LAGRANGE MULTIPLIERS AND RAYLEIGH QUOTIENT ITERATION IN CONSTRAINED TYPE EQUATIONS

DU NGUYEN
NGUYENDU@POST.HARVARD.EDU

Abstract. We generalize the Rayleigh quotient iteration to a class of functions called vector Lagrangians. The convergent theorem we obtained generalizes classical and nonlinear Rayleigh quotient iterations, as well as iterations for tensor eigenpairs and constrained optimization. In the latter case, our generalized Rayleigh quotient is an estimate of the Lagrange multiplier. We discuss two methods of solving the updating equation associated with the iteration. One method leads to a generalization of Riemannian Newton method for embedded manifolds in a Euclidean space while the other leads to a generalization of the classical Rayleigh quotient formula. Applying to tensor eigenpairs, we obtain both an improvements over the state-of-the-art algorithm, and a new quadratically convergent algorithm to compute all complex eigenpairs of sizes typical in applications. We also obtain a Rayleigh-Chebyshev iteration with cubic convergence rate, and give a clear criterion for RQI to have cubic convergence rate, giving a common framework for existing algorithms.

Key words. Lagrange multiplier, Rayleigh quotient, Newton-Raphson, Eigenvalue, Invariant subspace, Optimization, Tensor decomposition, SQP, Chebyshev.

AMS subject classifications. 65K10, 65F10, 65F15, 15A69

1. Introduction.

Consider three Euclidean spaces E_{in}, E_{out}, E_L with $\dim(E_{in}) = \dim(E_{out})$. We consider a map $\boldsymbol{L} : (\boldsymbol{x}, \boldsymbol{\lambda}) \mapsto \boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda})$ from $E_{in} \oplus E_L$ into E_{out} and a map $\boldsymbol{C} : \boldsymbol{x} \mapsto \boldsymbol{C}(\boldsymbol{x})$ from E_{in} to E_L. The direct sum $\mathcal{L} = \boldsymbol{L} \oplus \boldsymbol{C}$:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \begin{pmatrix} \boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) \\ \boldsymbol{C}(\boldsymbol{x}) \end{pmatrix} \tag{1.1}$$

is a map from $E_{in} \oplus E_L$ to $E_{out} \oplus E_L$. When the Jacobian of $\mathcal{L}$ is invertible in a domain of $E_{in} \oplus E_L$ near a zero of $\mathcal{L}$, $\boldsymbol{L}$ and $\boldsymbol{C}$ have Jacobians both of full row rank in the same domain. In that situation we will call $\boldsymbol{L}$ a *vector* Lagrangian and $\boldsymbol{C}$ a constraint in that domain. The equation

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = 0 \tag{1.2}$$

covers systems of equations where a number of equations in the system are dependent on a group of variables $\boldsymbol{x}$ while the remaining equations involve all variables $(\boldsymbol{x}, \boldsymbol{\lambda})$. The remaining variables are named $\boldsymbol{\lambda}$ in honor of Lagrange. Our assumptions ensure the number of variables in $\boldsymbol{\lambda}$ is the same as the number of constraints in $\boldsymbol{C}(\boldsymbol{x})$ (both equal to $\dim(E_L)$). For the rest of this article, we will refer to Lagrangians dropping the qualifier *vector* except when it could cause confusion, see below.

This setup covers at least four classes of problems encountered in the literature:

- The eigenvector/invariant subspace problem:

$$\begin{aligned} \boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) =& A\boldsymbol{x} - \boldsymbol{x}\boldsymbol{\lambda} \\ \boldsymbol{C}(\boldsymbol{x}) =& \frac{1}{2}(\boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{I}_k) \end{aligned} \tag{1.3}$$

  where $\boldsymbol{x}$ is an $n \times k$ matrix, A is an $n \times n$ matrix. In this case, E_{in} and E_{out} are both $n \times k$ matrices and E_L is the space of symmetric $k \times k$ matrices. The case where $k = 1$ is the eigenvector problem. We have $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = A\boldsymbol{x} - \boldsymbol{x}\boldsymbol{\lambda}$ hence $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda})\eta = A\eta - \eta\boldsymbol{\lambda}$, a fact we will use later on.
- The constraint optimization problem. Here $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \nabla f(\boldsymbol{x}) - \boldsymbol{J}_{\boldsymbol{C}}^T(\boldsymbol{x})\boldsymbol{\lambda}$ where f is a real-valued function. This is the case of the classical Lagrangian multiplier equations. $E_{in} = E_{out}$ is the domain where f is defined and E_L is the target space of the restrictions $\boldsymbol{C}(\boldsymbol{x}) = (\boldsymbol{C}_i(\boldsymbol{x}))$ on $\boldsymbol{x}$. The system (1.1) gives us the set of critical points. In this case, $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda})\eta = (\nabla^2 f(\boldsymbol{x}) - \sum_i \nabla^2 \boldsymbol{C}_i(\boldsymbol{x})\boldsymbol{\lambda}_i)\eta$ where $\boldsymbol{\lambda} = (\boldsymbol{\lambda}_i)$.

- The (real - the complex version is similar) nonlinear eigenvalue problem:

$$\boldsymbol{P}(\lambda)\boldsymbol{x} = 0 \tag{1.4}$$

  Here $\boldsymbol{P}$ is a matrix with polynomial entries in $\boldsymbol{\lambda} = (\lambda)$, $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{P}(\lambda)\boldsymbol{x}$. While this is not in the form (1.1) we can impose the constraint $\boldsymbol{C}(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{x} - 1$ (or $\boldsymbol{C}(\boldsymbol{x}) = \boldsymbol{z}^T\boldsymbol{x} - 1$ for a fixed vector $\boldsymbol{z}$). E_L is of dimension one and λ is a scalar. E_{in} and E_{out} are $\mathbb{R}^k$ where k is the dimension of $\boldsymbol{x}$. There is an extensive literature for this problem ([14]). Note $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{P}(\lambda)$.
- The tensor eigenpairs problem:

$$\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \mathcal{T}(\boldsymbol{I}, \boldsymbol{x}\cdots, \boldsymbol{x}) - \boldsymbol{x}\boldsymbol{\lambda}$$

  where $\mathcal{T}$ is a tensor of order m. We can impose various constraints, a popular one is $\boldsymbol{C}(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x}^T\boldsymbol{x} - 1)$ for the real case, and we will study $\boldsymbol{C}(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x}^*\boldsymbol{x} - 1)$ for the complex case. This is an area of active research. $\mathcal{T}$ is usually assumed to be symmetric. We note $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda}) = (m-1)\mathcal{T}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{x}, \cdots, \boldsymbol{x}) - \boldsymbol{\lambda}\boldsymbol{I}$.

An iteration method called Rayleigh quotient iteration (RQI) is among the most powerful methods to compute eigenvalues and vectors. For a vector $\boldsymbol{v}$, the Rayleigh quotient is

$$\boldsymbol{\lambda} = \frac{\boldsymbol{v}^T A\boldsymbol{v}}{\boldsymbol{v}^T\boldsymbol{v}}$$

(see [1], algorithm 2.2. The quotient reduces to $\boldsymbol{v}^T A\boldsymbol{v}$ on the unit sphere.) The iteration computes

$$\boldsymbol{v}_{i+1} = \frac{(A - \boldsymbol{\lambda}\boldsymbol{I})^{-1}\boldsymbol{v}_i}{||(A - \boldsymbol{\lambda}\boldsymbol{I})^{-1}\boldsymbol{v}_i||}$$

which is shown to have cubic convergence for almost all initial points if A is normal and quadratic otherwise on suitable initial points. We call the equation of form

$$(A - \boldsymbol{\lambda})\boldsymbol{x} = \boldsymbol{w} \tag{1.5}$$

a resolvent equation, as $(A - \boldsymbol{\lambda})^{-1}$ is called the resolvent in the literature. The letter $\boldsymbol{\lambda}$ used in the second example in honor of Lagrange is also often used to denote an eigenvalue, a fact we will show is more than a happy coincident. We note that the equation is defined in the ambient space E_{in}. Similar iterations have been studied in the literature for the three remaining problems mentioned:

- For the constrained optimization problem, [22] studied Feasibly-Projected Sequential Quadratic Program (FP-SQP). With $\boldsymbol{F}(\boldsymbol{x}) = \nabla f$ and with the choice of $\boldsymbol{\lambda} = (\nabla\boldsymbol{C}^T\nabla\boldsymbol{C})^{-1}\nabla\boldsymbol{C}^T\nabla f$ (equation 7 ibid., we changed the sign to conform with our convention), the increment z of the FP-SQP is a solution of equation 5 of that paper, a system defined on E_{in}:

$$(\nabla^2 f - \sum_{i=1}^{n}\nabla^2\boldsymbol{C}\boldsymbol{\lambda}_i)z + \nabla f(x) - \nabla\boldsymbol{C}\boldsymbol{\lambda}_+ = 0$$

$$\nabla\boldsymbol{C}^T z + \boldsymbol{C}(x) = 0$$

  (we renamed Φ to $\boldsymbol{C}$, please see the original paper for full details). The paper shows it is equivalent to Riemannian Newton on the embedded manifold. Note this choice of $\boldsymbol{\lambda}$ reduces to the choice of $\boldsymbol{\lambda}$ in the eigenvector case.
- For the nonlinear eigenvalue problem, for a fixed $\boldsymbol{x}$, λ is solved from the equation $\boldsymbol{x}^T\boldsymbol{P}(\lambda)\boldsymbol{x} = 0$, and the next iteration is obtained by solving the equation

$$\boldsymbol{P}(\lambda)\boldsymbol{x}_{i+1} = \boldsymbol{P}'(\lambda)\boldsymbol{x}_i$$

  Again, this is an equation on the ambient space E_{in}.

- The algorithm NCM in [16] also solves for the increment $\boldsymbol{y}$ satisfying:

$$((m-1)\mathcal{T}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{x},\cdots,\boldsymbol{x})-\boldsymbol{\lambda}\boldsymbol{I})\boldsymbol{y}=-\mathcal{T}(\boldsymbol{I},\boldsymbol{x},\boldsymbol{x},\cdots,\boldsymbol{x})+\boldsymbol{\lambda}\boldsymbol{x} \tag{1.6}$$

  with $\boldsymbol{\lambda}=\boldsymbol{x}^T\mathcal{T}(\boldsymbol{I},\boldsymbol{x},\cdots,\boldsymbol{x})=\mathcal{T}(\boldsymbol{x},\boldsymbol{x},\cdots,\boldsymbol{x})$.

We note the four algorithms listed above all involving equations containing $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{\lambda})$ on E_{in} with particular choices of $\boldsymbol{\lambda}$ as functions of $\boldsymbol{x}$. On the other hand, we have the Riemannian Newton method providing an iteration on the embedded manifold. The iteration equation, in this case, is defined on the *tangent space*, which is of dimension $\dim(E_{in})-\dim(E_L)$, and requires the inversion of the projected Hessian. In many instances, it has been perceived that Riemannian Newton is related to the iterations on the ambient space above: [10] suggested the classical RQI is an approximation of Riemannian Newton, and we have mentioned [22] showed FP-SQP is equivalent to Riemannian Newton. The algorithm O-NCM in [16] is the Riemannian Newton algorithm on the sphere, applying to the tensor eigenpair problem. It requires the inversion of the projected Hessian $U(\boldsymbol{x})^T\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{\lambda})U(\boldsymbol{x})$ where the columns of $U(\boldsymbol{x})$ together with $\boldsymbol{x}$ form an orthonormal basis of $\mathbb{R}^n$.

Depending on situations, inverting $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{\lambda})$ may be easier than inverting the projected Hessian, especially when the original Hessian has a special structure. For example, classical RQI is simpler if A is a tridiagonal matrix. To summarize, for each problem considered above we have an algorithm involving $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{\lambda})$, and in many instances we know they are related to Riemannian Newton on the embedded manifold. Therefore, it is desirable to provide a general framework that explains this relationship.

On the rate of convergence, it is known that we have quadratic convergence for classical RQI, and cubic convergence for normal matrices. For nonnormal matrices, we have the two-sided RQI which also has cubic convergence rate. However, proofs of cubic convergence seem to be specific to each instance. We know that Riemannian Newton has quadratic convergence rate, while we need a separate proof for quadratic convergence for each iteration on the ambient space above. For a general framework, we would like an identification of a class of iterations (containing Riemannian Newton) on the embedded manifold to the corresponding class of ambient space iterations, both having quadratic convergence rate. We would expect to have a generalization of the Rayleigh quotient to express $\boldsymbol{\lambda}$ as a function of $\boldsymbol{x}$, a generalization of the resolvent equation a common proof of quadratic convergence. We also would like simple criteria for cubic convergence.

Our main application will be in the tensor eigenpair problem. In terms of speed, the algorithm O-NCM mentioned above is state-of-the-art, outperforming NCM. O-NCM outperforms the previous state-of-the-art S-HOPM by around 40 percent in execution time. In practice, O-NCM can find all real eigenpairs within a few seconds, but we still need homotopy algorithm [9] to know if we have found all real eigenpairs.

Often, $\boldsymbol{L}(\boldsymbol{x},\boldsymbol{\lambda})$ is an affine function of $\boldsymbol{\lambda}$. We define an *explicit* Lagrangian to be a Lagrangian of the form

$$\boldsymbol{L}(\boldsymbol{x},\boldsymbol{\lambda})=\boldsymbol{F}(\boldsymbol{x})-\boldsymbol{H}(\boldsymbol{x})\boldsymbol{\lambda} \tag{1.7}$$

where $\boldsymbol{F}(\boldsymbol{x})$ is a vector function from an open set in E_{in} to E_{out} and $\boldsymbol{H}(\boldsymbol{x})$ is a linear map from E_L to E_{out} for each x. In the code, we refer to Lagrangians not necessarily of explicit form as implicit Lagrangians. As this may cause confusion, we will not use this term in the article. The Lagrangian of the nonlinear eigenvalue problem is not of explicit form, while the Lagrangians of other three problems are: $\boldsymbol{F}(\boldsymbol{x})=A\boldsymbol{x},\boldsymbol{H}(\boldsymbol{x})=\boldsymbol{x}$ for the eigenvalue problem; $\boldsymbol{F}(\boldsymbol{x})=\nabla f(\boldsymbol{x})$, $\boldsymbol{H}(\boldsymbol{x})=\boldsymbol{J}_{\boldsymbol{C}}^T(\boldsymbol{x})$ for the constrained optimization problem (we note our *vector* Lagrangian $\boldsymbol{L}(\boldsymbol{x},\boldsymbol{\lambda})$ is the gradient of the *scalar* Lagrangian $f(\boldsymbol{x})-\boldsymbol{C}^T(\boldsymbol{x})\boldsymbol{\lambda}$). Finally, $\boldsymbol{F}(\boldsymbol{x})=\mathcal{T}(\boldsymbol{I},\boldsymbol{x},\cdots,\boldsymbol{x})$ and $\boldsymbol{H}(\boldsymbol{x})=\boldsymbol{x}$ in the tensor eigenpair problem.

In this paper we provide a framework to extend Rayleigh quotient iteration to all Lagrangians:

- We show there is a simple variance of Newton-Raphson iteration for (1.1) with a suitable choice of $\boldsymbol{\lambda}$ that ensures quadratic convergence. Theorem 6.2 provides a systematic way to solve (1.2) with a quadratically convergent iteration on the ambient space, with feasible projections.
- In general, the generalized RQI consists of three steps:

  - Evaluating $\boldsymbol{\lambda}$ as a function of $\boldsymbol{x}$:
    - When the Lagrangian is of explicit form, $\boldsymbol{\lambda}$ could be defined by fixing a left inverse of $\boldsymbol{H}$. Using this choice of $\boldsymbol{\lambda}$ we can reduce the iteration to solving an equation in E_{in}.
    - When the Lagrangian is of general form (as in the case of the nonlinear eigenvalue problem), we will need to solve for $\boldsymbol{\lambda}$ in term of $\boldsymbol{x}$ nonlinearly. If this solution is consistent (see below) we again reduce the iteration to solving an equation in E_{in}.
    - The choice of $\boldsymbol{\lambda}$ that we allow is more general that what has been proposed in the literature. It works for all four problems mentioned.
  - Solving a system of linear equations for the updating step. This linear system could be solved in two different ways yielding identical results. One way is to reduce it to an equation on the tangent space $T\mathcal{M}$, giving an extension of the Riemannian Newton equation for all *vector* Lagrangians. The other way is to solve a generalization of the resolvent equation, that is to solve an equation with coefficients $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{\lambda})$. This solution is used to compute the updating step without the need to convert $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{\lambda})$ to an operator on the tangent space.
  - Retracting the solution from $T\mathcal{M}$ to the constraint manifold $\mathcal{M}$.

  As we derived the second method to solve the updating equation by applying Schur complement formula, we call this method the Schur form.

- Our framework provides a uniform approach to the four problems described above. It shows clearly the relationship between the Schur form iteration on E_{in} and Riemannian Newton for those problems. It leads to two new algorithms for the tensor eigenpairs:
- Applying this formulation to the real tensor eigenpair problem, we deduce that O-NCM in [16] could be done by a Schur form iteration. While equivalent, this Schur form iteration execution time improves around 16 percent over that of O-NCM, (30 percent with further code optimization). The difference between the Schur form O-NCM and NCM is subtle: the latter could be written as $-\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{\lambda})^{-1}\boldsymbol{F}(\boldsymbol{x})+\boldsymbol{\lambda}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{\lambda})^{-1}\boldsymbol{x}$, while the former is $-\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{\lambda})^{-1}\boldsymbol{F}(\boldsymbol{x})+\boldsymbol{\lambda}_*\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x},\boldsymbol{\lambda})^{-1}\boldsymbol{x}$ where $\boldsymbol{\lambda}_*$ is given by our main theorem. Because of this difference, Schur form O-NCM outperforms O-NCM (although they are equivalent), while NCM underperforms.
- There is a formula for the number of *complex* tensor eigenpairs [8]. While a number of polynomial algebra packages can solve the tensor eigenpair problem for small size tensors, up to now we are not aware of any attempt to compute all complex eigenpairs for higher rank/order tensors, although an attempt to find a few pairs were discussed in [17]. Computing all real pairs depends on the homotopy method which takes several hours for an $(m=4, n=8)$ tensor ([16]). This tensor has 3280 complex eigenpairs and typically a few hundred real pairs. Applying the general theory developed in this paper to the problem of finding all complex tensor eigenpairs, we found a unitary version of the Schur form O-NCM. This unitary RQI is very effective for tensors with a few thousand complex eigenpairs: For an $(m=4, n=8)$ tensor, it computes over 3000 pairs in a few minutes. The remaining pairs usually take much longer time, but we typically finish within 15 minutes. This allows us to compute (most of the time) *all* complex eigenpairs for such tensors. Our algorithm also identifies the real pairs as a by-product, without the need to run the homotopy algorithm.
- In the general setup, we analyze the cubic convergence criteria and give both a criterion for when an RQI has cubic convergence and constructing a second-order version of RQI called Rayleigh-Chebyshev iteration that has cubic convergence by design. In particular, we have a new cubic convergence algorithm for eigenvectors of nonnormal matrices. We have a similar algorithm for the nonlinear eigenvalue case. We found Rayleigh-Chebyshev underperformed the two-sided iteration for the latter case. However, it outperformed the RQI(quadratic) iteration. Similar to unconstrained high-order iterations, Rayleigh-Chebyshev, in general, does not improve execution time. Theoretically, our analysis implies several known cubic convergence results related to Rayleigh quotient.
- We provide a detailed analysis as well as open-source codes for Newton-Raphson for $\mathcal{L}$ on $E_L \oplus E_{in}$ and reduce it to a form clearly showing its relationship to Riemannian Newton on $\mathcal{M}$.

We note the use of a left inverse to define $\boldsymbol{\lambda}$ already appeared in Gabay's early papers ([11], [12]), where similar analysis was performed for the constrained optimization problem. The left inverse approach implies both the two-sided RQI and the unitary tensor eigenpairs algorithms.

In practice, E_{in}, E_{out}, E_L could be spaces of matrices or tensors. $\boldsymbol{H}$ is then a tensor, and so are the Jacobians of $\boldsymbol{F}$ and $\boldsymbol{L}$. On the theoretical side we will consider the vectorized version of all the spaces involved, leaving the tensor related treatment to specific implementations.

2. Newton-Raphson applying to the eigenvector problem. To explain the ideas involved here we look at the eigenvector problem in detail. Here

$$\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = A\boldsymbol{x} - \boldsymbol{x}\boldsymbol{\lambda}$$

and the constraint is $\boldsymbol{C}(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{I})$ The Jacobian of $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) \oplus \boldsymbol{C}(\boldsymbol{x})$ is

$$\boldsymbol{J}_{\mathcal{L}} = \begin{pmatrix} \boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda}) & -\boldsymbol{x} \\ \boldsymbol{x}^T & 0 \end{pmatrix}$$

where $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda})(\eta) = A\eta - \eta\boldsymbol{\lambda}$. We will try to solve $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda})$ by Newton-Raphson. Let (η, δ) be the Newton steps corresponding to $(\boldsymbol{x}, \boldsymbol{\lambda})$. Applying the Schur complement formula:

$$\begin{pmatrix} \eta \\ \delta \end{pmatrix} = \boldsymbol{J}_{\mathcal{L}}^{-1} \begin{pmatrix} -A\boldsymbol{x} + \boldsymbol{x}\boldsymbol{\lambda} \\ -\frac{1}{2}(\boldsymbol{x}^T\boldsymbol{x} - 1) \end{pmatrix}$$

With $\boldsymbol{x} = \boldsymbol{x}_i$ and $\boldsymbol{\lambda} = \boldsymbol{\lambda}_i$ and shorthand $\boldsymbol{L}_{\boldsymbol{x}}$ for $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda})$ we get:

$$\boldsymbol{\lambda}_{i+1} - \boldsymbol{\lambda}_i = \delta = (\boldsymbol{x}^T\boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{x})^{-1}(\boldsymbol{x}^T\boldsymbol{L}_{\boldsymbol{x}}^{-1}(A\boldsymbol{x} - \boldsymbol{x}\boldsymbol{\lambda})) - (\boldsymbol{x}^T\boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{x})^{-1}\frac{1}{2}(\boldsymbol{x}^T\boldsymbol{x} - 1)$$

$$\boldsymbol{x}_{i+1} - \boldsymbol{x}_i = \eta = -\boldsymbol{L}_{\boldsymbol{x}}^{-1}(A\boldsymbol{x} - \boldsymbol{x}\boldsymbol{\lambda}) + \boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{x}\delta$$

So with $\zeta = \boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{x} = (A - \boldsymbol{\lambda})^{-1}\boldsymbol{x}$ we simplify the updating equations to:

$$\delta = \boldsymbol{\lambda}_{i+1} - \boldsymbol{\lambda}_i = (2\boldsymbol{x}^T\zeta)^{-1}(1 + \boldsymbol{x}^T\boldsymbol{x})$$

$$\eta = -\boldsymbol{x} + \zeta\delta = -\boldsymbol{x} + \zeta(2\boldsymbol{x}^T\zeta)^{-1}(1 + \boldsymbol{x}^T\boldsymbol{x})$$

From here

$$\boldsymbol{x}_i + \eta = \zeta(2\boldsymbol{x}_i^T\zeta)^{-1}(1 + \boldsymbol{x}_i^T\boldsymbol{x}_i)$$

We see $\boldsymbol{x}_i + \eta$ is proportional to $\zeta = (A - \boldsymbol{\lambda})^{-1}\boldsymbol{x}_i$, a result known from classical Rayleigh iteration. The equation for ζ is exactly the resolvent equation. However, the formula for $\boldsymbol{\lambda}_{i+1}$ is iterative. Let us link $\boldsymbol{\lambda}$ with the Rayleigh quotient. Starting with the general equation for the explicit system:

$$\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{H}(\boldsymbol{x})\boldsymbol{\lambda} = 0$$

From the full rank assumption $\boldsymbol{H}(\boldsymbol{x})$ has a left inverse, which is a linear map $\boldsymbol{H}^- : E_{out} \to E_L$ such that $\boldsymbol{H}^-\boldsymbol{H} = \boldsymbol{I}_{E_L}$. The most popular left inverse is perhaps $\boldsymbol{H}^- = (\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T$. $\boldsymbol{H}^- = \boldsymbol{H}^-(\boldsymbol{x})$ is also a function of $\boldsymbol{x}$. We can solve for $\boldsymbol{\lambda}$:

$$\boldsymbol{\lambda} = \boldsymbol{H}^-(\boldsymbol{x})\boldsymbol{F}(\boldsymbol{x}) \tag{2.1}$$

In the eigenvector case, $\boldsymbol{H}(\boldsymbol{x}) = \boldsymbol{x}$, so this is exactly the Rayleigh quotient. For $\boldsymbol{H}^- = (\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T$, (2.1) appeared very early in the multiplier method literature as discussed below. We will show that the calculation above for eigenvectors generalizes naturally to all *vector* Lagrangians.

In the next few sections, we discuss a few set-ups needed to state and prove our theorems.

3. Higher derivatives as tensors.

3. Higher derivatives as tensors. The reader can consult [3] for this section. We use slightly different notations in this paper. Recall we can use tensors to denote linear maps between two vector spaces each represented as matrix or tensor. The map sending a tensor η to the tensor $T\eta$ formed by contracting to the right is the linear map represented by T.

If $\boldsymbol{F}$ is a map between two vector spaces $V = \mathbb{R}^n$ and $W = \mathbb{R}^m$ then its Jacobian $\boldsymbol{J}_{\boldsymbol{F}}$ is a map in the space $L(V, W)$ of linear maps between V and W and is represented by an $m \times n$-matrix.

A second derivative is a linear map between V and $L(V, W) \cong V^* \otimes W$ or an element of $L(V, L(V, W)) \cong V^* \otimes V^* \otimes W$ and can be represented as $m \times n \times n$ tensor. We will denote this map as well as this tensor as $\boldsymbol{J}_F^{(2)}$. In general, we will denote the $l - th$ derivatives as $\boldsymbol{J}_{\boldsymbol{F}}^{(l)}$ and this is an element of $L(V(L(V, \cdots L(V, W)))$ (with l copies of V and one copy of W. We can represent it as a tensor of size $m \times n \cdots \times n$.

For l vectors $\eta_1, \eta_2, \dots, \eta_l$ consider the tuple

$$[\eta_1, \eta_2, \dots, \eta_l]$$

we can define

$$T[\eta_1\eta_2 \dots \eta_l] = (\cdots((T\eta_1)\eta_2)\dots)\eta_l$$

which is the repeated contraction of η_i. We write

$$\eta^{[l]} = [\eta, \eta, \cdots, \eta]$$

l times. With this notation, we can write the Taylor series expansion up to order l around $\boldsymbol{v}$ as:

$$\boldsymbol{F}(\boldsymbol{v}) + \boldsymbol{J}_{\boldsymbol{F}}(\boldsymbol{v})(\boldsymbol{h}) + \frac{1}{2}\boldsymbol{J}_{\boldsymbol{F}}(\boldsymbol{v})(\boldsymbol{h}^{[2]}) + \cdots \frac{1}{l!}\boldsymbol{J}_{\boldsymbol{F}}^{(l)}(\boldsymbol{v})(\boldsymbol{h}^{[l]})$$

where $\boldsymbol{h} = \boldsymbol{x} - \boldsymbol{v}$.

To summarize, there are two maps related to higher derivatives. The map $\boldsymbol{x} \mapsto \boldsymbol{J}_{\boldsymbol{F}}^{(l)}(\boldsymbol{x})$ from V to $V^* \otimes \cdots \otimes V^* \otimes W$ is generally nonlinear resulting in a tensor. For a fixed $\boldsymbol{x}$, that tensor gives a multilinear map acting on the tangent space which is embedded in E_{in}, sending $\boldsymbol{h}$ to $\boldsymbol{J}_{\boldsymbol{F}}^{(l)}(\boldsymbol{v})(\boldsymbol{h}^{[l]})$. In code, we need two functions for these two maps. In general, the second map is just tensor contraction.

4. Retractions. Consider a submanifold $\mathcal{M}$ of $\mathbb{R}^n$ of class C^k. Recall the definitions of retractions from [2]:

- A first-order retraction R is a map from $T\mathcal{M}$ to $\mathcal{M}$ around a point $\bar{x}$ if there exists a neighborhood $\mathcal{U}$ of $(\bar{x}, 0)$ in $T\mathcal{M}$ such that:
  1. $\mathcal{U} \subset \mathrm{dom}(R)$ and the restriction $R : \mathcal{U} \to \mathcal{M}$ is of class C^{k-1}.
  2. $R(x, 0) = 0$ for all $(u, 0) \in \mathcal{U}$
  3. $\boldsymbol{J}_R(x, .) = Id_{T\mathcal{M}}(x) \in \mathcal{U}$
- A second-order retraction on $\mathcal{M}$ is a first-order retraction on $\mathcal{M}$ that satisfies for all $(x, u) \in T\mathcal{M}$,

$$\frac{d^2}{dt^2}R(x, tu)|_{t=0} \in N_{\mathcal{M}}(x) \tag{4.1}$$

$N_{\mathcal{M}}(x)$ is the normal space at x. The exponent map is a second-order retraction. It is shown in that paper that projection and orthographic projections are second-order retractions. The following is clear:

PROPOSITION 4.1. *If $\boldsymbol{r}$ is a retraction on $\mathcal{M}$ then $\boldsymbol{r} \times Id_{E_L}$ is a retraction on $\mathcal{M} \times E_L$, if $\boldsymbol{r}$ is a retraction of second-order then $\boldsymbol{r} \times Id_{E_L}$ is also of second-order.*

By Id_V we mean the identity map on the space V. From this proposition and the result of [2]), we can retract intermediate iteration points to $\mathcal{M} \times E_L$, as a result, $\boldsymbol{x}_i$ can be made elements of $\mathcal{M}$ while $\boldsymbol{\lambda}_i$ is unchanged. A point on $\mathcal{M}$ is called a feasible point and called infeasible otherwise. Iterations on the ambient space where iteration points get retracted to $\mathcal{M}$ at every step is called feasibly-projected iterations. We will consider such iterations for the RQI case. We consider the infeasible Newton-Raphson case on ambient manifolds only in the next section.

5. Newton-Raphson method for Lagrangians on ambient space. This section contains a few long calculations to explain how we come up with the RQI algorithm, in particular, the Schur form solution. Aside from the motivation and a few notations, the next section is independent of this section.

Let us focus on simplifying Newton-Raphson iteration for the case of Lagrangians. The Jacobian of $\mathcal{L}$ is

$$\boldsymbol{J}_{\mathcal{L}}(\boldsymbol{x},\boldsymbol{\lambda}) = \begin{pmatrix} \boldsymbol{L_x}(\boldsymbol{x},\boldsymbol{\lambda}) & \boldsymbol{L_\lambda}(\boldsymbol{x},\boldsymbol{\lambda}) \\ \boldsymbol{J_C}(\boldsymbol{x}) & 0 \end{pmatrix} \tag{5.1}$$

Instead of writing the full $\boldsymbol{h}(\boldsymbol{x},\boldsymbol{\lambda})$ we sometimes shorthand to $\boldsymbol{h}$ to save space in equations, this applies to $\boldsymbol{h} = \boldsymbol{L}, \boldsymbol{L_x}\boldsymbol{L_\lambda}, \boldsymbol{J_C}$ for example. Newton-Raphson iteration in the framework of constraint optimization has been studied by many authors in the literature, including [11], [12], [13], [23], [25]. We will make the connection between $\boldsymbol{L_x}$ with the resolvent equation more explicit, and show $\boldsymbol{\lambda}_i$ converges to the Rayleigh quotient expression. We transform the updating equations to a format closer to one derived from Riemannian Newton optimization which motivates the RQI in the next section.

To invert $\boldsymbol{J}_{\mathcal{L}}(\boldsymbol{x},\boldsymbol{\lambda})$, we can solve the second row block first, similar to Riemannian Newton. This inversion strategy gives us what we call *the tangent form solution*. For an alternative approach, already investigated in [23], [25], we will assume $\boldsymbol{L_x}$ is invertible. As we use the Schur complement formula in this second approach, we will call this solution the *Schur form solution*.

The Schur complement for the top block is $-\boldsymbol{J_C}\boldsymbol{L_x}^{-1}\boldsymbol{L_\lambda}$ evaluated at $(\boldsymbol{x},\boldsymbol{\lambda})$, and the inverse of the Jacobian applied on (a, b) is

$$\boldsymbol{J}_{\mathcal{L}}^{-1}\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \boldsymbol{L_x}^{-1}a - \boldsymbol{L_x}^{-1}\boldsymbol{L_\lambda}[(\boldsymbol{J_C}\boldsymbol{L_x}^{-1}\boldsymbol{L_\lambda})^{-1}\boldsymbol{J_C}\boldsymbol{L_x}^{-1}a - (\boldsymbol{J_C}\boldsymbol{L_x}^{-1}\boldsymbol{L_\lambda})^{-1}b] \\ (\boldsymbol{J_C}\boldsymbol{L_x}^{-1}\boldsymbol{L_\lambda})^{-1}\boldsymbol{J_C}\boldsymbol{L_x}^{-1}a - (\boldsymbol{J_C}\boldsymbol{L_x}^{-1}\boldsymbol{L_\lambda})^{-1}b \end{pmatrix} \tag{5.2}$$

evaluated at $(\boldsymbol{x},\boldsymbol{\lambda})$. With $a = -\boldsymbol{L}(\boldsymbol{x},\boldsymbol{\lambda})$ and $b = -\boldsymbol{C}(\boldsymbol{x})$ the Newton step is $\big(\eta, \delta\big)$ with

$$\delta = -(\boldsymbol{J_C}\boldsymbol{L_x}^{-1}\boldsymbol{L_\lambda})^{-1}\boldsymbol{J_C}\boldsymbol{L_x}^{-1}\boldsymbol{L}(\boldsymbol{x}) + (\boldsymbol{J_C}\boldsymbol{L_x}^{-1}\boldsymbol{L_\lambda})^{-1}\boldsymbol{C}(\boldsymbol{x}) \tag{5.3}$$

$$\eta = -\boldsymbol{L_x}^{-1}\boldsymbol{L}(\boldsymbol{x},\boldsymbol{\lambda}) - \boldsymbol{L_x}^{-1}\boldsymbol{L_\lambda}(\boldsymbol{x},\boldsymbol{\lambda})\delta \tag{5.4}$$

On a feasible starting point, $\boldsymbol{C}(\boldsymbol{x}) = 0$, we thus have:

$$\delta = -(\boldsymbol{J_C}\boldsymbol{L_x}^{-1}\boldsymbol{L_\lambda})^{-1}\boldsymbol{J_C}\boldsymbol{L_x}^{-1}\boldsymbol{L}(\boldsymbol{x},\boldsymbol{\lambda}) \tag{5.5}$$

We also note $\boldsymbol{J_C}\eta = 0$ for a feasible point. For the nonlinear eigenvalue problem, the above process is the Nonlinear Inverse Iteration in the literature ([18], [14]). We will review this in subsection 7.6. Note that Schur complement is widely used in equality constraint optimization, for example, chapter 10 of [6] has essentially the above calculation.

Algorithm 5.1 summarizes the iteration. (There, $Terminal(\eta, err, i)$ implements the exit conditions. Typical terminal criteria include $||err|| < max\_err$ or $||\zeta > max\_zeta||$ subjected to a max iteration count.)

When $\boldsymbol{L}$ is explicit, $\boldsymbol{L_x}$ and $\boldsymbol{L_\lambda}$ are two linear maps:

$$\boldsymbol{L_x}\eta = \boldsymbol{J_F}(\boldsymbol{x})\eta - \boldsymbol{J_H}(\boldsymbol{x})\eta\boldsymbol{\lambda}$$

$$\boldsymbol{L_\lambda}\delta = -\boldsymbol{H}\delta$$

$\boldsymbol{L_x}$ is a generalization of the operator $A - \boldsymbol{\lambda}I$ of eigenvalue problem. We do not need the full inverse of $\boldsymbol{L_x}$ in general, but will need to solve for $\boldsymbol{L_x}\eta = B$ for some matrix B in each iteration. We collect all the results thus far in Theorem 5.1.

Algorithm 5.1 Newton-Raphson with constrained iterations for general Lagrangian.

```
Initialize $\boldsymbol{x}_0$ and $\boldsymbol{\lambda}_0$
$i \leftarrow 0$
$\zeta \leftarrow SMALL\_NUMBER$
$err \leftarrow BIG\_NUMBER$
while not Terminal$(i, \zeta, err)$ do
    Solve for $\zeta$ in $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\zeta = -\boldsymbol{L}_{\lambda}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)$
    Solve for $\xi$ in $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\xi = \boldsymbol{L}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)$
    Compute $\delta \leftarrow (\boldsymbol{J_C}(\boldsymbol{x}_i)\zeta)^{-1}[-\boldsymbol{C}(\boldsymbol{x}_i) + \boldsymbol{J_C}(\boldsymbol{x}_i)\xi]$
    Compute $\boldsymbol{\lambda}_{i+1} \leftarrow \boldsymbol{\lambda}_i + \delta$
    Compute $\eta \leftarrow -\xi + \zeta\delta$
    Compute $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \eta$
    $i \leftarrow i + 1$
    $err \leftarrow \boldsymbol{L}(\boldsymbol{x}_{i+1}, \boldsymbol{\lambda}_{i+1})$
end while
```

Theorem 5.1. *The Newton-Raphson iteration equations for $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda})$ with constraint $\boldsymbol{C}(\boldsymbol{x}) = 0$ are*

$$\boldsymbol{\lambda}_{i+1} - \boldsymbol{\lambda}_i = \delta = (\boldsymbol{J_C}\boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{L}_{\boldsymbol{\lambda}})^{-1}[\boldsymbol{C}(\boldsymbol{x}_i) - \boldsymbol{J_C}\boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{L}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)] \tag{5.6}$$

$$\boldsymbol{x}_{i+1} - \boldsymbol{x}_i = \eta = -\boldsymbol{L}_{\boldsymbol{x}}^{-1}(\boldsymbol{L} + \boldsymbol{L}_{\boldsymbol{\lambda}}\delta) \tag{5.7}$$

If $\boldsymbol{x} \in \mathcal{M}$ then we have

$$\boldsymbol{J_C}\eta = 0$$

In the explicit case $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{H}(\boldsymbol{x})(\boldsymbol{\lambda})$ we have:

$$\boldsymbol{L}_{\boldsymbol{\lambda}}(\boldsymbol{x}, \boldsymbol{\lambda}) = -\boldsymbol{H}(\boldsymbol{x})$$

$$\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda})\eta = \boldsymbol{J_F}(\boldsymbol{x}, \boldsymbol{\lambda})\eta - \boldsymbol{J_H}(\boldsymbol{x})\eta\boldsymbol{\lambda}$$

$$\boldsymbol{\lambda}_{i+1} = (\boldsymbol{J_C}\boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{H})^{-1}[-\boldsymbol{C}(\boldsymbol{x}_i) + \boldsymbol{J_C}\boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{F}(\boldsymbol{x}_i)] \tag{5.8}$$

$$\boldsymbol{x}_{i+1} - \boldsymbol{x}_i = \eta = \boldsymbol{L}_{\boldsymbol{x}}^{-1}(-\boldsymbol{F}(\boldsymbol{x}_i) + \boldsymbol{H}(\boldsymbol{x}_i)\boldsymbol{\lambda}_{i+1}) \tag{5.9}$$

Proof. We already proved the general case. The explicit case is a direct substitution of $\boldsymbol{L}$ into (5.3).□

We note the $\boldsymbol{\lambda}$ is updated first, then $\boldsymbol{\lambda}_{i+1}$ is used in equation for η:

$$\boldsymbol{L}_{\boldsymbol{x}}\eta = -\boldsymbol{F}(\boldsymbol{x}_i) + \boldsymbol{H}(\boldsymbol{x}_i)\boldsymbol{\lambda}_{i+1} \tag{5.10}$$

While we have noted before $\boldsymbol{L}_{\boldsymbol{x}}$ is a generalization of the resolvent operator, the right-hand side of this equation is different from that of Rayleigh quotient. To compute $\boldsymbol{\lambda}_{i+1}$ and η we compute

$$\zeta = \boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)^{-1}\boldsymbol{H}(\boldsymbol{x}_i)$$

$$\nu = \boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)^{-1}\boldsymbol{F}(\boldsymbol{x}_i)$$

In the eigenvector case, ν and ζ are related:

$$\nu = \boldsymbol{x} + \zeta\boldsymbol{\lambda}$$

Algorithm 5.2 Newton-Raphson with constrained iterations for explicit Lagrangian.

```
Initialize $\boldsymbol{x}_0$ and $\boldsymbol{\lambda}_0$
$i \leftarrow 0$
$\zeta \leftarrow SMALL\_NUMBER$
$err \leftarrow LARGE\_NUMBER$
while not Terminal$(i, \zeta, err)$) do
    Solve for $\zeta$ in $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\zeta = \boldsymbol{H}(\boldsymbol{x}_i)$
    Solve for $\nu$ in $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\nu = \boldsymbol{F}(\boldsymbol{x}_i)$
    Compute $\boldsymbol{\lambda}_{i+1} \leftarrow (\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)\zeta)^{-1}[-\boldsymbol{C}(\boldsymbol{x}_i) + \boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)\nu]$
    Compute $\eta \leftarrow -\nu + \zeta\boldsymbol{\lambda}_{i+1}$
    Compute $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \eta$
    $i \leftarrow i + 1$
    $err \leftarrow \boldsymbol{L}(\boldsymbol{x}_{i+1}, \boldsymbol{\lambda}_{i+1})$
end while
```

and we only need to solve for ζ as seen before. In the general case, we need to solve for both ν and ζ. The expression of $\boldsymbol{\lambda}_{i+1}$ is different from the Rayleigh quotient.

Let us now focus on reconciling $\boldsymbol{\lambda}_{i+1}$ with the Rayleigh quotient. From (5.9)

$$\boldsymbol{H}(\boldsymbol{x})\boldsymbol{\lambda}_{i+1} = \boldsymbol{L}_{\boldsymbol{x}}\eta + \boldsymbol{F}(\boldsymbol{x})$$

As before let $\boldsymbol{H}^{-}(\boldsymbol{x})$ be a left invert to $\boldsymbol{H}(\boldsymbol{x})$, we solve for $\boldsymbol{\lambda}_{i+1}$:

$$\boldsymbol{\lambda}_{i+1} = \boldsymbol{H}^{-}(\boldsymbol{x})[\boldsymbol{L}_{\boldsymbol{x}}\eta + \boldsymbol{F}(\boldsymbol{x})] \tag{5.11}$$

We see if η converges to zero, $\boldsymbol{\lambda}_{i+1}$ converges to

$$\boldsymbol{\lambda}_{i+1} = \boldsymbol{H}^{-}(\boldsymbol{x})\boldsymbol{F}(\boldsymbol{x})$$

as noted before. $\boldsymbol{H}^{-}$ may be of a more general form than $(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T$, for example, we can replace $\boldsymbol{H}^T$ with any map $\boldsymbol{H}^{\dagger}$ such that $\boldsymbol{H}^{\dagger}\boldsymbol{H}$ is invertible.

We note that Gabay ([11], [12]) found the same expression for our Rayleigh quotient as an estimate of the Lagrange multiplier, together with a related quasi-Newton method. The special case of $(\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T$ appeared much earlier in the literature (for example [15]). Already in 1977, Tapia ([25]) suggested it is difficult to put a name on it without an extensive search of the literature.

Let $\Pi_{\boldsymbol{H}} = \boldsymbol{I}_{E_{out}} - \boldsymbol{H}\boldsymbol{H}^{-}$. This is a projection to $\mathrm{Im}(\Pi_{\boldsymbol{H}})$. We note $\mathrm{Im}(\Pi_{\boldsymbol{H}}) = \mathrm{Null}(\boldsymbol{H}^{-})$. If $\boldsymbol{H} = \boldsymbol{J}_{\boldsymbol{C}}^T$ and $\boldsymbol{H}^{-} = (\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{J}_{\boldsymbol{C}}^T)^{-1}\boldsymbol{J}_{\boldsymbol{C}}$ then $\Pi_{\boldsymbol{H}}$ is a projection to the tangent space of $\mathcal{M}$ at $\boldsymbol{x}$. Substitute the expression (5.11) in (5.10) and moving η term to one side:

$$\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}\eta = -\Pi_{\boldsymbol{H}}\boldsymbol{F}(\boldsymbol{x}) \tag{5.12}$$

This system of equations for η is of similar format to the Riemannian Newton equations. However, it is dependent on $\boldsymbol{\lambda}$ which is computed recursively. Interestingly, it involves $\boldsymbol{H}$ and not $\boldsymbol{C}$. In the general case, assuming $\boldsymbol{L}_{\boldsymbol{\lambda}}^{-}$ is a left inverse of $\boldsymbol{L}_{\boldsymbol{\lambda}}$, performing a similar substitution we have with $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}} = \boldsymbol{I}_{E_{out}} - \boldsymbol{L}_{\boldsymbol{\lambda}}\boldsymbol{L}_{\boldsymbol{\lambda}}^{-}$:

Proposition 5.2. *For equation (1.7), at a feasible point let η be the $\boldsymbol{x}$ component of the Newton-Raphson update. It satisfies:*

$$\boldsymbol{J}_{\boldsymbol{C}}\eta = 0$$

$$\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}\eta = -\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}.$$

which reduces to $\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}\eta = -\Pi_{\boldsymbol{H}}\boldsymbol{F}(\boldsymbol{x})$ in the explicit case.

For second-order iteration, we note the formula (see [7]):

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \boldsymbol{J}_{\mathcal{L}}^{-1}\mathcal{L} - \frac{1}{2}\boldsymbol{J}_{\mathcal{L}}^{(2)}((\boldsymbol{J}_{\mathcal{L}}^{-1}\mathcal{L})^{[2]}) \tag{5.13}$$

for Chebyshev iteration. We choose Chebyshev over Halley to avoid another linear operator inversion. As explained before with $V = E_{in} \oplus E_L$ and $W = E_{out} \oplus E_L$, $\boldsymbol{J}_{\mathcal{L}}^{[2]}$ is an element of $L(V, L(V, W)$ and $\boldsymbol{J}_{\mathcal{L}}^{-1}\mathcal{L}$ is an element of V and the last term is a contraction of the tensor $\boldsymbol{J}_{\mathcal{L}}^{(2)}$ twice on $\boldsymbol{J}_{\mathcal{L}}^{-1}\mathcal{L}$. We note that

$$\boldsymbol{J}_{\mathcal{L}}^{(2)}\begin{pmatrix}\eta\\ \delta\end{pmatrix}^{[2]} = \begin{pmatrix}\boldsymbol{J}_{\boldsymbol{F}}^{(2)}\eta^{[2]} - (\boldsymbol{J}_{\boldsymbol{H}}^{(2)}\eta^{[2]})\boldsymbol{\lambda} - 2\boldsymbol{J}_{\boldsymbol{H}}[\eta, \delta]\\ \boldsymbol{J}_{\boldsymbol{C}}^{(2)}\eta^{[2]}\end{pmatrix}$$

We proceed to use the Schur complement to evaluate the second-order term to arrive at Chebyshev iteration in Algorithm 5.3. While in general, this iteration may be difficult, when $\boldsymbol{F}$, $\boldsymbol{H}$ and $\boldsymbol{C}$ are at most quadratic the algorithm may be useful. In particular, we have a cubic convergent algorithm for eigenvectors, even when the matrix is not normal: in the Chebyshev term the only nonzero terms are $-\eta\delta$ and $\eta^T\eta$.

Algorithm 5.3 Chebyshev with constrained iterations

```
Initialize $\boldsymbol{x}_0$ and $\boldsymbol{\lambda}_0 = \mathcal{R}(\boldsymbol{x})$
$i \leftarrow 0$
$\zeta \leftarrow SMALL\_NUMBER$
$err \leftarrow LARGE\_NUMBER$
while not Terminal($i, \eta$, err) do
    Solve for $\zeta$ in $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\zeta = \boldsymbol{H}(\boldsymbol{x}_i)$
    Solve for $\nu$ in $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\nu = \boldsymbol{F}(\boldsymbol{x}_i)$
    Compute $\boldsymbol{\lambda}_{i+i} \leftarrow (\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)\zeta)^{-1}[-\boldsymbol{C}(\boldsymbol{x}_i) + \boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)\nu]$
    Compute $\eta \leftarrow -\nu + \zeta\boldsymbol{\lambda}_{i+1}$
    Compute $l_2 \leftarrow \boldsymbol{J}_{\boldsymbol{F}}^{(2)}(\eta^{[2]}) - \boldsymbol{J}_{\boldsymbol{H}}^{(2)}(\eta^{[2]})\boldsymbol{\lambda} - 2\boldsymbol{J}_{\boldsymbol{H}}(\eta)\delta$
    Compute $c_2 \leftarrow \boldsymbol{J}_{\boldsymbol{C}}^{(2)}(\eta^{[2]})$
    Compute $\mathrm{LxInvL}_2 \leftarrow \boldsymbol{L}_{\boldsymbol{x}}^{-1}l_2$
    Compute $\delta_2 \leftarrow (\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}(\boldsymbol{J}_{\boldsymbol{C}}(\mathrm{LxInvL}_2) - \boldsymbol{J}_{\boldsymbol{C}}^{(2)}(\eta^{[2]}))$
    Compute $\eta_2 \leftarrow \mathrm{LxInvL}_2 + (\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}l_2$
    Compute $\boldsymbol{\lambda}_{i+i} \leftarrow \boldsymbol{\lambda}_{i+i} - \frac{1}{2}\delta_2$
    Compute $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \eta - \frac{1}{2}\eta_2$
    $i \leftarrow i + 1$
    $err \leftarrow \boldsymbol{L}(\boldsymbol{x}_{i+1}, \boldsymbol{\lambda}_{i+1})$
end while
```

When $\boldsymbol{x}$ is a vector and $\boldsymbol{H}(\boldsymbol{x})$ is represented as a matrix, ζ is a matrix and $\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)\zeta$ can be represented as a square matrix, so the calculation is simple. When $\boldsymbol{x}$ is a matrix, $\boldsymbol{H}(\boldsymbol{x})$ could be a higher-order tensor, so ζ and $\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)\zeta$ in general are tensors. The main difficulty of this method is in evaluating these tensors and inverting the Schur complement $\boldsymbol{J}_C(\boldsymbol{x}_i)\zeta$.

6. Rayleigh Quotient Iteration.

Motivated by [1] (we learned about [11], [12] late in our research), Proposition 5.2 and the above analysis on Lagrange multipliers, with:

$$\boldsymbol{\lambda} = \mathcal{R}(\boldsymbol{x}) := \boldsymbol{H}^{-}(\boldsymbol{x})\boldsymbol{F}(\boldsymbol{x})$$

in the expression for $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda})$, it is plausible that the system:

$$\begin{aligned}\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}\eta =& -\Pi_{\boldsymbol{H}}\boldsymbol{F}(\boldsymbol{x})\\ \boldsymbol{J}_{\boldsymbol{C}}\eta =& 0\end{aligned} \tag{6.1}$$

would provide a generalization of RQI to vector Lagrangians. Using the augmented Lagrangian technique, Gabay ([12]) proposed a quasi-Newton method with this expression of $\boldsymbol{\lambda}$ as an estimate for the Lagrange multiplier. He showed that it converges superlinearly in general. We will prove quadratic convergence of (6.1) and also consider the Chebyshev version.

Similar to the Newton-Raphson case, if $\boldsymbol{L_x}$ is invertible we have a solution to this system in term of $\boldsymbol{L}_x^{-1}$. In fact, let $\nu = \boldsymbol{L_x}^{-1}\boldsymbol{F}(\boldsymbol{x})$ and $\zeta = \boldsymbol{L_x}^{-1}\boldsymbol{H}$ we see

$$\eta = \zeta(\boldsymbol{J_C}\zeta)^{-1}\boldsymbol{J_C}\nu - \nu \tag{6.2}$$

satisfies (6.1) by direct calculation. In essence, this is a projection of $-\nu$ to the tangent space in the direction of ζ. As before we call it the Schur form solution. Before we proceed with the theorems and the proofs, let us give evidence that the iterative process associated with equation (6.1) is a familiar one, in two instances.

The first instance is the eigenvector problem with $\boldsymbol{\lambda} = (\boldsymbol{x}^T\boldsymbol{x})^{-1}\boldsymbol{x}^T A\boldsymbol{x}$ and $\boldsymbol{L_x}\eta = A\eta - \eta\boldsymbol{\lambda}$. (6.1) is the resultant equation and (6.2) shows $\boldsymbol{x}_i + \eta = (A - \boldsymbol{\lambda})\boldsymbol{x}_i$. The iterative process is exactly the classical RQI if we use the projection to the sphere as the retraction.

The second instance is the constraint optimization problem: $\boldsymbol{F} = \nabla f$, $\boldsymbol{H} = \boldsymbol{J_C}^T = \nabla\boldsymbol{C}$ and $\boldsymbol{H}^- = (\boldsymbol{J_C}\boldsymbol{J_C}^T)^{-1}\boldsymbol{J_C} = (\nabla\boldsymbol{C}^T\nabla\boldsymbol{C})^{-1}\nabla\boldsymbol{C}^T$ and hence $\boldsymbol{\lambda} = (\nabla\boldsymbol{C}^T\nabla\boldsymbol{C})^{-1}\nabla\boldsymbol{C}^T\nabla f$. In the notation of [3], the projection to the tangent space of $\mathcal{M}$ at $\boldsymbol{x}$ is simply $\Pi_{\boldsymbol{H}}$:

$$\Pi_{\boldsymbol{H}}(\boldsymbol{x}) = \boldsymbol{I}_{E_{out}} - \boldsymbol{J_C}^T(\boldsymbol{J_C}\boldsymbol{J_C}^T)^{-1}\boldsymbol{J_C} = \boldsymbol{I}_{E_{out}} - \nabla\boldsymbol{C}(\nabla\boldsymbol{C}^T\nabla\boldsymbol{C})^{-1}\nabla\boldsymbol{C}^T = P_{\boldsymbol{x}}$$

Hence the Riemannian gradient is simply $\Pi_{\boldsymbol{H}}(\boldsymbol{x})\boldsymbol{F}$. Note $\boldsymbol{J_C} = (\nabla\boldsymbol{C})^T, \boldsymbol{J_H} = \nabla^2\boldsymbol{C}$ so

$$\boldsymbol{L_x}\eta = \boldsymbol{J_F}\eta - \nabla^2\boldsymbol{C}\eta\boldsymbol{\lambda} = \nabla^2 f\eta - \nabla^2\boldsymbol{C}\eta(\nabla\boldsymbol{C}^T\nabla\boldsymbol{C})^{-1}\nabla\boldsymbol{C}^T\nabla f$$

Formula (5.15) in section 5.3 of [3] shows the Riemannian Hessian of f is:

$$\operatorname{Hess} f[\eta] = P_{\boldsymbol{x}}(D(P_{\boldsymbol{x}}\nabla f)\eta = P_{\boldsymbol{x}}D(\nabla f - \boldsymbol{J_C}^T(\boldsymbol{J_C}\boldsymbol{J_C}^T)^{-1}\boldsymbol{J_C}\nabla f)$$

Here $D = \nabla$ is the classical derivatives with respect to $\boldsymbol{x}$. Expanding the above and keep exploiting the fact that $P_{\boldsymbol{x}} = \Pi_{\boldsymbol{H}}$ annihilates terms starting with $\boldsymbol{J_C}^T$

$$\begin{aligned}\operatorname{Hess} f[\eta] &= P_{\boldsymbol{x}}(D(\nabla f - \boldsymbol{J_C}^T(\boldsymbol{J_C}\boldsymbol{J_C}^T)^{-1}\boldsymbol{J_C}\nabla f)\eta \\ &= P_{\boldsymbol{x}}\nabla^2 f\eta - P_{\boldsymbol{x}}D(\boldsymbol{J_C}^T(\boldsymbol{J_C}\boldsymbol{J_C}^T)^{-1}\boldsymbol{J_C})\nabla f \\ &= P_{\boldsymbol{x}}\nabla^2 f\eta - P_{\boldsymbol{x}}\nabla^2\boldsymbol{C}\eta\boldsymbol{\lambda} = P_{\boldsymbol{x}}(\nabla^2 f - \nabla^2\boldsymbol{C}\eta\boldsymbol{\lambda})\end{aligned}$$

and this is exactly $\Pi_{\boldsymbol{H}}\boldsymbol{L_x}\eta$. So the first equation of (6.1) is the Riemannian Newton equation for the embedded manifold $\mathcal{M}$ in this case. This result is already known from [22].

We note $\Pi_{\boldsymbol{H}}\boldsymbol{L_x}$ restricts to a map from $T\mathcal{M}_{\boldsymbol{x}}$ to $\operatorname{Im}(\Pi_{\boldsymbol{H}}) = \operatorname{Null}(\boldsymbol{H}^-)$, both of the same dimension $\dim(E_{in}) - \dim(E_L)$ and hence it could have an inverse. From the above analysis, it is a generalization of the Riemannian Hessian, while $\Pi_{\boldsymbol{H}}\boldsymbol{L}$ is a generalization of the Riemannian gradient. As we need the Riemannian Hessian to be invertible for Riemannian Newton to work, we would need $\Pi_{\boldsymbol{H}}\boldsymbol{L_x}$ to be invertible and to satisfy a smoothness condition which we will assume in the following theorem:

Theorem 6.1. *Let $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{H}(x)\boldsymbol{\lambda}$ be a Lagrangian of explicit form, $\boldsymbol{H}^-(\boldsymbol{x})$ be a left inverse of $\boldsymbol{H}(\boldsymbol{x})$, $\Pi_{\boldsymbol{H}} = \boldsymbol{I}_{E_{out}} - \boldsymbol{H}(\boldsymbol{x})\boldsymbol{H}^-(\boldsymbol{x})$ be the associated projection, $\mathcal{R}(x) = \boldsymbol{H}^-(\boldsymbol{x})\boldsymbol{F}(\boldsymbol{x})$ be the generalized Rayleigh quotient. Let $(\boldsymbol{v}, \boldsymbol{\mu})$ be a solution to the equation (1.7) with $\boldsymbol{\mu} = \mathcal{R}(\boldsymbol{v})$ and $\boldsymbol{r}$ be a first-order retraction. Assume:*

- *$\boldsymbol{H}, \boldsymbol{F}$ are of class C^2.*
- *$\mathcal{R}$ is of class C^1 in a neighborhood of $(\boldsymbol{v})$.*
- *$\Pi_{\boldsymbol{H}}(\boldsymbol{x})$ is of class C^1.*

- *the map* $\Pi_{\boldsymbol{H}}(\boldsymbol{x})\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x})$ *from* $T\mathcal{M}_{\boldsymbol{x}}$ *to* $\mathrm{Im}(\Pi_{\boldsymbol{H}}(\boldsymbol{x})) = \mathrm{Null}(\boldsymbol{H}^{-}(\boldsymbol{x}))$ *is invertible and for* $\boldsymbol{x}$ *in a neighborhood of* $\boldsymbol{v}$*:*

$$||\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x},\mathcal{R}(\boldsymbol{x}))\psi|| \geq C||\psi|| \tag{6.3}$$

then for a starting point $\boldsymbol{x}_0$ *close enough to* $\boldsymbol{v}$*, the iteration*

$$\begin{aligned}\boldsymbol{\lambda}_i =&\mathcal{R}(\boldsymbol{x}_i)\\ \Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}_i}(\boldsymbol{x}_i,\mathcal{R}(\boldsymbol{x}_i))\eta =&-\Pi_{\boldsymbol{H}}\boldsymbol{F}(\boldsymbol{x}_i)\\ \boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)\eta =&0\\ \boldsymbol{x}_{i+1} =& \boldsymbol{r}(\boldsymbol{x}_i,\eta)\end{aligned}$$

where the step η *is a solution to the second and third equations, provides an update to an iteration converging to* $(\boldsymbol{v},\mathcal{R}(\boldsymbol{v}))$ *quadratically.*

If further $\boldsymbol{H},\boldsymbol{F}$ *are of class* C^3*,* $\mathcal{R}$ *is of class* C^2 *and* $\boldsymbol{r}$ *is a second-order retraction, let*

$$\boldsymbol{G}(\boldsymbol{x})[\eta^{[2]}] := -\frac{1}{2}\boldsymbol{J}_{\boldsymbol{F}}^{(2)}(\boldsymbol{x})[\eta^{[2]}] + \boldsymbol{J}_{\boldsymbol{H}}(\boldsymbol{x})[\eta]\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{x})[\eta] + \frac{1}{2}\boldsymbol{J}_{\boldsymbol{H}}^{(2)}(\boldsymbol{x})[\eta^{[2]}]\mathcal{R}(\boldsymbol{x}) \tag{6.4}$$

then for a starting point $\boldsymbol{x}_0$ *close enough to* $\boldsymbol{v}$ *the Rayleigh-Chebyshev iteration with update:*

$$\boldsymbol{x}_{i+1} = \boldsymbol{r}(\boldsymbol{x}_i,\tau)$$

where τ *is such that* $\tau \leq C_{\boldsymbol{v}}|\boldsymbol{x}_i - \boldsymbol{v}|$ *for some constant* $C_{\boldsymbol{v}}$*,* $\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}\tau = \Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}(\eta_* + \boldsymbol{T}(\boldsymbol{x}_i)[\eta_*^{[2]}])$ *with*

$$\begin{gathered}\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i,\mathcal{R}(\boldsymbol{x}_i))\eta_* = -\Pi_{\boldsymbol{H}}\boldsymbol{F}(\boldsymbol{x}_i)\\ \Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}\boldsymbol{T}(\boldsymbol{x}_i)[\eta_*^{[2]}] = \Pi_{\boldsymbol{H}}\boldsymbol{G}(\boldsymbol{x}_i)[\eta_*^{[2]}]\\ \boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)[\tau] = 0\end{gathered} \tag{6.5}$$

converges cubically to $\boldsymbol{v}$*.*

If $\Pi_{\boldsymbol{H}}(\boldsymbol{v})\boldsymbol{G}(\boldsymbol{v}) = 0$ *and* $\boldsymbol{r}$ *is a second-order retraction, the Rayleigh quotient iteration converges cubically.*

If $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i,\boldsymbol{\lambda}_i)$ *is invertible the above system has the below solution, called the Schur form:*

$$\begin{gathered}\nu = \boldsymbol{L}_{\boldsymbol{x}}^{-1}(\boldsymbol{x}_i,\boldsymbol{\lambda}_i)\boldsymbol{F}(\boldsymbol{x}_i)\\ \zeta = \boldsymbol{L}_{\boldsymbol{x}}^{-1}(\boldsymbol{x}_i)\boldsymbol{H}(\boldsymbol{x}_i)\\ \boldsymbol{\lambda}_* = (\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)\zeta)^{-1}\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)\nu\\ \eta = -\nu + \zeta\boldsymbol{\lambda}_*\\ \eta_* = \eta\\ \boldsymbol{T}(\eta_*) = \boldsymbol{L}_{\boldsymbol{x}}^{-1}(\boldsymbol{x}_i)\{-\frac{1}{2}\boldsymbol{J}_{\boldsymbol{F}}^{(2)}(\boldsymbol{x}_i)[\eta_*^{[2]}] - \boldsymbol{J}_{\boldsymbol{H}}(\boldsymbol{x}_i)[\eta_*]\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{x}_i)[\eta_*] + \frac{1}{2}\boldsymbol{J}_{\boldsymbol{H}}^{(2)}(\boldsymbol{x}_i)[\eta_*^{[2]}]\mathcal{R}(\boldsymbol{x}_i)\}\\ \tau = \eta_* + \boldsymbol{T}(\eta_*) - \zeta(\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)\zeta)^{-1}\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)[\eta_* + \boldsymbol{T}(\eta_*)]\end{gathered} \tag{6.6}$$

We will state and prove the theorem for the general Lagrangian case, as the notations turn out to be simpler and the estimates clearly show the relationship to Taylor series. The explicit case will follow as a corollary. Note that for Rayleigh-Chebyshev iteration, we can just choose $\eta_* = \eta$ but we state the theorem in slightly more general form in case there is an easier to compute η_* that may not be in the tangent space. The technical requirement is $\tau \leq C_{\boldsymbol{v}}|\boldsymbol{x}_i - \boldsymbol{v}|$. This requirement is satisfied if η_* is chosen as η of the RQI step, in that case we require $\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{T}[\eta^{[2]}] = 0$ and can solve for $\boldsymbol{T}[\eta^{[2]}]$ uniquely. (The Rayleigh Chebyshev step is in general expensive so we tried a few ways to pick a less expensive η_*, but we did not succeed. However we leave the more general statement here.)

Theorem 6.2. *Let $\boldsymbol{L}(\boldsymbol{x},\boldsymbol{\lambda})$ be a Lagrangian, $\boldsymbol{L_x}, \boldsymbol{L_\lambda}$ be its partial derivatives with respect to $\boldsymbol{x}$ and $\boldsymbol{\lambda}$. Let $(\boldsymbol{v},\boldsymbol{\mu})$ be a solution for the system* (1.1)*. Assuming*

- *$\boldsymbol{L}$ is of class C^2.*
- *$\mathcal{R}$ is a function of class C^1 from a neighborhood of $\boldsymbol{v}$ to E_L such that $\mathcal{R}(\boldsymbol{v}) = \boldsymbol{\mu}$.*
- *$\boldsymbol{L_\lambda}^-$ is a left inverse of $\boldsymbol{L_\lambda}$ of class C^1 and $\Pi_{\boldsymbol{L_\lambda}} = \boldsymbol{I}_{E_{out}} - \boldsymbol{L_\lambda}\boldsymbol{L_\lambda}^-$.*
- *$\Pi_{\boldsymbol{L_\lambda}}\boldsymbol{L_x}(\boldsymbol{x})$, as a map from $T\mathcal{M} = \mathrm{Null}(\boldsymbol{J_C}(\boldsymbol{x}))$ to $\mathrm{Im}(\Pi_{\boldsymbol{L_\lambda}}(\boldsymbol{x})) = \mathrm{Null}(\boldsymbol{L_\lambda}^-(\boldsymbol{x},\boldsymbol{\lambda}))$, is invertible, such that*

  $$||\Pi_{\boldsymbol{L_\lambda}}\boldsymbol{L_x}(\boldsymbol{x},\mathcal{R}(\boldsymbol{x}))\psi|| \geq C||\psi|| \tag{6.7}$$

  in a neighborhood of $\boldsymbol{v}$.
- *$\boldsymbol{r}$ is a first-order retraction.*

Then the generalized Rayleigh quotient iteration $\boldsymbol{x}_{i+1} = \boldsymbol{r}(\boldsymbol{x}_i,\eta)$ with

$$\begin{gathered}\boldsymbol{\lambda}_i = \mathcal{R}(\boldsymbol{x}_i)\\ \Pi_{\boldsymbol{L_\lambda}}\boldsymbol{L_x}(\boldsymbol{x}_i,\mathcal{R}(\boldsymbol{x}_i))\eta = -\Pi_{\boldsymbol{L_\lambda}}\boldsymbol{L}(\boldsymbol{x}_i,\mathcal{R}(\boldsymbol{x}_i))\\ \boldsymbol{J_C}(\boldsymbol{x}_i)\eta = 0\end{gathered} \tag{6.8}$$

converges quadratically to $(\boldsymbol{v},\boldsymbol{\mu})$. If further $\boldsymbol{L}$ is of class C^3, $\mathcal{R}$ is of class C^2 and $\boldsymbol{r}$ is a second-order retraction, let

$$\boldsymbol{G}(\boldsymbol{x})[\eta_*^{[2]}] := -\frac{1}{2}\boldsymbol{L_{xx}}(\boldsymbol{x})[\eta_*^{[2]}] - \boldsymbol{L_{x\lambda}}(\boldsymbol{x})[\eta_*,\boldsymbol{J}_{\mathcal{R}}[\eta_*]] - \frac{1}{2}\boldsymbol{L_{\lambda\lambda}}(\boldsymbol{x})[(\boldsymbol{J}_{\mathcal{R}}[\eta_*]^{[2]})] \tag{6.9}$$

then the Rayleigh-Chebyshev iterative process $\boldsymbol{x}_{i+1} = \boldsymbol{r}(x_i,\tau)$ with constructed to satisfy:

$$\begin{gathered}\boldsymbol{\lambda}_i = \mathcal{R}(\boldsymbol{x}_i)\\ \Pi_{\boldsymbol{L_\lambda}}\boldsymbol{L_x}(\boldsymbol{x}_i)\eta_* = -\Pi_{\boldsymbol{L_\lambda}}\boldsymbol{L}(\boldsymbol{x}_i,\mathcal{R}(\boldsymbol{x}_i))\\ \Pi_{\boldsymbol{L_\lambda}}\boldsymbol{L_x}(\boldsymbol{x}_i)\boldsymbol{T}(\boldsymbol{x}_i)[\eta_*^{[2]}] = \Pi_{\boldsymbol{L_\lambda}}\boldsymbol{G}(\boldsymbol{x}_i)[\eta_*^{[2]}]\\ \Pi_{\boldsymbol{L_\lambda}}\boldsymbol{L_x}(\boldsymbol{x}_i)\tau = \Pi_{\boldsymbol{L_\lambda}}\boldsymbol{L_x}(\boldsymbol{x}_i)(\eta_* + \boldsymbol{T}(\eta_*))\\ \boldsymbol{J_C}(\boldsymbol{x}_i)[\tau] = 0\\ ||\tau|| \leq C_{\boldsymbol{v}}||\boldsymbol{x}_i - \boldsymbol{v}|| \text{ for some constant } C_{\boldsymbol{v}}\end{gathered} \tag{6.10}$$

converges cubically. If $\Pi_{\boldsymbol{L_\lambda}}(\boldsymbol{v})\boldsymbol{G}(\boldsymbol{v}) = 0$ and $\boldsymbol{r}$ is of second-order the generalized RQI converges cubically. When $\boldsymbol{L_x}$ is invertible the Schur form of the solution exists with:

$$\begin{gathered}\boldsymbol{\lambda}_i = \mathcal{R}(\boldsymbol{x}_i)\\ \zeta = -\boldsymbol{L_x}^{-1}(\boldsymbol{x}_i,\boldsymbol{\lambda})\boldsymbol{L_\lambda}(\boldsymbol{x}_i,\boldsymbol{\lambda})\\ \xi = \boldsymbol{L_x}^{-1}(\boldsymbol{x}_i,\boldsymbol{\lambda})\boldsymbol{L}(\boldsymbol{x}_i,\boldsymbol{\lambda})\\ \theta_* = (\boldsymbol{J_C}(\boldsymbol{x}_i)\zeta)^{-1}\boldsymbol{J_C}(\boldsymbol{x}_i)\xi\\ \eta = -\xi + \zeta\theta_*\\ \eta_* = \eta\\ \boldsymbol{T}(\boldsymbol{x}_i)(\eta_*^{[2]}) = \boldsymbol{L_x}^{-1}\boldsymbol{G}(\boldsymbol{x}_i)[\eta_*^{[2]}]\\ \tau_* = \eta_* + \boldsymbol{T}(\boldsymbol{x}_i)(\eta_*^{[2]})\\ \tau = \tau_* - \zeta(\boldsymbol{J_C}(\boldsymbol{x}_i)\zeta)^{-1}\boldsymbol{J_C}(\boldsymbol{x}_i)\tau_*\end{gathered} \tag{6.11}$$

In existing Rayleigh quotient literature convergence proofs are usually discussed in terms of distance of $\boldsymbol{\lambda}$ to eigenvalues. (6.7) replaces the distance to eigenvalues discussion. If B is a linear operator and B has bounded inverse then $||B^{-1}\psi|| \leq ||B^{-1}||_{op}||\psi||$ and

$$||B^{-1}(B\psi)|| \leq ||B^{-1}||_{op}||B\psi||$$

and hence $||B\psi|| \geq \frac{1}{||B^{-1}||_{op}}||\psi||$. A continuous family of operators $B(\boldsymbol{x})$ in a bounded neighborhood of $\boldsymbol{v}$ would have $\frac{1}{||B^{-1}(\boldsymbol{x})||_{op}}$ locally bounded by the same constant C. So (6.7) only requires the inverse of $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}$ to be locally bounded. When $B = \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(A - \boldsymbol{\lambda}\boldsymbol{I})$, bounds on B can be discussed in term of distance from $\boldsymbol{\lambda}$ to eigenvalues. This condition allows us to translate an estimate of $||\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \mathcal{R}(\boldsymbol{x}))(\boldsymbol{x} - \boldsymbol{v})||$ to an estimate of convergence rate of $||\boldsymbol{x} - \boldsymbol{v}||$.

When $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda})$ is explicit, we can take $\mathcal{R}(x) = \boldsymbol{H}^{-}(x)\boldsymbol{F}(x)$. When it is not explicit as in the nonlinear eigenvalue example below, $\mathcal{R}$ is most often given as a solution to a system

$$\mathcal{N}(\boldsymbol{x}, \boldsymbol{\lambda}) = 0$$

where $\mathcal{N}$ is a map from $E_{in} \oplus E_L$ to E_L with continuous derivatives up to degree two such that if $(\boldsymbol{x}, \boldsymbol{\lambda})$ satisfy the general Lagrangian system $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = 0$, then $\mathcal{N}(\boldsymbol{x}, \boldsymbol{\lambda}) = 0$ and $\mathcal{N}_{\boldsymbol{\lambda}}$ is invertible. It means the defining equations for $\mathcal{N}$ are consequences of equations for $\mathcal{L}$. Assuming $\boldsymbol{J}_{\mathcal{N}}$ is of rank $\dim(E_L)$ in a neighborhood of $(\boldsymbol{v}, \boldsymbol{\mu})$ and $\mathcal{R}$ is the implicit function solution. Then $\mathcal{N}(\boldsymbol{x}, \mathcal{R}(\boldsymbol{x})) = 0$ for all $\boldsymbol{x}$ in a neighborhood of $\boldsymbol{v}$ and hence:

$$\boldsymbol{J}_{\mathcal{R}} = -\mathcal{N}_{\boldsymbol{\lambda}}^{-1}\mathcal{N}_{\boldsymbol{x}} \tag{6.12}$$

As before, we want to allow some flexibility in choosing η_* for the Chebyshev step, therefore giving the requirement $||\tau|| \leq C_{\boldsymbol{v}}||\boldsymbol{x}_i - \boldsymbol{v}||$. This requirement is satisfied if η_* is given by the RQI step and $\boldsymbol{T}(\eta_*^{[2]})$ is solved with the requirement $\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{T}(\eta_*^{[2]}) = 0$. The bound on τ comes from (6.7).

Proof. Our philosophy is Rayleigh and Rayleigh-Chebyshev iterations are modified forms of Newton-Raphson and Chebyshev iterations for $\mathcal{L}$. As expected, our proofs will involve Taylor series expansion of $\boldsymbol{L}$ to different degrees. We first look at the Rayleigh case where Taylor expansion to degree 1 gives

$$0 = \boldsymbol{L}(\boldsymbol{v}, \boldsymbol{\mu}) = \boldsymbol{L}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i)) + \boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i))(\boldsymbol{v} - \boldsymbol{x}_i) + \boldsymbol{L}_{\boldsymbol{\lambda}}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i))(\boldsymbol{\mu} - \mathcal{R}(\boldsymbol{x}_i)) +$$
$$O(||\begin{pmatrix}\boldsymbol{x}_i \\ \boldsymbol{\mu}\end{pmatrix} - \begin{pmatrix}\boldsymbol{v} \\ \mathcal{R}(\boldsymbol{x}_i)\end{pmatrix}||^2))$$

By the C^1 assumption of $\mathcal{R}$ and note $\boldsymbol{\mu} = \mathcal{R}(\boldsymbol{v})$ we have $||\boldsymbol{\mu} - \mathcal{R}(x_i)|| \leq C_1||\boldsymbol{v} - \boldsymbol{x}_i||$. So the last term above is $O(||\boldsymbol{v} - \boldsymbol{x}_i||^2)$. Applying $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\boldsymbol{x}_i)$ to both sides and rearrange the terms:

$$\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i))(\boldsymbol{x}_i - \boldsymbol{v}) = \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i)) + O(||\boldsymbol{v} - \boldsymbol{x}_i||^2)$$

Note that we use $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\boldsymbol{x}_i)\boldsymbol{L}_{\boldsymbol{\lambda}} = 0$ in the above, plus continuous derivative condition of $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}$ for the last term. Expressing $(\boldsymbol{x}_{i+1} - \boldsymbol{v}) = \boldsymbol{x}_i - \boldsymbol{v} + \eta + O(||\eta||^2)$ using the first-order retraction and adding $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i))\eta$ to both sides

$$\begin{aligned}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i))(\boldsymbol{x}_{i+1} - \boldsymbol{v}) = \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i)) + \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i))[\eta] + \\ +O(||\boldsymbol{v} - \boldsymbol{x}_i||^2) + O(||\eta||^2)\end{aligned} \tag{6.13}$$

Choose η in the tangent space satisfying:

$$\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}\eta = -\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i))$$

$\boldsymbol{L}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i))$ is $O||\boldsymbol{x}_i - \boldsymbol{v}||$, so (6.7) shows $O(||\eta||^2)$ is also $O(||\boldsymbol{x}_i - \boldsymbol{v}||^2)$. Finally:

$$||\boldsymbol{x}_{i+1} - \boldsymbol{v}|| \leq \frac{1}{C}||\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x})(\boldsymbol{x}_{i+1} - \boldsymbol{v})|| = O||\boldsymbol{x}_i - \boldsymbol{v}||^2.$$

(The first estimate follows from (6.7) and the second is from (6.13) and the choice of η). This proves quadratic convergence of RQI.

For the Chebyshev case we expand the series to second order. We denote $\boldsymbol{L}_{\boldsymbol{x}\boldsymbol{x}}, \boldsymbol{L}_{\boldsymbol{x}\boldsymbol{\lambda}}, \boldsymbol{L}_{\boldsymbol{\lambda}\boldsymbol{x}}, \boldsymbol{L}_{\boldsymbol{\lambda}\boldsymbol{\lambda}}$ to be tensors representing higher order partial derivatives of $\boldsymbol{L}$. Put $(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i)) = \hat{\boldsymbol{x}}_i$ we have

$$\begin{gathered}0 = \boldsymbol{L}(\boldsymbol{v}, \boldsymbol{\mu}) = \boldsymbol{L}(\hat{\boldsymbol{x}}_i) + \boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{v} - \boldsymbol{x}_i) + \boldsymbol{L}_{\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{\mu} - \mathcal{R}(\boldsymbol{x}_i)) \\ + \frac{1}{2}\boldsymbol{L}_{xx}(\hat{\boldsymbol{x}}_i)(\boldsymbol{v} - \boldsymbol{x}_i)^{[2]} + \boldsymbol{L}_{\boldsymbol{x}\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)[(\boldsymbol{v} - \boldsymbol{x}_i)(\boldsymbol{\mu} - \mathcal{R}(\boldsymbol{x}_i))] + \\ \frac{1}{2}\boldsymbol{L}_{\boldsymbol{\lambda}\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{\mu} - \mathcal{R}(\boldsymbol{x}_i))^{[2]} + O(||\boldsymbol{v} - \boldsymbol{x}_i||^3)\end{gathered}$$

The residual is $O(||\boldsymbol{v} - \boldsymbol{x}_i||^3)$ in the above because we apply the bound $||\boldsymbol{\mu} - \mathcal{R}(\boldsymbol{x}_i)|| \leq C_1||\boldsymbol{v} - \boldsymbol{x}_i||$ similar to the Rayleigh case. Again, move the term $\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{v} - \boldsymbol{x}_i)$ to the left-hand side then apply $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)$:

$$\begin{gathered}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{x}_i - \boldsymbol{v}) = \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}(\hat{\boldsymbol{x}}_i) + \frac{1}{2}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{xx}(\hat{\boldsymbol{x}}_i)(\boldsymbol{v} - \boldsymbol{x}_i)^{[2]} + \\ \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)[(\boldsymbol{v} - \boldsymbol{x}_i)(\mu - \mathcal{R}(\boldsymbol{x}_i))] + \frac{1}{2}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{\lambda}\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)(\mu - \mathcal{R}(\boldsymbol{x}_i))^{[2]} + O(||\boldsymbol{v} - \boldsymbol{x}_i||^3)\end{gathered} \tag{6.14}$$

We will choose the next iteration $\boldsymbol{x}_{i+1}$ by choosing η_*, $\boldsymbol{T}[\eta_*]$ and τ such that

$$\boldsymbol{x}_{i+1} = \boldsymbol{r}(\boldsymbol{x}_i, \tau) = \boldsymbol{x}_i + \tau + O(||\tau||^3)$$

and

$$\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)\tau = \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)(\eta_* + \boldsymbol{T}[\eta_*^{[2]}])$$

Using the two expressions below, substitute in (6.14):

$$\boldsymbol{x}_i - \boldsymbol{v} = (\boldsymbol{x}_{i+1} - \boldsymbol{v}) - \tau - O(||\tau||^3)$$

$$\boldsymbol{\mu} - \mathcal{R}(\boldsymbol{x}_i) = (\boldsymbol{\mu} - \mathcal{R}(\boldsymbol{x}_{i+1})) + \boldsymbol{J}_{\mathcal{R}}(\hat{\boldsymbol{x}}_i)\tau + O(||\tau||^2)$$

Expand and collect the terms with a factor of $\boldsymbol{v} - \boldsymbol{x}_{i+1}$ or $(\boldsymbol{\mu} - \mathcal{R}(\boldsymbol{x}_{i+1}))$ to a group A, while leaving the terms with only τ factors, together with any cubic term in the expression we get

$$\begin{gathered}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{x}_{i+1} - \boldsymbol{v}) - \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)\eta_* - \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)\boldsymbol{T}[\eta_*^{[2]}] = A + \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}(\hat{\boldsymbol{x}}_i) + \\ \frac{1}{2}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{xx}(\hat{\boldsymbol{x}}_i)\tau^{[2]} + \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)[\tau\boldsymbol{J}_{\mathcal{R}}\tau] + \frac{1}{2}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{\lambda}\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)[(\boldsymbol{J}_{\mathcal{R}}\tau)^{[2]}] \\ + O(||\boldsymbol{v} - \boldsymbol{x}_i||^3) + O(||\tau||^3)\end{gathered} \tag{6.15}$$

We do not need to list the terms of A explicitly. It is sufficient to know that A is sum of terms of total order two in term of τ or $(\boldsymbol{v} - \boldsymbol{x}_{i+1})$ or $(\boldsymbol{\mu} - \mathcal{R}(\boldsymbol{x}_{i+1}))$ and containing at least one term of the last two forms. If $\tau = O(||\boldsymbol{x}_i - \boldsymbol{v}||)$, the continuous derivative assumption shows:

$$||A|| \leq D||\boldsymbol{x}_{i+1} - \boldsymbol{v}|| \times ||\boldsymbol{x}_i - \boldsymbol{v}||$$

for some constant D near $(\boldsymbol{v}, \boldsymbol{\mu})$. Thus $||\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{x}_{i+1} - \boldsymbol{v}) - A||$ is dominated by $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{x}_{i+1} - \boldsymbol{v})$, therefore:

$$||\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{x}_{i+1} - \boldsymbol{v}) - A|| \leq C_2||\boldsymbol{x}_{i+1} - \boldsymbol{v}||$$

with $||C_2|| > 0$, hence we get an estimate for $\boldsymbol{x}_{i+1} - \boldsymbol{v}$. To eliminate the first order terms we choose η_* such that

$$\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)\eta_* + \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}(\hat{\boldsymbol{x}}_i) = 0$$

To eliminate the second order terms we need:

$$\begin{gathered}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)\boldsymbol{T}\eta_*^{[2]} + \frac{1}{2}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{xx}(\hat{\boldsymbol{x}}_i)(\eta_*) + \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)[\eta_*\boldsymbol{J}_{\mathcal{R}}\eta_*] + \\ \frac{1}{2}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{\lambda}\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{J}_{\mathcal{R}}\eta_*)^{[2]} = 0\end{gathered}$$

We note if η_* is $O(||\boldsymbol{x}_i-\boldsymbol{v}||)$ and $\boldsymbol{T}\eta_*^{[2]}$ is also $O(||\boldsymbol{x}_i-\boldsymbol{v}||)$, as long as we project $\tau_*=\eta_*+\boldsymbol{T}\eta_*^{[2]}$ smoothly to the tangent space, $\tau=O(||\boldsymbol{x}_i-\boldsymbol{v}||)$. We assume the last condition in the statement of the theorem. If both η_* and $\boldsymbol{T}\eta_*^{[2]}$ are on the tangent space then by (6.7) they are $O(||\boldsymbol{x}_i-\boldsymbol{v}||)$, and so is τ. We state our theorem slightly more generally, when η_* may not be on the tangent space but ultimately τ is.

From our argument on the dominance of $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{x}_{i+1}-\boldsymbol{v})$ over A cubic convergence follows. To show the Schur form provides a solution, we just need to substitute, noting $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)\zeta=-\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)=0$.

If $\boldsymbol{G}(\boldsymbol{v})=0$, we need to show the RQI step alone has cubic convergence. Instead of (6.15), we have:

$$\begin{aligned}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{x}_{i+1}-\boldsymbol{v}) = A + \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}(\hat{\boldsymbol{x}}_i) + \frac{1}{2}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{xx}(\hat{\boldsymbol{x}}_i)\eta^{[2]}+\\ \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)[\eta\boldsymbol{J}_{\mathcal{R}}\eta] + \frac{1}{2}\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{\lambda}\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)(\boldsymbol{J}_{\mathcal{R}}\eta)^{[2]}+\\ \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)\eta + O(||\boldsymbol{v}-\boldsymbol{x}_i||^3) + O(||\eta||^3)\\ = A + \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}(\hat{\boldsymbol{x}}_i) + \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{L}_{\boldsymbol{x}}(\hat{\boldsymbol{x}}_i)\eta - \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{G}(\hat{\boldsymbol{x}}_i)\eta^{[2]} + O(||\boldsymbol{v}-\boldsymbol{x}_i||^3) + O(||\eta||^3)\end{aligned} \tag{6.16}$$

Since $\boldsymbol{G}$ has continuous derivative and $\boldsymbol{G}(\boldsymbol{v})=0$:

$$||\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{G}(\hat{\boldsymbol{x}}_i)\eta^{[2]}|| = ||(\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\hat{\boldsymbol{x}}_i)\boldsymbol{G}(\hat{\boldsymbol{x}}_i) - \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\boldsymbol{v})\boldsymbol{G}(\boldsymbol{v}))\eta^{[2]}|| \leq C_{\boldsymbol{G}}||\hat{\boldsymbol{x}}_i-\boldsymbol{v}||(||\eta||^2)$$

for some constant $C_{\boldsymbol{G}}$ and from here cubic convergence of RQI follows.

For the explicit case, we just need to substitute $\boldsymbol{L}(\boldsymbol{x},\boldsymbol{\lambda})=\boldsymbol{F}(\boldsymbol{x})-\boldsymbol{H}(\boldsymbol{x})\boldsymbol{\lambda}$ and simplify the algebraic expressions. □

The theorem shows there much freedom in choosing $\mathcal{R}$. The main requirement is consistency: if $(\boldsymbol{v},\boldsymbol{\mu})$ is a solution then $\mathcal{R}(\boldsymbol{v})=\boldsymbol{\mu}$. So even for the explicit case $\mathcal{R}$ does not need to come from a left inverse. We numerically tested this general RQI with $\boldsymbol{L}$ is given by a (tensor) Taylor series up to degree 3 and constraint given by products of spheres. For each sphere, a Rayleigh functional is given by $p_i(\boldsymbol{x})\boldsymbol{L}(\boldsymbol{x},\boldsymbol{\lambda})$, where p_i is the projection to coordinate of the i^{th} sphere. We get convergence as expected.

Algorithm 6.1 Rayleigh quotient iteration for explicit Lagrangians in Schur form

```
Initialize $\boldsymbol{x}_0$
$i \leftarrow 0$
$\zeta \leftarrow SMALL\_NUMBER$
$err \leftarrow LARGE\_NUMBER$
while not Terminal$(i, \zeta, err)$ do
    Compute $\boldsymbol{\lambda}_i = \mathcal{R}(\boldsymbol{x}_i) = \boldsymbol{H}^{-}(\boldsymbol{x}_i)\boldsymbol{F}(\boldsymbol{x}_i)$
    Solve for $\zeta, \nu$ in $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i,\boldsymbol{\lambda}_i)\begin{bmatrix}\zeta, & \nu\end{bmatrix} = \begin{bmatrix}\boldsymbol{H}(\boldsymbol{x}_i), & \boldsymbol{F}(\boldsymbol{x}_i)\end{bmatrix}$
    Compute $\boldsymbol{\lambda}_* = (\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}(\boldsymbol{J}_{\boldsymbol{C}}\nu)$
    Compute $\eta \leftarrow -\nu + \zeta\boldsymbol{\lambda}_*$
    Compute $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{r}(\boldsymbol{x}_i, \eta)$
    Compute $err \leftarrow \boldsymbol{L}(\boldsymbol{x}_{i+1}, \boldsymbol{\lambda}_i)$
    $i \leftarrow i + 1$
end while
```

If not using Schur form, we can solve the system directly as a linear system from $T\mathcal{M}_{\boldsymbol{x}_i}$ to $\mathrm{Im}(\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}})=\mathrm{Null}(\boldsymbol{L}_{\boldsymbol{\lambda}}{}^{-})$. In the constrained optimization case, these three spaces are identical, and we have the Riemannian Newton updating equation. In general, we could use different bases for $\mathrm{Null}(\boldsymbol{J}_{\boldsymbol{C}})$ and $\mathrm{Null}(\boldsymbol{L}_{\boldsymbol{\lambda}}{}^{-})$ to represent $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}$. We will revisit this in the example of the eigenvector problem where we construct four RQIs and four Rayleigh-Chebyshev iterations for two different choices of $\boldsymbol{H}^{-}$, and solve the iteration equations in both Schur form and tangent form. We note an important feature in the proof is the use of $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}$ to eliminate the term $\boldsymbol{L}_{\boldsymbol{\lambda}}(\hat{\boldsymbol{x}}_i)(\mathcal{R}(\boldsymbol{x}_i)-\mathcal{R}(\boldsymbol{v}))$. If we look at the Schur form,

Algorithm 6.2 Rayleigh-Chebyshev iteration for explicit Lagrangians in Schur form

```
Initialize $\boldsymbol{x}_0$
$i \leftarrow 0$
$\zeta = SMALL\_NUMBER$
while not Terminal($i, \eta, err$) do
    Compute $\boldsymbol{\lambda}_i = \mathcal{R}(\boldsymbol{x}_i) = \boldsymbol{H}^{-}(\boldsymbol{x}_i)\boldsymbol{F}(\boldsymbol{x}_i)$
    Solve for $\zeta, \nu$ in $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\begin{bmatrix}\zeta, & \nu\end{bmatrix} = \begin{bmatrix}\boldsymbol{H}(\boldsymbol{x}_i), & \boldsymbol{F}(\boldsymbol{x}_i)\end{bmatrix}$
    Compute $\boldsymbol{\lambda}_* = (\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}(\boldsymbol{J}_{\boldsymbol{C}}\nu)$
    Compute $\eta_* \leftarrow -\nu + \zeta\boldsymbol{\lambda}_*$
    Compute $\boldsymbol{T}\eta_*^{[2]}$ as solution to $\boldsymbol{L}_{\boldsymbol{x}}\boldsymbol{T}\eta_*^{[2]} = -[\frac{1}{2}\boldsymbol{J}_F^{(2)}\eta_*^{[2]} - (\boldsymbol{J}_{\boldsymbol{H}}\eta_*)(\boldsymbol{J}_{\mathcal{R}}\eta_*) - \frac{1}{2}\boldsymbol{J}_{\boldsymbol{H}}^{(2)}\eta_*^{[2]}\mathcal{R}]$
    Compute $\tau_* = \eta_* + T\eta_*^{[2]}$
    Compute $\tau = \tau_* - \zeta(\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}\boldsymbol{J}_{\boldsymbol{C}}(\tau_*)$
    Compute $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{r}(\boldsymbol{x}_i, \tau)$, $\boldsymbol{r}$ is the retraction
    Compute $err \leftarrow \boldsymbol{L}(\boldsymbol{x}_{i+1}, \boldsymbol{\lambda}_i)$
    $i \leftarrow i + 1$
end while
```

besides solving $\boldsymbol{L}_{\boldsymbol{x}}$ for ν and ζ, we also need to compute $\boldsymbol{\lambda}_* = (\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}\boldsymbol{J}_{\boldsymbol{C}}\nu$ which could be an expensive computation depending on the manifold. The tensor eigenpairs example below will also illustrate clearly the work needed to apply each approach.

Another way to get a left inverse of $\boldsymbol{J}_{\boldsymbol{C}}$ is to use a different metric on the ambient space. In [19], the authors considered the metric choice as a preconditioning technique. However, their discussion focuses more on the case of a quotient manifold.

Remark 6.3.

- When $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{P}(\boldsymbol{\lambda})\boldsymbol{x}$ as in the classical and generalized eigenvalue problems (but we allow E_L to have dimension higher than 1), we have

$$\xi = \boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda})^{-1}\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{x}$$

  So $\boldsymbol{x} + \eta = \zeta\theta_*$. In particular, if E_L is of dimension 1 then $\boldsymbol{x} + \eta$ is proportional to $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda})^{-1}\boldsymbol{x}_i$, regardless of the constraint. In general, this is not the case.
- Note $-\nu + \zeta\boldsymbol{\lambda}_i$ as an iteration step does not work for the eigenvalue problem: it simplifies to $-\boldsymbol{x}$. We have also tested for a linear $\boldsymbol{F}$ and a linear constraint $\boldsymbol{C}$, using $\boldsymbol{\lambda}_i$ as a projection to the tangent space only gives linear convergence. So, in general, the theorem requires $\boldsymbol{\lambda}_*$. $\boldsymbol{\lambda}_*$ is more expensive when the codimension is high.

7. Examples.

The python/Matlab codes for these examples could be found at [20]. The Matlab codes are for the tensor eigenpair problem, while the python codes are set up for a general framework. If the ambient Euclidean spaces are vectors, the users only need to provide the functions, constraints and associated derivatives. If the ambient spaces are matrices or higher tensor the users need to provide their methods. These library methods are provided as a guide, we also implemented example-specific versions of the algorithm below.

For the library, we consulted [5] and [26]. To call the functions the user needs to specify the constraint in a constraint object and the function as well as its partial derivatives in a Lagrangian object. The solver is mainly of Schur form, but we also show the tangent form for Stiefel manifold. We also need a solver for $(\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}\boldsymbol{J}_{\boldsymbol{C}}\nu$ in some cases. For left inverse, we use $\boldsymbol{H}^{-} = (\boldsymbol{H}^{\dagger}\boldsymbol{H})^{-1}\boldsymbol{H}^{\dagger}$ for $\boldsymbol{H}^{\dagger}$ such that $(\boldsymbol{H}^{\dagger}\boldsymbol{H})$ is invertible. We use different choices of $\boldsymbol{H}^{\dagger}$ in our examples. Our aim is to verify the result, so we have not spent much effort to optimize the code.

7.1. RQI on the sphere I: Real tensor eigenpairs.

We will focus on symmetric tensors here. As mentioned in the introduction, the tensor eigenpair problem look to solve:

$$\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \mathcal{T}(\boldsymbol{I}, \boldsymbol{x}, \cdots, \cdots \boldsymbol{x}) - \boldsymbol{\lambda}\boldsymbol{x} = 0$$

For the real case, we impose the constraint $\boldsymbol{C}(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x}^T\boldsymbol{x} - 1) = 0$ where $\mathcal{T}$ is a tensor of order $m \geq 3$. With $\boldsymbol{F}(\boldsymbol{x}) = \mathcal{T}(\boldsymbol{I}, \boldsymbol{x}, \cdots, \cdots \boldsymbol{x})$ and $\boldsymbol{H}(\boldsymbol{x}) = \boldsymbol{x}$, $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda}) = (m-1)\mathcal{T}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{x}, \cdots, \boldsymbol{x}) - \boldsymbol{\lambda}\boldsymbol{I}$ and the RQI updating equation (6.1) becomes:

$$(7.1) \qquad (\boldsymbol{I} - \boldsymbol{x}\boldsymbol{x}^T)((m-1)\mathcal{T}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{x}, \cdots, \boldsymbol{x}) - \boldsymbol{\lambda}\boldsymbol{I})\eta = -(\boldsymbol{I} - \boldsymbol{x}\boldsymbol{x}^T)\mathcal{T}(\boldsymbol{I}, \boldsymbol{x}, \boldsymbol{x}, \cdots, \boldsymbol{x})$$

We note $(1 - \boldsymbol{x}\boldsymbol{x}^T)\boldsymbol{x} = 0$, so the right-hand side could also be written:

$$-(\boldsymbol{I} - \boldsymbol{x}\boldsymbol{x}^T)(\mathcal{T}(\boldsymbol{I}, \boldsymbol{x}, \boldsymbol{x}, \cdots, \boldsymbol{x}) - \boldsymbol{\lambda}\boldsymbol{x})$$

If we choose to solve this equation by parameterizing $T\mathcal{M}$, we arrive at algorithm O-NCM in [16]: the algorithm find the null space of $\boldsymbol{x}^T$, the projected Hessian $H_p = (\boldsymbol{I} - \boldsymbol{x}\boldsymbol{x}^T)\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda})(1 - \boldsymbol{x}\boldsymbol{x}^T)$ to get the updating vector $\boldsymbol{u} = \eta$. This method has the additional cost of transferring the equation to the tangent space and back. If we choose to solve the equation by the Schur complement method, we need to solve for ν, ζ with

$$\begin{aligned} \boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\nu =& \mathcal{T}(\boldsymbol{I}, \boldsymbol{x}, \cdots, \boldsymbol{x}) \\ \boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\zeta =& \boldsymbol{x} \end{aligned}$$

and the iteration step is $\eta = -\nu + (\boldsymbol{x}^T\zeta)^{-1}(\boldsymbol{x}^T\nu)\zeta = -\nu + \zeta\boldsymbol{\lambda}_*$. By our main theorem of RQI, our η is the same as $\boldsymbol{u}$ in O-NCM, but this method turns out to be 16 percent faster than O-NCM in our python/MATLAB implementations, which the reader can see from our notebook *EigenTensor.ipynb* on our GitHub page [20]. The main cost of tensor eigenpair problem may be in evaluating the tensor itself, as when we simplify the code to remove one extra tensor calculation, we get a total 34 percent improvement over O-NCM. The number of iterations and step values mostly agree between the three implementations (in practice there are some discrepancies for a number of initial points after some steps). The following table summarizes the run result.

ONCM vs Schur form RQI			
	ONCM	Schur RQI (raw)	Schur RQI (simplified)
Avg time	0.000550	0.000460	0.000363
Improv. v.s. ONCM		0.163196	0.340280

The Schur complement method has an advantage when we have a small number of constraints: the dimension of the ambient space is not much higher than that of the constraint manifold for the inversion of $\boldsymbol{L}_{\boldsymbol{x}}$ to be expensive versus that of projected $\boldsymbol{L}_{\boldsymbol{x}}$, while we can avoid performing the projection and recovering operations directly.

As pointed out earlier Schur form RQI uses the step $-\nu + \zeta\boldsymbol{\lambda}_*$ with $\boldsymbol{\lambda}_* = (\boldsymbol{x}^T\zeta)^{-1}\boldsymbol{x}^T\nu$ which guarantees the iterative step is on the tangent space, which is the main difference with NCM of [16] where the step $\boldsymbol{y} = -\nu + \zeta\boldsymbol{\lambda}$ is not on the tangent space.

7.2. RQI on the sphere II: Complex tensor eigenpairs. The number of real eigenpairs is dependent on the tensor under consideration, so there is no formula to count real eigenpairs. As pointed out by [16], the homotopy method of [9] is capable of computing all real eigenvalues but it is much slower: to run an $(m = 4, n = 8)$ tensor, it takes several hours.

As it often happens in algebra, the eigenpair count is simpler in the complex case: if $\mathcal{T}$ is a tensor of dimension n and order m, for complex eigenpairs, the number of eigenpairs is given by a simple formula:

$$\begin{aligned} ((m-1)^n - 1)/(m-2) &\text{ if } m > 2 \\ n &\text{ if } m = 2 \end{aligned}$$

according to [8]. Let us recall what this number means based on the exposition in [17]. We still consider the equation:

$$\boldsymbol{L}(\boldsymbol{z}, \boldsymbol{\lambda}) = \mathcal{T}(\boldsymbol{I}, \boldsymbol{z}, \cdots, \cdots \boldsymbol{z}) - \boldsymbol{\lambda}\boldsymbol{z} = 0$$

with $\boldsymbol{z}$ a complex vector. It is easy to see that if $(\boldsymbol{\lambda}, \boldsymbol{z})$ is an eigenpair then $(t^{m-2}\boldsymbol{\lambda}, t\boldsymbol{z})$ is another eigenpair, for any non zero $t \in \mathbb{C}$. We consider any two such pairs as equivalent. The count given by [8] is the count of equivalent pairs in this sense. We note if $m > 2$ then $t = \exp(-\frac{\text{angle}(\boldsymbol{\lambda})\sqrt{-1}}{m-2})$ would make $t^{m-2}\boldsymbol{\lambda}$ real (and nonnegative, see discussion below). Therefore, we can always assume $\boldsymbol{\lambda}$ is real. For each real $\boldsymbol{\lambda}$, $(\boldsymbol{\lambda}, \boldsymbol{z})$ and $(\boldsymbol{\lambda}, t\boldsymbol{z})$ are equivalent if $t^{m-2} = 1$. We will not address the case of zero eigenvalues here.

Since $\boldsymbol{\lambda}$ is real, we can assume E_L to have dimension 1 in our framework, and treat $\boldsymbol{z}$ as a real vector of dimension $2n$. The unitary constraint $\boldsymbol{C}(\boldsymbol{z}) = \boldsymbol{z}^*\boldsymbol{z} - 1$ is a convenient one, extending the real case. Here, $*$ is the Hermitian conjugate. It is clear now $\boldsymbol{J_C}(\eta) = 2\text{Re}(\boldsymbol{z}^*\eta)$. The Lagrangian is explicit in this case with $\boldsymbol{H}(\boldsymbol{z})(\boldsymbol{\lambda}) = \boldsymbol{z}\boldsymbol{\lambda}$. A left inverse $\boldsymbol{H}^-(\boldsymbol{z})$ is a map from $E_{out} = \mathbb{C}^n$ to $E_L = \mathbb{R}$, we choose it to be:

$$\boldsymbol{H}^-(\boldsymbol{z})\boldsymbol{w} = \text{Re}(\boldsymbol{z}^*\boldsymbol{w})$$

The fact that this is a left inverse follows from $\boldsymbol{z}^*\boldsymbol{z} = 1$. With this $\boldsymbol{H}^-$, we have the following algorithm:

Algorithm 7.1 Rayleigh quotient iteration for complex tensor eigenpairs

```
Initialize $\boldsymbol{z}_0$
$i \leftarrow 0$
$\zeta \leftarrow SMALL\_NUMBER$
$err \leftarrow LARGE\_NUMBER$
while not Terminal($i, \zeta, err$) do
   Compute $\boldsymbol{\lambda}_i = \mathcal{R}(\boldsymbol{z}_i) = \text{Re}(\boldsymbol{z}_i^*\mathcal{T}(\boldsymbol{I}, \boldsymbol{z}_i \cdots \boldsymbol{z}_i))$
   Solve for $\zeta, \nu$ in $((m-1)\mathcal{T}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{z}_i, \cdots, \boldsymbol{z}_i) - \boldsymbol{\lambda}_i\boldsymbol{I}) \begin{bmatrix}\zeta, & \nu\end{bmatrix} = \begin{bmatrix}\boldsymbol{z}_i & \mathcal{T}(\boldsymbol{I}, \boldsymbol{z}_i, \cdots, \boldsymbol{z}_i)\end{bmatrix}$
   Compute $\boldsymbol{\lambda}_* = \text{Re}(\boldsymbol{z}_i^*\zeta))^{-1}\text{Re}(\boldsymbol{z}_i^*\nu)$
   Compute $\eta \leftarrow -\nu + \zeta\boldsymbol{\lambda}_*$
   Compute $\boldsymbol{z}_{i+1} \leftarrow (\boldsymbol{z}_i + \eta)/||\boldsymbol{z}_i + \eta||$
   Compute $err \leftarrow \boldsymbol{L}(\boldsymbol{z}_{i+1}, \boldsymbol{\lambda}_i)$
   $i \leftarrow i + 1$
end while
```

This RQI has quadratic convergence rate and is a simple modification of the real case. While a detailed global convergence analysis is still needed, in practice we get a reasonably fast algorithm to compute all complex eigenpairs: if the number of complex pairs given by the Cartwright-Sturmfels formula is in the range of a thousand pairs, our program completes in a few minutes. The case $(m = 4, n = 8)$ with 3280 pairs is done within 15 minutes; $m = 3, n = 12$ (4095 pairs) we are done within half an hour, where the bulk of the search is done within a few minutes, the slow convergence was due to the last 10 percent of the pairs.

Similar to the real case, we choose a random point $\boldsymbol{z}_0$ on the unitary sphere $\boldsymbol{z}_0^*\boldsymbol{z}_0 = 1$ then start the iteration. The process, therefore, could be parallelized easily as noted in [16]. To make sure we only count distinct pairs under the equivalent relation, we keep a table tracking all eigenpairs we have found within our search and insert a new vector only if it is not proportional to an existing one. If a vector is found, we also check to see if it is equivalent to its conjugate ($u\bar{\boldsymbol{z}} = \boldsymbol{z}$ with $u = \boldsymbol{z}^T\boldsymbol{z}$.) If this is the case, we only insert $\boldsymbol{z}$ to the table, otherwise, we insert both $\boldsymbol{z}$ and $\bar{\boldsymbol{z}}$. We then check to see if the vector could be made real, that is if there is a factor t such that $t\boldsymbol{z}$ is real. If there is such at t, then $-t$ is also a candidate. We try to pick between the two choices of t to make $t^{m-2}\boldsymbol{\lambda}$ positive, which is always possible when m is odd but not always so when m is even. Our script returns a table with eigenpairs explicitly marked as real or not, so this algorithm gives us an effective way to find all real roots without a homotopy run. The following table summarizes the result where we generate 20 random matrix for each configuration (m, n), compute and compute all eigenpairs. We note there is a small number of cases where we have multiple inequivalent eigenvectors corresponding to one eigenvalue. For more complex tensors ($(m = 3, n = 9)$ and $(m = 4, n = 8)$) ninety percent of the eigenpairs are computed within minutes, while the remaining ten percents are responsible for the long search time. As the tensors get more complex, the percentage of real pairs becomes smaller.

Test runs for Schur RQI for all complex eigenpairs

		n_trys	n_pairs	n_real_pairs	n_multiple_eigen	time_90	time_all
m	n						
3	2	20	3	2.3	0.00	0.002145	0.002155
	3	20	7	4.4	0.00	0.001542	0.001542
	4	20	15	9.2	0.00	0.002313	0.002972
	5	20	31	12.8	0.00	0.001752	0.003800
	6	20	63	19.6	0.00	0.001653	0.005972
	7	20	127	34.0	0.00	0.001837	0.012043
	8	20	255	51.5	0.00	0.002060	0.042443
	9	20	511	79.1	0.15	0.002359	0.042205
4	2	20	4	2.8	0.00	0.000962	0.000963
	3	20	13	6.7	0.00	0.001443	0.001724
	4	20	40	15.0	0.00	0.001778	0.003655
	5	20	121	29.9	0.00	0.001552	0.006803
	6	20	364	56.1	0.00	0.002380	0.012324
	7	20	1093	123.4	0.00	0.003644	0.025138
	8	20	3280	227.6	4.45	0.006668	0.083328
5	2	20	5	3.1	0.00	0.000838	0.000839
	3	20	21	8.5	0.00	0.000880	0.001655
	4	20	85	21.9	0.00	0.001447	0.004274
6	2	20	6	3.6	0.00	0.000987	0.000988
	3	20	31	10.2	0.00	0.001119	0.002017
	4	20	156	32.1	0.00	0.002099	0.006781

7.3. Optimization on embedded manifolds. As mentioned, in this case, $\boldsymbol{H} = \boldsymbol{J}_{\boldsymbol{C}}^T$ and the Rayleigh iteration equation is exactly the Riemannian Newton equation. It is clear now that the FP-SQP equations are a special case of the RQI equations as already known from [22]. We would like to point out that chapter 10 of [6] contains several examples where the Schur complement method was used to solve constraint optimization problems, and our result here was very much inspired by those examples. [11] and [12] are also very early results addressing the same topic. We have not seen the Schur form solution, expressed in ζ and ν being discussed before in this context. For an optimization problem with a small number of critical points and number of constraints, we expect Schur form RQI to be an effective optimization method.

7.4. Eigenvectors. First, we consider the eigenvector problem under the quadratic constraint:

$$\frac{1}{2}(\boldsymbol{x}^T\boldsymbol{x} - 1) = 0$$

We tested all explicit Lagrangian algorithms listed here. We obtained convergent for all cases as expected with appropriate initial points. Care needs to be taken in determining when $\boldsymbol{L}_x$ becomes singular as a terminal condition. We already looked at Newton-Raphson for eigenvector in section 2 and have showed $\boldsymbol{x}_{i+1}$ is proportional to ζ. However, the second derivative

$$\begin{pmatrix} -2\eta\delta \\ \eta^T\eta \end{pmatrix}$$

is not dependent on A, so section 2 does not have cubic convergence even for normal matrices. Applying (5.2), the Chebyshev adjustment involves one more matrix inversion $(A - \boldsymbol{\lambda I})^{-1}\eta$ and is simple to calculate.

We have shown our RQI is the classical RQI in this case. With Rayleigh quotient $(\boldsymbol{x}^T\boldsymbol{x})^{-1}\boldsymbol{x}^T A\boldsymbol{x}$

$$\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{x})(\eta) = -(\boldsymbol{x}^T\eta + \eta^T\boldsymbol{x})\boldsymbol{x}^T A\boldsymbol{x} + \eta^T A\boldsymbol{x} + \boldsymbol{x}^T A\eta$$

When A is normal $\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{v})=0$ hence $\boldsymbol{G}(\boldsymbol{v})=0$, therefore we have cubic convergence.

Let us illustrate the different approaches by consider the eigenvalue problem with the constraint:

$$\boldsymbol{u}^T\boldsymbol{x}=1$$

We construct different iterations under two different left inverses summarized in the following table:

$\boldsymbol{H}^-(\boldsymbol{x})$	$\Pi_{\boldsymbol{H}}$	λ	$\boldsymbol{J}_{\mathcal{R}}\eta$	$\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}$
$(\boldsymbol{x}^T\boldsymbol{x})^{-1}\boldsymbol{x}$	$\boldsymbol{I}-\boldsymbol{x}\boldsymbol{x}^T(\boldsymbol{x}^T\boldsymbol{x})^{-1}$	$(\boldsymbol{x}^T\boldsymbol{x})^{-1}\boldsymbol{x}^TA\boldsymbol{x}$	$(\boldsymbol{x}^T\boldsymbol{x})^{-1}(\boldsymbol{x}^T(A+A^T)-2\lambda\boldsymbol{x}^T)\eta$	$U^T(A-\lambda)U$
$\boldsymbol{z}^T$	$\boldsymbol{I}-\boldsymbol{x}\boldsymbol{z}^T$	$\boldsymbol{z}^TA\boldsymbol{x}$	$\boldsymbol{z}^TA\eta$	$V^T(A-\lambda)U$

We note $\zeta=(A-\boldsymbol{\lambda})^{-1}\boldsymbol{x}$ and $\nu=\boldsymbol{x}+\zeta\boldsymbol{\lambda}$, saving us one matrix inversion for the Schur form. Also $\boldsymbol{\lambda}_*=(\boldsymbol{z}^T\nu)/(\boldsymbol{z}^T\zeta)$ in both cases. For the tangent form, the tangent space could be parametrized by a matrix U whose columns form an orthonormal basis of $\mathrm{Null}(\boldsymbol{z}^T)$. Let V be an orthonormal basis of $\mathrm{Null}(\boldsymbol{x}^T)$, we can represent $\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}$ as in the above table for the tangent form iteration. The Chebyshev step is $(\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}})^{-1}(\Pi_{\boldsymbol{H}}\eta)\boldsymbol{J}_{\mathcal{R}}\eta$. The readers can find the code in the workbook Eigen.ipynb.

For the nonnormal case, the Chebyshev term is again proportional to $(A-\lambda\boldsymbol{I})^{-1}\eta$. However, the Schur form iteration becomes unstable in this case, the matrix $(A-\lambda\boldsymbol{I})$ is singular when we approach an eigenvalue, and it makes the Chebyshev term unstable. However, the tangent form works: the number of iterations for tangent form Chebyshev is lower than that of tangent RQI, but not by much. Time-wise, we see Chebyshev occasionally offers some improvement but barely so. The best time performance is still the classical RQI on ambient space. The following table summarizes the run result for a 500 runs of a matrix of size 50:

	xtx tgt	xtx chev tgt	xtx schur	xtx chev schur	ztx tgt	ztx chev tgt	ztx schur	ztx chev schur
avg. iter	6.5767	6.0911	6.5538	40.4059	9.6913	9.3579	9.6980	62.0247
avg. time	0.0039	0.0038	0.0013	0.0129	0.0027	0.0029	0.0019	0.0181

7.5. Two-sided Rayleigh quotient.

This algorithm by Ostrowski [21] has cubic convergence rate even for nonnormal matrix A. We show this could be derived from our Lagrangian approach. We will focus on the complex spaces, considered as real spaces of twice the dimension. Since the eigenvalue $\boldsymbol{\lambda}$ is in $\mathbb{C}$ of real dimension 2, we would expect to have two constraints. We will use $\boldsymbol{z}$ in place of $\boldsymbol{x}$ in the Hermitian case as above. When dealing with derivatives, we use $\boldsymbol{z}$ and $\boldsymbol{z}^*$ instead of real and imaginary parts to simplify the calculations. Let A be a square, complex matrix. Define:

$$\boldsymbol{z}=\begin{pmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{pmatrix}$$

We thus define E_{in}, E_{out} to be $\mathbb{C}^{2n}\cong\mathbb{R}^{4n}$ while E_L is $\mathbb{C}\cong\mathbb{R}^2$.

$$\boldsymbol{F}(\boldsymbol{z})=\begin{pmatrix}0 & A^*\\ A & 0\end{pmatrix}\begin{pmatrix}\boldsymbol{u}\\ \boldsymbol{v}\end{pmatrix}=\begin{pmatrix}A^*\boldsymbol{v}\\ A\boldsymbol{u}\end{pmatrix}$$

$$\boldsymbol{H}(\boldsymbol{z})(\boldsymbol{\lambda})=\begin{pmatrix}\boldsymbol{v}\boldsymbol{\lambda}^*\\ \boldsymbol{u}\boldsymbol{\lambda}\end{pmatrix}$$

$$\boldsymbol{C}(\boldsymbol{z})=\begin{pmatrix}\boldsymbol{v}^*\boldsymbol{v}-1\\ \boldsymbol{u}^*\boldsymbol{u}-1\end{pmatrix}$$

The corresponding Lagrangian gives us the left and right eigenvectors for the matrix A. To define the RQI, we use $\boldsymbol{H}^-=(\boldsymbol{H}^\dagger\boldsymbol{H})^{-1}\boldsymbol{H}^\dagger$ with $\boldsymbol{H}^\dagger$ is the map from $\mathbb{C}^{2n}$ to $\mathbb{C}$ defined by $\boldsymbol{H}^\dagger(\boldsymbol{z})\begin{pmatrix}a\\ b\end{pmatrix}=\boldsymbol{v}^*b$. We recover Ostrowski's algorithm from:

$$\boldsymbol{H}^\dagger\boldsymbol{H}(\boldsymbol{z})(\boldsymbol{\mu})=\boldsymbol{v}^*\boldsymbol{u}\boldsymbol{\mu}\text{ for }\boldsymbol{\mu}\in E_L\cong\mathbb{C}$$

$$\mathcal{R}(\boldsymbol{z}) = (\boldsymbol{v}^* A\boldsymbol{u})/(\boldsymbol{v}^*\boldsymbol{u})$$

$$\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{z}_*)\begin{pmatrix}\eta\\ \phi\end{pmatrix} = (\eta^* A\boldsymbol{u} + \boldsymbol{v}^* A\phi)(\boldsymbol{v}^*\boldsymbol{u})^{-1} - (\boldsymbol{v}^* A\boldsymbol{u})(\eta^*\boldsymbol{u} + \boldsymbol{v}^*\phi)(\boldsymbol{v}^*\boldsymbol{u})^{-2} = 0$$

at the eigenvector $\boldsymbol{z}_e = \begin{pmatrix}\boldsymbol{u}_e\\ \boldsymbol{v}_e\end{pmatrix}$, using $\boldsymbol{v}_e^* A = \boldsymbol{\lambda}\boldsymbol{v}_e$, $A\boldsymbol{u}_e = \boldsymbol{\lambda} u_e$. Cubic convergence of two-sided Rayleigh follows from the main theorem. See [4] for an invariant subspace version.

7.6. General Lagrangian: Nonlinear eigenvalue problem. Newton-Raphson method was applied to the nonlinear eigenvalue problem (also called λ-matrix problem) in [14]. Again, we will be working with the complex case. With $\boldsymbol{\lambda} = (\lambda)$ is a scalar, the equation has the form

$$\boldsymbol{L}(\boldsymbol{z}, \lambda) = \boldsymbol{P}(\lambda)\boldsymbol{z}$$

Applying Algorithm 5.1 we have $\boldsymbol{L}_{\boldsymbol{z}} = \boldsymbol{P}(\lambda)$, $\boldsymbol{L}_{\lambda} = \boldsymbol{P}_{\lambda}(\lambda)\boldsymbol{z}$ and the Schur complement is either $\boldsymbol{z}^T\boldsymbol{P}^{-1}(\lambda)\boldsymbol{P}_{\lambda}(\lambda)\boldsymbol{z}$ for the quadratic constraint or $\boldsymbol{z}^T\boldsymbol{P}^{-1}(\lambda)\boldsymbol{P}^{\lambda}(\lambda)\boldsymbol{z}$ for the linear constraint. Algorithm 5.1 reduces to algorithm 4.7 in the above citation.

To recover known results of the nonlinear eigenvalue problem, in particular, algorithm 4.9 in [14], proposed in [24], we first assume the unit sphere constraint, which is now $\frac{1}{2}(\boldsymbol{z}^*\boldsymbol{z}-1)$. We have $\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{z}) = \boldsymbol{z}^*$ and $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{z}, \lambda) = \boldsymbol{P}(\boldsymbol{z})$. Apply Theorem 6.2:

$$\xi = \boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{z}, \lambda)^{-1}\boldsymbol{L}(\boldsymbol{z}, \lambda) = \boldsymbol{z}$$

$$\eta = -\boldsymbol{z} + \zeta(\boldsymbol{z}^*\zeta)^{-1}(\boldsymbol{z}^*\boldsymbol{z})$$

This implies $\boldsymbol{z} + \eta$ is proportional to ζ, exactly like the classical eigenvalue case, as pointed out in Remark 6.3. This means we only need to evaluate ζ. The only missing part is the Rayleigh quotient, which we define to be the root λ of the equation:

$$\mathcal{N}(\boldsymbol{z}, \lambda) := \boldsymbol{z}^*\boldsymbol{P}(\lambda)\boldsymbol{z} = 0$$

We see this Rayleigh quotient is consistent because $\mathcal{N}(\boldsymbol{z}, \lambda) = 0$ is a consequence of $\boldsymbol{P}(\lambda)\boldsymbol{z} = 0$, so the general RQI applies. Note:

$$\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{z}) = -(\boldsymbol{z}^*\boldsymbol{P}_{\lambda}(\lambda)\boldsymbol{z})^{-1}\boldsymbol{z}^*(\boldsymbol{P}(\lambda) + \boldsymbol{P}^*(\lambda))$$

In particular if $\boldsymbol{P}$ is normal $\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{v}) = 0$ for any nonlinear eigenvector ($\boldsymbol{z}^*\boldsymbol{P}(\lambda) = \boldsymbol{z}^*\boldsymbol{P}(\lambda)^* = 0$ in that case). Hence, we have cubic convergence. For the nonnormal case, we can define $\boldsymbol{z} = \begin{pmatrix}u\\ v\end{pmatrix}$ as before and

$$\hat{\boldsymbol{P}} = \begin{pmatrix}0 & \boldsymbol{P}^*\\ \boldsymbol{P} & 0\end{pmatrix}$$

Take $\mathcal{N}(\boldsymbol{z}, \boldsymbol{\lambda}) = v^*\boldsymbol{P}u + u^*\boldsymbol{P}^*v$ we recover the nonlinear two-sided RQI with cubic convergence. Therefore Theorem 6.2 implies the two known algorithms for nonlinear eigenvalue problem.

We also consider the linear constraint $\boldsymbol{C}(\boldsymbol{z}) = \boldsymbol{u}\boldsymbol{z} - 1$, both the RQI and the Chebyshev case. The following table summarizes the result of 500 test runs for a polynomial matrix of size $n = 100$ and degree 4 in λ:

Nonlinear eigenvalue: Two-sided, RQI and Rayleigh Chebyshev			
	Two-sided	RQI	RC
Avg. iter.	5.440000	9.062000	5.720000
Avg. time	0.017428	0.024496	0.020807

We also tested with various sizes and degrees. We find two-sided RQI is the fastest algorithm as expected. Rayleigh Chebyshev is faster than regular RQI but underperforms two-sided RQI. We have to impose a constraint to apply the Chebyshev step only if the RQI step is not too large, otherwise, the algorithm fails to converge.

7.7. Vector Lagrangian with various constraints. We tested the algorithms with two nonlinear constraints. If $\boldsymbol{x}$ is of size n and $\boldsymbol{C}$ consists of k constraints, we assume that it has been solved for the first $n-k$ variables, so $\boldsymbol{x}[n-k+i] = c_i(\boldsymbol{x}[0:n-k])$, with c_i a function of the first $n-k$ variables. Locally any constraint could be transformed into this form. We use the orthographic retraction, which is simpler in this case. The constraint functions we tested are of form:

$$x[n_f] = x[0:n_f] + \sin(x[0:n_f])$$

$$x[n_f+1] = x[0:n_f] + \cos(x[0:n_f])$$

We take $\boldsymbol{H}$ to be either a constant function or a quadratic function. For $\boldsymbol{F}$ we take $\boldsymbol{F}(\boldsymbol{x}) = A\boldsymbol{x}$ or $\boldsymbol{F}(\boldsymbol{x}) = A\boldsymbol{x} + \sin(B\boldsymbol{x})$ for some square matrices A and B. The difficulty is with choosing the initial point, otherwise the algorithms converge sufficiently fast.

7.8. RQI on Stiefel manifolds. The constraint for Stiefel manifolds is $\frac{1}{2}(\boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{I}_p)$. We will focus on the case $\boldsymbol{H} = \boldsymbol{J}_{\boldsymbol{C}}^T$. Here $E_{in} = E_{out} = M_{n,p}$ and E_L is the space of $p\times p$ symmetric matrices. The tangent space is defined by the equation:

$$\boldsymbol{J}_{\boldsymbol{C}}(\eta) = \frac{1}{2}(\boldsymbol{x}^T\eta + \eta^T\boldsymbol{x}).$$

So it could be represented as an average of right and left multiplication tensors. Its conjugate $\boldsymbol{J}_{\boldsymbol{C}}^T$ is the map $\gamma \mapsto \boldsymbol{x}\gamma$. So the Rayleigh quotient turns out to be:

$$\mathcal{R}(\boldsymbol{x}) = \frac{1}{2}(x^T\boldsymbol{F}(\boldsymbol{x}) + \boldsymbol{F}(\boldsymbol{x})^T\boldsymbol{x})$$

and the projection $\Pi_{\boldsymbol{H}}$ is

$$\xi \mapsto \xi - \frac{1}{2}\boldsymbol{x}\boldsymbol{x}^T\xi - \frac{1}{2}\boldsymbol{x}\xi^T\boldsymbol{x}$$

as seen in [3]. We tested the explicit Lagrangian with $\boldsymbol{F}(x) = A\boldsymbol{x} + b$ where A is an (n,p,n,p) tensor and b is an (n,p) matrix. For the Schur form, we need to compute $\zeta = \boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{J}_{\boldsymbol{C}}^T$ and $\nu = \boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{F}(\boldsymbol{x})$. We note ζ is an $(n,p,p(p+1)/2)$ tensor in this case. We form a (sparse) matrix formed by concatenating the vectorized $\boldsymbol{F}(\boldsymbol{x})$ and $\boldsymbol{J}_{\boldsymbol{C}}^T$ (represented as a tensor reshaped as an $(np, p(p+1)/2)$ matrix). That way we can solve for ν and η in the same step. $\boldsymbol{\lambda}_* = (\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}\boldsymbol{J}_{\boldsymbol{C}}\nu$ is a $(p(p+1)/2, p(p+1)/2)$ matrix, so $\zeta\boldsymbol{\lambda}_*$ is an (n,p) matrix. The Schur form requires solving a larger system with dimension np instead of $np - p(p+1)/2$ of the Riemannian Newton method, but if the codimension of the Stiefel manifold is not too big the Schur form could be a useful alternative. We also tested the solution in tangent form.

8. RQI on Grassmann manifolds. Functions on Grassmann manifolds could be considered as function on fixed rank matrices equivariant under right multiplication by invertible matrices, or on Stiefel manifolds equivariant under the orthogonal group. Here we assume $\boldsymbol{H}$ to be the right multiplication by $\boldsymbol{x}^T$. The orthogonal group O_p acts on E_{in}, E_{out} and E_L, generating vector fields on these spaces. Invariance under the action of O_p allows us to identify the tangent space of the Grassmann manifold with the space of $n\times p$ matrices η with $x^T\eta = 0$. The action on E_{out} defines a subspace of $\mathrm{Im}(\Pi_{\boldsymbol{H}})$ orthogonal to the vector fields generated by the action. We call the projection to this space Π_G, which turns out to be $(\boldsymbol{I} - \boldsymbol{x}\boldsymbol{x}^T)$. We arrive at the equations:

$$\begin{aligned} \boldsymbol{L}_{\boldsymbol{x}}\eta &= \boldsymbol{J}_{\boldsymbol{F}}\eta - \eta\boldsymbol{x}^T\boldsymbol{F}(\boldsymbol{x}) \\ \Pi_G\boldsymbol{L}_{\boldsymbol{x}}\eta &= -\Pi_G\boldsymbol{F} \\ \boldsymbol{x}^T\eta &= 0 \end{aligned} \tag{8.1}$$

We can try to solve it in Schur form. $\zeta = \boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{H}$ is now a tensor. In general, it could not be expressed as a matrix multiplication. This is because we do not always have $\boldsymbol{J}_{\boldsymbol{F}}(\boldsymbol{x})(\psi\gamma) = (\boldsymbol{J}_{\boldsymbol{F}}(\boldsymbol{x})\psi)\gamma$ for $n\times p$

matrix ψ and $p \times p$ matrix γ. But when this is the case the Schur form is simpler:

$$\begin{aligned} \zeta &= \boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{x} \\ \nu &= \boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{F}(x) \\ \eta &= -\nu + \zeta(\boldsymbol{x}^T\zeta)^{-1}(\boldsymbol{x}^T\nu) \end{aligned} \tag{8.2}$$

which we can verify directly, using the associativity mentioned above. An important case where we have this associativity is the sphere. Another important case is invariant subspace:

$$\begin{aligned} A\boldsymbol{x} - \boldsymbol{x}\Lambda &= 0 \\ \frac{1}{2}(\boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{I}_p) &= 0 \end{aligned} \tag{8.3}$$

with $\boldsymbol{F}(\boldsymbol{x}) = A\boldsymbol{x}$. The Rayleigh quotient is $\mathcal{R}(\boldsymbol{x}) = \boldsymbol{x}^T A\boldsymbol{x}$. Now if $z = \boldsymbol{x} + \zeta\boldsymbol{\lambda}$ then

$$Az - z\boldsymbol{\lambda} = A\boldsymbol{x}$$

so we can take $\nu = \boldsymbol{x} + \boldsymbol{x}\boldsymbol{\lambda}$, hence:

$$\eta = -\boldsymbol{x} + \zeta(\boldsymbol{\lambda}_* - \mathcal{R}(x))$$

where $\boldsymbol{\lambda}_* = (\boldsymbol{x}^T\zeta)^{-1}(\boldsymbol{x}^T\nu)$. Thus $x_{i+1} = \zeta(\boldsymbol{\lambda}_* - \mathcal{R}(x))$ is in the same space spanned by ζ. So the result of [1] is the Schur form of the Rayleigh quotient algorithm. We are not sure that these are the only examples where ζ is representable as a matrix.

We consider another example, finding critical point for the function:

$$\frac{1}{2}\operatorname{Tr}(\boldsymbol{x}^T L\boldsymbol{x}) + \frac{\alpha}{4}\rho(\boldsymbol{x})^T L^{-1}\rho(\boldsymbol{x})$$

$$\rho(\boldsymbol{x}) = \operatorname{diag}(\boldsymbol{x}\boldsymbol{x}^T)$$

with $\boldsymbol{x}^T\boldsymbol{x} = \boldsymbol{I}_p$ ([27], [5]). We get the gradient

$$\boldsymbol{F}(x) = L\boldsymbol{x} + \alpha\operatorname{diag}(L^{-1}\rho(\boldsymbol{x}))\boldsymbol{x}) \tag{8.4}$$

and we use the $GL(p)$ equivariant form

$$\rho(\boldsymbol{x}) = \operatorname{diag}(\boldsymbol{x}(\boldsymbol{x}\boldsymbol{x}^T)^{-1}\boldsymbol{x}^T) \tag{8.5}$$

$\boldsymbol{F}$ is equivariant under the action of right multiplication. $\boldsymbol{J}_{\boldsymbol{F}}\eta$ is

$$\boldsymbol{J}_{\boldsymbol{F}}\eta = A\eta + 2\alpha\operatorname{diag}(L^{-1}\operatorname{diag}((\boldsymbol{I} - \boldsymbol{x}\boldsymbol{x}^T)\eta\boldsymbol{x}^T))\boldsymbol{x} + \alpha\operatorname{diag}(L^{-1}\rho(\boldsymbol{x}))\eta$$

Because of the middle term, $\boldsymbol{J}_{\boldsymbol{F}}$, ζ cannot be represented as a matrix multiplication. In the code, we computed ζ as a tensor and found critical points as expected.

REFERENCES

[1] P. Absil, R. Mahony, R. Sepulchre, and P. Van Dooren, *A Grassmann–Rayleigh quotient iteration for computing invariant subspaces*, SIAM Rev., 44 (2002), pp. 57–73, https://doi.org/10.1137/S0036144500378648.

[2] P. Absil and J. Malick, *Projection-like retractions on matrix manifolds*, SIAM J. Optim, 22 (2012), pp. 135–158, https://doi.org/10.1137/100802529.

[3] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*, Princeton University Press, Princeton, NJ, USA, 2007.

[4] P.-A. Absil and P. Van Dooren, *Two-sided Grassmann-Rayleigh quotient iteration*, Numer. Math., 114 (2010), pp. 549–571, https://doi.org/10.1007/s00211-009-0266-y.

[5] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre, *Manopt, a Matlab toolbox for optimization on manifolds*, Journal of Machine Learning Research, 15 (2014), pp. 1455–1459, http://www.manopt.org.

[6] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004, https://doi.org/10.1017/CBO9780511804441.

[7] V. Candela and A. Marquina, *Recurrence relations for rational cubic methods ii: The Chebyshev method*, Computing, 45 (1990), pp. 355–367, https://doi.org/10.1007/BF02238803.

[8] D. Cartwright and B. Sturmfels, *The number of eigenvalues of a tensor*, Linear Algebra and its Applications, 438 (2013), pp. 942 – 952, https://doi.org/https://doi.org/10.1016/j.laa.2011.05.040. Tensors and Multilinear Algebra.

[9] L. Chen, L. Han, and L. Zhou, *Computing tensor eigenvalues via homotopy methods*, SIAM Journal on Matrix Analysis and Applications, 37 (2016), pp. 290–319, https://doi.org/10.1137/15M1010725.

[10] A. Edelman, T. A. Arias, and S. T. Smith, *The geometry of algorithms with orthogonality constraints*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 303–353, https://doi.org/10.1137/S0895479895290954.

[11] D. Gabay, *Minimizing a differentiable function over a differential manifold*, Journal of Optimization Theory and Applications, 37 (1982), pp. 177–219, https://doi.org/10.1007/BF00934767.

[12] D. Gabay, *Reduced quasi-Newton methods with feasibility improvement for nonlinearly constrained optimization*, Springer Berlin Heidelberg, Berlin, Heidelberg, 1982, pp. 18–44, https://doi.org/10.1007/BFb0120946.

[13] D. Gabay and D. Luenberger, *Efficiently converging minimization methods based on the reduced gradient*, SIAM J. Control Optim., 14 (1976), pp. 42–61, https://doi.org/10.1137/0314004.

[14] S. Güttel and F. Tisseur, *The nonlinear eigenvalue problem*, Acta Numer., 26 (2017), p. 1–94, https://doi.org/10.1017/S0962492917000034.

[15] M. R. Hestenes, *Multiplier and gradient methods*, J. Optim. Theory Appl., 4 (1969), pp. 303–320, https://doi.org/10.1007/BF00927673.

[16] A. Jaffe, R. Weiss, and B. Nadler, *Newton correction methods for computing real eigenpairs of symmetric tensors*, SIAM Journal on Matrix Analysis and Applications, 39 (2018), pp. 1071–1094, https://doi.org/10.1137/17M1133312.

[17] T. G. Kolda and J. R. Mayo, *Shifted power method for computing tensor eigenpairs*, SIAM Journal on Matrix Analysis and Applications, 32 (2011), pp. 1095–1124, https://doi.org/10.1137/100801482.

[18] P. Lancaster, *Some applications of the Newton-Raphson method to non-linear matrix problems*, Proc. Roy. Soc. London. Ser. A., 271 (1963), pp. 324–331, https://doi.org/10.1098/rspa.1963.0021.

[19] B. Mishra and R. Sepulchre, *Riemannian preconditioning*, SIAM Journal on Optimization, 26 (2016), pp. 635–660, https://doi.org/10.1137/140970860.

[20] D. Nguyen, *Project lagrange_rayleigh.* https://github.com/dnguyend/lagrange_rayleigh, 2019.

[21] A. M. Ostrowski, *On the convergence of the Rayleigh quotient iteration for the computation of the characteristic roots and vectors. iii*, Arch. Ration. Mech. Anal., 3 (1959), pp. 325–340, https://doi.org/10.1007/BF00284184.

[22] A. P.A., T. J., M. R., and A. B., *All roads lead to Newton: Feasible second-order methods for equality-constrained optimization*, tech. report, UCL-INMA, 2009.

[23] M. J. D. Powell, *Algorithms for nonlinear constraints that use lagrangian functions*, Math. Program., 14 (1978), pp. 224–248, https://doi.org/10.1007/BF01588967.

[24] K. Schreiber, *Nonlinear eigenvalue problems: Newton-type methods and nonlinear Rayleigh functionals*, PhD thesis, Technische Universitat Berlin, 2008, https://depositonce.tu-berlin.de/bitstream/11303/2168/2/Dokument_42.pdf.

[25] R. A. Tapia, *Diagonalized multiplier methods and quasi-newton methods for constrained optimization*, J. Optim. Theory Appl., 22 (1977), pp. 135–194, https://doi.org/10.1007/BF00933161.

[26] J. Townsend, N. Koep, and S. Weichwald, *Pymanopt: A python toolbox for optimization on manifolds using automatic differentiation*, J. Mach. Learn. Res., 17 (2016), pp. 1–5, http://jmlr.org/papers/v17/16-177.html.

[27] Z. Zhao, Z. Bai, and X. Jin, *A Riemannian Newton algorithm for nonlinear eigenvalue problems*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 752–774, https://doi.org/10.1137/140967994.