Algorithme Ichimoku + ATR — Explications et réglages de seed Date: 2025-08-21

1) Vue d'ensemble $pipeline_web6$

- Données Binance $ccxt$, timeframe 2h, cache local `data/`.
- Stratégie systématique Ichimoku + trailing stop ATR, long/short, exécution réaliste $frais$, $funding/rollover$, $slippage$, $latence$, $haltes$, $margesBinance$.
- Optimisation par essaim génétique/Optuna $ASHA$ avec walk-forward annuel, score multi-critères $Sharpe/CAGR/MaxDD/Stabilité$.

Diagramme de flux:

```
flowchart LR
  A["Données Binance (ccxt)"] --> B["Cache local data/*.csv"]
  B --> C["Validation données (qualité, gaps, volumes)"]
  C --> D["Calcul Ichimoku + ATR"]
  D --> E["Backtest Long/Short (risk, coûts, exécution)"]
  E --> F["Métriques (CAGR, Sharpe, Calmar, VaR, etc.)"]
  F --> G["Exports CSV outputs/"]
  E -. Optimisation Optuna (ASHA, folds annuels) .-> D
```

2) Paramètres Ichimoku et logique de trading

- Composants (fenêtres N/M/K, décalage `shift`):
- Tenkan-sen: $Tenkan_N = (HH_N + LL_N/2)$
- Kijun-sen: $Kijun_M = (HH_M + LL_M/2)$
- Senkou Span B: $SenkouB_K = (HH_K + LL_K/2)$, tracé en avance de `shift` périodes.
- Entrées:
- Long: croisement Tenkan au-dessus de Kijun ET close > nuage (après décalage `shift`).
- Short: croisement inverse ET close < nuage (après décalage `shift`).
- Sorties:
- Croisement inverse OU trailing stop ATR.
- ATR période = max$14, Kijun$; trailing dynamique:
  - Long: $TS_t = \max(TS_{t-1}, Close_t - m \cdot ATR_t)$
  - Short: $TS_t = \min(TS_{t-1}, Close_t + m \cdot ATR_t)$
- Paramètres optimisés: `tenkan`, `kijun`, `senkou_b`, `shift`, `atr_mult`.

Référence code: `backtest_long_short()` et `run_profile()` dans `ichimoku_pipeline_web_v4_8_fixed.py`.

3) Gestion du risque $réglagescourants$

- Taille de position: 1% du capital par trade $positionunitaire$.
- Levier: 10×.
- Jusqu'à 3 entrées maximum par côté/symbole; jamais long et short simultanément sur un même symbole.
- Protections: stop global, haltes sur gaps extrêmes, vérification continue des données, limites/marges Binance réalistes.

4) Métriques clés

- Multiplicateur final, CAGR, Max Drawdown MDD, Calmar, Sharpe/Sortino, VaR 95%, proxy de stabilité $Lyapunov$.

5) Réglages de seed $reproductibilité$

- À quoi sert le seed ?
- Il fixe les générateurs aléatoires Python et NumPy: `random.seed(seed)` et `np.random.seed(seed)`.
- Il initialise l'exploration d'Optuna $samplerTPE$ et l'ordre des essais/pruning $ASHA$.
- Impacte: tirages de paramètres, populations initiales $sigénétique$, permutations/folds internes. N'impacte pas les données brutes.
- Où est-il appliqué ?
- `run_profile(..., seed=...)` et fonctions Optuna dans `ichimoku_pipeline_web_v4_8_fixed.py`.
- Comment le définir ?
- CLI direct: `bash python .\ichimoku_pipeline_web_v4_8_fixed.py pipeline_web6 --trials 5000 --seed 42 --out-dir outputs`
- Script complet $PowerShell$: `bash pwsh -NoProfile -File .\run_full_analysis.ps1 -ProfileName pipeline_web6 -Trials 1000 -Seed 999 -BaselineJson .\outputs\BEST_BASELINE.json -OpenReport`
- Démarrage rapide par seed: `bash pwsh -NoProfile -File .\run_seed_python.ps1 -ProfileName pipeline_web6 -Trials 5000 -Seed 123 -OutDir outputs -Label s123`
- Bonnes pratiques seed

- Tester quelques seeds canoniques $ex.:42, 123, 777, 999$ et comparer la stabilité $p50/p5MC, MDDmédiane, Sharpe.$
- Geler les versions des dépendances (`requirements.txt`) et conserver le cache `data/` pour une reproductibilité stricte.
- Archiver les sorties $HTML/CSV/JSON$ avec labels incluant le seed.

6) Notes de robustesse $aperçu$

- Monte Carlo $blockbootstrap$: évalue la sensibilité au chemin des rendements; privilégier des baselines au bon compromis p50/p5 et DD médiane.
- Stabilité $proxyLyapunov$: pénalise les dynamiques trop sensibles $(\lambda > 0)$.

7) Export de ce document en PDF

Pré-requis: Microsoft Edge ou Google Chrome installé.

Commande:

```
python .\scripts\export_docs_to_pdf.py --docs .\docs\ALGORITHME_ET_SEED_FR
```

Liens utiles: - Guide d'usage: `docs/USAGE.md` - Rapport exécutif: `docs/HSBC_REPORT_FR.md` - Formules & exemples: `docs/FORMULES_ET_EXEMPLES.md` - Logique technique: `LOGIQUE_PROGRAMME.md`