Projet de recrutement - Quantitative Trader

Premiums robustes, regimes de stress et phase transitions en crypto

Framework directement codable en 1 semaine pour futures/perps crypto : estimation robuste de premiums, detection de regimes (transitoire vs structurel) et regles de decision (Trade / Widen / Risk-off).

Themes : Premium Analysis (Robust Estimation) + Statistical Mechanics Applications.

Hook : un premium synthetique (BTCUSDC vs BTCUSDT) melange stablecoin basis et microstructure. On le rend exploitable en le de-biaisant, en filtrant les chocs, et en detectant les regimes avec des variables d'etat interpretables (H_t, T_t, chi_t).

Date : 2026-02-09 - Version : v5

1. Specification executable (contrat)

Section prescriptive : fixe les entrees, sorties, definitions, parametres par defaut, et l'ordre d'execution. C'est le point de reference pendant le coding.

1.1 Entrees minimales (data contract)

Champ	Type	Description / regles
timestamp_utc	datetime	UTC, monotone par symbole ; dedoublonner si besoin.
symbol	string	Ex: BTCUSDT-PERP, BTCUSDC-PERP (venue optionnel).
price	float	Mark price recommande ; sinon last/mid. Choisir 1 seule convention.
venue (optionnel)	string	Si multi-venues : Binance/Bybit/OKX, etc.
volume (optionnel)	float	Pour ponderations / diagnostics.

Normalisation recommandee :

- Resample cible : 1s ou 1m (choisir une seule frequence).
- Forward-fill limite (max 2 intervalles). Au-dela : missing.
- Filtrer glitches : returns absurdes (ex: > 20 sigma) et timestamps dupliques.
- Convertir raw -> Parquet des J1 pour iterer vite.

1.2 Sorties attendues (outputs contract)

Objet	Definition	Usage
p_naive(t)	log P(BTCUSDC) - log P(BTCUSDT)	diagnostic brut
log(USDT/USDC)_hat(t)	proxy cross-asset (mediane inter-actifs)	debias stablecoin
p(t)	p_naive(t) - log(USDT/USDC)_hat(t)	signal tradable
p_smooth(t)	filtre robuste de p(t)	niveau lisse
sigma_hat(t)	echelle robuste locale	standardisation
z_t	(p - p_smooth)/sigma_hat	evenements |z|>u
H_t, T_t, chi_t	entropie / temperature / susceptibilite	regimes + seuils adaptatifs
regime_t	transient vs stress (change-point)	gating / risk mode

1.3 Parametres par defaut (demarrage)

Parametre	Valeur	Rationale
resample	1s (ou 1m)	1s si data propre ; 1m si volume/iterations lourds
W (fenetre robuste)	3600s (1h)	stabilite / reactivite
u (seuil extremes)	3.0	evenements si |z| > 3
delta depeg	0.002 (log)	~20 bps
L depeg	5 minutes	eviter faux positifs
k entree	2.0	entrer si |m| > k*T

regimes	change-point	robuste et explicable

1.4 Ordre d'execution (run order)

- Ingestion & cleaning -> table alignee (UTC) au pas choisi.
- Construire p_naive(t) sur la paire cible.
- Estimer log(USDT/USDC)_hat(t) via replication cross-asset ; calcul p(t).
- Validation on-chain et depeg-flag (delta, L).
- Filtre robuste -> p_smooth, sigma_hat, z_t ; marquer evenements |z|>u.
- Calcul H_t, T_t, chi_t.
- Regimes : change-point -> regime_t.
- Regles de decision : Trade / Widen / Risk-off.
- Backtest + metriques + export figures.

Le notebook orchestre. Les algorithmes vivent dans src/ (anti-notebook spaghetti).

2. Architecture recommandee (repo + notebook)

Notebook = report (narration + figures). Modules src/ = calculs testables.

2.1 Arborescence type

```
repo/
README.md
AGENT.md
requirements.txt
configs/config.yaml
notebooks/01_report.ipynb
src/
data_ingest.py
premium.py
robust_filter.py
statmech.py
regimes.py
strategy.py
backtest.py
plots.py
data/raw/
data/processed/
reports/figures/
reports/tables/
```

2.2 Contrat des modules (qui produit quoi)

Module	Entrees	Sorties
data_ingest	raw files, resample	table alignee + logs QA
premium	prices + underlyings	p_naive, USDT/USDC_hat, p(t), depeg_flag
robust_filter	p(t), W	p_smooth, sigma_hat, z_t, events
statmech	z_t, m_t, W	H, T, chi
regimes	features/score	regime_t + segments
strategy	m, T, chi, regime	decision_t + thresholds
backtest	decision + prices	metrics.csv + trade log
plots	series + episodes	Figure1/2/3 exports

2.3 Notebook report : cellules minimales

- Config + episodes
- Load processed data
- Premium (debias + depeg flag)
- Robust filter (z_t + events)
- Stat-mech + regimes
- Decision rules
- Backtest + metrics
- Figures export

3. Decision logic (sans code)

```
Mispricing m_t : baseline m_t = p_smooth(t).
```

- Override securite : si depeg_flag -> Risk-off.
- Si regime=stress -> Risk-off (ou au minimum Widen).
- Si regime=transient : Trade seulement si |m| > k*T ; Widen si T ou chi eleves.

4. Episodes fixes + evaluation

Couvrir au moins 2 episodes ; idealement 4 (LUNA, FTX, USDC depeg, yen carry unwind).

Episode	Fenetre	Success criteria
LUNA/UST	9-13 May 2022	eviter drawdown (gating) ; decision stable
FTX	6-11 Nov 2022	regime stress detecte ; Sharpe_gated > naive
USDC depeg	10-11 Mar 2023	depeg_flag fires (delta,L) ; no mean-revert
Yen carry	5-6 Aug 2024	seuils adaptatifs ; flip-rate bas

Metriques a reporter :

- Sharpe_naive vs Sharpe_gated, max drawdown, turnover, PnL net.
- flip-rate, hit-rate, ablation (debias / robust / regime).

Figures :

- Fig1 : timeline (p_naive, p_smooth, regime, events).
- Fig2 : T and chi + episodes.
- Fig3 : phase space (T, |m|) decision regions.

5. Plan 7 jours (realiste)

- J1 : ingest + QA + Parquet + p_naive.
- J2 : debias + depeg flag (delta,L).
- J3 : robust filter + z_t + events.
- J4 : stat-mech + change-point regimes + decisions.
- J5 : backtest + metrics + 3 figures.
- J6 : polish + ablation ; HMM only if needed.
- J7 : packaging (README, notebook clean, results, limitations, Q&A;).

Fin.