EE 201C

Project 1 (due Feb 8)

Wei Wu

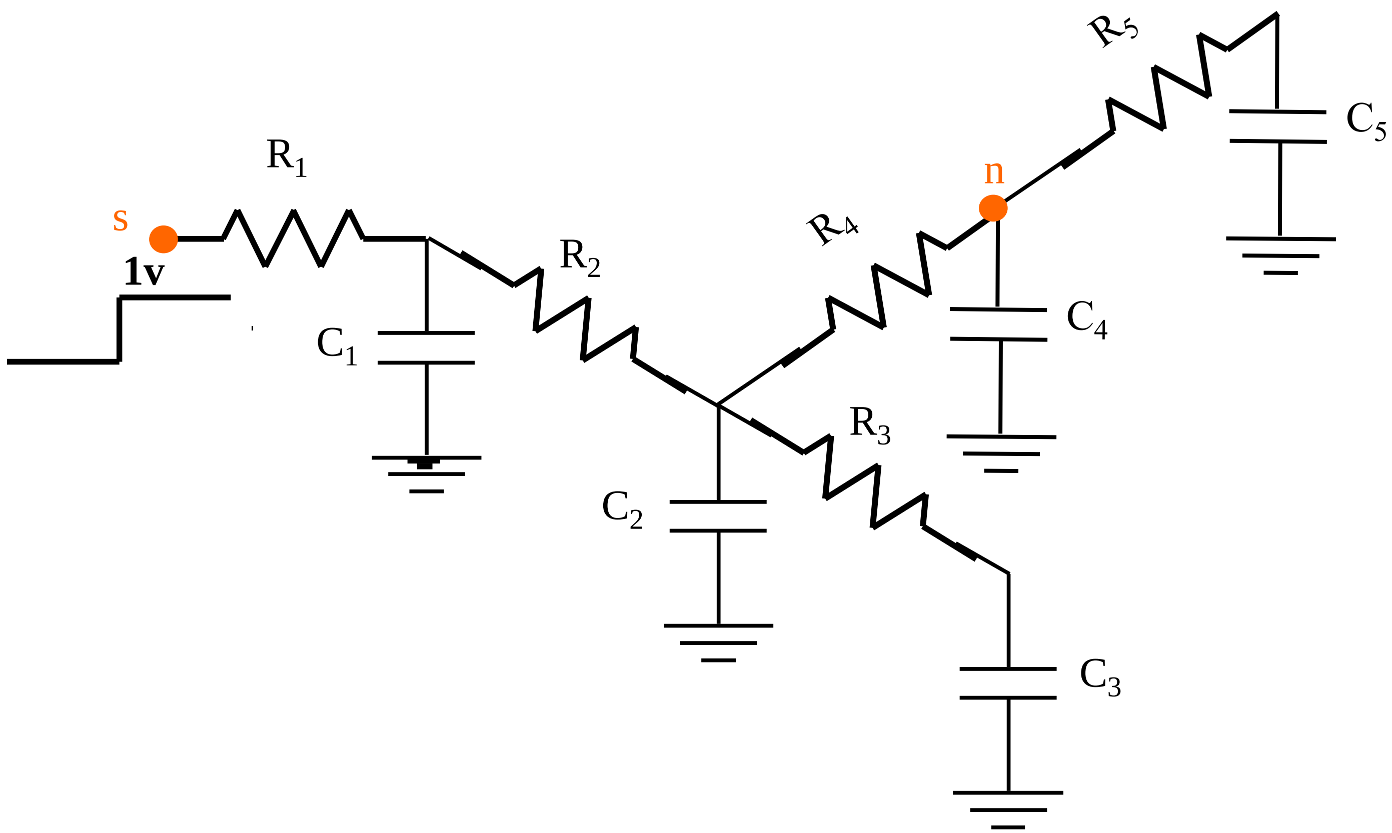
Submit code and report to:

Xiao shi(xshi2091@gmail.com)

Email Subject: EE201C_PRJ1_Name_UID

Project 1 [due Feb 8]

[Problem #1] For the same circuit, use DC analysis method in SPICE to get the 0th -3rd moments for C4.



$R_1 = 2m\Omega$
$R_2 = 2m\Omega$
$R_3 = 3m\Omega$
$R_4 = 3m\Omega$
$R_5 = 4m\Omega$

$C_1 = 2nF$
$C_2 = 2nF$
$C_3 = 4nF$
$C_4 = 4nF$
$C_5 = 2nF$

Steps for Problem 1

1. Follow the DC analysis method to reconstruct the circuit (e.g. replace C with zero current source for 0th moment calculation, etc).

2. Write the corresponding netlist for SPICE analysis.

3. Run DC analysis in SPICE to get the voltage across the capacitance as the moment.

4. use the above moments to approximate the two pole model V4 = k1/(s-p1) + k2 / (s-p2) for voltage at C4 under a unit step input at the root of the tree (hint: compare this with SPICE simulation helps to debug the moments you calculated, refer to the S2P paper).

5 Use the frequency domain expression ($\hat{h}(s)$) to derive the time domain expression ($\hat{h}(t)$).

6 Plot the obtained time domain waveform to get the 50% delay for the S2P model.

7. Run Transient simulation in HSPICE and measure the 50% delay, compare it with the delay calculated in step 4.

Project 1 [due Feb 8]

[Problem #2] Modify the PRIMA code with single frequency expansion to multiple points expansion. You should use a vector fspan to pass the frequency expansion points. Compare the waveforms of the reduced model between the following two cases:

1. Single point expansion at s=1e4.

2. Four-point expansion at s=1e3, 1e5, 1e7, 1e9.

Matlab Files

We provide two matlab files:

- prima.m

  PRIMA on single point expansion

- demo2_11.m

  perform single-point MOR, calculate and compare corresponding time and frequency domain response between original matrix and MATLAB reduced matrix. prima function is called.

Format of the input matrices for test

```
1 1  19.4595 1.43391e-14
1 2  0.000464141 -2.9702e-15
1 3 -0.000542882 0.0
1 4  0.000152585 -7.5288e-15
1 5  0.000464074 -2.9702e-15
1 6 -0.000542801 0.0
1 68 -19.4595 0.0
2 1  0.0  -2.9702e-15
2 2  3.66672 2.44291e-13
2 3  0.0  -2.3594e-13
2 4  0.0  -5.3806e-15
2 72 -1.425 0.0
2 329 -2.06075 0.0
2 341 -0.091255 0.0
2 343 -0.0897199 0.0
3 1 -2.44188e-06 0.0
3 2  -0.000464141 -2.3594e-13
3 3  40.8898 2.42089e-13
.....
```

The input files GC8 and GC9 each has 4 columns. They are:

row number m, column number n, (m,n) entry in G matrix - G(m,n), (m,n) entry in C matrix - C(m,n) .

If both G(m,n) and C(m,n) are zero, that entry is omitted in input file.

Editor - C:\Users\Yiyu Shi\Desktop\PRIMA\PRIMA\demo2_11.m*

File Edit Text Go Cell Tools Debug Desktop Window Help

Stack: Base

```
clear;

%load G,C matrices
a=load('GC9.txt');
G=sparse(a(:,1),a(:,2),a(:,3));
C=sparse(a(:,1),a(:,2),a(:,4));
kv=3;   %number of current sources (number of input ports)

%define matched moments, size of original system, number of input sources
q=5;          %define matched moments
N=size(G,1);
gmin=0; %perturbation to make reduced G nonsigular

%define start frequency, end frequency, and step
fspan=[1e6];      %span vectors at the frequency of 1MHz
fe=1e9;

%set parameters for time domain response calculation
h=2e-9;       %step for back-ward time domain simulation
inputno=8000;     %the number of time steps for input
outputno=inputno;  %the number of time steps for output; for simplicity we make it equal to the number for inpu

%Generate L
L=zeros(N,1);
L(8)=1;

%Generate B
B=zeros(size(G,1),kv);
B(6,1)=-1;
B(10,2)=-1;
B(11,3)=-1;

%Generate U
```

demo2_11.m* | prima.m

Editor - C:\Users\Yiyu Shi\Desktop\PRIMA\PRIMA\demo2_11.m

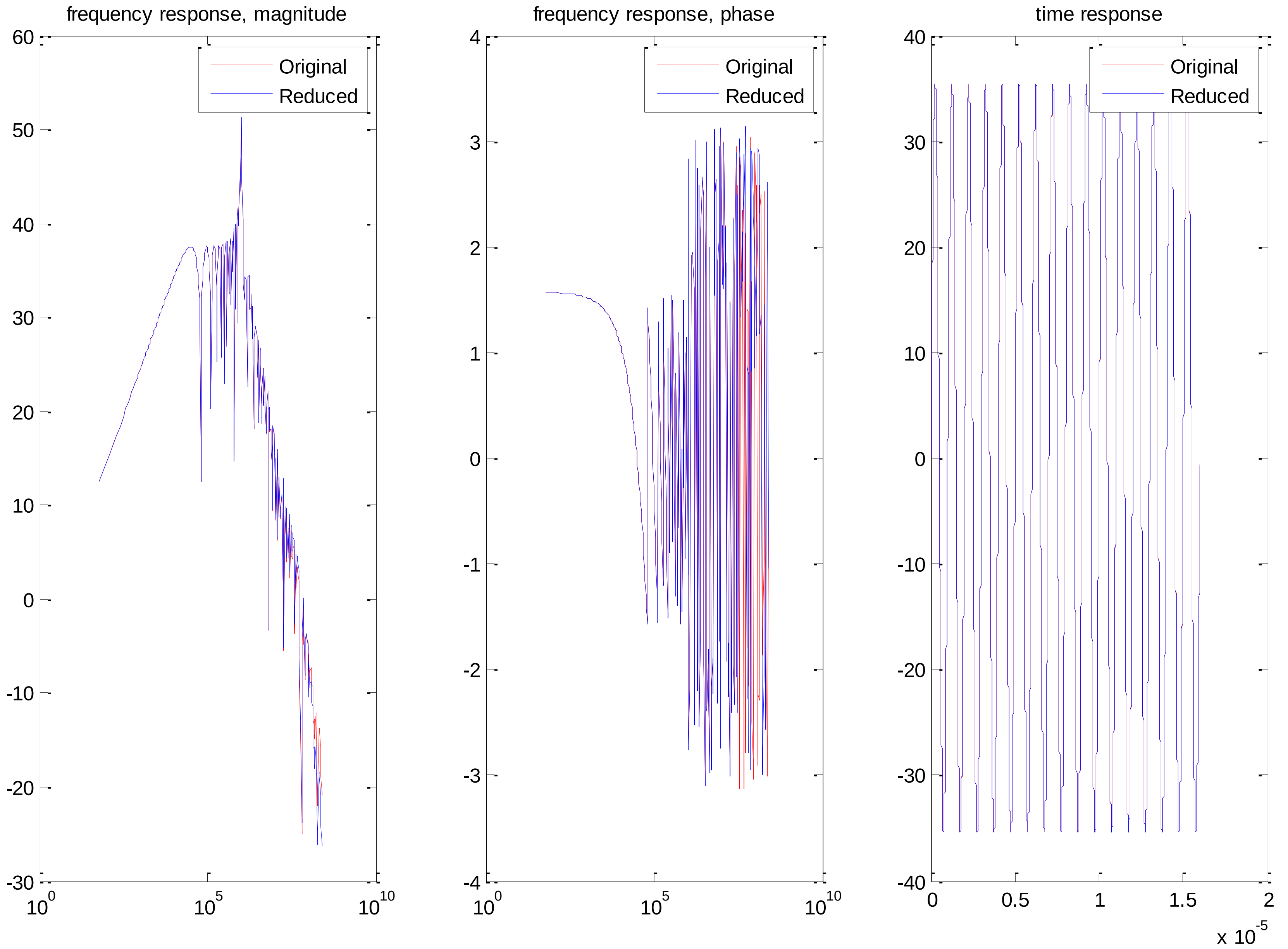
File Edit Text Go Cell Tools Debug Desktop Window Help

Stack: Base

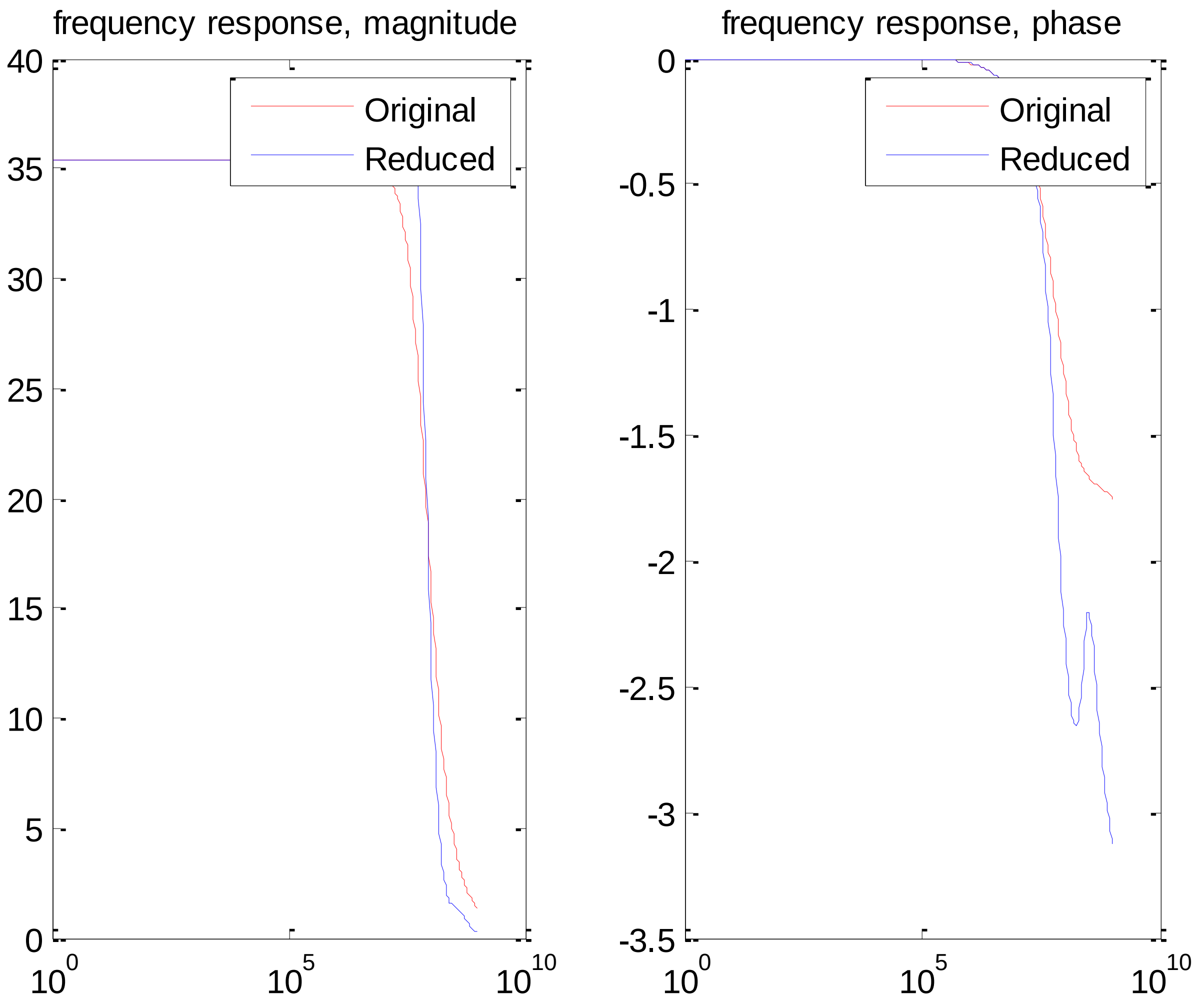
```
%
%Voltage source 1       0V    --------------------
%Voltage source 2        0.5V*sin(2*pi*1MHz*t)
%Voltage source 3       -0.5V*sin(2*pi*1MHz*t)
U=[0*[h:h:h*inputno];0.5*sin(2*pi*1e6*[h:h:h*inputno]);-0.5*sin(2*pi*1e6*[h:h:h*inputno])];
Us=fft(U,inputno*1000,2);
f=1/h/2*linspace(0,1,inputno*1000/2);

G=G+gmin*eye(length(G));

%Prima reduction
fprintf('\n\n\n\n');
fprintf('G,C,B,U,L matrices have been generated.');

fprintf('\n');
fprintf('Prima begins:\n');
tic
[Gr,Cr,Br,Lr,V]=prima(G,C,B,L,q,2*pi*fspan,gmin);
toc
fprintf('Prima done!\n');

%calculate original time domain response
fprintf('\n');
fprintf('Calculate original time domain response:\n');
tic
vo=-1e-9*ones(N,1);
A=G+1/h*C;
[LA,UA]=lu(A);

for j=1:outputno
    if (j<=inputno)
        b=1/h*C*vo(:,end) + B*U(:,j);
    else
```

demo2_11.m   prima.m

Frequency and time domain response for single point expansion



Impulse response for single point expansion



THANK YOU!

Due on Feb 8, 2016

Submit code and report to:

Xiao shi(xshi2091@gmail.com)

Email Subject: EE201C_PRJ1_Name_UID

PRIMA review

- Any RLC circuit can be represented by a first order differential equation

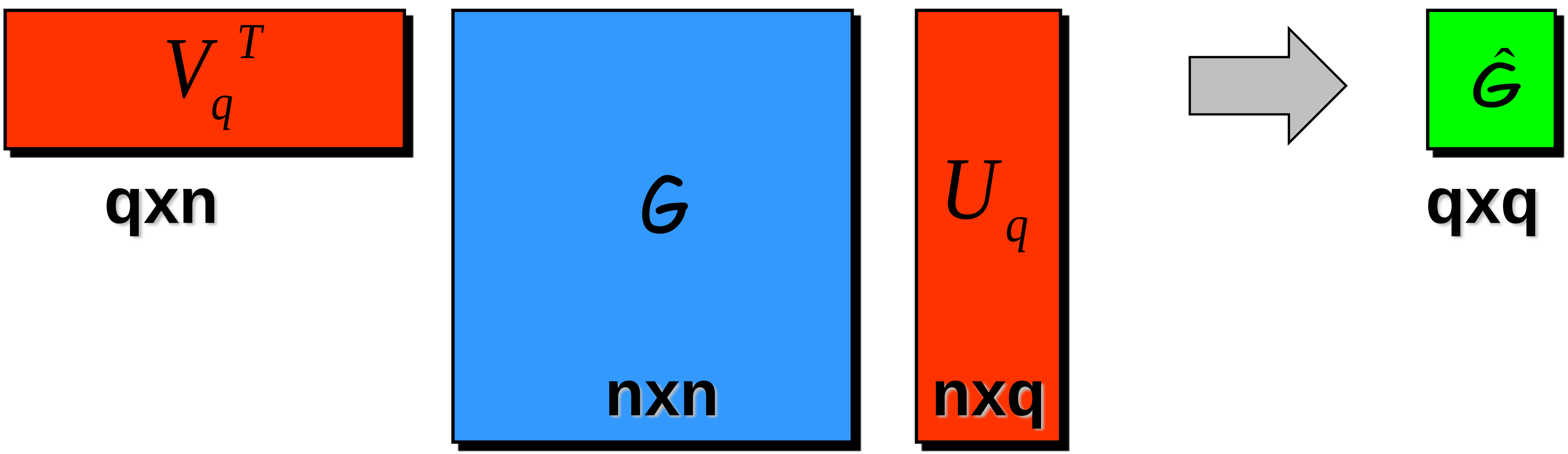
$$G\,x(t) + C\,\frac{dx(t)}{dt} = B\,u(t)$$

- $(G + sC)x(s) = Bu(s)$ (Laplace(s) domain)

- Can we reduce the equation size?
  - Reduce the number of variables (column # of G and C)
  - Reduce the number of equations (row # of G and C)

Projection Framework

$$(G+sC)x = Bu \;\Rightarrow\; (\overbrace{\boxed{V_q^T G U_q}}^{\hat{G}} + s\overbrace{\boxed{V_q^T C U_q}}^{\hat{C}})x = \overbrace{\boxed{V_q^T B}}^{\hat{B}}u$$



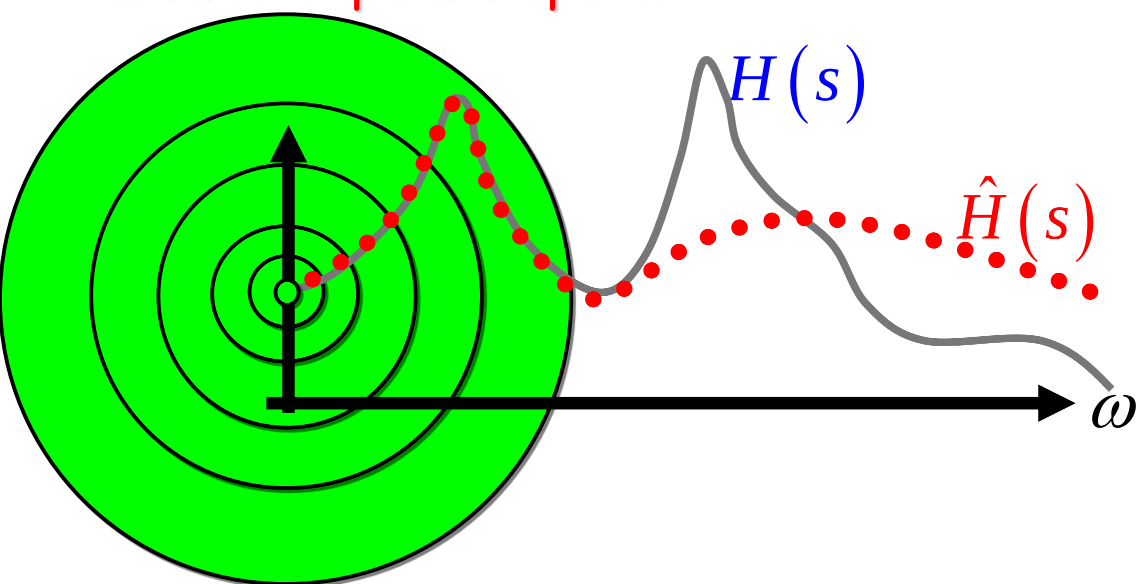
Intuitive view of Krylov subspace choice for U_q

$$(G+sC)x = Bu \quad \Rightarrow \quad x = (G+sC)^{-1}Bu$$

Taylor series expansion: $A=-G^{-1}C$, $R=G^{-1}B$ $\quad V_q^T(G+sC)U_q x = V_q^T Bu$

$$x = (R + sAR + s^2A^2R + \ldots)u \quad \Rightarrow \quad x \in \mathrm{span}\{R, AR, A^2R, \ldots\}$$

- change base and use only the first few vectors of the Taylor series expansion: equivalent to match first derivatives around expansion point



$$\left[\, x \,\right] = \left[\, R, \quad AR, \quad A^2R, \quad \ldots \,\right] \left[\, \hat{x} \,\right]$$

Orthonormalization of U_q: The Arnoldi Algorithm

$$\vec{u}_1 = R/||R||$$

For i = 1 to q-1

$$\vec{u}_{i+1} = A\vec{u}_i$$

Generates k+1 vectors!

For j = 1 to i

$$\vec{u}_{i+1} \leftarrow \vec{u}_{i+1} - (\vec{u}_{i+1}^T \vec{u}_j)\vec{u}_j$$

Orthogonalize new vector: Remove the projection on other normalized vectors

end

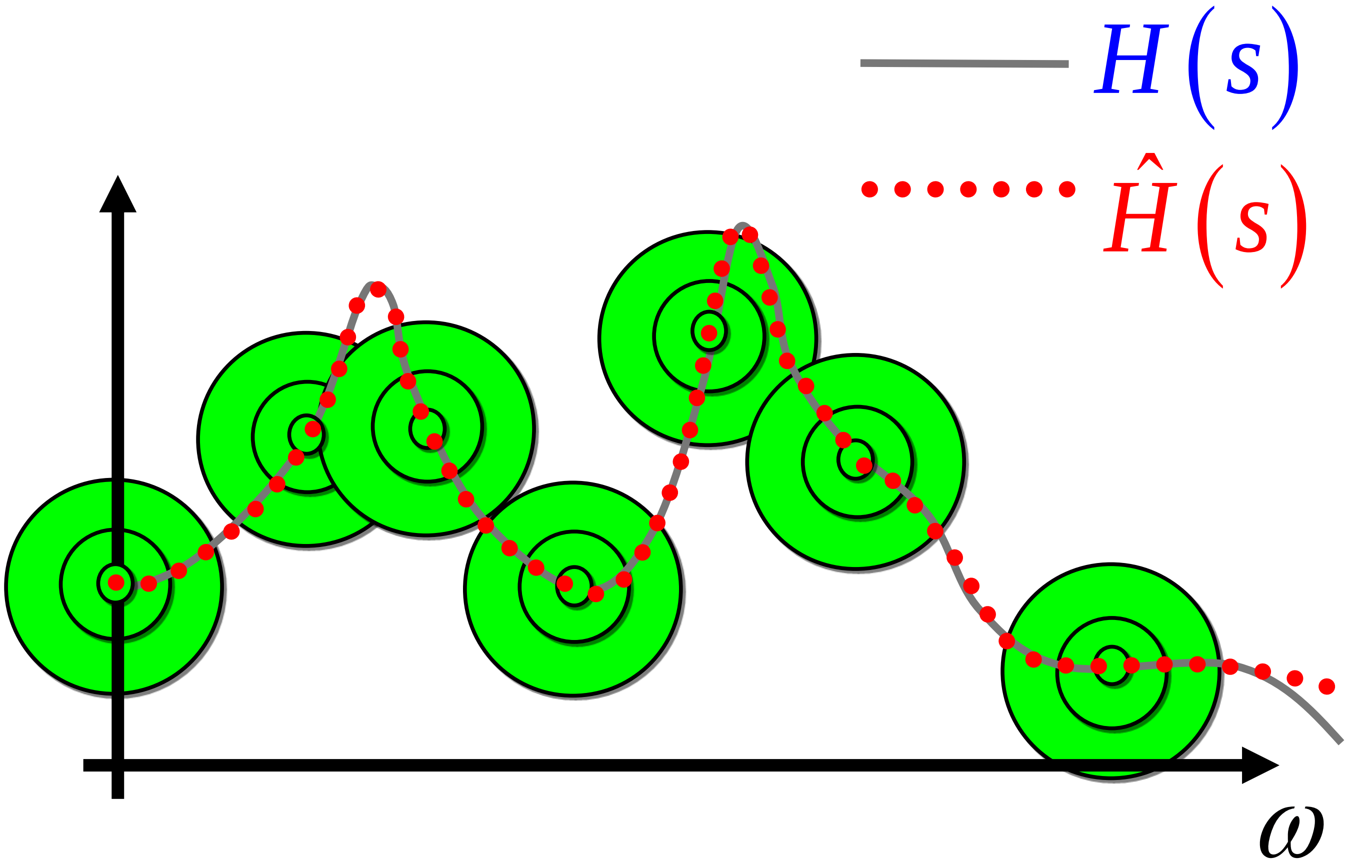
$$\vec{u}_{i+1} \leftarrow \frac{1}{\left\|\vec{u}_{i+1}\right\|}\vec{u}_{i+1}$$

Normalize new vector

end

Combine point and moment matching: multipoint moment matching

- Multiple expansion points give larger band
- Moment (derivates) matching gives more accurate behavior in between expansion points



We know how to select U_q now…

but how about V_q?

PRIMA (for preserving passivity)
(Odabasioglu, Celik, Pileggi TCAD98)

Select $V_q = U_q$ with Arnoldi Krylov Projection Framework:

$$(G + sC)x = Bu \quad \Rightarrow \quad (\overbrace{\boxed{U_q^T G U_q}}^{\hat{G}} + s\overbrace{\boxed{U_q^T C U_q}}^{\hat{C}})x = \overbrace{\boxed{U_q^T B}}^{\hat{B}}u$$

$$y = L^T x \qquad\qquad y = (\underbrace{U_q^T L}_{\hat{L}})^T x$$

$$\mathrm{span}\{\vec{u}_1, \dots, \vec{u}_q\} = \kappa_q(A, R) = \mathrm{span}\{R, AR, \dots, A^{q-1}R\}$$

$$U_q = V_q = \{\vec{u}_1, \dots, \vec{u}_q\}$$

$U_q^T U_q = I \longleftarrow$ Use Arnoldi: Numerically very stable

Moment Matching Theorem

PRIMA preserves the moments of the transfer function up to the q-th order, i.e.,

$$\frac{\partial^i H(s)}{\partial s^i} = \frac{\partial^i \hat{H}(s)}{\partial s^i} \qquad i = 0, 1, \ldots, q-1$$