# Fast weight programmers (FWP)

Implementation

*Schmidhuber 1993*

Evan Drake 2024

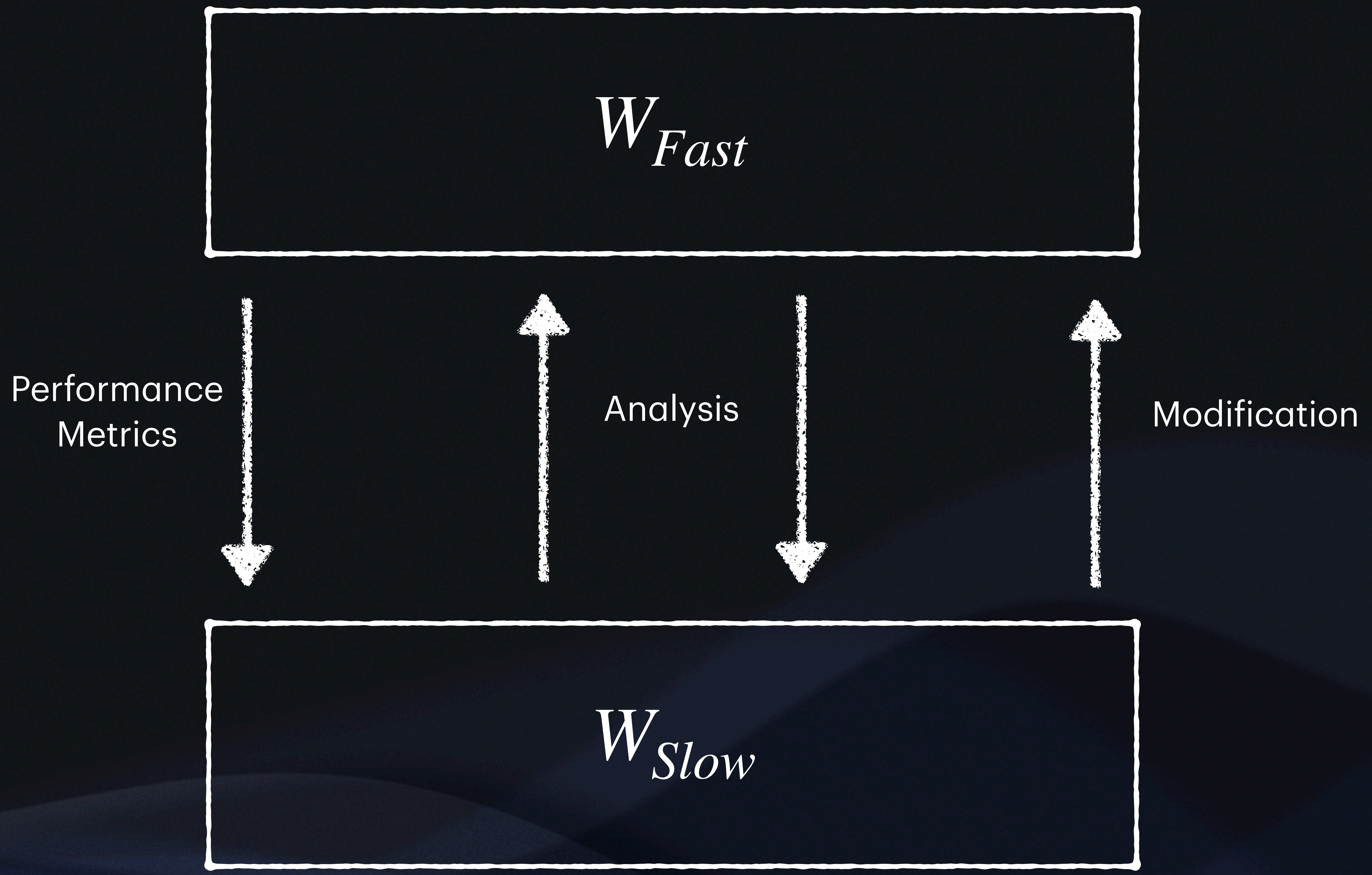
## What is a FWP?

## What is a FWP?

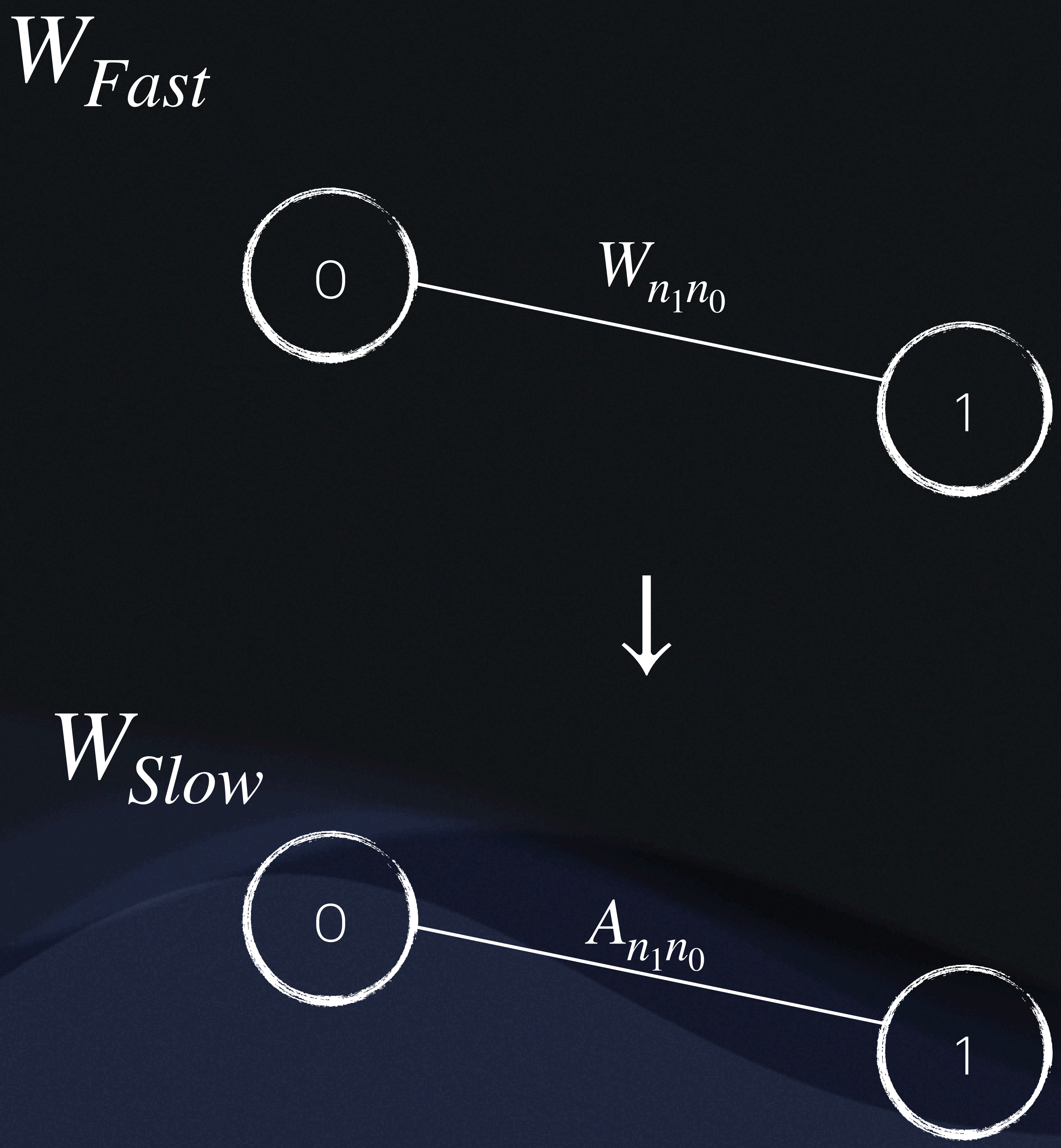
- Neural network
- Introspective
- Self modification

Meta learning

## Visualizing a FWP



$\Rightarrow$



Irie et al. (2022)

## What is a FWP

- Activations
  - Nodal traffic per time step
- Temporal analysis
  - Four dimensional structure

## Why?

## Why are FWPs useful?

- Expands accuracy of temporally bound networks
- Broadens predictable topics of analyzation
- Problem moves us towards understanding Gödel problem of self reflection.

Its a cool problem!

## *Meta learning tangent...*

Sorry professor...

## How?

## How does one build a FWP?

- Fast weight matrix
  - Design to work around an input vector x
  - Design to work around an output vector o
- Slow weight matrix
  - Design to map FWM weighted by activations rather than resistance
  - Design to analys and modify FWM

## How does this relate to Graph theory

- Neural networks
- Graph rewiring
- Activation is based on path traversal
- Combinatoric activation analysis (gradient)

# Axiomatic functions

- One neural network $\{W_{Slow}, W_{Fast}\}$
- Some $\Delta \mid \Delta : time \rightarrow W_{Slow}, W_{Fast} -> W'_{Fast}$
- Analysis & modification units are derived
  - $w_{1,0} \in W_{Fast}, a_{1,0} \in W_{Fast} \mid ana : time \rightarrow \int \quad w_{1,0} - a_{1,0} \quad dt \rightarrow ?$

## Problems (so far...)

- Language decisions
- Fine grain math conversion
- Testing methods (speed, efficacy)
- Bias of testing methods

| Haskell | Python |
|---|---|
| Functional (easier math) | Imperative (harder math) |
| Library scarcity | Library support |
| Curried typing | Unwieldy type system |
| Easy axiomatic build cycles | Easy MVP build cycles |

## Questions?