CME338 Final Project
LSLQ Solver

Ron Estrin and Anjan Dwaraknath

1 Abstract

In the spirit of LSQR and LSMR, which apply CG and MINRES respectively to the normal equations, we derive LSLQ, which applies SYMMLQ to the normal equations. Starting from the normal equations, and SYMMLQ subproblem, we derive short recurrences for the computation of iterates and residual norm estimates. We then evaluate the effectiveness of SYMMLQ on some linear systems and least-squares problems, and make comparisons to LSMR.

1.1 Notation

We denote matrices by capital letters, vectors by lower case letters, and scalars are denoted by greek lower case letters. There will be some occasions where we abuse matrix notation by multiplying matrices of incorrect sizes when the matrices are meant to be applied to only a subset of the rows or columns of another matrix. We denote the Krylov subspace of size k by

$$\mathcal{K}_k = \mathcal{K}_k(A^T A, b) = \operatorname{span}\left(b, A^T A b, \ldots, (A^T A)^{k-1} b\right)$$

.

2 Derivation of LSLQ

In this section, we derive the short recurrence formulas for LSLQ beginning from the Golub-Kahan process.

2.1 Golub-Kahan Process

The Golub-Kahan process is defined by the recurrence defined in Algorithm 1.

Algorithm 1 Golub-Kahan Process

Set $\beta_1 u_1 = b$, $\alpha_1 v_1 = A^T u_1$
for $k = 1, 2, \ldots$ **do**
    $\beta_{k+1} u_{k+1} = A v_k - \alpha_k u_k$
    $\alpha_{k+1} v_{k+1} = A^T u_{k+1} - \beta_{k+1} v_k$
end for

After performing k iterations of this process, we obtain the decompositions

$$
\begin{aligned}
AV_k &= U_{k+1}B_k && (1)\\
A^T U_{k+1} &= V_{k+1}L_{k+1}^T. && (2)
\end{aligned}
$$

where $U_k = (u_1|\dots|u_k)$, $V_k = (v_1|\dots|v_k)$, and

$$
B_k = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{pmatrix} \qquad L_{k+1} = \left(B_{k+1}|\alpha_{k+1}e_{k+1}\right).
$$

We can then observe that V_k is a basis for $\mathcal{K}_k(A^TA, b)$, since

$$
A^TAV_k = A^TU_{k+1}B_k = V_{k+1}L_{k+1}^TB_k = V_{k+1}\begin{pmatrix} B_k^TB_k \\ \alpha_{k+1}\beta_{k+1}e_k^T \end{pmatrix}.
$$

2.2 The LSLQ subproblem

In order to solve the linear system or least squares problem $Ax = b$, we instead solve the normal equations $A^TA = A^Tb$. Since A^TA is symmetric positive semidefinite and we assume that the normal equations are consistent, we may consider applying SYMMLQ to this system. We solve the normal equations iteratively, where at iteration k we solve the problem

$$
\begin{aligned}
x_k = \quad \arg\min_{x\in\mathcal{K}_k} & \quad \|x\|_2 \\
\text{s.t.} & \quad A^Tr \perp \mathcal{K}_{k-1}.
\end{aligned} \tag{3}
$$

where $r = b - Ax$.

In order to solve this problem, we first note that we may formulate this as an unconstrained problem in a smaller space if we minimize in y_k and set $x_k = V_ky_k$. Then

$$
\begin{aligned}
0 = V_{k-1}^TA^Tr_k &= V_{k-1}^TA^T(b - Ax_k) \\
&= V_{k-1}^TA^Tb - V_{k-1}^TA^TAV_ky_k \\
&= \alpha_1\beta_1e_1 - B_{k-1}^TU_k^TAV_ky_k \\
&= \alpha_1\beta_1e_1 - B_{k-1}^TL_ky_k.
\end{aligned}
$$

Thus in order to solve 3, we can solve

$$
\begin{aligned}
y_k = \quad \arg\min_{y\in\mathbb{R}^k} & \quad \|y\|_2 \\
\text{s.t.} & \quad B_{k-1}^TL_ky = \alpha_1\beta_1e_1.
\end{aligned} \tag{4}
$$

2.3 First QR decomposition

We first take the QR factorization of $Q_kR_k = B_{k-1}$. Suppose we have the QR factorization of $Q_{k-1}R_{k-1} = B_{k-2}$, with

$$R_{k-1} = \begin{pmatrix} \rho_1 & \theta_2 & & \\ & \rho_2 & \ddots & \\ & & \ddots & \theta_{k-1} \\ & & & \rho_{k-2} \end{pmatrix}.$$

Then we may recurse to obtain the factorization of B_{k-1}

$$\begin{aligned} B_{k-1} &= \left(\begin{array}{cccc|c} \alpha_1 & & & & 0 \\ \beta_2 & \alpha_2 & & & \vdots \\ & \ddots & \ddots & & \vdots \\ & & \beta_{k-2} & \alpha_{k-2} & 0 \\ & & & \beta_{k-1} & \alpha_{k-1} \\ \hline 0 & \cdots & \cdots & 0 & \beta_k \end{array}\right) = Q_{k-1} \left(\begin{array}{cccc|c} \rho_1 & \theta_2 & & & 0 \\ & \rho_2 & \ddots & & \vdots \\ & & \ddots & \theta_{k-2} & 0 \\ & & & \rho_{k-2} & \theta_{k-1} \\ & & & 0 & \hat{\rho}_{k-1} \\ \hline 0 & \cdots & \cdots & 0 & \beta_k \end{array}\right) \\ &= Q_{k-1} G_k^{(1)} \left(\begin{array}{cccc|c} \rho_1 & \theta_2 & & & 0 \\ & \rho_2 & \ddots & & \vdots \\ & & \ddots & \theta_{k-2} & \vdots \\ & & & \rho_{k-2} & \theta_{k-1} \\ & & & 0 & \rho_{k-1} \\ \hline 0 & \cdots & \cdots & 0 & 0 \end{array}\right) = Q_{k-1} G_k^{(1)} \begin{pmatrix} R_{k-1} \\ \hline 0 \end{pmatrix}, \end{aligned}$$

by defining $Q_k = Q_{k-1} G_k^{(1)}$, where $G_k^{(1)}$ is the Givens rotation

$$G_k^{(1)} = \begin{pmatrix} c_1 & s_1 \\ -s_1 & c_1 \end{pmatrix}.$$

Note that we are abusing notation, where we intend $G_k^{(1)}$ to be applied to rows $k-2$ and $k-1$. Using $G_{k-1}^{(2)}$ defined in the previous iteration, we have

$$\begin{pmatrix} \theta_{k-1} \\ \hat{\rho}_{k-1} \end{pmatrix} = \begin{pmatrix} c_1^{(k-1)} & -s_1^{(k-1)} \\ s_1^{(k-1)} & c_1^{(k-1)} \end{pmatrix} \begin{pmatrix} 0 \\ \alpha_{k-1} \end{pmatrix},$$

$$\begin{pmatrix} \theta_k \\ \hat{\rho}_k \end{pmatrix} = \begin{pmatrix} c_1 & -s_1 \\ s_1 & c_1 \end{pmatrix} \begin{pmatrix} 0 \\ \alpha_k \end{pmatrix},$$

$$\begin{pmatrix} \rho_{k-1} \\ 0 \end{pmatrix} = \begin{pmatrix} c_1 & -s_1 \\ s_1 & c_1 \end{pmatrix} \begin{pmatrix} \hat{\rho}_{k-1} \\ \beta_k \end{pmatrix},$$

and we therefore obtain the recurrences

$$\hat{\rho}_{k-1} = \frac{\alpha_{k-1} \hat{\rho}_{k-2}}{\rho_{k-2}} = c_1^{(k-1)} \alpha_{k-1}, \tag{5}$$

$$\rho_{k-1} = \sqrt{\hat{\rho}_{k-1}^2 + \beta_k^2}, \tag{6}$$

$$c_1 = \hat{\rho}_{k-1} / \rho_{k-1}, \tag{7}$$

$$s_1 = -\beta_k / \rho_{k-1}, \tag{8}$$

$$\theta_k = \frac{\alpha_k \beta_k}{\rho_{k-1}} = -s_1 \alpha_k. \tag{9}$$

2.4 Forward Substitution

With the previous QR decomposition, the system we intend to solve becomes

$$\begin{aligned}
\begin{pmatrix} \alpha_1\beta_1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} &= \left(\begin{array}{c|c} B_{k-1}^T B_{k-1} & \begin{matrix} 0 \\ \vdots \\ 0 \\ \alpha_k\beta_k \end{matrix} \end{array}\right) y_k = \left(\begin{array}{c|c} R_k^T R_k & \begin{matrix} 0 \\ \vdots \\ 0 \\ \alpha_k\beta_k \end{matrix} \end{array}\right) y_k \\
&= R_k^T \left(\begin{array}{c|c} R_k & \begin{matrix} 0 \\ \vdots \\ 0 \\ \frac{\alpha_k\beta_k}{\rho_{k-1}} \end{matrix} \end{array}\right) y_k = R_k^T \left(\begin{array}{c|c} R_k & \begin{matrix} 0 \\ \vdots \\ 0 \\ \theta_k \end{matrix} \end{array}\right) y_k.
\end{aligned}$$

Define

$$z_k = \begin{pmatrix} \zeta_2 \\ \vdots \\ \zeta_k \end{pmatrix} = \left(\begin{array}{c|c} R_k & \begin{matrix} 0 \\ \vdots \\ 0 \\ \theta_k \end{matrix} \end{array}\right) y_k = \tilde{R}_k y_k \tag{10}$$

so that we have $R_k^T z_k = \alpha_1\beta_1 e_1$. As in the Conjugate Gradient method, we obtain a short recurrence for ζ_k,

$$\zeta_k = -\frac{\theta_{k-1}}{\rho_{k-1}}\zeta_{k-1}. \tag{11}$$

2.5 Second QR decomposition

Using the recurrence of the previous section, we now need to solve the minimum norm problem

$$\tilde{R}_k y_k = z_k.$$

We accomplish this by taking the QR decomposition of $\hat{Q}_k\hat{R}_k = \tilde{R}_k^T$. Suppose we have the QR decomposition from the previous iteration, $\hat{Q}_{k-1}\hat{R}_{k-1} = \tilde{R}_{k-1}^T$, with

$$\hat{R}_{k-1} = \begin{pmatrix} \sigma_1 & \eta_2 & & \\ & \sigma_2 & \ddots & \\ & & \ddots & \eta_{k-1} \\ & & & \sigma_{k-2} \end{pmatrix}.$$

Then as was done in the first QR decomposition, we can recurse to obtain a fast update for the second QR decomposition.

$$
\begin{aligned}
\tilde{R}_k^T &= \left(\begin{array}{ccccc|c}
\rho_1 & & & & & 0 \\
\theta_2 & \rho_2 & & & & \vdots \\
& \ddots & \ddots & & & \vdots \\
& & \theta_{k-2} & \rho_{k-2} & & 0 \\
& & & \theta_{k-1} & & \rho_{k-1} \\
\hline
0 & \cdots & \cdots & 0 & & \theta_k
\end{array}\right) = \hat{Q}_k \left(\begin{array}{ccccc|c}
\sigma_1 & \eta_2 & & & & 0 \\
& \sigma_2 & \ddots & & & \vdots \\
& & \ddots & \eta_{k-2} & & \vdots \\
& & & \sigma_{k-2} & & \eta_{k-1} \\
& & & 0 & & \hat{\sigma}_{k-1} \\
\hline
0 & \cdots & \cdots & 0 & & \theta_k
\end{array}\right) \\
&= \hat{Q}_k G_k^{(2)} \left(\begin{array}{ccccc|c}
\sigma_1 & \eta_2 & & & & 0 \\
& \sigma_2 & \ddots & & & \vdots \\
& & \ddots & \eta_{k-2} & & \vdots \\
& & & \sigma_{k-2} & & \eta_{k-1} \\
& & & 0 & & \sigma_{k-1} \\
\hline
0 & \cdots & \cdots & 0 & & 0
\end{array}\right)
\end{aligned}
$$

We define $\hat{Q}_k = \hat{Q}_{k-1} G_k^{(2)}$, where $G_k^{(2)}$ is the Givens rotation

$$
G_k^{(2)} = \begin{pmatrix} c_2 & s_2 \\ -s_2 & c_2 \end{pmatrix}.
$$

We are again abusing notation, so that $G_k^{(2)}$ is applied to rows $k-2$ and $k-1$. Using $G_{k-1}^{(2)}$ defined in the previous iteration, we have

$$
\begin{pmatrix} \eta_{k-1} \\ \hat{\sigma}_{k-1} \end{pmatrix} = \begin{pmatrix} c_2^{(k-1)} & -s_2^{(k-1)} \\ s_2^{(k-1)} & c_2^{(k-1)} \end{pmatrix} \begin{pmatrix} 0 \\ \rho_{k-1} \end{pmatrix},
$$

$$
\begin{pmatrix} \eta_k \\ \hat{\sigma}_k \end{pmatrix} = \begin{pmatrix} c_2 & -s_2 \\ s_1 & c_2 \end{pmatrix} \begin{pmatrix} 0 \\ \rho_k \end{pmatrix},
$$

$$
\begin{pmatrix} \sigma_{k-1} \\ 0 \end{pmatrix} = \begin{pmatrix} c_2 & -s_2 \\ s_2 & c_2 \end{pmatrix} \begin{pmatrix} \hat{\sigma}_{k-1} \\ \theta_k \end{pmatrix}.
$$

We then obtain the following recurrences,

$$
\begin{aligned}
\eta_{k-1} &= -s_2^{(k-1)} \rho_{k-1} && (12) \\
\hat{\sigma}_{k-1} &= \frac{\rho_{k-1} \hat{\sigma}_{k-2}}{\sigma_{k-2}} = c_2^{(k-1)} \rho_{k-1}, && (13) \\
\sigma_{k-1} &= \sqrt{\hat{\sigma}_{k-1}^2 + \theta_k^2}, && (14) \\
c_2 &= \hat{\sigma}_{k-1} / \sigma_{k-1}, && (15) \\
s_2 &= -\theta_k / \sigma_{k-1}, && (16) \\
\eta_k &= \frac{\rho_k \theta_k}{\sigma_{k-1}} = -s_2 \rho_k. && (17) \\
& && (18)
\end{aligned}
$$

2.6 Recurrence for x_k

We now derive a fast recurrence for x_k using the second QR decomposition. From the second QR decomposition, we have

$$\hat{R}_k^T \hat{Q}_k^T y_k = z_k.$$

Define

$$\hat{R}_k^T \hat{z}_k = z_k \tag{19}$$

$$\hat{Q}_k \begin{pmatrix} I_{k-1} \\ 0 \end{pmatrix} \hat{z}_k = y_k, \qquad \hat{z}_k = \begin{pmatrix} \hat{\zeta}_2 \\ \vdots \\ \hat{\zeta}_k \end{pmatrix} \tag{20}$$

$$W_k = V_k \hat{Q}_k = (w_2^{(k)} | \dots | w_k^{(k)}). \tag{21}$$

With these definitions, we have,

$$W_k \begin{pmatrix} I_{k-1} \\ 0 \end{pmatrix} \hat{z}_k = V_k \hat{Q}_k \begin{pmatrix} I_{k-1} \\ 0 \end{pmatrix} \hat{z}_k = V_k y_k = x_k,$$

and so

$$x_k = W_{k-1} \begin{pmatrix} I_{k-2} \\ 0 \end{pmatrix} \hat{z}_{k-1} + w_{k-1}^{(k)} \hat{\zeta}_k.$$

The recursion for $\hat{z}_k$ is similar to that of z_k, since it is a similar triangular solve via forward substitution, where we obtain

$$\hat{\zeta}_k = \frac{1}{\sigma_{k-1}} (\zeta_k - \eta_{k-1} \hat{\zeta}_{k-1}). \tag{22}$$

To get the recursion for W_k, we observe that

$$\begin{aligned} W_k &= V_k \hat{Q}_k & (23) \\ &= (V_{k-1} | v_k) \begin{pmatrix} \hat{Q}_{k-1} & \\ & 1 \end{pmatrix} \begin{pmatrix} I_{k-2} & \\ & G_k^{(2)} \end{pmatrix} & (24) \\ &= (W_{k-1} | v_k) \begin{pmatrix} I_{k-2} & \\ & G_k^{(2)} \end{pmatrix} & (25) \\ &= (w_1^{(k-1)} | \dots | w_{k-2}^{(k-1)} | w_{k-1}^{(k-1)} | v_k) \begin{pmatrix} I_{k-2} & \\ & G_k^{(2)} \end{pmatrix}. & (26) \end{aligned}$$

Then we see that the first $k-2$ columns of W_{k-1} and W_k are equal to each other, and so the only update that is required is

$$\begin{pmatrix} w_{k-1}^{(k)} & w_k^{(k)} \end{pmatrix} = \begin{pmatrix} w_{k-1}^{(k-1)} & v_k \end{pmatrix} \begin{pmatrix} c_2 & s_2 \\ -s_2 & c_2 \end{pmatrix}. \tag{27}$$

Although we compute both $w_{k-1}^{(k)}$ and $w_k^{(k)}$, we need only $w_{k-1}^{(k)}$ in order to compute x_k, while $w_k^{(k)}$ is necessary for the computation of W_{k+1}.

We summarize this procedure in Algorithm 2.

3 Norms and Stopping Criteria

Here we will derive recurrences for computing estimates of $\|r_k\|$, $\|A^T r_k\|$, $\|A\|$ and $cond(A)$.

Algorithm 2 LSLQ

$\beta_1 u_1 = b$, $\alpha_1 v_1 = A^T u_1$
$\beta_2 u_2 = A v_1 - \alpha_1 u_1$
$\alpha_2 v_2 = A^T u_2 - \beta_2 v_1$
$\rho_2 = \sqrt{\alpha_1^2 + \beta_2^2}$
$c_1^{(2)} = \alpha_1/\rho_2, \qquad s_1^{(2)} = \beta_2/\rho_2$
$\theta_2 = \alpha_2\beta_2/\rho_2$

$\zeta_2 = \alpha_1\beta_1/\rho_2$

$\hat{\sigma}_2 = \rho_2$
$\sigma_2 = \sqrt{\hat{\sigma}_2^2 + \theta_2^2}$
$c_2^{(2)} = \hat{\sigma}_2/\sigma_2, \qquad s_2^{(2)} = -\theta_2/\sigma_2$

$\hat{\zeta}_2 = \zeta_2/\sigma_2$

$\begin{pmatrix} w_1^{(2)} & w_2^{(2)} \end{pmatrix} = \begin{pmatrix} v_1 & v_2 \end{pmatrix} \begin{pmatrix} c_2^{(2)} & -s_2^{(2)} \\ s_2^{(2)} & c_2^{(2)} \end{pmatrix}$
$x_k = \hat{\zeta}_2 w_1^{(2)}$
for $k = 3, \ldots$ **do**
  $\beta_k u_k = A v_{k-1} - \alpha_{k-1} u_{k-1}$
  $\alpha_k v_k = A^T u_k - \beta_k v_{k-1}$

  $\hat{\rho}_{k-1} = c_1^{(k-1)} \alpha_{k-1}$
  $\rho_{k-1} = \sqrt{\hat{\rho}_{k-1}^2 + \beta_k^2}$
  $c_1^{(k)} = \hat{\rho}_{k-1}/\rho_{k-1}, \qquad s_1^{(k)} = \beta_k/\rho_{k-1}$
  $\theta_k = -s_k^{(k)} \alpha_k$

  $\zeta_k = -\zeta_{k-1}\theta_{k-1}/\rho_{k-1}$

  $\eta_{k-1} = -s_2^{(k-1)} \rho_{k-1}$
  $\hat{\sigma}_{k-1} = c_2^{(k-1)} \rho_{k-1}$
  $\sigma_{k-1} = \sqrt{\hat{\sigma}_{k-1}^2 + \theta_k^2}$
  $c_2^{(k)} = \hat{\sigma}_{k-1}/\sigma_{k-1}, \qquad s_2^{(k)} = -\theta_k/\sigma_{k-1}$

  $\hat{\zeta}_k = (\zeta_k - \eta_{k-1}\hat{\zeta}_{k-1})/\sigma_{k-1}$

  $\begin{pmatrix} w_{k-1}^{(k)} & w_k^{(k)} \end{pmatrix} = \begin{pmatrix} w_{k-1}^{(k-1)} & v_k \end{pmatrix} \begin{pmatrix} c_2^{(k)} & -s_2^{(k)} \\ s_2^{(k)} & c_2^{(k)} \end{pmatrix}$
  $x_k = x_{k-1} + \hat{\zeta}_k w_{k-1}^{(k)}$
end for

3.1 Recurrence for y_k

Here we derive a recurrence for the last two entries of y_k, which will be used in the estimates of the norm quantities in which we are interested. From equation 20, we have that

$$\begin{aligned}
y_k &= \hat{Q}_k \begin{pmatrix} I_{k-1} \\ 0 \end{pmatrix} \begin{pmatrix} \hat{\zeta}_2 \\ \vdots \\ \hat{\zeta}_k \end{pmatrix} \\
&= \begin{pmatrix} G_{k-1}^{(2)} & \\ & 1 \end{pmatrix} \begin{pmatrix} I_{k-2} & \\ & G_k^{(2)} \end{pmatrix} \begin{pmatrix} \hat{\zeta}_2 \\ \vdots \\ \hat{\zeta}_k \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} \times & \cdots & \times \\ \times & \cdots & \times \\ & \ddots & \vdots \\ & s_{k-1}^{(2)} & -c_{k-1}^{(2)} c_k^{(2)} \\ & 0 & s_k^{(2)} \end{pmatrix} \begin{pmatrix} \hat{\zeta}_2 \\ \vdots \\ \hat{\zeta}_k \end{pmatrix}.
\end{aligned}$$

Thus we can obtain the last 2 entries of y_k from

$$\begin{pmatrix} \psi_1^{(k)} \\ \psi_2^{(k)} \end{pmatrix} = \begin{pmatrix} s_{k-1}^{(2)} & -c_{k-1}^{(2)} c_k^{(2)} \\ 0 & s_k^{(2)} \end{pmatrix} \begin{pmatrix} \hat{\zeta}_{k-1} \\ \hat{\zeta}_k \end{pmatrix}. \tag{28}$$

3.2 Recurrence for $\|r_k\|$

We observe the following relationships based on the equations

$$\tilde{R}_k y_k = z_k \tag{29}$$
$$R_{k+1}^T z_{k+1} = \alpha_1 \beta_1 e_1 \tag{30}$$
$$Q_{k+1} R_{k+1} = B_k. \tag{31}$$

Taking the transpose of the second equation and defining $q^{(k+1)} = \beta_1 Q_{k+1}^T e_1$, we see that

$$R_{k+1}^T Q_{k+1}^T = B_k^T \tag{32}$$
$$R_{k+1}^T q^{(k+1)} = \alpha_1 \beta_1 e_1. \tag{33}$$

We can obtain a recurrence for $q^{(k+1)}$ by observing that

$$\begin{aligned}
q^{(k+1)} &= \beta_1 Q_{k+1}^T e_1 \\
&= \beta_1 \begin{pmatrix} I_{k-2} & \\ & \left(G_{k+1}^{(1)}\right)^T \end{pmatrix} \begin{pmatrix} Q_k^T & \\ & 1 \end{pmatrix} e_1 \\
&= \begin{pmatrix} I_{k-2} & \\ & \left(G_{k+1}^{(1)}\right)^T \end{pmatrix} \begin{pmatrix} q^{(k)} \\ 0 \end{pmatrix} = \begin{pmatrix} q_1^{(k)} \\ \vdots \\ q_{k-1}^{(k)} \\ q_k^{(k)} c_1^{(k+1)} \\ -q_k^{(k)} s_1^{(k+1)} \end{pmatrix}.
\end{aligned}$$

Now, since R_{k+1}^T is nonsingular, we have that

$$q^{(k+1)} = \begin{pmatrix} z_{k+1} \\ -q_k^{(k)} s_1^{(k+1)} \end{pmatrix}. \tag{34}$$

We now use this relationship to obtain a short recurrence for $\|r_k\|$. Thus

$$\begin{aligned}
\|r_k\| &= \|b - Ax_k\| \\
&= \|U_{k+1}(\beta_1 e_1 - B_k y_k)\| \\
&= \|Q_{k+1}^T(\beta_1 e_1 - B_k y_k)\| \\
&= \left\| \beta_1 Q_{k+1}^T e_1 - \begin{pmatrix} R_{k+1} \\ 0 \end{pmatrix} y_k \right\| \\
&= \left\| \beta_1 Q_{k+1}^T e_1 - \begin{pmatrix} R_{k+1} \\ 0 \end{pmatrix} y_k \right\| \\
&= \left\| \beta_1 Q_{k+1}^T e_1 - \begin{pmatrix} z_k \\ \rho_k \psi_2^{(k)} \\ 0 \end{pmatrix} \right\| \\
&= \left\| \begin{pmatrix} 0 \\ -q_k^{(k)} c_1^{(k+1)} - \rho_k \psi_2^{(k)} \\ q_k^{(k)} s_1^{(k+1)} \end{pmatrix} \right\|.
\end{aligned}$$

Since we require only the last 2 entries of $q^{(k+1)}$ for which we have a fast recurrence, the computation of $\|r_k\|$ can be achieved in $O(1)$ flops. Note that in order to estimate the residual at iteration k, we need values computed at iteration $k+1$.

3.3 Recurrence for $\|A^T r_k\|$

We can obtain an estimate of $\|A^T r_k\|_2$ with $O(1)$ flops. We have

$$\begin{aligned}
\|A^T r_k\| &= \|A^T(b - Ax_k)\| \\
&= \|V_{k+1}(\alpha_1 \beta_1 e_1 - L_{k+1}^T B_{k+1} y_k)\| \\
&= \|\alpha_1 \beta_1 e_1 - L_{k+1}^T B_{k+1} y_k\|.
\end{aligned}$$

We note that

$$\begin{aligned}
L_{k+1}^T B_{k+1} y_k &= \left(\begin{array}{ccc|c} & & & 0 \\ & B_{k-1}^T & & \vdots \\ & & & 0 \\ \hline 0 & \cdots & \alpha_k & \beta_{k+1} \\ \hline 0 & \cdots & 0 & \alpha_{k+1} \end{array}\right) \begin{pmatrix} L_k^T \\ \beta_{k+1} e_k^T \end{pmatrix} y_k \\
&= \begin{pmatrix} B_{k-1}^T L_k \\ \alpha_k \beta_k e_{k-1}^T + (\alpha_k^2 + \beta_{k+1}^2) e_k^T \\ \alpha_{k+1} \beta_{k+1} e_k^T \end{pmatrix} y_k \\
&= \begin{pmatrix} \alpha_1 \beta_1 e_1 \\ (\alpha_k \beta_k e_{k-1} + (\alpha_k^2 + \beta_{k+1}^2) e_k)^T y_k \\ \alpha_{k+1} \beta_{k+1} e_k^T y_k \end{pmatrix}.
\end{aligned}$$

Thus we have

$$\|\alpha_1\beta_1 e_1 - L_{k+1}^T B_{k+1} y_k\| = \left\| \begin{pmatrix} 0 \\ (\alpha_k\beta_k e_{k-1} + (\alpha_k^2 + \beta_{k+1}^2)e_k)^T y_k \\ \alpha_{k+1}\beta_{k+1} e_k^T y_k \end{pmatrix} \right\|,$$

and so we need only the last 2 entries of y_k which we can obtain in $O(1)$ flops as described in Section 4.1.

3.4 Estimate of $\|A\|$ and $cond(A)$

As in LSMR, we may estimate $\|A\|$ by using $\|B_k\|_F$. In order to estimate cond(A), we note that cond$(A)^2$ = cond(A^TA), and that we may estimate cond(A^TA) by cond$(B_k^TB_k)$. Since $B_k^TB_k = R_{k+1}^TR_{k+1}$, we can estimate cond$(A)$ by cond(R_k) at each iteration. cond(R_k) may the be estimated by the largest and smallest entries on its diagonal, that is, $\max\sigma_i/\min\sigma_i$.

4 Numerical Experiments

In this section we apply LSLQ to some linear systems and least-squares problems to evaluate its effectiveness. LSLQ has been implemented in Matlab for these experiments.