Novel Power Grid Reduction Method based on L1 Regularization

Ye Wang, Meng Li, Xinyang Yi, Zhao Song[*], Michael Orshansky, and Constantine Caramanis

Department of Electrical and Computer Engineering, [*]Department of Computer Science
The University of Texas at Austin, Austin, TX, USA, 78712
{lhywang,meng_li,yixy,zhaos,orshansky,constantine}@utexas.edu

ABSTRACT

Model order reduction exploiting the spectral properties of the admittance matrix, known as the graph Laplacian, to control the approximation accuracy is a promising new class of approaches to power grid analysis. In this paper we introduce a method that allows a dramatic increase in the resulting graph sparsity and can handle large dense input graphs. The method is based on the observation that the information about the realistic ranges of port currents can be used to significantly improve the resulting graph sparsity. In practice, port currents cannot vary unboundedly and the estimates of peak currents are often available early in the design cycle. However, the existing methods including the sampling-based spectral sparsification approach [11] cannot utilize this information.

We propose a novel framework of graph **Sparsification** by **L**1 regularization on **L**aplacians (SparseLL) to exploit the available range information to achieve a higher degree of sparsity and better approximation quality. By formulating the power grid reduction as a sparsity-inducing optimization problem, we leverage the recent progress in stochastic approximation and develop a stochastic gradient descent algorithm as an efficient solution.

Using established benchmarks for experiments, we demonstrate that SparseLL can achieve an up to 10X edge sparsity improvement compared to the spectral sparsification approach assuming the full range of currents, with an up to 10X accuracy improvement. The running time of our algorithm also scales quite favorably due to the low complexity and fast convergence, which leads us to believe that our algorithm is highly suitable for large-scale dense problems.

1. INTRODUCTION

Design, analysis, and simulation of large integrated circuits with full power delivery networks (PDN) is computationally expensive due to the tremendous number of elements in the power grid. The complexity of the analysis and simulation is determined by the number of nodes and edges in the power grid. The goal of power grid reduction is to produce a sparse approximation of the original grid with far fewer edges, that nevertheless approximately preserves the linear port behavior for DC analysis.

DAC '15, June 07 - 11, 2015, San Francisco, CA, USA
Copyright 2015 ACM 978-1-4503-3520-1/15/06 ...$15.00
http://dx.doi.org/10.1145/2744769.2744877 .

We propose a new method for power grid reduction, focusing on the computational tractability. We demonstrate its effectiveness on commonly used benchmarks. Our approach adapts new tools from sparse reconstruction and stochastic optimization, to provide a convex optimization problem. This convex optimization problem allows us to take advantage of the physical constraints that, while natural, have not been exploited previously. To be more specific, we design an algorithm that guarantees quality of sparse approximation *only for currents within realistic ranges*. This is in contrast to other methods that try to guarantee accuracy over an unbounded space of currents. As a result, the previous methods are overly conservative in terms of degree of sparsity in the reduced graph for the given accuracy level. As our results in Section 4 demonstrate, our algorithm significantly outperforms earlier methods.

The ability to specify the peak value and a realistic range of the current delivered from each port is based on the following. Even when the design is still incomplete, the peak values can be estimated from a system-level design description [5]. Once the netlist of an entire chip is extracted, more accurate estimates of the bounds (i.e., peak current values) can be determined from transistor-level SPICE simulation. This provides us with the ability to specify the range of the currents and limit the possible current values to a much smaller subset rather than the entire space R^n.

Several model order reduction methods have been proposed for power-grid reduction [3, 9, 12]. Moment matching methods have been widely used to handle large circuits due to their efficiency in projecting the original system onto a Krylov subspace [3]. The Time-Constant Equilibration Reduction algorithm [9] eliminates low-degree nodes by connecting their neighbors. This works well for tree-like graphs; however, for graphs of general topology, it may result in a graph much denser than the original one. Multigrid-based methods [12] can reduce the number of nodes and edges simultaneously. However, they do so without proper control of approximation error on the reduced power grid.

Recent advances in spectral graph theory provide novel theoretical tools for such reductions with performance guarantees. Work in [11] provides a spectral sparsification algorithm running in nearly linear time that produces a graph with $O(n\log(n)/\epsilon^2)$ edges – a sparsified graph with ϵ-similar spectral characteristics, where n denotes the number of nodes and m the number of edges. The spectral sparsification approach uses the admittance matrix of the grid, also called the graph Laplacian L (which we use in later sections), to model the linear port relationship of the power grid in terms of currents (i) and voltage (v): $Lv = i$ [10], see Section 3 for more details. This Laplacian matrix representation provides a good notion of approximation: we want to sparsify L while maintaining similar linear port behavior of given v and i. The framework of spectral sparsification builds the sparsifier G' and the corresponding Laplacian $L_{G'}$ from L_G by introducing random edge sampling. An edge (i, j) in the sparsified graph G' is sampled with probability proportional to its effective resistance. Here the effective resistance across nodes i and j is defined to be the voltage difference $v_i - v_j$ when unit current flows from i to j while other nodes are left open-circuit. In particular, this procedure requires computing all pairs of effective resistance for existing edges. Then, the new edge weight is assigned according to its equivalent resistance and sampling probability. One of the key results in [11] shows that with edge-sampling one can obtain a much sparser graph that maintains similar spectral characteristics as the original Laplacian, L, with high probability.

The graph sparsification method has been applied to power grid reduction [14]. While the results, and, therefore the theoretical guarantees from [11] are broadly applicable, their practical applicability is not apparent due to large constants in the run-time scaling estimates. In dense graphs, the random sparsification of the Laplacian L_G requires the computation of the equivalent resistance of all existing edges; this step can be computationally quite taxing. For real power grids with over 100k nodes, [14] observed that even the spectral sparsification approach is not affordable in practice. Techniques, such as geometric partitioning (e.g., the divide-and-conquer heuristic proposed in [14]), have been proposed to combat this specific issue by applying the spectral sparsification approach on the small sub-blocks. The direct application on smaller sub-blocks may not yield useful results. The scaling of the algorithm may not make sense for small and mid-scale problems. For example, for graphs with n ranging from 100 to $1,000$, even with modest error level $\epsilon = 5\%$, sparsification on the order of $n \log n/\epsilon^2$ results in $O(n^2)$ edges, producing a near-complete graph.

Our experimental results also seem to confirm that for dense graphs with number of nodes in the order of 100 or $1,000$, the spectral sparsification approach does not seem to be practically effective (see Section 4 for more). This should not be surprising for several reasons. First, the accuracy of approximation of the spectral sparsification (guaranteed by the theory) is captured by measuring the difference of the total power rather than the individual errors of currents or voltages; as our results show, the total power guarantees do not translate well to the individual voltage and current errors, in real applications. Second, as mentioned above, the results and guarantees suffer from over-conservatism in terms of the degree of sparsity and approximation error, due to the failure to build guarantees only for bounded ranges of currents.

Our work differs in several important ways from the sampling-based spectral sparsification approach. We also begin with the graph Laplacian, L and hope to sparsify the graph through the spectral properties of L. Rather than sparsifying by random sampling, we consider a *sparsity-inducing optimization formulation* that is designed to guarantee the approximation quality *by design* only on the relevant range of currents. In contrast, the existing spectral sparsification approach [11] relies on guarantees that imply approximation on the full range of voltages, and hence result in potentially over-conservative answers. Finally, we give a first-order, iterative algorithm based on stochastic gradient descent, to efficiently solve the resulting convex optimization problem.

A key advantage of our formulation is that it offers the possibility to define, and utilize, prior information on the range of possible current values described above. This paper aims to exploit the fact that guaranteeing a good approximation of the power grid over the entire space instead of the much smaller subset of realistically possible current values leads to a conservative result and sacrifices the sparsity of the generated power grid. It is not clear how the sampling-based spectral graph sparsification could take this important range information into account and thus would not result in a similar sparsity and low error rate.

We develop an efficient power grid reduction method by adapting ideas from sparse reconstruction and harmonic analysis [13] to impose the ℓ^1 regularization on the graph Laplacian to encourage sparsity. We call it graph **Sparsification** by **L**1-regularization on **L**aplacians (**SparseLL**). To the best of our knowledge, though much work has focused on the various structured sparsity models on graphs [4], the idea of using the ℓ^1-penalty on the graph Laplacian to create a graph sparsifier that preserves a linear relation, is novel in the context of power grid reduction. We also show how recent first-order, and hence highly scalable, iterative algorithms for stochastic gradient descent (e.g., [8]) can be leveraged to efficiently solve our regularized, constrained optimization problem. Thus our framework can also be view in the class of methods exploiting the spectral properties of graph Laplacian.

Our simulation experiments on benchmarks show that by utilizing the range information of currents, the proposed framework SparseLL can achieve an up to 10X edge sparsity improvement compared to the spectral sparsification approach, together with an up to 10X accuracy improvement in the practical current range considered. We consider both synthetic and real-world data. We randomly generate near-dense test cases with 100, 500 and 1000 nodes. For a target of 5% error accuracy, without prior current information, the spectral sparsification approach only obtains a very modest edge reduction, with edge density decreases from 25% to 80%. We also consider a large-scale power grid extracted from a real design. Here as well, we observe similarly modest density reduction by using spectral sparsification. We see that by similar level of 75% edge reduction, the spectral sparsification has error of about 5%. In contrast, our approach, described fully in Section 3, can guarantee an average error of less than 0.05%.

We show, moreover, that the running time of our algorithm scales quite favorably as we increase the size of the problems, therefore leading us to believe that our algorithm is highly suitable for large-scale dense problems. With ultra-large scale problems, we believe that the SparseLL can be extended with similar divide-and-conquer and parallelization heuristics as in [14]. Thus in this paper we only focus on the sparsifying single graphs and do not implement the wrapping-up procedures of geometrically partitioning for benchmarks over 100k nodes.

2. BACKGROUND AND OVERVIEW

2.1 Graph Laplacian and Spectral Sparsification

As we mention above, the theory of spectral graph sparsification relies on forming the graph Laplacian matrix, and then sparsifying it by sampling according to a specific distribution that is also derived from resistance properties of the network [10]. We give the basic definitions here, and also outline the spectral sparsification procedure. It is useful to define the Laplacian, because this is also the point of departure for our own algorithm, which we will describe in Section 3.

Any weighted graph $G(V,E,\omega)$ with vertex set V, edge set E, and edge weights ω (which we assume to be non-negative) has an associated $|V|\times|V|$ matrix called the Laplacian, denoted by L_G:

$$L_G(i,j)=\begin{cases}\sum_{k,k\neq i}\omega_{ik}, & \text{if } i=j,\\ -\omega_{ij}, & \text{if } i\neq j \text{ and } \{i,j\}\in E,\\ 0, & \text{otherwise.}\end{cases}$$

Spectral properties of the graph Laplacian L_G include the diagonal dominant symmetry (SDD), zero sum over rows or columns, being positive semidefinite (PSD) and having 0 as the smallest eigenvalue. More importantly, the graph Laplacian can be used to characterize the linear port behavior between currents i and voltages v by $L_Gv=i$. The current vector i denotes the current injected into each node. The Laplacian matrix is also a direct measure of sparsity: the number of non-zeros on the diagonal indicates the number of nodes and the number of non-zeros off the diagonal denotes the number of edges.

The spectral sparsification approach [11] (abbreviated as **Spectral** in later sections) proceeds to the sparsifier $G'(V,\tilde{E},\tilde{\omega})$ by 1) computing effective resistance R_e for all edges $e\in E$ by solving a linear equation $Lv=i$; 2) sampling edges with probability p_e proportional to ω_eR_e; 3) scaling the weights of edges sampled by $\tilde{\omega_e}=\frac{K\omega_e}{Mp_e}$, if edge e is chosen K times in M samples. Given a weighted graph $G(V,E,\omega)$ and for any $0<\epsilon\leq 1$, Spectral can produce a sparsifier G' with $|\tilde{E}|=O(n\log n/\epsilon^2)$ in time $O(|E|\log n/\epsilon^2)$ which have the following property:

$$(1-\epsilon)v^TL_Gv\leq v^TL_{G'}v\leq(1+\epsilon)v^TL_Gv,\ \forall v\in\mathbb{R}^n.$$

Due to the fact that $L_Gv=i$, the metric $v^TL_Gv=v^Ti$ is just the total power of the network.

2.2 Stochastic Optimization

In stead of assuming $v\in\mathbb{R}^n$, we aim to exploit and utilize the prior range information to achieve a higher degree of sparsity. Decision making problems involving given parameter ranges can be formulated as an optimization problem where the objective is the average loss function:

$$\min_{x\in\mathcal{X}}\int_{\Omega_Z}f(x,z)dz,$$

in which z is has a range Ω_Z and $f(x,z)$ denotes the loss function for decision variable x under parameter z. When the distribution of parameters is available, we can formulate a stochastic optimization problem with objective of expected loss function:

$$\min_{x\in\mathcal{X}}\mathbf{E}_{Z\sim D_Z}[f(x,Z)].$$

Here Z can be modeled as a random variable following a distribution D_Z.

By formulating the objective as the average loss function in range-constrained problems, we are assuming equal weights for all point z in the parameter space Ω_z. This is equivalent to the expectation of loss function if we introduce a uniform distribution for parameter Z over the same range Ω_Z. Moreover, the equivalence is natural due to the fact that the uniform distribution in a given range achieves the maximum entropy. If no further information about the parameter is available, the uniform distribution is the best candidate to model the behavior of parameters in the range considered. Hence without loss of generality we are able to introduce a uniform distribution over the given range and treat the range-constrained problem the same as the associated stochastic optimization problem.

Expectation of the loss function for a bounded random variable always exists for common engineering optimization problems. Although it is preferable to obtain the analytical expression of the average or expectation of loss function, direct integration is hard and even intractable for irregular-shaped distributions. Instead of expected risk $\mathbf{E}_Z[f]$ we settle for the empirical risk $\mathbf{E}_n[f]$ as an approximation:

$$\mathbf{E}_n[f]=\frac{1}{n}\sum_{i=1}^{n}f(x,z_i).$$

Here, the z_i's are independent and identically distributed (i.i.d.) samples from the distribution D_Z of Z. By the law of large number, empirical risk will converge to the expected risk with the increase of number of samples n.

Conventional convex optimization algorithms such as gradient descent and Newton's method [2] are inefficient per iteration for stochastic optimization problems. From the perspective of storage complexity, all the samples need to be stored when applying these methods. Each iteration updates the solution x based on the gradient or Hessian of $\mathbf{E}_n[f]$, namely the average of the gradient or Hessian at each sample point. For n large, this can be computationally expensive per iteration.

The stochastic gradient descent (SGD) algorithm overcomes these disadvantages by estimating the gradient based on a single random sample z_t:

$$x_{t+1}=x_t-\gamma_t\nabla_xf(x,z_t).$$

Here, γ_t is the step size at iteration t.

By sampling z_t from the distribution D_Z, $\nabla_xf(x,z_t)$ is an unbiased estimator of the true gradient of $\mathbf{E}_Z[f(x,z)]$. The unbiasedness of the resulting stochastic process $\{x_t,\ t=1,2,\ldots\}$ guarantees the convergence to the optimal solution in expectation with rate $O(1/\sqrt{t})$ for general convex problems and $O(1/t)$ for strongly convex problems [1]. Since the stochastic algorithm does not need to remember all the samples and, at least for many common problems, the procedure of generating samples from a known distribution is inexpensive, the framework of stochastic optimization can be cast as an online optimization problem. The online optimization framework only requires sampling one point from the distribution at a time, computing the gradient and updating the current solution, which saves storage and computational complexity simultaneously.

3. POWER GRID REDUCTION BASED ON L1 REGULARIZATION

We now formally describe the power grid modeling, problem formulation and algorithm of the proposed SparseLL framework which exploits the underlying information of realistic current ranges.

3.1 Power Grid Model by Graph Laplacian

There are three major types of analysis or simulation in designing power grids: DC (steady-state), AC (frequency-domain) and transient (time-varying) analysis. DC analysis only involves solving a linear equation of associated voltages and currents as $L_G v = i$. In this paper, we focus on DC analysis with purely resistive networks but we believe our methods are extendable.

We can model a power grid with n nodes by an undirected weighted graph $G(V, E, \omega)$. V denotes the set of nodes in the circuits. Among all the nodes in V, we need to distinguish two different subsets of nodes: internal nodes N_{int} and ports (external nodes) N_{ext}. Only ports have current and energy exchange with the outside. The total currents injected to the internal nodes are zero. The vertex corresponding to V_{DD} is a special port as it connects to the voltage source to provide all the currents to the entire system. There is an edge e_{ij} in the set of edges E if there is a resistor R_{ij} connecting nodes i and j. The weight ω_{ij} of edge e_{ij} is defined to be the conductance of R_{ij}, namely $\omega_{ij} = 1/R_{ij}$. In the application of power grid reduction, weights are always non-negative as we allow only passive resistors in the network.

The corresponding graph Laplacian L_G can also be constructed based on the undirected graph modeling for power grids. Both mappings from a pure-resistor power grid to an undirected weighted graph and from an undirected weighted graph to its graph Laplacian are bijections. Thus the problem of finding a reduced power grid G' is equivalent to finding the corresponding sparsified $L_{G'}$.

3.2 Modeling of Current Information

Now we give the mathematical modeling of current information in terms of both the range information extracted from simulations and the associated uniform distribution suitable for the proposed stochastic optimization framework.

Notice that there is no current injected into the internal nodes: $i_k = 0$ for $k \in N_{int}$. Let e_k denote the n-dimensional unit current vector from V_{DD} to node k. Then the overall current i can be represented as the linear combination of e_k's of ports by:

$$i = \sum_{k \in N_{ext}} \beta_k e_k,$$

where $\beta = (\beta_k)$ lies in a non-negative bounded range. The peak value allowed for each β_k can be estimated from system-level simulation or determined by transistor-level SPICE simulation mentioned in Section 1.

As we justified in Section 2, in order to solve a range constrained problem, we can assume a uniform distribution D_I over the same range Ω_I and arrive at the associated stochastic optimization formulation. Because of this equivalence, for a range-constrained problem, we still assume that it is a stochastic optimization problem with uniform distribution. This distributional formulation allows us to take advantage of the efficient stochastic optimization framework.

According to the linear port behavior $L_G v = i$, the voltage range Ω_V and its associated uniform distribution D_V can be obtained from the current domain (Ω_I D_I) by:

$$L_G \Omega_V = \Omega_I,$$
$$L_G D_V = D_I.$$

3.3 Optimization Problem Formulation

In contrast to the total power metric used in Spectral, we plan to access the approximation quality of sparsifiers by measuring the difference of currents for given voltage:

$$\|i - i'\|_2 = \|L_G v - L_{G'} v\|_2.$$

Thus the objective function in the stochastic optimization formulation of power grid reduction with the current distribution is defined by:

$$\min_{L_{G'}} \quad \mathbf{E}_{v \sim D_V}[\|(L_G - L_{G'})v\|_2^2].$$

In order for L to be a valid graph Laplacian, we can impose linear constraints on $L_{G'}$ such that

$$\begin{aligned} L_{G'}(i,j) &\le 0, \text{ with } i \neq j, \\ L_{G'} &= L_{G'}^T, \\ \sum_{j=1}^{n} L_{G'}(i,j) &= 0, \forall i \in \{1, 2, \ldots, n\}. \end{aligned}$$

We impose sparsity of $L_{G'}$ using the ℓ^0-norm or cardinality constraint on the off-diagonal elements:

$$\|L_{G'}\|_0 \le m_0.$$

3.4 Convex Relaxation by L1 Regularization

The formulation derived above is still not convex, due to the cardinality constraint, thus NP-hard to solve. In order to solve this problem, we borrow the ideas from Lasso [13] in signal processing and statistics to relax the ℓ^0 norm to be the convex ℓ^1 norm to encourage sparsity. The intuition comes from the fact that the ℓ^1 norm is the tightest convex relaxation of the ℓ^0 norm and works well in other applications such as compressed sensing and sparse recovery. We believe this novel formulation can also work for graph sparsification preserving linear port relation and confirm this intuition with experiments in Section 4.

General ℓ^1 constrained or regularized problems are non-smooth. Fortunately, the off-diagonal values of a physically realizable graph Laplacian are always non-positive. This fact can help us remove the signs of the absolute value function in ℓ^1-norm of Laplacians. According to the 0-sum property along rows and columns of Laplacian matrices, the ℓ^1 norm of off-diagonal elements is the same as the summation of diagonal elements, i.e. $\sum_i L_{G'}(i,i)$.

Combined with the ℓ^1 penalty, we arrive at the following stochastic convex programming problem:

$$\begin{aligned} \min_{L_{G'}} \quad & \frac{1}{2}\mathbf{E}_{v \sim D_V}[\|(L_G - L_{G'})v\|_2^2] + \lambda \sum_{i=1}^{n} L_{G'}(i,i) \\ \text{s.t.} \quad & L_{G'}(i,j) \le 0, \text{ with } i \neq j, \\ & L_{G'} = L_{G'}^T, \\ & \sum_{j=1}^{n} L_{G'}(i,j) = 0, \forall i \in \{1, 2, \ldots, n\}. \end{aligned}$$

As with standard Lasso optimization, the effect of λ is to control the sparsity level. A larger λ gives us a sparser solution. As λ tends to zero, we recover the (unregularized) regression solution. The value of λ is typically chosen by cross validation with testing data.

3.5 The Stochastic Gradient Descent Algorithm for SparseLL

This stochastic optimization problem is indeed a convex, bounded and generalized linear problem due to its convex objective function and linear constraints.

We describe the algorithm based on stochastic gradient descent in full detail in Algorithm 1. Here, ϵ denotes the threshold of solution accuracy. We are not required to check the solution accuracy at each iteration as the stochastic gradient algorithm converges in expectation but may have some fluctuation over iterations. An easy and efficient stopping criterion checking method is to periodically sample new points, compute the empirical risk under the new samples and stop if no further improvements are observed. The step-size γ_t can be chosen by estimating the Lipschitz constant of the gradient, which is roughly proportional to the diameter of range Ω_I.

Algorithm 1 Stochastic Gradient Descent Algorithm for SparseLL

```
1:  procedure SPARSELL($L_G$,$D$,$\lambda$,$\epsilon$,$\gamma_t$)
    initialization
2:      $t = 0$
3:      $L_0 = 0$
4:      while accuracy above $\epsilon$ do
5:          $t = t + 1$;
6:          Sample a current $i$ randomly from $D_I$;
7:          Get the corresponding $v$ by solving $L_G v = i$;
8:          Calculate the gradient $Grad = \nabla_{L_t}(\frac{1}{2}\|(L_G - L_t)v\|_2^2) + \lambda I$
9:          Update graph Laplacian by $L_{t+1} = L_t - \gamma_t Grad$;
10:         Project $L_{t+1}$ in order to be a valid Laplacian;
11:     end while
12: end procedure
```

The computational complexity in each iteration can be decomposed to three components: solving a linear I-V equation, computing the gradient and doing projection. For a graph Laplacian L_G of n vertices and m edges, $L_G v = i$ can be solved in time $O(m\text{polylog}(n)\log(1/\epsilon))$ to obtain an ϵ-approximate solution [6]. Gradient computation with efficient rank-1 matrix operation has complexity $O(n^2)$. Projection to a valid Laplacian can be implemented simply as projecting off-diagonal elements to non-positive space and modifying the diagonal elements to satisfy the zero row/column sum property, which contributes an additional $O(n^2)$ complexity. By taking the $O(1/\sqrt{t})$ convergence rate of stochastic gradient descent, the overall time complexity of SparseLL is $O(\frac{1}{\epsilon^2}n^2\text{polylog}(n))$, which is the same as the $O(m\text{polylog}(n))$ of theoretical work [11] on spectral graph theory under the condition $m = O(n^2)$.

The stochastic gradient is efficient in terms of space complexity as the online algorithm does not require remembering samples at each iteration.

4. EXPERIMENTAL RESULTS

In this section we report on the extensive experiments to demonstrate how the proposed SparseLL framework based on ℓ^1 regularization and stochastic approximation achieves higher degrees of sparsity by exploiting the underlying realistic range of currents.

As described in section 1, Spectral uses the difference of total power to quantify the approximation quality. We define the measure of approximation quality of sparsification in terms of the average current error at the same applied voltage.

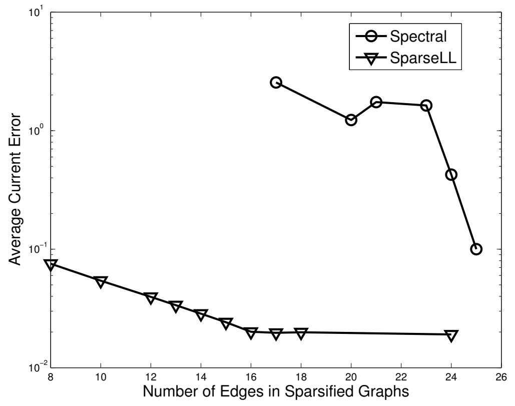

Figure 1: Accuracy with the number of edges.

$$\mathbf{E}_{v\sim D_V}[\frac{\|(L_G - L_{G'})v\|_2}{\|L_G v\|_2}]$$

Notice that the average current error is generally higher than the difference of the total power metric since it can capture the difference in each individual current.

The stochastic gradient descent algorithm in the proposed SparseLL framework has been implemented in C++ for parsing and in MATLAB for numerical computation. We also implement the spectral sparsification approach in MATLAB for comparison. Both our framework and Spectral require calling the standard Laplacian solver to solve $L_G v = i$ $O(n^2)$ times during the procedure of sparsification. Thus we skip the implementation described in [6] and substitute it with MATLAB left division for simple implementation. We focus on the comparison of central parts between SparseLL and Spectral, namely the stochastic gradient descent algorithm and the sampling procedure respectively. All experiments were run on an Intel Xeon 2.93G Linux workstation with 74G memory.

Our experiments use the IBM power benchmarks [7] and the synthetic benchmarks generated by sampling the near-dense graph Laplacians randomly to show the power on dense graphs. Originally used for DC analysis of solving I/V equation $Lv = i$, the IBM power benchmarks specify only the nominal current values i_0's for each port. To enable our experiments, we assume without loss of generality that the peak current at each port k to be $2i_0(k)$, which could cover all possible values of currents. In order to solve by the framework of stochastic optimization, we introduce a uniform distribution for currents in the given range $[0, 2i_0]$ by the principle of maximum entropy.

We first discuss the observed behavior using one of the small IBM benchmarks (we call it ibmpg0). The V_{DD} net of ibmpg0 has 17 nodes with 25 resistors after edge contraction of shorted nodes. In Figure 1, we compare the result of approximation in terms of the average current error by varying the maximum allowed number of edges.

In order to be connected for a graph with 17 nodes, at least 16 edges are needed. Notice that SparseLL produces a drastic improvement in graph sparsity compared to Spectral:

Table 1: Sparsity results with error constraint on large benchmarks.

			Spectral		SparseLL 100%		SparseLL 80%		SparseLL 50%	
Case #	Node #	Edge #	Error	Edge #	Error	Edge #	Error	Edge#	Error	Edge #
rand1	100	4000	5.40%	1031	0.33%	99	0.21%	99	0.08%	99
rand2	500	100000	4.44%	8120	0.07%	499	0.04%	499	0.02%	499
rand3	1000	400000	4.80%	15114	0.03%	999	0.03%	999	0.03%	999
ibmpg1	5388	27000	3.80%	6703	0.03%	4667	0.03%	5379	0.03%	5387

Table 2: Exploiting current range for further sparsity.

Method	Spectral	SparseLL 100%	SparseLL 80%	SparseLL 50%
# Edges	25	16	14	12

Table 3: Runtime comparison.

Benchmark	SparseLL	Spectral
ibmpg0	<1s	<0.01s
rand1	1s	0.02s
rand2	5s	0.5s
rand3	20s	3.1s
ibmpg1	140s	3.2s

it reduces the number of edges from 25 to 12 at the level of 4% accuracy, which is even below the minimum number of edges allowed for a connected graph.

Next we show the possibility of further sparsification through the utilization of additional range shrinkage. The resulting sparsity at the error level of 2% for different ranges is listed in Table 2. For the shrunken ranges, we assume that the currents are centered at i_0 and vary within the specified percentage ranges. As an example, a 80% range means that the current is distributed in the range from $0.2i_0$ to $1.8i_0$. Results in shrunken ranges are consistent with our intuition that smaller ranges result in better sparsification in terms of higher degree of sparsity for the given accuracy level.

To test the behavior of our algorithm on large-scale and dense power grids, we use three randomly generated graph Laplacians in addition to the ibmpg1 benchmark with 5k nodes and 27k edges. Benchmarks rand1-rand3 refer to the randomly generated graphs. The edge weights of each random graph Laplacian are sampled from a uniform distribution to resemble the real power benchmarks.

Table 1 lists the results of optimal edge reduction within the given 5% error level if the only information available is the range of currents. Note that by bringing the practical current range to formulating the corresponding uniform distribution, we could obtain a much sparser result at the same accuracy level. We also find that for a linear network with a single source and multiple sinks, the tree structure is always a good approximator. Notice that as the range of currents is reduced, the benefits of our proposed framework are more pronounced with an up to 5X accuracy improvement at the same degree of sparsity.

In terms of runtime comparison of core parts between SparseLL and Spectral, the results are summarized in Table 3. The runtime result agrees with our discussion of the $O(n^2\text{polylog}(n))$ time complexity. For dense graphs, we can see that the time complexity of both method scales similarly. For the sparse benchmark ibmpg1, the SparseLL approach also could reach a more sparse result with better approximation accuracy given the practical current range in comparable time.

Our framework can be extended from range-constrained problems with a uniform distribution to other distributions if further statistics such as the mean and variance are available.

5. CONCLUSIONS

In this paper, we present an ℓ^1 regularization based approach to seek the reduced power grid by exploiting the underlying realistic ranges of currents. We demonstrate that by introducing a uniform distribution given the realistic range, formulating the problem into a sparsity-inducing convex stochastic optimization problem and solving by the efficient stochastic gradient descent algorithm. Our results demonstrate promise, showing that significant accuracy and edge sparsity improvement are possible in terms of practical range of currents.

6. REFERENCES

[1] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *COMPSTAT*. 2010.

[2] S. Boyd and L. Vandenberghe. *Convex optimization.* 2009.

[3] T.-H. Chen and C. C.-P. Chen. Efficient large-scale power grid analysis based on preconditioned krylov-subspace iterative methods. In *DAC*, 2001.

[4] G. Huang, M. Kaess, and J. J. Leonard. Consistent sparsification for graph optimization. In *ECMR*, 2013.

[5] A. B. Kahng, B. Li, L.-S. Peh, and K. Samadi. Orion 2.0: a fast and accurate noc power and area model for early-stage design space exploration. In *DATE*, 2009.

[6] J. A. Kelner, L. Orecchia, A. Sidford, and Z. A. Zhu. A simple, combinatorial algorithm for solving sdd systems in nearly-linear time. In *STOC*, 2013.

[7] S. R. Nassif. Power grid analysis benchmarks. In *ASP-DAC*, 2008.

[8] S. Shalev-Shwartz, O. Shamir, K. Sridharan, and N. Srebro. Stochastic convex optimization. 2009.

[9] B. N. Sheehan. Ticer: Realizable reduction of extracted rc circuits. In *ICCAD*, 1999.

[10] D. A. Spielman. Algorithms, graph theory, and linear equations in laplacian matrices. In *ICM*, 2010.

[11] D. A. Spielman and N. Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 2011.

[12] H. Su, E. Acar, and S. R. Nassif. Power grid reduction based on algebraic multigrid principles. In *DAC*, 2003.

[13] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. R. Stat. Soc*, 1996.

[14] X. Zhao, Z. Feng, and C. Zhuo. An efficient spectral graph sparsification approach to scalable reduction of large flip-chip power grids. In *ICCAD*, 2014.