Riemannian Optimization Applied to AC Optimal Power Flow

Preprint

Jonathan Maack,[1] Devon Sigler,[1] and Ariel Goodwin[2]

1 National Renewable Energy Laboratory
2 Cornell University

Presented at the 2024 IEEE Power & Energy Society General Meeting
Seattle, Washington
July 21-25, 2024

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Contract No. DE-AC36-08GO28308

Conference Paper
NREL/CP-2C00-88090
November 2024

Riemannian Optimization Applied to AC Optimal Power Flow

Preprint

Jonathan Maack,[1] Devon Sigler,[1] and Ariel Goodwin[2]

1 National Renewable Energy Laboratory
2 Cornell University

Suggested Citation
Maack, Jonathan, Devon Sigler, and Ariel Goodwin. 2024. *Riemannian Optimization Applied to AC Optimal Power Flow: Preprint*. Golden, CO: National Renewable Energy Laboratory. NREL/CP-2C00-88090. https://www.nrel.gov/docs/fy25osti/88090.pdf.

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Contract No. DE-AC36-08GO28308

Conference Paper
NREL/CP-2C00-88090
November 2024

National Renewable Energy Laboratory
15013 Denver West Parkway
Golden, CO 80401
303-275-3000 • www.nrel.gov

Riemannian Optimization Applied to AC Optimal Power Flow

1st Jonathan Maack
Computational Science Center
National Renewable Energy Laboratory
Golden, USA
jonathan.maack@nrel.gov

2nd Devon Sigler
Computational Science Center
National Renewable Energy Laboratory
Golden, USA
devon.sigler@nrel.gov

3rd Ariel Goodwin
Center for Applied Mathematics
Cornell University
Ithaca, USA
awg77@cornell.edu

***Abstract*—The nonlinear, nonconvex AC optimal power flow problem is of growing importance as the nature of the power grid evolves. This problem can be difficult to solve for interior point methods. However, the advent of optimization algorithms over smooth Riemannian manifolds presents an alternative approach. The nonlinear, nonconvex constraints in the AC power flow problem form an embedded submanifold of Euclidean space. In this paper, the authors explore the performance of Riemannian optimization algorithms for the ACOPF problem where the optimization is performed directly on the AC power flow manifold. They demonstrate that these are viable computational alternatives to interior point methods. This is done by using Julia and the packages PowerModels.jl and Manopt.jl.**

***Index Terms*—Optimal Power Flow, Nonlinear Programming, Manifold Optimization, Numerical Optimization**

I. Introduction

With the aging transmission infrastructure and turn towards renewable power generation, there is a need for higher fidelity models of power systems. In particular, significant interest has arisen in using full alternating current (AC) physics in power systems design, analysis and operation [1]. At the core of many operational optimization problems is the optimal power flow (OPF) problem. However, using AC power flow equations in an OPF problem results in a nonlinear, nonconvex optimization problem. Such problems are normally solved with an interior point algorithm such as that implemented in Interior Point OPTimizer (IPOPT) [2]. The ACOPF problem can be difficult for these solvers [3], particularly as the size of the power system increases. This is likely due to the challenging linear algebra problem (known as a saddle point problem) required in interior point methods [4], [5].

However, the recent development of optimization on smooth manifolds [6] presents an alternative approach. These methods exploit differential geometry, which generalizes calculus to non-Euclidean settings [7], and Riemannian geometry, which extends the Euclidean notions of angles and distances to smooth manifolds [8], to solve optimization problems on manifolds. These approaches are applicable since the AC power flow constraints form an embedded submanifold of Euclidean space [9], which we call the power flow manifold. In this paper, we contribute to the literature by demonstrating that Riemannian optimization applied to ACOPF is a viable computational alternative to interior point methods through a set of computational experiments. Specifically, we present implementations and results that show some of these algorithms can produce similar quality solutions as IPOPT, where solution quality is measured by the terminal objective value and maximum constraint violation. To show this, we use the Julia programing language [10] and the packages PowerModels.jl [11] and Manopt.jl [12].

The remainder of this paper is organized as follows: in section II, we briefly describe the needed Riemannian geometry. Then, in section III, we generally describe Riemannian optimization algorithms while also defining the components not found in standard optimization algorithms. In section IV, we describe the algorithms and computational tests we performed before giving our results in section V. Finally, we give our conclusions and discuss future work in section VI.

II. Riemannian Manifolds

For the purposes of this paper, we consider a smooth manifold $\mathcal{M}$ to be given by

$$\mathcal{M} = \{x \in \mathbb{R}^n : q(x) = 0\}, \tag{1}$$

where $q : \mathbb{R}^n \to \mathbb{R}^m$ is infinitely differentiable, $m < n$ and Dq_x, the Jacobian of q evaluated at x, is of full-rank for all $x \in \mathbb{R}^n$. This case is known as an embedded submanifold of Euclidean space.

Given a point $x \in \mathcal{M}$, consider any smooth curve $\gamma : I \to \mathcal{M}$ where $I \subset \mathbb{R}$ contains zero and $\gamma(0) = x$ (see [7] for the definition of a smooth curve). The tangent space $T_x\mathcal{M}$ is defined by

$$T_x\mathcal{M} := \{v \in \mathbb{R}^n : v = \gamma'(0)\}. \tag{2}$$

This work was authored in part by the National Renewable Energy Laboratory (NREL), operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. This work was supported by the Laboratory Directed Research and Development (LDRD) Program at NREL. This research was performed using computational resources sponsored by the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy and located at the National Renewable Energy Laboratory. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

This can be thought of as the set of all possible velocity vectors at the point x for any path on the manifold. For a submanifold of Euclidean space, the tangent space at x coincides with the kernel of the Jacobian:

$$T_x\mathcal{M} = \{v \in \mathbb{R}^n : Dq_x v = 0\}. \tag{3}$$

That is, the tangent space $T_x\mathcal{M}$ is the tangent (hyper-)plane of the manifold at the point x.

The tangent bundle is the disjoint union of all tangent spaces

$$T\mathcal{M} = \{(x, v) : x \in \mathcal{M}, v \in T_x\mathcal{M}\}. \tag{4}$$

A smooth manifold $\mathcal{M}$ becomes a Riemannian manifold when paired with a Riemannian metric $g_x : T_x\mathcal{M} \times T_x\mathcal{M} \to \mathbb{R}$. The Riemannian metric generalizes the notion of the Euclidean inner product so that geometric notions such as angles can be given on general smooth manifolds. In our case, we take g_x to be the standard Euclidean inner product

$$g_x(u, v) = \langle u, v\rangle_x := \sum_{i=1}^{n} u_i v_i \tag{5}$$

where $u, v \in T_x\mathcal{M}$. The subscript x here is used to emphasize the fact that the vectors are restricted to the tangent space of the manifold at the point x.

For a function $f : \mathcal{M} \to \mathbb{R}$, the Riemannian gradient of f is defined as the unique vector field grad$f$ on $\mathcal{M}$ such that for all $(x, v) \in T\mathcal{M}$, we have

$$Df(x)[v] = \langle \text{grad} f(x), v\rangle_x \tag{6}$$

where Df is the differential of f [7]. For a submanifold of Euclidean space, it can be shown that

$$\text{grad} f(x) = P_x(\nabla \hat{f}(x)) \tag{7}$$

where $\hat{f}$ is any smooth extension of f to $\mathbb{R}^n$, ∇ denotes the standard Euclidean gradient and $P_x : \mathbb{R}^n \to T_x\mathcal{M}$ is the orthogonal projection and is given by the matrix

$$P_x = I - Q_x, \quad Q_x = Dq_x(Dq_x Dq_x^T)^{-1} Dq_x^T. \tag{8}$$

The definitions given here can be generalized to abstract manifolds which are not simply a subset of Euclidean space. For the full mathematical details, see [7] and [8]. For a more optimization oriented explanation, see [6].

III. Riemannian Optimization

In this section, we briefly describe the additional needed concepts to solve the optimization problem

$$\min_{x \in \mathcal{M}} f(x) \quad \text{s.t.} \quad h(x) \le 0. \tag{9}$$

We limit ourselves to those concepts needed for algorithms that use only first-order information (that is, gradients and Jacobians) since we apply only those methods in this paper.

Before discussing a general framework for Riemannian optimization, we need to introduce two things: retraction and vector transport. (The following definitions are taken from [6].)

A retraction is a smooth map $R : T\mathcal{M} \to \mathcal{M} : (x, v) \to R_x(v)$ such that for each curve $\gamma(t) = R_x(tv)$ we have $\gamma(0) = x$ and $\gamma'(0) = v$.

From an optimization perspective, the main use of a retraction is to ensure that new iterates are on the manifold. To see this, consider an iterate $x_k \in \mathcal{M}$ and search direction $s_k \in T_{x_k}\mathcal{M}$. In the Euclidean setting, we generally set $x_{k+1} = x_k + \alpha_k s_k$ for some $\alpha_k \in \mathbb{R}$. Such an iterate is almost surely not on the manifold. However, using the retraction, we can set

$$x_{k+1} = R_{x_k}(\alpha_k s_k) \tag{10}$$

so that we have guaranteed $x_{k+1} \in \mathcal{M}$. The notion of a retraction is a generalization of the exponential map [6], [8].

A vector transport on $\mathcal{M}$ is a smooth linear map $\mathcal{T} : T\mathcal{M} \bigoplus T\mathcal{M} \to T\mathcal{M} : (u, v) \to \mathcal{T}_u(v)$ such that, for all $x \in \mathcal{M}$ and for all $u, v \in T_x\mathcal{M}$, there exists a retraction R where $\mathcal{T}_u(v) \in T_{R_x(u)}\mathcal{M}$ and $\mathcal{T}_0(v) = v$. The notion of a vector transport is a generalization of parallel transport [8].

In an optimization algorithm, vector transport allows us to move vectors from one tangent space to another tangent space. For example, many optimization tasks require computing the difference $\nabla f(x_{k+1}) - \nabla f(x_k)$. On a manifold, this difference is not defined since grad$f(x_{k+1}) \in T_{x_{k+1}}\mathcal{M}$ and grad$f(x_k) \in T_{x_k}\mathcal{M}$. Instead, we use the vector transport and compute the difference as

$$\text{grad} f(x_{k+1}) - \mathcal{T}_{\alpha_k s_k}(\text{grad} f(x_k)) \tag{11}$$

where x_{k+1} satisfies (10).

A generic (first-order) Riemannian optimization algorithm iterates a three-step procedure:

1) Determine search direction s_k. This often uses a vector transport $\mathcal{T}_u(v)$. For example, Riemannian conjugate gradient takes
$$s_k = -\text{grad} f(x_k) + \beta_k \mathcal{T}_{\alpha_{k-1} s_{k-1}}(s_{k-1}) \tag{12}$$
for some $\beta_k \in \mathbb{R}$.
2) Determine step size α_k. Since $\phi(\alpha) = f(R_{x_k}(\alpha s_k))$ is a function from $\mathbb{R}$ to $\mathbb{R}$, we can directly use Euclidean line search techniques to determine α_k.
3) Set $x_{k+1} = R_{x_k}(\alpha_k s_k)$. As previously stated, the retraction $R_x(u)$ guarantees that the new iterate is on the manifold.

IV. Test Setup

A. Retraction

The retraction we chose is taken from [13]. The basic idea is to search for the manifold using the normal vector to the tangent space. This is often called the orthographic retraction in the literature.

We now outline the procedure. For a point x_k on the manifold, a search direction s_k and step size α_k, we set

$$y_k^0 = x_k + \alpha_k s_k. \tag{13}$$

and perform the iteration

$$y_k^{\ell+1} = y_k^{\ell} - Dq_{x_k}^T (Dq_{x_k} Dq_{x_k}^T)^{-1} q(y_k^{\ell}) \tag{14}$$

until $||q(y_k^\ell)||$ is sufficiently small at which point we set

$$x_{k+1} = y_k^\ell. \tag{15}$$

It is clear that for $\alpha_k = 0$, we recover x_k immediately. Further, defining

$$L = Dq_{x_k}^T (Dq_{x_k} Dq_{x_k}^T)^{-1}$$

and differentiating (14) gives

$$\frac{dy_k^{\ell+1}}{d\alpha_k} = (I - LDq_{y_k^\ell}) \frac{dy_k^\ell}{d\alpha_k}. \tag{16}$$

By induction, we have

$$\frac{dy_k^{\ell+1}}{d\alpha_k} = (I - LDq_{y_k^\ell}) \dots (I - LDq_{y_k^0}) s_k. \tag{17}$$

Upon setting $\alpha_k = 0$, this reduces to

$$\frac{dy_k^{\ell+1}}{d\alpha_k} = P_{x_k}^{\ell+1} s_k = s_k \tag{18}$$

where we have used the fact that $s_k \in T_{x_k}\mathcal{M}$ and the fact that $P_{x_k}^\ell = P_{x_k}$. It follows that this is a retraction.

B. Vector Transport

For the embedded submanifold case we are considering, there is an obvious vector transport. This is given by simply taking the orthogonal projection of a vector into the necessary tangent space. In particular, for any retraction R, we define

$$\mathcal{T}_u(v) = P_{R_x(u)} v \tag{19}$$

where $x \in \mathcal{M}$, $u, v \in T_x\mathcal{M}$ and $P_{R_x(u)}$ is given by (8).

Clearly, this satisfies the first condition to be a vector transport. Then, since $R_x(0) = x$, we have

$$\mathcal{T}_0(v) = P_{R_x(0)} v = P_x v = v \tag{20}$$

where we have used the fact that $v \in T_x\mathcal{M}$. It follows that this is a vector transport.

C. Direction and Line Search Algorithms

To handle the inequality constraints, we use the two methods from [14]. These are Riemannian augmented Lagrangian (RAL) method and Riemannian exact penalty (REP) method. These methods both require the solution of unconstrained subproblems. For these we test out several different algorithms: Riemannian gradient descent (RGD), Riemannian conjugate gradient descent (RCG) and Riemannian quasi-Newton method (RQN). For all cases we used the default line search algorithm, which in the case of RGD and RCG was the Riemannian Armijo condition (called ArmijoLinesearch in Manopt), and in the case of RQN was the Riemannian Wolfe-Powell conditions (called WolfePowellLinesearch in Manopt). We implemented the retraction and vector transport algorithms as described in Section IV. Otherwise, we used the algorithms as implemented in Manopt.jl version v0.4.41 [12].

The Manopt documentation provides a thorough discussion and set of references to their implemented algorithms. As a result, we only briefly discuss some references on the algorithms we selected. The generalization of gradient descent to the Riemannian case is fairly straight forward. It is discussed in [6]. Note that for embedded submanifolds given by (1), RGD reduces to projected gradient descent [13].

There are numerous coefficient updates for the conjugate gradient descent algorithm many of which have been generalized to the Riemannian setting. We used the default coefficient first proposed by Fletcher in [15]. This is generalized to Riemannian manifolds in a straight-forward manner [16].

The RQN algorithm is a Riemannian generalization of the Broyden–Fletcher–Goldfarb–Shanno (BFGS) method [13]. It was proposed and analyzed in [17] and allows simpler line searches and vector transports than other implementations without significantly affecting the efficiency of the algorithm.

D. AC Optimal Power Flow Formulation

Our AC optimal power flow model is constructed by version v0.19.9 of PowerModels.jl [11] which is a Julia [10] package which leverages the domain-specific language JuMP [18] for optimization. The detailed formulation of the problem is given in the PowerModels documentation. In brief, we used the standard polar-coordinates for voltage and Cartesian-coordinates for power form of the AC power flow. All equality constraints imposed by this model form our manifold (that is, the function q in (1)). See equations 8-10 in the ACOPF model in the powermodels.jl documentation.

E. Test Problems

Our test problems are drawn from version 23.07 of the pglib-opf repository [19]. We chose cases from the repository with fewer than 300 buses so that the optimization completed in a reasonable amount of time.

V. Results

Fig. 1 gives the relative difference in terminal objective value for each tested algorithm. This value is computed as

$$\left| \frac{f(x_{ro}) - f(x_{ipopt})}{f(x_{ipopt})} \right|, \tag{21}$$

where ro subscript indicates Riemannian optimization solution and the *ipopt* subscript indicates the IPOPT solution. Omitted values indicate that the algorithm failed to terminate in a reasonable amount of time.

Fig. 2 gives the maximum constraint violation for the tested algorithms as well as IPOPT. For the manifold methods, the equality constraints always had small residuals (typically, smaller than 10^{-11}) so that the maximal constraint violation is nearly always for an inequality constraint. Again, values are omitted when the algorithm failed to terminate.

Fig. 3 gives the number of iterations performed prior to termination. For IPOPT, this is the number of iterations reported by the optimizer output. For the Riemannian algorithms, this is the number of iterations performed by the unconstrained subproblem optimizer. Once more, failure to terminate is indicated by omitted values.

We now discuss the solution quality of the specific algorithms before concluding with a few comments about the

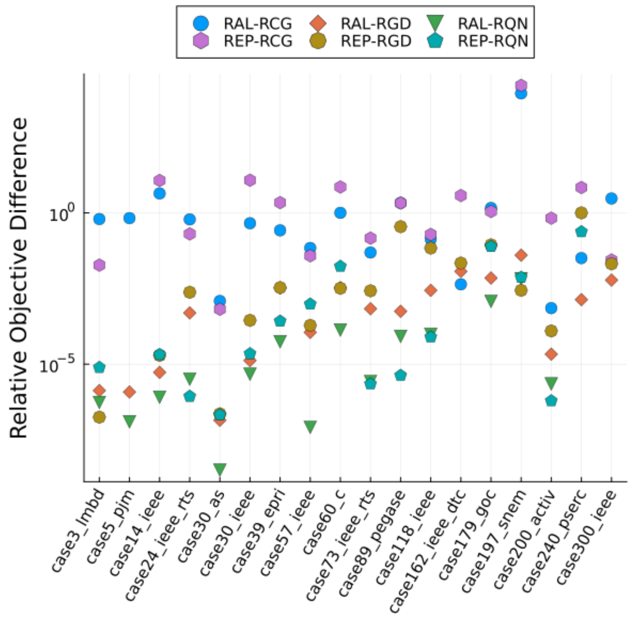

Fig. 1. Relative difference in objective function compared to IPOPT value. Values are omitted when the algorithm failed to terminate.

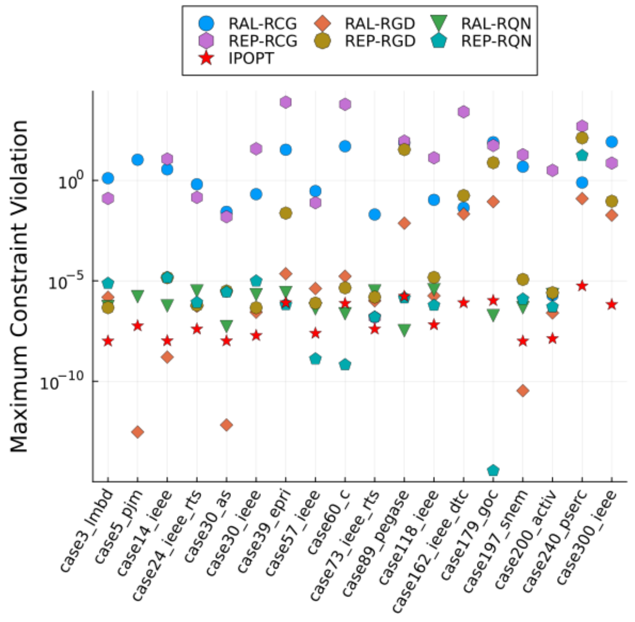

Fig. 2. Maximum constraint violation. Values are omitted when the algorithm failed to terminate.

iteration counts. The RAL and REP with RGD subsolver perform reasonably well with RAL-RGD outperforming REP-RGD. These two methods terminate for all but one case (REP-RGD did not terminate for case5_pjm). For roughly half the tests, RAL-RGD has a relative objective difference on the order of 10^{-6} and in all but one case is within 1% of the IPOPT objective function whereas REP-RGD differences tend to be higher ($\sim 10^{-3}$) with one poor solution (case240_pserc). For both RAL-RGD and REP-RGD, the constraint violations are also small being on the order of 10^{-5} in the majority of tests. These are larger than the IPOPT values which are between 10^{-8} and 10^{-5}. Note also that when REP-RGD does sometime produce solutions with significant (greater than 1) constraint violation.

In our tests, both RAL and REP with RCG subsolver perform poorly. While all cases terminated, the solutions are of poor quality. The relative objective function differences are significant: they are typically on the order of 1 or larger and never less than 10^{-3}. The constraint violation is also frequently quite large and rarely less than 10^{-2}. It should be noted that the equality constraints are small: generally on the order of 10^{-12} or smaller. Since we are using objective based methods to enforce the inequality constraints, the large constraint violation and the large relative objective difference are related. Indeed, the RCG based methods frequently terminated with negative objective values. Since the objective function in our OPF problems is a polynomial with positive coefficients, this is only possible if the inequality constraints are violated (specifically, the generator set points must have negative values). The fact that both the RAL and REP algorithms terminated in these conditions is odd and difficult to explain.

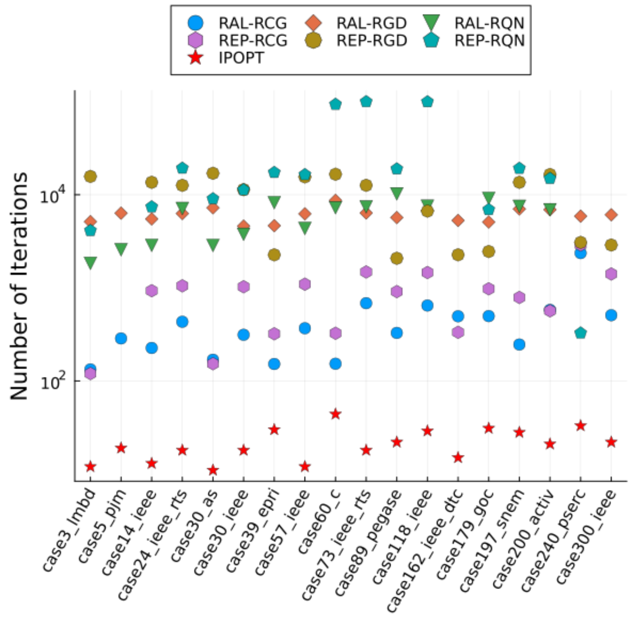

Fig. 3. Number of total inner loop iterations before termination. Values are omitted when the algorithm failed to terminate.

The RQN based algorithms clearly performed the best with RAL-RQN slightly outperforming REP-RQN. However, these two methods also failed to terminate more than the others. This is likely due to the increased computational complexity of the method (and un-optimized implementations with regard to rectractions, vector transport, and memory allocation) rather than a failing in the algorithm. Both of these algorithms consistently produce solutions comparable to IPOPT in terms of objective function value and constraint violation. Indeed, there are several cases where one or both RQN based methods have smaller constraint violation than IPOPT. Note that REP-RQN does have one outlier to this success where it produced a bad solution (case240_pserc).

With regard to iteration count, somewhat surprisingly, the RAL-RCG and REP-RCG algorithms took the fewest iterations of the Riemannian algorithms. RCG methods do have super-linear convergence [16], so we would expect them to take fewer iterations than RGD. However, the RQN methods also converge super-linearly [17] so, at least naively, we would expect these methods to have similar iteration counts. It is possible that the small number of iterations and the poor performance of the RAL-RCG and REP-RCG are related. For example, it is possible the RCG algorithm became trapped in a local minimum early on and the RAL and REP algorithms failed to drive the penalty parameter high enough to move the iterates to a feasible point.

Comparing to IPOPT, all the Riemannian algorithms take significantly more iterations. There are two obvious possible reasons. First, because the Riemannian methods stick to the manifold, they have to travel a longer distance from the initial point to the optimal point since the Riemannian algorithms cannot travel "through" the manifold. Second, due to the curvature of the manifold, the Riemannian algorithms may be forced to take smaller steps. Indeed, we observed that our retraction failed for step sizes that are too large. The exact cause for the much larger iteration count is not clear and requires further investigation.

VI. Conclusion

From our computational experiments, we see that RQN, and to a lesser extent RGD, are capable of producing solutions of similar quality as IPOPT, at least on cases of fewer than 300 buses. The exact reason for the poor performance of RCG methods is unclear and should be further investigated. Further, in all cases, these methods seem to require a significantly larger number of iterations than IPOPT. The reason for this too is unclear. It may be that the Riemannian algorithms require parameter tuning or different line search algorithms.

While these algorithms have proven effective on small scale test systems, it is important to test them on larger systems as well. Many computational problems do not appear until algorithms are used on large scale problems. Such testing requires a more computationally efficient implementation of RQN and RGD than used here.

Despite these questions, there are some notable advantages to this approach. First, since these algorithms search the power flow manifold, every iterate of the algorithm satisfies the AC power flow equations to a tight tolerance (on the order of 10^{-12}). Checking the feasibility of a point (such as that provided by an early termination) is reduced to a check on the inequality constraints. This is much simpler than checking the AC power flow equations.

Second, using computational accelerators (e.g., GPUs) for interior point methods has proven difficult largely due to the numerical linear algebra [4] and the poor performance of current GPU linear solver on these problems [20]. The primary linear solve in the chosen Riemannian methods is the inversion of the symmetric positive-definite matrix $Dq_x Dq_x^T$ as part of the projection operator given in (8). This is likely to be a significantly easier numerical linear algebra problem than the inversion of the saddle point matrix. As a result, these Riemannian algorithms may be easier to accelerate with GPUs than the classical interior point methods.

References

[1] D. K. Molzahn, I. A. Hiskens *et al.*, "A survey of relaxations and approximations of the power flow equations," *Foundations and Trends® in Electric Energy Systems*, vol. 4, no. 1-2, pp. 1–221, 2019.

[2] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, pp. 25–57, 2006.

[3] F. Capitanescu, "Critical review of recent advances and further developments needed in AC optimal power flow," *Electric Power Systems Research*, vol. 136, pp. 57–68, 2016.

[4] J. Maack, "Optimal power flow derived sparse linear solver benchmarks," National Renewable Energy Laboratory (NREL), Golden, CO (United States), Tech. Rep., 2023.

[5] M. Benzi, G. H. Golub, and J. Liesen, "Numerical solution of saddle point problems," *Acta numerica*, vol. 14, pp. 1–137, 2005.

[6] N. Boumal, *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023.

[7] J. M. Lee, *Smooth manifolds*. Springer, 2012.

[8] ——, *Introduction to Riemannian manifolds*. Springer, 2018, vol. 2.

[9] S. Bolognani and F. Dörfler, "Fast power system analysis via implicit linearization of the power flow manifold," in *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2015, pp. 402–409.

[10] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing," *SIAM Review*, vol. 59, no. 1, pp. 65–98, 2017. [Online]. Available: https://epubs.siam.org/doi/10.1137/141000671

[11] C. Coffrin, R. Bent, K. Sundar, Y. Ng, and M. Lubin, "PowerModels.jl: An open-source framework for exploring power flow formulations," in *2018 Power Systems Computation Conference (PSCC)*, June 2018, pp. 1–8.

[12] R. Bergmann, "Manopt.jl: Optimization on manifolds in Julia," *Journal of Open Source Software*, vol. 7, no. 70, p. 3866, 2022.

[13] D. Luenberger and Y. Ye, *Linear and Nonlinear Programming*, ser. International Series in Operations Research & Management Science. Springer International Publishing, 2015.

[14] C. Liu and N. Boumal, "Simple algorithms for optimization on Riemannian manifolds with constraints," *Applied Mathematics & Optimization*, vol. 82, pp. 949–981, 2020.

[15] R. Fletcher, *Practical methods of optimization*. John Wiley & Sons, 2000.

[16] H. Sato, "Riemannian conjugate gradient methods: General framework and specific algorithms with convergence analyses," *SIAM Journal on Optimization*, vol. 32, no. 4, pp. 2690–2717, 2022.

[17] W. Huang, P.-A. Absil, and K. A. Gallivan, "A Riemannian BFGS method without differentiated retraction for nonconvex optimization problems," *SIAM Journal on Optimization*, vol. 28, no. 1, pp. 470–495, 2018.

[18] M. Lubin, O. Dowson, J. Dias Garcia, J. Huchette, B. Legat, and J. P. Vielma, "JuMP 1.0: Recent improvements to a modeling language for mathematical optimization," *Mathematical Programming Computation*, 2023.

[19] S. Babaeinejadsarookolaee, A. Birchfield, R. D. Christie, C. Coffrin, C. DeMarco, R. Diao, M. Ferris, S. Fliscounakis, S. Greene, R. Huang *et al.*, "The power grid library for benchmarking AC optimal power flow algorithms," *arXiv preprint arXiv:1908.02788*, 2019.

[20] K. Świrydowicz, E. Darve, W. Jones, J. Maack, S. Regev, M. A. Saunders, S. J. Thomas, and S. Peleš, "Linear solvers for power grid optimization problems: a review of GPU-accelerated linear solvers," *Parallel Computing*, vol. 111, p. 102870, 2022.