Two-stage OpAmp design

- Cascode opamp: Hard to determine transistor ratio
- Two-stage opamp: Hard to determine current partition.
- Given a bandwidth spec, how to find the minimum power two-stage opamp design?
  - What are the variables?
Minimizing Power

• Assume transistor ratios within a stage is known.
  • Easy to determine by specifying/playing with $V^*$

• We basically have 3 variables, $I_1$, $I_2$, and $C_f$.

• Which variable should we determine first? Why?
OpAmp TF v.s. $C_f$

- We find $C_f$ first, because it doesn’t change total power.
- As $C_f$ increase, $f_{UG}$ monotonically decreases and phase margin improves.
  - Meaning binary search works well.
- Given an amplifier, We always want to use minimum $C_f$. 
Unbounded binary search

• Start with $C_f = C_{min}$ (i.e. 0.1fF).
• For current $C_f$, check if amplifier is stable.
• If not stable, update $C_{min}$, then double $C_f$.
  • Safety: if $C_f$ too big (i.e. 1uF), raise Exception.
• If stable, we found an upper bound; transition to binary search.
• Implemented with bag.util.search.FloatBinaryIterator
OpAmp current partitioning

• Now that we know how to find $C_f$, how do we partition the current?

• Let’s just sweep and plot some curves…

• Set $I_1 = I_{\text{min}}$. For each $K = I_2/I_1$, find optimal $C_f$ that meets phase margin spec, then plot the performance.

• Why does increasing stage 2 size make sense?
Specs v.s. $K$

- $f_{UG}$ is unimodal (goes up then down).
- Given $I_1$, There exists a maximum achievable $f_{UG}$.
- This maximum seems to occur when $C_f$ hits minimum.
Specs with/without Miller comp.

• At some critical $K$ value, the first pole is low enough that the system is stable.
  • Increase $K$ beyond this point only slows down the system.
• For $K$ value below the critical value, Miller comp. trades speed with stability.
2-stage OpAmp design v.1

• Start with $I_1 = I_2 = I_{\text{min}}$.

• Determine $C_f$ that stabilizes opamp, and get performance. If $f_{UG}$ meets spec, done. If not, increase $I_2$.

• If increasing $I_2$ decreases $f_{UG}$, Increase $I_1$ and start over.

• Drawbacks:
  • $O(N^2 \log_2 N)$ runtime.
  • Resolution limited by current step size.

• Can we do better?
Golden-Section Search.

- Finds the max/min of unimodal functions using only $O(\log N)$ function calls.
  - Requires 3 data points to start.
  - Data points partitioned using golden ratio.
2-stage OpAmp design

• Start with $I_1 = I_2 = I_{\text{min}}$.
• Search for $I_2$ using Fibonacci search.
• If past critical point, change to use golden section search.
• If maximum found and it’s less than spec, increase $I_1$ using binary search.
• Run time: $O(\log^3 N)$
• Example implementation: 
  bag'util'search'minimize_cost_golden_float()