Generalized Geometric Programming for Circuit Design

Stephen Boyd Seung Jean Kim

4/4/05

Outline

- Basic approach & applications
- Geometric programming & generalized geometric programming
- Digital circuit design applications
- Conclusions

Basic approach

1. formulate circuit design problem as **geometric program** (GP) or **generalized geometric program** (GGP), optimization problems with special form

2. solve GP or GGP using specialized, tailored method

- this talk focuses on step 1 (a.k.a. **GP modeling**)
- step 2 is **technology**

Applications

- wire and device sizing using Elmore delay
- digital circuit sizing and extensions (focus of this talk)
- analog and mixed signal design
  - opamps, comparators
  - ADCs, DACs, PLLs, SC filters
- RF design
  - CMOS inductors, oscillators
  - LNAs, mixers
- optimal doping profiles

Monomial & posynomial functions

$x = (x_1, \ldots, x_n)$: vector of positive optimization variables

- function g of form

$$g(x) = c x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n},$$

  with $c > 0$, $\alpha_i \in \mathbf{R}$, is called **monomial**

- sum of monomials, $i.e.$, function f of form

$$f(x) = \sum_{k=1}^{t} c_k x_1^{\alpha_{1k}} x_2^{\alpha_{2k}} \cdots x_n^{\alpha_{nk}},$$

  with $c_k > 0$, $\alpha_{ik} \in \mathbf{R}$, is called **posynomial**

Examples

with x, y, z variables,

- 0.23, $2z\sqrt{x/y}$, $3x^2y^{-.12}z$ are monomials (hence also posynomials)
- $0.23 + x/y$, $2(1+xy)^3$, $2x+3y+2z$ are posynomials
- $2x+3y-2z$, $x^2+\tan x$ are neither

Generalized posynomials

f is a **generalized posynomial** if it can be formed using addition, multiplication, positive power, and maximum, starting from posynomials

examples:

- $\max\left\{1 + x_1, 2x_1 + x_2^{0.2}x_3^{-3.9}\right\}$
- $\left(0.1x_1x_3^{-0.5} + x_2^{1.7}x_3^{0.7}\right)^{1.5}$
- $\left(\max\left\{1 + x_1, 2x_1 + x_2^{0.2}x_3^{-3.9}\right\}\right)^{1.7} + x_2^{1.1}x_3^{3.7}$

Composition rules

- **monomials** closed under product, division, positive scaling, power, inverse
- **posynomials** closed under sum, product, positive scaling, division by monomial, positive integer power
- **generalized posynomials** closed under sum, product, max, positive scaling, division by monomial, positive power

Generalized geometric program (GGP)

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 1, \quad i = 1, \ldots, m \\ & g_i(x) = 1, \quad i = 1, \ldots, p \end{array}$$

f_i are **generalized posynomials**, g_i are monomials

- called **geometric program (GP)** when f_i are **posynomials**
- a highly nonlinear constrained optimization problem

GP example

- maximize volume of box with width w, height h, depth d
- subject to limits on wall and floor areas, aspect ratios h/w, d/w

$$\begin{array}{ll} \text{maximize} & hwd \\ \text{subject to} & 2(hw+hd) \leq A_{\text{wall}}, \quad wd \leq A_{\text{flr}} \\ & \alpha \leq h/w \leq \beta, \quad \gamma \leq d/w \leq \delta \end{array}$$

in standard GP form:

$$\begin{array}{ll} \text{minimize} & h^{-1}w^{-1}d^{-1} \\ \text{subject to} & (2/A_{\text{wall}})hw + (2/A_{\text{wall}})hd \leq 1, \quad (1/A_{\text{flr}})wd \leq 1 \\ & \alpha h^{-1}w \leq 1, \quad (1/\beta)hw^{-1} \leq 1 \\ & \gamma wd^{-1} \leq 1, \quad (1/\delta)w^{-1}d \leq 1 \end{array}$$

Trade-off analysis

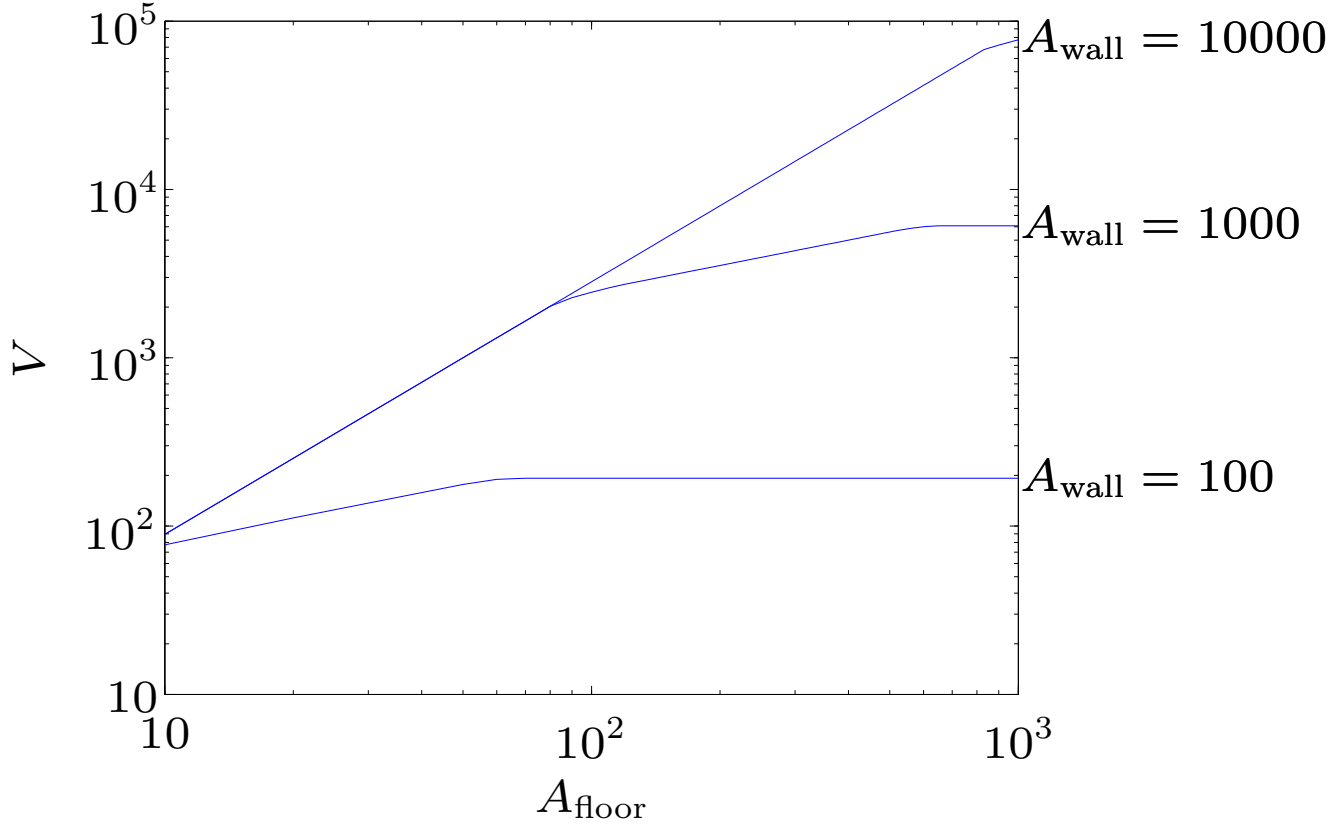
(no equality constraints, for simplicity)

- form perturbed version of original GGP, with changed righthand sides:

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq u_i, \quad i = 1, \ldots, m \end{array}$$

- $u_i > 1$ ($u_i < 1$) means ith constraint is relaxed (tightened)
- let $p(u)$ be optimal value of perturbed problem
- plot of p vs. u is (globally) **optimal trade-off surface** (of objective against constraints)

Trade-off curves for maximum volume box example



- maximum volume V vs. A_{flr}, for $A_{\text{wall}} = 100,\ 1000,\ 10000$
- h/w, d/w aspect ratio limits 0.5, 2

GP and GGP attributes

- after log transform of variables/constraints, they become **convex problems**

- can convert GGP to GP, *e.g.*, $f(x) + \max\{g(x), h(x)\} \leq 1$ becomes

$$f(x) + t \leq 1, \quad g(x)/t \leq 1, \quad h(x)/t \leq 1$$

  where t is new (dummy) variable

- **conversion tricks can be automated**
  - parser scans problem description, forms GP
  - efficient GP solver solves GP
  - solution transformed back (dummy variables eliminated)

How GPs (and GGPs) are solved

the practical answer: **none of your business**

more politely: **you don't need to know**

it's **technology:**

- good algorithms are known
- good software implementations are available

How GPs are solved

- work with log of variables: $y_i = \log x_i$
- take log of monomials/posynomials to get

$$\begin{array}{ll} \text{minimize} & \log f_0(e^y) \\ \text{subject to} & \log f_i(e^y) \leq 0, \quad i = 1, \ldots, m \\ & \log g_i(e^y) = 0, \quad i = 1, \ldots, p \end{array}$$

- $\log f_i(e^y)$ are (smooth) **convex** functions
- $\log g_i(e^y)$ are affine functions, $i.e.$, linear plus a constant
- solve (nonlinear) **convex optimization problem** above using interior-point method

Current state of the art

- basic interior-point method that exploits sparsity, generic GP structure
- approaching efficiency of linear programming solver
  - sparse 1000 vbles, 10000 monomial terms: few seconds
  - sparse 10000 vbles, 100000 monomial terms: minute
  - sparse 10^6 vbles, 10^7 monomial terms: hour

  (these are order-of-magnitude estimates, on simple PC)

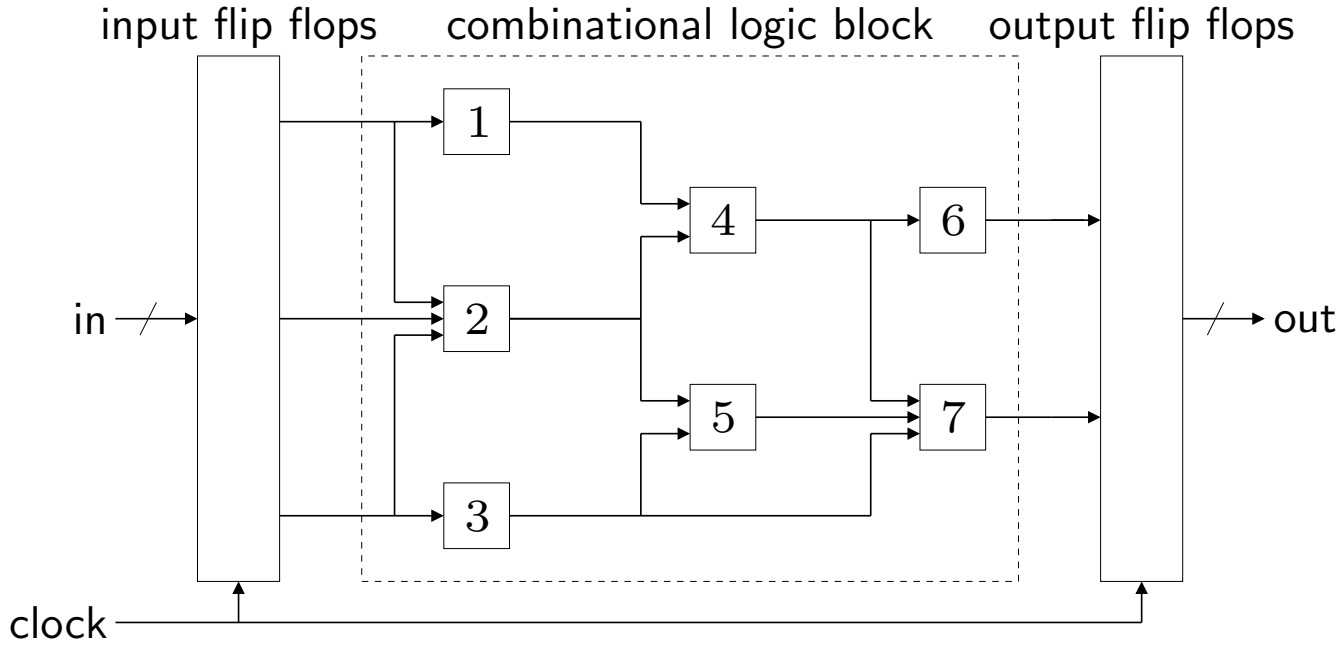
History

- GP (and term 'posynomial') introduced in 1967 by Duffin, Peterson, Zener

- engineering applications from the very beginning
  - early applications in chemical, mechanical, power engineering
  - digital circuit transistor and wire sizing with Elmore delay since 1984 (Fishburn & Dunlap's TILOS, Sapatnekar, Kang, . . .)
  - analog circuit design since 1997 (Hershenson, Boyd, Lee)
  - other applications in statistics, wireless power control, . . .

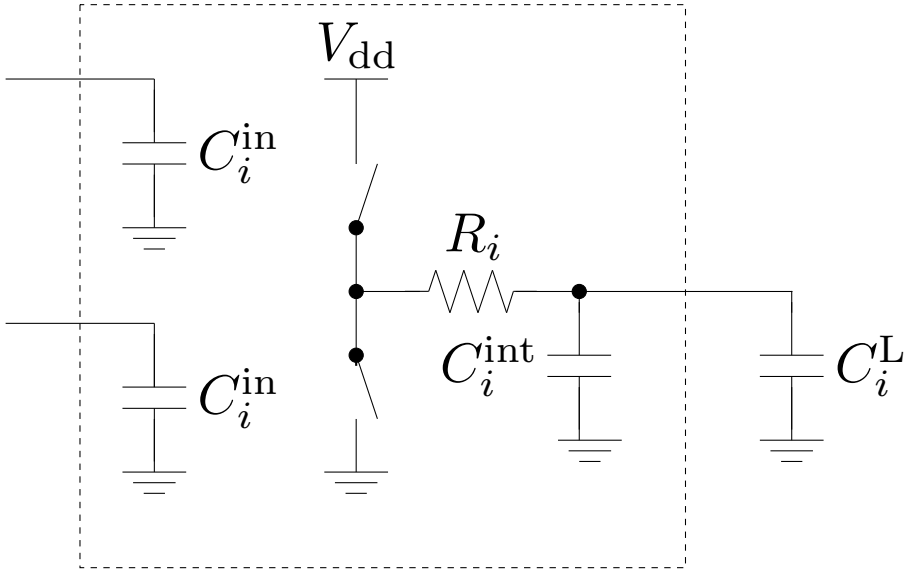
- extremely efficient solution methods since 1994 or so (Nesterov & Nemirovsky)

Gate scaling



- combinational logic; circuit topology & gate types given
- gate sizes (scale factors $x_i \geq 1$) to be determined
- scale factors affect total circuit area, power and delay

RC gate delay model



- input & intrinsic capacitances, driving resistance, load capacitance

$$C_i^{\mathrm{in}} = \bar{C}_i^{\mathrm{in}} x_i, \qquad C_i^{\mathrm{int}} = \bar{C}_i^{\mathrm{int}} x_i, \qquad R_i = \bar{R}_i / x_i, \qquad C_i^{\mathrm{L}} = \sum_{j \in \mathrm{FO}(i)} C_j^{\mathrm{in}}$$

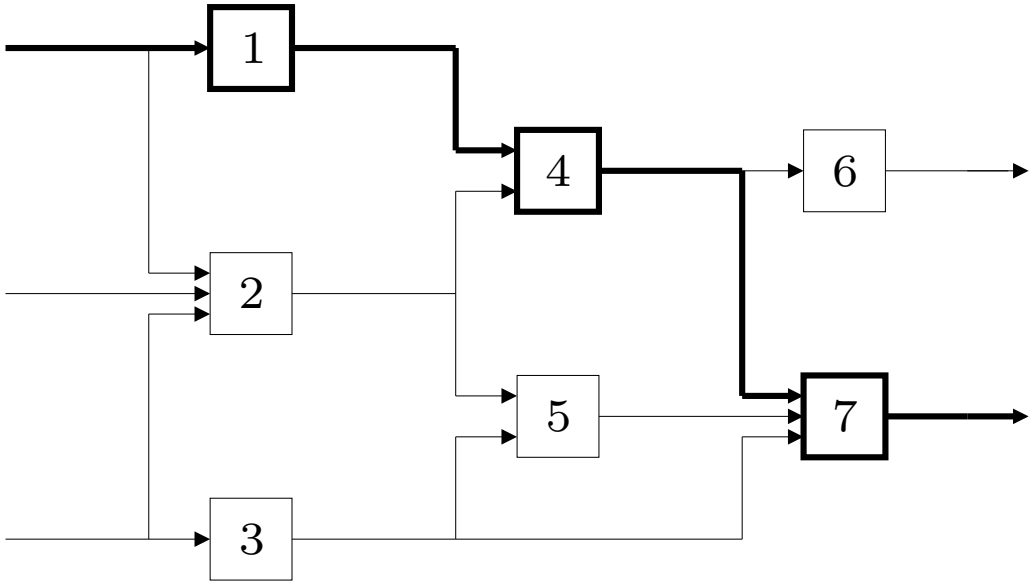
RC gate model

- RC gate delay:

$$D_i = 0.69 R_i (C_i^{\mathrm{L}} + C_i^{\mathrm{int}}) = 0.69 \left(\bar{R}_i \bar{C}_i^{\mathrm{in}} + (\bar{R}_i / x_i) \sum_{j \in \mathrm{FO}(i)} \bar{C}_j^{\mathrm{in}} x_j \right)$$

- D_i are **posynomials** (of scale factors)

Path and circuit delay



- delay of a path: sum of delays of gates on path
  . . . **posynomial**

- circuit delay: maximum delay over all paths
  . . . **generalized posynomial**

Area & power

- total circuit area: $A = x_1\bar{A}_1 + \cdots + x_n\bar{A}_n$
- total power is $P = P_{\mathrm{dyn}} + P_{\mathrm{stat}}$
  - dynamic power $P_{\mathrm{dyn}} = \sum_{i=1}^{n} f_i(C_i^{\mathrm{L}} + C_i^{\mathrm{int}})V_{\mathrm{dd}}^2$

    f_i is gate switching frequency
  - static power $P_{\mathrm{stat}} = \sum_{i=1}^{n} x_i \bar{I}_i^{\mathrm{leak}} V_{\mathrm{dd}}$

    $\bar{I}_i^{\mathrm{leak}}$ is leakage current (average over input states) of unit scaled gate
- A and P are linear functions of x, with positive coefficients, hence posynomials

Basic gate scaling problem

$$\begin{array}{ll} \text{minimize} & D \\ \text{subject to} & P \leq P^{\max}, \quad A \leq A^{\max} \\ & 1 \leq x_i, \quad i = 1, \ldots, n \end{array}$$

. . . a **GGP**

extensions/variations:

- minimize area, power, or some combination
- maximize clock frequency subject to area, power limits
- add other constraints
- optimal trade-off of area, power, delay

Example: 32-bit Ladner-Fisher adder

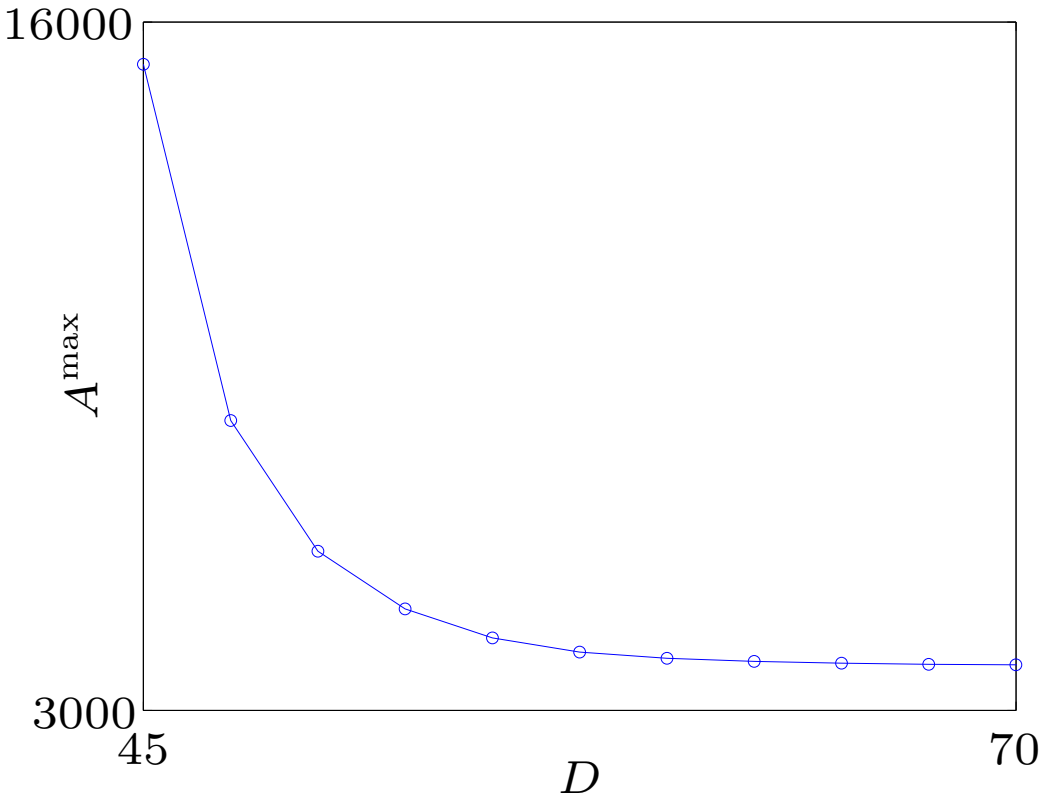
- 451 gates (scale factors), 5 gate types, 64 inputs, 32 outputs
- logical effort gate delay model parameters:

gate type	$\bar{C}^{\text{in}}$	$\bar{C}^{\text{int}}$	$\bar{R}$	$\bar{A}$	$\bar{I}^{\text{leak}}$
INV	3	3	0.48	3	0.006
NAND2	4	6	0.48	8	0.007
NOR2	5	6	0.48	10	0.009
AOI21	6	7	0.48	17	0.003
OAI21	6	7	0.48	16	0.003

- time unit is τ, delay of min-size inverter $(0.69 \cdot 0.48 \cdot 3 = 1)$
- area (total width) unit is width of NMOS in min-size inverter

Example: 32-bit Ladner-Fisher adder

- typical optimization time: few seconds on PC



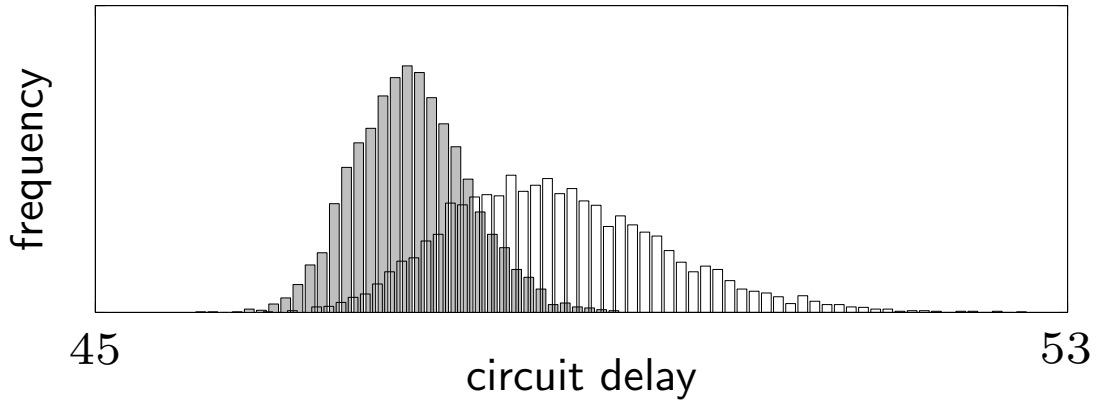
Extensions

- can use better (GP-compatible) models of delay, area, power, . . .
- can distinguish rising/falling transitions, input pins, . . .
- can add effect of signal slope

. . . problem remains a GGP

Statistical parameter variation

- circuit peformance depends on random device and process parameters
- hence, performance measures like P, D are random variables $\mathbf{P}$, $\mathbf{D}$
- delay **D** is max of many random variables; often skewed to right
- **distributions** of $\mathbf{P}$, $\mathbf{D}$ depend on gate scalings x_i



- related to (parametric) yield, DFM, DFY . . .

Statistical design

- measure random performance measures by 95% quantile (say)

$$\begin{array}{ll} \text{minimize} & \mathbf{Q}^{\cdot 95}(\mathbf{D}) \\ \text{subject to} & \mathbf{Q}^{\cdot 95}(\mathbf{P}) \leq P^{\max}, \qquad A \leq A^{\max} \\ & 1 \leq x_i, \quad i = 1, \ldots, n \end{array}$$

- **extremely difficult** stochastic optimization problem; almost no analytic/exact results

- but, (GP-compatible) heuristic method works well

Statistical model

- for simplicity consider $V_{\rm th}$ variation only
- Pelgrom's model: $\sigma_{V_{\rm th}} = \bar{\sigma}_{V_{\rm th}} x^{-1/2}$
- alpha-power law model: $D \propto V_{\rm dd}/(V_{\rm dd} - V_{\rm th})^{\alpha}$, with $\alpha \approx 1.3$
- for small variation in $V_{\rm th}$,

$$\sigma_D = \left| \frac{\partial D}{\partial V_{\rm th}} \right| \sigma_{V_{\rm th}} = \alpha (V_{\rm dd} - V_{\rm th})^{-1} \bar{\sigma}_{V_{\rm th}} x^{-0.5} D$$

- σ_D is posynomial
- get similar (posynomial) models for σ_D with more complex gate delay statistical models

Heuristic for statistical design

- assume generalized posynomial models for gate delay mean $D_i(x)$ and variance $\sigma_i(x)^2$

- optimize using **surrogate gate delays**

$$\tilde{D}_i(x) = D_i(x) + \kappa_i \sigma_i(x)$$

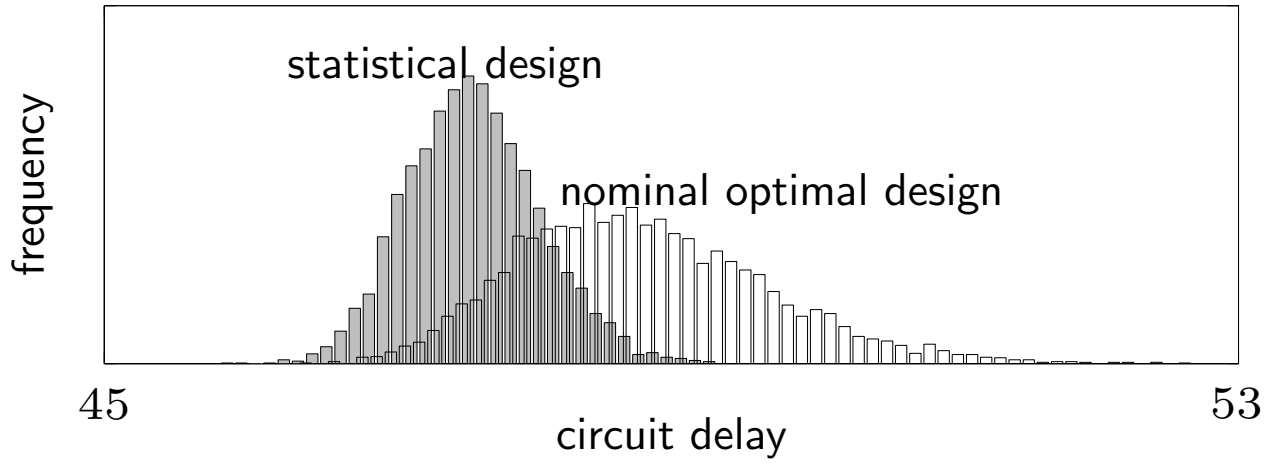
  $\kappa_i \sigma_i(x)$ are **margins** on gate delays (κ_i is typically 2 or 3)

- verify statistical performance via Monte Carlo analysis
  (can update κ_i's and repeat)

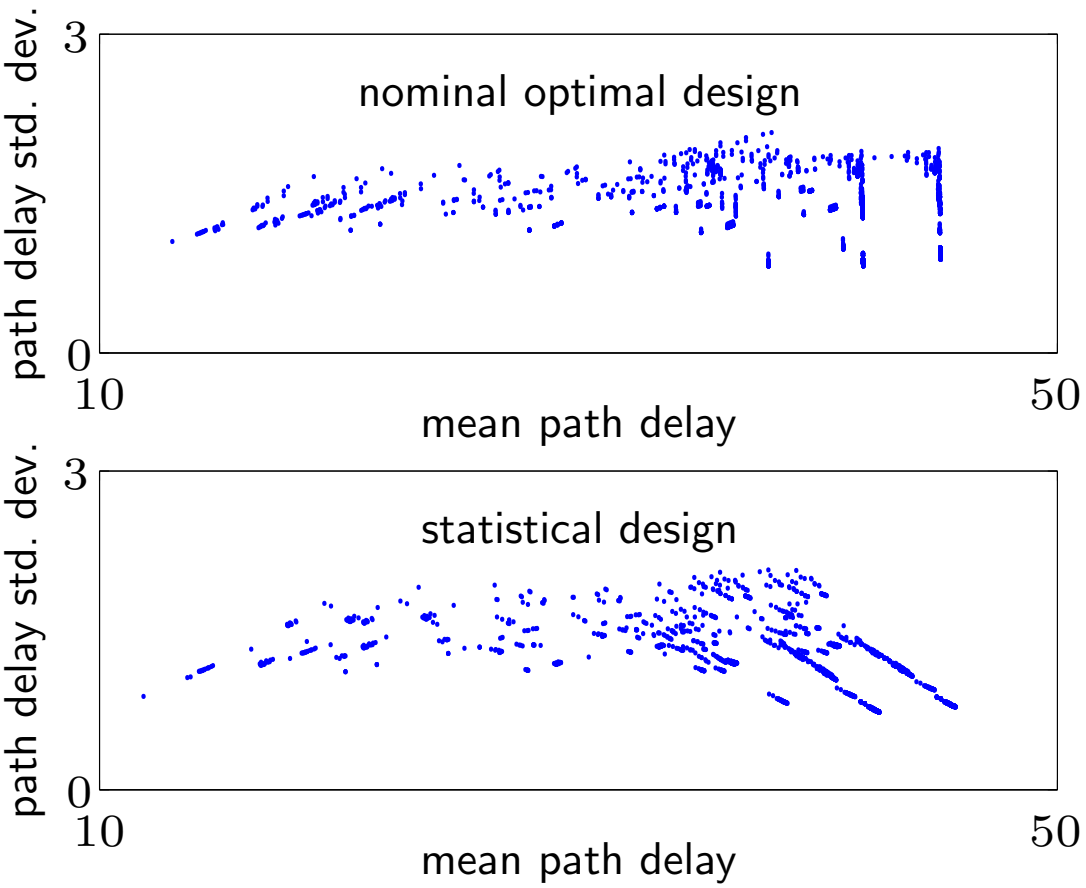
Heuristic for statistical design

heuristic statistical design

- often far superior to design obtained ignoring statistical variation
- not very sensitive to details of process variation statistics (distribution shape, correlations, . . .)
- below: 32-bit Ladner-Fisher adder, Pelgrom variance model



Path delay mean/std. dev. scatter plots



Joint size and supply/threshold voltage optimization

- **goal:** jointly optimize gate size, supply and threshold voltages via GGP
- **need to:** model delay, power as generalized posynomial functions of gate size, supply and threshold voltages

Generalized posynomial delay model

- alpha-power law model predicts variation in gate delay with V_{dd}, V_{th}:

$$D_i = \frac{V_{\mathrm{dd},i}}{(V_{\mathrm{dd},i} - V_{\mathrm{th},i})^{\alpha}} \tilde{D}_i(x)$$

  $\tilde{D}_i$ is generalized posynomial gate delay model, function of scalings x

- generalized posynomial approximation

$$\widehat{D}_i = V_{\mathrm{dd},i}^{1-\alpha}(1 + V_{\mathrm{th},i}/V_{\mathrm{dd},i} + \cdots + (V_{\mathrm{th},i}/V_{\mathrm{dd},i})^5)^{\alpha} \tilde{D}_i(x)$$

  error under 1% for $V_{\mathrm{dd},i} \geq 2V_{\mathrm{th},i}$, $1.3 \leq \alpha \leq 2$

Generalized posynomial power model

- gate dynamic power: $P_{\mathrm{dyn}} = \sum_{i=1}^{n} f_i (C_i^{\mathrm{L}} + C_i^{\mathrm{int}}) V_{\mathrm{dd},i}^2$

- simple static power model:

$$P_{\mathrm{stat}} = \sum_{i=1}^{n} x_i \bar{I}_i^{\mathrm{leak}} V_{\mathrm{dd},i}, \qquad \bar{I}_i^{\mathrm{leak}} \propto e^{-(V_{\mathrm{th},i} - \gamma V_{\mathrm{dd},i})/V_0}$$

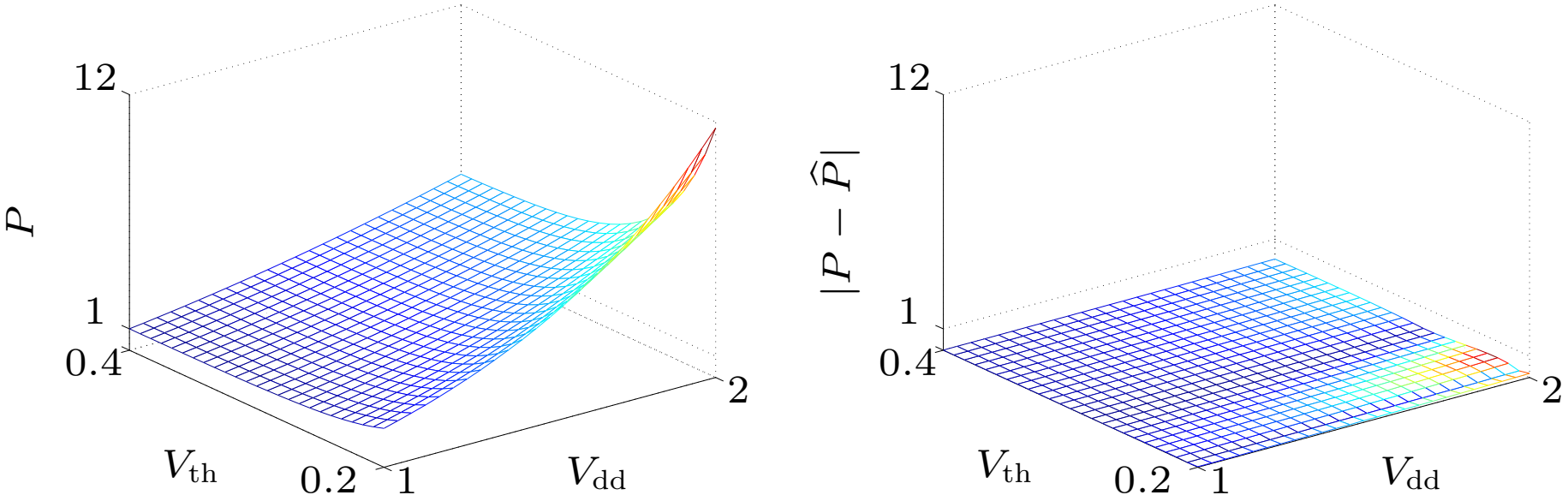
  γ, V_0 are (process) constants

- P_{stat} (by itself) **cannot** be approximated well by a generalized posynomial over large range of V_{dd}, V_{th}

- but, total power $P = P_{\mathrm{dyn}} + P_{\mathrm{stat}}$ **can** be approximated well by a generalized posynomial

Generalized posynomial power model example

total power $P = V_{\mathrm{dd}}^2 + 30V_{\mathrm{dd}}e^{-(V_{\mathrm{th}}-0.06V_{\mathrm{dd}})/0.039}$ (up to scaling)



- generalized posynomial approximation
  $\widehat{P} = V_{\mathrm{dd}}^2 + 0.06V_{\mathrm{dd}}(1 + 0.0031V_{\mathrm{dd}})^{500}(V_{\mathrm{th}}/0.039)^{-6.16}$

- error under 3% (well under accuracy of model!)

Joint optimization of gate sizes, $V_{\rm dd}$, & $V_{\rm th}$

basic problem, with variables: x_i, $V_{{\rm th},i}$, $V_{{\rm dd},i}$

$$
\begin{array}{ll}
\text{minimize} & D \\
\text{subject to} & P \leq P^{\max}, \qquad A \leq A^{\max} \\
& V_{\rm th}^{\min} \leq V_{{\rm th},i} \leq V_{\rm th}^{\max}, \quad i = 1, \ldots, n \\
& V_{\rm dd}^{\min} \leq V_{{\rm dd},i} \leq V_{\rm dd}^{\max}, \quad i = 1, \ldots, n \\
& \text{other constraints} \ldots
\end{array}
$$

. . . **a GGP**

- discrete constraints such as $V_{{\rm th},i} \in \{0.2, 0.3, 0.4\}$, $V_{{\rm dd},i} \in \{0.6, 1.0\}$ yield **mixed-integer** GGP
- ignoring discrete constraints gives lower bound (limit on performance)
- simple rounding, or branch-and-bound, gives valid design

Extensions/variations

- clustering, with single V_{dd}, V_{th} per cluster:

$$V_{\mathrm{dd},i} = V_{\mathrm{dd},j}, \quad V_{\mathrm{th},i} = V_{\mathrm{th},j} \quad \text{for } i, j \text{ in same cluster}$$

  . . . monomial (equality) constraints

- clustered voltage scaling (CVS): low V_{dd} cells cannot drive high V_{dd} cells

$$V_{\mathrm{dd},j} \leq V_{\mathrm{dd},i} \quad \text{for } j \in \mathrm{FO}(i)$$

  . . . monomial (inequality) constraints

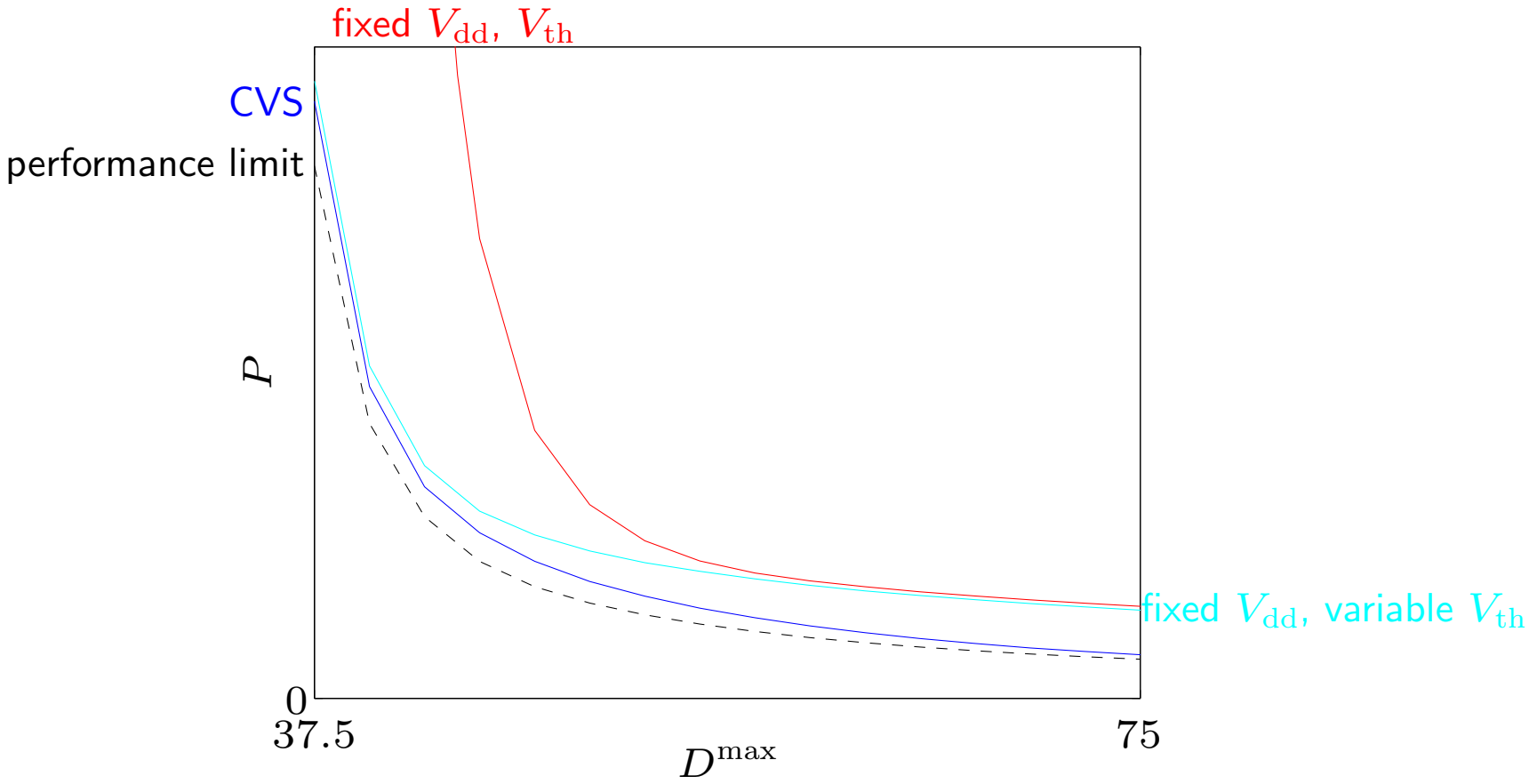
- multimode design: choose single set of gate scalings, different $V_{\mathrm{dd}}^{(k)}$, $V_{\mathrm{th}}^{(k)}$ for each scenario $k = 1, \ldots, K$

  related to **dynamic voltage scaling**, **adaptive bulk biasing**, . . .

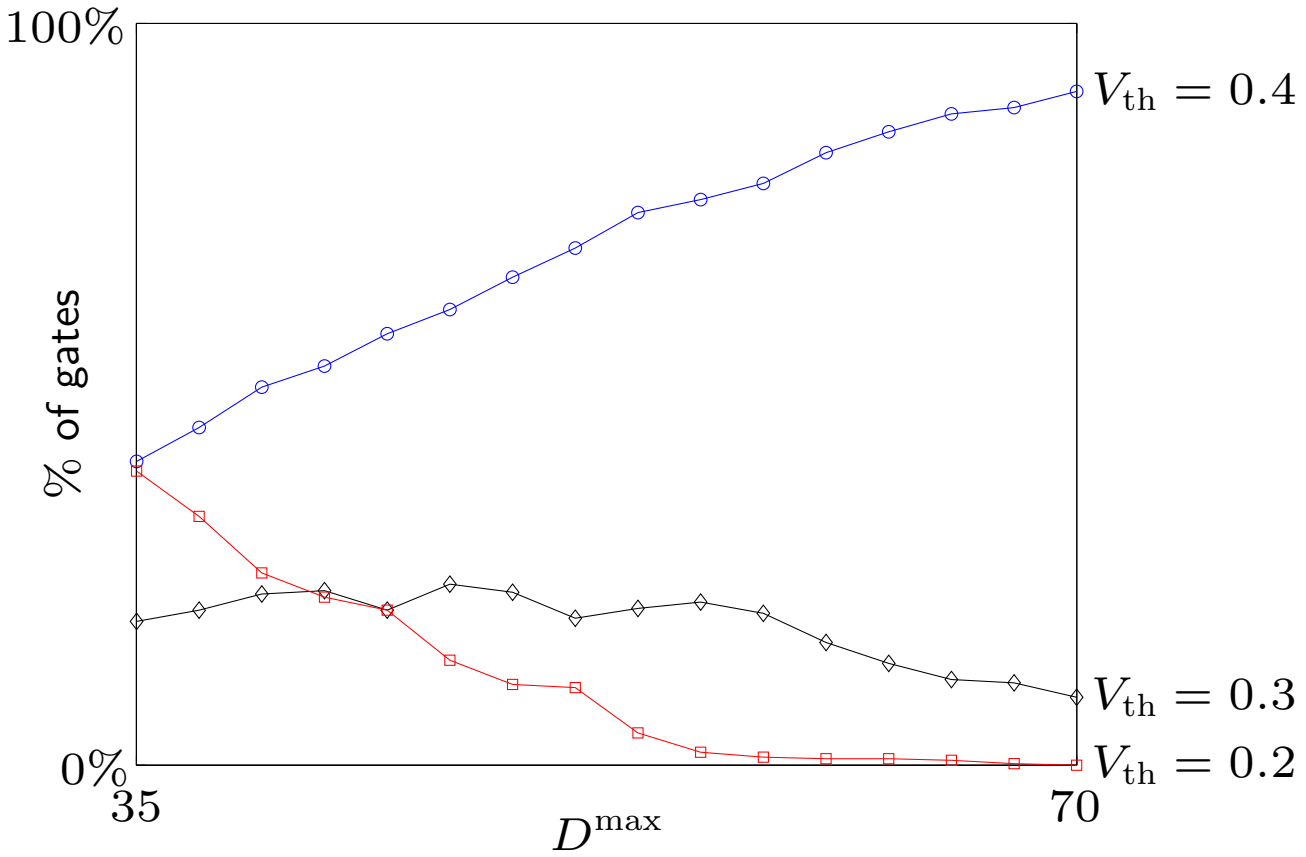
Joint optimization examples

- Ladner-Fisher adder
- variables: gate scalings x_i, supply voltages $V_{\mathrm{dd},i}$, threshold voltages $V_{\mathrm{th},i}$
- four delay-power trade-off curves:
  - fixed $V_{\mathrm{dd},i} = 1.0$, fixed $V_{\mathrm{th},i} = 0.3$
  - fixed $V_{\mathrm{dd},i} = 1.0$, variable $V_{\mathrm{th},i} \in \{0.2, 0.3, 0.4\}$
  - CVS with $V_{\mathrm{dd},i} \in \{0.6, 1.0\}$, $V_{\mathrm{th},i} \in \{0.2, 0.3, 0.4\}$
  - continuous V_{dd}, V_{th}, $0.6 \leq V_{\mathrm{dd},i} \leq 1.0$, $0.2 \leq V_{\mathrm{th},i} \leq 0.4$
    (not practical, but gives performance limit)

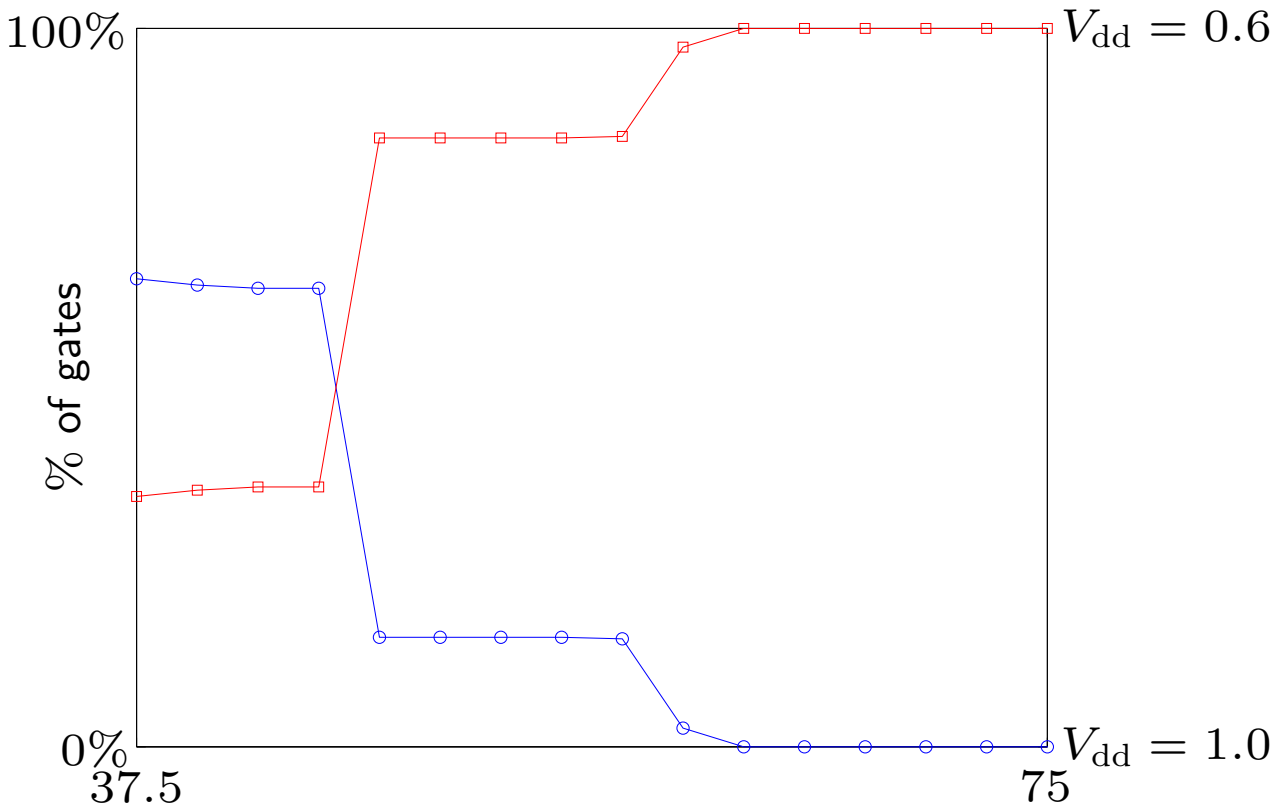
Trade-off curve analysis



Design with multiple threshold voltages



Clustered voltage scaling



Conclusions

(generalized) geometric programming

- comes up in a variety of circuit sizing contexts
- can be used to formulate a variety of problems
- admits fast, reliable solution of large-scale problems
- is good at concurrently balancing lots of coupled constraints and objectives
- is useful even when problem has discrete constraints

Approach

- most problems don't come naturally in GP form; be prepared to reformulate and/or approximate
- GP modeling is not a "try my software" method; it requires thinking
- our approach:
  - start with simple analytical models (RC, square-law, Pelgrom, . . .) to verify GP might apply
  - then fit GP-compatible models to simulation or measured data
  - for highest accuracy, revert to local method for final polishing

References

- this talk taken from *DATE 2005 tutorial*
- *A tutorial on geometric programming*
- *Digital circuit sizing via geometric programming*
- *Convex optimization*, Cambridge Univ. Press 2004

(these include hundreds of references)

available at `www.stanford.edu/~boyd/research.html`

Software

- MOSEK: `www.mosek.com`
- COPL-GP: (Yinyu Ye, in process of being re-worked): `www.stanford.edu/~yyye/Col.html`
- GPGLP: `ftp://ftp.pitt.edu/dept/ie/GP/`
- YALMIP: `control.ee.ethz.ch/~joloef/yalmip.msql`
- a simple matlab GP solver `gp.m` at Boyd's EE364 site (support for GGPs soon)