1 The Derivative of a Schur Decomposition

1.1 Definitions and Properties

For an arbitrary matrix B, let

$$B(\alpha) = B + \alpha \frac{\partial B}{\partial \alpha} =: B + \alpha \dot{B} \tag{1}$$

and let B^* represent the hermitian transpose of B (*matlab*'s B').

A Schur Decomposition is defined as

$$Q^* A Q = S \Leftrightarrow AQ = QS \tag{2}$$

Where Q is orthogonal, and S is block triangular.

The last piece we need is to define U as follows[1]

$$\begin{align}
Q(\alpha) &= Q + \alpha \dot{Q} \tag{3}\\
&= Q(I + \alpha \tilde{U}) \tag{4}\\
&= Q(0) \cdot U(\alpha) \tag{5}\\
U &= U^{(1)} U^{(2)} \tag{6}\\
U^{(1)} &= \begin{bmatrix} I_k & -P^* \\ P & I_{n-k} \end{bmatrix} \tag{7}\\
U^{(2)} &= \begin{bmatrix} (I_k + P^*P)^{-1/2} & 0 \\ 0 & (I_{n-k} - PP^*) \end{bmatrix} \tag{8}
\end{align}$$

Note that as $\alpha \to 0$, $U \to I$, leading to the following as $\alpha \to 0$

$$\begin{align}
&P \to 0 \tag{9}\\
&\frac{\partial}{\partial \alpha}(I_k + P^*P)^{-1/2} \to 0 \tag{10}\\
&\frac{\partial}{\partial \alpha}(I_{n-k} + PP^*)^{-1/2} \to 0 \tag{11}\\
&U(0) = I \tag{12}\\
&\left.\frac{\partial U}{\partial \alpha}\right|_{\alpha=0} = \begin{bmatrix} I_k & -\dot{P}^* \\ \dot{P} & I_{n-k} \end{bmatrix} \tag{13}
\end{align}$$

Thus

$$\dot{Q} = Q \cdot \dot{U}(0) = Q \cdot \begin{bmatrix} I_k & -\dot{P}^* \\ \dot{P} & I_{n-k} \end{bmatrix} \tag{14}$$

and finding $\dot{A}$ is now reduced to finding $\dot{P}$, which can be done as follows.

[1]Why we can make the assumption that U takes this form is beyond me. But Anderson does...

1.2 Finding $\dot{P}$

Using definition 1 in equation 2, differentiating, and evaluating at $\alpha = 0$ gives

$$\frac{\partial}{\partial\alpha}\left[(A+\alpha\dot{A})QU(\alpha) = QU(\alpha)S(\alpha)\right]_{\alpha=0} \tag{15}$$
$$\dot{A}QU(0) + AQ\dot{U}(0) = Q\dot{U}(0)S + QU(0)\dot{S} \tag{16}$$

Where $A = A(0)$, $Q = Q(0)$, and $S = S(0)$. Which left multiplying by Q^* and using $U(0) = I$ reduces to

$$Q^*\dot{A}Q + S\dot{U}(0) = \dot{U}(0)S + \dot{S} \tag{17}$$

Representing Q into $[Q_1 Q_2]$, where $Q_1 \in \mathbb{R}^{n\times(n-k)}$, and breaking S into 4 components such that $S_{1,1} \in \mathbb{R}^{k\times k}$ leads to the previous statement being represented by

$$\begin{bmatrix} Q_1^* \\ Q_2^* \end{bmatrix} \dot{A} \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} + \begin{bmatrix} S_{1,1} & S_{1,2} \\ 0 & S_{2,2} \end{bmatrix} \begin{bmatrix} I_k & -\dot{P}^* \\ \dot{P} & I_{n-k} \end{bmatrix} = \begin{bmatrix} \dot{S}_{1,1} & \dot{S}_{1,2} \\ 0 & \dot{S}_{2,2} \end{bmatrix} + \begin{bmatrix} I_k & -\dot{P}^* \\ \dot{P} & I_{n-k} \end{bmatrix} \begin{bmatrix} S_{1,1} & S_{1,2} \\ 0 & S_{2,2} \end{bmatrix} \tag{18}$$

Examining just the lower left corner of the previous equation, namely rows $n-k:n$ and columns $k:n$ leads to the following subproblem:

$$Q_2^*\dot{A}Q_1 + S_{2,2}\dot{P} = \dot{P}S_{1,1} \tag{19}$$

Which reorganized is

$$-S_{2,2}\dot{P} + \dot{P}S_{1,1}\dot{P} = Q_2^*\dot{A}Q_1 \tag{20}$$

If k is chosen such that $k = n-k$, then this is the Sylvester Equation for which exist LAPACK routines. Better yet, Octave itself can solve this for $\dot{P}$ using the command

$$\dot{P} = syl\left(-S_{2,2},\ S_{1,1},\ Q_2^*\dot{A}Q_1\right) \tag{21}$$

Using equation 14 now yields the desired $\dot{Q}$ needed for our algorithm!