MA5680: TRS

Daniel Henderson

March 29, 2025

In the language of your choice implement step 2 of algorithm 5.1 on p14 of the article.

Solution. A bit of a digression but we show that out generalized eigenvalue problem may be written in standard form:

$$\left(\begin{array}{c|c} -B & A \\ \hline A & -\frac{gg^\top}{\Delta^2} \end{array}\right)\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = -\lambda^* \left(\begin{array}{c|c} 0 & B \\ \hline B & 0 \end{array}\right)\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

$$\left(\begin{array}{c|c} 0 & B \\ \hline B & 0 \end{array}\right)^{-1}\left(\begin{array}{c|c} -B & A \\ \hline A & -\frac{gg^\top}{\Delta^2} \end{array}\right)\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = -\lambda^* \left(\begin{array}{c|c} 0 & B \\ \hline B & 0 \end{array}\right)^{-1}\left(\begin{array}{c|c} 0 & B \\ \hline B & 0 \end{array}\right)\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

$$\left(\begin{array}{c|c} 0 & B^{-1} \\ \hline B^{-1} & 0 \end{array}\right)\left(\begin{array}{c|c} -B & A \\ \hline A & -\frac{gg^\top}{\Delta^2} \end{array}\right)\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = -\lambda^* \left(\begin{array}{c|c} 0 & B^{-1} \\ \hline B^{-1} & 0 \end{array}\right)\left(\begin{array}{c|c} 0 & B \\ \hline B & 0 \end{array}\right)\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

$$\left(\begin{array}{c|c} -B^{-1}B & -B^{-1}\frac{gg^\top}{\Delta^2} \\ \hline -B^{-1}A & B^{-1}A \end{array}\right)\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = = -\lambda^* \left(\begin{array}{c|c} B^{-1}B & 0 \\ \hline 0 & B^{-1}B \end{array}\right)\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

$$\left(\begin{array}{c|c} -B^{-1}B & -B^{-1}\frac{gg^\top}{\Delta^2} \\ \hline -B^{-1}A & B^{-1}A \end{array}\right)\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = -\lambda^* I_{2n\times 2n}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

$$\left(\left(\begin{array}{c|c} -B^{-1}B & -B^{-1}\frac{gg^\top}{\Delta^2} \\ \hline -B^{-1}A & B^{-1}A \end{array}\right) - \lambda^* I_{2n\times 2n}\right)\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = 0$$

Note, that $\left(\begin{array}{c|c} 0 & B \\ \hline B & 0 \end{array}\right)$ is invertible, as $\det\left(\begin{array}{c|c} 0 & B \\ \hline B & 0 \end{array}\right) = \det(0_{n\times n} - B) * \det(0_{n\times n} + B) = -\det(B)^2 < 0$ as B is symmetric positive definite, which implies $\det(B) > 0$. Concluding our digression, we make note that the hard-case occurs when λ^* equals the largest eigenvalue of $A + \lambda B$, or equivalently, when $\mathrm{alge}(\lambda^*) > 1$.

Code 1: Implementation of Algorithm 5.1 step 2

```julia
julia> using LinearAlgebra
julia> A = let X = randn(n, n); X + X'; end;
julia> B = let X = randn(n, n); X * X'; end;
julia> Δ = 0.1
julia> g, p0 = randn(n), randn(n)
julia> m = (p) -> p' * g + 0.5 * p' * A * p
julia> M0 = [-B A; A -g*g'/Δ^2];
julia> M1 = -[zeros(n,n) B; B zeros(n,n)]
julia> F = eigen(M0, M1)
julia> λstar = maximum(abs.(F.values)) # HACK: okay since we are ignoring the hard case
  22.045702676755557
julia> y = F.vectors[:, end] # And a worse hack
julia> y1, y2 = y[1:10], y[11:20]
julia> pstar = sign(g'*y2)*Δ*y1./sqrt(y1'*B*y1)
10-element Vector{ComplexF64}:
   -0.0696182291622581 + 0.0im
 -0.029856150657183066 + 0.0im
  0.021457658057046993 + 0.0im
 -0.015672083161465084 + 0.0im
   0.03743652574264288 + 0.0im
  -0.04079621450785599 + 0.0im
 -0.017286212026279373 + 0.0im
   0.01829971992538563 + 0.0im
  -0.00794105054155718 + 0.0im
 -0.056551058280238306 + 0.0im
julia> sqrt(pstar'*B*pstar)
 0.1 + 0.0im # Hence our solution is on the boundary
```

∎