine.WriterCore | Write 1 # succeeded.
2025-12-03 16:55:53.967 Backtest InvestIQ.BatchWriter Engine.WriterCore | Finalizing writer core with encoded data.
2025-12-03 16:55:53.967 Backtest InvestIQ.BatchWriter Engine.WriterCore | Finalizing ExcelWriterCore and clearing buffer.
2025-12-03 16:55:53.967 Backtest InvestIQ.BatchWriter Engine.WriterCore | WriterCoreWriter finalized successfully.
2025-12-03 16:55:53.968 Backtest InvestIQ.BatchWriter Engine.WriterCore | WriterCoreWriter finalized successfully.
2025-12-03 16:55:53.968 Backtest InvestIQ.BatchWriter Engine.WriterCore | Sink committed successfully.
2025-12-03 16:55:53.968 Backtest InvestIQ.BatchWriter Engine.WriterCore | Closing written from state BatchWriterState.COMMITTED
2025-12-03 16:55:53.968 Backtest InvestIQ.Engine.Sink | Closing committed sink - finalizing resources.
2025-12-03 16:55:53.968 Backtest InvestIQ.BatchWriter Engine.WriterCore | FileBatchSink finalized successfully.
2025-12-03 16:55:53.969 Backtest InvestIQ.BatchWriter Engine.WriterCore | Writer closed cleanly.
2025-12-03 16:55:53.969 Backtest InvestIQ.BatchWriter Engine.WriterCore | Export pipeline completed successfully.
Realized Pm : 167.75
```

Résultats Excel des positions prises en fonction de la stratégie et de la configuration du moteur :

7 Licence / disclaimers

- Ce projet est fourni à des fins éducatives

- Aucune garantie n'est donnée pour l'utilisation en trading réel.