Weekly Report for YiYu Shi's work in week1

August 27, 2005

1 Work1: Implement SMOR, ENOR in MATLAB

The first work of this week is to read four papers on model order reduction methods such as PRIMA, SAPOR, SMOR, and ENOR. Implement two already existed model reduction method, namely SMOR and ENOR, in MATLAB. These two algorithms are both for second order formulations. ENOR uses implicitly moment matching and thus suffering from numerical instability. SMOR employs the Krylov subspaces method and therefore is more stable. I implemented these two algorithms and part of the results are shown in the figures below.

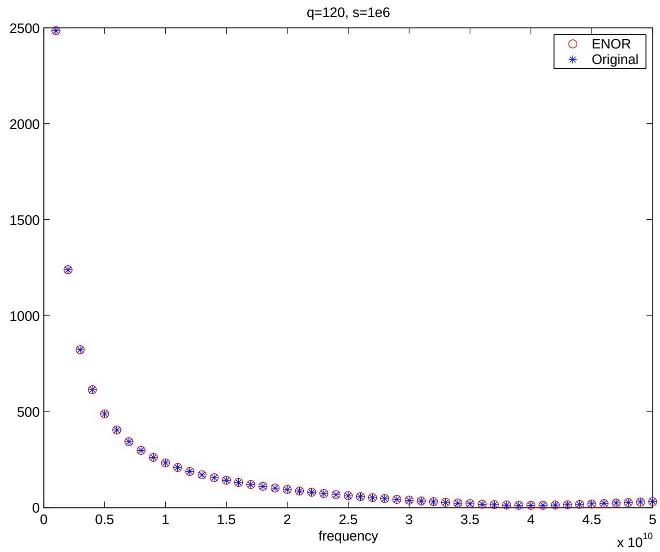

Figure 1: MOR results from ENOR

During my implementation, I found the following problems:

A. The accuracy of all the algorithms for the second order formulations are related to the selection of expansion point, s0. In my experiment, I observed that generally speaking, the larger s0 is, the less singular the iteration matrix A will be. Among these methods, SAPOR is most sensitive on the selection of s0 and it puts a high requirment on s0.

B. It is also difficult to find residues and poles for these second order reduction methods. That is to say, it is difficult to obtain their time domain responses directly. SMOR seems to have provided a method to obtain residues and poles, but experiment results show that the poles thus obtained have been shifted due to the Im variable substitution presented in the paper. So the time domain response is not correctly if we use these residues and poles directly.

C. There are also numerical instability problems in SAPOR method. Actually I observed that the obtained orthonormal projection matrix V is not strictly orthogonal. With the increase of iteration number, the inner products between the newly generated vector and the other vectors are increasing. I found out that the problem

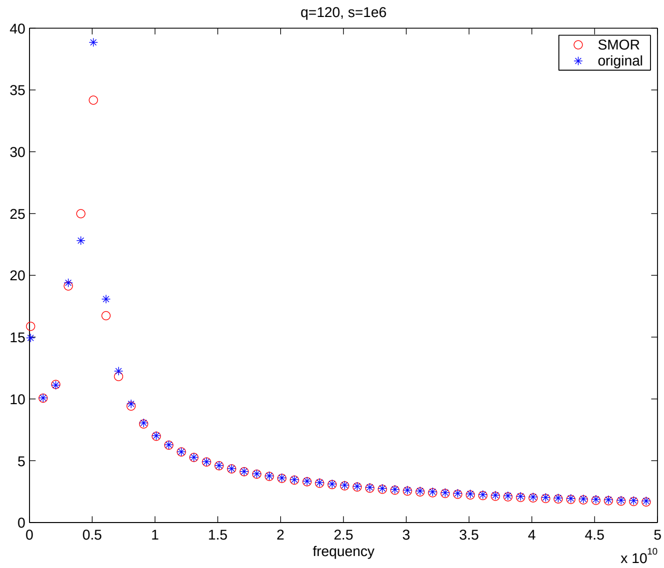

Figure 2: MOR results from SMOR

should be ascribed to the singularity of iteration matrix A. And the singularity of A is due to the singularity of G and Γ.

D. The SAPOR method can be generalized to solve three-order formulations. This idea will be empolyed in the PWL current source, which is also a part of the work this week and is presented in section three.

E. The expansion frequency s0 should be real to guarantee the result quality. No paper explicitly mentioned this.

2 Work2: Fix Jun Chen's SAPOR code

In this work, I tried to read the SAPOR code implemented by Jun Chen and fix some little bugs in his code. I mainly found three bugs as following:

A. The method for C0, G0 matrices extraction is not strictly correct. It might produce wrong results in certain cases. So I changed the extraction method to fix it.

B. The orignal code forgot to update q0 value after each iteration.

C. There is also a small problem in output expressions.

The results from the fixed code are shown below.

3 Work3: Start to work on the Model Order Reduction for RCS Circuit with Multiple PWL current sources

I read the three papers assigned by Hao and found that all the existing reduction methods are not satisfying for the following reasons.

A. Only EKS method uses implicit moment matching. The remaining method are numerically instable.

B. Although EKS is more stable, it only performs reduction on first-order formulation. That is to say, no method yet has been proposed to perform model order reduction on RCS circuit.

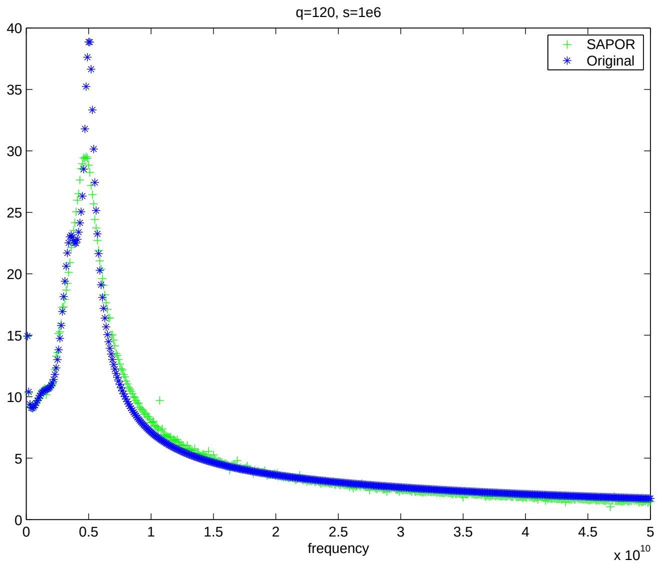

Figure 3: MOR results from SAPOR

The PWL current source can generally be approximated as: $I(s) = \frac{b_{-2}}{s^2} + \frac{b_{-1}}{s} + b_0 + b_1 s + ... + b_k s^k$

So we have $(s^2C + sG + \Gamma)V(s) = B(\frac{b_{-2}}{s^2} + \frac{b_{-1}}{s} + b_0 + b_1 s + ... + b_k s^k)$

I have roughly finished the reduction procedure. The main idea is that we can divide the V(s) into two parts according to the superposition characteristic of linear equations.

$(s^2C + sG + \Gamma)V_1(s) = B(\frac{b_{-2}}{s^2} + \frac{b_{-1}}{s} + b_0)$

$(s^2C + sG + \Gamma)V_2(s) = B(b_1 s + ... + b_k s^k)$

$V(s) = V_1(s) + V_2(s)$

We can perform explicit moment matching for the second equation, as it only contains higher order of s (the time domain response is mainly decided by the main poles, namely $\frac{(}{1})(s^2)$ and $\frac{(}{1})(s)$ terms) and therefore will not bring great error to the final result even there are some accumulate errors during the iterations. The first equation is solved by implicitly moment matching, quite similar to SAPOR. The detailed procedure is shown in figure 4.

In the next week, I'll try to conduct some further research on it.

suppose PLW current source with the form: $J(s) = \frac{J_0}{s^2} + \frac{J_1}{s} + J_2$
(not necessary to introduce higher order s^n in practice)

So the nodal equation for an RCS circuit can be rewritten as

$$(s^2C + sG + \Gamma)V(s) = sB\left(\frac{J_0}{s^2} + \frac{J_1}{s} + J_2\right)$$

$$\Rightarrow \cancel{(s^4C + s^3G + s^2\Gamma)V(s) = s}$$

$$\Rightarrow (s^3C + s^2G + s\Gamma)V(s) = BJ_0 + BJ_1 s + BJ_2 s^2$$

$$(s = s_0 + \delta)$$

$$\Rightarrow [(s_0+\delta)^3C + (s_0+\delta)^2G + (s_0+\delta)\Gamma]V(\delta) = BJ_0 + BJ_1(s_0+\delta) + BJ_2(s_0+\delta)^2$$

introduce auxilary matrix D, E, F and b_0, b_1, b_2

$$\Rightarrow [C\cdot\delta^3 + D\delta^2 + E\delta + F]V(\delta) = b_0 + b_1\delta + b_2\delta^2 \quad (*)$$

~~Use~~ two-step procedure to linearize $(*)$

First introduce a new variable $Y(\delta)$, satisfying

$$\delta\cdot CV(\delta) + Y(\delta) = b_2 \quad (1)$$

$$\Rightarrow -Y(\delta)\cdot\delta^2 + D\cdot V(\delta)\cdot\delta^2 + E\delta\cdot V(\delta) + F\cdot V(\delta) = b_0 + b_1\delta$$

Then introduce $Z(\delta)$:

$$\delta[D V(\delta) - Y(\delta)] + Z(\delta) = b_1 \quad (2)$$

$$\Rightarrow -\delta\cdot Z(\delta) + E\delta\cdot V(\delta) + F\cdot V(\delta) = b_0 \quad (3)$$

combine (1), (2) and (3), we can get

$$\begin{bmatrix} \delta E + F & \delta I & 0 \\ \delta D & I & -\delta I \\ \delta C & 0 & I \end{bmatrix} \begin{bmatrix} V \\ Z \\ Y \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix}$$

$$\Rightarrow (I - \delta\cdot A)\cdot\begin{bmatrix} Y \\ Z \\ Y \end{bmatrix} = \begin{bmatrix} r_0 \\ q_0 \\ p_0 \end{bmatrix} \quad (r_0 = F^{-1}\cdot b_0)$$

After we get A, follow the Generalized SOAR algs in SAPOR and we can simply obtain projection matrix Q_n.

Figure 4: Procedure