# Arquitectura de orquestación TRM:
Sistemas dinámicos heterocedásticos

Santiago Javier Espino Heredero

15 de octubre de 2025

**Resumen**

Se presenta una propuesta formal para una arquitectura de inteligencia artificial basada en la orquestación de $n$ Modelos Recursivos Pequeños (Tiny Recursive Models, TRM) interconectados, diseñada para abordar problemas científicos de alta complejidad en dominios como mecánica estadística, dinámica de fluidos, biología molecular, genética (CRISPR) e inmunología. Se desarrollan fundamentos matemáticos, estructura arquitectónica, estrategias de entrenamiento, análisis de estabilidad, implementación práctica y lineamientos para validación interdisciplinaria y ética.

## Índice

# 1. Resumen Ejecutivo

La propuesta consiste en un sistema compuesto por $n$ módulos TRM, cada uno capaz de realizar iteraciones internas para resolver sub-problemas locales. Los módulos se comunican mediante canales de mensajes y convergen hacia soluciones cooperativas mediante iteración conjunta (síncrona o asincrónica).

**Ventajas:**

- Eficiencia computacional: pequeños módulos reemplazan a un modelo monolítico.
- Modularidad y explicabilidad.
- Escalabilidad mediante sparsidad y paralelismo.
- Incorporación de conocimiento experto específico.

**Riesgos:**

- Inestabilidad dinámica (oscilaciones, divergencia).
- Dificultad en el entrenamiento conjunto.
- Dependencia alta de datos en dominios biológicos.

# 2. Planteamiento Matemático

## 2.1. Modelo Local TRM

Cada TRM $i$ con parámetros $\theta_i$ actualiza un estado latente $s_i^{(r)}$ mediante:

$$s_i^{(r+1)} = f_{\theta_i}(s_i^{(r)}, m_i^{(r)}, x_i) \tag{1}$$

Donde $x_i$ son observaciones locales y $m_i^{(r)}$ el mensaje recibido. El resultado final del módulo es $o_i = g_{\phi_i}(s_i^{(R)})$.

## 2.2. Mensajería Inter-Módulo

Definimos $M^{(r)} \in \mathbb{R}^{n \times d_m}$ con emisiones $u_j^{(r)} = \text{emit}(s_j^{(r)})$. Entonces:

$$m_i^{(r)} = \mathcal{A}(\{u_j^{(r)} : j \in \mathcal{N}(i)\}; W) \tag{2}$$

Por ejemplo, mediante atención:

$$m_i^{(r)} = \sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(r)} \, \text{proj}(u_j^{(r)}), \quad \alpha_{ij}^{(r)} = \text{softmax}_j(q(s_i^{(r)}) \cdot k(u_j^{(r)})) \tag{3}$$

## 2.3. Función de Pérdida Global

$$\mathcal{L} = \sum_{i=1}^{n} \lambda_i \ell_i(o_i, y_i) + \gamma \mathcal{R}(S^{(0:R)}) \tag{4}$$

Donde $\mathcal{R}$ es una regularización espectral o de consistencia entre módulos.

# 3. Arquitectura Propuesta

La topología global se define sobre un grafo dirigido $G = (V, E)$ con $|V| = n$. Los módulos TRM están organizados jerárquicamente y pueden compartir una memoria común. La comunicación se realiza mediante canales compactos y asincrónicos. Un orquestador superior controla iteraciones, pesos y asignación de tareas.

# 4. Estrategias de Entrenamiento

## 4.1. Unrolling BPTT

Entrenamiento explícito desenrollando $R$ pasos con retropropagación completa.

## 4.2. Modelos Implícitos (DEQ)

Enfoque de punto fijo $S^* = F_\Theta(S^*, X)$, diferenciado implícitamente. Ahorra memoria y estabiliza la convergencia.

## 4.3. Entrenamiento por Bloques

Actualización alternada de subconjuntos de módulos; favorece colaboración con expertos.

## 4.4. Diseño de Pérdidas

Cada TRM tiene pérdida local $\ell_i$, y una pérdida cooperativa de consistencia. Se incluye una pérdida heterocedástica:

$$\ell_i = \frac{1}{2} \sum_t \left( \frac{(y_i - \mu_i)^2}{\sigma_i^2} + \log \sigma_i^2 \right) \tag{5}$$

# 5. Análisis de Estabilidad

Sea $F_\Theta$ el mapeo global. Se busca contractividad:

$$\exists \rho < 1 : \ |F(S_1) - F(S_2)| \leq \rho |S_1 - S_2| \tag{6}$$

Control mediante normalización espectral y regularización del Jacobiano.

# 6. Implementación Práctica

```
# PyTorch pseudocode for TRM orchestrator
class TinyTRM(nn.Module):
    def __init__(...): ...
    def forward_step(...): ...
class Orchestrator(nn.Module):
    def forward(...): ...
```

El código implementa un sistema con $n$ TRMs interconectados, iterando mensajes hasta converger.

# 7. Evaluación Experimental

## 7.1. Métricas Generales

RMSE, NLL, residual $|S^{r+1} - S^r|$, calibración, eficiencia comunicacional, interpretabilidad.

## 7.2. Métricas por Dominio

- Mecánica estadística: error en observables, conservación energética.
- Fluidos: error L2 en campos, divergencia nula.
- Genética: AUPR/AUC en off-targets.
- Inmunología: ROC/AUC en binding predictivo.

## 8. Datos y Requisitos Computacionales

Cada TRM consume datos locales sincronizados. ETL debe garantizar coherencia temporal y normalización. Requiere GPU media (24–48GB) o clúster multi-GPU.

## 9. Riesgos, Ética y Colaboración Interdisciplinaria

Se definen especialistas y preguntas específicas por área (biología, inmunología, fluidos, ética) para validación experimental y revisión de riesgos.

## 10. Plan de Trabajo

- Fase 0: prototipo toy (4 TRMs, PDE 1D).
- Fase 1: solver implícito + regularización.
- Fase 2: piloto de dominio (CFD o bio).
- Fase 3: validación experimental.

## 11. Líneas Futuras

Estudio de acoplamientos óptimos, multiestabilidad, transferencia entre dominios y explicabilidad física.

## 12. Conclusiones y Recomendaciones

1. Prototipar orquestador (n=4, d=64).
2. Regularizar singular values.
3. Añadir cabeza heterocedástica.
4. Validar OOD y adversarial.
5. Consultar especialistas antes de despliegue biomédico.