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. Our generalized Rayleigh quotient is an expression often used as an estimate of the Lagrange multiplier in constrained optimization. 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 and its invariant subspace extension. We also obtain a Rayleigh-Chebyshev iteration with cubic convergent rate. We discuss applications of this result to linear and nonlinear eigenvalue and subspace problems as well as potential applications in optimization.

Key words. Lagrange multiplier, Rayleigh quotient, Newton-Raphson, Eigenvalue, Invariant subspace, Optimization

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

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 $\mathcal{L} = \boldsymbol{L} \oplus \boldsymbol{C}$:

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

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 root of $\mathcal{L}$, $\boldsymbol{L}$ and $\boldsymbol{C}$ have Jacobians both of full row rank. In that situation we will call $\boldsymbol{L}$ a *vector* Lagrangian and $\boldsymbol{C}$ a constraint. 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. The full row rank assumption for $\boldsymbol{L}$ ensures that $\boldsymbol{\lambda}$ is solvable in $\boldsymbol{x}$ (but not always explicitly). For a constrained optimization problem a Lagrangian is a scalar function in the literature and what we call vector Lagrangian here is its differential. Since our focus will be on vector Lagrangian in this article we will drop the qualifier *vector* for the remain of the article.

Of particular interest is the case when $\boldsymbol{L}$ is affine in $\boldsymbol{\lambda}$. Let $\boldsymbol{F}$ be a vector function from an open set in E_{in} to E_{out}. For each $\boldsymbol{x}$, assume $\boldsymbol{H}(\boldsymbol{x})$ is a linear function from E_L to E_{out}. Then

$$\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{F}(x) - \boldsymbol{H}(x)\boldsymbol{\lambda} \tag{1.3}$$

is a Lagrangian if it has continuous differential and is of full row rank. We will call this special form explicit Lagrangian.

On the constraint side, from our full row rank assumption $\boldsymbol{C}$ defines a submanifold $\mathcal{M} = \boldsymbol{C}^{-1}(0)$ of codimension $\mathrm{rank}(\boldsymbol{J_C}(\boldsymbol{x}))$.

This set up covers three classes of equations encountered in the literature:

- The eigenvector/invariant subspace problem:

$$\boldsymbol{F} = A\boldsymbol{x}; \boldsymbol{H}(\boldsymbol{x}) = \boldsymbol{x} \tag{1.4}$$

$$\boldsymbol{C}(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{I}_{E_L})$$

  where $\boldsymbol{x}$ is a $n \times k$ matrix, A is a $n \times n$ matrix. In this case E_{in} and E_{out} are both (n, k) matrices and E_L is the space of symmetric $k \times k$ matrices. The case where $k = 1$ is the eigenvector problem.

- The constraint optimization problem, one of the most important problems in applied mathematics. Here $\boldsymbol{F} = \nabla f$ where f is a real value function and $\boldsymbol{H} = \boldsymbol{J}_{\boldsymbol{C}}^T$. 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 $\boldsymbol{C}(\boldsymbol{x})$ on $\boldsymbol{x}$. The system ((1.1)) gives us the set of critical points.
- The nonlinear eigenvalue problem:

  $$P(\lambda)\boldsymbol{x} = 0 \tag{1.5}$$

  Here P is a matrix with polynomial entries in λ. While this is not in the form ((1.1)) we can impose the constraint $\boldsymbol{C}(x) = \boldsymbol{x}^T\boldsymbol{x} - 1$ (or alternatively $\boldsymbol{C}(x) = \boldsymbol{z}^T\boldsymbol{x} - 1$ for a fixed vector $\boldsymbol{z}$) This equation is not of explicit form. 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 ([10]) and Newton-Raphson is an important algorithm.

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

$$\frac{v^TAv}{v^Tv}$$

and the iteration computes

$$v_{i+1} = \frac{(A-\lambda)^{-1}v_i}{||(A-\lambda)^{-1}v_i||}$$

which is shown to have cubic converge in for almost all initial points if A is normal and quadratic otherwise on suitable initial points. ([1]) gives an extension of Rayleigh quotient iteration for the invariant subspace problem where $\boldsymbol{x}$ is a $n \times k$-matrix and $\boldsymbol{\lambda}$ is a $k \times k$-matrix.

The letter $\boldsymbol{\lambda}$ used in the second example in honor of Lagrange is also often used to denote an eigenvalue. The fact that RQI is related to Lagrange multipliers and Newton-Raphson has been known for a long time but it is probably fair to say the relationship is not yet transparent. We will try to clarify the relationship, namely applying Newton-Raphson iteration to ((1.1)), the updating equation for $\boldsymbol{x}$ involves solving a linear equation similar to $(A-\lambda)x = y$. The iterative series of $\boldsymbol{\lambda}$ will have form similar to Rayleigh quotient. When the Lagrangian is of explicit form we can use that form to estimate $\boldsymbol{\lambda}$, resulting in a variation of Newton-Raphson extending RQI. We also obtain an iteration when the Lagrangian is implicit.

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 space involves as vector, leaving the tensor related treatment to specific implementations.

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

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

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

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

where $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x})(\eta) = A\eta - \eta\boldsymbol{\lambda}$. Applying the Schur complement formula we can evaluate the Newton step:

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

and get (with $\boldsymbol{x} = \boldsymbol{x}_i$ and $\boldsymbol{\lambda} = \boldsymbol{\lambda}_i$)

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

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

So with $\zeta = \boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{x}$ 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

$$x_{i+1} = \zeta(2x^T\zeta)^{-1}(1 + x_i^Tx_i)$$

We see x_{i+1} is proportional to ζ, a result known from classical Rayleigh iteration, and the equation for ζ is exactly the Rayleigh equation. However the formula for λ_{i+1} is iterative. Starting with the general equation for the explicit system:

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

From the full rank assumption $\boldsymbol{H}$ has a left inverse (for example $\boldsymbol{H}^- = (\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T$). We can solve for $\boldsymbol{\lambda}$:

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

In the eigenvector case this is exactly the Rayleigh quotient. This expression appeared very early in the multiplier method literature as discussed below.

We will show if we modify the Newton-Raphson process to use this expression to solve ((1.1)), we get a generalization of the Rayleigh quotient iteration that also have quadratic convergent rate in general. We also provide a formula generalizing the Chebyshev iteration. We will show a similar process exists for implicit equation, in particular for nonlinear eigenvalue problem and provide both the RQI and Rayleigh-Chebyshev results, in particular recover the cubic convergent algorithm of Schwetlick and Schreiber ([10], [17] [18]

Many problems in numerical analysis and optimization problems could be reduced to the form ((1.2)). We will apply the results of this paper to a number of situations. In most cases the constraints are at most quadratic, and in many cases the function is also at most quadratic. In those cases, second order methods would have simple form which we will try to exploit. In particular, we have a new cubic algorithm for eigenvectors of nonnormal matrices.

In future research we hope to study inequality constraints and applications of this method to optimization problem.

In the next few sections we discuss a few set ups needed to state the main results.

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 tensor. If the domain space is of shape s_1 and the range space is of shape s_2 then a map between them could be represented as a tensor of shape (s_1, s_2). 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 (may be nonlinear) 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 $L(V, W)$ of linear maps between V and W and is represented by a $m \times n$-matrix.

A second derivative is a linear map between V and $L(V, W) \cong V^* \otimes W$, 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 $\eta_1, \eta_2, \ldots, \eta_l$ consider the tuple

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

we can define

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

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:

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

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

To summarize, there are two maps related to higher derivatives. The map $\boldsymbol{x} \mapsto \boldsymbol{J}^{(l)}_{\boldsymbol{F}}(\boldsymbol{x})$ 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 $\boldsymbol{J}^{(l)}_{\boldsymbol{F}}(\boldsymbol{v})(\boldsymbol{h}^{[l]})$. In code, we need two functions for these two maps. The second map is in general just tensor contraction however depending on problem a custom implementation may be useful.

4. Retractions.

Consider a submanifold $\mathcal{M}$ of $\mathbb{R}^n$. 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. $\mathcal{U} \subset \operatorname{dom}(R)$ 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 M then $\boldsymbol{r} \times Id_{E_L}$ is a retraction on $M \times E_L$, if $\boldsymbol{r}$ is a retraction of second order then $\boldsymbol{r} \times Id_{E_L}$ is a retraction of second order.*

From this proposition and the result of [2]), we can retract intermediate iteration points to $M \times E_L$, as a result $\boldsymbol{x}_i$ can be made elements of $\mathcal{M}$ while $\boldsymbol{\lambda}_i$ is unchanged. Following the common literature, we call a Newton-Raphson iteration on $E_{in} \oplus E_L$ with no retraction to $\mathcal{M}$ infeasible start iteration. When there is retraction to $\mathcal{M}$ we have feasible start iteration. We note Lagrange multiplier with feasible improvement has already been studied been studied, e.g. in [8].

5. Newton-Raphson iteration of Lagrangians.

We first examine the Newton-Raphson iteration. The Jacobian of $\mathcal{L}$ is

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

We will drop $\boldsymbol{x}$ and $\boldsymbol{\lambda}$ from time to time to save space in the expressions. Newton-Raphson iteration in the framework of constraint optimization has been studied by many authors in the literature, including [7], [8], [9], [16], [19]. We will make the connection between $\boldsymbol{L_x}$ with the eigenvector 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 this Jacobian without inverting the whole matrix, we describe two approaches. One approach would focus on first parametrize η using the equation $\boldsymbol{J_C}(\boldsymbol{x})\eta = -\boldsymbol{C}(\boldsymbol{x})$. If the tangent space has an explicit description this would help, otherwise it is parametrized by the null space of $\boldsymbol{J_C}$ ($\eta = \eta_0 + Z\alpha$ where Z is a basis of the null space and α's are the new parameters). This will help reduce the number of variables to $\dim(E_{in}) - \dim(E_L)$. Substitute back to the first equation we get a system of $\dim(E_{out})$

equations and $\dim(E_{in})$ variables ($\dim(E_{in})-\dim(E_L)$ from α and $\dim(E_L)$ from δ) which we can solve. This solves the problem in the most general case, including the case where $\boldsymbol{L}_{\boldsymbol{x}}$ is not invertible and is closely related to the Riemannian Newton approach on the embedded manifold $\mathcal{M}$ defined by $\boldsymbol{C}$. We present a second approach, already investigated in [[16]], [19]] where we assume $\boldsymbol{L}_{\boldsymbol{x}}$ is invertible. This approach reveals some interesting relations mentioned above. As we use the Schur complement formula in this second approach, we will call this solution the Schur form. The approach to parametrize η first is called the tangent form. In a sense, the classical Rayleigh iteration and its extension in [1] is a Schur form solution for a modified Newton-Raphson equation.

The Schur complement with respect to the top block is $-\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{L}_x^{-1}\boldsymbol{L}_{\boldsymbol{\lambda}}$ evaluated at $(\boldsymbol{x},\boldsymbol{\lambda})$ and the inverse of the Jacobian applied on (a,b) is

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

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

$$\delta = -(\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{L}_x^{-1}\boldsymbol{L}_{\boldsymbol{\lambda}})^{-1}\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{L}(\boldsymbol{x}) + (\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{L}_x^{-1}\boldsymbol{L}_{\boldsymbol{\lambda}})^{-1}\boldsymbol{C}(\boldsymbol{x}) \tag{5.3}$$

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

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

$$\delta = -(\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{L}_x^{-1}\boldsymbol{L}_{\boldsymbol{\lambda}})^{-1}\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{L}(\boldsymbol{x}) \tag{5.5}$$

We also note $\boldsymbol{J}_{\boldsymbol{C}}\eta = 0$ from the second row block of the system in that case. For nonlinear eigenvalue problem the above process is the Nonlinear inverse iteration in the literature ([12], [10]). We will review this in subsection 7.4. We note that Schur complement is widely use in equality constraint optimization, for example see chapter 10 of [6] where we see essentially the above calculation. We summarize the steps in algorithm Algorithm 5.1.

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

```
Set $feasible$ to True or False
Initialize $\boldsymbol{x}_0$ and $\boldsymbol{\lambda}_0$
$i \leftarrow 0$
$\zeta \leftarrow SMALL\_NUMBER$
$err \leftarrow BIG\_NUMBER$
while not Terminal[1] $(i, \zeta, err)$ do
    Solve for $\zeta$ in $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\zeta = -\boldsymbol{L}_\lambda(\boldsymbol{x}_i)$
    Solve for $\nu$ in $\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\nu = \boldsymbol{L}(\boldsymbol{x}_i)$
    Compute $\delta \leftarrow (\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)\zeta)^{-1}[-\boldsymbol{C}(\boldsymbol{x}_i) + \boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x}_i)\nu]$
    Compute $\boldsymbol{\lambda}_{i+1} \leftarrow \boldsymbol{\lambda}_i + \delta$
    Compute $\eta \leftarrow -\nu + \zeta\delta$
    if $feasible$ then
        Compute $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{r}(\boldsymbol{x}_i, \eta)$
    else
        Compute $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \eta$
    end if
    $i \leftarrow i + 1$
    $err \leftarrow \boldsymbol{L}(\boldsymbol{x}_{i+1}, \boldsymbol{\lambda}_{i+1})$
end while
```

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

$$\boldsymbol{L}_{\boldsymbol{x}}\eta = \boldsymbol{J}_{\boldsymbol{F}}(\boldsymbol{x})\eta - \boldsymbol{J}_{\boldsymbol{H}}(\boldsymbol{x})\eta\boldsymbol{\lambda}$$

$$\boldsymbol{L}_{\boldsymbol{\lambda}}\delta = -\boldsymbol{H}\delta$$

$\boldsymbol{J}_{\boldsymbol{H}}\eta\boldsymbol{\lambda}$ actually has a simpler form. Write it as

$$\sum_b\sum_c J_{abc}\eta_c\boldsymbol{\lambda}_b = \sum_c(\sum_b \boldsymbol{\lambda}_b J_{abc})\eta_c$$

we see it is $(\boldsymbol{\lambda}\boldsymbol{J}_{\boldsymbol{H}}^{T[12]})\eta$. Here, $T[12]$ means transposing $\boldsymbol{J}_{\boldsymbol{H}}$ with respect to the first two indices.

$$\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\lambda})\eta = (\boldsymbol{J}_{\boldsymbol{F}} - \boldsymbol{\lambda}\boldsymbol{J}_{\boldsymbol{H}}^{T[12]})\eta \tag{5.6}$$

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

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

$$\boldsymbol{\lambda}_{i+1} - \boldsymbol{\lambda}_i = \delta = (\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{L}_{\boldsymbol{\lambda}})^{-1}[\boldsymbol{C}(X) - \boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{L}(\boldsymbol{x})] \tag{5.7}$$

$$\boldsymbol{x}_{i+1} - \boldsymbol{x}_i = \eta = -\boldsymbol{L}_{\boldsymbol{x}}^{-1}(\boldsymbol{L} + \boldsymbol{L}_{\boldsymbol{\lambda}}\delta) \tag{5.8}$$

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

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

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

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

$$\boldsymbol{L}_{\boldsymbol{x}} = \boldsymbol{J}_{\boldsymbol{F}} - \boldsymbol{\lambda}\boldsymbol{J}_{\boldsymbol{H}}^{T[12]}$$

$$\boldsymbol{\lambda}_{i+1} = \boldsymbol{\gamma} = (\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{H})^{-1}[-\boldsymbol{C}(X) + \boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{F}(\boldsymbol{x})] \tag{5.9}$$

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

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

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

$$\boldsymbol{L}_{\boldsymbol{x}}\eta = -\boldsymbol{F}(x) + \boldsymbol{H}(x)\boldsymbol{\gamma} \tag{5.11}$$

While we have noted before $\boldsymbol{L}_x$ is a generalization of the eigenvalue operator, the right hand side of this equation is different from that of Rayleigh quotient. To compute $\boldsymbol{\gamma}$ and η we compute

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

[1] *Terminal* is a function of η, err and i used to decide when to exit the iterations. Typical terminal criteria include $||err|| < max\_err$ or $||\zeta > max\_zeta||$ subjected to a max iteration count.

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

```
Set $feasible$ to True or False
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 $\boldsymbol{L_x}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\zeta = \boldsymbol{H}(\boldsymbol{x}_i)$
    Solve for $\nu$ in $\boldsymbol{L_x}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)\nu = \boldsymbol{F}(\boldsymbol{x}_i)$
    Compute $\boldsymbol{\lambda}_{i+1} \leftarrow (\boldsymbol{J_C}(\boldsymbol{x}_i)\zeta)^{-1}[-\boldsymbol{C}(\boldsymbol{x}_i) + \boldsymbol{J_C}(\boldsymbol{x}_i)\nu]$
    Compute $\eta \leftarrow -\nu + \zeta\boldsymbol{\lambda}_{i+1}$
    if $feasible$ then
        Compute $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{r}(\boldsymbol{x}_i, \eta)$
    else
        Compute $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \eta$
    end if
    $i \leftarrow i + 1$
    $err \leftarrow \boldsymbol{L}(\boldsymbol{x}_{i+1}, \boldsymbol{\lambda}_{i+1})$
end while
```

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

In the eigenvector case ν and ζ are related:

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

and we have seen the calculation get simplified. In the general case we need to compute both ν and ζ . The expression of γ is iterative.

Let us now focus on getting a different expression on γ. From ((5.10))

$$\boldsymbol{H}(\boldsymbol{x})\gamma = \boldsymbol{L_x}\eta + \boldsymbol{F}(\boldsymbol{x})$$

As before let $\boldsymbol{H}^-$ be a left invert to $\boldsymbol{H}$, we solve for $\boldsymbol{\gamma}$:

$$\boldsymbol{\gamma} = \boldsymbol{H}^-[\boldsymbol{L_x}\eta + \boldsymbol{F}(\boldsymbol{x})] \tag{5.12}$$

We see as η converges to zero $\gamma = \boldsymbol{\lambda}_{i+1}$ converges to

$$\boldsymbol{\gamma} = \boldsymbol{H}^-\boldsymbol{F}(\boldsymbol{x})$$

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. This general set up is relevant when we discuss the two-sided Rayleigh quotient.

We note that the papers [7], [8] found the same expression for the general 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 [11]). According to [19] it is difficult to put a name on it without an extensive search of the literature. Lagrange multiplier approximators are widely used in constrained optimization.

Let $\Pi_{\boldsymbol{H}} = I_n - \boldsymbol{H}\boldsymbol{H}^-$. This is a projection. If $\boldsymbol{H} = \boldsymbol{J}_{\boldsymbol{C}}^T$ then it is a projection to the tangent space of $\mathcal{M}$ at $\boldsymbol{x}$, but in general it is not. Substitute the expression ((5.12)) in ((5.11)) and moving η term to one side

$$\Pi_{\boldsymbol{H}}\boldsymbol{L_x}\eta = -\Pi_{\boldsymbol{H}}\boldsymbol{F}(\boldsymbol{x}) \tag{5.13}$$

This system of equations for η is of similar format to the Riemannian Newton equations. However, it is dependent on $\boldsymbol{\lambda}$ which needs to be solved recursively. It is interesting that it involves only $\boldsymbol{H}$ here, and not $\boldsymbol{C}$. In the implicit case, assuming $\boldsymbol{L}_{\boldsymbol{\lambda}}^{-}$ is a left inverse of $\boldsymbol{L}_{\boldsymbol{\lambda}}$ we have with $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}} = I - \boldsymbol{L}_{\boldsymbol{\lambda}}\boldsymbol{L}_{\boldsymbol{\lambda}}^{-}$

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

Thus far the analysis is mostly algebraic, using the Schur complement formula and the special form of $\mathcal{L}$.

For second order iteration we note the formula for Chebyshev iteration (we choose Chebyshev over Halley to avoid another operator inversion) for high dimension is given by

$$v_{i+1} = v_i - \boldsymbol{J}_{\mathcal{L}}^{-1}\mathcal{L} - \frac{1}{2}\boldsymbol{J}_{\mathcal{L}}^{(2)}((\boldsymbol{J}_{\mathcal{L}}^{-1}\mathcal{L})^{[2]}) \tag{5.14}$$

As explain 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 Schur complement to evaluate the second order term to arrive at Chebyshev iteration in algorithm Algorithm 5.3.

Algorithm 5.3 Chebyshev with constrained iterations

```
Set $feasible$ to True or False
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 $LxInvL_2 \leftarrow \boldsymbol{L}_{\boldsymbol{x}}^{-1}l_2$
    Compute $\delta_2 \leftarrow (\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}(\boldsymbol{J}_{\boldsymbol{C}}(LxInvL_2) - \boldsymbol{J}_{\boldsymbol{C}}^{(2)}(\eta^{[2]}))$
    Compute $\eta_2 \leftarrow LxInvL_2 + (\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}l_2$
    Compute $\boldsymbol{\lambda}_{i+i} \leftarrow \boldsymbol{\lambda}_{i+i} - \frac{1}{2}\delta_2$
    if $feasible$ then
        Compute $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{r}(\boldsymbol{x}_i, \eta - \frac{1}{2}\eta_2)$
    else
        Compute $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \eta - \frac{1}{2}\eta_2$
    end if
    $i \leftarrow i + 1$
    $err \leftarrow \boldsymbol{L}(\boldsymbol{x}_{i+1}, \boldsymbol{\lambda}_{i+1})$
end while
```

While this looks relatively complex, 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: the only nonzero terms are $-\eta\delta$ and $\eta^T\eta$ in the Chebyshev expression.

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

6. Rayleigh Quotient Iteration. Motivated by [1] (we learned about [7], [8] late in our research) and the above analysis on Lagrange multipliers, with

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

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

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

would provide a generalization of RQI for to vector Lagrangians. Using augmented Lagrangian technique, [8] 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 prove our particular process has quadratic convergence and extend our result to to the Chebyshev case.

Similar to the Newton-Raphson case, if $\boldsymbol{L}_{\boldsymbol{x}}$ is invertible we have a solution to this system providing a procedure generalizing the classical Rayleigh quotient. In fact, let $\nu = \boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{F}(\boldsymbol{x})$ and $\zeta = \boldsymbol{L}_{\boldsymbol{x}}^{-1}\boldsymbol{H}$ we see

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

satisfies the above equation by direct calculation. As before we call it the Schur form solution. For the eigen problem this provides exactly the classical algorithm. Likewise, we will see later on the invariant subspace RQI of [1] is the Schur form of an equivariant constrained problem on Grassmann manifolds.

We note $\boldsymbol{L}_{\boldsymbol{x}}$ is now solely dependent on $\boldsymbol{x}$ via Rayleigh quotient expression for $\boldsymbol{\lambda}$. We can reparametrize to express $\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}$ as a map from the tangent space of the constraint set ($\boldsymbol{J}_{\boldsymbol{C}}\eta = 0$) to the image of $\Pi_{\boldsymbol{H}}$, both of the same dimension $\dim(E_{in}) - \dim(E_L)$. For the case where $\boldsymbol{H} = \boldsymbol{J}_{\boldsymbol{C}}^T$ and $\boldsymbol{H}^{-} = (\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{J}_{\boldsymbol{C}}^T)^{-1}\boldsymbol{J}_{\boldsymbol{C}}$,

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

$$\boldsymbol{L}_{\boldsymbol{x}}\eta = \boldsymbol{J}_{\boldsymbol{F}}\eta - (\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{J}_{\boldsymbol{C}}^T)^{-1}\boldsymbol{J}_{\boldsymbol{C}}\boldsymbol{F}(\boldsymbol{J}_{\boldsymbol{H}}^{T[12]})\eta$$

($(\boldsymbol{J}_{\boldsymbol{H}}^{T[12]})$ is a just a collection of Hessians for each constraint equation of $\boldsymbol{C}$) and the equation reduces to the emmbedded Riemannian Newton updating equation. When $\boldsymbol{F} = \nabla f$ the last expression is just the expression for the projected Hessian of f. We have the following theorem:

THEOREM 6.1. *Assuming that $\boldsymbol{H}, \boldsymbol{F}, \mathcal{R}$ having continuous derivatives to degree 2, and in a neigborhood of a solution $\boldsymbol{v}$ to the equation $\boldsymbol{L}(v, \boldsymbol{\lambda})$ the map $\Pi_{\boldsymbol{H}}(\boldsymbol{x}_i)\boldsymbol{L}_{\boldsymbol{x}}$ from the tangent space to $\boldsymbol{x}_i$ to the image of $\Pi_{\boldsymbol{H}}(\boldsymbol{x}_i)$ is invertible satisfying*

$$||\Pi_{\boldsymbol{H}}(\boldsymbol{x}_i)\boldsymbol{L}_{\boldsymbol{x}}\psi|| \geq C||\psi|| \tag{6.3}$$

for any ψ in $\mathrm{Ker}(\boldsymbol{J}_{\boldsymbol{C}})$ for some constant C. If $\boldsymbol{r}$ is a first order retraction then for a starting point x_0 close enough to $\boldsymbol{v}$, the system ((6.1)) provides an update to an iteration process which converges with quadratic rate to a solution of ((1.3)).

Assuming ((6.3)), $\boldsymbol{H}, \boldsymbol{F}$ having continuous derivatives to degree 3 and $\mathcal{R}$ have continuous derivatives to degree 2. If $\boldsymbol{r}$ is a second order retraction, then for a starting point $\boldsymbol{x}_0$ close enough to $\boldsymbol{v}$ the Rayleigh-Chebyshev iteration with update

$$\boldsymbol{x}_{i+1} = \boldsymbol{r}(\boldsymbol{x}_i, \eta_* + T(\boldsymbol{x})(\eta_*^{[2]}))$$

where η_* and $T(\boldsymbol{x})(\eta_*^{[2]})$ are defined in the following equations

$$\begin{gathered}\Pi_{\boldsymbol{H}}\boldsymbol{L_x}\eta_* = -\Pi_{\boldsymbol{H}}\boldsymbol{F}(x)\\ \Pi_{\boldsymbol{H}}\boldsymbol{L_x}T(\boldsymbol{x})(\eta_*^{[2]}) = -\frac{1}{2}\boldsymbol{J}_F^{(2)}(\eta_*^{[2]}) - \boldsymbol{J_H}(\eta_*)\boldsymbol{J}_{\mathcal{R}}(\eta_*) + \frac{1}{2}\boldsymbol{J_H}^{(2)}(\eta_*^{[2]})\mathcal{R}(\boldsymbol{x}_i)\\ \boldsymbol{J_C}(\eta_* + T(\boldsymbol{x})(\eta_*^{[2]})) = 0\end{gathered}\tag{6.4}$$

converges cubically to $\boldsymbol{v}$.

If $T(\boldsymbol{v}) = 0$ the Rayleigh quotient iteration converges cubically up to degree 3.

If $\boldsymbol{L_x}$ is invertible the Schur form solution exists

$$\begin{gathered}\nu = \boldsymbol{L_x}^{-1}\boldsymbol{F}(\boldsymbol{x})\\ \zeta = \boldsymbol{L_x}^{-1}\boldsymbol{H}(\boldsymbol{x})\\ \eta = -\nu + \zeta(\boldsymbol{J_C}\zeta)^{-1}\boldsymbol{J_C}\nu\\ \eta_* = -\nu\\ \tau_* = \boldsymbol{L_x}^{-1}\{-\frac{1}{2}\boldsymbol{J}_F^{(2)}(\eta_*^{[2]}) - \boldsymbol{J_H}(\eta_*)(\boldsymbol{J}_{\mathcal{R}}(\eta_*) + \frac{1}{2}\boldsymbol{J_H}^{(2)}(\eta_*^{[2]})\mathcal{R}(\boldsymbol{x}_i)\}\\ \tau = \tau_* - \zeta(\boldsymbol{J_C}\zeta)^{-1}\boldsymbol{J_C}(\tau_*)\end{gathered}\tag{6.5}$$

Proof. Both algorithms require similar estimates. For optimization problem it is just the Riemannian Newton iteration, so this theorem could be considered an extension to vector Lagrangians. We focus on Rayleigh-Chebyshev as regular Rayleigh is the easier version and we will prove it again in the implicit case below. Let $\boldsymbol{v}$ be a solution of ((1.3)). As this is a variant of Newton-Raphson and Chebyshev, implicitly we look at Taylor series expansion of $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda})$ with modifications to make sure the steps belong to the tangent space ($\boldsymbol{J_C}(\eta) = 0$) and note $\Pi_{\boldsymbol{H}}$ will remove some extra terms. When the Lagrangian is explicit, the Taylor series expansion with term $\boldsymbol{\lambda}$ gets cut off early and the second order term would include second derivative with respect to $\boldsymbol{x}$, as well as the cross term of $\boldsymbol{x}$ and $\boldsymbol{\lambda}$. This is the famous cross term in Rayleigh quotient literature linking the estimate of eigenvalues to eigenvector and is responsible for the quadratic convergence rate (see [15]). We have:

$$\begin{aligned}\boldsymbol{H}(\boldsymbol{x}_i)\mathcal{R}(\boldsymbol{x}_i) - \boldsymbol{H}(\boldsymbol{v})\mathcal{R}(\boldsymbol{v}) = -(\boldsymbol{H}(\boldsymbol{x}_i) - \boldsymbol{H}(\boldsymbol{v})(\mathcal{R}(\boldsymbol{x}_i) - \mathcal{R}(\boldsymbol{v})) +\\ \boldsymbol{H}(\boldsymbol{x}_i)(\mathcal{R}(\boldsymbol{x}_i) - \mathcal{R}(\boldsymbol{v})) + (\boldsymbol{H}(\boldsymbol{x}_i) - \boldsymbol{H}(\boldsymbol{v}))\mathcal{R}(\boldsymbol{x}_i)\end{aligned}\tag{6.6}$$

This is purely algebraic (Taylor series of the two variable vector function xy). The first term will converge quadratically while the other two are linear and will be made disappear with an appropriate iteration. We also note the first term is cubic if $\mathcal{R}(\boldsymbol{x}_i) - \mathcal{R}(\boldsymbol{v})$ converges quadratically. Also :

$$\boldsymbol{L}(\boldsymbol{x}, \mathcal{R}(\boldsymbol{x})) = \boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{H}(\boldsymbol{x})\mathcal{R}(\boldsymbol{x}) = \Pi_{\boldsymbol{H}}(\boldsymbol{x})\boldsymbol{F}(\boldsymbol{x})\tag{6.7}$$

So if an iteration step $x_{i+1} = N(\boldsymbol{x}_i)$ is to cancel out $-\boldsymbol{L}(\boldsymbol{x}, \mathcal{R}(\boldsymbol{x}))$, then $\Pi_{\boldsymbol{H}}(\boldsymbol{x}_i)N(\boldsymbol{x}_i)$ also cancel out the same term as $\Pi^2_{\boldsymbol{H}}(\boldsymbol{x}_i) = \Pi_{\boldsymbol{H}}(\boldsymbol{x}_i)$. As $\Pi_{\boldsymbol{H}}(\boldsymbol{x}_i)$ sends terms of form $\boldsymbol{H}D(x)$ to zero, we get a simpler equation to solve.

With these two observations, using the notation of higher derivatives as tensor discussed previously we have the Taylor series expansion

$$\boldsymbol{F}(\boldsymbol{v}) = \boldsymbol{F}(\boldsymbol{x}_i) + \boldsymbol{J_F}(\boldsymbol{x}_i)(\boldsymbol{v} - \boldsymbol{x}_i) + \frac{1}{2}\boldsymbol{J_F}^{(2)}(\boldsymbol{x}_i)(\boldsymbol{v} - \boldsymbol{x}_i)^{[2]} + O(||\boldsymbol{v} - \boldsymbol{x}_i||)^3$$

From here

$$-\boldsymbol{F}(\boldsymbol{x}_i) = -\boldsymbol{F}(\boldsymbol{v}) + \boldsymbol{J_F}(\boldsymbol{x}_i)(\boldsymbol{v} - \boldsymbol{x}_i) + \frac{1}{2}\boldsymbol{J_F}^{(2)}(\boldsymbol{v} - \boldsymbol{x}_i)^2 + O(||\boldsymbol{v} - \boldsymbol{x}_i||)^3$$

Replace $\boldsymbol{F}(\boldsymbol{v})$ on the right hand side with $\boldsymbol{H}(\boldsymbol{v})\mathcal{R}(\boldsymbol{v})$ and add to both sides $\boldsymbol{H}(\boldsymbol{x}_i)\mathcal{R}(\boldsymbol{x}_i)$:

$$\begin{aligned}\boldsymbol{H}(\boldsymbol{x}_i)\mathcal{R}(\boldsymbol{x}_i) - \boldsymbol{F}(\boldsymbol{x}_i) = \boldsymbol{H}(\boldsymbol{x}_i)\mathcal{R}(\boldsymbol{x}_i) - \boldsymbol{H}(\boldsymbol{v})\mathcal{R}(\boldsymbol{v}) + \boldsymbol{J}_{\boldsymbol{F}}(\boldsymbol{x}_i)(\boldsymbol{v} - \boldsymbol{x}_i) + \\ \frac{1}{2}\boldsymbol{J}_{\boldsymbol{F}}^{(2)}(\boldsymbol{v} - \boldsymbol{x}_i)^2 + O(||\boldsymbol{v} - \boldsymbol{x}_i||^3)\end{aligned}$$

Using ((6.6)) and ((6.7)), with $\Pi_{\boldsymbol{H}} = \Pi_{\boldsymbol{H}}(\boldsymbol{x}_i)$

$$\begin{aligned}-\Pi_{\boldsymbol{H}}\boldsymbol{F}(\boldsymbol{x}_i) = -\Pi_{\boldsymbol{H}}(\boldsymbol{H}(\boldsymbol{x}_i) - \boldsymbol{H}(\boldsymbol{v}))(\mathcal{R}(\boldsymbol{x}_i) - \mathcal{R}(\boldsymbol{v})) + \Pi_{\boldsymbol{H}}(\boldsymbol{H}(\boldsymbol{x}_i) - \boldsymbol{H}(\boldsymbol{v}))\mathcal{R}(\boldsymbol{x}_i) + \\ \Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{F}}(\boldsymbol{x}_i)(\boldsymbol{v} - \boldsymbol{x}_i) + \frac{1}{2}\Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{F}}^{(2)}(\boldsymbol{v} - \boldsymbol{x}_i)^2 + O(||\boldsymbol{v} - \boldsymbol{x}_i||^3)\end{aligned}$$

We have

$$\begin{aligned}0 = \Pi_{\boldsymbol{H}}\boldsymbol{F}(\boldsymbol{x}_i) - \Pi_{\boldsymbol{H}}(\boldsymbol{H}(\boldsymbol{x}_{i+1}) - \boldsymbol{H}(\boldsymbol{v}) + (\boldsymbol{H}(\boldsymbol{x}_i) - \boldsymbol{H}(\boldsymbol{x}_{i+1}))(\mathcal{R}(\boldsymbol{x}_{i+1}) - \mathcal{R}(\boldsymbol{v}) + \mathcal{R}(\boldsymbol{x}_i) - \mathcal{R}(\boldsymbol{x}_{i+1})) + \\ \Pi_{\boldsymbol{H}}(\boldsymbol{H}(\boldsymbol{x}_{i+1}) - \boldsymbol{H}(\boldsymbol{v}) + \boldsymbol{H}(\boldsymbol{x}_i) - \boldsymbol{H}(\boldsymbol{x}_{i+1}))\mathcal{R}(\boldsymbol{x}_i) + \\ \Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{F}}(\boldsymbol{x}_i)(\boldsymbol{v} - \boldsymbol{x}_{i+1} + \tau) + \frac{1}{2}\Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{F}}^{(2)}(\boldsymbol{v} - \boldsymbol{x}_{i+1} + \tau)^2 + O(||\boldsymbol{v} - \boldsymbol{x}_i||^3)\end{aligned}$$

Replacing:

$$\boldsymbol{H}(\boldsymbol{x}_{i+1}) - \boldsymbol{H}(\boldsymbol{x}_i) = \boldsymbol{J}_{\boldsymbol{H}}\tau + \frac{1}{2}\boldsymbol{J}_{\boldsymbol{H}}^{[2]}(\tau^{[2]}) + O(\tau^3)$$

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

in the above expression, expanding and separate the terms on the right hand side to two groups, one with terms containing a factor of form $(K(\boldsymbol{x}_{i+1}) - K(\boldsymbol{v}))$ where K is one of the functions $\boldsymbol{H}$, $\boldsymbol{F}$ or just $\boldsymbol{x}_{i+1} - \boldsymbol{v}$, and the other the rest of the terms. The idea is for terms of the first group we can use an inverse function argument to estimate $\boldsymbol{x}_{i+1} - \boldsymbol{v}$, while we choose τ to make sure only cubic order terms remain in the second group. The first group would be

$$\begin{aligned}A = -\Pi_{\boldsymbol{H}}(\boldsymbol{H}(\boldsymbol{x}_{i+1}) - \boldsymbol{H}(\boldsymbol{v}))(\mathcal{R}(\boldsymbol{x}_{i+1}) - \mathcal{R}(\boldsymbol{v})) + \Pi_{\boldsymbol{H}}(\boldsymbol{H}(\boldsymbol{x}_{i+1}) - \boldsymbol{H}(\boldsymbol{v}))\mathcal{R}(\boldsymbol{x}_i) + \\ \Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{F}}(\boldsymbol{x}_i)(\boldsymbol{v} - \boldsymbol{x}_{i+1}) + \frac{1}{2}\Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{F}}^{(2)}(\boldsymbol{v} - \boldsymbol{x}_{i+1})^2 + A_1)\end{aligned}$$

A_1 consists of cross terms, containing factors of form $K(\boldsymbol{x}_{i+1} - K(\boldsymbol{v}))O(||\tau||)$. The second group containing the remaining terms. We will write down terms with order at most two, while grouping the remaining terms in $O(||\tau^3||)$:

$$\begin{aligned}B = \Pi_{\boldsymbol{H}}\boldsymbol{F}(\boldsymbol{x}_i) - \Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{H}}(\tau)\boldsymbol{J}_{\mathcal{R}}(\tau) - \Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{H}}(\tau)\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{x}_i) - \frac{1}{2}\Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{H}}^{[2]}(\tau^{[2]})\mathcal{R}(\boldsymbol{x}_i) - \Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{H}}(\tau)\mathcal{R}(\boldsymbol{x}_i) + \\ \Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{F}}(\tau) + \frac{1}{2}\Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{F}}^{(2)}(\tau^2) + O(||\tau||^3) + O(||\boldsymbol{v} - \boldsymbol{x}_i||^3)\end{aligned}$$

For the linear and quadratic terms in B to cancel we pick $\tau_* = \eta_* + T(\eta_*^{[2]})$. η_* is chosen by comparing first order term and T is chosen by comparing second order terms. We see easily the following choices of η_* and $T(\eta_*^{[2]})$ satisfying

$$\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}(\eta_*) = -\Pi_{\boldsymbol{H}}\boldsymbol{F}(\boldsymbol{x}_i)$$

$$\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}T(\eta_*^{[2]}) = [-\frac{1}{2}\boldsymbol{J}_{F}^{(2)}(\eta_*^{[2]}) - (\boldsymbol{J}_{\boldsymbol{H}}(\eta_*)(\boldsymbol{J}_{\mathcal{R}}(\eta_*) + \frac{1}{2}\boldsymbol{J}_{\boldsymbol{H}}^{(2)}(\eta_*^{[2]})\mathcal{R}]$$

With these choices of η_* and $T(\eta_*^{[2]})$, assuming $\Pi_H\boldsymbol{L}_x$ is invertible near $\boldsymbol{v}$, using the retraction we can assume we work on the tangent space of $\boldsymbol{x}_{i+1}$. In A, $\Pi_{\boldsymbol{H}}(\boldsymbol{H}(\boldsymbol{x}_{i+1}) - \boldsymbol{H}(\boldsymbol{v}))\mathcal{R}(\boldsymbol{x}_i)$ is approximated to first degree by $\Pi_{\boldsymbol{H}}\boldsymbol{J}_{\boldsymbol{H}}(\boldsymbol{x}_{i+1}-\boldsymbol{v})$ and so A is approximated to first order by $\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{v}-\boldsymbol{x}_{i+1})$. That is the dominant term in A, the remaining terms are all quadratic so we have the estimate

$$A \leq (C+m)||\boldsymbol{x}_{i+1}-\boldsymbol{v}||.$$

Here $m \ll C$ is a constant bound estimate from the remaining terms of A. and because of our choice of the iteration step, the B is $O(||v-\boldsymbol{x}_i||^3)$. Hence we have cubic convergence. For the Schur form we note for any vector τ_*

$$\tau = \tau_* - \zeta(\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}\boldsymbol{J}_{\boldsymbol{C}}(\tau_*) \tag{6.8}$$

is a projection of τ_* to the tangent space ($\boldsymbol{J}_{\boldsymbol{C}}(\tau)=0$) satisfying:

$$\Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}\tau = \Pi_{\boldsymbol{H}}\boldsymbol{L}_{\boldsymbol{x}}\tau_*$$

In implementation we use $\eta_* = -\nu + \zeta\boldsymbol{\lambda}_i$. Its projection is still $-\Pi_{\boldsymbol{H}}\boldsymbol{F}(\boldsymbol{x})$ but it is closer to η in RQI. The final projection of τ_* would ensure τ is in the tangent space.

When $T(\boldsymbol{v})=0$ and from our assumption T is continuously differentiable, cubic convergence follows from expansion near $\boldsymbol{v}$. □

Algorithm 6.1 Rayleigh quotient iteration for constrained-type equations 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 $\zeta$ 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 $\boldsymbol{\lambda}_* = (\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}\boldsymbol{J}_{\boldsymbol{C}}(\nu)$
    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
```

Algorithm 6.2 Rayleigh-Chebyshev iteration for constrained-type equations 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 $\zeta$ 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 $\eta \leftarrow -\nu + \zeta\boldsymbol{\lambda}_i$
    Compute $T_*(\eta^{[2]})$ as solution to $\boldsymbol{L}_{\boldsymbol{x}}T_*(\eta^{[2]}) = -[\frac{1}{2}\boldsymbol{J}_F^{(2)}\eta^{[2]} - (\boldsymbol{J}_{\boldsymbol{H}}\eta)(\boldsymbol{J}_{\mathcal{R}}\eta) - \frac{1}{2}\boldsymbol{J}_{\boldsymbol{H}}^{(2)}\eta^{[2]}\mathcal{R}]$
    Compute $\tau_* = \eta + T_*(\eta^{[2]})$
    Compute $\tau = \tau_* - \boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}\boldsymbol{J}_{\boldsymbol{C}}(\tau_*)$
    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
```

If not using Schur form, we can solve the system directly as a linear system from the tangent space to the image of $\Pi_{\boldsymbol{H}}$. This is essentially an extension of the Riemannian Newton approach. We note an important feature in the proof is the use of $\Pi_{\boldsymbol{H}}$ to eliminate the term $(\boldsymbol{H}(\boldsymbol{x}_i)(\mathcal{R}(\boldsymbol{x}_i) - \mathcal{R}(\boldsymbol{v}))$ (if we keep this term it would generate a derivative of $\mathcal{R}$ in the Newton step). In the implicit case we could do the same to eliminate the term involving $\boldsymbol{L}_{\boldsymbol{\lambda}}(\boldsymbol{x}_i)$. The algorithms generalize to the implicit case:

Theorem 6.2. *Assume $\boldsymbol{L}$ has continuous derivatives to degree two. Let $(\boldsymbol{v}, \boldsymbol{\mu})$ be a solution for the system. Assuming $\mathcal{R}(\boldsymbol{x})$ is a C^1 function such that $\boldsymbol{L}(\boldsymbol{v}, \mathcal{R}(\boldsymbol{v})) = 0$ Let $\boldsymbol{L}_{\boldsymbol{\lambda}}{}^{-}$ be a left inverse of $\boldsymbol{L}_{\boldsymbol{\lambda}}$ and $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}} = \boldsymbol{I} - \boldsymbol{L}_{\boldsymbol{\lambda}}\boldsymbol{L}_{\boldsymbol{\lambda}}{}^{-}$. Assume $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x})$ is invertible (as a map from* $\mathrm{Ker}(\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{x})$ *to* $\mathrm{Im}(\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\boldsymbol{x})))$ *for $\boldsymbol{x}$ in a neighborhood of $\boldsymbol{v}$ and*

$$||\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x})\psi|| \geq C||\psi||$$

The iteration $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \eta$ with

$$\begin{gathered} \boldsymbol{\lambda} = \mathcal{R}(\boldsymbol{x}) \\ \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}\eta = 0 \\ \boldsymbol{J}_{\boldsymbol{C}}\eta = 0 \end{gathered} \tag{6.9}$$

together with a first order retraction converges quadratically to $(\boldsymbol{v}, \boldsymbol{\mu})$ If further $\boldsymbol{L}$ is C^3 and $\mathcal{R}$ is a C^2 function then the iterative process $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \eta_* + T(\eta_*^{[2]})$ with

$$\begin{gathered} \boldsymbol{\lambda} = \mathcal{R}(\boldsymbol{x}) \\ \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}\eta_* = 0 \\ \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}T(\eta_*^{[2]}) = -\frac{1}{2}\boldsymbol{L}_{\boldsymbol{x}\boldsymbol{x}}(\eta_*^{[2]}) - \boldsymbol{L}_{\boldsymbol{x}\boldsymbol{\lambda}}([\eta_*, \boldsymbol{J}_{\mathcal{R}}(\eta)]) - \frac{1}{2}\boldsymbol{J}_{\boldsymbol{\lambda}\boldsymbol{\lambda}}(\boldsymbol{J}_{\mathcal{R}}(\eta_*)^{[2]}) \\ \boldsymbol{J}_{\boldsymbol{C}}(\eta_* + T(\eta_*^{[2]})) = 0 \end{gathered} \tag{6.10}$$

together with a second order retraction converges cubically. When $T(\boldsymbol{v}) = 0$ as a tensor, the implicit Rayleigh quotient converges cubically. When $\boldsymbol{L}_{\boldsymbol{x}}$ is invertible the Schur form of the solution exists with

$$\begin{gathered} \zeta = -\boldsymbol{L}_{\boldsymbol{x}}^{-1}(\boldsymbol{x}, \boldsymbol{\lambda})\boldsymbol{L}_{\boldsymbol{\lambda}}(\boldsymbol{x}, \boldsymbol{\lambda}) \\ \nu = \boldsymbol{L}_{\boldsymbol{x}}^{-1}(\boldsymbol{x}, \boldsymbol{\lambda})\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) \\ \boldsymbol{\lambda}_* = (\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}\boldsymbol{J}_{\boldsymbol{C}}\nu \\ \eta = -\nu + \zeta\boldsymbol{\lambda}_* \end{gathered} \tag{6.11}$$

What we call a Rayleigh quotient here is called a Lagrange multiplier approximation formula in [19]. $\mathcal{R}$ is most often given implicitly 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 implicit Lagrangian system $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = 0$, then $\mathcal{N}(\boldsymbol{x}, \boldsymbol{\lambda}) = 0$. Assuming $\boldsymbol{J}_{\mathcal{N}}$ is of rank $\dim E_L$ in a neighborhood of $(\boldsymbol{v}, \boldsymbol{\mu})$ and $\mathcal{R}$ is the inverse function, that means $\mathcal{N}(\boldsymbol{x}, \mathcal{R}(\boldsymbol{x})) = 0$ for all $\boldsymbol{x}$ in a neighborhood of $\boldsymbol{v}$ and hence:

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

In section on nonlinear eigenvalue problem below we give a concrete example how this is done.

Proof. The proof is a simple extension of the explicit case involving Taylor expansion of $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda})$ then removing the term $\boldsymbol{L}_{\boldsymbol{\lambda}}(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)(\boldsymbol{\lambda}_i - \boldsymbol{\mu})$ by applying $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}$. We will do this for the Rayleigh case:

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

Applying $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}(\boldsymbol{x}_i)$ to both sides we get

$$0 = \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i)) + \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{v} - \boldsymbol{x}_i) + O(||\boldsymbol{v} - \boldsymbol{x}_i||)$$

Expressing $\boldsymbol{x}_i - \boldsymbol{v} = (\boldsymbol{x}_{i+1} - \boldsymbol{v}) - \eta + O(||\eta||^2)$ using the first order retraction and substitute in, we get

$$\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}(\boldsymbol{x}_{i+1} - \boldsymbol{v}) = -\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}(\boldsymbol{x}_i, \mathcal{R}(\boldsymbol{x}_i)) - \Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}(\eta) + O(||\eta||^2) + O(||\boldsymbol{x}_i - \boldsymbol{v}||^2))$$

From here with the choice of η solving

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

applying an implicit function estimate on the left hand side and that $O(||\eta||^2)$ is at least $O(||\boldsymbol{x}_i - \boldsymbol{v}||^2)$ on the right hand side we get quadratic convergence. □

The theorem shows there much freedom in choosing $\mathcal{R}$, the main requirement is it is consistent with the system: if $(\boldsymbol{v}, \boldsymbol{\mu})$ is a solution then $\mathcal{R}(\boldsymbol{v}) = \mu$. So $\mathcal{R}$ does not need to come from a pseudo inverse (note that we did not need to use the fact that $\mathcal{R}$ comes from a pseudo inverse in proving the explicit RQI). We numerically tested this implicit 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.

To solve the updating equations, as mentioned we can parametrize η using $\boldsymbol{J}_{\boldsymbol{C}}\eta = 0$ or use the Schur form. For the first case since we may not have a convenient matrix for $\mathrm{Im}(\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}})$, a strategy could be to start with a basis for $\mathrm{Ker}(\boldsymbol{J}_{\boldsymbol{C}})$ and map it by $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}_{\boldsymbol{x}}$ to a basis of $\mathrm{Im}(\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}})$. Representing $\Pi_{\boldsymbol{L}_{\boldsymbol{\lambda}}}\boldsymbol{L}(\boldsymbol{x}_i)$ in this basis by would solve the updating equation. The last step may be done by a least square method. If we look at the Schur form, beside solving $\boldsymbol{L}_{\boldsymbol{x}}$ for ν and ζ, we also need to compute $\boldsymbol{\lambda}_* = (\boldsymbol{J}_{\boldsymbol{C}}\zeta)^{-1}\boldsymbol{J}_{\boldsymbol{C}}\nu$. A natural question is if we can use $\mathcal{R}(\boldsymbol{x})$ instead of $\boldsymbol{\lambda}_*$. The resulting $-\nu + \zeta\mathcal{R}(\boldsymbol{x}_i)$ is no longer on the tangent space, but would be close so if the retraction is extended near $T\mathcal{M}$ in E_{in}, the iterative process could converge. [8] showed a quasi-Newton method based on this Rayleigh quotient converges superlinearly in general. We analyzed the use of this approximation of $\boldsymbol{\lambda}_*$ in the linear constraint example below.

7. Examples. The python code for these examples could be found at [13]. The code is set up for a general framework but for matrix or higher tensor the user needs to provide his own method. We consulted [5] and [20]. 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 $(\boldsymbol{J}_{\boldsymbol{C}}\zeta)\boldsymbol{J}_{\boldsymbol{C}}\nu$ in some cases. We construct the left inverse $\boldsymbol{H}^- = (\boldsymbol{H}^\dagger\boldsymbol{H})^{-1}\boldsymbol{H}^\dagger$ 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 double check the result so we have not spent much effort to optimize the code.

7.1. Optimization of 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. RQI provides a way to find critical points. As in the unconstrained case, a critical point finding method need to be used with a gradient method to find local optimal points. There is an extensive literature on quasi-Newton method in constrained optimization, including Sequential Quadratic Programming method (SQP) where an approximation of $\boldsymbol{L}_{\boldsymbol{x}}$ is used in iterations. [6], chapter 10 has a number of examples using Schur form. As explained there, when the Schur form is sparse this algorithm becomes valuable.

7.2. Eigenvectors. First we consider the eigenvector problem under the quadratic constraint

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

The code shows convergent for all cases as expected with appropriate initial points. Care need to be taken in determine when $\boldsymbol{L}_x$ becomes singular as terminal condition. The Newton-Raphson method has been examined earlier in the article and we have found $\boldsymbol{x}_{i+1}$ is proportional to ζ. However, the second derivative is

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

is not dependent on A and we do not have cubic convergence here. The Chebyshev adjustment is simple but essentially involves a second application of $(A-\lambda)^{-1}$.

The Rayleigh case is exactly the classical case and has been considered extensively in the literature. With Rayleigh quotient $(\boldsymbol{x}^T\boldsymbol{x})^{-1}\boldsymbol{x}^TA\boldsymbol{x}$

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

We have cubic convergent when A is normal.

For the nonnormal case the Chebyshev term is proportional to $(A-\lambda\boldsymbol{I})^{-1}\eta$. We note we use the same $A-\lambda\boldsymbol{I}$ in the second step so intermediate data, for example LU factorization could be reused. However, the second step is sequential to the first. It would be interesting to study global convergence property of the Rayleigh-Chebyshev algorithm.

Using the same $\boldsymbol{H}(\boldsymbol{x}) = \boldsymbol{x}$ but imposing the constraint

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

instead of the quadratic constraint, we can still use $\boldsymbol{H}^\dagger(\boldsymbol{x}) = \boldsymbol{x}^T$. The Rayleigh quotient will still be $(\boldsymbol{x}^T\boldsymbol{x})^{-1}(\boldsymbol{x}^TA\boldsymbol{x})$. In the code we also tested the case where $\boldsymbol{H}^\dagger(\boldsymbol{x}) = \boldsymbol{z}^T$, which gives the Rayleigh quotient $(\boldsymbol{z}^T\boldsymbol{x})^{-1}(\boldsymbol{z}^TA\boldsymbol{x})$. We note $\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{x})(\eta) = \boldsymbol{z}^TA(\boldsymbol{I}-\boldsymbol{x}\boldsymbol{z}^T)\eta$ on the constraint manifold. A consequence of it is $\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{v}) = 0$ if $\boldsymbol{z}^T$ is a left eigenvector. The Rayleigh quotient iteration converges cubically for the corresponding right eigenvector. This may not be of practical use but may be an interesting note in view of the two-sided Rayleigh quotient method below. The Rayleigh-Chebyshev iteration requires solving $(A-\mathcal{R}(\boldsymbol{x})\boldsymbol{I})^{-1}\eta$ as before.

7.3. Two-sided Rayleigh quotient. This algorithm by Ostrowski [14] has cubic convergent rate even for nonnormal matrix A. In our Lagrangian approach it is equivalent to

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

$$\boldsymbol{\lambda} = (\lambda, \mu)$$

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

$$\boldsymbol{F}(\boldsymbol{x}) = \begin{pmatrix} 0 & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} A^Tv \\ Au \end{pmatrix}$$

$$\boldsymbol{H}(\boldsymbol{x})(\boldsymbol{\lambda}) = \begin{pmatrix} v\lambda \\ u\mu \end{pmatrix}$$

$\boldsymbol{H}^\dagger$ is a map from E_{in} to E_L defined by and $\boldsymbol{H}^\dagger \begin{pmatrix} a \\ b \end{pmatrix} = (u^Ta, v^Tb)$.

$$\boldsymbol{C}(\boldsymbol{x}) = \begin{pmatrix} v^Tv - 1 \\ u^Tu - 1 \end{pmatrix}$$

We see

$$(\boldsymbol{H}^\dagger\boldsymbol{H}) = \begin{pmatrix} u^Tv & 0 \\ 0 & v^Tu \end{pmatrix}$$

$$\mathcal{R}(\boldsymbol{x}) = ((u^TA^Tv)/(u^Tv), (v^TAu)/(v^Tu))$$

and we recover Ostrowski's algorithm. While $\mathcal{R}$ has two components they are in fact identical. Cubic convergent rate is a consequence of $\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{v}) = 0$ at the eigenvector which we can verify directly. See [4] for an invariant subspace version.

7.4. Implicit Lagrangian: Nonlinear eigenvalue problem. Newton-Raphson method was applied to nonlinear eigenvalue problem (also called λ-matrix problem) in [10]. With $\boldsymbol{\lambda} = (\lambda)$, the equation has the form

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

Applying algorithm Algorithm 5.1 we have $\boldsymbol{L}_{\boldsymbol{x}} = P(\lambda)$, $\boldsymbol{L}_{\boldsymbol{\lambda}} = P'(\lambda)\boldsymbol{x}$ and the Schur complement is either $\boldsymbol{x}^T P^{-1}(\lambda)P'(\lambda)\boldsymbol{x}$ for the quadratic constraint or $\boldsymbol{z}^T P^{-1}(\lambda)P'(\lambda)\boldsymbol{x}$ for the linear constraint and reduces to algorithm 4.7 in the above citation. To apply theorem Theorem 6.2, we need a Rayleigh quotient, which we define to be the $\boldsymbol{\lambda}$ satisfying of the system

$$\mathcal{N}(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{x}^H P(\lambda)\boldsymbol{x}$$

so

$$\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{x}) = -P_{\lambda}^{-1}\boldsymbol{x}^H(P + P^H)$$

In particular if P is normal $\boldsymbol{J}_{\mathcal{R}}(\boldsymbol{v}) = 0$ for any nonlinear eigenvector. The RQI update reduces to algorithm 4.9 in [10]. For nonsymmetric case we can define $\boldsymbol{x} = \begin{pmatrix} u \\ v \end{pmatrix}$ as before and

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

Take $\mathcal{N}(\boldsymbol{x}, \lambda) = v^H P u + u^H P^H v$ we recover the nonlinear two-sided RQI with cubic convergence. For the quadratic eigenvalue problem, the Rayleigh-Chebyshev algorithm could be competitive.

7.5. Vector Lagrangian with various constraints. We tested the code with multiple vector constraints. If $\boldsymbol{x}$ is if of size n and $\boldsymbol{C}$ consist of k constraints, we assume that it has been solved for for the first $n - k$ variables, so $\boldsymbol{x}[n - k + i] = c_i(\boldsymbol{x}[0..n - 1])$. Locally any constraint could be transformed to this form. We use the orthographic retraction, which is simpler in this case. The constraint function we take 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.

7.6. An experiment with linear constraints. We consider the case $E_{in} = E_{out} = \mathbb{R}^n, E_L = \mathbb{R}^p$ and $b \in \mathbb{R}^p$ with the constraint $\boldsymbol{C}(\boldsymbol{x}) = C\boldsymbol{x} - b$ and $\boldsymbol{H}(\boldsymbol{x}) = \boldsymbol{J}_{\boldsymbol{C}}^T = C^T$. Here C is a $p \times n$ matrix. As $\boldsymbol{J}_{\boldsymbol{H}}$ is zero $\boldsymbol{L}_{\boldsymbol{x}} = \boldsymbol{J}_{\boldsymbol{F}}$. The Rayleigh quotient is $\mathcal{R}(\boldsymbol{x}) = (C^T C)^{-1} C\boldsymbol{F}(\boldsymbol{x})$. However only $\boldsymbol{\lambda}_* = (C\zeta)^{-1}(C\nu)$ appear in the iteration, so we have a situation that has been analyzed in [6].

We did a little more analysis to understand the effect of replacing $\boldsymbol{\lambda}_*$ with $\boldsymbol{\lambda} = \mathcal{R}(\boldsymbol{x})$ in the Schur form solution $\eta = -\nu + \zeta\boldsymbol{\lambda}_*$. Recall the projection of x_0 to $\mathcal{M}$ in this case is

$$\Pi(\boldsymbol{x}) = \Pi_{\boldsymbol{H}}(\boldsymbol{x}) = \boldsymbol{I}_n - C^T(CC^T)^{-1}C$$

If $f(x) = \frac{1}{2}x^T A x + d^T x$ then $\boldsymbol{F}(x) = \nabla f = Ax + d$ and $\boldsymbol{L}(\boldsymbol{x}) = A\boldsymbol{x} - C^T\boldsymbol{\lambda}$. Here A is a positive definite matrix. The closed form solution to the constrained system:

$$A\boldsymbol{x} + d - C^T\boldsymbol{\lambda} = 0$$
$$C\boldsymbol{x} = b$$

is

$$\boldsymbol{x} = A^{-1}C^T(CAC^T)^{-1}b + A^{-1}C^T(CA^{-1}C^T)^{-1}CA^{-1}d.$$

If we isolate the tangent space in the Riemannian Newton method, we arrive at the exact solution in one step. The Schur form with

$$\eta = -\nu + \zeta\boldsymbol{\lambda}_*$$

also gives the exact solution after one iteration. If we use $\boldsymbol{\lambda}$ as a proxy for $\boldsymbol{\lambda}_*$, the Rayleigh quotient iteration with the retraction Π above gives an iterative process

$$\boldsymbol{x}_{i+1} = C^T(CC^T)^{-1}b - \Pi A^{-1}Ad - (\Pi A^{-1}\Pi A - \Pi)\boldsymbol{x}_i$$

When this series converges, it does converge linearly to the closed form solution, a fact not hard to establish but non trivial at first look. This example shows $\boldsymbol{\lambda}_*$ is needed when we want quadratic convergence.

7.7. RQI on Stiefel manifolds.

The constraint for Stiefel manifolds is $\frac{1}{2}(x^Tx - \boldsymbol{I}_p)$. 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

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

as seen in [3]. We build the code to test solve the explicit Lagrangian with $\boldsymbol{F}(x) = A\boldsymbol{x} + b$ where A is a (n, p, n, p) tensor and b is a (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 a $(n, p, p(p+1)/2)$ 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 a $(np, p(p+1)/2)$ 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 a (n, p) matrix. The Schur form requires solving the larger system with dimension np instead of $np - p(p+1)/2$ of the Riemannian Newton method, if the codimension of the Stiefel manifold is not too big the Schur form could be a useful alternative. We also include the code for 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 $\boldsymbol{H} = \boldsymbol{J}_{\boldsymbol{C}}^T$. The orthogonal group O_p acts on E_{in}, E_{out} and E_L, generating vector fields on these spaces. The invariance under the action of allows us to identify the tangent space of the Grassmann manifold with the space of matrix with $x^T\eta = 0$. The action on E_{out} define a subspace of $\mathrm{Im}(\Pi_{\boldsymbol{H}})$ orthogonal to these vector fields, and 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:

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

In general, we do not 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. However when we do have this relation the tensor ζ could be expressed as a left multiplication by a matrix (also called ζ). In this case the Schur form is simpler:

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

Which we can verify directly, using the associativity mentioned above. An important case is the sphere, as γ is then a scalar. 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 $\mathcal{R}(\boldsymbol{x}) = \boldsymbol{x}^T A^T \boldsymbol{x}$. Now if $z = \boldsymbol{x} + \zeta\boldsymbol{\lambda}$ then

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

so we can take $\nu = \boldsymbol{x} + \boldsymbol{x}\boldsymbol{\lambda}$ and so

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

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 can also try the Rayleigh-Chebyshev form, which should be of the same format as before. 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 $\boldsymbol{x}^T\boldsymbol{x} = \boldsymbol{I}_p$ ([21], [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.

9. Discussion.

We show that Rayleigh quotient iteration is a modification of Newton-Raphson on with an estimate of Lagrange multiplier as a function of $\boldsymbol{x}$ only. It provides an updating equation coincide with Riemannian Newton methods with embedded metric. It also has a Schur form solution that simplifies the computation in many instances, and this form is responsible for the original RQI algorithm. In view of this, we think it should be a valuable technique to study constraint problems.

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] 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.

[8] 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.

[9] 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.

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

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

[12] 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.

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

[14] 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.

[15] B. N. Parlett, *The rayleigh quotient iteration and some generalizations for nonnormal matrices*, Math. of Comp., 28 (1974), pp. 679–693.

[16] 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.

[17] 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.

[18] H. Schwetlick and K. Schreiber, *Nonlinear rayleigh functionals*, Linear Algebra Appl., 436 (2012), pp. 3991 – 4016, https://doi.org/10.1016/j.laa.2010.06.048. Special Issue dedicated to Heinrich Voss's 65th birthday.

[19] 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.

[20] 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.

[21] 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.