Crosstalk Noise Estimation for use in an Efficient Noise Management Strategy

Paul B. Morton
Wayne Dai

UCSC-CRL-01-09
December 14, 2001

Jack Baskin School of Engineering
University of California, Santa Cruz
Santa Cruz, CA 95064 USA

ABSTRACT

One of the main challenges in developing an effective crosstalk noise management strategy is to develop a crosstalk noise estimate which is both accurate and leads to a tractable optimization problem that can be used to optimally redistribute uncommitted routing resources to resolve crosstalk noise violations. Devgan's [1] estimate comes very close to meeting these objectives, however, it is extremely pessimistic for nets with long couplings or aggressor nets with short rise times. The increased complexity of more sophisticated estimates, such as that proposed by Vittal et. al. [2] lead to a crosstalk management problem that is very hard to solve. In this paper we develop two estimates, similar in form to Devgan's estimate, that are based on local approximations of Vittal's estimate. Our estimates are substantially more accurate than Devgan's estimate while still allowing us to formulate the crosstalk management problem in a form that can be solved efficiently.

Keywords: crosstalk, noise, estimation, local approximation, optimal spacing, noise management

1 Introduction

One of the main challenges in developing and effective crosstalk noise management strategy is to develop a crosstalk noise estimate which is both accurate and leads to a tractable optimization problem that can be used to optimally redistribute uncommitted routing resources [3] to resolve crosstalk noise violations. To date, the vast majority of work that has been done on crosstalk noise management, see for example [4, 5, 6, 7], has been based on a simple capacitive coupling estimate of crosstalk noise. However, coupling capacitance alone leads to an extremely unreliable estimate of crosstalk noise. In particular, a capacitive coupling estimate ignores the multiplicative effects of aggressor net driver rise time, and ignores the moderating effects of interconnect resistance and self capacitance.

Devgan's estimate [1] is a more sophisticated crosstalk noise estimate that comes close to meeting our objectives. It is easily computable, and it incorporates capacitive coupling, victim net interconnect resistance and aggressor net driver rise time. The increased sophistication of Devgan's estimate necessitates a more complex formulation of the crosstalk noise management problem. In particular, it can be formulated as a separable, convex, nonlinear programming problem [8]. However, even with the increased sophistication of Devgan's estimate, it still does not account for self capacitance and aggressor net resistance, making it overly pessimistic for nets with long couplings or aggressor nets with short rise times.

Recently, Vittal et. al. [2] have developed a crosstalk noise estimate that takes all of these variables into consideration while maintaining the same computational complexity as Devgan's estimate. While advances in crosstalk noise estimation can be directly and easily applied to the task of crosstalk noise analysis, they are much more difficult to effectively apply to the task of crosstalk noise management, since changes that do not increase the computational complexity of crosstalk noise estimation and analysis can dramatically increase the complexity of the crosstalk noise management problem. In particular, when Vittal's estimate is applied to the crosstalk management problem, the resulting nonlinear programming problem is no longer separable, and is, in general, nonconvex. It is unlikely that a programming problem of this form can be directly solved in a reasonable amount of time.

In this paper we develop two estimates, each of which combines the best properties of Devgan's and Vittal's estimates. To do this we take advantage of the fact that, for the purpose of the crosstalk noise management problem, the crosstalk noise estimate need only be accurate over a relatively small range of net spacings. This small range allows us to develop two estimates based on local approximation of Vittal's estimate, which are similar in form to Devgan's estimate. Both of our estimates are significantly more accurate than Devgan's estimate, while they still allow us to formulate the crosstalk management problem as separable, convex, nonlinear programs. In addition, we will demonstrate that one of these programs can be solved in $n\,log(n)$ time.

2 Vittal's Noise Estimate

In this paper we represent the routing for a net, v, as a tree where the nodes of the tree for net v form a set, N_v. Each wire segment of a net, v, connects two nodes in $v's$ routing tree. All information describing the wire segment connecting node i to its parent, $Par(i)$, is associated with the node i, that is, all information is stored in the down stream node for a wire segment.

We base our estimate of the crosstalk noise seen at any node of a net on Vittal's crosstalk noise estimate. This estimate takes into account capacitive coupling, driver rise times, net resistance, and self capacitance. Vittal's estimate, for unit amplitude excitation, can be written as

$$V_n = \sum_{x \in N_{net(n)}} \sum_{m \in CP(x,n)} R_m \sum_{y \in Adj(x)} \frac{C_{xy}}{T_{net(y),net(n)}} \tag{1}$$

where: V_n is the estimated crosstalk noise voltage induced on node n by all nets adjacent to $net(n)$; $net(n)$ is the net containing node n; $N_{net(n)}$ is the set of all nodes of $net(n)$; $CP(x,n)$ is the set of nodes, of $net(n)$, that are common to the path from the root to node x and the path from the root node to node n; R_m is the resistance of the wire segment connecting node m to node $Par(m)$; $Adj(x)$ is the set of adjacent nodes capacitively coupled into node x; C_{xy} is the capacitive coupling between nodes x and y. $T_{net(y),net(n)}$ is the "effective rise time" of $net(y)$ as seen by $net(n)$. The effective rise time is defined as

$$T_{net(y),net(n)} = D_{net(n)} + D_{net(y)} + \frac{T_{net(y)}}{2} \tag{2}$$

where $T_{net(y)}$ is the rise time of the driver for $net(y)$, and $D_{net(n)}$ is the "total delay" associated with $net(n)$. The total delay associated with $net(n)$ is defined as

$$D_{net(n)} = \sum_{i \in N_{net(n)}} \sum_{m \in CP(i,i)} R_m \left[C_i^a + C_i^s + \sum_{j \in Adj(i)} C_{ij} \right] \tag{3}$$

where C_i^a is the capacitance associated with the area covered by the wire segment connecting node i to $Par(i)$, and C_i^s is the capacitance associated with any sink pin connected to node i.

3 Problem Formulation

Crosstalk noise violations can be resolved by using uncommitted routing resources to modify wire spacings and/or wire widths. In this paper we will concentrate on the wire spacing approach, and note that a similar development can also be carried out based on modifications to wire widths. To formulate the underlying optimal spacing problem (OSP) for a net, v, with crosstalk noise violations, we write the coupling capacitance between two adjacent nodes, x and y, in terms of the adjacency length and adjacency spacing

$$C_{xy} = C_T \frac{L_{xy}}{S_{xy}} \tag{4}$$

where C_T is a proportionality constant determined by the technology, L_{xy} is the length of the adjacency between nodes x and y, and S_{xy} is the spacing between nodes x and y. Substituting (4) into (1) and (3) we can express the crosstalk noise at any node of a net, v, in terms of a set of spacing variables

$$S_v = \{ S_{xy} \mid x \in N_v,\ y \in Adj(x) \} \tag{5}$$

The routing resources consumed by this spacing for net v can be expressed as

$$A_v = \sum_{x \in N_v} \sum_{y \in Adj(x)} L_{xy} S_{xy} \tag{6}$$

Using (1), (3), (4), (5), and (6) we can formulate the OSP as the following constrained minimization problem:

$$\min_{S_v} (A_v) \tag{7}$$

subject to

$$S_{xy} \geq S_{xy\_min} \qquad \forall\ x \in N_v,\ y \in Adj(x) \tag{8}$$
$$S_{xy} \leq S_{xy\_max} \qquad \forall\ x \in N_v,\ y \in Adj(x) \tag{9}$$
$$V_x \leq M_x \qquad \forall\ x \in Pins(N_v) \tag{10}$$

where (8) and (9) are routing resource constraints, and (10) are the crosstalk noise constraints. Specifically, S_{xy_min} is the spacing between nodes x and y prior to optimization, S_{xy_max} is the maximum spacing allowed between nodes x and y, and M_x is the noise margin for node x.

4 Local Approximation

While (7), (8), and (9) are linear in S_v, (10) is nonlinear, nonseparable, and, in general, nonconvex (see appendix A). Since most of the routing resources on a routed design have already been committed and the remaining uncommitted routing resources are dispersed throughout the design, we expect S_{xy_max} to be a small multiple of S_{xy_min}. Further more, because nonconvex and/or inseparable programming problems are, in general, difficult to solve in a reasonable amount of time, we would like to make a convex, separable, local approximation for (1).

4.1 Taylor Series Estimate

One common approach to making a local approximation is to use several of the low order terms of a Taylor series expansion. Since we are interested in a separable approximation, we are limited to the first two terms of the series. This would give us a linear approximation, however, while the range of spacings is relatively small, it is still too large for this to be an accurate approximation. To improve the accuracy of the approximation we take advantage of the fact that crosstalk noise is primarily a function of capacitive coupling and is therefore a strong function of $1/\overline{S}_v$, where $\overline{S}_v$ is the vector of all spacing variables for net v. With this in mind, we can express (1) as

$$V_n(\overline{S}_v) = g\left(\frac{1}{\overline{S}_v}\right) = g(\overline{Q}) \tag{11}$$

where $\overline{Q} = 1/\overline{S}_v$. Using the first two terms of the Taylor series expansion of $g\left(\overline{Q}\right)$ around $\overline{Q}_1$ we can approximate $g\left(\overline{Q}_2\right)$ as

$$g\left(\overline{Q}_2\right) \approx g\left(\overline{Q}_1\right) + \left(\overline{Q}_2 - \overline{Q}_1\right)^T \nabla_{\overline{Q}}\ g\left(\overline{Q}\right)\Big|_{\overline{Q}=\overline{Q}_1} \tag{12}$$

Substituting for $\overline{Q}$, $\overline{Q}_1$, and $\overline{Q}_2$, we have

$$g\left(\frac{1}{\overline{S}_{v2}}\right) \approx g\left(\frac{1}{\overline{S}_{v1}}\right) + \left(\frac{1}{\overline{S}_{v2}} - \frac{1}{\overline{S}_{v1}}\right)^T \nabla_{\frac{1}{\overline{S}_v}}\ g\left(\frac{1}{\overline{S}_v}\right)\Big|_{\frac{1}{\overline{S}_v}=\frac{1}{\overline{S}_{v1}}} \tag{13}$$

and from (11) we see that we can approximate $V_n\left(\overline{S}_{v2}\right)$ as

$$V_n\left(\overline{S}_{v2}\right) \approx V_n\left(\overline{S}_{v1}\right) + \left(\frac{1}{\overline{S}_{v2}} - \frac{1}{\overline{S}_{v1}}\right)^T \nabla_{\frac{1}{\overline{S}_v}} V_n\left(\overline{S}_v\right)\Big|_{\overline{S}_v=\overline{S}_{v1}} \tag{14}$$

We can now define the local optimal spacing problem (LOSP1) using (7) through (10), where (10) is defined using (14), expanded around $\overline{S}_{v1}$.

While (14) is separable, and similar in form to Devgan's estimate, it is not, in general, convex, since one or more of the gradients could be negative. However, using the simple substitution of variables $\overline{S}_v = 1/\overline{q}$, this problem can be easily converted to a convex programming problem that can be solved using sequential quadratic programming, as described in [8].

From (1) we see that Vittal's estimate can be written such that it has at most $|S_v|$ terms, one for each C_{xy}. Additionally, since each term contains a reference to every C_{xy} through $D_{net(n)}$, the gradient in (14), and thus the Taylor series estimate, have a computational complexity of $O(|S_v|^2)$. From [8] it can be seen that the computational complexity of LOSP1 is roughly cubic in the number of spacing variables, $|S_v|$. Given these computational complexities, we see that this estimation technique would be useful for problems with a moderate number of spacing variables, but would be impractical for use on problems with large numbers of spacing variables.

4.2 MERT Estimate

To develop an estimate for (1) that can be used for problems containing large numbers of spacing variables we again take advantage of the fact that we expect S_{xy_max} to be a small multiple of S_{xy_min}. Because of this we can assume that $D_{net(n)}$ and $D_{net(y)}$ can be reasonably approximated by their minimum values, $D^{min}_{net(n)}$ and $D^{min}_{net(y)}$, over the range $S_{xy\_min} \leq S_{xy} \leq S_{xy\_max}$. Defining the "minimum effective rise time" (MERT) as

$$T^{min}_{net(y),net(n)} = D^{min}_{net(n)} + D^{min}_{net(y)} + \frac{T_{net(y)}}{2} \tag{15}$$

we can define the local approximation as

$$V_n \approx \sum_{x \in N_{net(n)}} \sum_{m \in CP(x,n)} R_m \sum_{y \in Adj(x)} \frac{C_{xy}}{T^{min}_{net(y),net(n)}} \tag{16}$$

This gives us an approximation, identical in form to Devgan's estimate, which is an upper bound on Vittal's estimate over $S_{xy\_min} \leq S_{xy} \leq S_{xy\_max}$. Substituting (4) into (16) we have

$$V_n \approx \sum_{x \in N_{net(n)}} \sum_{m \in CP(x,n)} R_m \sum_{y \in Adj(x)} \frac{C_T L_{xy}}{S_{xy} T^{min}_{net(y),net(n)}} \tag{17}$$

which is both convex and separable in S_v. We can now define the second local optimal spacing problem (LOSP2) using (7) through (10), where (10) is defined using (17).

5 Optimal Spacing Solution

For the nontrivial cases, the LOSP2 forms a convex programming problem with at least one strictly feasible solution. These nontrivial cases can be solved using Lagrange relaxation [9]. Additionally, this approach can easily incorporate timing considerations, as illustrated in [10]. To begin, we associate with each sink pin, p, a Lagrange multiplier, λ_p, which is used to relax that pin's noise constraint into the objective function. This gives us the following Lagrange subproblem (LSP):

$$\min_{S_v} \left(\sum_{x \in N_v} \sum_{y \in Adj(x)} L_{xy} S_{xy} + \sum_{p \in Pins(N_v)} \lambda_p (V_p - M_p) \right) \tag{18}$$

subject to

$$S_{xy} \geq S_{xy\_min} \qquad \forall\ x \in N_v,\ y \in Adj(x) \tag{19}$$

$$S_{xy} \leq S_{xy\_max} \qquad \forall\ n \in N_v,\ y \in Adj(x) \tag{20}$$

From this we define the Lagrange dual problem (LDP) as

$$\max_{\lambda} (LSP) \tag{21}$$

subject to

$$\lambda_p \geq 0 \quad \forall\ p \in Pins(N_v) \tag{22}$$

where $\lambda = \{\lambda_p \mid p \in Pins(N_v)\}$.

Since the LOSP2 is convex, we know that the optimal objective values of the LOSP2 and the LDP are equal, and therefore we can solve the LOSP2 indirectly by solving the LDP.

We solve the LDP using the subgradient optimization technique outlined in Fig 1. This is a generalized ascent algorithm which takes into account the possibility that the search direction computed in line 5 may not be an ascent direction, due to the fact that the LSP may not, in general, be differentiable. As a consequence, the search direction computed in line 5 is based on the subgradient of the LDP. Since the search direction is not necessarily an ascent direction, we cannot rely on a line search to determine the optimal step length, thus, in line 6, we use a predefined sequence of step lengths which will guarantee the convergence of the subgradient optimization algorithm. To enforce the constrains imposed by (22), line 8 projects any infeasible solution back onto the LDP's feasible region.

To determine when the subgradient optimization algorithm has converged, we take advantage of the fact that the optimal values of the LOSP2 and LDP are equal. This implies that the algorithm converges when

$$\sum_{p \in Pins(N_v)} \lambda_p (V_p - M_p) = 0 \tag{23}$$

In practice, however, this criteria is too strict and needs to be loosened to

$$\left| \frac{\sum_{p \in Pins(N_v)} \lambda_p (V_p - M_p)}{\sum_{x \in N_v} \sum_{y \in Adj(x)} L_{xy} S_{xy}} \right| \leq error\ bound \tag{24}$$

1. $\lambda_p \leftarrow 0 \ \forall \ p \in Pins(N_v)$
2. $i \leftarrow 1$
3. Repeat
4. Solve LSP
5. $d_p \leftarrow V_p - M_p \ \forall \ p \in Pins(N_v)$
6. Set θ_i s.t. $\lim_{i\to\infty} \theta_i = 0$ and $\sum_{j\leftarrow 1}^{i} \theta_j \to \infty$
7. $\lambda_p \leftarrow \lambda_p + \theta_i d_p \ \forall \ p \in Pins(N_v)$
8. $\lambda_p \leftarrow max(0, \lambda_p) \ \forall \ p \in Pins(N_v)$
9. $i \leftarrow i + 1$
10. Until convergence

Figure 1: Subgradient optimization algorithm.

However, in loosening the convergence criteria we have to be mindful of the fact that feasible, nonoptimal solutions to the LDP do not necessarily generate feasible solutions to the LOSP2. We overcome this difficulty by adding the following feasibility constraints to the convergence criteria:

$$\frac{V_p - M_p}{M_p} \leq error\ bound \ \ \forall \ p \in Pins(N_v) \tag{25}$$

In order for the subgradient optimization to be useful in practice, we need to be able to efficiently solve the LSP and efficiently compute a new set of spacings and Lagrange multipliers for each iteration of the algorithm. To efficiently solve the LSP, we take advantage of the fact that (18) is separable in S_v and can be written as

$$\min_{S_v} \left(\sum_{x \in N_v} \sum_{y \in Adj(x)} g_{xy}(S_{xy}) \right) \tag{26}$$

where

$$g_{xy}(S_{xy}) = K_{1xy} S_{xy} + \frac{K_{2xy}}{S_{xy}} + K_{3xy} \tag{27}$$

From the Karush-Kuhn-Tucker optimality conditions [9] we find that the LSP can be solved as

$$S_{xy} = \min \left(S_{xy\_max}, \ \max \left(S_{xy\_min}, \ \sqrt{K_{2xy}/K_{1xy}} \right) \right) \tag{28}$$

where

$$K_{1xy} = L_{xy} \tag{29}$$

To determine K_{2xy} we collect all terms of (18) which contain $1/S_{xy}$. This gives us

$$K_{2xy} = \sum_{p \in Pins(N_v)} \lambda_p \sum_{m \in CP(x,p)} \frac{R_m C_T L_{xy}}{T^{min}_{net(y),v}} \tag{30}$$

Rearranging the order of summation we have

$$K_{2xy} = \sum_{m \in Ans(x)} R_m \sum_{p \in DesPins(m)} \frac{\lambda_p C_T L_{xy}}{T^{min}_{net(y),v}} \tag{31}$$

where $Ans(x)$ is the set of nodes which contain x and all of its ancestors. $DesPins(m)$ is the set of m's descendant nodes which contain sink pins and the node m if it contains a sink pin. This can be written as

$$K_{2xy} = \frac{R_x^V C_T L_{xy}}{T_{net(y),v}^{min}} \tag{32}$$

where R_x^V is the "weighted up stream resistance" [11] seen from node x and is computed as

$$R_x^V = \sum_{y \in Ans(x)} \Lambda_y R_y \tag{33}$$

where the "weighting factor", Λ_y, associated with node y is computed as

$$\Lambda_y = \sum_{p \in DesPins(y)} \lambda_p \tag{34}$$

From (34) we see that the set of weighting factors can be computed in one bottom up traversal of the routing tree [12]. From (33) we see that the set of weighted up stream resistances can be computed in a single top down traversal of the routing tree. Further, once the weighted up stream resistance for a node has been computed we can immediately compute the node's spacings using (32), (29), and (28). From these three observations we have the following:

Theorem 1: *The LSP can be solved in $O(|S_v| + |N_v|)$ time.*

From Fig. 1, we see that each iteration of the solution of the LDP must compute a new set of Lagrange multipliers. This can be accomplished by two traversals of the routing tree. One bottom up traversal to compute the down stream capacitance seen by each node, and one top down traversal to compute the noise seen at each sink pin. From these observations, Theorem 1, and the fact that the number of sink pins is bounded above by the number of nodes in the routing tree, we have the following:

Theorem 2: *Each iteration in the solution of the LDP requires $O(|S_v| + |N_v|)$ time.*

6 Delay Estimation

The total delay for a net consists of three components, area capacitance delay, sink pin capacitance delay, and coupling capacitance delay. Of these three, only the coupling capacitance delay is affected by changes in S_v. To compute the coupling component of $D_{net(n)}^{min}$ we use the maximum allowed spacing, S_{xy_max}, for each spacing variable in S_v. To compute the coupling component of each $D_{net(y)}^{min}$ we need to divide the set of spacing variables for each $net(y)$

$$S_{net(y)} = \{S_{ab} \mid \forall\ a \in N_y,\ b \in Adj(a)\} \tag{35}$$

into two sets, the set of spacing variables common to both net v and $net(y)$

$$S_{net(y)}^{v} = S_v \cap S_{net(y)} \tag{36}$$

and the set of spacing variables of $net(y)$ which are not common to net v

$$S_{net(y)}^{\overline{v}} = S_{net(y)} - S_{net(y)}^{v} \tag{37}$$

To compute the coupling component of $D_{net(y)}^{min}$ we use the maximum allowed spacings, S_{xy_max}, for each spacing variable in $S_{net(y)}^{v}$ and for each spacing variable in $S_{net(y)}^{\overline{v}}$ we use the spacing determined by the layout prior to optimization.

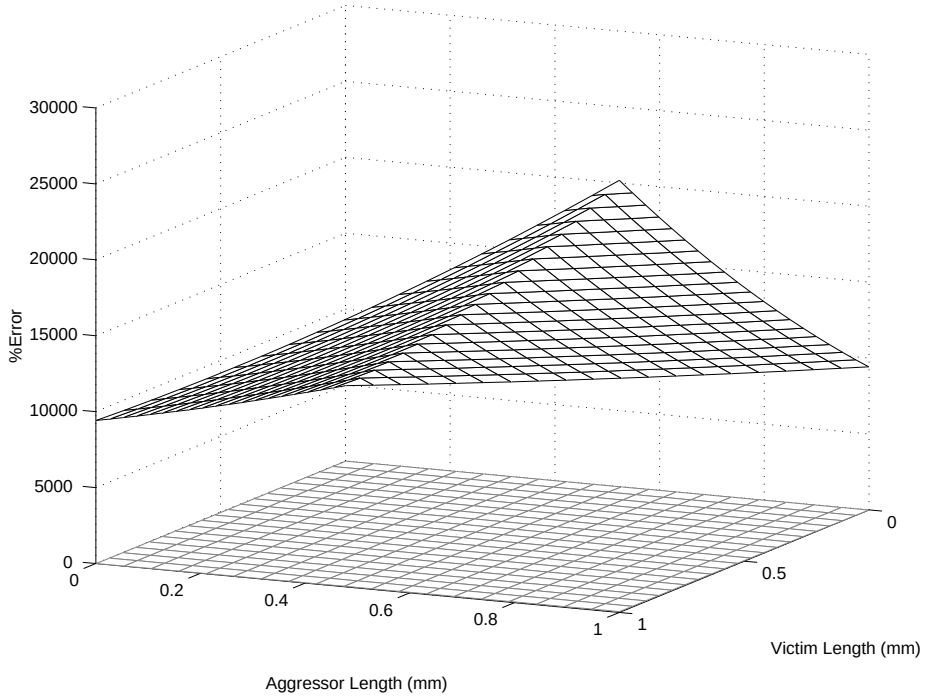

Figure 2: Error comparison for Devgan (upper surface), MERT (middle surface), and Taylor series (obscured lower surface) estimates with an aggressor net driver rise time of $1ps$.

7 Results

To illustrate the improved accuracy of our estimates compared to Devgan's estimate, we computed the worst case errors between our two estimates, as determined by (14) and (17), and Vittal's estimate. We compared these errors to the error between Devgan's estimate and Vittal's estimate. In order to be consistent with the MERT estimate we chose to expand our Taylor series estimate around the maximum allowable spacing for each spacing variable. The results of our comparison are summarized in the plots shown in Fig. 2 through Fig. 4. In each of these figures we have plotted three surfaces, the upper surface represents the error associated with Devgan's estimate, the middle surface represents the error associated with our MERT estimate, and the lower surface represents the error associated with our Taylor series estimate.

The plots in Fig. 2 through Fig. 4 have been generated based on a pair of adjacent single segment nets, one representing the victim net, and one representing the aggressor net. The length of each of these nets ranges from $1\mu m$ to $1mm$. Each net has a source resistance of 500Ω and a load capacitance of $50fF$. Each has a self capacitance of $0.051fF/\mu m$ and a resistivity of $0.25\Omega/\mu m$. The minimum spacing between the nets is $0.33\mu m$ and the maximum spacing is three times the minimum spacing. The coupling capacitance at minimum spacing is $0.116fF/\mu m$. The wire resistance and capacitance have been lumped at every $0.1\mu m$.

In order to capture the worst case errors for our approximations we calculated all results using the minimum spacing between the victim and aggressor nets. In order to illustrate the behaviors of our estimates with respect to aggressor net driver rise time we have plotted the

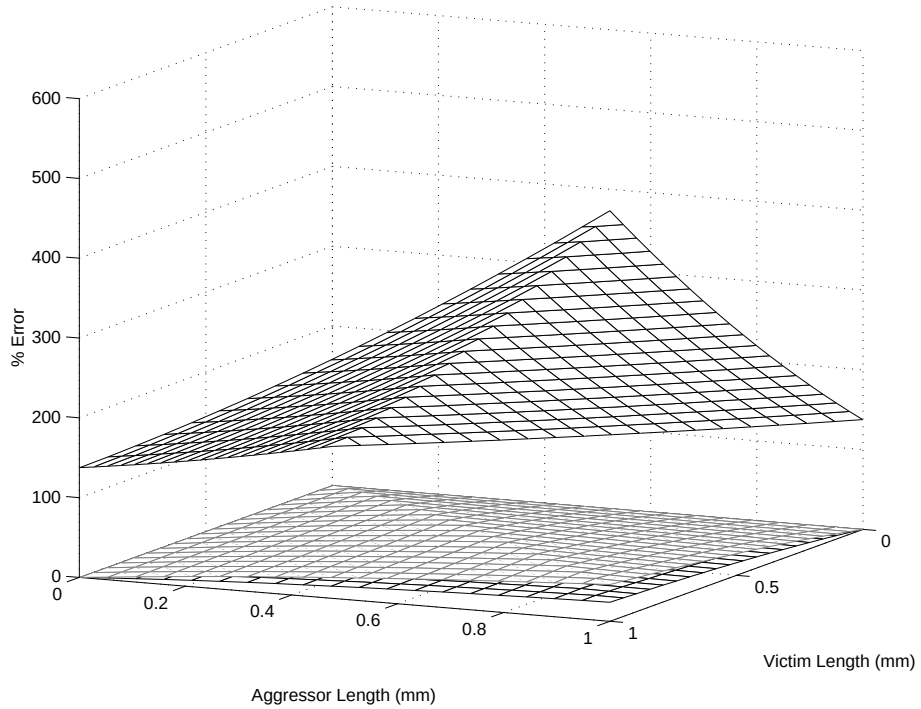

Figure 3: Error comparison for Devgan (upper), MERT (middle), and Taylor series (lower) estimates with an aggressor net driver rise time of $50ps$.

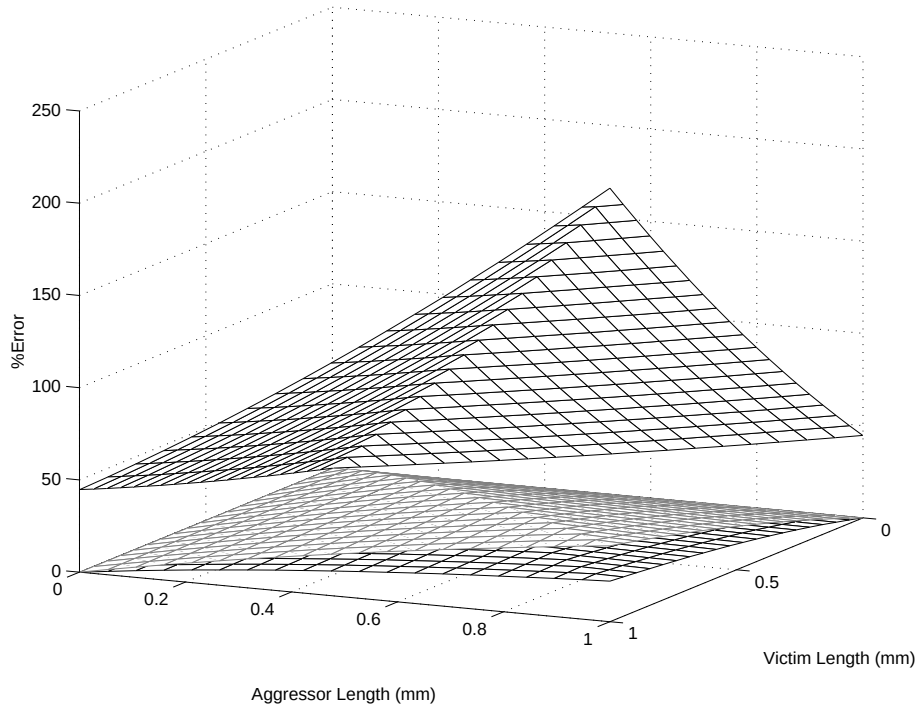

Figure 4: Error comparison for Devgan (upper), MERT (middle), and Taylor series (lower) estimates with an aggressor net driver rise time of $100ps$.

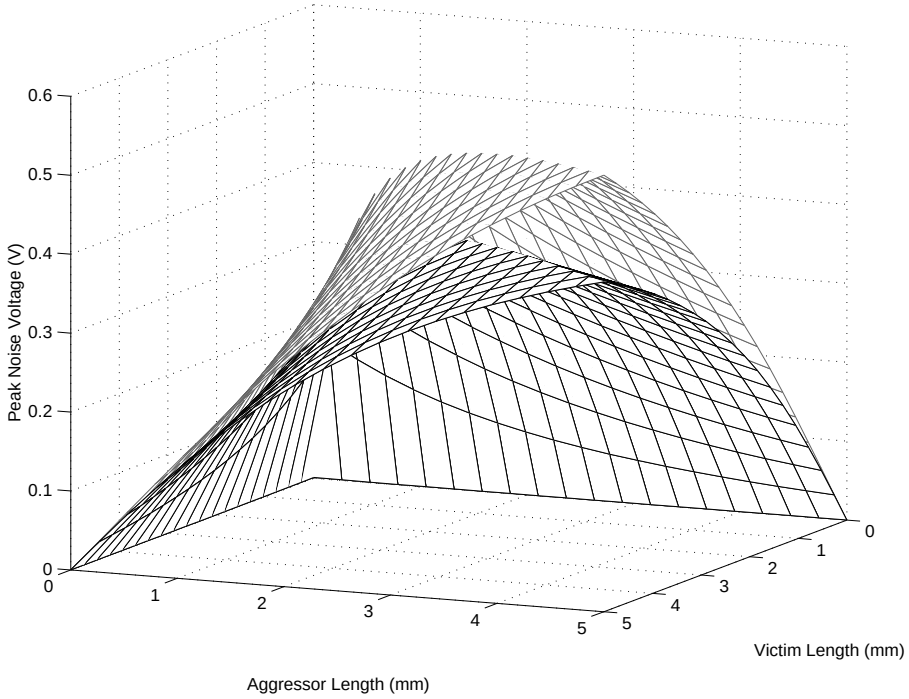

Figure 5: Peak noise predicted by MERT (upper), and Taylor series (lower) estimates with an aggressor net driver rise time of $0s$.

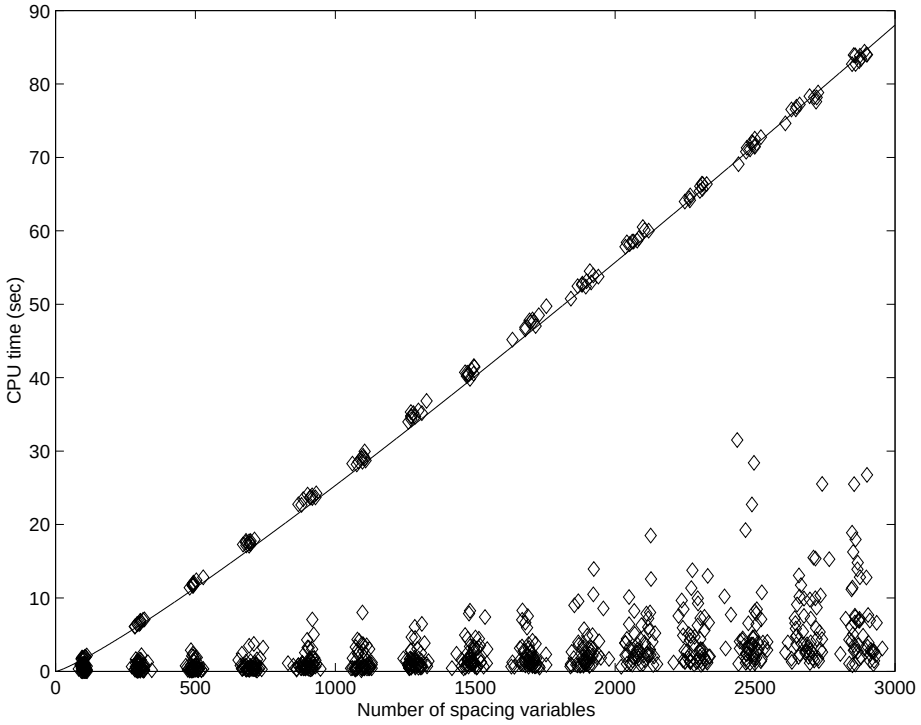

Figure 6: CPU time to compute optimal spacing based on MERT estimate for randomly selected noise margins (lower "cloud"), and for worst case performance (upper curve).

results for three different rise times ranging from $1ps$ to $100ps$. From Fig. 2 through Fig. 4 we can clearly see the dramatic improvements in accuracy obtained by our estimates. The improvements are particularly dramatic for aggressor nets with short rise times, and when there is a long coupling length between the aggressor and victim nets. Additionally, we can see that our Taylor series estimate is also an upper bound on the Vittal estimate, and, as can be seen in Fig. 3 and Fig. 4, produces an estimate that is roughly twice as accurate as the MERT estimate.

In Fig. 5 we have plotted the peak noise voltages predicted by our estimates (using the same values stated above), for an aggressor net driver rise time of $0s$, over a wide range of aggressor and victim net lengths. From this plot we see that, unlike Devgan's estimate, our estimates remain bounded as the rise time goes to zero, and also remain bounded as the coupling length increases.

To evaluate the performance of our optimal spacing algorithm for the MERT estimate we have tested it on a large set of randomly generated multisegment nets. The capacitance and resistance characteristics of these nets are the same as that outlined earlier. Our results have been generated on a Sun Enterprise 450 with a $300\,MHz$ Ultra SPARC-II CPU and $1\,GB$ of memory. The minimum effective rise time of each adjacent net was randomly selected between $20\,ps$ and $500\,ps$. The upper bound for each spacing variable was randomly selected between $0.35\,\mu m$ and $1.0\,\mu m$. The sequence of step lengths used in line 6 of the subgradient optimization algorithm, outlined in Fig. 1, was determined as

$$\theta_i = 1000(0.999)^{(i-1)} \tag{38}$$

For each of the randomly generated nets, we measured the CPU time for our algorithm, coded in C++, to find an optimal set of spacings for the net. We considered a solution to be optimal when the error bounds in (24) and (25) were set to 0.5%.

The results of our experiments are summarized in Fig. 6. In particular, we show two sets of results, one set forming a "cloud" of points close to the x-axis, and one set of points forming a bounding curve above the cloud. The cloud of points were obtained when the noise margin, M_x, for each pin, x, on a randomly generated net, was randomly selected from

$$M_x^{min} + 0.001(M_x^{max} - M_x^{min}) \leq M_x \leq M_x^{max} \tag{39}$$

where M_x^{min} and M_x^{max} are the minimum and maximum feasible noise margins for the pin. From our experiments we found that when the noise margin, M_x, for a pin, x, was less than approximately $M_x^{min} + 0.0005(M_x^{max} - M_x^{min})$ our algorithm occasionally had trouble converging. Note that these cases start to approach the trivial case where $M_x = M_x^{min}$. Because of this, the Theorem of Strong Duality [9] may not apply, and thus we cannot guarantee that our algorithm will converge. However, this difficulty does not pose any problems in practice since these cases can be easily identified and solved by inspection. In particular, if the noise margin for any pin on a net is approximately equal to its minimum feasible noise margin

$$M_x^{min} \leq M_x < M_x^{min} + 0.001(M_x^{max} - M_x^{min}) \tag{40}$$

then each of the nets spacing variables should be set to its maximum value.

From our experiments we also found that as we narrowed the range from which the noise margins were selected, the performance of the algorithm converges to the worst case performance indicated by the points on the bounding curve in Fig. 6. In particular, we

found that the worst case performance occurred when we set the noise margin, for each pin on a randomly generated net, to approximately $M_x^{min} + 0.75(M_x^{max} - M_x^{min})$. From these results we found that the performance of our algorithm can best be described as $n\,log(n)$, as indicated by the line in Fig. 6.

8 Conclusion

In this paper we have presented two crosstalk noise estimates for use in efficiently solving the crosstalk noise management problem. We have presented results that demonstrate that these estimates are significantly more accurate than Devgan's estimate and, in particular, they remain bounded with increasing coupling length and/or decreasing aggressor net rise time. The more accurate of these estimates allows us to formulate the crosstalk noise management problem as a mathematical programming problem that can be solved using sequential quadratic programming, which operates in roughly cubic time. We have shown that the less accurate estimate allows us to formulate the crosstalk noise management problem as a separable, convex, nonlinear programming problem. We have presented a Lagrange relaxation algorithm which can solve this mathematical program and present results that demonstrate that this algorithm operates in $n\,log(n)$ time.

A Convexity Issues

To see that (10) is nonconvex, we start by considering the effective rise time of each term in (1). In particular, we look at $D_{net(n)}$ and $D_{net(y)}$. These can be split into two sets of terms, those containing a spacing variable in S_v and those that don't. Specifically, $D_{net(n)}$ can be written as

$$\begin{aligned} D_{net(n)} &= \sum_{i \in N_{net(n)}} \sum_{m \in CP(i,i)} R_m [C_i^a + C_i^s] \\ &+ \sum_{i \in N_{net(n)}} \sum_{m \in CP(i,i)} \sum_{j \in Adj(i)} R_m C_T \frac{L_{ij}}{S_{ij}} \end{aligned} \tag{41}$$

$D_{net(y)}$ contains a set of spacing variables, $S_{net(y)}$, where $S_{net(y)} \neq S_v$ and $S_{net(y)} \bigcap S_v \neq \emptyset$. By defining

$$Adj(i) = Adj(net(n), i) \cup Adj(\overline{net(n)}, i) \tag{42}$$

where $Adj(net(n), i)$ is the set of adjacent nodes of $net(n)$ that are coupled into node i, and $Adj(\overline{net(n)}, i)$ is the set of adjacent nodes that are not in $net(n)$ that are coupled into node i. $D_{net(y)}$ can now be written as

$$\begin{aligned} D_{net(y)} &= \sum_{i \in N_{net(n)}} \sum_{m \in CP(i,i)} R_m [C_i^a + C_i^s] \\ &+ \sum_{i \in N_{net(n)}} \sum_{m \in CP(i,i)} \sum_{j \in Adj(\overline{net(n)}, i)} R_m C_T \frac{L_{ij}}{S_{ij}} \\ &+ \sum_{i \in N_{net(n)}} \sum_{m \in CP(i,i)} \sum_{j \in Adj(net(n), i)} R_m C_T \frac{L_{ij}}{S_{ij}} \end{aligned} \tag{43}$$

Combining those terms of (41) and (43) that are not associated with S_v into the constants $K_{net(n)}$ and $K_{net(y)}$, respectively, where $K_{net(n)} > 0$ and $K_{net(y)} > 0$, we have

$$\begin{aligned} D_{net(n)} &= K_{net(n)} \\ &+ \sum_{i \in N_{net(n)}} \sum_{m \in CP(i,i)} \sum_{j \in Adj(net(n),i)} R_m C_T \frac{L_{ij}}{S_{ij}} \end{aligned} \tag{44}$$

and

$$\begin{aligned} D_{net(y)} &= K_{net(y)} \\ &+ \sum_{i \in N_{net(y)}} \sum_{m \in CP(i,i)} \sum_{j \in Adj(net(n),i)} R_m C_T \frac{L_{ij}}{S_{ij}} \end{aligned} \tag{45}$$

Defining the constants

$$K^{ijy} = \begin{cases} L_{ij} C_T \left[\sum_{m \in CP(i,i)} R_m + \sum_{m \in CP(j,j)} R_m \right] & j \in N_{net(y)} \\ L_{ij} C_T \sum_{m \in CP(i,i)} R_m & j \notin N_{net(y)} \end{cases} \tag{46}$$

and

$$K_{net(y)}^{net(n)} = K_{net(n)} + K_{net(y)} + \frac{T_{net(y)}}{2} \tag{47}$$

where $K^{ijy} > 0$, and $K_{net(y)}^{net(n)} > 0$ we have

$$T_{net(y),net(n)} = K_{net(y)}^{net(n)} + \sum_{i \in N_{net(n)}} \sum_{j \in adj(i)} \frac{K^{ijy}}{S_{ij}} \tag{48}$$

Defining the constants

$$K_{xyn} = \sum_{m \in CP(x,n)} R_m C_T L_{xy} \tag{49}$$

where $K_{xyn} > 0$, we can now write (1) in terms of a set of positive constants, K^{ijy}, $K_{net(y)}^{net(n)}$, K_{ij}, and a set of spacings, S_v, for $net(n)$

$$V_n = \sum_{x \in N_{net(n)}} \sum_{y \in Adj(x)} \frac{\frac{K_{xyn}}{S_{xy}}}{K_{net(y)}^{net(n)} + \sum_{i \in N_{net(n)}} \sum_{j \in Adj(i)} \frac{K^{ijy}}{S_{ij}}} \tag{50}$$

Assuming, without loss of generality, that we have a net with only two spacing variables, S_1 and S_2, we can write the cross talk noise seen at a node, n, as

$$f(S_1, S_2) = \frac{\frac{K_1}{S_1}}{K_2 + \frac{K_3}{S_1} + \frac{K_4}{S_2}} + \frac{\frac{K_5}{S_2}}{K_6 + \frac{K_7}{S_1} + \frac{K_8}{S_2}} \tag{51}$$

where $K_1, K_2, \cdots K_8$ are positive constants. Rearranging, we have

$$f(S_1, S_2) = \frac{K_1}{S_1(K_2 + \frac{K_4}{S_2}) + K_3} + \frac{S_1 K_5}{S_1(S_2 K_6 + K_8) + S_2 K_7} \tag{52}$$

In order for (52) to be convex over $S_1 > 0$ and $S_2 > 0$, $f(S_1, S_2)|_{S_2=K}$, where $K > 0$, must also be convex over $S_1 > 0$. Looking at the limits of (52) as $S_1 \to 0$ and $S_1 \to \infty$ we find that

$$\lim_{s_1 \to 0^+} f(S_1, S_2)|_{S_2=K} = \frac{K_1}{K_3} \tag{53}$$

and

$$\lim_{S_1 \to \infty} f(S_1, S_2)|_{S_2=K} = \frac{K_5}{K K_6 + K_8} \tag{54}$$

If, however, the values of $K_1, K_2, \cdots K_8$ are such that

$$\frac{K_1}{K_3} < \frac{K_5}{K K_6 + K_8} \tag{55}$$

then it is not possible for $f(S_1, S_2)|_{S_2=K}$ to go from (53) to (54) without forming a concave region.

References

[1] A. Devgan, "Efficient coupled noise estimation for on-chip interconnects," in *Proc. IEEE/ACM Int. Conf. on Computer Aided Design*, pp. 147–151, 1997.

[2] A. Vittal, L. H. Chen, M. Marek-Sedowska, K.-P. Wang, and S. Yang, "Crosstalk in VLSI Interconnections," *IEEE Trans. Computer-Aided Design*, vol. 18, no. 12, pp. 1817–1824, 1999.

[3] P. Morton and W. Dai, "Routing Resource Management for Post-route Optimization," Tech. Rep. UCSC-CRL-02-12, UCSC, 2002.

[4] A. Onozawa, K. Chaudhary, and E. Kuh, "Performance Driven Spacing Algorithms using Attractive and Repulsive Constraints for Submicron LSI's," *IEEE Trans. Computer-Aided Design*, vol. 14, no. 6, pp. 707–719, 1995.

[5] T. Gao and C. Liu, "Minimum Crosstalk Channel Routing," *IEEE Trans. Computer-Aided Design*, vol. 15, no. 5, pp. 465–474, 1996.

[6] P. Saxena and C. Liu, "Crosstalk Minimization using Wire Perturbations," in *Proc. Design Automation Conf.*, pp. 100–103, 1999.

[7] H.-R. Jiang, Y.-W. Chang, and J.-Y. Jou, "Crosstalk-Driven Interconnect Optimization by Simultaneous Gate and Wire Sizing," *IEEE Trans. Computer-Aided Design*, vol. 19, no. 9, pp. 999–1010, 2000.

[8] P. Morton and W. Dai, "An Efficient Sequential Quadratic Programming Formulation of Optimal Wire Spacing for Cross-Talk Noise Avoidance Routing," in *Proc. Int. Symposium on Physical Design*, pp. 22–28, 1999.

[9] M. Bazaraa, H. Sherali, and C. Shetty, *Nonlinear Programming.* New York: John Wiley and Sons, 1993.

[10] P. Morton and W. Dai, "Optimal Spacing for Managing Cross-talk Induced Noise and Delay in Resistive VLSI Interconnections," Tech. Rep. UCSC-CRL-00-06, UCSC, 2000.

[11] C. Chu and D. Wong, "Greedy Wire-Sizing Is Linear Time," *IEEE Trans. Computer-Aided Design*, vol. 18, no. 4, pp. 398–405, 1999.

[12] C. Chen and D. Wong, "A Fast Algorithm for Optimal Wire-Sizing Under Elmore Delay Model," in *Proc. IEEE Int. Symposium on Circuits and Systems*, pp. 412–415, 1996.