Abstract—The power distribution network (PDN) of an integrated circuit (IC) must undergo various checks throughout the design flow, in order to guarantee that the voltage fluctuations are within certain user-specified safety thresholds. Vectorless verification of the PDN is one approach for verification that requires little information about the on-die logic. This verification problem has been studied extensively over the past few years and has been generally solved by first discretizing time using a particular user-defined time-step. We investigate the effect of this time-step on the quality of the solutions produced (both exact and estimates). We also propose an efficient method to specify the time-step in a way to minimize the errors introduced by the voltage drop estimates.

I. INTRODUCTION

Verifying the power grid has become a crucial part of every modern integrated circuits (IC) design. A power grid that is not designed properly may not be able to deliver the required levels of voltage to the underlying logic circuitry. Voltage fluctuations are bound to arise thus slowing down the on-die logic and possibly causing logic hazards.

Voltage fluctuations have two main causes: IR drops that result from the resistivity of the metal lines across the grid, and $LdI/dt$ noise, which are mainly due to the inductance arising from grid-package interconnections. Often, the inductance in the on-die grid is ignored, leading to a purely RC grid where only undershoots can appear. In order to verify the grid, engineers must check that these undershoots remain within certain user-defined safety thresholds.

Today, power grid verification is done by simulation. Essentially, this boils down to assuming a certain set of transient currents drawn from the grid, and then finding the corresponding voltage drops at all the nodes. The process is repeated for different sets of current waveforms until the grid can be declared safe with a certain level of confidence. Unfortunately, this approach proves to be problematic because the number of current traces needed to ensure enough coverage is nearly impossible to simulate in practice, so that users are left with very little confidence in the results. Moreover, simulation based approaches cannot be done early in the design flow, when grid modifications can be most easily integrated.

For that reason, the constraints-based framework for verification has been developed in [1] and subsequent publications.

This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada.

II. BACKGROUND

A. Power Grid Model

Consider an RC model of the power grid which has resistive connections between nodes and a capacitor from every node (to ground). Some of the nodes have ideal current sources to ground representing the currents drawn by the underlying circuitry. Further, some nodes have ideal voltage sources ($V_{dd}$) (to ground) which represent connections to external supply voltage. Assume that the grid has $n$ nodes that are not connected to a voltage source, and let $i_s(t) \geq 0$ be the $n \times 1$ vector of all the current sources connected to the grid. An entry of $i_s(t)$ is set zero if the corresponding node is not connected to a current source.

Let $u(t)$ be the vector of node voltages, relative to ground, and $v(t)$ the vector of voltage drops $v(t) = V_{dd} - u(t)$. Using Nodal Analysis [3], we can obtain the following:

$$Gv(t) + C\ddot{v}(t) = i_s(t) \quad (1)$$

where $C \geq 0$ is an $n \times n$ diagonal non-singular matrix consisting of all node-to-ground capacitances and $G$ is the $n \times n$
conductance matrix, which is known to be an $M$-matrix [4] when the grid is connected and has at least one voltage source or one current source. This implies that $G^{-1}$ exists and is non-negative, $G^{-1} \geq 0$, and all the eigenvalues of $G$ are real and positive.

Using a finite-difference approximation of the derivative (Backward Euler) in (1), we have:

$$A v(t) \approx B v(t - \Delta t) + i_s(t)$$

where $\Delta t$ is the chosen time-step, $B \triangleq \frac{G}{\Delta t}$, and $A \triangleq G + B$. The choice of $\Delta t$ relates only to the spectral properties of the node voltage signals. For now, we assume that it is chosen to provide an accurate enough model (2). The choice of $\Delta t$ will be investigated in depth in sections IV and V. Assuming then that (2) is accurate, it leads to a recurrence relation that captures the evolution of the system over time, so that the voltage drop at any time $t$ is given by:

$$v(t) = A^{-1} B v(t - \Delta t) + A^{-1} i_s(t)$$

(3)

with the key observation that, like $G$, the matrix $A$ is an $M$-matrix, so that $A^{-1} \geq 0$ exists and $A^{-1} B \geq 0$.

B. Current Constraints

Generally, it is not practical to expect the user to specify exact values for the currents drawn from the power grid. Vectorless verification, first introduced in [1], is an approach that verifies that the voltage drops remain within certain thresholds for all possible transient waveforms satisfying a certain set of local and global constraints. Local constraints are upper bounds on individual current sources. They can be expressed as $0 \leq i_s(t) \leq i_L, \forall t \in \mathbb{R}$, where $i_L$ is an $n \times 1$ vector of the maximum currents that the current sources can draw. Global constraints represent the maximum total power dissipation of a group of circuit cells or blocks. Assuming we have a total of $\kappa$ global constraints, they can be expressed in matrix form as $0 \leq U i_s(t) \leq i_G, \forall t \in \mathbb{R}$, where $U$ is a $\kappa \times n$ matrix that consists only of 0s and 1s which indicates (with a 1) which current sources are present in each global constraint, and $i_G$ is an $\kappa \times 1$ vector of the upper bound values. Together, the local and global constraints define a feasible space of currents, which we denote by $\mathcal{F}$, so that $i_s(t) \in \mathcal{F}$ for every $t \in \mathbb{R}$ if and only if it satisfies both types of constraints.

III. PROBLEM DEFINITION

A. Exact Solution

Before we proceed, we will show how the RC grid verification can be solved exactly using a set of linear programs (LP). We start by defining an extremal operator to help express the exact worst-case vector of voltage drops.

**Definition 1.** (emax). Let $f(i) : \mathbb{R}^n \rightarrow \mathbb{R}^n$ be a vector function whose components will be denoted $f_1(i), \ldots, f_n(i)$, and let $A \subseteq \mathbb{R}^n$. We define the operator $\text{max}_{i \in A} [f(i)]$ as one that provides the $n \times 1$ vector $y = \max_{i \in A} [f(i)]$ such that, for every $j \in \{1, 2, \ldots, n\}$, $y_j = \max_{i \in A} [f_j(i)]$.

**Definition 1** allows us to capture the worst case voltage drops in a single vector $v^*(t) \triangleq \max_{i_s(t) \in \mathcal{F}, \forall t \in A} [v(t)]$. As in [5], one can prove that the resulting vector is in fact independent of $t$, and that $\forall t$:

$$v^*(t) = v^* \triangleq \max_{i_s \in \mathcal{F}} \left[ \sum_{q=0}^{\infty} \text{emax}_{i_s \in \mathcal{F}} \left[ (A^{-1} B)^q A^{-1} i_s \right] \right]$$

Unfortunately, (4) is impossible to compute exactly as it requires an infinite number linear programs. The authors of [2] proposed an upper bound, presented below, that is much easier to compute.

B. Upper Bound

Because $(A^{-1} B)^q \geq 0, \forall q$, it can be proven using (4) that:

$$v^* \leq \left( \sum_{q=0}^{\infty} (A^{-1} B)^q \right) \text{emax}_{i_s \in \mathcal{F}} [A^{-1} i_s]$$

This simplifies to [2]:

$$v^* \leq \bar{v} \triangleq G^{-1} A \text{emax}_{i_s \in \mathcal{F}} [A^{-1} i_s]$$

The bound $\bar{v}$ above is relatively easy to compute as it requires a single $\text{emax}$ computation. Its accuracy however must be investigated in depth. Clearly, the choice of $\Delta t$ affects both the exact worst-case voltage drops and the proposed bound. This will be examined in depth in the following sections.

IV. LIMITING BEHAVIOR

In this section, we investigate the behavior of the bound and the exact summation as $\Delta t \rightarrow \infty$ and as $\Delta t \rightarrow 0$. This will give us an intuition about the error introduced by the upper bound at different $\Delta t$ values. Thus, we will denote $\bar{v}$ by $\bar{v}(\Delta t)$ and $v^*$ by $v^*(\Delta t)$ to emphasize their dependence on $\Delta t$.

A. Behavior of the Upper Bound

We start by computing the limit of the bound as $\Delta t \rightarrow 0$ and as $\Delta t \rightarrow \infty$. First, notice that:

$$\lim_{\Delta t \rightarrow 0} A = \lim_{\Delta t \rightarrow \infty} (G + \frac{C}{\Delta t}) = G$$

$$\lim_{\Delta t \rightarrow \infty} \Delta t A = \lim_{\Delta t \rightarrow 0} (\Delta t (G + C)) = C$$

Using (7), one can prove that $\lim_{\Delta t \rightarrow \infty} \bar{v}(\Delta t) = \text{emax}_{i_s \in \mathcal{F}} [ G^{-1} i_s ]$ which does not depend on $C$. On the other hand, using (8), one can also prove that $\lim_{\Delta t \rightarrow 0} \bar{v}(\Delta t) = G^{-1} i_L$. We observe that the upper bound deteriorates as $\Delta t \rightarrow 0$, as it becomes independent of $C$ and of the global constraints. Therefore, even though BE becomes more accurate as $\Delta t \rightarrow 0$, doing so makes the upper bound less accurate. This can be seen in Fig.1 where the upper bound seems to diverge from the exact solution as $\Delta t$ decreases. This shows the need for an algorithm that properly chooses $\Delta t$ as to to minimize the error between the bound and the exact solution.

B. Behavior of the Infinite Sum

Similarly, we investigate the limit of the infinite sum (exact solution) as $\Delta t \rightarrow 0$ and as $\Delta t \rightarrow \infty$. It can be proven that $\lim_{\Delta t \rightarrow \infty} v^*(\Delta t) = \text{emax}_{i_s \in \mathcal{F}} [ G^{-1} i_s ]$. One can observe that both the bound and the exact summation converge to the same value when $\Delta t \rightarrow \infty$. Unfortunately, the accuracy of the BE discretization scheme becomes unacceptable for large $\Delta t$. If $\Delta t$ is sufficiently large, the error between the $v^*$ and the exact physical solution becomes unacceptably high. On the other
hand, if we let $\Delta t \to 0$, computing the summation becomes a complicated task, and unfortunately, we do not have a definitive closed form expression for the limit. Experimentally, the summation seems to converge to a specific value which is significantly smaller than the upper bound (See Fig. 1).

V. CHOICE OF $\Delta t$

We now describe a method for choosing $\Delta t$ based on empirical observations. Recall that the differential equation that governs the dynamics of the power grid can be written, using (1), as follows:

$$\dot{v}(t) = -C^{-1}Gv(t) + C^{-1}i_s(t)$$  \hspace{1cm} (9)

Let $A = C^{-1}G$. The eigenvalues of $A$ can be thought of as the inverses of the “time constants” of the grid ($C^{-1}G = \frac{1}{R}$ for a 1-node grid). Because $G$ is an $M$-matrix and $C$ is a positive diagonal matrix, then $A = C^{-1}G$ has positive real eigenvalues [4]. Let $\lambda_{\text{max}}$ and $\lambda_{\text{min}}$ be the maximum and the minimum eigenvalues of $A$ respectively. Experimental results have shown that choosing $\Delta t = \frac{1}{\lambda_{\text{max}}}$. Let $\lambda_{\text{max}}$ makes $v^*(\Delta t)$ close to $\lim_{\Delta t \to 0} v^*(\Delta t)$, while choosing $\Delta t = \frac{1}{\lambda_{\text{min}}}$ makes $\ddot{v}(\Delta t)$ close to the same value. Therefore, we propose finding the minimum eigenvalue of $A$ and using it in $\ddot{v}(\Delta t)$.

For example, consider Fig.1 which shows how both the upper bound and the exact solution vary with respect to $\log_{10}(\Delta t)$ for a certain node in a small power grid. In this example, $\frac{1}{\lambda_{\text{min}}}$ was found to be $10^{-11.58}$. We can see that for $\log_{10}(\Delta t) = -11.58$, $\ddot{v}(\Delta t)$ and $\lim_{\Delta t \to 0} v^*(\Delta t)$ have nearly the same value whereas for other values of $\Delta t$, the bound becomes unacceptable. We can clearly see that if $\Delta t$ is chosen such that BE is very accurate (closer to $10^{-14}$), the corresponding upper bound becomes highly inaccurate.

The power method [6] is an efficient and widely used method to find the dominant eigenvalue of a matrix, i.e. the eigenvalue with the largest magnitude. Since our goal is to find the smallest eigenvalue of $A$, we propose finding the dominant eigenvalue of $A^{-1}$ first. Let $x_0$ be a non-zero vector in $\mathbb{R}^n$, and let $x_k = (A^{-1})^k x_0$. For sufficiently large powers of $k$, and according to the power method, $x_k$ becomes a good approximation of the dominant eigenvector of $A^{-1}$. To find the dominant eigenvalue, it remains to write $\lambda_d \approx \frac{(A^{-1}x_k)^T x_k}{x_k^T x_k} \equiv \lambda_d^{(k)}$.

where $\lambda_d$ is the actual dominant eigenvalue of $A^{-1}$, and $\lambda_d^{(k)}$ is the approximate dominant eigenvalue computed using the power method, at iteration $k$. Accordingly, $\lambda_{\text{min}} \approx \frac{1}{\lambda_d}$ and the best $\Delta t$ to use in the expression of $\ddot{v}(\Delta t)$ is $\frac{1}{\lambda_{\text{min}}} \approx \lambda_d^{(k)}$.

Finding $(A^{-1})^k x_0$ is done iteratively by solving the system $Gx_{k+1} = Cx_k$ for every $k$. This requires performing an LU factorization of $G$, and a sequence of forward/backward solves for $k \in \{1, 2, \ldots\}$. The Algorithm stops when $\frac{\lambda_d^{(k)} - \lambda_d^{(k-1)}}{\lambda_d^{(k-1)}}$ becomes smaller than some parameter $\delta > 0$. The flow of the Algorithm is summarized below:

1: $LU$-solve $Gx_1 = Cx_0$ for $x_1$
2: Set $\lambda_d^{(1)} = \frac{x_1^T x_0}{x_0^T x_0}$
3: while $\left| \frac{\lambda_d^{(k)} - \lambda_d^{(k-1)}}{\lambda_d^{(k-1)}} \right| \geq \delta$ do
4: $k = k + 1$
5: $LU$-solve $Gx_{k+1} = Cx_k$ for $x_{k+1}^*$
6: Set $\lambda_d^{(k)} = \frac{x_{k+1}^T x_k}{x_k^T x_k}$
7: end while

Notice that the current constraints do not impose any restrictions on the slopes of the current waveforms. In other words, the current waveforms are allowed to change from any value to another in zero time as long as $|\dot{v}(t)|$ remains in the feasible space $F$. However, in some cases, the user may provide slope constraints, from which one can derive a duration $\Delta t_s$ that determines the minimum amount of time required for the current waveforms to go from their minimum to their maximum values. Alternatively, the user may provide a maximum current waveform frequency $f_{\text{max}}$, from which, one can find a similar value $\Delta t_s = 0.5/f_{\text{max}}$ [7]. Either way, we can extend the approach described above to account for such constraints as follows. If $\Delta t_s$ is the time-step found using the algorithm above, and if $\Delta t_s > \Delta t_u$, then we simply use $\Delta t_s$ in the expression of the bound, otherwise we use $\Delta t_u$. We have found that this approach works well. We have also observed that, when $\Delta t_s > \Delta t_u$, the bound goes down, on average, by up to 5 mV depending on the slope constraints or the frequency constraints provided by the user.

VI. EXPERIMENTAL RESULTS

Let $\Delta t_u$ be the time-step selected, using the Algorithm of section V, to compute $\ddot{v}$, and let $\Delta t_c = \frac{1}{\lambda_{\text{max}}}$ be the time-step to be used in the computation of $v^*$, where $\lambda_{\text{max}}$ is the maximum eigenvalue of $A$. As mentioned earlier, choosing $\Delta t = \frac{1}{\lambda_{\text{max}}}$ makes $v^*(\Delta t)$ close to $\lim_{\Delta t \to 0} v^*(\Delta t)$. Also, let $\Delta t_g = 1/\text{ps}$ be the value of the time-step used in prior vectorless work in [8]. Moreover, let $A_u = G + \frac{C}{\Delta t_u}$, $A_g = G + \frac{C}{\Delta t_g}$, and $A_c = G + \frac{C}{\Delta t_c}$.

Finding $\lambda_{\text{max}}$ can be done using the power method; the details are skipped because $\lambda_{\text{max}}$ is only useful to test the accuracy of the upper bound and will not be used in our proposed verification flow.

We implemented both the exact summation algorithm (at $\Delta t_u$) and the upper bound algorithm (at $\Delta t_u$ and $\Delta t_g$) in C++. We approximated the infinite summation at $\Delta t_u$ as follows:

$$v^*(\Delta t_u) \approx \ddot{v}(\Delta t_u) \approx \sum_{k=0}^{\infty} \max_{i_s \in F} \left( A_{e,1}^k \frac{C}{\Delta t_u} A_{e,1}^{-1} i_s \right)$$
TABLE I: Accuracy of the Upper Bound

<table>
<thead>
<tr>
<th>Grid Size</th>
<th>$\Delta t_{u}$ (is)</th>
<th>$\Delta t_{g}$ (ps)</th>
<th>$E_{u,\text{max}}$ (mV)</th>
<th>$E_{g,\text{max}}$ (mV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4,305</td>
<td>26.3</td>
<td>305.3</td>
<td>0.58</td>
<td>5.96</td>
</tr>
<tr>
<td>17,318</td>
<td>23.5</td>
<td>471.6</td>
<td>0.76</td>
<td>5.21</td>
</tr>
<tr>
<td>32,635</td>
<td>23.6</td>
<td>684.1</td>
<td>0.67</td>
<td>4.39</td>
</tr>
<tr>
<td>68,610</td>
<td>20.7</td>
<td>544.8</td>
<td>0.78</td>
<td>3.13</td>
</tr>
<tr>
<td>103,689</td>
<td>20.3</td>
<td>582.3</td>
<td>0.98</td>
<td>3.33</td>
</tr>
</tbody>
</table>

* The error corresponds to a random set of 20000 nodes

TABLE II: Run-time of Computing $\Delta t_{u}$ and $\bar{v}_{u}$

<table>
<thead>
<tr>
<th>Grid Size</th>
<th>Finding $\Delta t_{u}$</th>
<th>Computing $\bar{v}<em>{u} = \bar{v}(\Delta t</em>{u})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>119,431</td>
<td>5.11 sec</td>
<td>13.46 min</td>
</tr>
<tr>
<td>171,594</td>
<td>7.25 sec</td>
<td>21.71 min</td>
</tr>
<tr>
<td>325,014</td>
<td>14.23 sec</td>
<td>1.58 h</td>
</tr>
<tr>
<td>510,622</td>
<td>30.22 sec</td>
<td>3.99 h</td>
</tr>
<tr>
<td>1,006,625</td>
<td>3.01 min</td>
<td>11.60 h</td>
</tr>
</tbody>
</table>

Fig. 2: Error Plots (4305-nodes grid)

for some large enough integer $N$. We also computed the upper bound at $\Delta t_{u}$ as: $\bar{v}_{u} = \bar{v}(\Delta t_{u}) = G^{-1}A_{u}\max_{i\in\mathbb{F}}\left[A_{u}^{-1}i_{s}\right]$, and at $\Delta t_{g}$ as: $\bar{v}_{g} = \bar{v}(\Delta t_{g}) = G^{-1}A_{g}\max_{i\in\mathbb{F}}\left[A_{g}^{-1}i_{s}\right]$. The emax operations are performed using linear programs (LP). The tests were performed on a set of power grids generated based on user specifications including pitch and width per layer, and C4 and current source distribution. The local and global constraints were chosen based on some user-defined power profile of the grids.

Let $E_{u} = |\bar{v}^{*}-\bar{v}_{u}|$ be the error vector introduced by $\bar{v}_{u}$, and $E_{g} = |\bar{v}^{*}-\bar{v}_{g}|$ be the error vector introduced by $\bar{v}_{g}$. Table I shows, for each grid, the choice of $\Delta t$ for the exact solution as well as for the upper bounds. It also shows the maximums $E_{u,\text{max}}$ and $E_{g,\text{max}}$ of $E_{u}$ and $E_{g}$ respectively. As we can see, the maximum error of $\bar{v}_{u}$ is never exceeding 1mV for all the tested grids while the maximum error introduced by $\bar{v}_{g}$ can be as high as 6.96mV.

To further showcase the accuracy of the upper bound $\bar{v}_{u}$, Fig.2 shows a scatter plot (top) of the relative errors versus the maximum worst-case voltage drops on the 4305-nodes grid. It also shows the correlation plot (bottom) between the upper bounds $\bar{v}_{u}$ and $\bar{v}_{g}$ on one hand, and $\bar{v}^{*}$ on other hand, for the same grid. In the scatter plot, the curve corresponding to an error of 1mV is also shown. A point with an error larger than 1mV will lie in the region above the 1mV-curve. In the correlation plot, the diagonal line $(y = x)$ is also shown. One can observe that the data points of $\bar{v}_{u}$ fit well on the $y = x$ line, which means that the true worst-case voltage drops and the upper bound values are almost identical. However, the data points of $\bar{v}_{g}$ are far from the diagonal. Both plots show that the upper bound is very accurate when used with a time-step $\Delta t_{u}$.

Our algorithm is very efficient because it only requires one $LU$-factorization of the matrix $G$ and few forward/backward solves. The second column of Table II shows the run-time of the algorithm on a 3.4GHz Linux machine with 32GB of memory, for another set power grids. We can see that for the largest grid tested (more than 1 million nodes), finding $\Delta t_{u}$ takes around 3 minutes only. Thus, our method for finding $\Delta t$ is practical for large industrial power grids, and can be easily incorporated in the overall vectorless verification flow without affecting the total run-time of the verification process. Computing the upper bound using $\Delta t_{u}$ is also efficient, as shown in the third column of Table II. We can observe that the verification process takes less than 12 hours for a 1 million nodes grid when $\Delta t_{u}$ is used. That includes the run-time for finding the inverse of the matrix $A_{u}$, and solving $n$ linear programs.

VII. CONCLUSION

In this paper, we investigated the accuracy of the upper bound, first derived in [2], on the worst-case voltage drops in an RC power grid. We studied the limiting behavior of both the bound and the exact solution as $\Delta t \to \infty$ and as $\Delta t \to 0$. We also proposed an efficient method for finding a value for $\Delta t$ to be used when computing the upper bound. We have shown that using such a value makes the upper bound very close to the actual worst-case voltage drops vector.

REFERENCES