4. The GMRES algorithm: An iterative least-squares algorithm

- We start with some initial approximation (guess) $\mathbf{x}_0$ (it is arbitrary; it can be zero or random vector). We also have as input the tolerance ϵ (e.g., $\epsilon = 10^{-6}$), and an upper bound of the number of iteration steps to be performed, $m_{\max}$ (e.g., $m_{\max} = 50$).
- We compute the initial residual $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$.
- We form $\mathbf{p}_1 = \frac{1}{\|\mathbf{r}_0\|}\mathbf{r}_0$.
- Compute $\mathbf{b}_1 = A\mathbf{p}_1$.
- Solve the least-squares problem $\|\mathbf{r} - t\mathbf{b}_1\| \mapsto \min$, which gives $t = \frac{\mathbf{b}_1^T\mathbf{r}}{\mathbf{b}_1^T\mathbf{b}_1}$.
  (1) The next iterate is
  $$\mathbf{x}_1 = \mathbf{x}_0 + t\mathbf{p}_1,$$
  and
  (2) the next residual is
  $$\mathbf{r}_1 = \mathbf{r}_0 - t\mathbf{b}_1 \ (= \mathbf{b} - A\mathbf{x}_1).$$
- **Loop:** For $m = 2, 3, \ldots$ until convergence and $m \le m_{\max}$.
  At step $m \ge 2$, we construct a vector $\mathbf{p}_m$ which is orthogonal at all previous vectors $\mathbf{p}_1, \ldots \mathbf{p}_{m-1}$, using $\mathbf{r}_{m-1}$ as follows
  $$\widetilde{\mathbf{p}}_m = \mathbf{r}_{m-1} - \beta_1\mathbf{p}_1 - \beta_2\mathbf{p}_2 - \ldots - \beta_{m-1}\mathbf{p}_{m-1}.$$
  The constants β are computed from the orthogonality conditions $\mathbf{p}_i^T\widetilde{\mathbf{p}}_m = 0$, $i = 1, 2, \ldots, m-1$. The latter conditions give,
  $$\beta_i = \mathbf{p}_i^T\mathbf{r}_{m-1}, \text{for } i = 1, \ldots, m-1.$$
  Then
  $$\mathbf{p}_m = \frac{1}{\|\widetilde{\mathbf{p}}_m\|}\widetilde{\mathbf{p}}_m.$$
- Form $P = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_m]$.
- Compute $\mathbf{b}_m = A\mathbf{p}_m$ and form $B = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m]$ $(= AP)$.
- Solve the least-quares problem
  $$\|\mathbf{r}_{m-1} - B\mathbf{t}\| \mapsto \min.$$
  This can be done by using QR factorization of B.
  (1) Compute $B = QR$.
  (2) Solve $R\mathbf{t} = Q^T\mathbf{r}_{m-1}$ by back substitution.
- The next iterate is
  $$\mathbf{x}_m = \mathbf{x}_{m-1} + P\mathbf{t}.$$
- The next residual is
  $$\mathbf{r}_m = \mathbf{r}_{m-1} - B\mathbf{t} \ (= \mathbf{b} - A\mathbf{x}_m).$$
- Stopping criterion: Check if $\|\mathbf{r}_m\| < \epsilon$ (ϵ is an input desired tolerance). If **yes** exit. Otherwise set $m := m + 1$ and if $m \le m_{\max}$ repeat by going to **Loop**. Otherwise, either exit, or (optionally) go to the beginning of the algorithm to **restart** by setting $\mathbf{x}_0 = \mathbf{x}_{m_{\max}}$ to compute the restarted initial residual $\mathbf{r}_0 := \mathbf{b} - A\mathbf{x}_0$.

5. Discussion

We go over the algorithm and highlight all the steps and operations needed to implement this algorithm.

Note that by construction, we have the monotinicity property

$$\|\mathbf{r}_m\| = \min_{\mathbf{t}} \|\mathbf{r}_{m-1} - B\mathbf{t}\| \leq \|\mathbf{r}_{m-1}\|.$$

I.e.,

$$\|\mathbf{r}_0\| \geq \|\mathbf{r}_1\| \geq \ldots \geq \|\mathbf{r}_m\| \geq \ldots$$

Also, by choosing the size of $m_{\max}$, we control the number of columns of the matrix B, which impacts the work required to compute its QR factorization. A lot of savings can be taken advantage of, using the fact that B at step m, is obtained from B at step $m-1$ by adding only one new column. This only adds only one step to the Gram-Schmidt algorithm done at the previous iteration (i.e., we do not have to start from the beginning).

The algorithm needs to allocate memory for storing $\mathbf{x} := \mathbf{x}_m$ (one vector), $\mathbf{r} := \mathbf{r}_m = \mathbf{b} - A\mathbf{x}_m$ and $m_{\max}$ vectors for $P = [\mathbf{p}_1, \ldots, \mathbf{p}_m]$. We also need to store the m columns of Q (from the QR factorization of B). The matrix R requires small amount of memory, $\frac{m(m+1)}{2}$, which is much smaller than nm (the memory required for P, for example). Compare these values for $n = 10^6$ and $m = 50$ (1275 versus 50 million).