**Automated MOSFET Multi-Objective Optimisation using LTspice and Python**

**⚡️ Project Overview**

This document describes a powerful automation script designed for **multi-objective optimisation** of MOSFET parameters (Channel Length and Width ) within an **LTspice** simulation environment.

The core objective is to find the optimal and values that **maximise the transient peak current ()** while **minimising the transistor area ()**. This is solved using a weighted cost function minimised by a global optimisation algorithm.

The project demonstrates a robust, automated workflow for integrating industry-standard SPICE simulation tools with advanced Python scientific libraries for parameter optimisation in analog/mixed-signal design.

**⚙️ Technical Approach & Features**

The optimisation loop uses Python to control the simulation, eliminating the need for manual parameter sweeping in LTspice.

**Key Components**

1. **LTspice Integration:** The script programmatically generates a new **Netlist** file (optimization\_transient.net) for each optimisation iteration, inserting the new and parameters.
2. **Batch Simulation:** LTspice is executed in **batch mode (-b flag)** via subprocess to perform the transient simulation and generate the binary .raw file.
3. **Data Extraction:** The **PyLTSpice** library is used to read the raw simulation data, and **NumPy** extracts the critical performance metric: the maximum transient load current, .
4. **Multi-Objective Cost Function:** The optimiser minimises a cost function that balances performance () and cost (Area): $$ \text{Cost} = (-I\_{on}) + (\text{AREA\_PENALTY\_FACTOR} \cdot L \cdot W) $$ (Where is maximised by minimising )
5. **Differential Evolution:** The **SciPy differential\\_evolution** algorithm is employed for robust, global optimisation across the design search space.
6. **Visualization:** Generates high-quality plots showing the convergence of the cost function, the evolution of and over iterations, and a 3D visualization of the search path.

**Design Under Test (DUT)**

The Netlist template defines a basic NMOS switch circuit with a parasitic inductor () on the supply line to introduce dynamic effects, making the transient an important metric.

Vdd Vdd\_clean 0 DC 5

Lp Vdd\_clean Vdd\_supply 1n

Vg G 0 PULSE(0 5 0 1n 1n 1u 2u)

Rload Vdd\_supply D 1k

M1 D G 0 0 NMOS L={L\_val}u W={W\_val}u

.model NMOS NMOS (VTO=0.7 KP=120u)

.tran 0.1u 5u

**🚀 Getting Started**

**Prerequisites**

1. **LTspice:** Must be installed on your machine.
2. **Python 3.x:** Installed environment.

**Installation**

1. Clone this repository:
2. git clone [https://github.com/YourUsername/RepoName.git](https://github.com/YourUsername/RepoName.git)
3. cd RepoName
4. Install the required Python packages:
5. pip install numpy scipy matplotlib PyLTspice

**Configuration**

You must update the path to your LTspice executable in the Python script:

In optimize\_mosfet\_v5.py:

# IMPORTANT: Update this path to match your local LTspice installation executable path.

LTSPICE\_PATH = r"C:\Users\Kris\AppData\Local\Programs\ADI\LTspice\LTspice.exe"

**Running the Optimisation**

Execute the main script:

python optimize\_mosfet\_v5.py

The script will print the real-time progress of the differential evolution algorithm to the console, showing the , , , and Cost for each simulation run.

Upon completion, the final optimal parameters and performance metrics will be displayed, and the file optimization\_results.png will be generated with the convergence plots.