An Algorithm for the Generalized Symmetric Tridiagonal Eigenvalue Problem

Kuiyuam Li, Tien-Yien Li, Zhonggang Zeng

Giulio Masetti

Università di Pisa
Corso Metodi di Approssimazione 2012-2013

July 21, 2013

Generalized Eigenvalue Problem

Def

$T, S \in \mathbb{R}^{n \times n}$. We call (T, S) *pencil*.

We consider *only* $\boxed{\begin{array}{l}\text{symmetric}\\ \text{and}\\ \text{tridiagonal}\end{array}}$ pencil.

Generalized Eigenvalue Problem

Def

$T, S \in \mathbb{R}^{n \times n}$. We call (T, S) *pencil*.

We consider *only* $\boxed{\begin{array}{l}\text{symmetric}\\ \text{and}\\ \text{tridiagonal}\end{array}}$ pencil.

Def (Problem)

Find λ such that $Tx = \lambda Sx$.

T, S symmetric implies $\lambda \in \mathbb{R}$.

Algorithm philosophy

We find zeros of the polynomial equation

$$\mathcal{F}_{(T,S)}(\lambda) = \det(T - \lambda S) = 0$$

using an iterative method, living on real line.

Brainstorming

We want:	Fast and secure iterative method. Starting points for our method. Scalability.
We have:	Laguerre's method. Cuppen's divide and conquer method. Symmetric tridiagonal matrices.
We add:	Unreducible condition. Dynamic programming (Bottom-up). Efficient matrix storing.

Unreducible pencil

Def (as in [1])

(T, S) is an *unreducible pencil* if $t_{i,i+1}^2 + s_{i,i+1}^2 \neq 0$ for $i = 1, 2, \ldots, n-1$.

Matrix storin

$$T = trid(sub, diag, super)$$

But T is symmetric, so $sub = super$. We define and use

```
integer, parameter :: dp = kind(1.d0)
real(dp), dimension(1:n,0:1) :: T, S
```

with $T(:,0) = diag$ and $T(:,1) = super$.

Fast and secure iterative method

$\mathcal{F}_{T,S}(\lambda)$ is a polynomial with only real zeros; we call them

$$\lambda_1 < \lambda_2 < \cdots < \lambda_n$$

where we count with multiplicity.
If λ_m and λ_{m+1} are simple zeros (mlt=1), then we consider now the quadric

$$g_u(X) = (x - X)^2 \sum_{i=1}^{n} \frac{(u - \lambda_i)^2}{(x - \lambda_i)^2} - (u - X)^2$$

if $\boxed{u \neq x}$ then $g_u(x) < 0$ and $g_u(\lambda_m) > 0$.
So we have two sign change.
Bolzano's Theorem tell us that there are two zeros of g_u between λ_m and λ_{m+1}. We call them X_-, X_+.

Fast and secure iterative method

$$\lambda_m < X_- < x < X_+ < \lambda_{m+1}$$

We have one freedom: the u parameter. Calling $\hat{X}_- = min_u X_-$ and $\hat{X}_+ = max_u X_+$ we can obtain

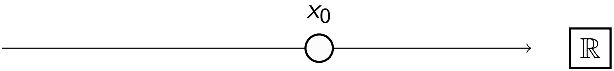
$$\lambda_m \approx \hat{X}_- < x < \hat{X}_+ \approx \lambda_{m+1}$$

With algebraic manipulations
we obtain

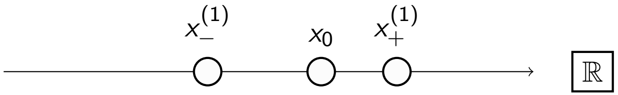
$$\hat{X}_-, \hat{X}_+ = L_\pm(x) = x + \frac{n}{-\frac{f'}{f} \pm \sqrt{(n-1)[(n-1)(-\frac{f'}{f}) - n\frac{f''}{f}]}}$$

with $\frac{f'}{f} = \frac{(\mathcal{F}_{T,S}(\lambda))'}{\mathcal{F}_{T,S}(\lambda)}$.

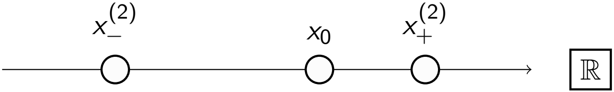
Fast and secure iterative method



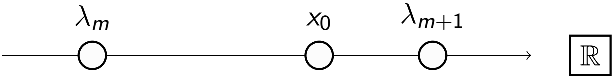
Fast and secure iterative method



Fast and secure iterative method



Fast and secure iterative method



Fast and secure iterative method

Def (Laguerre's iteration)

If $mlt(\lambda_m) = mlt(\lambda_{m+1}) = 1$ then

$$x_+^{(k)} = L_+^k(x) = L_+(L_+(\dots(x_0)))$$
$$x_-^{(k)} = L_-^k(x) = L_-(L_-(\dots(x_0)))$$

else we have a similar expression.

We can prove that

$$\lambda_m \leftarrow \dots x_-^{(2)} < x_-^{(1)} < x_0 < x_+^{(1)} < x_+^{(2)} \cdots \rightarrow \lambda_{m+1}$$

Fast and secure iterative method

It's clear that we need a powerfull method to obtain x_0 and an algorithm to estimate $mlt(\lambda_m)$.
Overstimate $mlt(\lambda_m)$ (as we can read in [2]) causes no trouble, so the most importan aspects of our calculation are: good x_0 and good evaluation of $L_\pm(x)$.

Split

Consider $(\hat{T}, \hat{S})$ with

$$\hat{T} = \begin{bmatrix} T_0 & 0 \\ 0 & T_1 \end{bmatrix}$$

$$\hat{S} = \begin{bmatrix} S_0 & 0 \\ 0 & S_1 \end{bmatrix}$$

Grazie per l'attenzione.