# Circuit Design via Geometric Programming

**Sunghee Yun**

CAE Team, Samsung Semiconductor

CoSoC Seminar II, SNU, 9/23/2005

# Outline

- Basic approach
- Geometric programming & generalized geometric programming
- Digital circuit design applications
- Statistical digital circuit design applications
- Monomial and posynomial fitting (?)
- Statistical power and ground network design (?)
- Conclusions

# Basic Approach

# Basic approach

1. formulate circuit design problem as **geometric program** (GP), an optimization problem with special form

2. solve GP using specialized method exploiting problem structure

- this talk focuses on step 1 (a.k.a. **GP modeling**)
- step 2 is **technology**
  - you don't need to know

# Why?

- we can solve even large GPs very effectively, using recently developed methods

- so once we have a GP formulation, we can solve circuit design problem effectively

we will see that

- GP is especially good at handling a large number of concurrent constraints

- GP formulation is useful even when it is approximate

# Trade-offs in optimization

- general trade-off between **generality** and **effectiveness**
- generality
  - number of problems that can be handled
  - accuracy of formulation
  - ease of formulation
- effectiveness
  - speed of solution, scale of problems that can be handled
  - global vs. local solutions
  - reliability, baby-sitting, starting point
- example: least-squares vs. simulated annealing

# Where GP fits in

| **Least-squares** | **Convex optimization** | **Simulated annealing** |
| --- | --- | --- |
| large problems solved | optimality condition | any nonlinear problem |
| no initial point theory | duality theory | accuray of formulation |
| solves restricted forms | **somewhere in between** | eash of formulation |
| | good trade-off | |
| | formulations takes effort | |
| | high payoff, however | |

need to break this barrier

# Geometric Programming

# 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

# Geometric program (GP)

a special form of optimization problem:

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

$f_i$ are posynomials and $g_i$ are monomials

- a highly nonlinear constrained optimization problem
- **but, can be solved extremely efficiently**
  - dense $1000$ vbles, $10000$ constraints: one minute on PC
  - sparse $1$M vbles, $10$M constraints: one hour on PC

# Example

$$
\begin{array}{ll}
\text{minimize} & x^{-1}y \\
\text{subject to} & 2x^{-1} \leq 1, \\
& (1/3)x \leq 1, \\
& x^2y^{-1/2} + 3y^{1/2}z^{-1} \leq 1, \\
& xy^{-1}z^{-2} = 1
\end{array}
$$

- this one could be solved by hand, or by sweeping values of $x$, $y$, and $z$
- but a GP with $1000$ variables (which is easily solved if you know how) cannot be solved by hand or sweeping

# Posynomial and monomial algebra

- monomials closed under products, division, positive scaling, powers (hence, inverse), *e.g.*,

$$\left(2x^{-0.2}y^{1.1}\right)\left(0.3xy^{-0.3}z^{2}\right) = 0.6x^{0.8}y^{0.8}z^{2}$$

- posynomials closed under sums, products, positive scaling, division by monomials, positive integer powers

# Simple GP extensions

- maximizing a monomial objective $g$
  - same as minimizing $g^{-1}$, a monomial (hence also posynomial)
- monomial-monomial equality constraint $g_1 = g_2$
  - same as monomial equality constraint $g_1/g_2 = 1$
- posynomial-monomial inequality constraint $f \leq g$
  - same as posynomial inequality constraint $f/g \leq 1$

## Example

- maximize volume of box with width $w$, height $h$, length $d$
- 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

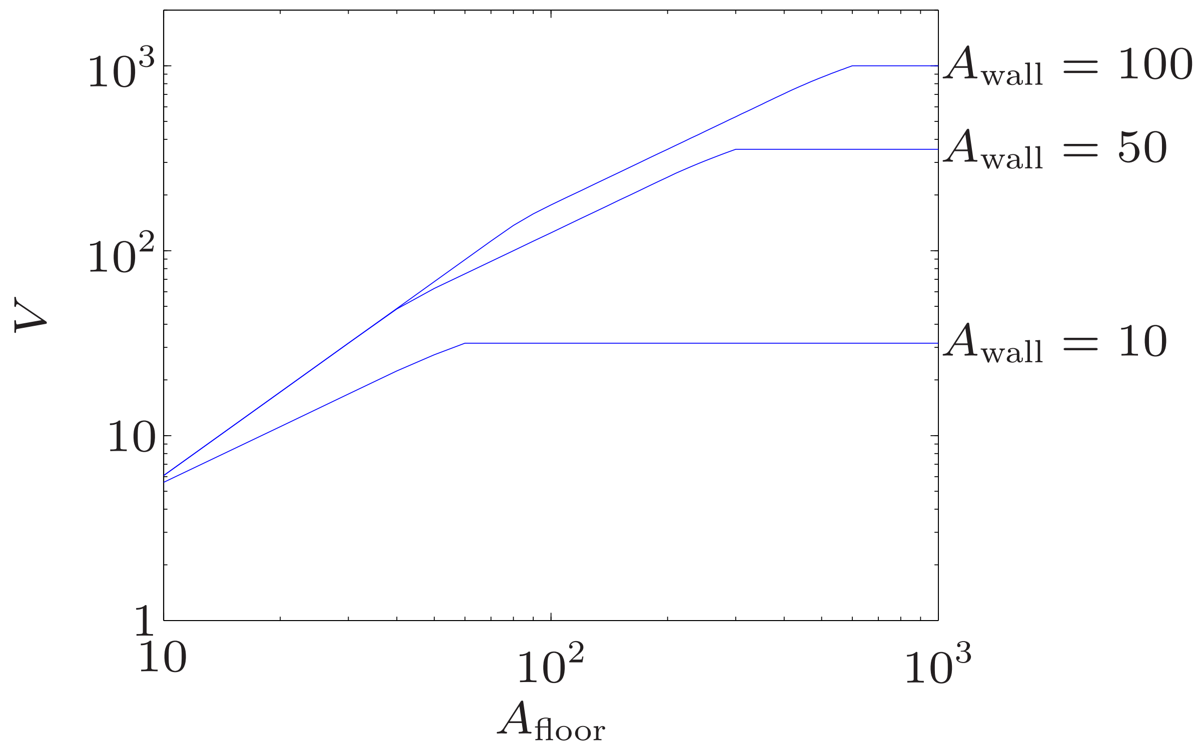
(no equality constraints, for simplicity)

- form perturbed version of original GP, 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 $i$th 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. $A_{\text{flr}}$, for $A_{\text{wall}} = 10,\ 50,\ 100$
- $h/w$, $d/w$ aspect ratio limits $0.5$, $2$

# Sensitivity analysis

- optimal sensitivity of $i$th constraint is

$$S_i = \left. \frac{\partial p/p}{\partial u_i/u_i} \right|_{u=1}$$

- $S_i$ predicts fractional change in optimal objective value if $i$th constraint is (slightly) relaxed or tightened

- very useful in practice; give quantitative measure of how tight a binding constraint is

- when we solve a GP **we get all optimal sensitivities at no extra cost**

# Example

- minimize circuit delay, subject to power, area constraints (details later)

$$\begin{array}{ll} \text{minimize} & D(x) \\ \text{subject to} & P(x) \leq P^{\max}, \quad A(x) \leq A^{\max} \end{array}$$

- both constraints tight at optimal $x^\star$: $P(x^\star) = P^{\max}$, $A(x^\star) = A^{\max}$
- suppose optimal sensitivities are $S^{\text{pwr}} = -2.1$, $S^{\text{area}} = -0.3$
- we predict:
  - for $1\%$ increase in allowed power, optimal delay decreases $2.1\%$
  - for $1\%$ increase in allowed area, optimal delay decreases $0.3\%$

# How GPs are solved

the practical answer: **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

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

- $\log f_i(e^y)$ are **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

# Example

$$f(x, y, z) := xyz$$

substitute $(x, y, z)$ with $(e^v, e^w, e^u)$ respectively and take $\log$ of $f$

$$g(v, w, u) := \log f(e^v, e^w, e^u) = v + w + u$$

- $f$ is **not** a convex function in $(x, y, z)$
- $g$ **is** a convex function in $(v, w, u)$, however

# 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)
  - analog circuit design since 1997 (Hershenson, Boyd, Lee)
- extremely efficient solution methods since 1994 or so (Nesterov & Nemirovsky)

# Generalized Geometric Programming

# Handling positive fractional powers

- suppose $f_1$, $f_2$ are posynomials
- we can handle $f_1 + f_2^3 \leq 1$ directly, since LHS is posynomial
- we can't handle $f_1 + f_2^{3.1} \leq 1$, since $f_2^{3.1}$ isn't posynomial
- **trick:** replace inequality $f_1(x) + f_2(x)^{3.1} \leq 1$ with two (posy) inequalities

$$f_1(x) + t^{3.1} \leq 1, \qquad f_2(x) \leq t$$

  $t$ is new variable (called dummy or slack)

# Handling maximum

- suppose $f_1$, $f_2$, $f_3$ are posynomials
- can't handle $f_1 + \max\{f_2, f_3\} \leq 1$ since $\max\{f_2, f_3\}$ isn't posynomial
- **trick:** replace $f_1 + \max\{f_2, f_3\} \leq 1$ with three (posy) inequalities

$$f_1 + t \leq 1, \qquad f_2 \leq t, \qquad f_3 \leq t$$

  $t$ is new slack variable

- can be applied recursively, together with fractional power trick

# Example

$$
\begin{array}{ll}
\text{minimize} & xyz + 4x^{-1}y^{-3/2} \\
\text{subject to} & \max\{x, y\} + z \leq 1 \\
& y(x^{1/2} + 3z)^{1/2} + z^2 \leq 1
\end{array}
$$

equivalent to GP

$$
\begin{array}{ll}
\text{minimize} & xyz + 4x^{-1}y^{-3/2} \\
\text{subject to} & t_1 + z \leq 1, \quad x \leq t_1, \quad y \leq t_1 \\
& yt_2^{1/2} + z^2 \leq 1, \quad x^{1/2} + 3z \leq t_2
\end{array}
$$

($t_1$ and $t_2$ are new variables)

# Generalized posynomials

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

**examples:**

- $\max\left\{1+x_1, 2x_1+x_2^{0.2}x_3^{-3.9}\right\}$
- $\left(0.1x_1x_3^{-0.5}+x_2^{1.7}x_3^{0.7}\right)^{1.5}$
- $\left(\max\left\{1+x_1, 2x_1+x_2^{0.2}x_3^{-3.9}\right\}\right)^{1.7}+x_2^{1.1}x_3^{3.7}$
- $4x_1^{-0.1}x_2^{2.7}\max\left\{\max\left\{1+x_1, 2x_1+x_2^{0.2}x_3^{-3.9}\right\}+x_2^{1.1}, x_1x_2x_3\right\}$

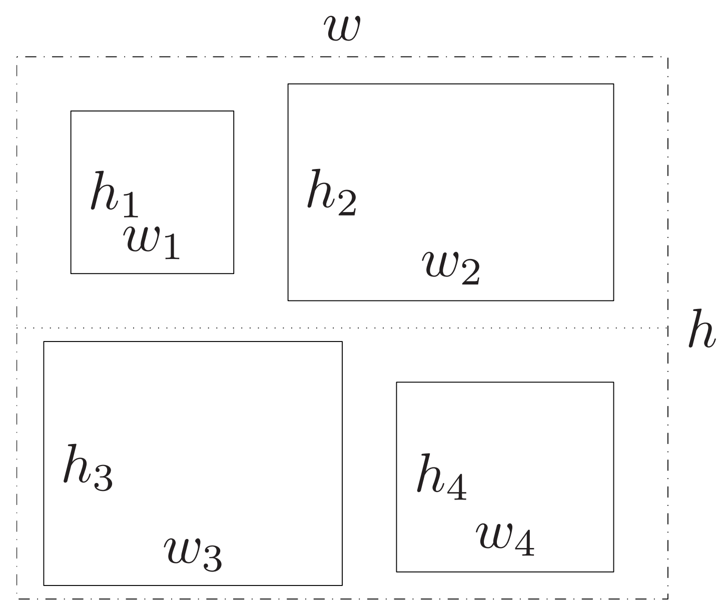
# Generalized geometric program (GGP)

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

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

- using tricks, can convert GGP to GP, then solve efficiently
- **conversion tricks can be automated**
  - parser scans problem description, forms GP
  - GP solver solves GP
  - solution transformed back (dummy variables eliminated)

# Floor planning



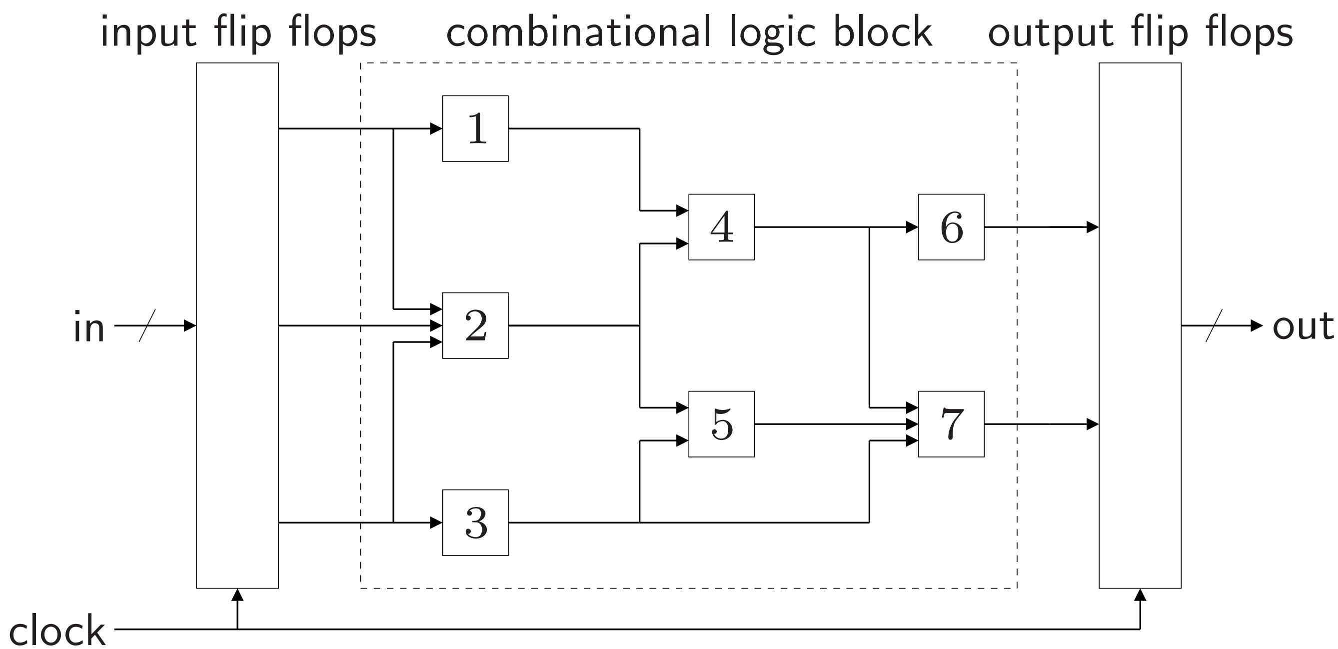

- configure cell widths, heights
- minimize bounding box area
- fixed cell areas
- aspect ratio constraints

$$
\begin{array}{ll}
\text{minimize} & hw \\
\text{subject to} & h_i w_i = A_i, \quad 1/\alpha_{\max} \leq h_i/w_i \leq \alpha_{\max}, \\
& \max\{h_1, h_2\} + \max\{h_3, h_4\} \leq h, \\
& \max\{w_1 + w_2, w_3 + w_4\} \leq w
\end{array}
$$

. . . a GGP

# Digital Circuit Design Applications

# 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

# Area & power

- (approximate) total circuit area: $A = (a_1 x_1 + \cdots + a_n x_n)\bar{A}$
  - $\bar{A}$: area of unit scaled inverter
  - $a_i$: area of unit scaled gate $i$ (in units of $\bar{A}$)

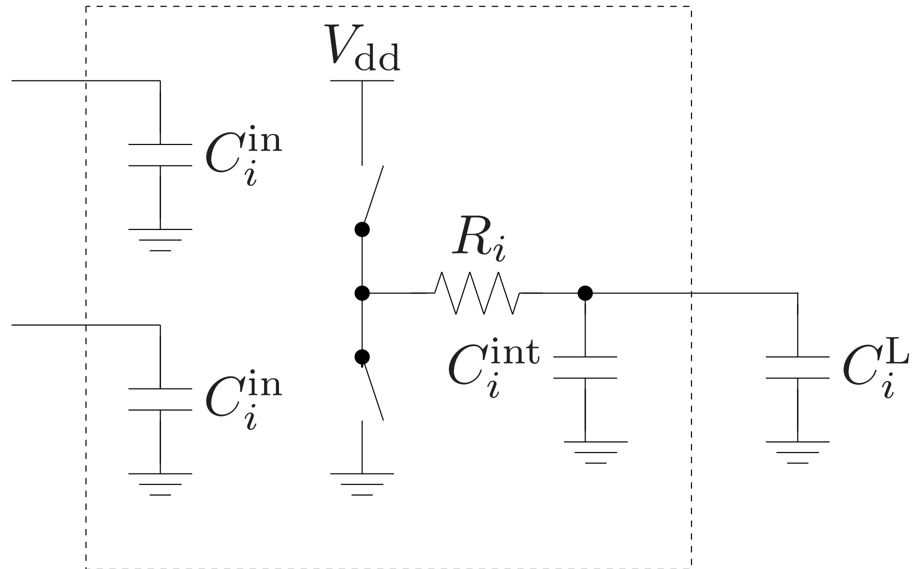
- total power (dynamic + static):

$$P = (b_1 x_1 + \cdots + b_n x_n) f_{\text{clk}} \bar{E} + (c_1 x_1 + \cdots + c_n x_n)$$

  - $f_{\text{clk}}$: clock frequency
  - $\bar{E}$: energy lost per transition by unit scaled inverter driving no load

- $A$ and $P$ are linear functions of $x$, with positive coefficients, hence posynomials

# RC gate delay model




- input & intrinsic capacitances, driving resistance, load capacitance

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

# RC gate delay model

- model

$$\bar{C}_i^{\text{in}} = \alpha_i \eta \bar{C}, \qquad \bar{C}_i^{\text{int}} = \beta_i \bar{C}, \qquad R_i = \gamma_i \bar{R}/x_i$$

  - $\bar{C}$: intrinsic capacitance of unit scaled inverter
  - $\eta$: (input capacitance of unit scaled inverter)$/\bar{C}$
  - $\bar{R}$: driving resistance of unit scaled inverter

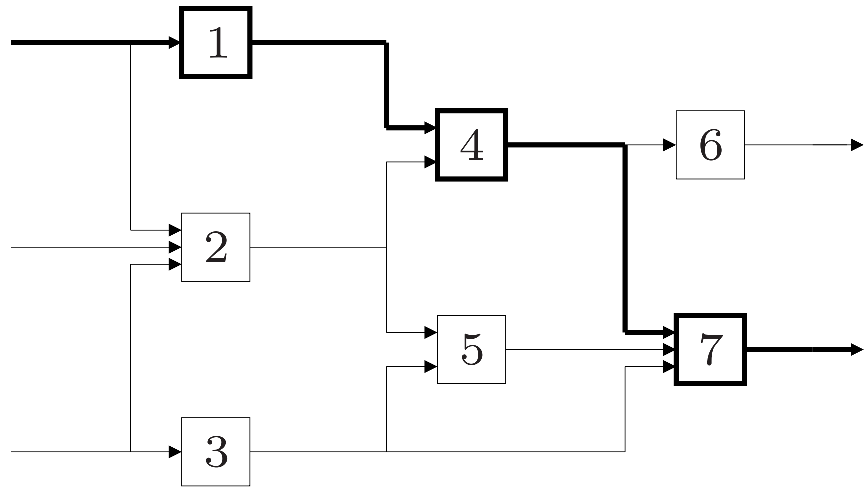
- RC gate delay:

$$D_i = 0.69 R_i (C_i^{\text{L}} + C_i^{\text{int}}) = \left( \gamma_i \beta_i + (\gamma_i/x_i) \sum_{j \in \text{FO}(i)} \eta \alpha_j x_j \right) \bar{D}$$

  $\bar{D} = 0.69 \bar{R} \bar{C}$: delay of unit scaled inverter with no load

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

# Paths and circuit delay




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

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

# 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
- add other constraints
- optimal trade-off of area, power, delay

## Extensions

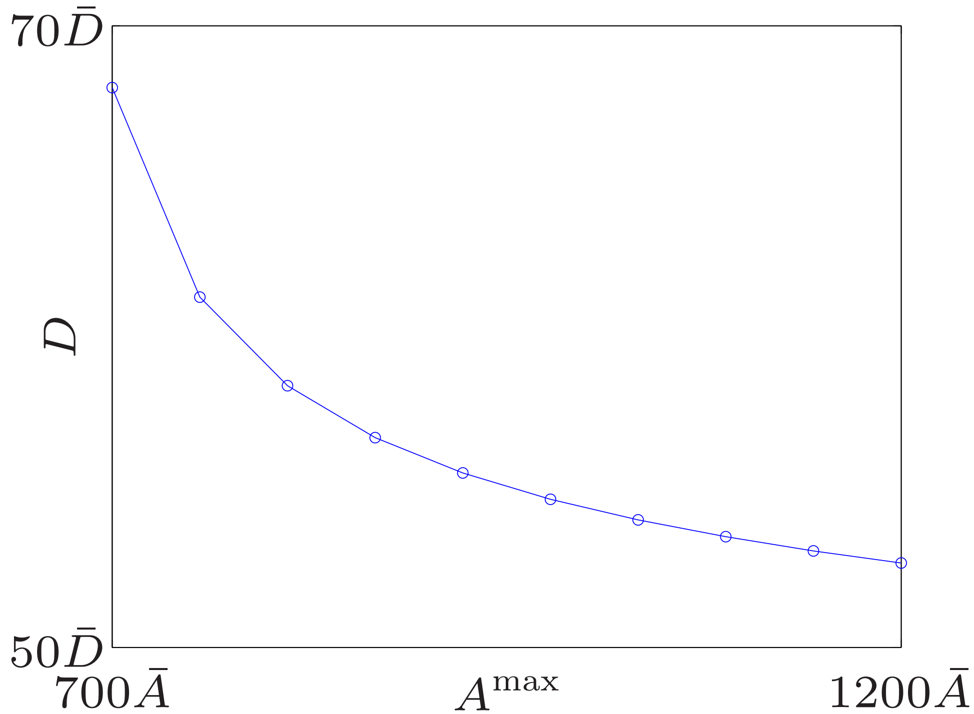
$$
\begin{array}{ll}
\text{minimize} & P \\
\text{subject to} & D \le D^{\max}, \quad A \le A^{\max} \\
& l_i \le x_i \le u_i, \quad i = 1, \ldots, n \\
& \max\{x_{i_k}, x_{j_k}\} \le p_k, \quad k = 1, \ldots, m
\end{array}
$$

$$
\begin{array}{ll}
\text{minimize} & PD \\
\text{subject to} & A \le A^{\max} \\
& l_i \le x_i \le u_i, \quad i = 1, \ldots, n \\
& p_k \le x_{i_k}/x_{j_k} \le q_k, \quad k = 1, \ldots, m
\end{array}
$$

. . . a **GGP**

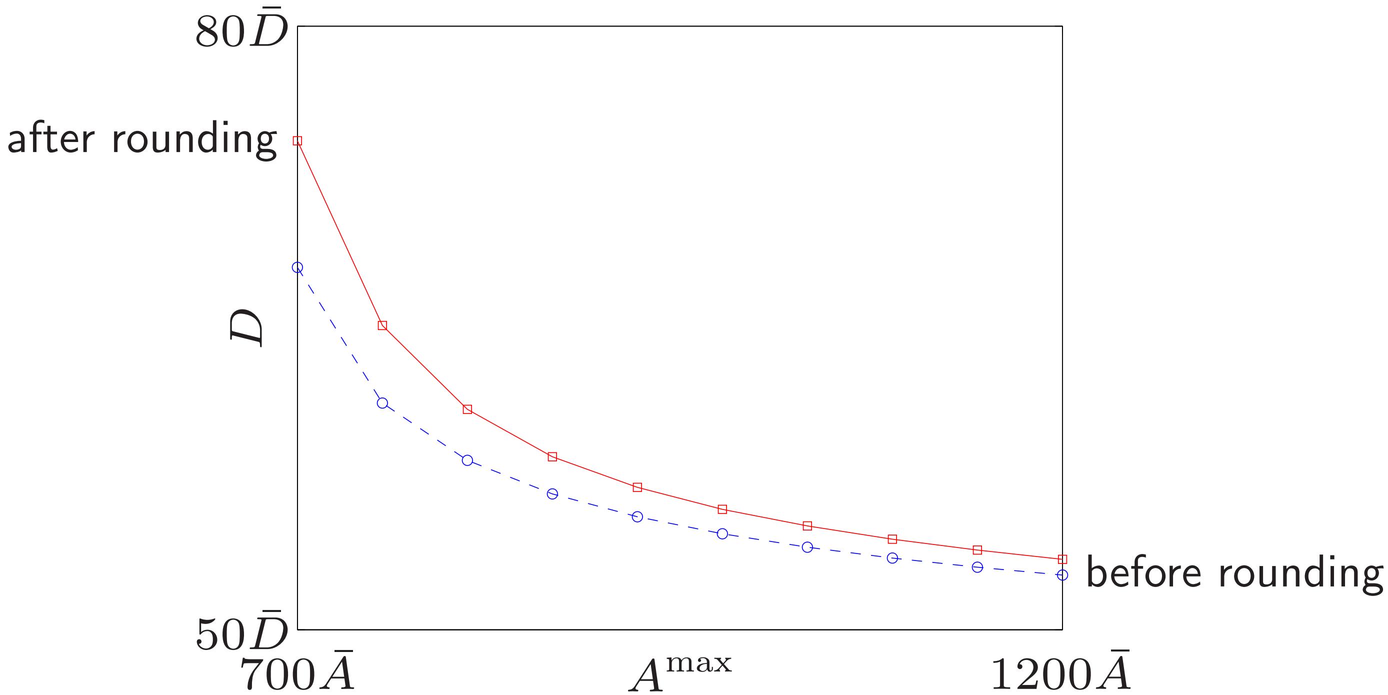
# Example: Ladner-Fisher 32-bit adder

- $451$ gates (scale factors); RC gate delay model
- typical optimization time: few seconds on PC



# Ladner-Fisher 32-bit adder with integer scale factors

- add constraints $x_i \in \{1, 2, 3, \ldots\}$
- simple rounding of optimal continuous scalings



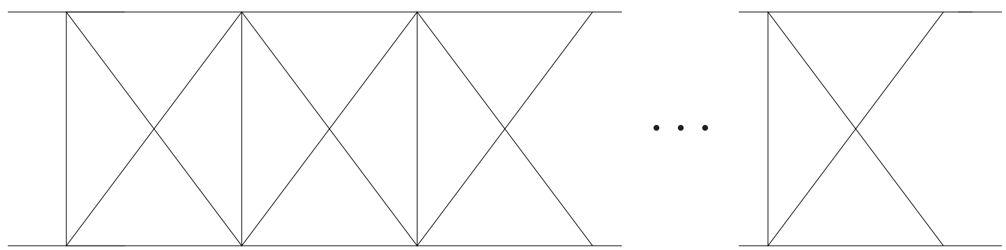
# Sparse GP gate scaling problem

$$
\begin{array}{ll}
\text{minimize} & D \\
\text{subject to} & T_j \leq D \quad \text{for } j \text{ an output gate} \\
 & T_j + D_i \leq T_i \quad \text{for } j \in \mathrm{FI}(i) \\
 & P \leq P^{\max}, \quad A \leq A^{\max} \\
 & 1 \leq x_i, \quad i = 1, \ldots, n
\end{array}
$$

- $T_i$ are upper bounds on signal arrival times
- **extremely sparse GP**; can be solved very efficiently

# Basic formulation vs. Sparse formulation

- assume combinational circuit structure below



- number of contraints grows $\sim \mathcal{O}(2^n)$ in basic formulation
- number of contraints grows $\sim \mathcal{O}(n)$ in sparse formulation
- hence sparse formulation is far superior to basic formulation

## Better (generalized posynomial) models

can greatly improve model, while retaining GP compatibility
(hence efficient global solution)

- area, delay, power can be any generalized posynomials of scale factors, *e.g.*,

$$D_i = a_i + b_i (C_i^{\mathrm{L}})^{1.05} x_i^{-0.9}, \qquad P_i = c_i + d_i (C_i^{\mathrm{L}})^{1.2} + e_i x_i^{1.1}$$

- these can be found by more refined analysis, or fitting generalized posynomials to simulation/characterization data

# Distinguishing gate transitions

- can distinguish rising and falling transitions, with different delay, energy, $C^{\mathrm{in}}$, for each gate input/transition

- (bounds on) signal arrival times can be propagated through recursions, *e.g.*,

$$T_i^{\mathrm{r}} = \max_{j \in \mathrm{FI}(i)} \left\{ T_j^{\mathrm{r}} + D_{ji}^{\mathrm{rr}},\ T_j^{\mathrm{f}} + D_{ji}^{\mathrm{fr}} \right\}, \quad T_i^{\mathrm{f}} = \max_{j \in \mathrm{FI}(i)} \left\{ T_j^{\mathrm{r}} + D_{ji}^{\mathrm{rf}},\ T_j^{\mathrm{f}} + D_{ji}^{\mathrm{ff}} \right\}$$

- **still a GGP**, hence can be efficiently solved

## Further Improvement

- modeling signal slopes
- arrival time propagation with soft maximum
- design with a standard library
- robust design over corners
- multiple-scenario design

# Modeling signal slopes

- associate (worst-case) output signal transition time $\tau$ with each gate
- model delay, energy, input capacitance as (generalized posynomial) functions of scale factor, load capacitance, input transition time
- propagate output transition time using (generalized posynomial) function of scale factor, load capacitance, input transition time
- common model:

$$D_i = a_i C_i^{\mathrm{L}}/x_i + \kappa_i \tau_i^{\mathrm{in}}, \qquad E_i = b_i(C_i^{\mathrm{L}} + c_i x_i) + \lambda_i x_i \tau_i^{\mathrm{in}}, \qquad \tau_i = \nu_i D_i$$

- gate scaling problem **still a GGP**

# Arrival time propagation with soft maximum

- can even generalize max function used to propagate signal arrival times
- replace with soft maximum, *e.g.*, $(T_1^p + \cdots + T_k^p)^{1/p}$ (say, $p \approx 10$)
- can account for increased delay when inputs switch simultaneously
- can choose soft maximum function by fitting simulation data
- gate scaling problem **remains a GGP**

# Design with a standard library

- circuit topology is fixed; choose size for each gate from **discrete library**
- a combinatorial optimization problem, difficult to solve exactly
- GP approach
  - for each gate type in library, fit given library data to find GP-compatible models of delay, power, . . .
  - size with **continuous** fitted models, using GP
  - snap continuous scale factors back to standard library

# Robust design over corners

- have $K$ corners or scenarios, $e.g.$, combinations of
  - process parameters
  - supply voltage
  - temperature
- for each corner have (slightly) different models for delay, power, . . .
- **robust design** finds gate scalings that work well for **all corners**

# Robust design over corners

- basic (worst-case) robust design over corners:

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

- many variations, $e.g.$, minimize average delay over corners,

$$
(1/K)\left(D^{(1)} + \cdots + D^{(K)}\right)
$$

- results in (very large, but sparse) **GGP**

# Multiple-scenario design

- have $K$ scenarios or operating modes, with $K$ models for $P$, $D$, ...
- scenarios are combinations of
  - supply & threshold voltages
  - clock frequency
  - specifications & constraints
- like corner-based robust design, but scenarios are **intentional**
- find one set of gate scalings that work well in all scenarios

# Example

- find single set of gate scalings to support both high performance mode and low power mode
  - in high performance mode: $P^{\mathrm{fast}} \leq \bar{P}^{\mathrm{fast}}$, $D^{\mathrm{fast}} \leq \bar{D}^{\mathrm{fast}}$
  - in low power mode: $P^{\mathrm{slow}} \leq \bar{P}^{\mathrm{slow}}$, $D^{\mathrm{slow}} \leq \bar{D}^{\mathrm{slow}}$

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

. . . a **GGP**

# Statistical Digital Circuit Design Applications

# Statistical parameter variations in circuits

- statistical variations in process
  - random defects: random particles while CMP, CVD, PVD, *etc.*
  - systematic defects: OPC, *etc.*

- statistical variations in environment: supply voltage, temperature, *etc.*

$\Rightarrow$ induces statistical variations in (physical) parameters, *e.g.*, $L_{\mathrm{eff}}$, $W$, $T_{\mathrm{ox}}$, $V_{\mathrm{tho}}$, $\mu_{\mathrm{n}}$, *etc.*

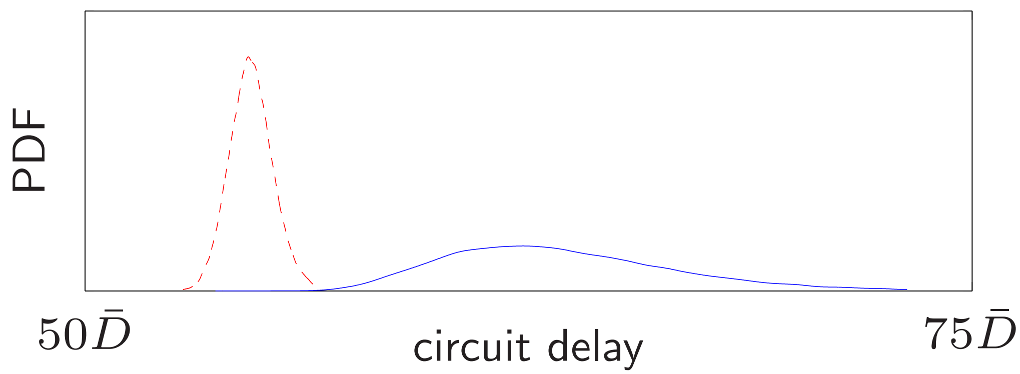
# DFM & yield enhancement

- statistical variation significantly affects performance in DSM regime
- statistical variation is very complex and extremely hard; modeling still open
- merely start exploring statistical design methods; DFM, DFY, DFT, *etc.*
- everyone is having difficulty achieving this goal
  (but everyone is doing it!)

refer to ITRS

# Statistical performance variation

- circuit peformance depends on random device and process parameters
- hence, performance measures like $P$, $D$ are random variables $\mathbf{P}$, $\mathbf{D}$
- delay $\mathbf{D}$ 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)

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

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

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

# Heuristic for statistical design

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

- $e.g.$, $\sigma_i(x) = \eta_i x_i^{-1/2} D_i(x)$ (Pelgrom's model)

- optimize using surrogate gate delays

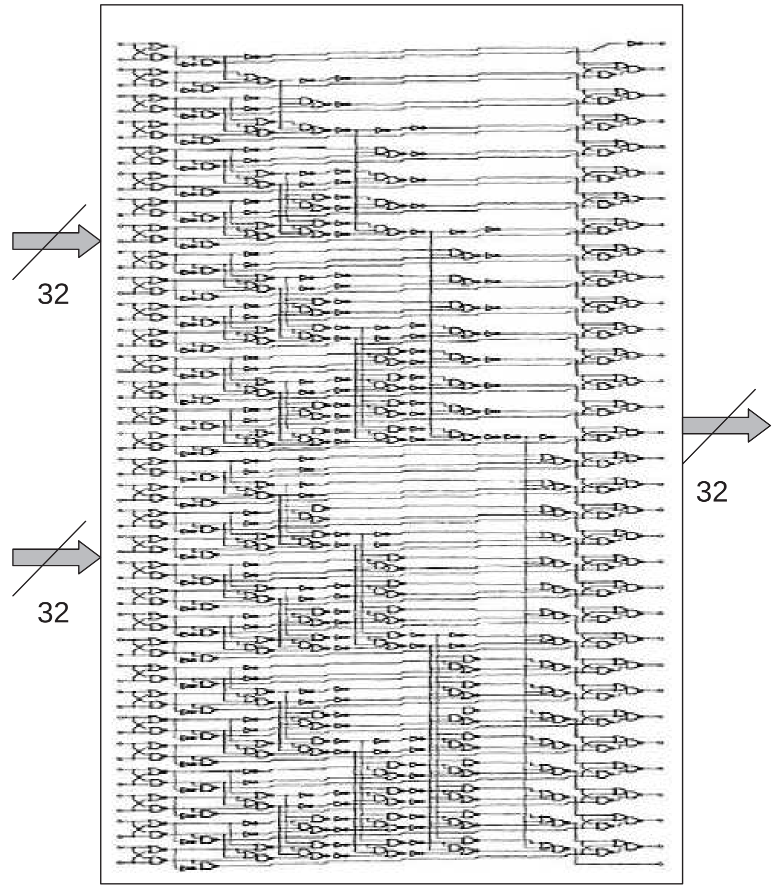
$$\tilde{D}_i(x) = D_i(x) + \kappa_i \sigma_i(x)$$

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

- verify statistical performance via Monte Carlo
  (can update $\kappa_i$'s and repeat)

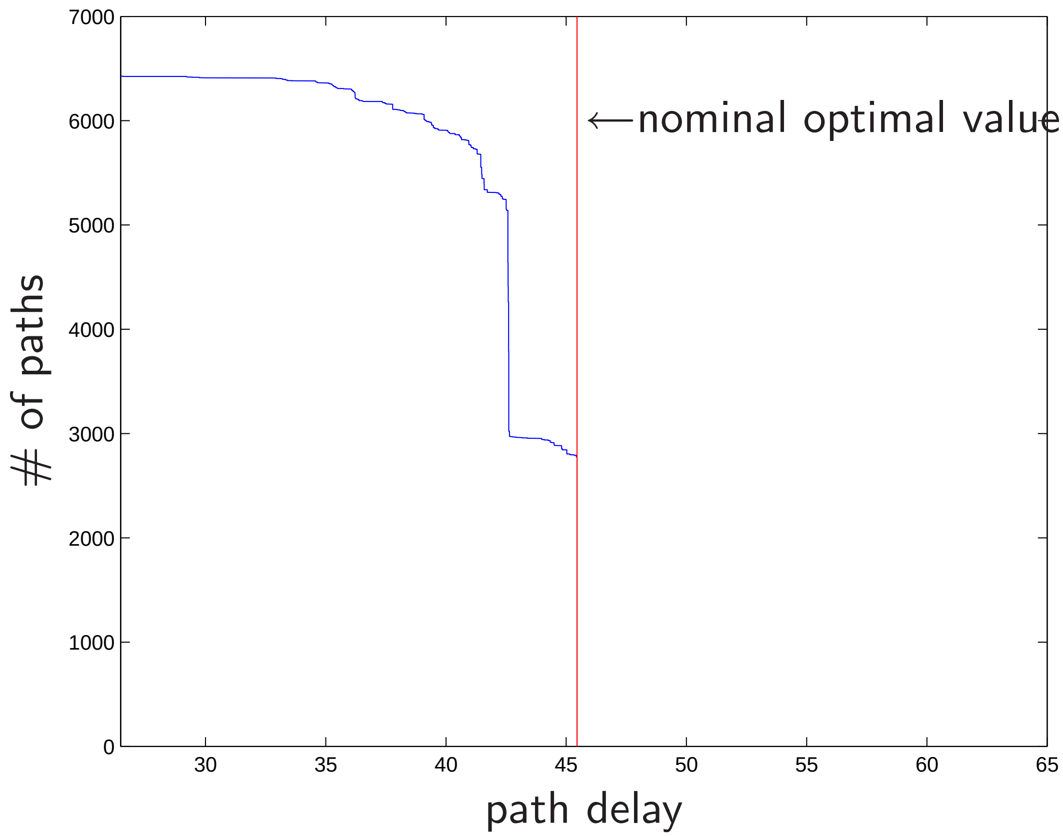
# Ladner-Fisher 32-bit adder example

- minimize maximum delay with constraints
- simplified RC delay model
- Pelgrom variation model ($15\%$ $\sigma/\mu$ for min size devices)
- design variables: device widths for $451$ gates . . .




Schematic of Ladner-Fisher 32-bit adder

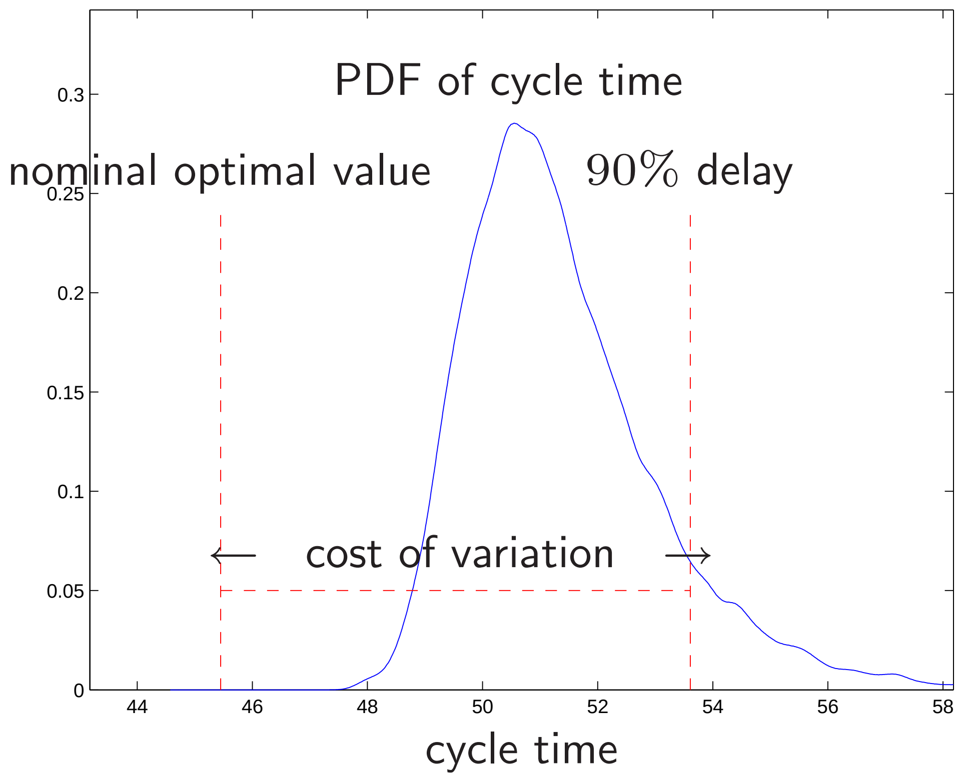
# Nominal (or deterministic) optimization result




. . . around $2800$ of $6400$ total paths are critical

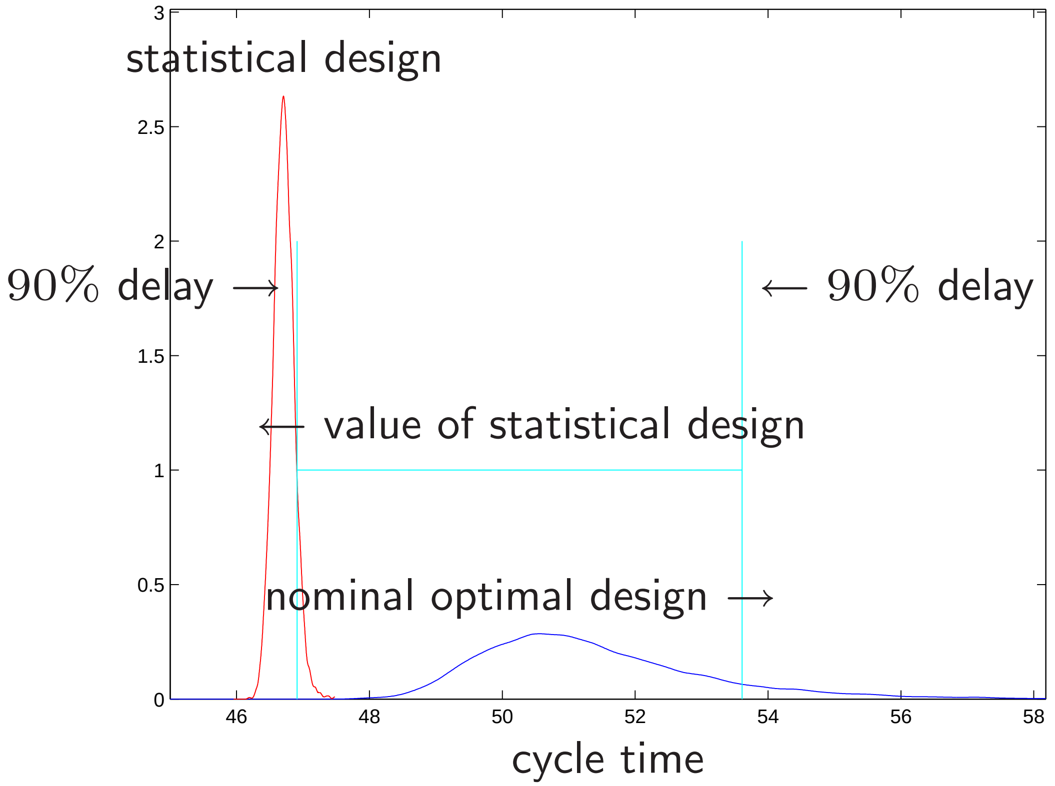
# Cost of statistical variation

Monte Carlo analysis of nominal optimal design



# Statistically robust design via new method

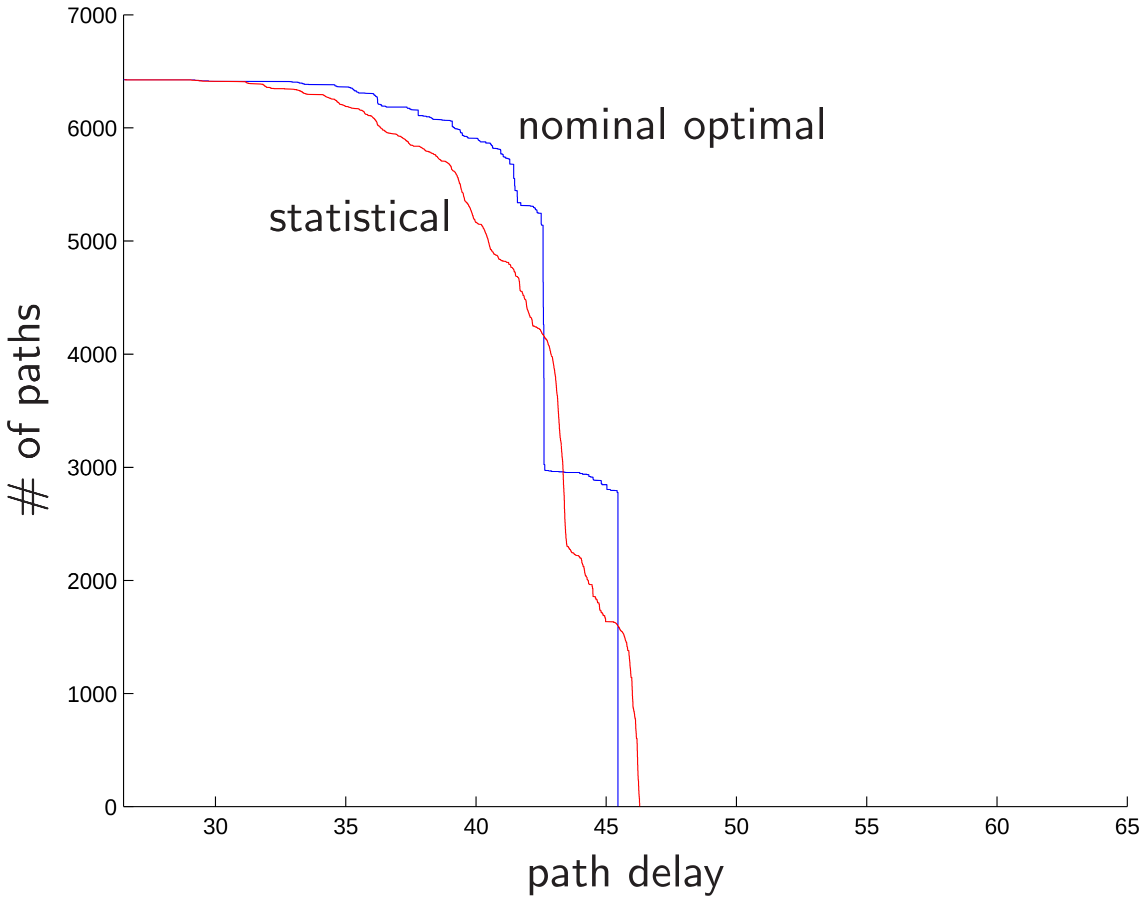
same circuit, uncertainty model, and constraints



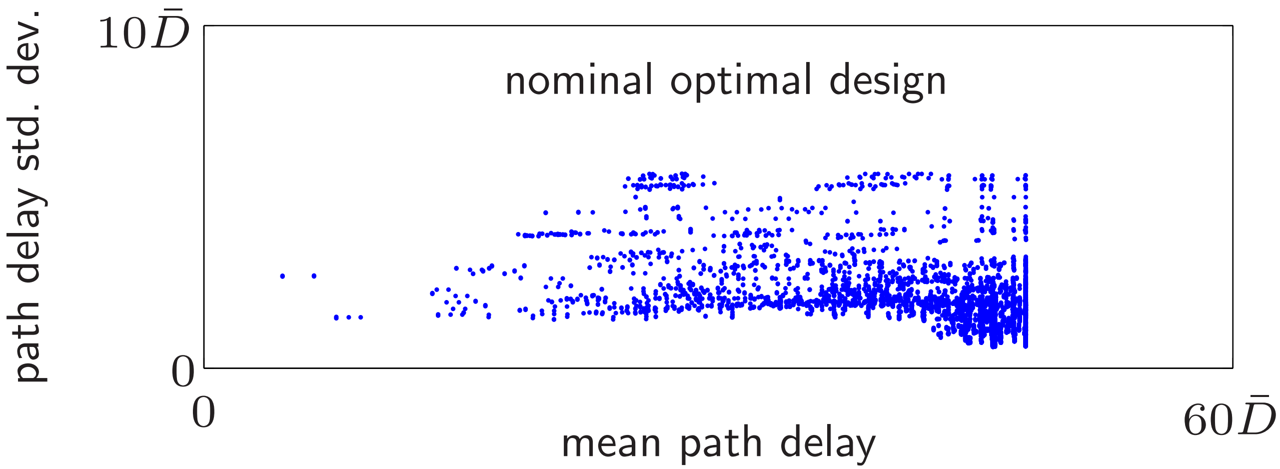
## Statistically robust design via new method

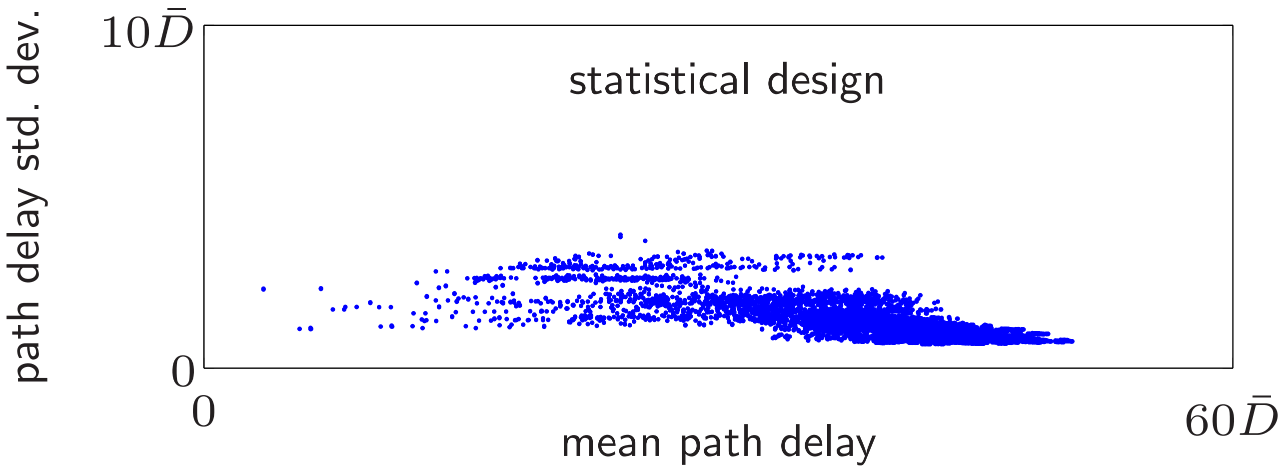
| | Nominal delay | 90% delay |
|---|---|---|
| Nominal design | **45.4** | 53.6 |
| Statistical design | 46.3 | **46.9** |

# Nominal optimal versus statistical design



# Path delay mean/std. dev. scatter plots



# Monomial and Posynomial Fitting

# A basic property of posynomials

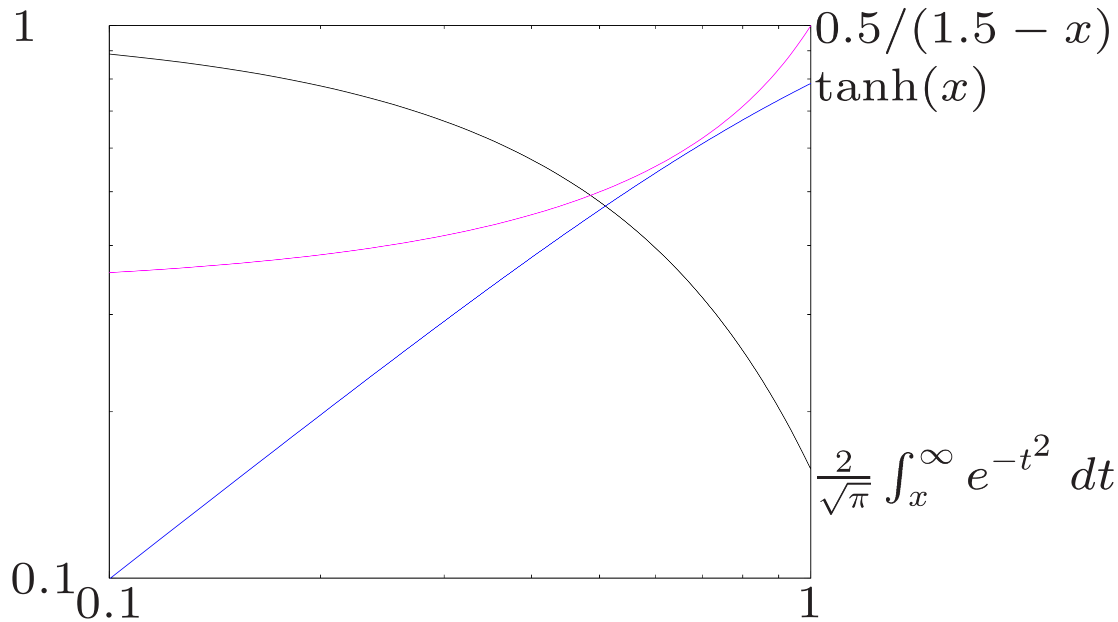
- if $f$ is a monomial, then $\log f(e^y)$ is **affine** (linear plus constant)
- if $f$ is a posynomial, then $\log f(e^y)$ is **convex**
- roughly speaking, a posynomial is convex when plotted on log-log plot
- midpoint rule for posynomial $f$:
  - let $z$ be elementwise geometric mean of $x$, $y$, $i.e.$, $z_i = \sqrt{x_i y_i}$
  - then $f(z) \leq \sqrt{f(x)f(y)}$
- a converse: if $\log \phi(e^y)$ is convex, then $\phi$ can be approximated as well as you like by a posynomial

# Monomial/posynomial approximation: Theory

when can a function $f$ be approximated by a monomial or generalized posynomial?

- form function $F(y) = \log f(e^y)$
- $f$ can be approximated by a monomial if and only if $F$ is nearly affine (linear plus constant)
- $f$ can be approximated by a generalized posynomial if and only if $F$ is nearly convex

# Examples




- $\tanh(x)$ can be reasonably well fit by a monomial
- $0.5/(1.5 - x)$ can be fit by a generalized posynomial
- $(2/\sqrt{\pi}) \int_x^\infty e^{-t^2}\, dt$ cannot be fit very well by a generalized posynomial

# What problems can be approximated by GGPs?

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

- transformed objective and inequality constraint functions $F_i(y) = \log f_i(e^y)$ must be nearly convex

- transformed equality constraint functions $G_i(y) = \log g_i(e^y)$ must be nearly affine

# Monomial fitting via log-regression

find coefficient $c > 0$ and exponents $a_1, \ldots, a_n$ of monomial $f$ so that

$$f(x^{(i)}) \approx f^{(i)}, \qquad i = 1, \ldots, N$$

- rewrite as

$$\begin{aligned} \log f(x^{(i)}) &= \log c + a_1 \log x_1^{(i)} + \cdots + a_n \log x_n^{(i)} \\ &\approx \log f^{(i)}, \qquad i = 1, \ldots, N \end{aligned}$$

- use least-squares (regression) to find $\log c$, $a_1, \ldots, a_n$ that minimize

$$\sum_{i=1}^{N} \left( \log c + a_1 \log x_1^{(i)} + \cdots + a_n \log x_n^{(i)} - \log f^{(i)} \right)^2$$

# Posynomial fitting via Gauss-Newton

find coefficients and exponents of posynomial $f$ so that

$$f(x^{(i)}) \approx f^{(i)}, \qquad i = 1, \ldots, N$$

- minimize sum of squared fractional errors

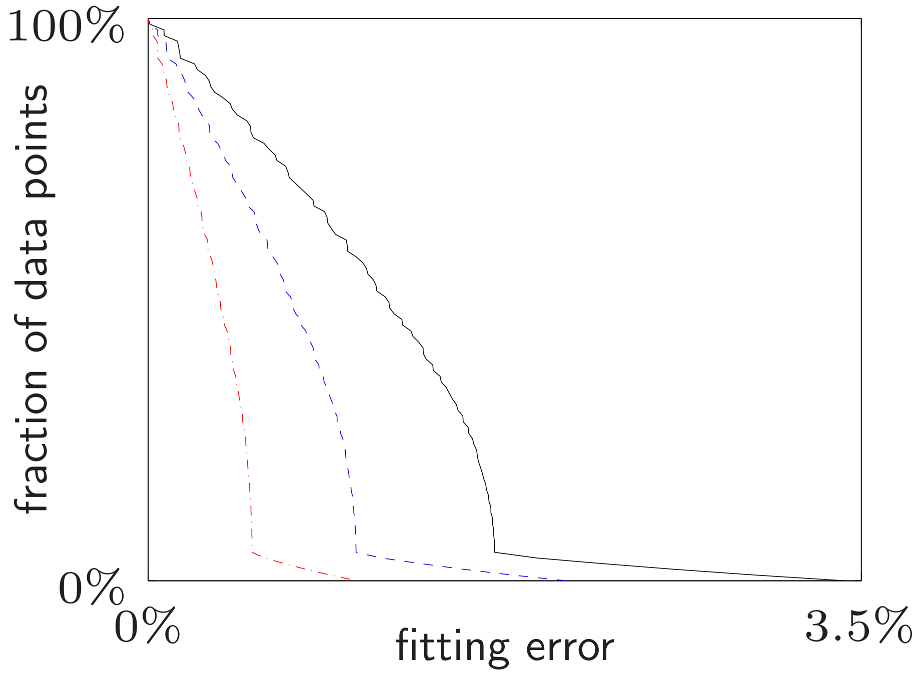
$$\sum_{i=1}^{N} \left( \frac{f^{(i)} - f(x^{(i)})}{f^{(i)}} \right)^2$$

  can be (locally) solved by Gauss-Newton method

- needs starting guess for coefficients, exponents

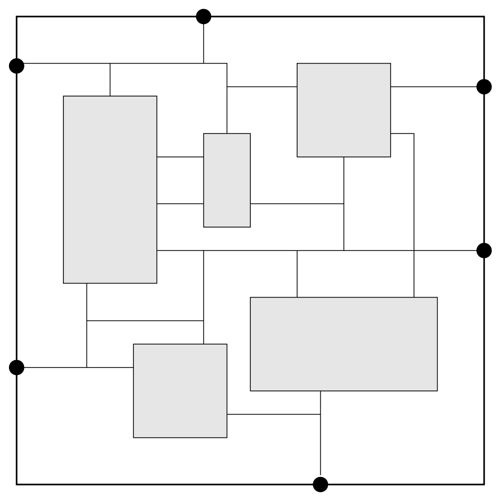
# Posynomial fitting example

- $1000$ data points from $f(x) = (1 - 0.5(x_1^2 + x_2 + x_3^{-1} - 1)^2)^{1/2}$ over $0.1 \le x_i \le 1$
- cumulative error distribution for $3$-, $5$-, and $10$-term posynomial fits



# Statistical Power and Ground Network Design

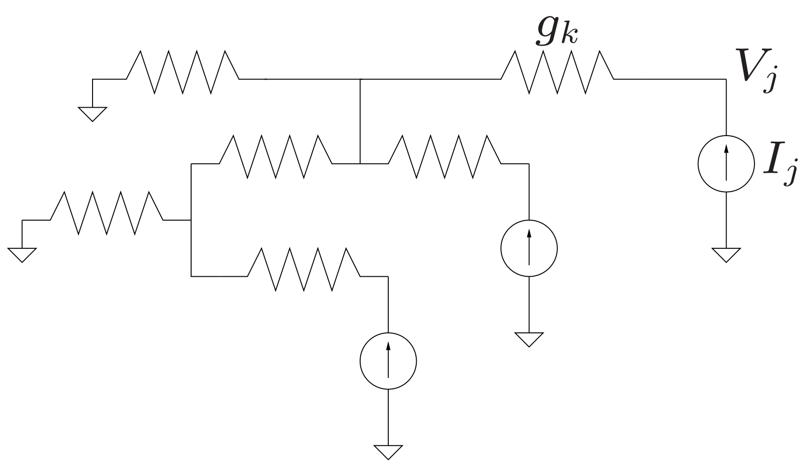
# Global power & ground network design



**Problem:** size wires (choose topology)

- minimize wire area subject to node voltage, current density constraints
- don't consider fast dynamics (C,L)
- **do consider (slow) variation in block currents**

# (Quasi-)static model




- segment conductance $g_k = w_k/(\rho l_k)$; current density $j_k = i_k/w_k$
- conductance matrix $G(w) = \sum_k w_k a_k a_k^T$; node voltages $V = G(w)^{-1} I$
- **statistical model for block currents:** $\mathbf{E}\, II^T = \Gamma$
  - $\Gamma$ is block current correlation matrix
  - $\Gamma_{jj}^{1/2} = \text{RMS}(I_j)$; $\Gamma_{ij}$ gives correlation between $I_i$, $I_j$

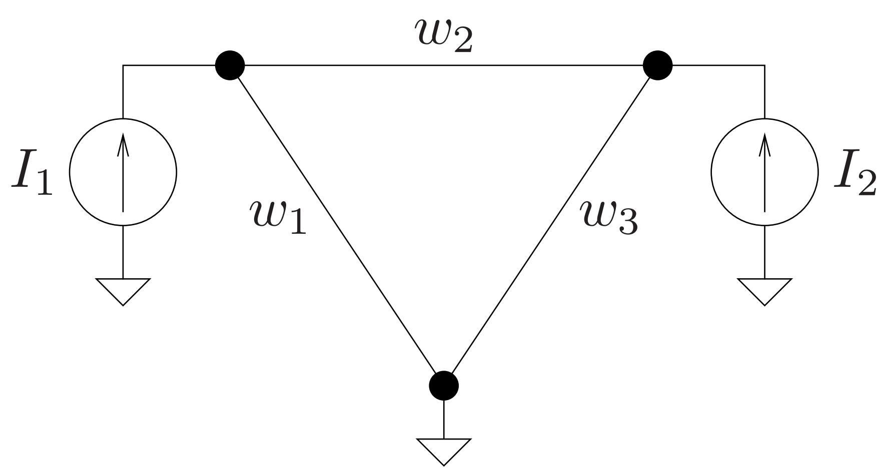
# Sizing problem

$$
\begin{array}{lll}
\text{minimize} & A = \sum_k l_k w_k & \text{(area)} \\
\text{subject to} & V_j \leq V_{\max} & \text{(node voltage limit)} \\
& \mathbf{E}\, j_k^2 \leq j_{\max}^2 & \text{(RMS current density limit)} \\
& w_k \geq 0 & \text{(nonneg. wire widths)}
\end{array}
$$

can't solve, except special case $I$ constant

- (Erhard & Johannes) can improve any mesh design by pruning to a tree
- (Chowdhury & Breuer) can size P&G trees via geometric programming

# Meshes, trees and current variation




- $I_1$, $I_2$ constant (or highly correlated): set $w_2 = 0$ (yields tree)
- $I_1$, $I_2$ anti-correlated: better to use $w_2 > 0$ (yields mesh)

# Average power formulation

- power dissipated in wires: $P = V^T I = I^T G(w)^{-1} I$
- average power: $\mathbf{E}\, P = \mathbf{E}\, I^T G(w)^{-1} I = \mathbf{Tr}\, G(w)^{-1} \Gamma$

$$\begin{array}{ll} \text{minimize} & \mathbf{Tr}\, G(w)^{-1}\Gamma + \mu \sum_k l_k w_k \quad \text{(average power } +\mu\cdot\text{area)} \\ \text{subject to} & w_k \geq 0 \end{array}$$

- parameter $\mu > 0$ trades off average power, area
- nonlinear but **convex problem**, readily (globally) solved
- indirectly limits $\mathbf{E}\, j_k^2$, $V_j$

# Properties of solution

observation: many $w_k$'s are zero, $i.e.$, many wires aren't used

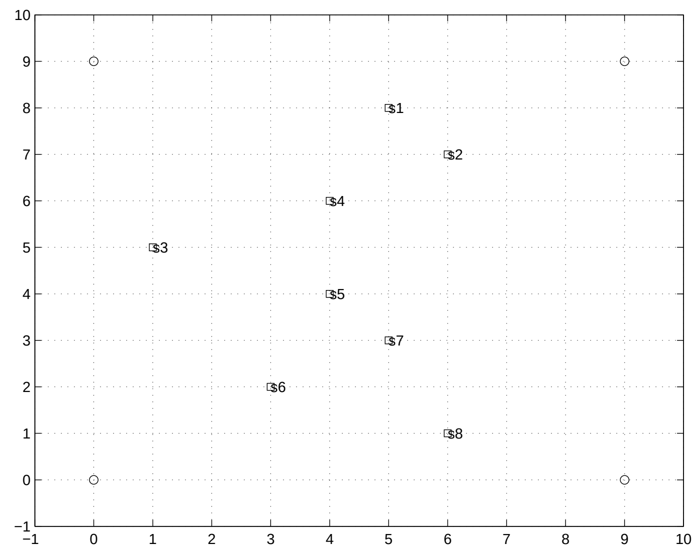
average power formulation can be used for **P&G topology selection:**

- start with lots of (potential) wires
- let average power formulation choose among them
- topology (given by nonzero $w_k$) independent of $\mu$

resulting current density and node voltages:

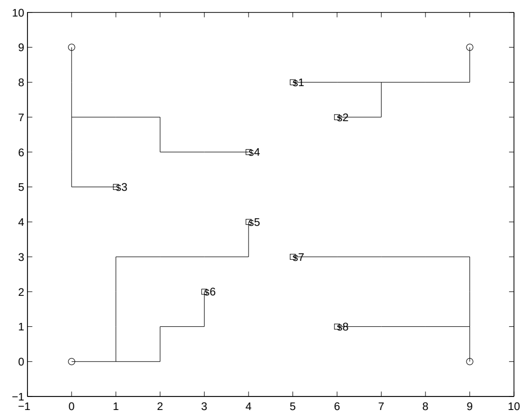
- RMS current density is equal in all (nonzero) segments
  in fact $\mu = \rho j_{\max}^2$ yields $\mathbf{E}\, j_k^2 = j_{\max}^2$ in all (nonzero) segments
- observation: $V_j$ are small

# Example



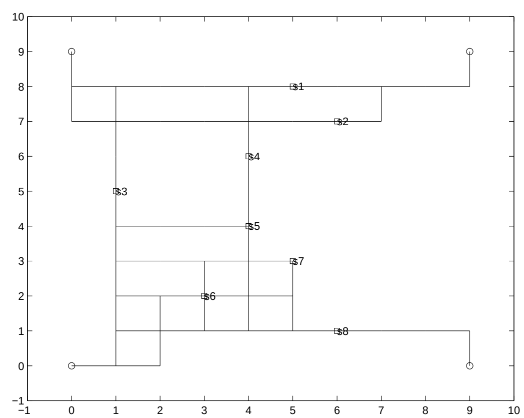

- $10 \times 10$ grid, each node connected to neighbors (180 segments)
- 8 current sources, $I \in \mathbf{R}^8$ is random with three possible values
- 4 ground pins on the perimeter (at corner points)

**design for constant currents** (with same RMS values)




- a tree; each source connected to nearest ground pin
- RMS current density 1, area = 448, max. voltage = 7.7

**design via average power formulation**




- mesh, not a tree
- RMS current density 1, area = 347, max. voltage = 5.7

# Barrier method

use Newton's method to minimize

$$\mathbf{Tr}\, G(w)^{-1}\Gamma + \mu l^T w - \beta^{(i)} \sum_k \log w_k$$

- barrier term $-\beta \sum_k \log w_k$ ensures $w_k > 0$
- solve for decreasing sequence of $\beta^{(i)}$
- can show $w^{(i)}$ is at most $n\beta^{(i)}$ suboptimal
- $O(n^3)$ cost per Newton step

works very well for $n < 1000$ or so; easy to add other convex constraints

# Pruning

- often clear in few iterations which $w_k$ are converging to $0$
- removing these $w_k$ early greatly speeds up convergence
- sizes 1000s of $w_k$s in minutes

## Where $\Gamma$ comes from

- from simulation: $\Gamma = \dfrac{1}{T_{\rm sim}} \displaystyle\int_0^{T_{\rm sim}} I(t)I(t)^T \, dt$
- or, from block RMS currents and estimates of correlation:

$$\Gamma_{ij} = \mathsf{RMS}(I_i)\ \mathsf{RMS}(I_j)\ \rho_{ij}$$

- can use eigenvalue decomposition to simplify $\Gamma$

$$\Gamma = \sum_i \lambda_i q_i q_i^T, \qquad \hat{\Gamma} = \sum_{i=1}^{r} \lambda_i q_i q_i^T$$

(reduced rank approximation speeds up avg. pwr. solution)

# Observations

- P&G meshes outperform trees when current variation taken into account
- Average power formulation
  - yields tractable convex optimization problem
  - chooses topology
  - guarantees RMS current density limit
  - indirectly limits node voltages

# Conclusions

# 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

**Do we remember what this talk has been about?**

**Let me re·emphasize moral!**

# 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

- looking for keys under street light
  (not where keys were lost, but lighting is good)

- forcing problems into GP-compatible form
  (problems aren't GPs, but solving is good)

- can achieve robust and statistically better design
  even though cannot do good statistical analysis

# References

- *A tutorial on geometric programming*
- *Digital circuit sizing via geometric programming*
- *Analog circuit design via geometric programming*
- *Convex optimization*, Cambridge Univ. Press 2004

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

# Thank You

e-mail: sunghee_yun@hotmail.com