ELLIPSOID METHOD AND ITS AMAZING ORACLES*

WAI-SHING LUK†

Abstract. Ellipsoid method is revisited. Besides that, three separation oracles are investigated for applications. They are robust optimization, semidefinite programming, and network optimization. Discuss the stability issue. Finally, the parallel cut is described.

1. Introduction. The bad reputation of the ellipsoid method is not good. And that is unfair. It is commonly believed that the method is inefficient in practice for large-scale convex problems. The convergent rate is slow. It cannot exploits sparsity. It was supplanted by the interior-point methods. It can be treated as a theoretical tool for proving the polynomial-time solvability of combinatorial optimization problems.

However, the ellipsoid method works very differently compared with the interior point method. Also, it only requires a separation oracle. Thus, it can play nicely with other techniques. Consider Ellipsoid Method When the number of optimization variables is moderate, e.g. ECO flow, analog circuit sizing, parametric problems. The number of constraints is large, or even infinite. Whenever separation oracle can be implemented efficiently.

2. Cutting-plane Method Revisited.

2.1. Convex Feasibility Problem. Let $\mathcal{K} \subseteq \mathbb{R}^n$ be a convex set. Consider the feasibility problem:

1. Find a point $x^* \in \mathbb{R}^n$ in $\mathcal{K}$, or
2. Determine that $\mathcal{K}$ is empty (i.e., no feasible solution)

When a *separation oracle* Ω is *queried* at x_0, it either

1. Asserts that $x_0 \in \mathcal{K}$, or
2. Returns a separating hyperplane between x_0 and $\mathcal{K}$:

$$g^\top (x - x_0) + \beta \leq 0, \beta \geq 0, g \neq 0, \ \forall x \in \mathcal{K}. \tag{2.1}$$

The pair (g, h) is called a *cutting-plane*, or cut, since it eliminates the halfspace $\{x \mid g^\top (x - x_0) + h > 0\}$ from our search. If $h = 0$ (x_0 is on the boundary of halfspace that is cut), cutting-plane is called *neutral cut*. If $h > 0$ (x_0 lies in the interior of halfspace that is cut), cutting-plane is called *deep cut*.

The $\mathcal{K}$ is usually given by a set of inequalities $f_j(x) \leq 0$ or $f_j(x) < 0$ for $j = 1 \cdots m$, where $f_j(x)$ is a convex function. A vector $g \equiv \partial f(x_0)$ is called a *subgradient* of a convex function f at x_0 if $f(z) \geq f(x_0) + g^{\mathrm{T}}(z - x_0)$. Hence, the cut (g, h) is given by $(\partial f(x_0), f(x_0))$.

Note that if $f(x)$ is differentiable, we can simply take $\partial f(x_0) = \nabla f(x_0)$. Cutting-plane method consists of two key components: separation oracle Ω and a search space $\mathcal{S}$ initially big enough to cover $\mathcal{K}$. For example,

*This work was supported by the Society for Industrial and Applied Mathematics

†Fudan University

- Polyhedron $\mathcal{P} = \{z \mid Cz \preceq d\}$.
- Interval $\mathcal{I} = [l, u]$ (for one-dimensional problem).
- Ellipsoid $\mathcal{E} = \{z \mid (z - x_c)P^{-1}(z - x_c) \leq 1\}$.

Generic Cutting-plane method:

- **Given** initial $\mathcal{S}$ known to contain $\mathcal{K}$.
- **Repeat**
  1. Choose a point x_0 in $\mathcal{S}$.
  2. Query the cutting-plane oracle at x_0.
  3. **If** $x_0 \in \mathcal{K}$, quit.
  4. **Else**, update $\mathcal{S}$ to a smaller set that covers:

$$\mathcal{S}^+ = \mathcal{S} \cap \{z \mid g^\top (z - x_0) + h \leq 0\}.$$

  5. **If** $\mathcal{S}^+ = \emptyset$ or it is small enough, quit.

2.2. Convex Optimization Problem. Consider:

$$\begin{array}{ll} \text{minimize} & f_0(x), \\ \text{subject to} & x \in \mathcal{K}. \end{array} \tag{2.2}$$

The optimization problem is treated as a feasibility problem with an additional constraint $f_0(x) < t$. Here, $f_0(x)$ could be a convex function or a quasiconvex function. t is the best-so-far value of $f_0(x)$. The problem can be reformulated as:

$$\begin{array}{ll} \text{minimize} & t, \\ \text{subject to} & \Phi(x, t) < 0, \\ & x \in \mathcal{K}, \end{array} \tag{2.3}$$

where $\Phi(x, t) < 0$ is the t-sublevel set of $f_0(x)$. Note that $\mathcal{K}_t \subseteq \mathcal{K}_u$ if and only if $t \leq u$ (monotonicity). One easy way to solve the optimization problem is to apply the binary search on t.

Another possible way is, to update the best-so-far t whenever a feasible solution x_0 is found such that $\Phi(x_0, t) = 0$. We assume that the oracle takes responsibility for that.

Generic Cutting-plane method (Optim)

- **Given** initial $\mathcal{S}$ known to contain $\mathcal{K}_t$.
- **Repeat**
  1. Choose a point x_0 in $\mathcal{S}$
  2. Query the separation oracle at x_0
  3. **If** $x_0 \in \mathcal{K}_t$, update t such that $\Phi(x_0, t) = 0$.
  4. Update $\mathcal{S}$ to a smaller set that covers:

$$\mathcal{S}^+ = \mathcal{S} \cap \{z \mid g^\top (z - x_0) + h \leq 0\}$$

  5. **If** $\mathcal{S}^+ = \emptyset$ or it is small enough, quit.

2.3. Example: Profit Maximization Problem. Consider:

$$\begin{array}{ll} \text{maximize} & p(Ax_1^\alpha x_2^\beta) - v_1 x_1 - v_2 x_2, \\ \text{subject to} & x_1 \leq k, \end{array} \tag{2.4}$$

where p is the market price per unit, A is the scale of production, $p(Ax_1^\alpha x_2^\beta)$ is the Cobb-Douglas production function, (α, β) is the output elasticities, x is the input quantity, v is the output price, and k is a given constant that restricts the quantity of x_1.

The formulation is not in the convex form. Rewrite the problem in the following form:

$$\begin{array}{ll} \text{maximize} & t, \\ \text{subject to} & t + v_1 x_1 + v_2 x_2 < pAx_1^\alpha x_2^\beta, \\ & x_1 \le k. \end{array}$$

By taking the logarithm of each variable, we have the problem in a convex form:

$$\begin{array}{ll} \max & t, \\ \text{s.t.} & \log(t + v_1 e^{y_1} + v_2 e^{y_2}) - (\alpha y_1 + \beta y_2) < \log(pA), \\ & y_1 \le \log k, \end{array} \tag{2.5}$$

where $y_1 = \log x_1$, $y_2 = \log x_2$.

3. Area of Applications.

- Robust convex optimization
  - oracle technique: affine arithmetic
- Parametric network potential problem
  - oracle technique: negative cycle detection
- Semidefinite programming
  - oracle technique: Cholesky factorization

3.1. Robust Convex Optimization.

Consider:

$$\begin{array}{ll} \text{minimize} & \sup_{q \in \mathbb{Q}} f_0(x, q), \\ \text{subject to} & f_j(x, q) \le 0, \ \forall q \in \mathbb{Q}, \ j = 1, 2, \cdots, m, \end{array} \tag{3.1}$$

where q represents a set of varying parameters. The problem can be reformulated as:

$$\begin{array}{ll} \text{minimize} & t, \\ \text{subject to} & f_0(x, q) < t, \\ & f_j(x, q) \le 0, \ \forall q \in \mathbb{Q}, \ j = 1, 2, \cdots, m. \end{array}$$

The oracle only needs to determine:

- If $f_j(x_0, q) > 0$ for some j and $q = q_0$, then
- the cut $(g, h) = (\partial f_j(x_0, q_0), f_j(x_0, q_0))$
- If $f_0(x_0, q) \ge t$ for some $q = q_0$, then
- the cut $(g, h) = (\partial f_0(x_0, q_0), f_0(x_0, q_0) - t)$
- Otherwise, x_0 is feasible, then
- Let $q_{\max} = \text{argmax}_{q \in \mathbb{Q}} f_0(x_0, q)$.
- $t := f_0(x_0, q_{\max})$.
- The cut $(g, h) = (\partial f_0(x_0, q_{\max}), 0)$

Random sampling trick.

3.1.1. Example: Profit Maximization Problem (convex). Consider

$$\begin{array}{ll} \max & t \\ \text{s.t.} & \log(t + \hat{v}_1 e^{y_1} + \hat{v}_2 e^{y_2}) - (\hat{\alpha} y_1 + \hat{\beta} y_2) \le \log(\hat{p}\, A) \\ & y_1 \le \log \hat{k}. \end{array}$$

Now assume that $\hat{\alpha}$ and $\hat{\beta}$ vary $\bar{\alpha} \pm e_1$ and $\bar{\beta} \pm e_2$ respectively, where $\hat{p}$, $\hat{k}$, $\hat{v}_1$, and $\hat{v}_2$ all vary $\pm e_3$. By detail analysis, the worst case happens when:

- $p = \bar{p} + e_3$, $k = \bar{k} + e_3$
- $v_1 = \bar{v}_1 - e_3$, $v_2 = \bar{v}_2 - e_3$,
- if $y_1 > 0$, $\alpha = \bar{\alpha} - e_1$, else $\alpha = \bar{\alpha} + e_1$
- if $y_2 > 0$, $\beta = \bar{\beta} - e_2$, else $\beta = \bar{\beta} + e_2$

Remark: for more complicated problems, affine arithmetic could be used.

3.2. Parametric Network Potential Problem. Given a network represented by a directed graph $G = (V, E)$. Consider:

$$\begin{array}{ll} \text{minimize} & t, \\ \text{subject to} & u_i - u_j \le h_{ij}(x, t),\ \forall (i,j) \in E, \\ \text{variables} & x, u, \end{array}$$

where $h_{ij}(x, t)$ is the weight function of edge (i, j).

Assume that the network is large but the number of parameters is small. Given x and t, the problem has a feasible solution if and only if G contains no negative cycle. Let $\mathcal{C}$ be a set of all cycles of G. The problem can be formulated as:

$$\begin{array}{ll} \text{minimize} & t, \\ \text{subject to} & W_k(x, t) \ge 0, \forall C_k \in C, \\ \text{variables} & x, \end{array}$$

where C_k is a cycle of G:

$$W_k(x, t) = \sum_{(i,j) \in C_k} h_{ij}(x, t).$$

The oracle only needs to determine:

- If there exists a negative cycle C_k under x_0, then
- the cut $(g, h) = (-\partial W_k(x_0), -W_k(x_0))$
- If $f_0(x_0) \ge t$, then
- the cut $(g, h) = (\partial f_0(x_0), f_0(x_0) - t)$
- Otherwise, x_0 is feasible, then
- $t := f_0(x_0)$.
- The cut $(g, h) = (\partial f_0(x_0), 0)$

3.2.1. Example: Optimal Matrix Scaling. Given a sparse matrix $A = [a_{ij}] \in \mathbb{R}^{N \times N}$. Find another matrix $B = UAU^{-1}$ where U is a nonnegative diagonal matrix, such that the ratio of any two elements of B in absolute value is as close to 1 as possible.

Let $U = \mathrm{diag}([u_1, u_2, \ldots, u_N])$. Under the min-max-ratio criterion, the problem can be formulated as:

$$\begin{array}{ll} \text{minimize} & \pi/\psi \\ \text{subject to} & \psi \le u_i |a_{ij}| u_j^{-1} \le \pi,\ \forall a_{ij} \neq 0, \\ & \pi,\ \psi, u, \text{positive} \\ \text{variables} & \pi, \psi, u\,. \end{array}$$

By taking the logarithms of variables, the above problem can be transformed into:

$$\begin{array}{ll} \text{minimize} & t \\ \text{subject to} & \pi' - \psi' \le t \\ & u'_i - u'_j \le \pi' - a'_{ij},\ \forall a_{ij} \neq 0\,, \\ & u'_j - u'_i \le a'_{ij} - \psi',\ \forall a_{ij} \neq 0\,, \\ \text{variables} & \pi', \psi', u'\,. \end{array}$$

where k' denotes $\log(|k|)$ and $x = (\pi', \psi')^\top$.

Consider the following problem:

$$\begin{array}{ll} \text{minimize} & t, \\ \text{subject to} & F(x,t) \succeq 0, \end{array}$$

where $F(x,t)$ is a matrix-valued function, $A \succeq 0$ denotes A is positive semidefinite. Recall that a matrix A is positive semidefinite if and only if $v^\top A v \ge 0$ for all $v \in \mathbb{R}^N$. The problem can be transformed into:

$$\begin{array}{ll} \text{minimize} & t, \\ \text{subject to} & v^\top F(x,t) v \ge 0,\ \forall v \in \mathbb{R}^N. \end{array}$$

Consider $v^\top F(x,t)v$ is concave for all $v \in \mathbb{R}^N$ w.r.t. x, then the above problem is a convex programming. Reduce to *semidefinite programming* if $F(x,t)$ is linear w.r.t. x, i.e., $F(x) = F_0 + x_1 F_1 + \cdots + x_n F_n$.

The oracle only needs to:

- Perform a *row-based* Cholesky factorization such that $F(x_0, t) = R^\top R$.
- Let $A_{:p,:p}$ denotes a submatrix $A(1:p, 1:p) \in \mathbb{R}^{p \times p}$.
- If Cholesky factorization fails at row p,
  - there exists a vector $e_p = (0, 0, \cdots, 0, 1)^\top \in \mathbb{R}^p$, such that
    - $v = R^{-1}_{:p,:p} e_p$, and
    - $v^\top F_{:p,:p}(x_0) v < 0$.
  - The cut $(g, h) = (-v^\top \partial F_{:p,:p}(x_0) v, -v^\top F_{:p,:p}(x_0) v)$

3.2.2. Example: Matrix Norm Minimization. Let $A(x) = A_0 + x_1 A_1 + \cdots + x_n A_n$. Problem $\min_x \|A(x)\|$ can be reformulated as

$$\begin{array}{ll} \text{minimize} & t, \\ \text{subject to} & \begin{pmatrix} t\,I & A(x) \\ A^\top(x) & t\,I \end{pmatrix} \succeq 0. \end{array}$$

Binary search on t can be used for this problem.

3.2.3. Example: Estimation of Correlation Function.

$$\begin{array}{ll} \min_{\kappa,p} & \|\Omega(p) + \kappa I - Y\| \\ \text{s. t.} & \Omega(p) \succcurlyeq 0, \kappa \geq 0 \ . \end{array}$$

Let $\rho(h) = \sum_i^n p_i \Psi_i(h)$, where p_i's are the unknown coefficients to be fitted Ψ_i's are a family of basis functions. The covariance matrix $\Omega(p)$ can be recast as:

$$\Omega(p) = p_1 F_1 + \cdots + p_n F_n,$$

where $\{F_k\}_{i,j} = \Psi_k(\|s_j - s_i\|_2)$.

4. Ellipsoid Method Revisited. Some History of Ellipsoid Method. Introduced by Shor and Yudin and Nemirovskii in 1976. Used to show that linear programming (LP) is polynomial-time solvable (Kachiyan 1979), settled the long-standing problem of determining the theoretical complexity of LP. In practice, however, the simplex method runs much faster than the method, although its worst-case complexity is exponential.

4.1. Basic Ellipsoid Method. An ellipsoid $\mathcal{E}(x_c, P)$ is specified as a set

$$\{x \mid (x - x_c)P^{-1}(x - x_c) \leq 1\},$$

where x_c is the center of the ellipsoid.

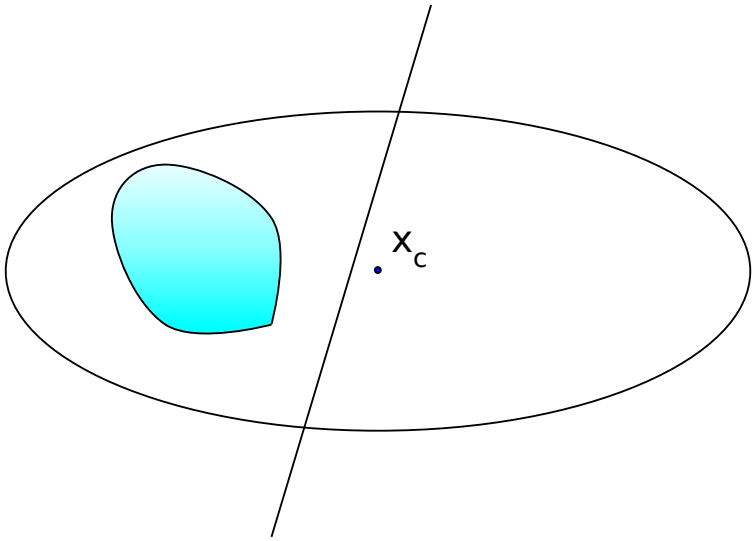

Figure 4.1: Ellipsoid, feasible region, and cut

Updating the ellipsoid (deep-cut)

Calculation of minimum volume ellipsoid covering:

$$\mathcal{E} \cap \{z \mid g^\top (z - x_c) + h \leq 0\}$$

Let $\tilde{g} = P\, g$, $\tau^2 = g^\top P g$.

- If $n \cdot h < -\tau$ (shallow cut), no smaller ellipsoid can be found.
- If $h > \tau$, intersection is empty.

Otherwise,

$$x_c^+ = x_c - \frac{\rho}{\tau^2}\tilde{g}, \qquad P^+ = \delta \cdot \left(P - \frac{\sigma}{\tau^2}\tilde{g}\tilde{g}^\top\right)$$

where

$$\rho = \frac{\tau + nh}{n+1}, \qquad \sigma = \frac{2\rho}{\tau + h}, \qquad \delta = \frac{n^2(\tau^2 - h^2)}{(n^2-1)\tau^2}$$

Even better, split P into two variables $\kappa \cdot Q$. Let $\tilde{g} = Q \cdot g$, $\omega = g^\top \tilde{g}$, $\tau = \sqrt{\kappa \cdot \omega}$.

$$x_c^+ = x_c - \frac{\rho}{\omega}\tilde{g}, \qquad Q^+ = Q - \frac{\sigma}{\omega}\tilde{g}\tilde{g}^\top, \qquad \kappa^+ = \delta \cdot \kappa$$

Reduce n^2 multiplications per iteration. Note that:

- The determinant of Q decreases monotonically.
- The range of δ is $(0, \frac{n^2}{n^2-1})$

4.2. Central Cut. A Special case of deep cut when $\beta = 0$. Deserve a separate implement because it is much simplier. Let $\tilde{g} = Q\,g$, $\tau = \sqrt{\kappa \cdot \omega}$,

$$\rho = \frac{\tau}{n+1}, \qquad \sigma = \frac{2}{n+1}, \qquad \delta = \frac{n^2}{n^2-1}$$

4.3. Parallel Cuts. Oracle returns a pair of cuts instead of just one. The pair of cuts is given by g and (β_1, β_2) such that:

$$g^\top(x - x_c) + \beta_1 \le 0,$$
$$g^\top(x - x_c) + \beta_2 \ge 0,$$

for all $x \in \mathcal{K}$.

Only linear inequality constraint can produce such parallel cut:

$$l \le a^\top x + b \le u, \qquad L \preceq F(x) \preceq U$$

Usually, provide faster convergence.

Updating the ellipsoid

Let $\tilde{g} = Q\,g$, $\tau^2 = \kappa \cdot \omega$.

- If $\beta_1 > \beta_2$, intersection is empty.
- If $\beta_1\beta_2 < -\tau^2/n$, no smaller ellipsoid can be found.
- If $\beta_2^2 > \tau^2$, it reduces to deep-cut with $\alpha = \alpha_1$.

Otherwise,

$$x_c^+ = x_c - \frac{\rho}{\omega}\tilde{g}, \qquad Q^+ = Q - \frac{\sigma}{\omega}\tilde{g}\tilde{g}^\top, \qquad \kappa^+ = \delta\kappa$$

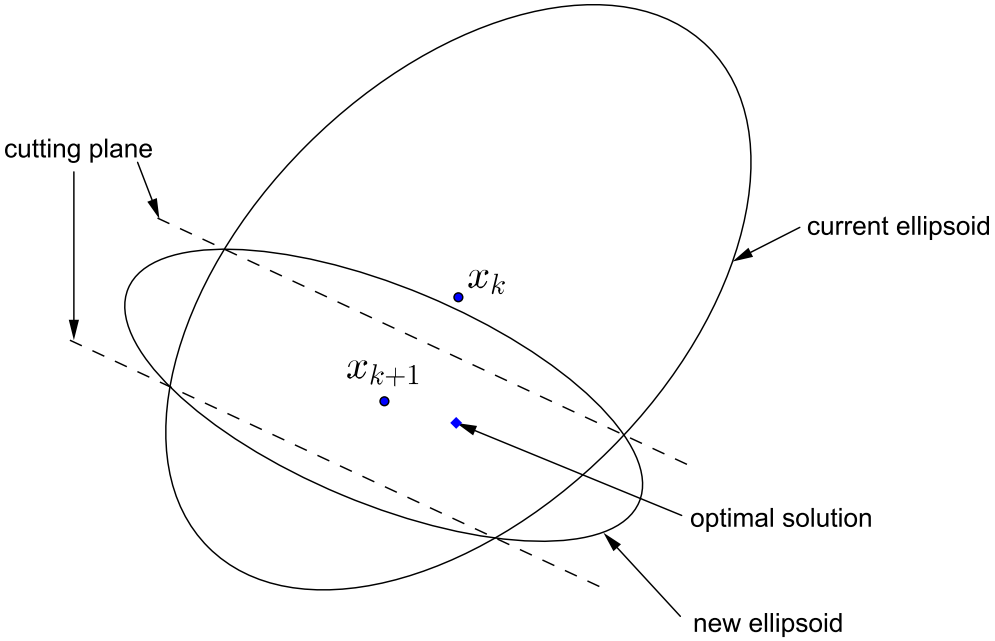

Figure 4.2: Parallel cuts

where

$$\begin{aligned}
\bar{\beta} &= (\beta_1 + \beta_2)/2 \\
\xi^2 &= (\tau^2 - \beta_1^2)(\tau^2 - \beta_2^2) + (n(\beta_2 - \beta_1)\bar{\beta})^2, \\
\sigma &= (n + (\tau^2 - \beta_1\beta_2 - \xi)/(2\bar{\beta}^2))/(n+1), \\
\rho &= \bar{\beta} \cdot \sigma, \\
\delta &= (n^2/(n^2-1))(\tau^2 - (\beta_1^2 + \beta_2^2)/2 + \xi/n)/\tau^2.
\end{aligned}$$

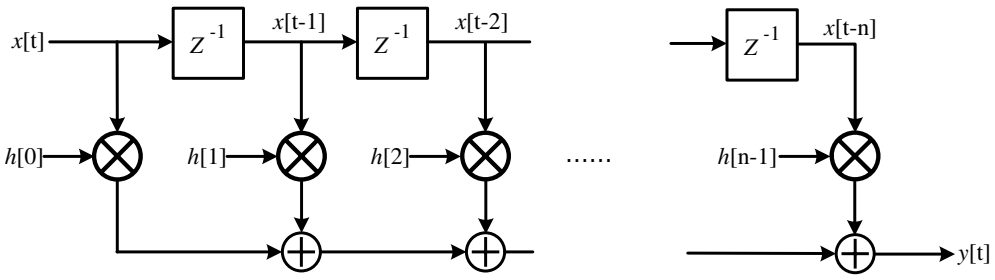

Figure 4.3: FIR filter structure

4.3.1. Example: FIR filter design. The time response is:

$$y[t] = \sum_{k=0}^{n-1} h[k]u[t-k].$$

The frequency response:

$$H(\omega) \;=\; \sum_{m=0}^{n-1} h(m)e^{-jm\omega}.$$

The magnitude constraints on frequency domain are expressed as

$$L(\omega) \;\le\; |H(\omega)| \;\le\; U(\omega),\; \forall\, \omega \in (-\infty, +\infty),$$

where $L(\omega)$ and $U(\omega)$ are the lower and upper (nonnegative) bounds at frequency ω respectively.

Note that the constraint is non-convex in general.However, via *spectral factorization*, it can transform into a convex one [Wu et al., 1999]:

$$L^2(\omega) \;\le\; R(\omega) \;\le\; U^2(\omega),\; \forall\, \omega \in (0, \pi),$$

where $R(\omega) = \sum_{i=-1+n}^{n-1} r(t)e^{-j\omega t} = |H(\omega)|^2$. The vector $\mathbf{r} = (r(-n+1), r(-n+2), ..., r(n-1))$ contains the autocorrelation coefficients, which can be determined by $\mathbf{h}$:

$$r(t) \;=\; \sum_{i=-n+1}^{n-1} h(i)h(i+t),\; t \in \mathbf{Z},$$

where $h(t) = 0$ for $t < 0$ or $t > n-1$. The whole problem can be formulated as:

$$\begin{array}{ll} \min & \gamma \\ \text{s.t.} & L^2(\omega) \le R(\omega) \le U^2(\omega),\; \forall \omega \in [0, \pi], \\ & R(\omega) > 0, \forall \omega \in [0, \pi]. \end{array}$$

4.3.2. Example: Maximum Likelihood estimation. Consider

$$\begin{array}{ll} \min_{\kappa,p} & \log\det(\Omega(p) + \kappa \cdot I) + \mathrm{Tr}((\Omega(p) + \kappa \cdot I)^{-1} Y), \\ \text{s.t.} & \Omega(p) \succeq 0, \kappa \ge 0. \end{array}$$

Note that the 1st term is concave, the 2nd term is convex. However, if there are enough samples such that Y is a positive definite matrix, then the function is convex within $[0, 2Y]$. Therefore, the following problem is convex:

$$\begin{array}{ll} \min_{\kappa,p} & \log\det V(p) + \mathrm{Tr}(V(p)^{-1} Y), \\ \text{s.t.} & \Omega(p) + \kappa \cdot I = V(p) \\ & 0 \preceq V(p) \preceq 2Y, \kappa{>}0. \end{array}$$

4.4. Discrete Optimization. Many engineering problems can be formulated as a convex/geometric programming, e.g. digital circuit sizing. Yet in an ASIC design, often there is only a limited set of choices from the cell library. In other words, some design variables are discrete. The discrete version can be formulated as Mixed-Integer Convex programming (MICP) by mapping the design variables to integers.

What's wrong with the existing methods? Mostly based on relaxation. Then use the relaxed solution as a lower bound and use the branch–and–bound method for the

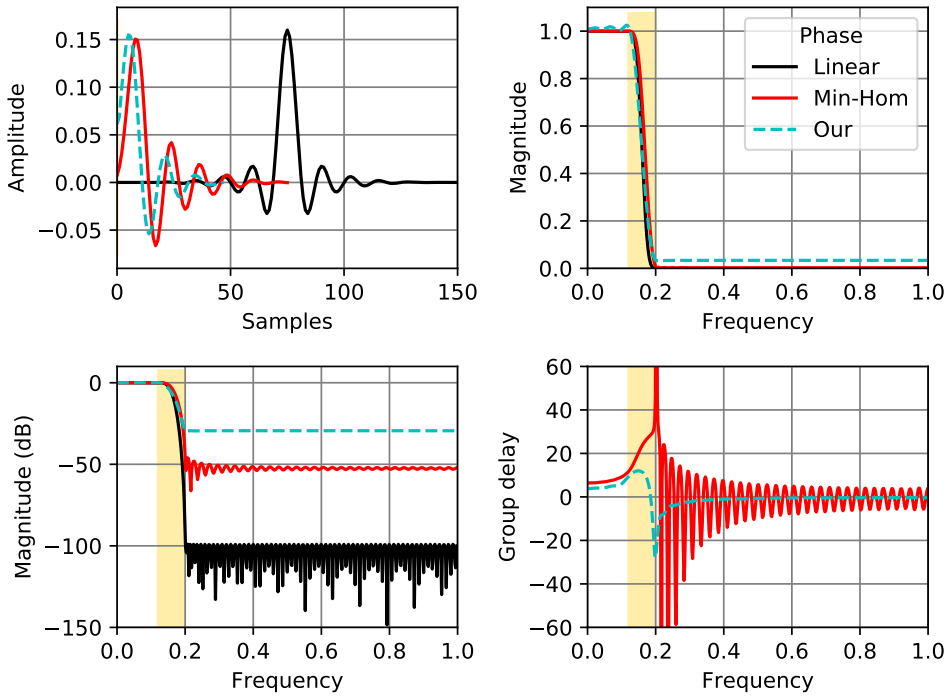

Figure 4.4: Result

discrete optimal solution. Note that the branch-and-bound method does not utilize the convexity of the problem. What if I can only evaluate constraints on discrete data?Workaround: convex fitting?

Consider:

$$\begin{array}{ll} \text{minimize} & f_0(x), \\ \text{subject to} & f_j(x) \leq 0,\ \forall j = 1, 2, \ldots, \\ & x \in \mathbb{D}, \end{array}$$

where $f_0(x)$ and $f_j(x)$ are "convex". Note that some design variables are discrete. The oracle looks for the nearby discrete solution x_d of x_c with the cutting-plane:

$$g^\top (x - x_d) + \beta \leq 0, \beta \geq 0, g \neq 0.$$

Note that the cut may be a shallow cut. Suggestion: use different cuts as possible for each iteration (e.g. round-robin the evaluation of constraints).

References.

Shao-Po Wu, Stephen Boyd, and Lieven Vandenberghe. FIR filter design via spectral factorization and convex optimization. In *Applied and Computational Control, Signals, and Circuits*, pages 215–245. Springer, 1999.

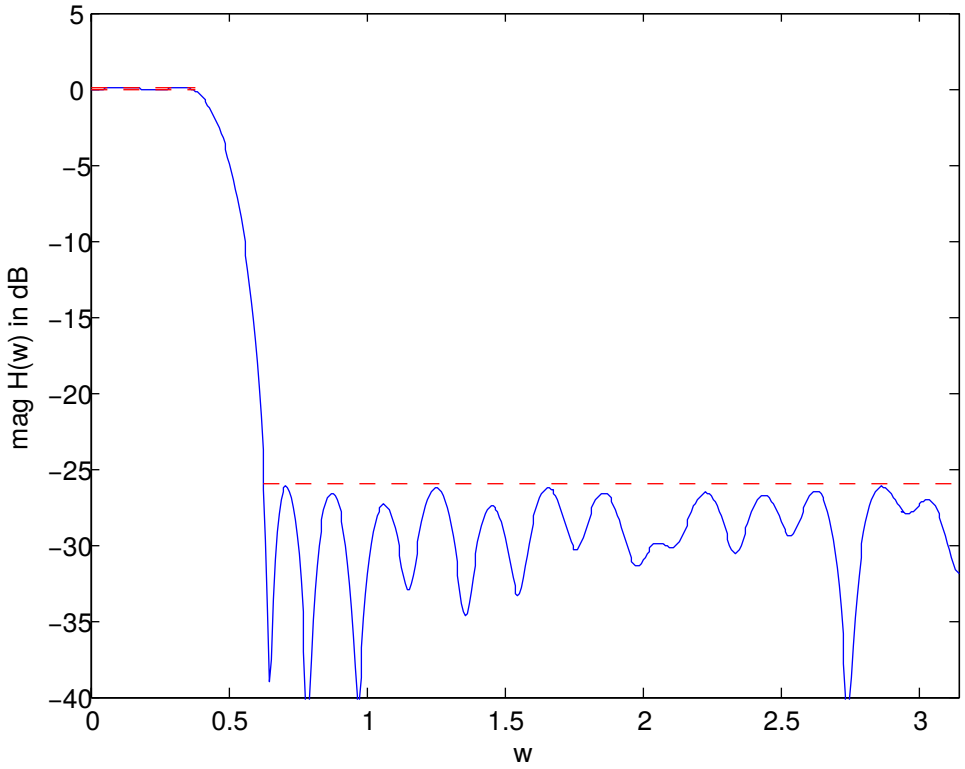

Figure 4.5: Lowpass ripple