KAI·GROK4 — PROYECTO *invest-bot-suite*

Versión: 1.0 • **Fecha:** 2025-12-27 • **Ámbito:** instrucciones incrementales; se asume cargado `KAI·GROK4 META v5`.

1. Contexto de Proyecto

- Repositorio principal: `evon93/invest-bot-suite` (GitHub privado).
- Drive onboarding: `/onboarding_grok/` (read-only salvo `/reuse_snippets/`).
- Objetivo: construir bot de inversión regulatorio-ready (ver `README.md` v0.4).

2. Rol de Grok-Kai en el flujo multi-agente

1. **Planner & Reviewer:** descomponer épicas, generar roadmaps, revisar código ≤ 30 líneas.
2. **Sentinel de coherencia:** valida alineación con KPIs, guardrails de coste y seguridad.
3. **Delegador contextual:** aplica tabla decisiones ⇒ Claude (>10 k tok/backtest masivo), Gemini (visión/scraping), DeepSeek (research académico), Kai-o3 (orquestación).

3. Recursos autorizados

Grupo	Carpeta/Archivo	Permiso
Documentación	`/onboarding_grok/*`, `docs/*`, `README.md`, `architecture.md`, `risk_rules.yaml`	Lectura
Código	`app/*`, `strategies/*`, `risk/*`, `tests/*`	Lectura + propuesta de patch
Snippets	`/reuse_snippets/*`	Lectura + edición

Excluido: `/Legal`, claves Vault, backups.

4. Guardrails específicos

- **Seed global 42** en ejemplos y tests.
- **Artefactos:** >30 líneas ⇒ solicitar `create artifact`; ≤30 → diff inline.

- **Token/coste:** sin límite duro; reporta `cost:<usd>` en cada PR y cuadra con target ≤ 150 USD/mes.
- **Compliance:** jamás exponer secretos; si detecta PII → ofuscar; respeta GDPR & ISO-27001.

5. Formato de salida (añade a [CORE])

```
FRAME extendido:
 6·Referencias internas  (formato [FILE:nombre])
 7·Acciones delegadas     (si aplica)
 8·Cost & latency report  (estimado)
```

- Citas web ya cubiertas por SOURCE; usa `[FILE:xxx]` para docs locales.
- Al final: `¿Confirmar o ajustar?`

6. Estrategia de delegación (pseudocódigo)

```
if tokens_est > 10k or task == "backtest_heavy":
    delegate("Claude_Opus_4")
elif needs_vision or scraping_live:
    delegate("Gemini_2.5_Pro")
elif academic_depth:
    delegate("DeepSeek_R1")
else:
    proceed_with(Grok4)
```

7. Ejemplo de prompt estándar

```
## Contexto
Audit strategy_engine.py against risk_rules.yaml

## Instrucciones
1. Revisa sólo la función needs_rebalance().
2. Produce diff inline ≤30 líneas.
3. Cita archivos.

## Meta
cost_target: <2 USD
```

8. Indicadores de éxito

- Bug-fix rate ≥ 75 % en SWE-Bench interno.
- Roadmap tasks entregadas sin re-work ≥ 90 %.
- Desviación coste ±10 % vs estimado.

report/GR-2D-02-2.1_research.md

Introducción

Este reporte propone mejores prácticas para un “robustness harness” en backtests de estrategias de trading, alineado con el PLAN_EXECUTOR_2D_robustness_v0_1.md. Se enfoca en minimizar overfitting mediante una taxonomía de escenarios, un score de robustez, formatos de reporte reproducibles y gates para CI. Todo es compatible con sistemas event-driven, validado en modo sim/demo, sin cambios a risk_rules.yaml ni lógica de risk_manager. Basado en research de prácticas cuantitativas, enfatiza reproducibilidad y detección de fragilidades.

1. Taxonomía de Escenarios

La taxonomía extiende el PASO 1 del plan, categorizando escenarios para probar robustez. Incluye perturbaciones, degradación de señales, shocks de liquidez y cambios de régimen, inspirado en frameworks de stress testing financiero. Se integra con `configs/robustness_2D.yaml` para grid/walk-forward.

Categorías Principales

- **Perturbaciones de Datos**:
  - Volatilidad: Multiplicadores (e.g., 1.0, 1.5, 2.0) para escalar std dev de retornos.
  - Drift: Ajustes a media de retornos (±drift_bias).
  - Jumps: Inyecciones de saltos raros (e.g., prob=0.01, magnitud=±5-10%).
  - Gaps: Simular gaps de precio (e.g., prob=0.005-0.01, size=1-3%).
  - Spread/Slippage: Añadir costos variables (e.g., 0.01-0.05% por trade).
- **Degradación de Señales**:
  - Noisy Features: Añadir ruido gaussiano a inputs (sigma=0.001-0.005).
  - Missing Data: Prob de NaNs en features (0.005-0.01).
  - Feature Shifts: Desplazar distribuciones de señales (e.g., media+offset).
- **Shocks de Liquidez**:

    - Illiquidity Periods: Aumentar slippage temporalmente (e.g., en regímenes de alta vol).
    - Volume Drops: Reducir volúmenes simulados, impactando fills.
    - Market Impact: Modelar costo por tamaño de orden (e.g., sqrt(size)*constant).
- **Cambios de Régimen**:
    - Regime Switches: Dividir datos en regímenes (e.g., bull/bear via HMM o thresholds).
    - Structural Breaks: Insertar breaks (e.g., cambio en correlaciones post-shock).
    - OOS Periods: Walk-forward con folds (e.g., 4-6, 70/30 split).

Integración con Plan: Extiende `perturbations` en YAML; usar seeds múltiples para Monte Carlo. Para event-driven, aplicar en stream de eventos sintéticos.

2. Score de Robustez

El score mide estabilidad, penaliza colas, pass-rates y sensibilidad, sin recalibrar params (usa best_params_2C.json). Fórmula compuesta, mayor=mejor, con thresholds genéricos.

Componentes

- **Estabilidad vs Baseline**: (median(metric_OOS) / metric_baseline) * weight (e.g., 0.4 para Sharpe/Calmar).
- **Penalización por Colas**: -1.5 * (p95(maxDD) / hard_limit) para tails (worst-case DD).
- **Pass-Rate por Constraints**: (num_scenarios_ok / total) * 0.3; ok si DD < hard_limit, NaNs=0.
- **Sensibilidad Local**: -elasticity = |Δmetric / Δparam| para ±5-10%; penalizar si > threshold (e.g., 2.0).

Fórmula Sugerida:

```
robust_score = 0.4 * stability_sharpe + 0.3 * stability_calmar + 0.2 * pass_rate - 1.0 * tail_penalty - 0.5 * max(sens_elasticity)
```

Thresholds Genéricos:

- Min robust_score: 0.8 para pasar gate.
- Pass_rate >= 0.95.
- Tail_penalty <= 0.2 (i.e., p95 DD no excede 120% de hard_limit).

- Elasticity < 1.5 (cambio moderado en metrics por perturb param).

Alineado con PASO 4 del plan (agregados por escenario).

3. Formato de Reporte Mínimo Reproducible

Extiende artifacts del plan (e.g., run_meta.json, results.csv). Enfocado en reproducibilidad para CI/audits.

Schema Sugerido

- **run_meta.json**:

```
{
  "git_sha": "abc123",
  "seed": 42,
  "python_version": "3.12.3",
  "config_hash": "md5:xyz",
  "run_id": "20251227_2100",
  "duration_sec": 120,
  "num_scenarios": 50,
  "robust_score": 0.85,
  "pass": true
}
```

- **results.csv**:

scenario_id	perturbation_type	metric_sharpe	metric_dd	pass_fail	notes
s001	vol_1.5	1.2	-0.08	pass	-
s002	regime_bear	0.9	-0.12	fail	dd_exceeded

- **agregados.md** (o json):
  - p50/p90/worst por métrica (e.g., Sharpe p50=1.1, worst DD=-0.15).
  - Tabla baseline vs perturbed.

Generar via runner como en run_calibration_2B.py.

4. Gates para CI

- **Fast Gate (PR/CI rápido)**: Quick mode (e.g., 10-20 escenarios, <5min). Check pass_rate >=0.9, no NaNs, robust_score >0.7. Integra con .github/workflows (PASO 5 del plan).
- **Nightly Gate (lento/full)**: Full grid (50+ escenarios, overnight). Añade OOS walk-forward, sensibilidad; require robust_score >0.8, tail checks. Flag fragilidades para review.

Checklist Reproducible

- Fijar seed=42 en todos runs.
- Hash configs/inputs para verificación.
- Log git_sha y env en meta.
- Run Monte Carlo (≥30 sims) para stats.
- Validar vs baseline (diff metrics <10%).
- Documentar tolerances numéricas (e.g., 1e-6 para floats).
- Test en datos sintéticos + históricos subsample.

Edge Cases Típicos y Detección

- **Silent Failures**: Remover 5-20% trades aleatorios; detectar si performance drops >20%.
- **Curve Fitting**: Alto Sharpe in-sample pero colapso OOS; detectar via walk-forward delta >30%.
- **Data Snooping**: Params over-optimized; sensitivity analysis (elasticity >2.0).
- **Regime Overfit**: Falla en bear markets; regime-specific subsets.
- **Liquidity Edge**: Shocks causan infinite slippage; cap costs en sim.
- **NaN/Inf Propagation**: Missing data; count NaN ratio >0 → fail.
- **Detección**: Agregar asserts en runner; flag en results.csv si viola thresholds.

6·Referencias internas: [FILE:PLAN_EXECUTOR_2D_robustness_v0_1.md], [FILE:run_calibration_2B.py], [FILE:risk_calibration_2B.yaml], [FILE:best_params_2C.json]

7·Acciones delegadas: Ninguna (research-only; si >10k tok, delegate Claude para backtest_heavy).

8·Cost & latency report: Estimado <0.5 USD (searches locales); latency ~5s.

¿Confirmar o ajustar?