Chapter 1: Admittance Gradient Method

1.1 Motivation

Gradient methods in optimization seek the fastest improving direction in the solution space. The methods for extracting a vector of improving given solution can be deterministic (e.g. derivation of a cost function) or stochastic (e.g. Hill Climbing). Using gradient method for circuit sizing has been primarily of the stochastic variety, mainly because of the complexity of the equations and lack in confidence in sizing factors influence on performance properties. Hill Climbing method, for example, attempts to draw an improving direction from a random sample of direction and thus ratchet the circuit closer to a feasible solution. It cannot guarantee speed of convergance or convergance at all. This method, albeit potentially slow, is useful in obtaining a feasible solution for a given spec. For the application of obtaining a large set of solutions that map the size/performance Pareto fronts, a faster method for improving a given circuit is necessary. The ΓVM needs to make a probable guess of a vector that points a circuit to a better value of a given property. The ΓVM compiler has an advantage of performing a symbolic modified nodal analysis (herein: MNA). The equations that translate down the compilation cascade into ΓVM instructions can be used to analyze the value of each property as well as the gradient of that property in the solution space.

1.2 Example Circuit

A common-source nmos circuit has the following MNA matrix:

$$\begin{bmatrix} 0 & 0 & 0\,0\,1 \\ gm_{M1} & go_{M1} & 0\,0\,0 \\ 0 & 0 & 0\,1\,0 \\ 0 & 0 & 1\,0\,0 \\ 1 & 0 & 0\,0\,0 \end{bmatrix} * \begin{bmatrix} V_{IN} \\ V_{OUT} \\ V_{DD} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ I_{SOURCE}\text{-}Id_{M1} \\ 0 \\ V_{DD} \\ V_{IN} \end{bmatrix}$$

The equation ΓVM compiles to calculate V_{OUT} operating point is:

$$V_{OUT} = \frac{gm_{M1}V_{IN} + I_{SOURCE} - Id_{M1}}{-go_{M1}}$$

The DC Gain of the circuit is:

$$A_{DC} = \frac{\partial V_{OUT}}{\partial V_{IN}} = -\frac{gm_{M1}}{go_{M1}}$$

How can ΓVM increase the gain?

In other words, how does A_{DC} depend on the length and width of the M1 transistor?

$$\frac{\partial A_{DC}}{\partial L_{M1}} = \frac{\partial A_{DC}}{\partial gm_{M1}} * \frac{\partial gm_{M1}}{\partial L_{M1}} + \frac{\partial A_{DC}}{\partial go_{M1}} * \frac{\partial go_{M1}}{\partial L_{M1}} = \frac{-1}{go_{M1}} * \frac{\partial gm_{M1}}{\partial L_{M1}} + \frac{gm_{M1}}{go^2_{M1}} * \frac{\partial go_{M1}}{\partial L_{M1}}$$

Similarly:

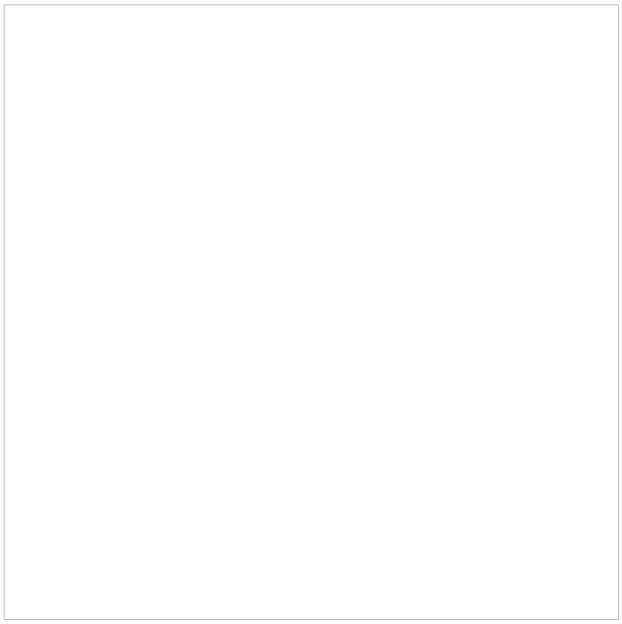
$$\frac{\partial A_{DC}}{\partial W_{M1}} = \frac{\partial A_{DC}}{\partial gm_{M1}} * \frac{\partial gm_{M1}}{\partial W_{M1}} + \frac{\partial A_{DC}}{\partial go_{M1}} * \frac{\partial go_{M1}}{\partial W_{M1}} = \frac{-1}{go_{M1}} * \frac{\partial gm_{M1}}{\partial W_{M1}} + \frac{gm_{M1}}{go^2_{M1}} * \frac{\partial go_{M1}}{\partial W_{M1}}$$

gm and go are obtained from the nmos LUT look up and interpolation.

The derivatives

$$\frac{\partial gm_{M1}}{\partial W_{M1}}, \frac{\partial go_{M1}}{\partial W_{M1}}, \frac{\partial gm_{M1}}{\partial L_{M1}} \text{and} \frac{\partial go_{M1}}{\partial L_{M1}}$$

are also looked up in a modified interpolation algorithm:



The modified Lagrange algorithm weighs every dimension as before, but for each dimension it forks a side interpolation sequence that begins with subtracting the two sides of the hypercube instead of weighing them. The result is the original interpolated value in addition to the derivative per dimension.

There are some fundamental flaws to this approach:

1. The actual gradient of the circuit's property (A_{DC} in the example) contains more components than the admittance values direct L and W derivatives. In a multi-transistor circuit, gm and go values of one transistor may have some dependence on the L and W of another transistor. The reason for that is that a change in the geometry of the other transistor may change the operating point of the first one and thus change the latter's gm and go. This method in only admittance-gradient based, rather than full-gradient based, which requires more calculations.

2. The Newton Raphson method can't be applied to the found gradient, because the conditions of function definition, diferentiability and smoothness are not met in the global sense of the circuit's behaviour. Stepping with a full or even too large fraction of the NR step can easily throw the some of the transistors to the cut-off region, where derivatives are not useful for further steps.

3. Only properties that can be estimated statically and using algebraic methods of pre-compiling and pre-deriving rational functions can be measured and improved. THD, for instance, is outside of the scope of the ΓVM Pareto mapper.

However, the method has two benefits for the ΓVM Pareto application:

1. It generates an improvement step without the need to fully calculate multiple circuits' operating points and performance properties. This method of using only LUT's outputs and few pre-compiled equations is magnitudes faster than using SPICE simulations and scripts tailored to measure circuit performance properties.

2. Although imperfect, the calculated step is likely to improve the performance of the circuit. The Pareto mechanism is designed to filter out useless circuits, so an occasional degredation of performance (in case OP effects are two large to neglect) does not detract from the overall quality of results.

Produce:

1. Rate of improving vs. degrading gradients calculated by AG

2. Run time figures of AG steps vs. BSIM convergance

3. A list of improveable properties

Cite: Optimization of custom MOS circuits by transistor sizing