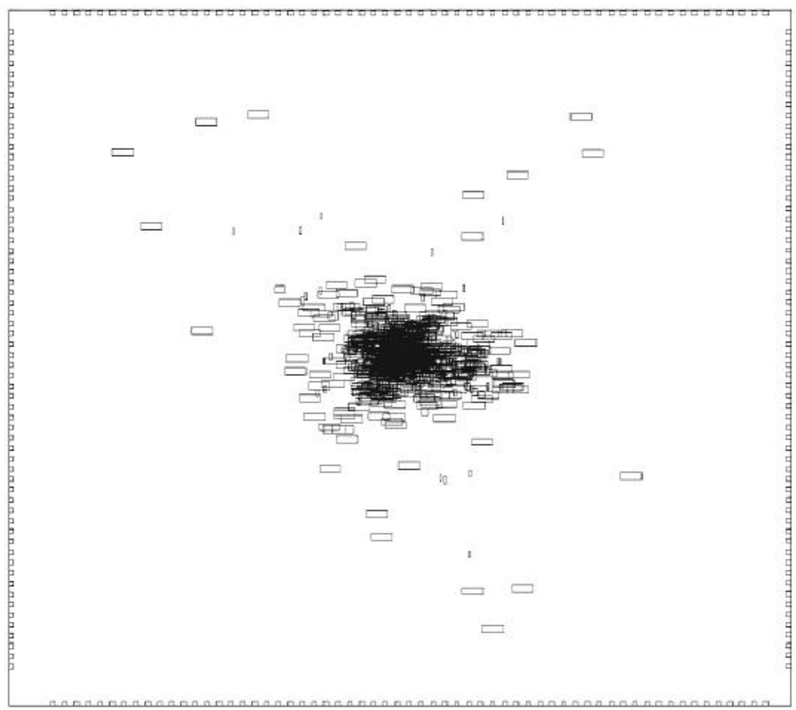
Ignoring Nonoverlapping Constraints

- Consider WL minimization alone
  - If no fixed pins, a trivial solution is to place all modules at the same place
  - If there are fixed pins (e.g., I/O pins at boundary), it tends to get a lot of overlaps at the center of the placement

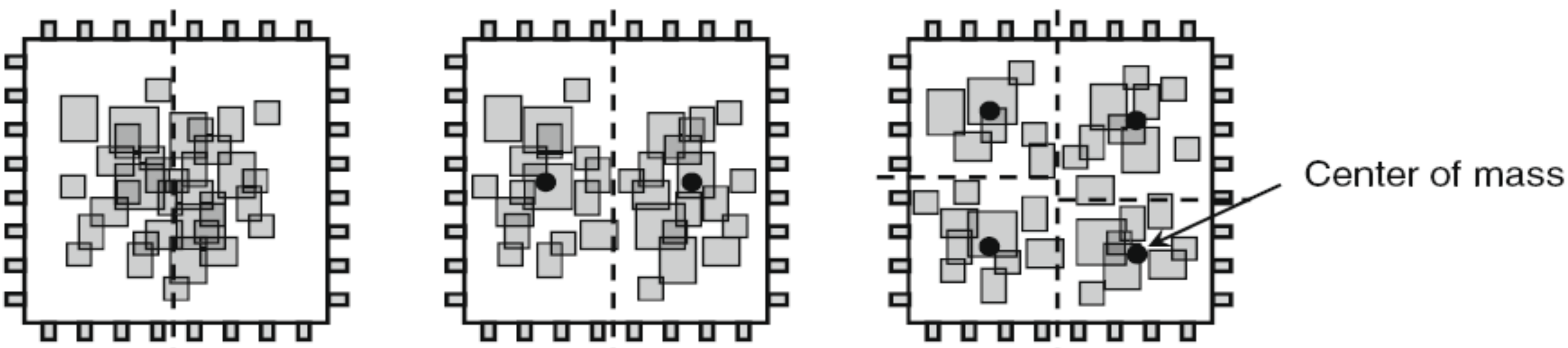


Handling Nonoverlapping Constraints

- Ways to make module distribution more even in quadratic placement
  - Adding center-of-mass constraints
  - Adding forces to pull modules from dense regions to sparse regions
- Constraints/forces are added in **an iterative manner** to gradually spread out the modules
- Transformed into a sequence of convex quadratic programs

Center-of-Mass Constraints in GORDIAN [TCAD-91]

- Given an uneven global placement solution
  - Find a good cut direction and position
    - Improve the cut value using FM
  - For each partition, add constraints that the center of gravity of cells should be in the center of region
    - The constraints are linear
  - Then perform quadratic placement again
    - Therefore, solving a single convex QP



Density-based Force by Kraftwerk [ICCAD-06]

- Pull cells away from dense to sparse regions
- Definitions:
  - $\boldsymbol{x}'$= vector of **current** placement positions
  - $\boldsymbol{x}$= vector of **new** placement positions to be determined
  - $\hat{\boldsymbol{x}}$= vector of **target** placement positions
    - Based on module density $D(x,y)$

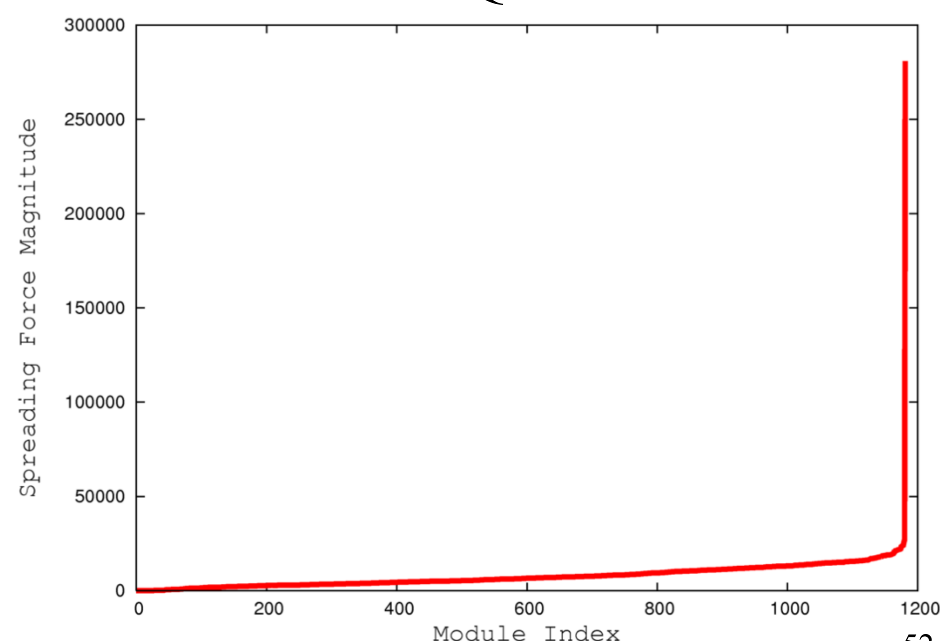
$$\Delta\phi = -D(x,\ y) \qquad \hat{x}_i = x'_i - \left|\frac{\partial\Phi}{\partial x}\right|_{(x'_i,\, y'_i)}$$

- **Hold force**: $\boldsymbol{F}_x^{hold} = -(\boldsymbol{Q}\boldsymbol{x}' + \boldsymbol{d}_x)$
- **Move force**: $\boldsymbol{F}_x^{move} = \hat{\boldsymbol{Q}}(\boldsymbol{x} - \hat{\boldsymbol{x}})$ here $\hat{\boldsymbol{Q}} = \mathrm{diag}(\hat{c}_i)$
- **Force equilibrium**:

$$(\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{d}_x) - (\boldsymbol{Q}\boldsymbol{x}' + \boldsymbol{d}_x) + \hat{\boldsymbol{Q}}(\boldsymbol{x} - \hat{\boldsymbol{x}}) = 0$$

Force-Vector Modulation in RQL [DAC-07]

- Some additional spreading forces are huge
  - A module is pulled far away from its natural position
  - Causes significant increase in WL
  - Only a few percent of all additional forces are huge
- Idea: Nullifies the huge forces before next QP
  - Correcting mistakes made during spreading
  - Significant WL reduction
  - Minor effect in spreading



Non-Quadratic Techniques

- Formulate the placement problem as a single non-linear & non-quadratic program

$$\text{Minimize} \sum_{e \in E} c_e \times \mathrm{WL}_e(\boldsymbol{x}, \boldsymbol{y})$$

$$\text{Subject to } D_b(\boldsymbol{x}, \boldsymbol{y}) <= T_b \text{ for all bin } b$$

  – $\mathrm{WL}_e()$ is continuously differentiable and more accurate in approximating HPWL than quadratic WL
  – Placement region is divided into **bins** such that non-overlapping constraints are replaced by **bin density constraints**

Choices of Wirelength Functions

HPWL $$\max_{v_i, v_j \in e, i<j} |x_i - x_j| + \max_{v_i, v_j \in e, i<j} |y_i - y_j|$$

Quadratic $$\sum_{e \in E} \Big(\sum_{v_i, v_j \in e, i<j} w_{ij} |x_i - x_j|^2 + \sum_{v_i, v_j \in e, i<j} w_{ij} |y_i - y_j|^2 \Big)$$

Log-Sum-Exp (LSE) $$\eta \sum_{e \in E} \Big(\log \sum_{v_k \in e} \exp(x_k/\eta) + \log \sum_{v_k \in e} \exp(-x_k/\eta) + \log \sum_{v_k \in e} \exp(y_k/\eta) + \log \sum_{v_k \in e} \exp(-y_k/\eta) \Big)$$

L_p-norm $$\sum_{e \in E} \Big(\big(\sum_{v_k \in e} x_k^p\big)^{\frac{1}{p}} - \big(\sum_{v_k \in e} x_k^{-p}\big)^{-\frac{1}{p}} + \big(\sum_{v_k \in e} y_k^p\big)^{\frac{1}{p}} - \big(\sum_{v_k \in e} y_k^{-p}\big)^{-\frac{1}{p}} \Big)$$

Choices of Wirelength Functions (cont'd)

CHKS

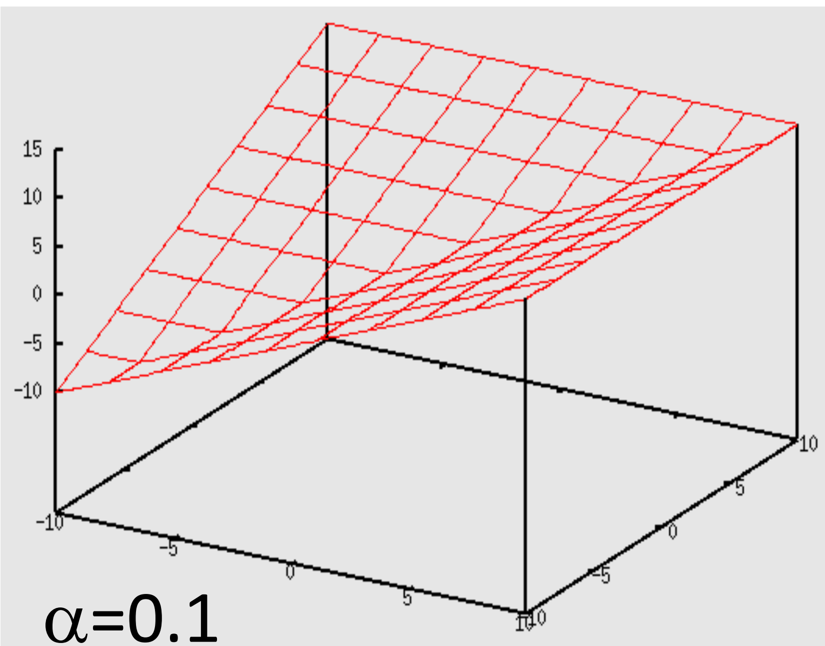
$$CHKS(x_1,x_2) = \frac{\sqrt{(x_1 - x_2)^2 + t^2} + x_1 + x_2}{2},$$

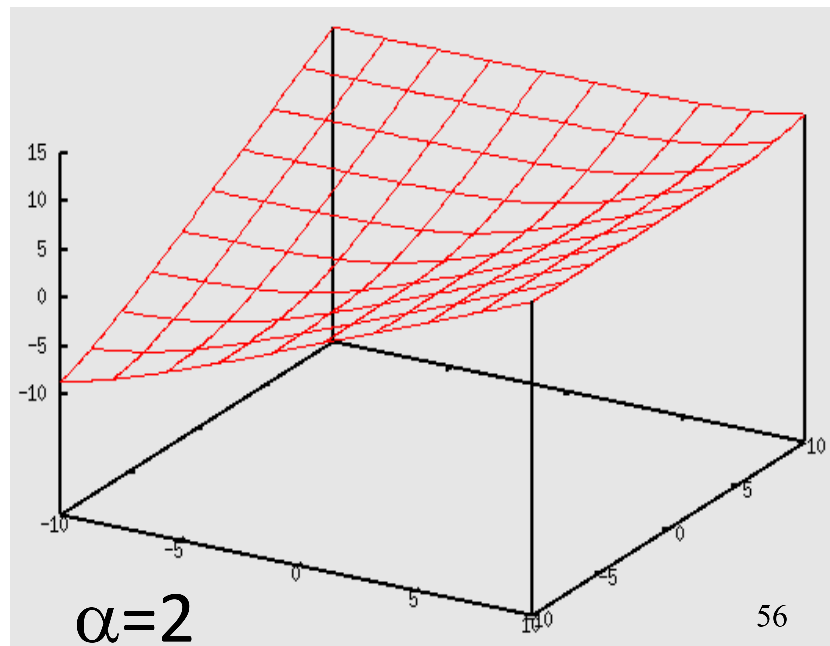
Weighted-average (WA)

$$\sum_{e\in E}\left(\frac{\sum_{v_i\in e} x_i \exp(x_i/\gamma)}{\sum_{v_i\in e}\exp(x_i/\gamma)} - \frac{\sum_{v_i\in e} x_i \exp(-x_i/\gamma)}{\sum_{v_i\in e}\exp(-x_i/\gamma)} + \frac{\sum_{v_i\in e} y_i \exp(y_i/\gamma)}{\sum_{v_i\in e}\exp(y_i/\gamma)} - \frac{\sum_{v_i\in e} y_i \exp(-y_i/\gamma)}{\sum_{v_i\in e}\exp(-y_i/\gamma)}\right).$$

Log-Sum-Exponential (LSE) Function

- An approximation of the maximum function:
  - $\mathrm{LSE}_\alpha(z_1, \ldots, z_n) = \alpha \times \left(\log \left(\sum_{i=1}^{n} e^{z_i/\alpha} \right) \right) \approx \max(z_1, \ldots, z_n)$
  - Strictly convex and continuously differentiable
  - α : smoothing parameter (exact when $\alpha \rightarrow 0$)





Expression of HPWL using LSE Function

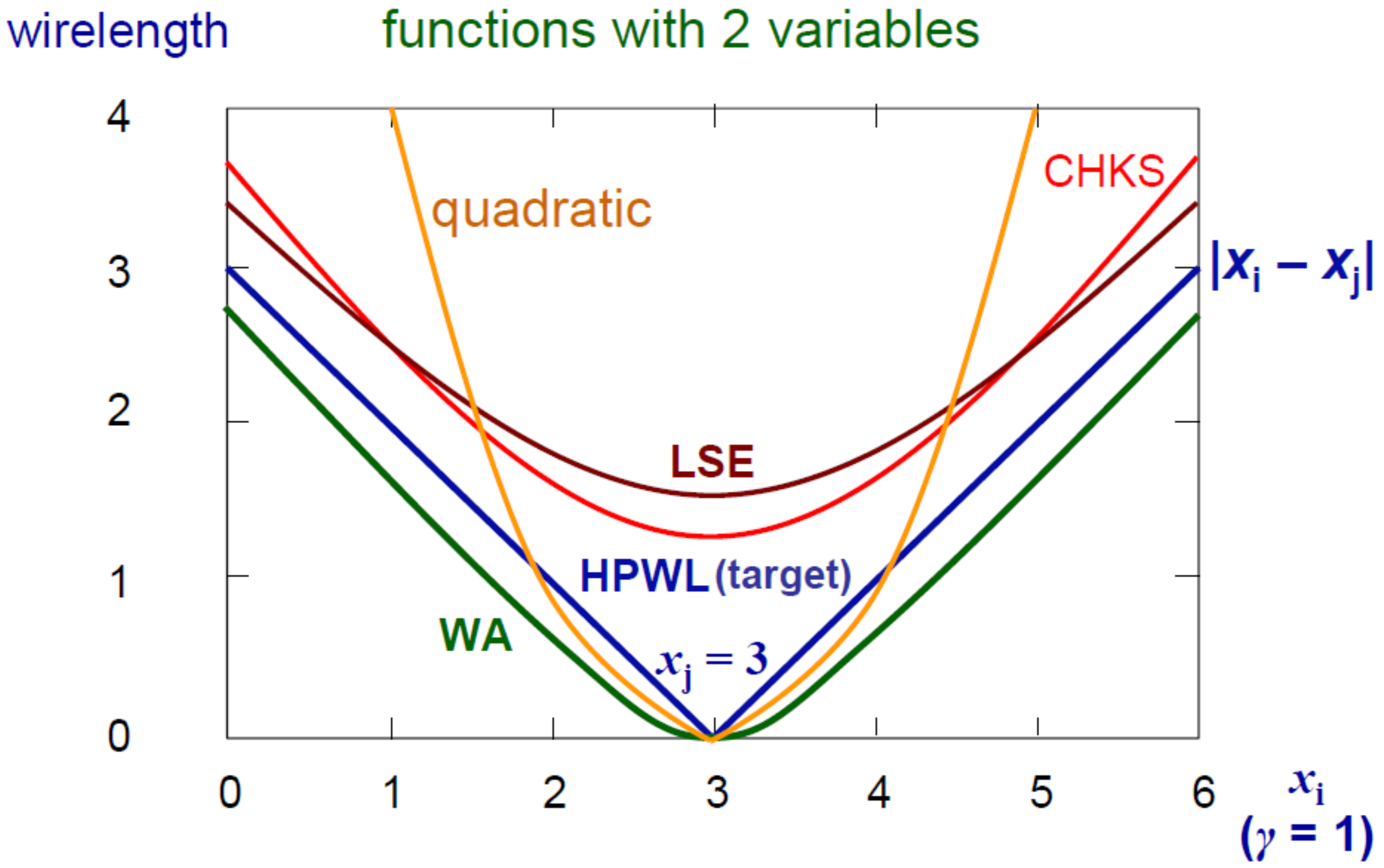
- HPWL in terms of max:

$$\begin{aligned}
&\mathrm{HPWL}_e(x_1, \ldots, x_n,\ y_1, \ldots, y_n) \\
&= \left(\max_{i\in e}\{x_i\} - \min_{i\in e}\{x_i\}\right) + \left(\max_{i\in e}\{y_i\} - \min_{i\in e}\{y_i\}\right) \\
&= \left(\max_{i\in e}\{x_i\} + \max_{i\in e}\{-x_i\}\right) + \left(\max_{i\in e}\{y_i\} + \max_{i\in e}\{-y_i\}\right)
\end{aligned}$$

- After approximating max by LSE:

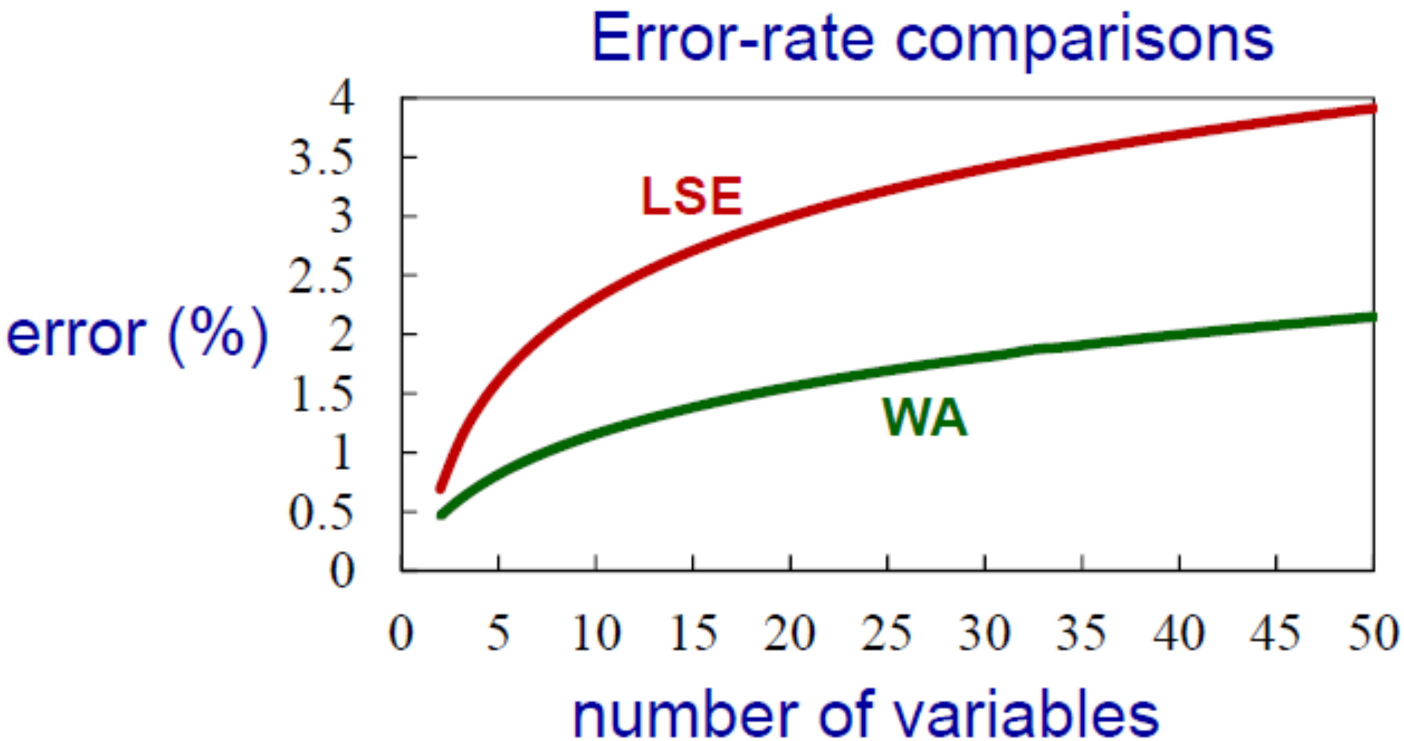
$$\begin{aligned}
&\mathrm{LSEWL}_{e,\alpha}(x_1, \ldots, x_n,\ y_1, \ldots, y_n) \\
&= \alpha \times \left(\log\left(\sum_{i\in e} e^{x_i/\alpha}\right) + \log\left(\sum_{i\in e} e^{-x_i/\alpha}\right)\right. \\
&\qquad \left. + \log\left(\sum_{i\in e} e^{y_i/\alpha}\right) + \log\left(\sum_{i\in e} e^{-y_i/\alpha}\right)\right)
\end{aligned}$$

Wirelength Function Comparisons



Error Rate Comparisons Between WA and LSE Functions

$$\varepsilon_{WA}(\mathbf{x}_e) \le \varepsilon_{LSE}(\mathbf{x}_e) = \gamma \ln n$$



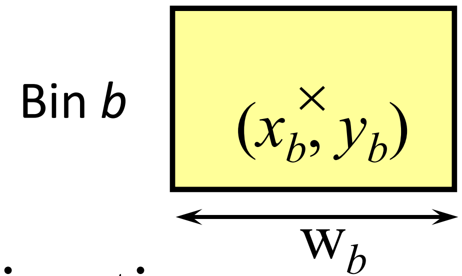
Density Constraint Smoothing

$$\text{Minimize} \sum_{e \in E} c_e \times \mathrm{WL}_e(\boldsymbol{x}, \boldsymbol{y})$$

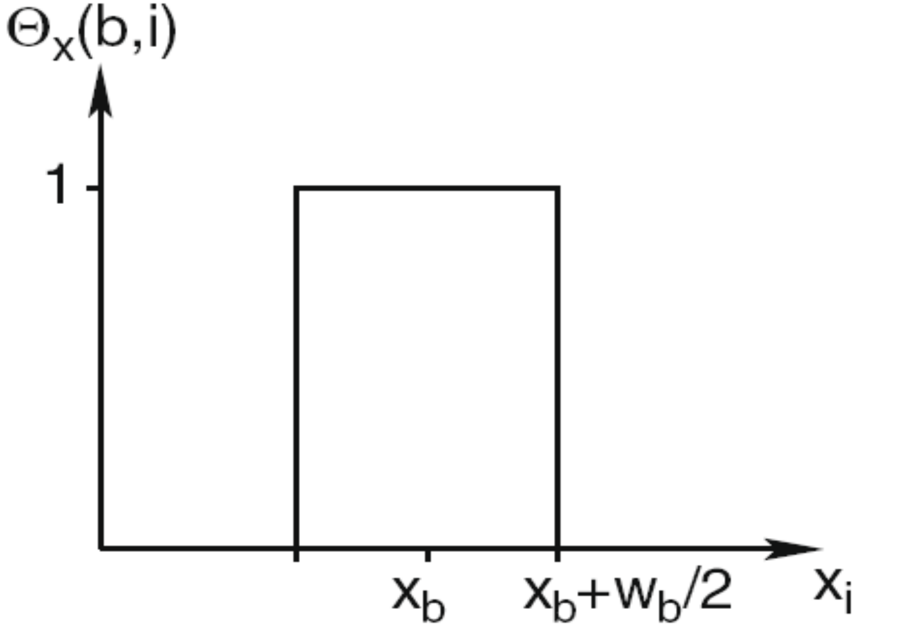
$$\text{Subject to } D_b(\boldsymbol{x}, \boldsymbol{y}) <= T_b \text{ for all bin } b$$

- $D_b(x,y)$ gives the density of bin b with respect to placement solution $\boldsymbol{x}$ and $\boldsymbol{y}$
- Need to be smoothed
- Smoothing techniques
  - Bell-shaped function
  - Sigmoid fucntion

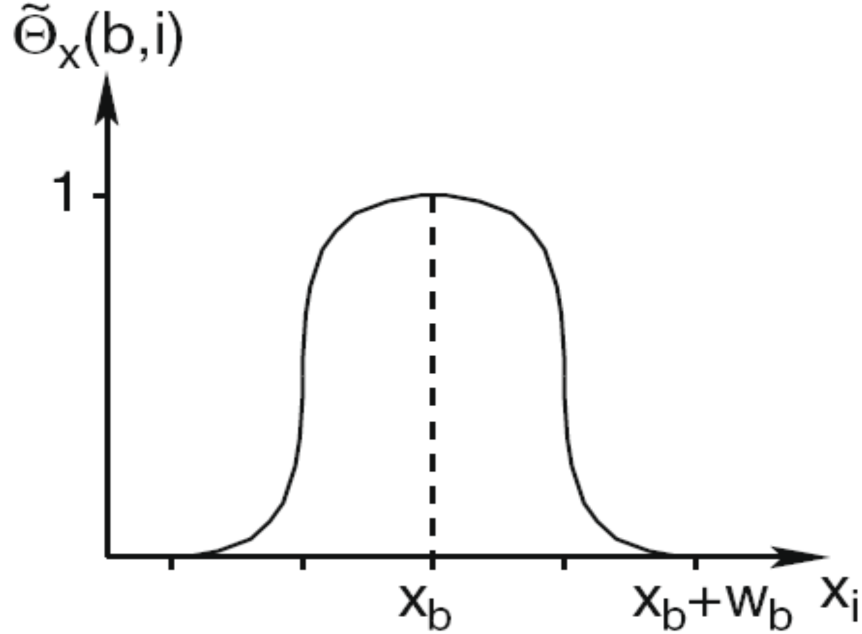
Smoothing by Bell-Shaped Function



- Assume
  - Each module is much smaller than bins
  - Each module has a unit area
- Overlap between bin b and module i in x-direction:



Exact rectangle–shaped function



Smooth bell–shaped function

Equations for Bell-Shaped Function

- Let $d_x = |x_i - x_b|$

$$\tilde{\Theta}_x(b,i) = \begin{cases} 1 - 2 \times d_x^2/\omega_b^2 & \text{if } 0 \leq d_x \leq \omega_b/2 \\ 2 \times (d_x - \omega_b)^2/\omega_b^2 & \text{if } \omega_b/2 \leq d_x \leq \omega_b \\ 0 & \text{if } \omega_b \leq d_x \end{cases}$$

$$D_b(x,\ y) = \sum_{i \in V} C_i \times \tilde{\Theta}_x(b,i) \times \tilde{\Theta}_y(b,i)$$

- Extension for large modules:

$$\tilde{\Theta}_x(b,i) = \begin{cases} 1 - a \times d_x^2 & \text{if } 0 \leq d_x \leq w_b/2 + w_i/2 \\ b \times (d_x - w_b - w_i/2)^2 & \text{if } w_b/2 + w_i/2 \leq d_x \leq w_b + w_i/2 \\ 0 & \text{if } w_b + w_i/2 \leq d_x \end{cases}$$

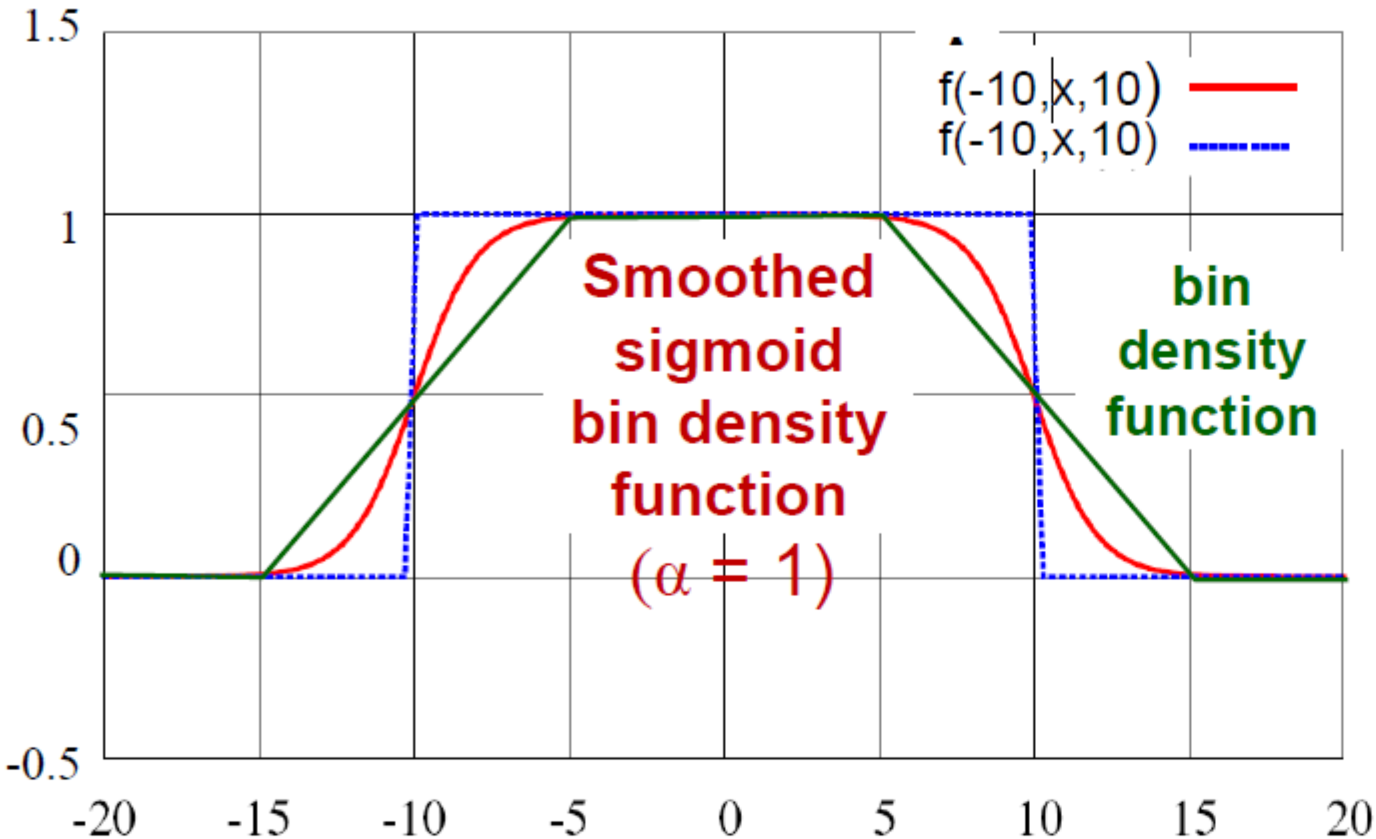
where $a = 4/((w_b + w_i)(2w_b + w_i))$

$b = 4/(w_b(2w_b + w_i))$

Smoothing by Sigmoid Function

[Hsu & Chang, ICCAD'11]

$$f(l,x,u)=\begin{cases}1, & \text{if } l<x<u\\ 0, & \text{otherwise}\end{cases} \xrightarrow{\quad p(t)=\frac{1}{1+e^{-\alpha t}}\quad} f(l,x,u)\cong p(x-l)\,p(u-x)$$



Algorithm for Nonlinear Programs

- **Quadratic penalty method**
  - Convert into a sequence of unconstrained minimization problems
  - Each unconstrained problem can be solved by the **conjugate gradient method**

$$\text{Minimize } \sum_{e \in E} c_e \times \mathrm{WL}_e(\boldsymbol{x},\ \boldsymbol{y}) + \beta \times \sum_{b} (D_b(\boldsymbol{x},\ \boldsymbol{y}) - T_b)^2$$

Extension to Multilevel Placement

- Three phases
  1. A hierarchy of coarser netlists is constructed
  2. An initial placement of the coarsest netlist is generated
  3. The netlist is successively unclustered, and placement at each level is refined

Clustering Technique: First Choice

- Traverse modules in an arbitrary order
- Each module *i* is clustered with an unclustered neighbor *j* with the largest affinity (or weight):

$$r_{ij} = \sum_{e \in E \wedge i,\, j \in e} \frac{c_e}{|e| - 1}$$

- To reduce variation in cluster size:

$$r_{ij} = \sum_{e \in E \wedge i,\, j \in e} \frac{c_e}{(|e| - 1) \times \text{area}(e)}$$

- To consider proximity information in initial placement:

$$r_{ij} = \sum_{e \in E \wedge i,\, j \in e} \frac{c_e}{(|e| - 1) \times \text{area}(e) \times \text{dist}(i,\ j)}$$

Clustering Technique: Best Choice

- Affinity: $r_{ij} = \sum_{e \in E \wedge i, j \in e} \frac{c_e}{|e| \times (\text{area}(i) \times \text{area}(j))}$
- Always select the globally best pair of modules for clustering (and, in principle, update netlist immediately)
- In practice, **lazy updating technique** is proposed
  - Affinities affected by previous clustering are marked as invalid and are updated only after they have been selected for clustering
- Impose a hard upper limit for cluster size