

# Introduction to Switched-Mode Converter Modeling using MATLAB/Simulink

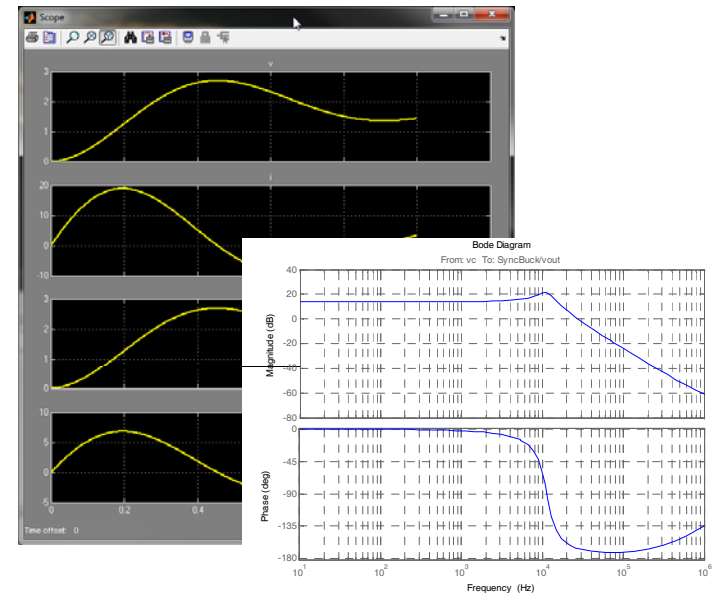
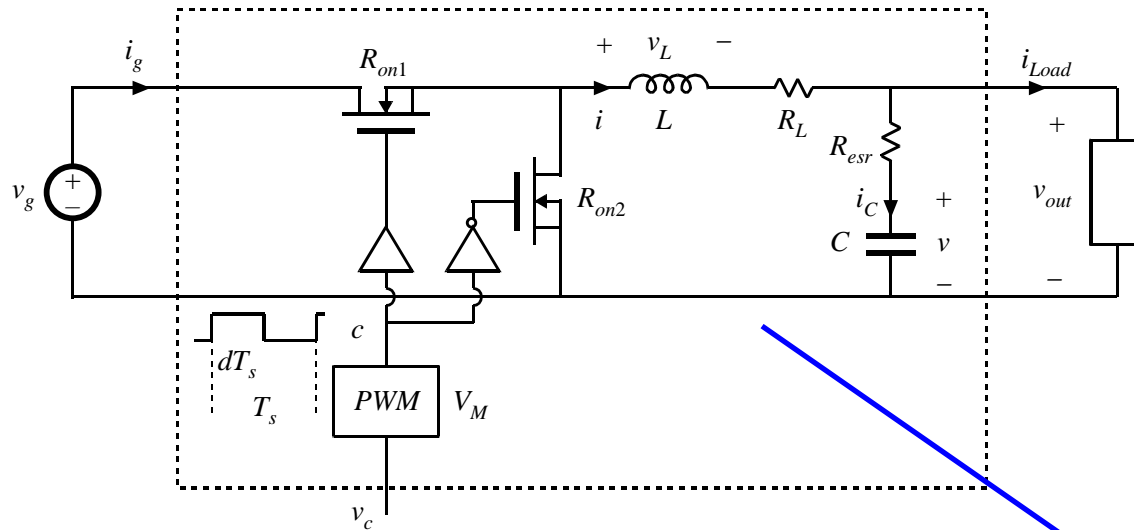
---

- MATLAB: programming and scripting environment
- Simulink: block-diagram modeling environment inside MATLAB
- Motivation:
  - Powerful environment for system modeling and simulation
  - More sophisticated controller models, analysis and design tools
- But\*:
  - Block-diagram based Simulink models, unidirectional signals
  - Not a traditional circuit simulator; specialized physics-based Spice device models or component libraries are not readily available

\*Various add-ons to Simulink are available to allow traditional circuit diagram entry and circuit simulations (e.g. SimPowerSystems, PLECS), or to embed Spice within MATLAB/Simulink environment. These add-ons are not required and will not be used in ECEN5807.

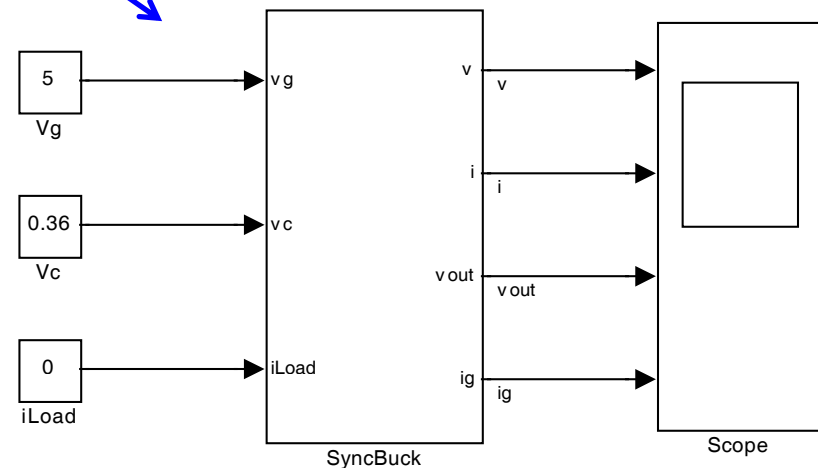
# Introduction through an example

Synchronous buck converter



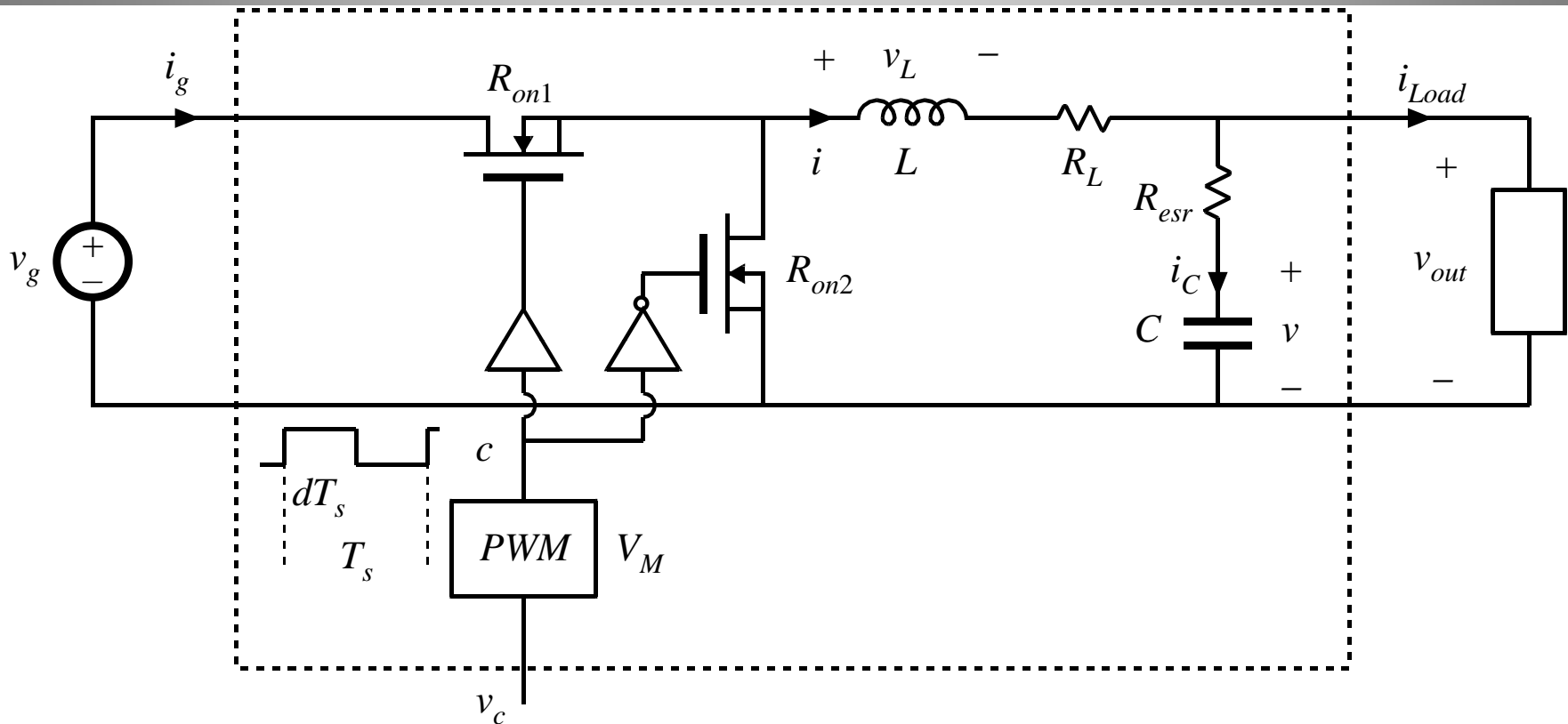
- Switching model
- Averaged model
- Small-signal linearization and frequency responses

See MATLAB/Simulink page on the course website (“Materials” page) for complete step-by-step details, and to download the example files



Simulink model: [syncbuck\\_OL.mdl](#)

# Synchronous Buck Converter

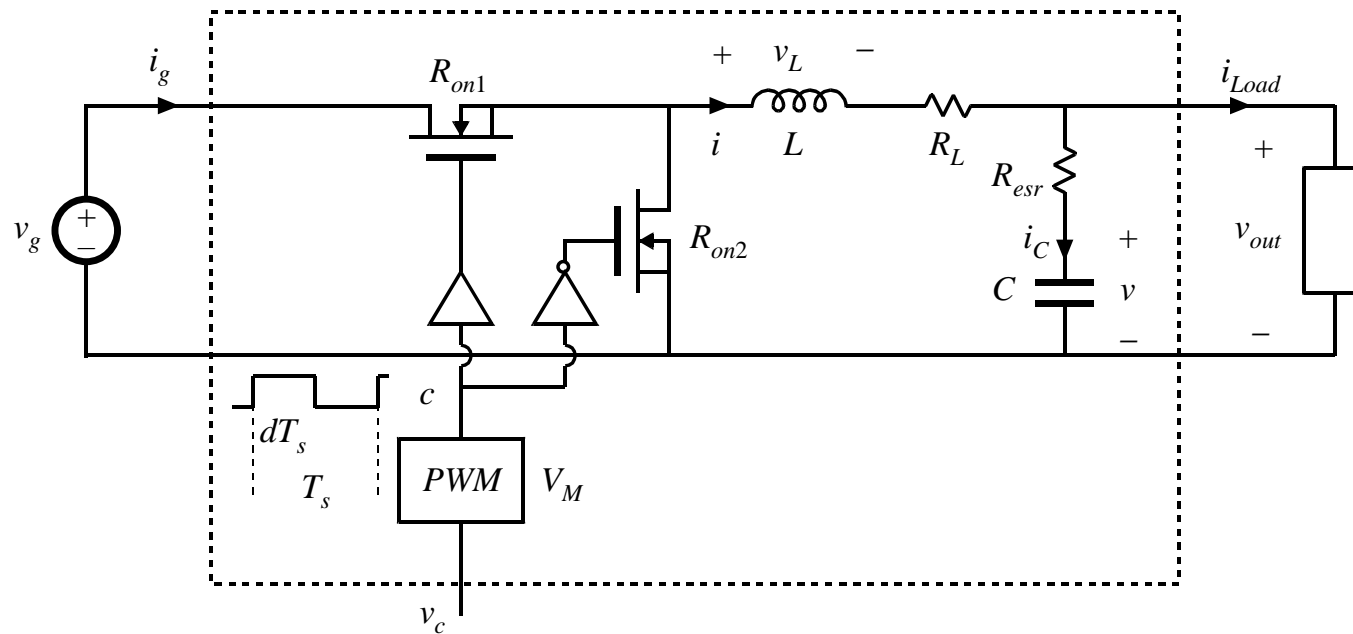


Inputs:  $v_g$ ,  $i_{Load}$ ,  $v_c$

Outputs:  $v_{out}$ ,  $i_g$

State variables:  $v$ ,  $i$

# Converter state equations



## State equations

$$v_L = L \frac{di}{dt} = \begin{cases} v_g - (R_{on1} + R_L)i - v_{out} & (c = 1) \\ -(R_{on2} + R_L)i - v_{out} & (c = 0) \end{cases}$$

$$i_C = C \frac{dv}{dt} = i - i_{Load}$$

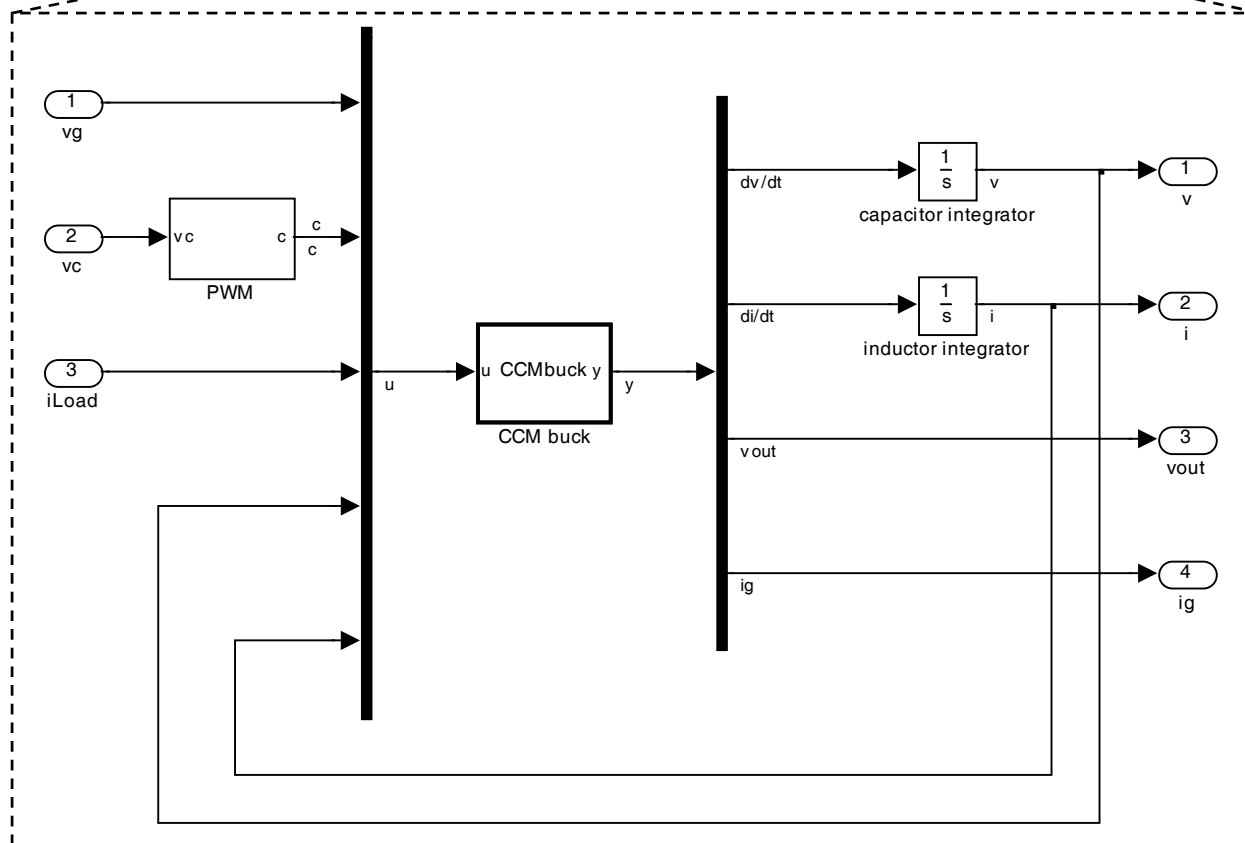
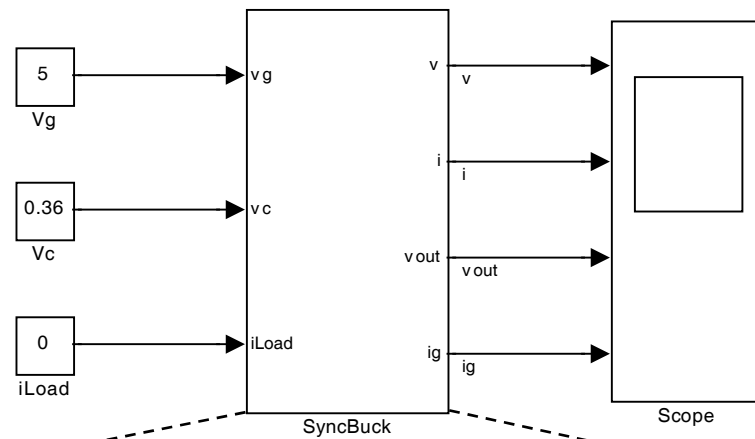
## Output equations

$$i_g = \begin{cases} i & (c = 1) \\ 0 & (c = 0) \end{cases}$$

$$v_{out} = v + R_{esr}(i - i_{Load})$$

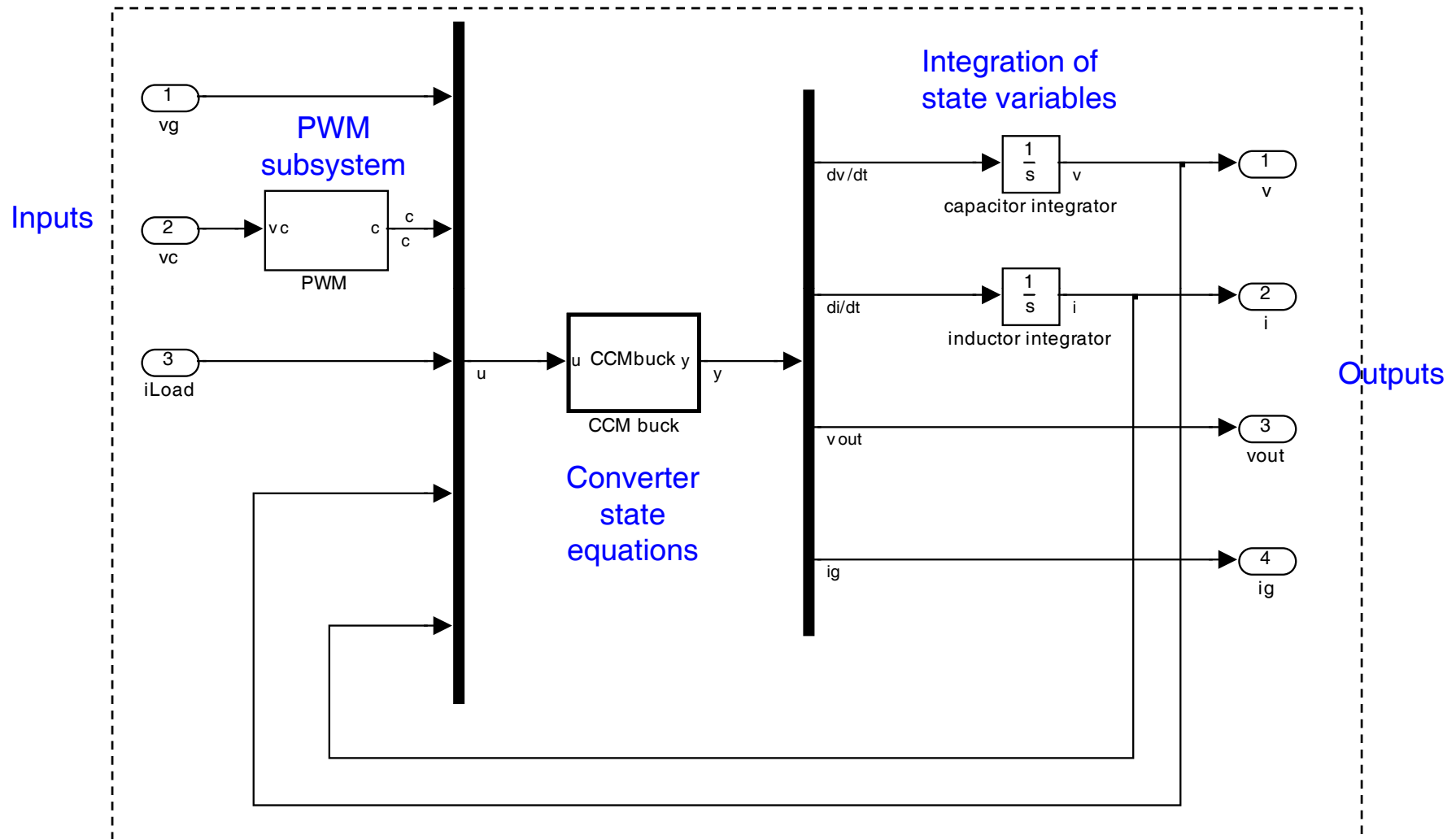
# Simulink model

## SyncBuck subsystem block

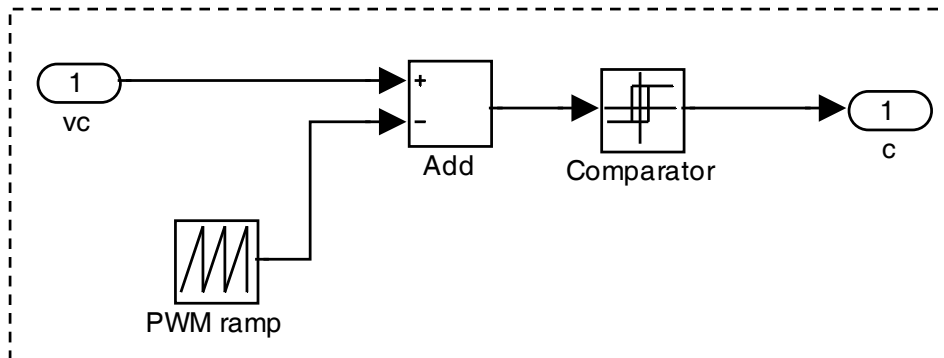
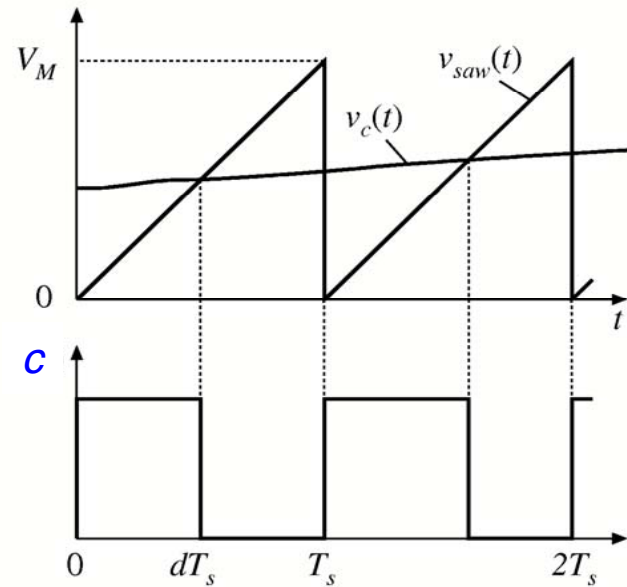
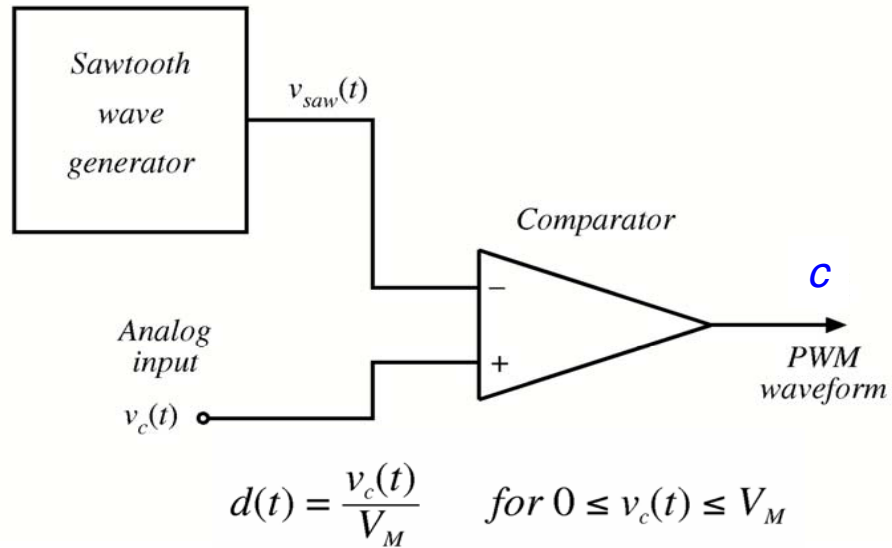


SyncBuck  
subsystem  
block  
internals

# Synchronous buck (SyncBuck) subsystem

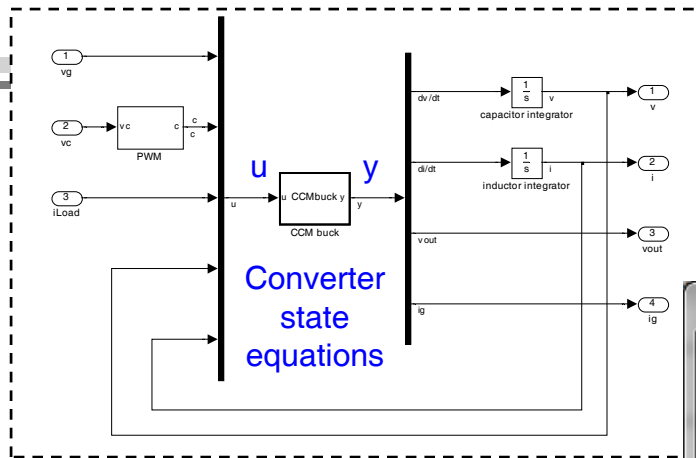


# PWM operation and model



Simulink PWM model

# Converter state equations: embedded MATLAB script



$$u = \text{inputs} = [v_g \ c \ i_{Load} \ v \ i]$$

$$y = \text{outputs} = [iC/C \ vL/L \ v_{out} \ i_g]$$

```

Embedded MATLAB Editor - Block: syncbuck_OL/SyncBuck/CCM buck
File Edit Text Debug Tools Window Help
1 function y = CCMbuck(u,L,C,RL,Ron1,Ron2,Resr)
2 % State equations of a synchronous buck converter
3 % Conduction losses due to RL, Ron1, Ron2, Resr are included
4 % Inputs: u = [vg d iLoad v i]
5 % Outputs: y = [iC/C vL/L vout ig]
6 % Parameters: L, C, RL, Ron1, Ron2, Resr
7 %
8 % variables
9 - vg = u(1); % input voltage
10 - d = u(2); % switch control d=c (in the switching model), d in the averaged model
11 - iLoad = u(3); % load current
12 - v = u(4); % capacitor voltage
13 - i = u(5); % inductor current
14 %
15 % state equations
16 - vout = v + Resr*(i-iLoad); % output voltage
17 - ig = d*i; % input current
18 - iC = i - iLoad; % capacitor current
19 - vL = d*(vg-(Ron1+RL)*i-vout)+(1-d)*(-(Ron2+RL)*i - vout); % inductor voltage
20 %
21 % output
22 - y = [iC/C vL/L vout ig];
    
```

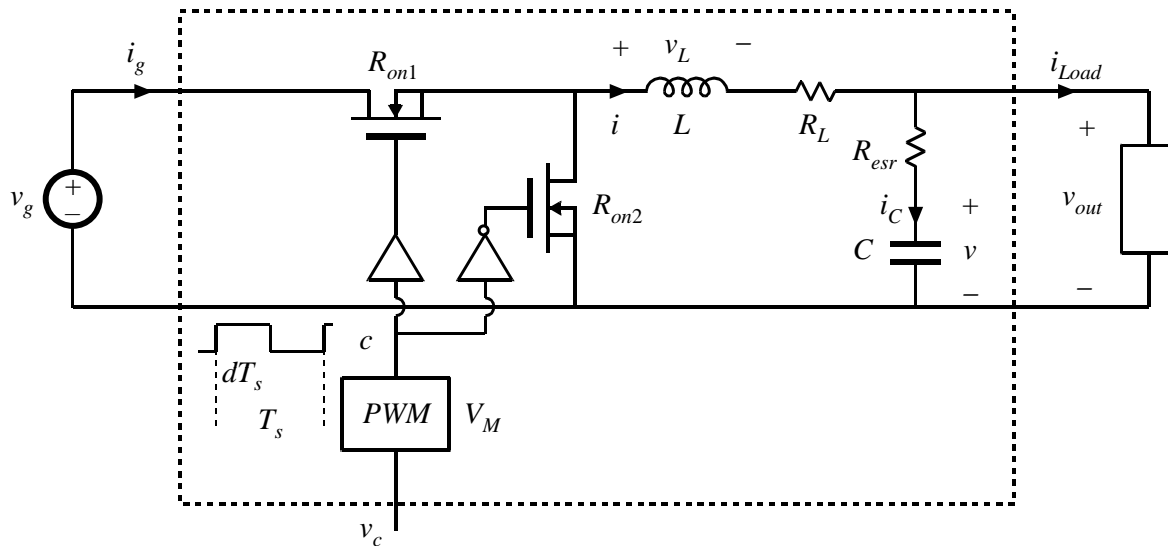
$$v_{out} = v + R_{esr} (i - i_{Load})$$

$$i_g = \begin{cases} i & (c = 1) \\ 0 & (c = 0) \end{cases}$$

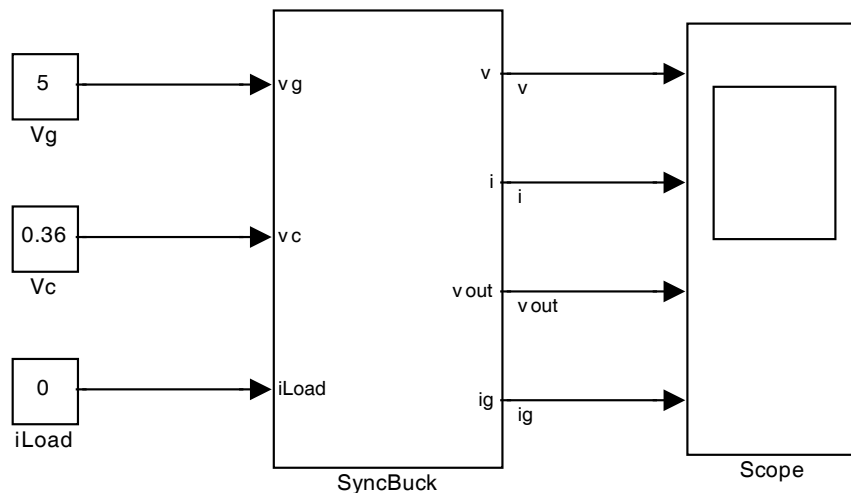
$$i_C = C \frac{dv}{dt} = i - i_{Load}$$

$$v_L = L \frac{di}{dt} = \begin{cases} v_g - (R_{on1} + R_L)i - v_{out} & (c = 1) \\ -(R_{on2} + R_L)i - v_{out} & (c = 0) \end{cases}$$

# Numerical example

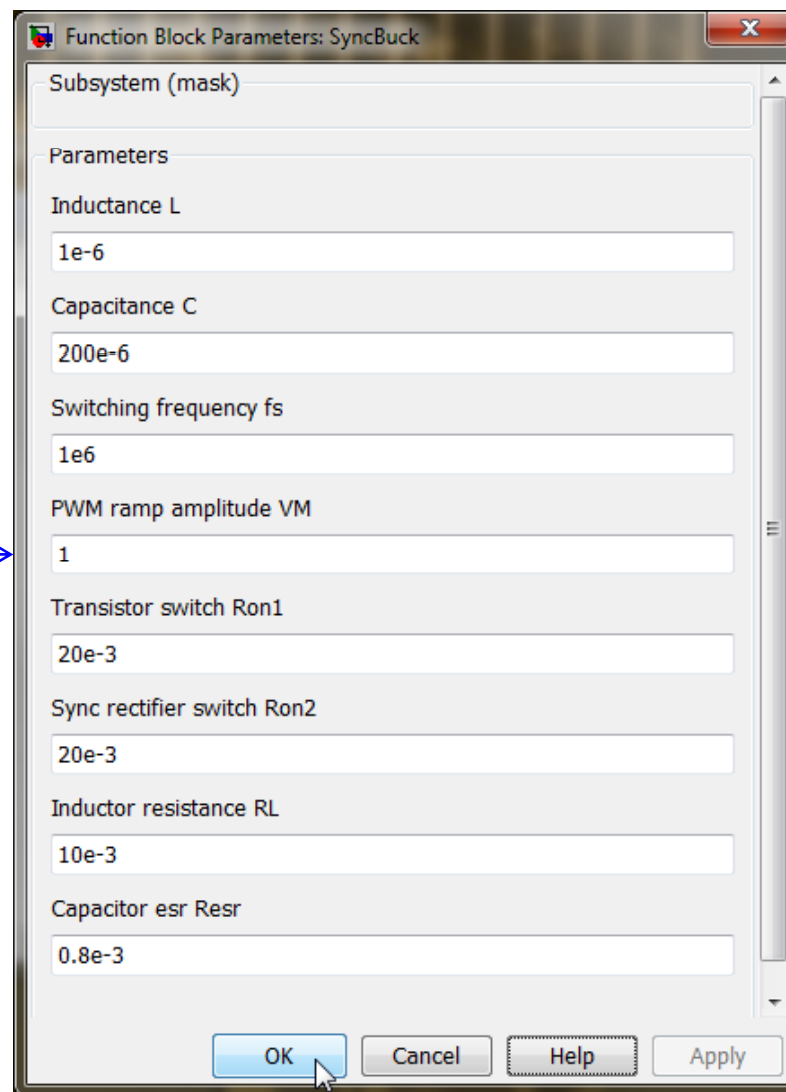
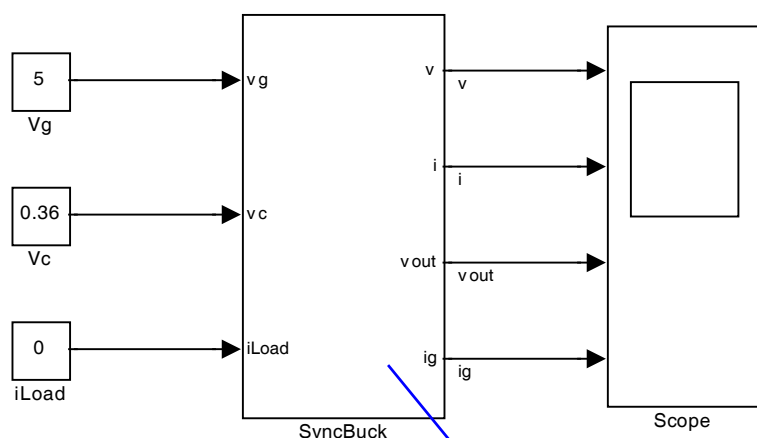


- Switching frequency:  
 $f_s = 1\text{MHz}$
- $I_{out} = 0$
- $V_g = 5\text{V}$
- $L = 1\ \mu\text{H}$
- $R_L = 10\ \text{m}\Omega$
- $R_{on1} = R_{on2} = 20\ \text{m}\Omega$
- $C = 200\ \mu\text{F}$
- $R_{esr} = 0.8\ \text{m}\Omega$
- PWM ramp amplitude  
 $V_M = 1\ \text{V}$
- $V_c = 0.36, D = 0.36$



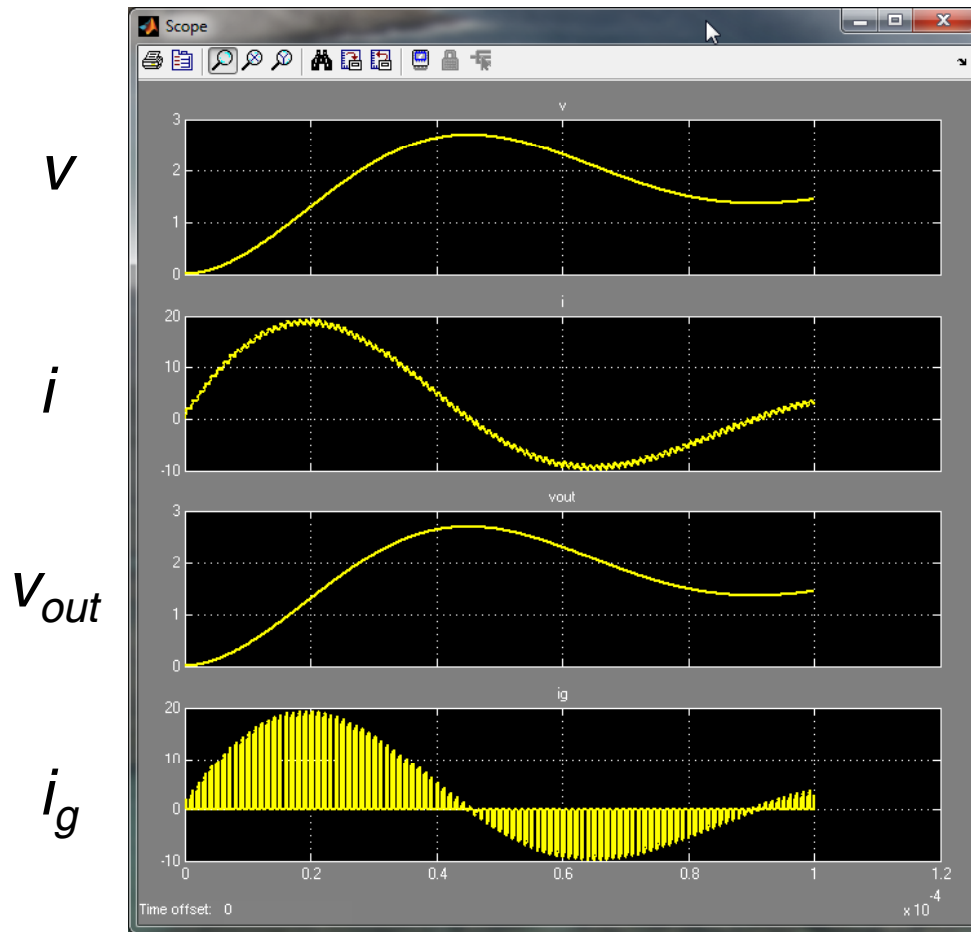
Simulink model: [syncbuck\\_OL.mdl](#)

# Numerical example: synchronous buck converter model

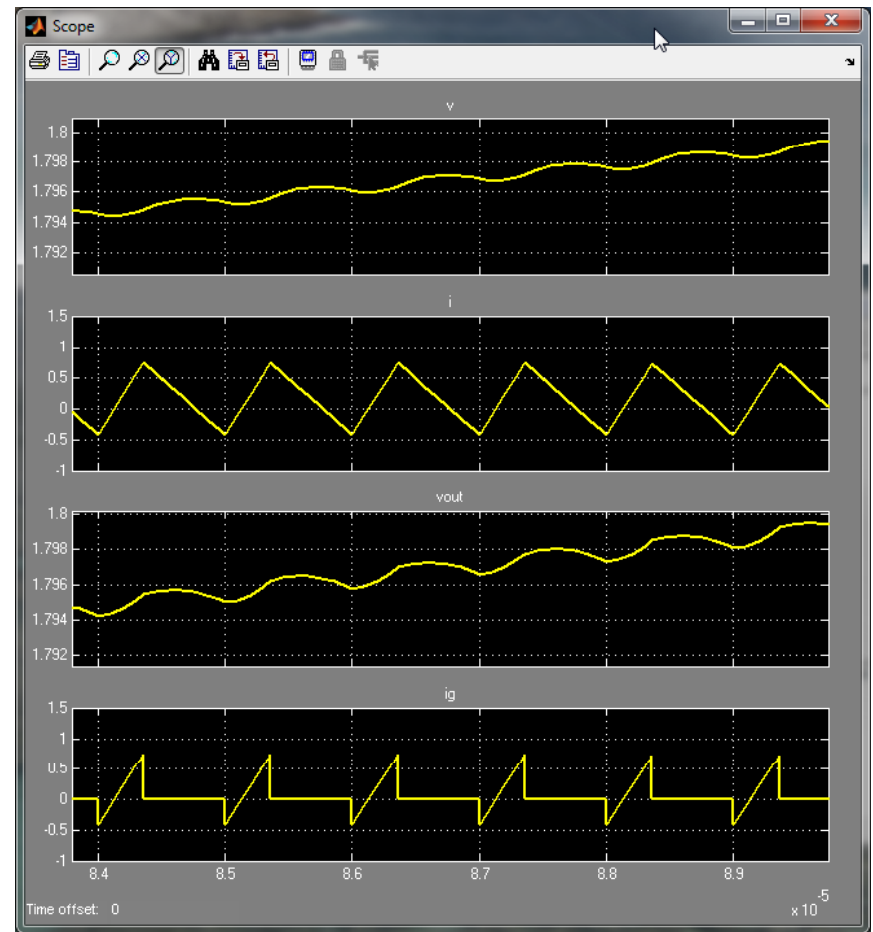


- “Masking” a Simulink subsystem allows parameterization
- Same subsystem model can be re-used
- Models and MATLAB scripts can be collected in a library

# Switching simulation: open-loop start-up transient



20  $\mu$ s/div



Zoom in, 1  $\mu$ s/div

# Averaged model

$$v_L = L \frac{di}{dt} = \begin{cases} v_g - (R_{on1} + R_L)i - v_{out} & (c = 1) \\ -(R_{on2} + R_L)i - v_{out} & (c = 0) \end{cases}$$

$$i_C = C \frac{dv}{dt} = i - i_{Load}$$

$$i_g = \begin{cases} i & (c = 1) \\ 0 & (c = 0) \end{cases}$$

$$v_{out} = v + R_{esr} (i - i_{Load})$$

Switching model

State-space averaging (review Textbook Sections 7.1-7.3)

$$\langle v_L \rangle_{T_s} = L \frac{d\langle i \rangle_{T_s}}{dt} = d \left( \langle v_g \rangle_{T_s} - (R_{on1} + R_L) \langle i \rangle_{T_s} - \langle v_{out} \rangle_{T_s} \right) + (1-d) \left( -(R_{on2} + R_L) \langle i \rangle_{T_s} - \langle v_{out} \rangle_{T_s} \right)$$

$$\langle i_C \rangle_{T_s} = C \frac{d\langle v \rangle_{T_s}}{dt} = \langle i \rangle_{T_s} - \langle i_{Load} \rangle_{T_s}$$

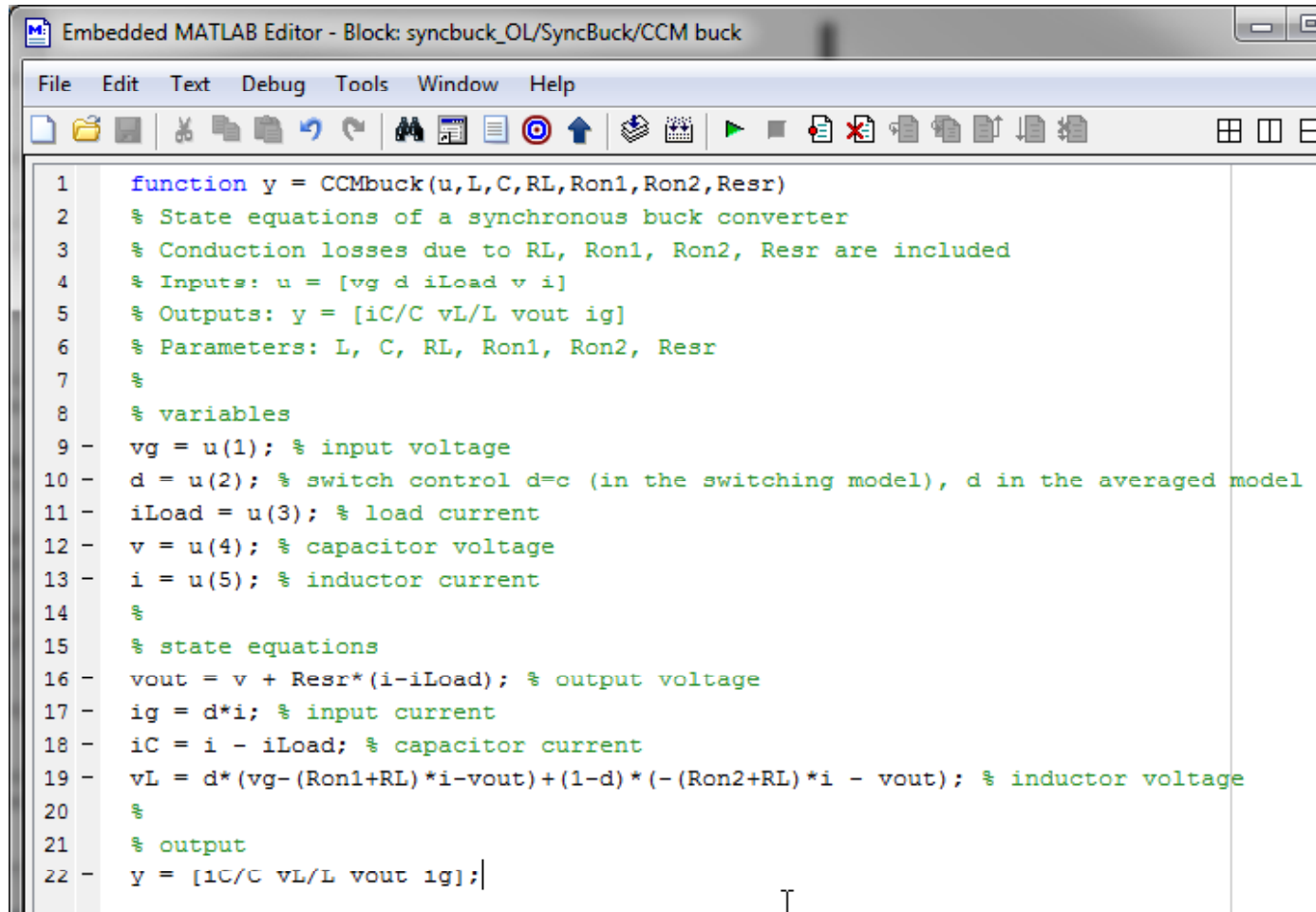
$$\langle i_g \rangle_{T_s} = d \langle i \rangle_{T_s}$$

$$\langle v_{out} \rangle_{T_s} = \langle v \rangle_{T_s} + R_{esr} \left( \langle i \rangle_{T_s} - \langle i_{Load} \rangle_{T_s} \right)$$

Large-signal  
averaged model

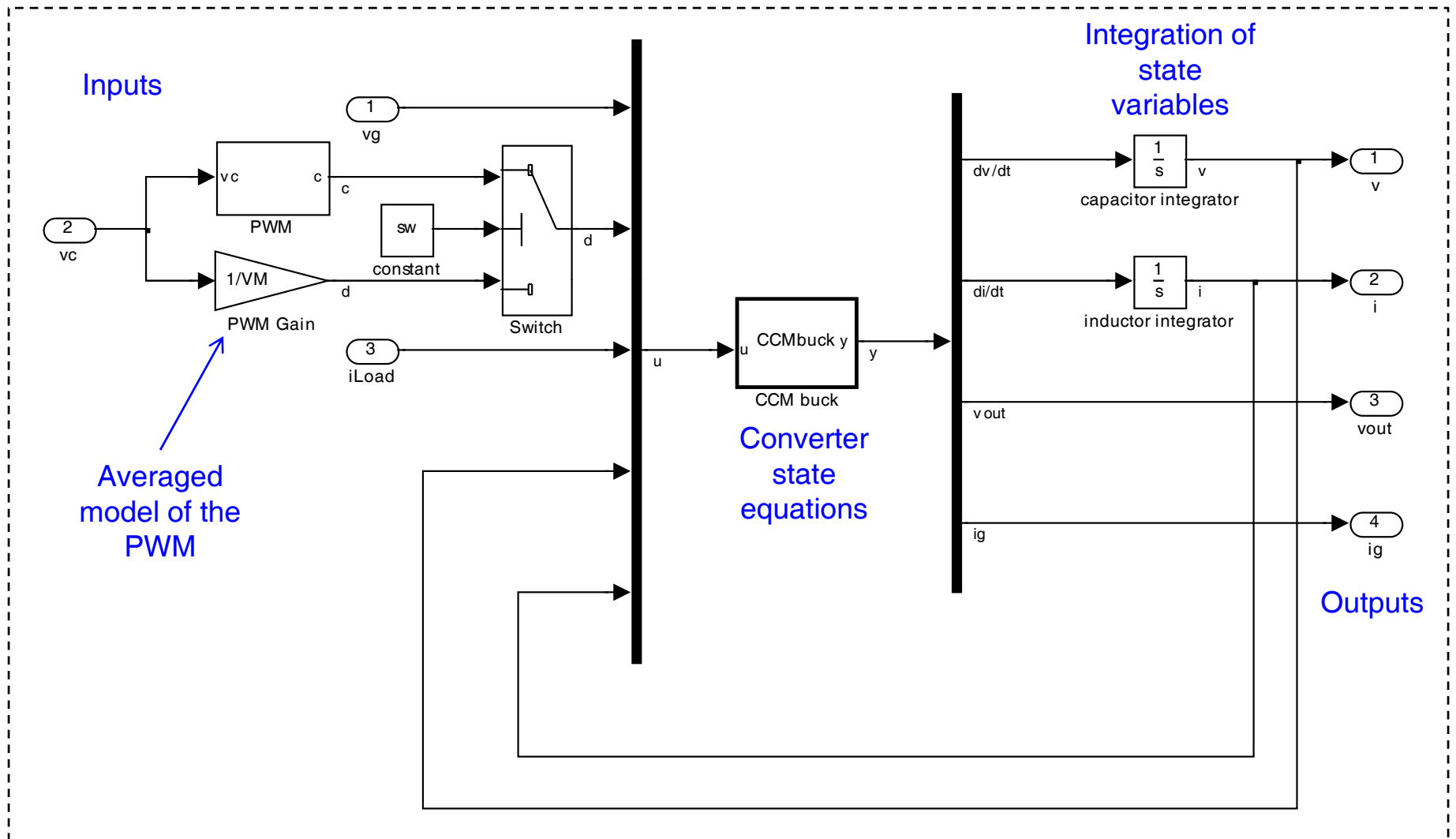
# Converter averaged state equations: MATLAB

The MATLAB function stays exactly the same, except  $d$  (duty-cycle) replaces  $c$  (switch control)



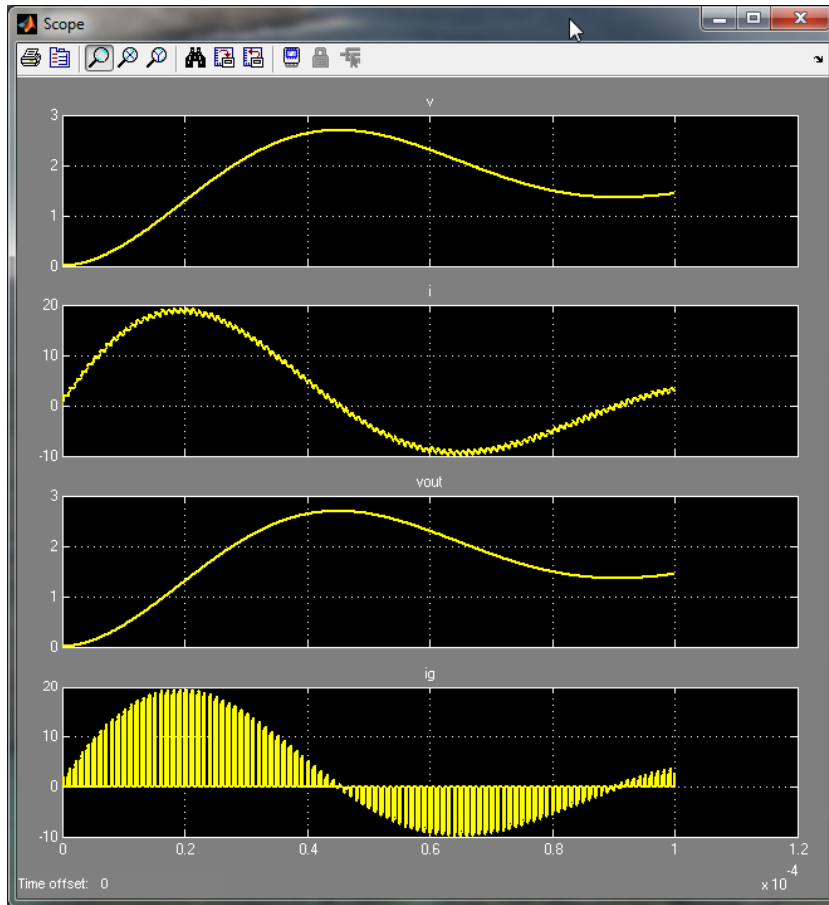
```
Embedded MATLAB Editor - Block: syncbuck_OL/SyncBuck/CCM buck
File Edit Text Debug Tools Window Help
1 function y = CCMbuck(u,L,C,RL,Ron1,Ron2,Resr)
2 % State equations of a synchronous buck converter
3 % Conduction losses due to RL, Ron1, Ron2, Resr are included
4 % Inputs: u = [vg d iLoad v i]
5 % Outputs: y = [iC/C vL/L vout ig]
6 % Parameters: L, C, RL, Ron1, Ron2, Resr
7 %
8 % variables
9 - vg = u(1); % input voltage
10 - d = u(2); % switch control d=c (in the switching model), d in the averaged model
11 - iLoad = u(3); % load current
12 - v = u(4); % capacitor voltage
13 - i = u(5); % inductor current
14 %
15 % state equations
16 - vout = v + Resr*(i-iLoad); % output voltage
17 - ig = d*i; % input current
18 - iC = i - iLoad; % capacitor current
19 - vL = d*(vg-(Ron1+RL)*i-vout)+(1-d)*(-(Ron2+RL)*i - vout); % inductor voltage
20 %
21 % output
22 - y = [iC/C vL/L vout ig];
```

# Synchronous buck (SyncBuck) subsystem: switching or averaged model



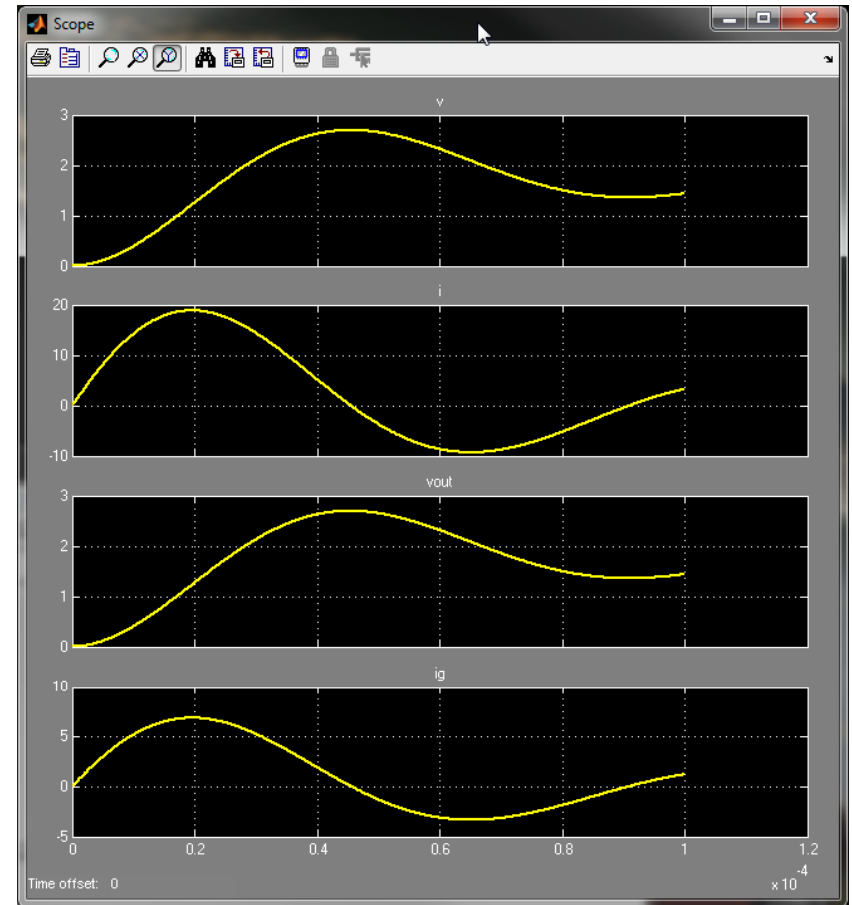
# Start-up transient simulations

## Switching model



$v$   
 $i$   
 $V_{out}$   
 $i_g$

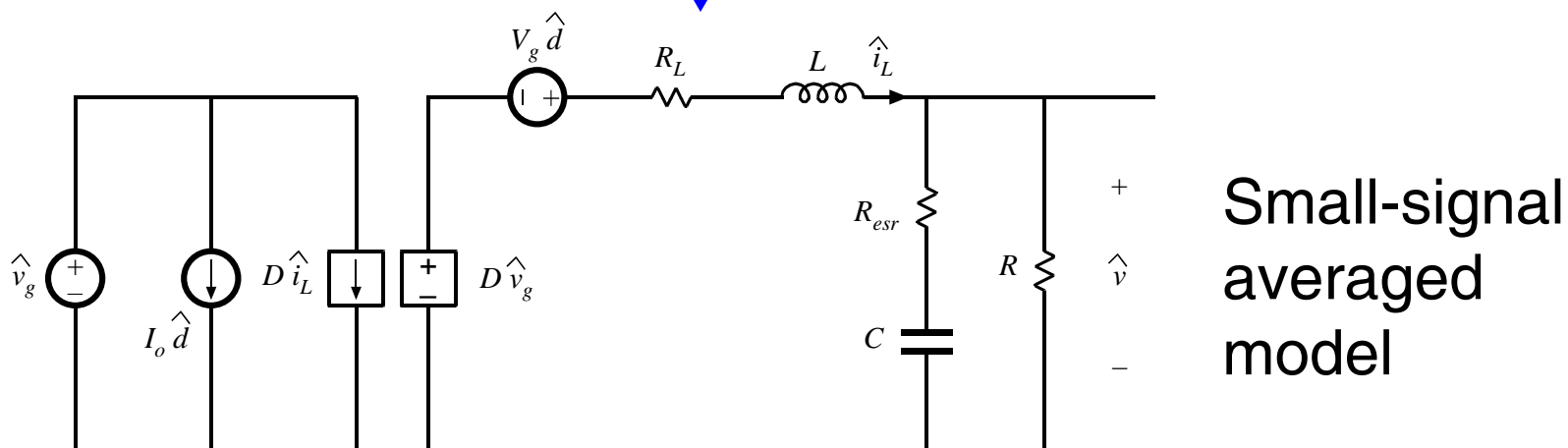
## Averaged model



# Linearization of the large-signal averaged model

Large-signal (nonlinear) averaged model

Linearization at an operating point



Small-signal averaged model

The small-signal model can be solved for all important converter transfer functions:

$$G_{vd}(s) = \frac{\hat{v}}{\hat{d}}$$

**Control-to-output**

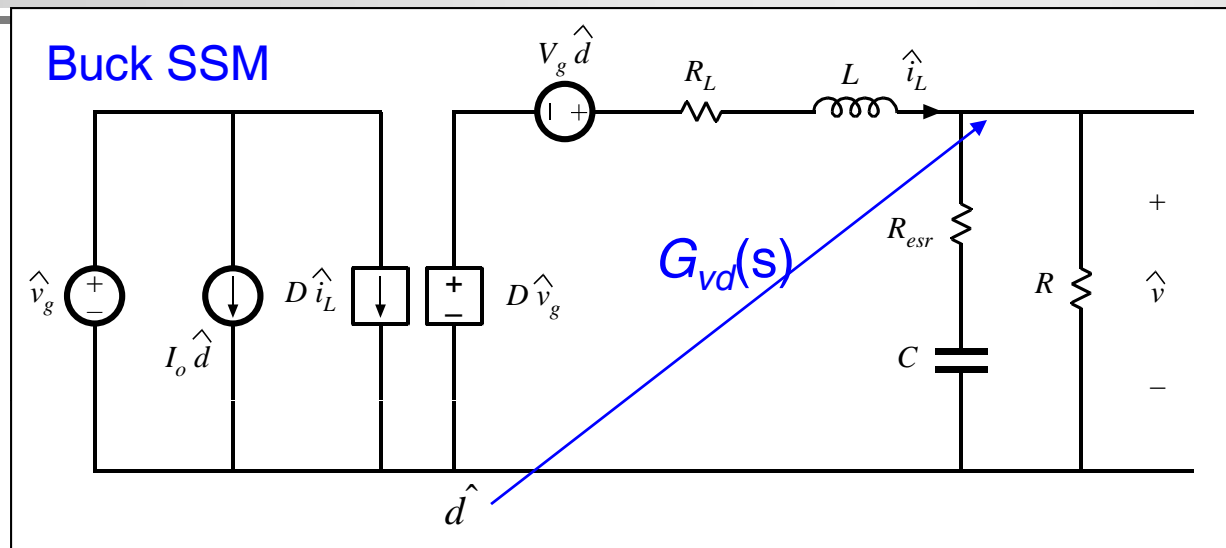
$$G_{vg}(s) = \frac{\hat{v}}{\hat{v}_g}$$

**Line-to-output**

$$Z_{out}(s) = \frac{\hat{v}}{\hat{i}_{load}}$$

**Output impedance**

# Synchronous buck converter example



$$G_{vd}(s) = \frac{\hat{v}_o}{\hat{d}}$$

$$G_{vd}(s) = V_g \frac{1 + \frac{s}{\omega_{esr}}}{1 + \frac{1}{Q} \frac{s}{\omega_o} + \left(\frac{s}{\omega_o}\right)^2}$$

Pair of poles:

$$f_o = \frac{1}{2\pi\sqrt{CL}} = 11 \text{ kHz}$$

$$Q_{loss} = \frac{\sqrt{L/C}}{R_{esr} + R_L} = 2.3 \rightarrow 7.2 \text{ dB} \quad Q_{load} = \frac{R}{\sqrt{L/C}} > 5$$

$$Q = Q_{loss} \parallel Q_{load} = \frac{Q_{loss} Q_{load}}{Q_{loss} + Q_{load}} < 2.3 \rightarrow 7.2 \text{ dB}$$

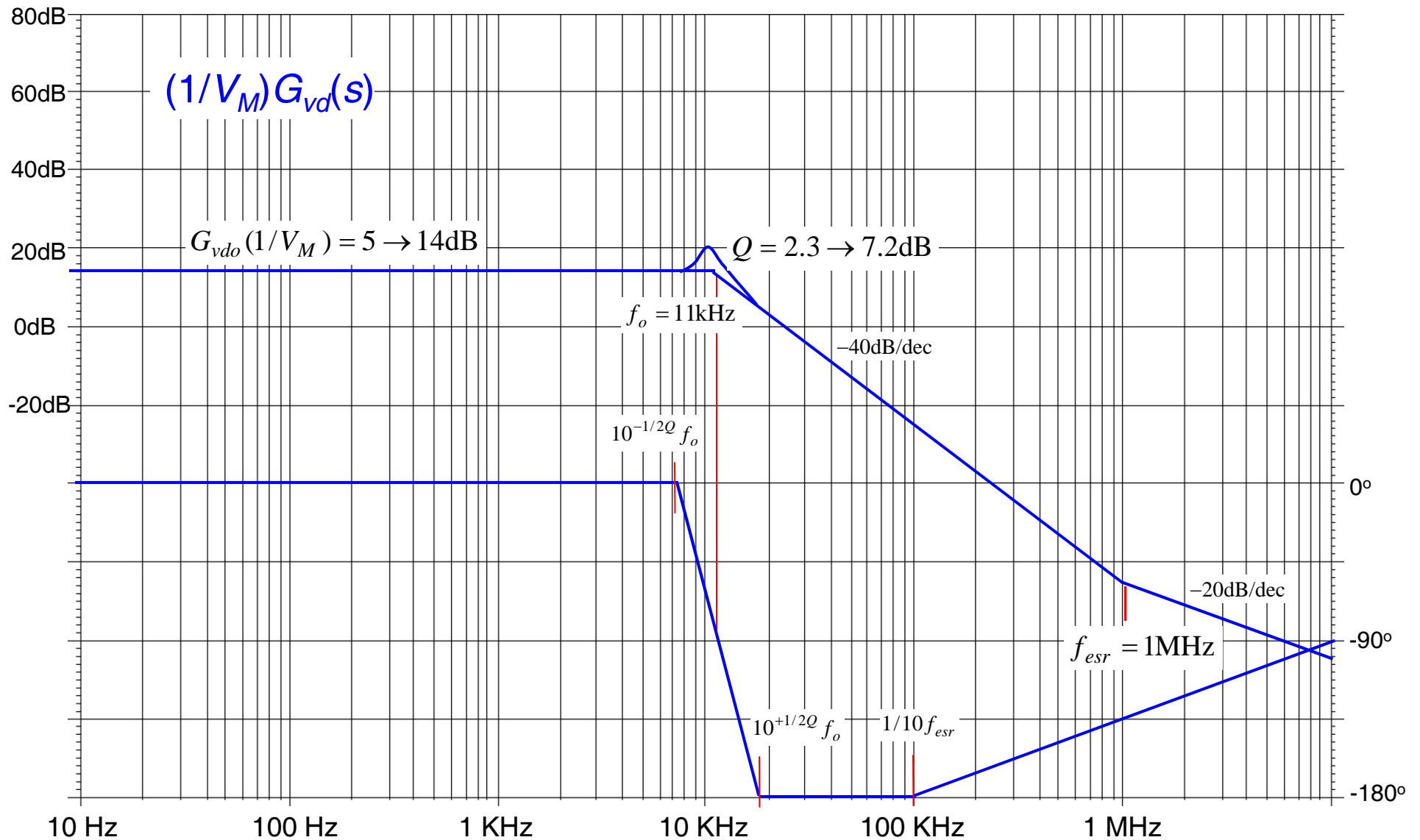
Low-frequency gain:

$$G_{vdo} = 5\text{V} \rightarrow 14\text{dBV}$$

ESR zero:

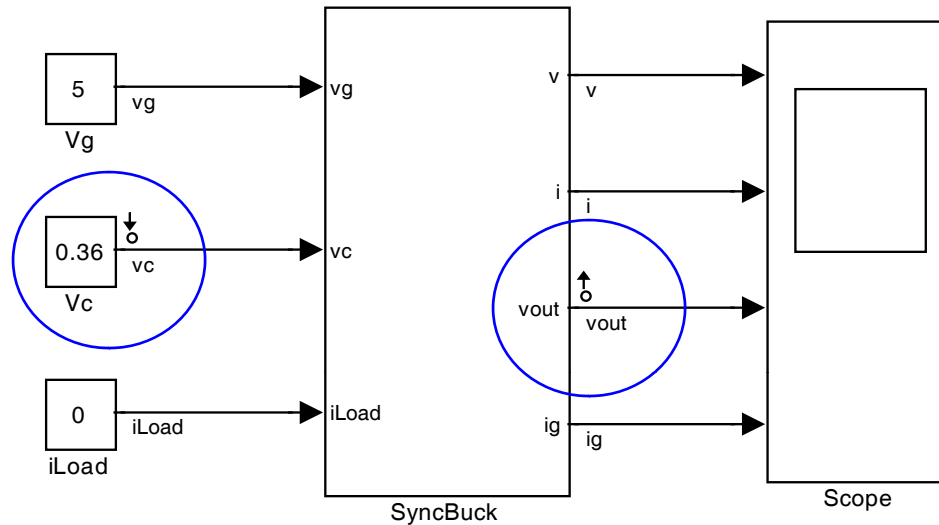
$$f_{esr} = \frac{1}{2\pi CR_{esr}} = 1 \text{ MHz}$$

# Magnitude and phase Bode plots of $G_{vd}$



# Linearization and frequency responses in MATLAB/Simulink

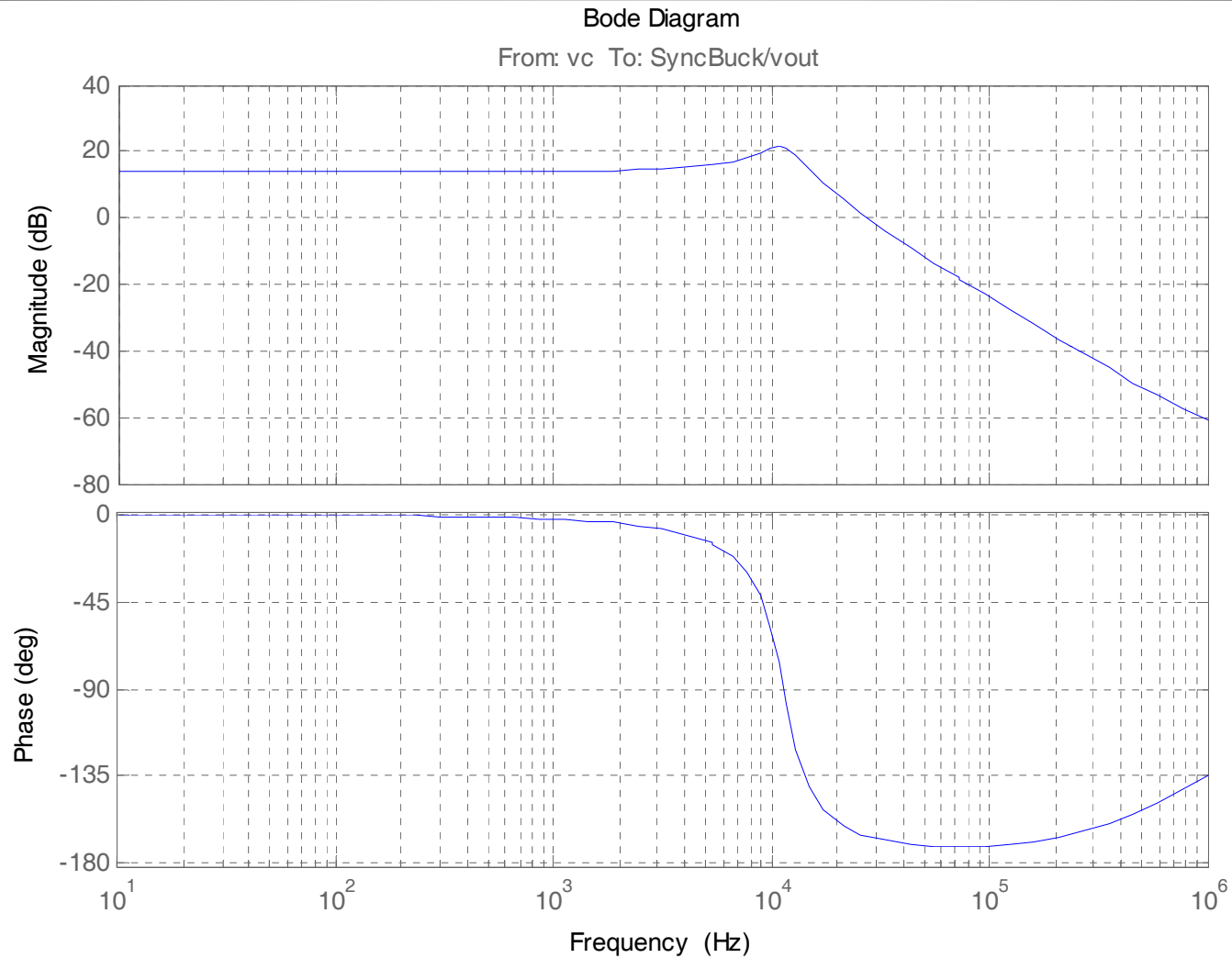
1. Set transfer function input and output points



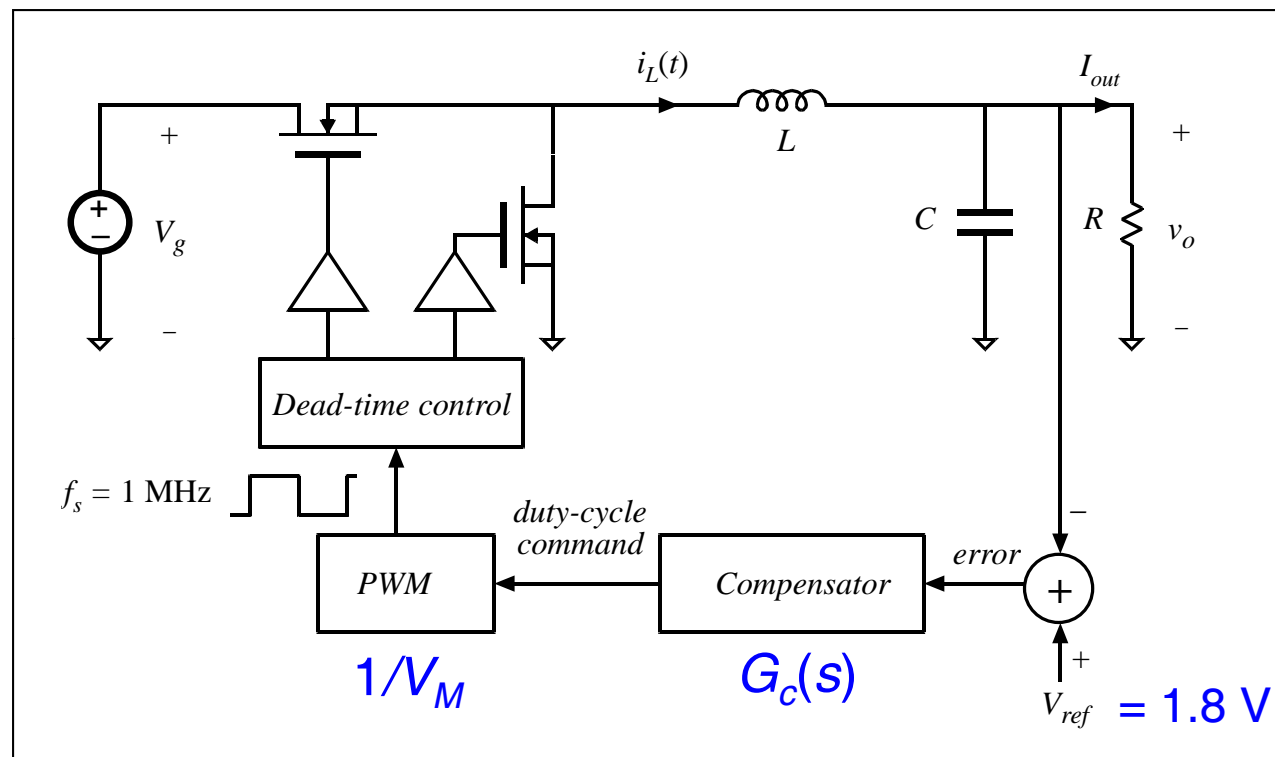
2. MATLAB script (BodePlotter\_script.m) computes DC operating point, linearizes the model, computes and plots the transfer function magnitude and phase responses

```
1 %% Bode plotter using linearization tool
2 % requires Simulink Control Design toolbox
3 %
4 %
5 model = 'syncbuck_OL'; % set to file name of simulink model. Must have i/o points set within this model
6 io = getlinio(model) % get i/o signals of model
7 op = operspec(model)
8 op = findop(model,op) % calculate model operating point
9 ssm = linearize(model,op,io) % compute state space model of linearized system
10 %
11 %
12 ltiview('bode',ssm) % send linearized model to LTI Viewer tool
13 %
```

# Magnitude and phase Bode plots of $G_{vd}$

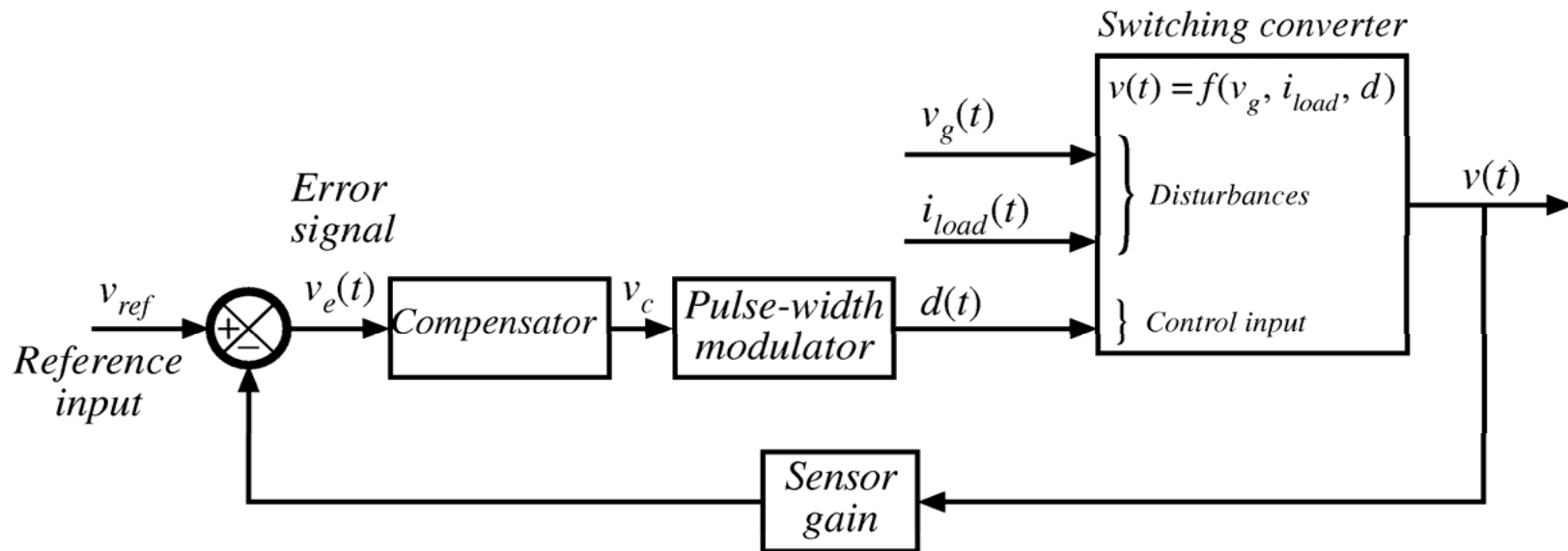


# Closed-loop (voltage-mode) control



Point-of-Load (POL) Synchronous Buck Regulator

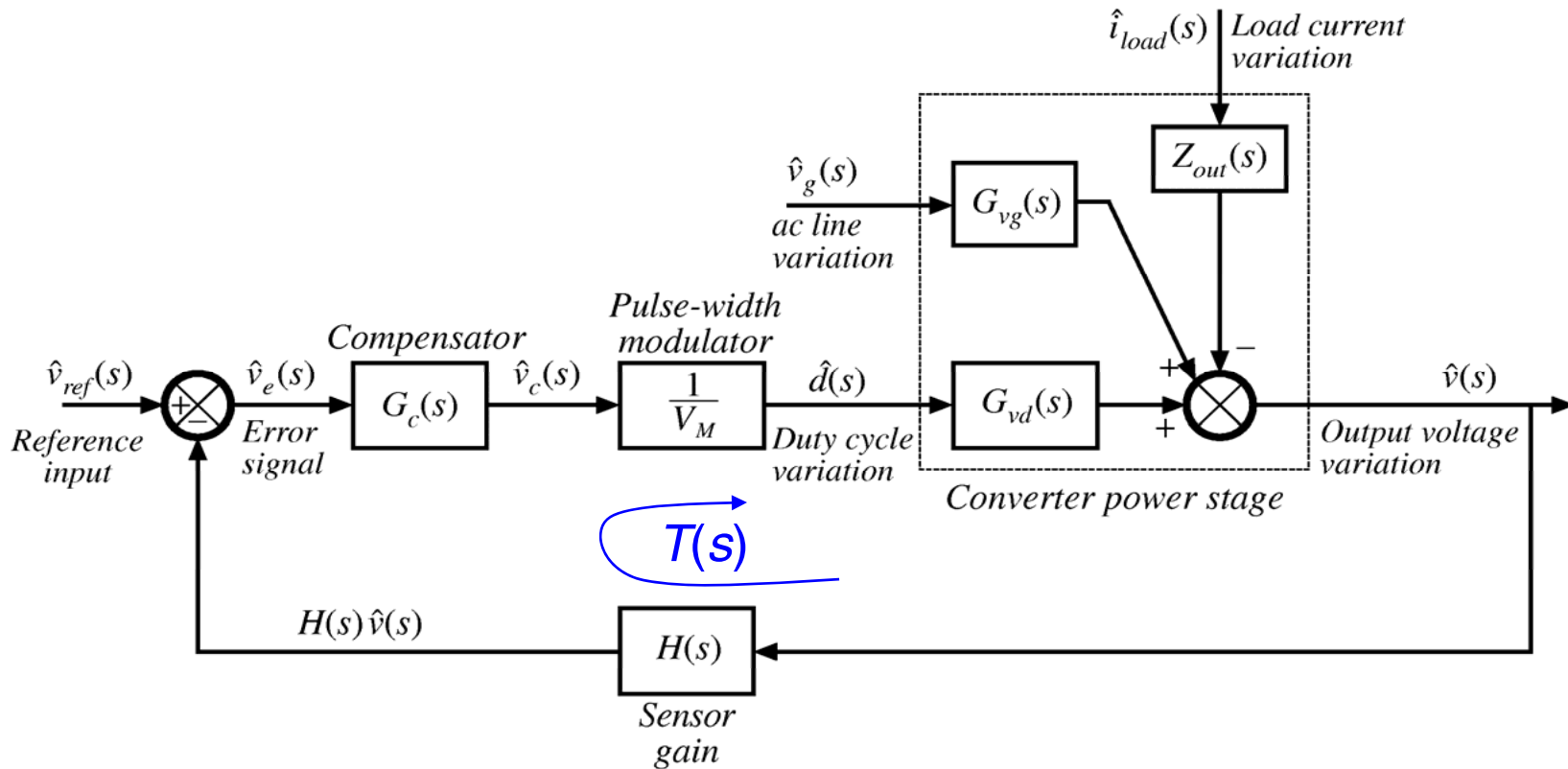
# Closed-loop SMPS block diagram



**Control objectives:** tight output voltage regulation

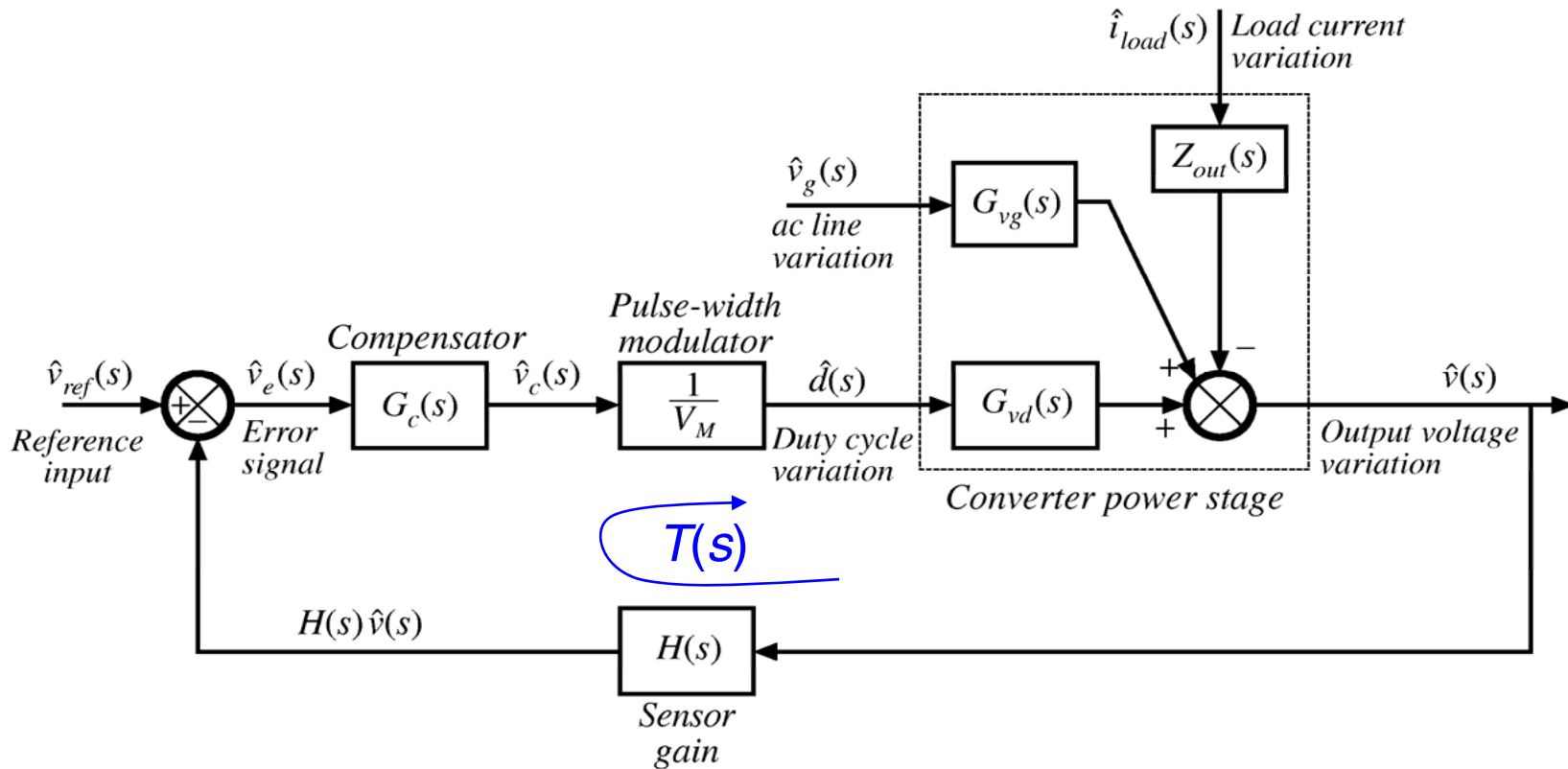
- Static or dynamic disturbances
  - Input (line) voltage  $v_g$
  - Load current  $i_{load}$
- Component tolerances

# Small-signal model: loop gain $T$



Loop gain:  $T(s) = H(s)G_c(s)(1/V_M)G_{vd}(s)$

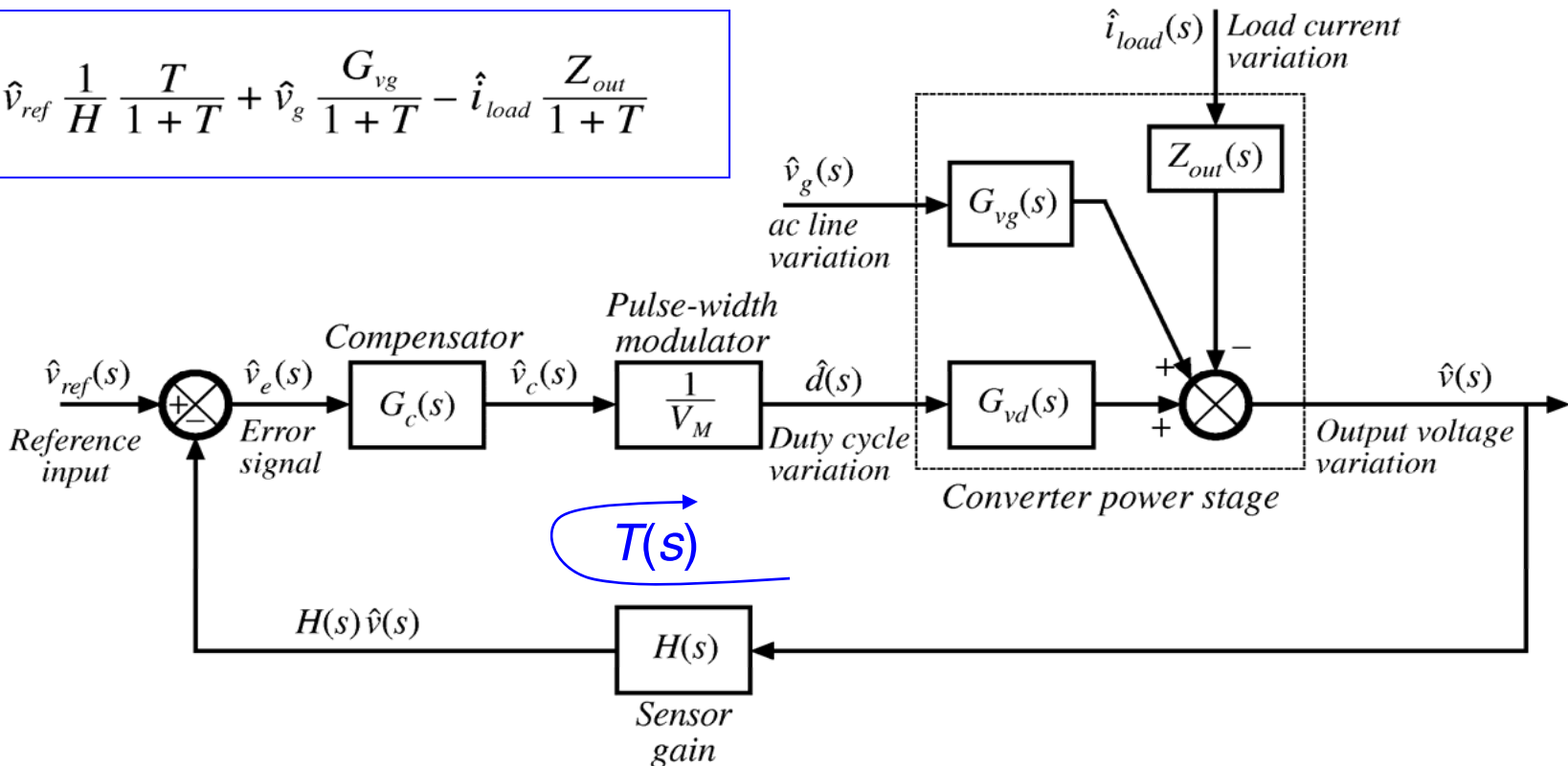
# Small-signal model: closed-loop responses



$$\hat{v} = \hat{v}_{ref} \frac{1}{H} \frac{T}{1+T} + \hat{v}_g \frac{G_{vg}}{1+T} - \hat{i}_{load} \frac{Z_{out}}{1+T}$$

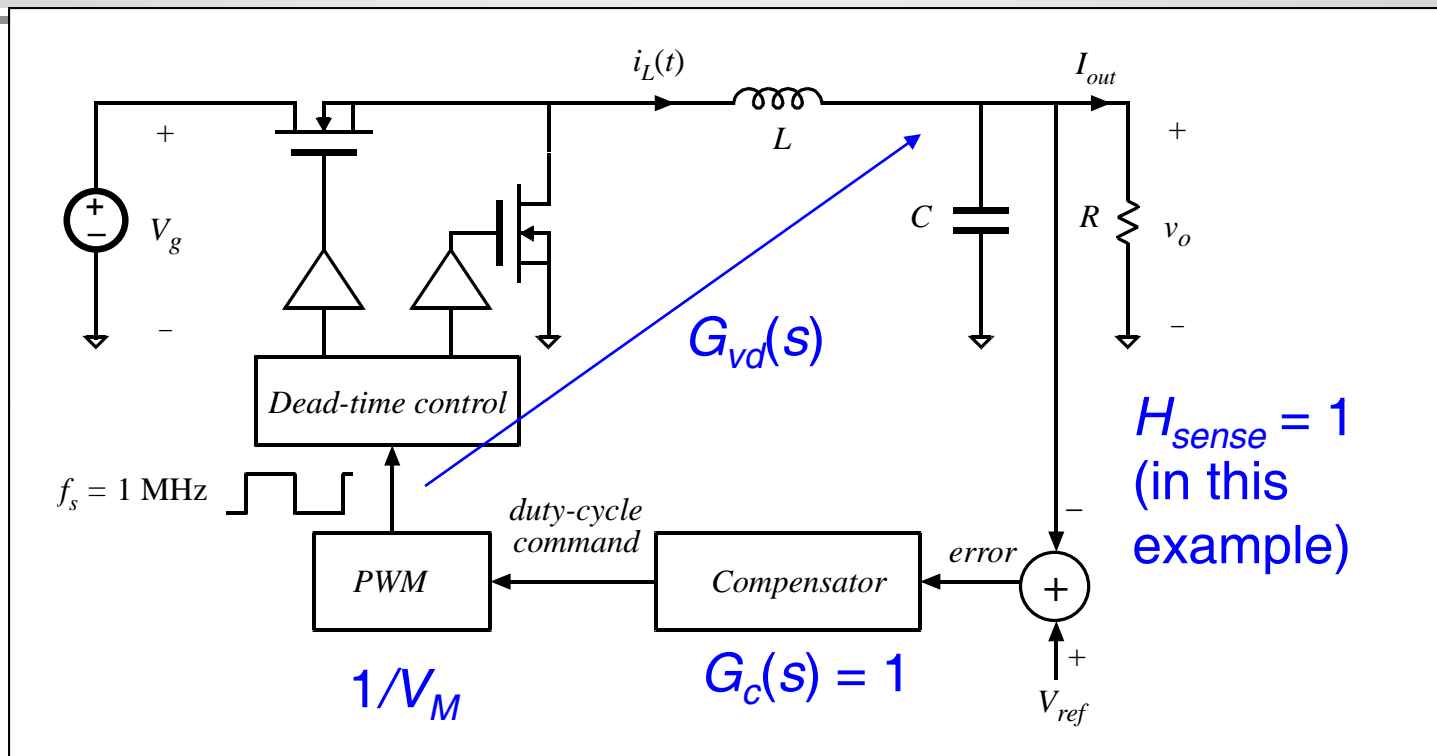
# Feedback loop design objectives

$$\hat{v} = \hat{v}_{ref} \frac{1}{H} \frac{T}{1+T} + \hat{v}_g \frac{G_{vg}}{1+T} - \hat{i}_{load} \frac{Z_{out}}{1+T}$$



- To meet the control objectives, design  $T$  as large as possible in as wide frequency range as possible, i.e. with as high  $f_c$  as possible
- Limitation: stability and quality of closed-loop responses

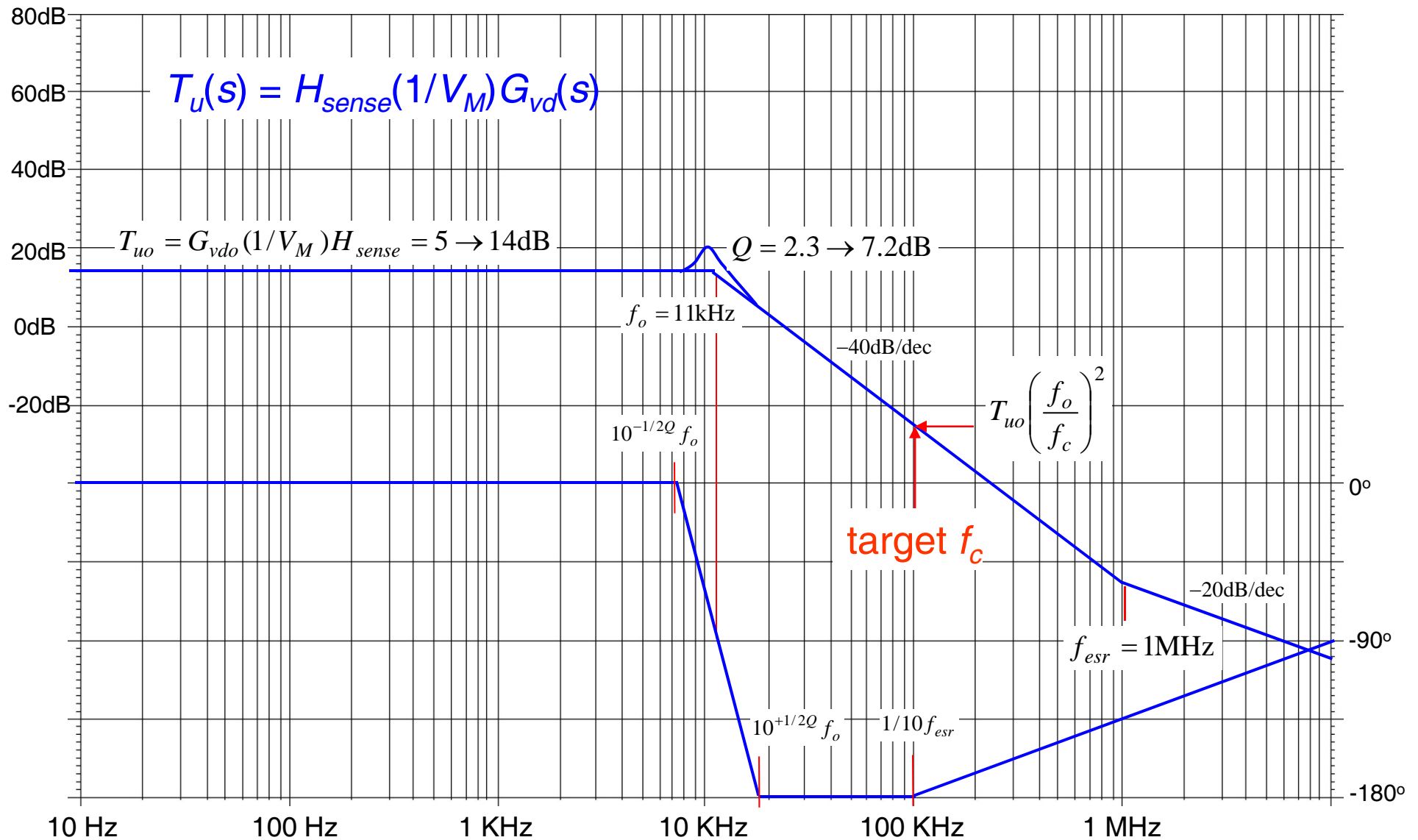
# Uncompensated loop gain $T_u$



$$T_u(s) = H_{sense}(1/V_M)G_{vd}(s)$$

Plot magnitude and phase responses of  $T_u(s)$  to plan how to design  $G_c(s)$

# Magnitude and phase Bode plots of $T_u$



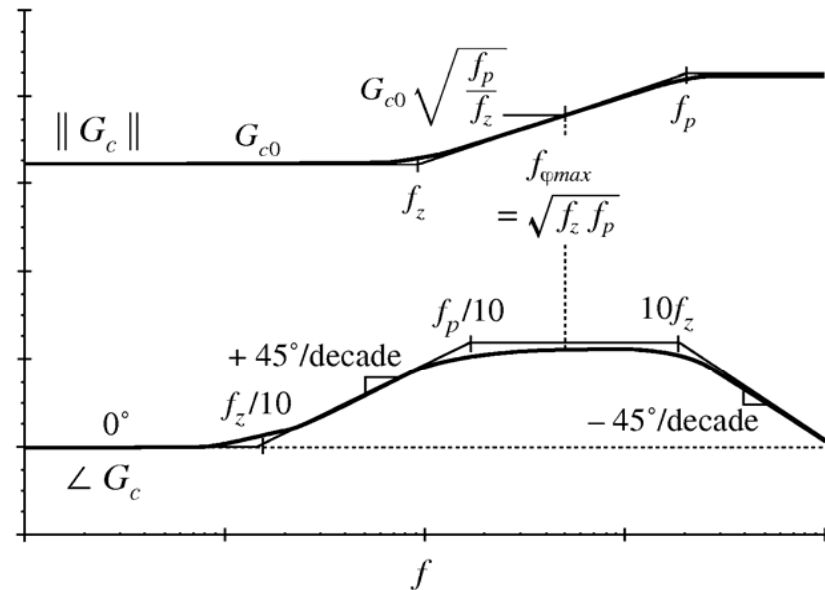
# Lead (PD) compensator design

1. Choose:  $f_c = 100 \text{ kHz}$   
 $\theta = \varphi_m = 53^\circ$

2. Compute:

$$f_z = f_c \sqrt{\frac{1 - \sin(\theta)}{1 + \sin(\theta)}} = 33 \text{ kHz}$$

$$f_p = f_c \sqrt{\frac{1 + \sin(\theta)}{1 - \sin(\theta)}} = 300 \text{ kHz}$$



3. Find  $G_{co}$  to position the crossover frequency:

$$\underbrace{T_{uo} \left( \frac{f_o}{f_c} \right)^2}_{\text{Magnitude of } T_u \text{ at } f_c} \underbrace{G_{co} \sqrt{\frac{f_p}{f_z}}}_{\text{Magnitude of } G_c \text{ at } f_c} = 1 \quad \rightarrow \quad G_{co} = \frac{1}{T_{uo}} \left( \frac{f_c}{f_o} \right)^2 \sqrt{\frac{f_z}{f_p}} = 5.45 \rightarrow 15 \text{ dB}$$

Magnitude of  $T_u$  at  $f_c$     Magnitude of  $G_c$  at  $f_c$

# Lead (PD) compensator summary

---

$$G_c(s) = G_{co} \frac{\left(1 + \frac{s}{\omega_z}\right)}{\underbrace{\left(1 + \frac{s}{\omega_{p1}}\right)}_{\text{Lead compensator}} \underbrace{\left(1 + \frac{s}{\omega_{p2}}\right)}_{\text{HF pole}}}$$

$$G_{co} = 5.45 \rightarrow 15 \text{ dB}$$

$$f_z = 33 \text{ kHz}$$

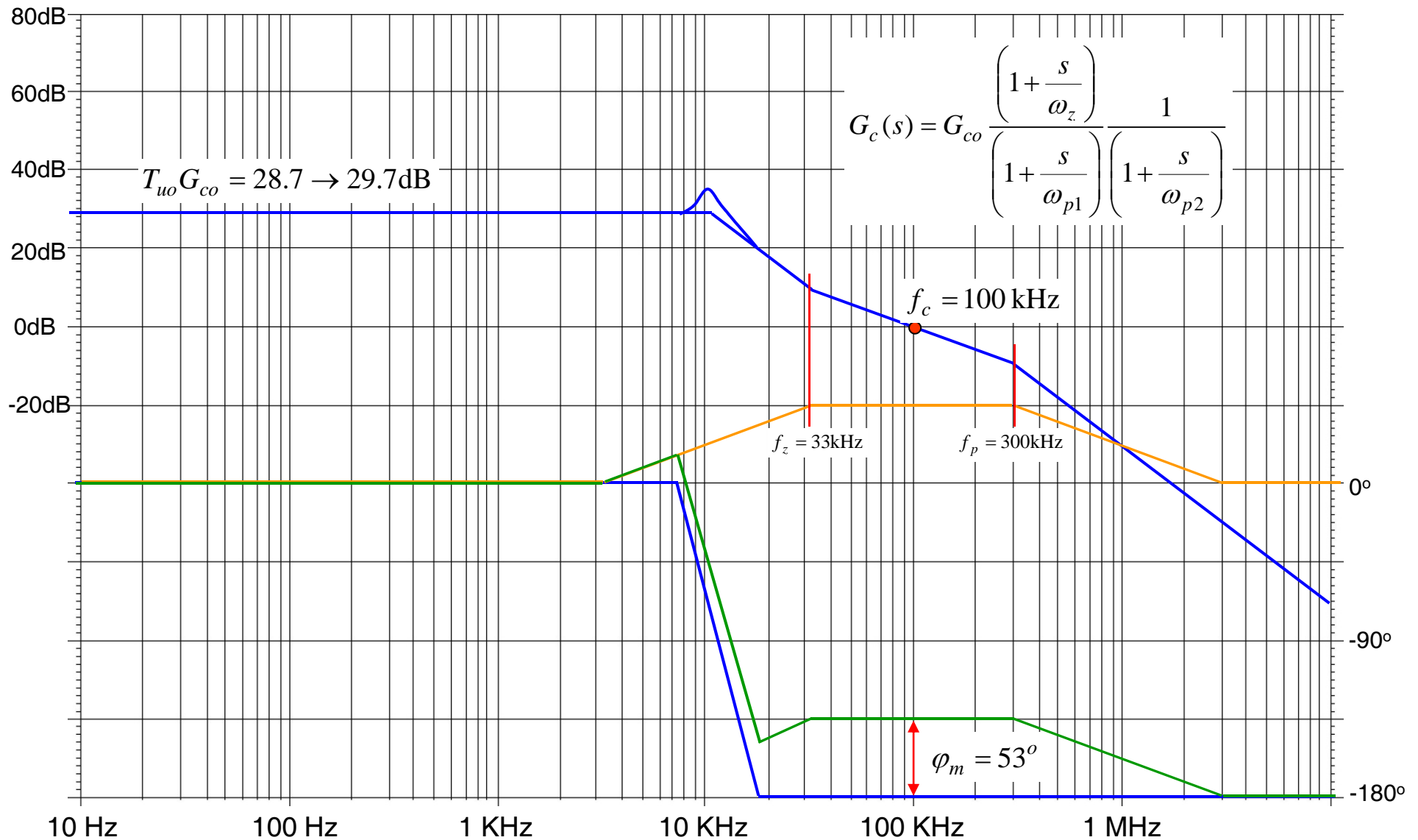
$$f_{p1} = 300 \text{ kHz}$$

$$f_c = 100 \text{ kHz} \quad (=1/10 \text{ of } f_s)$$

High-frequency gain of the lead compensator:  $G_{co} f_{p1}/f_z = 49$  (34 dB)

Added high-frequency pole:  $f_{p2} = 1 \text{ MHz}$  ( $= f_{esr} = f_s$  in this example)

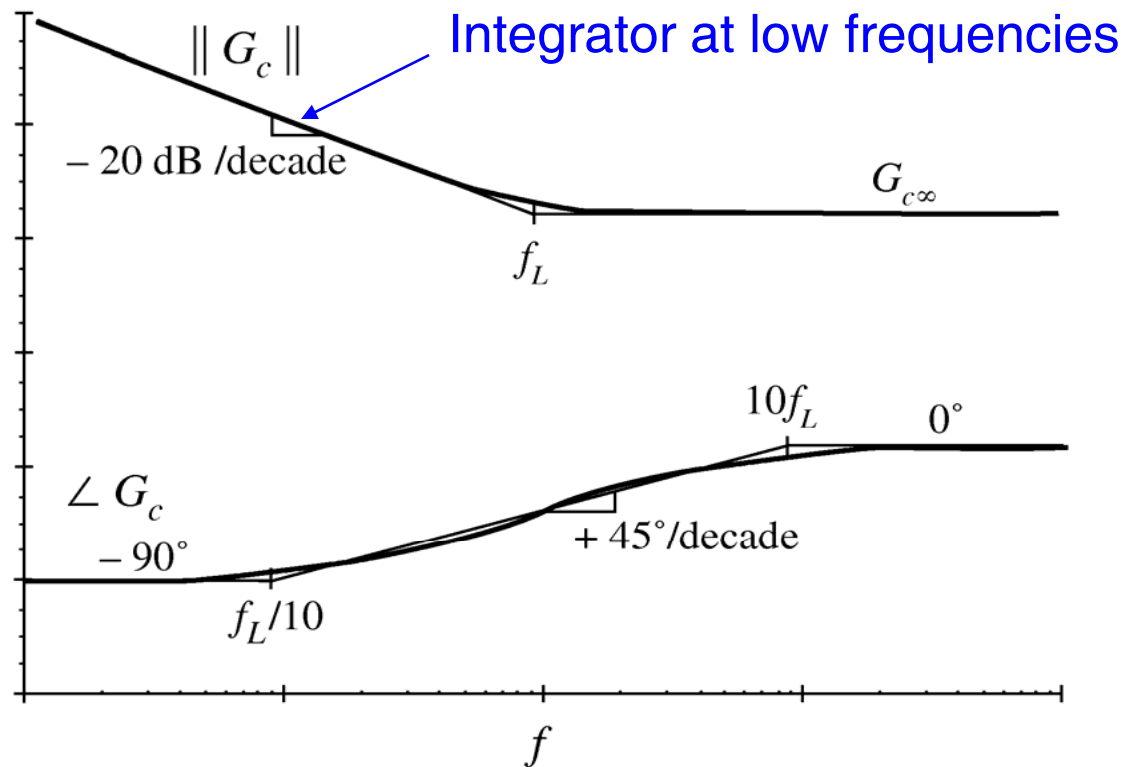
# Loop gain with lead (PD) compensator



# Add lag (PI) compensator

$$G_c(s) = G_{c\infty} \left( 1 + \frac{\omega_L}{s} \right)$$

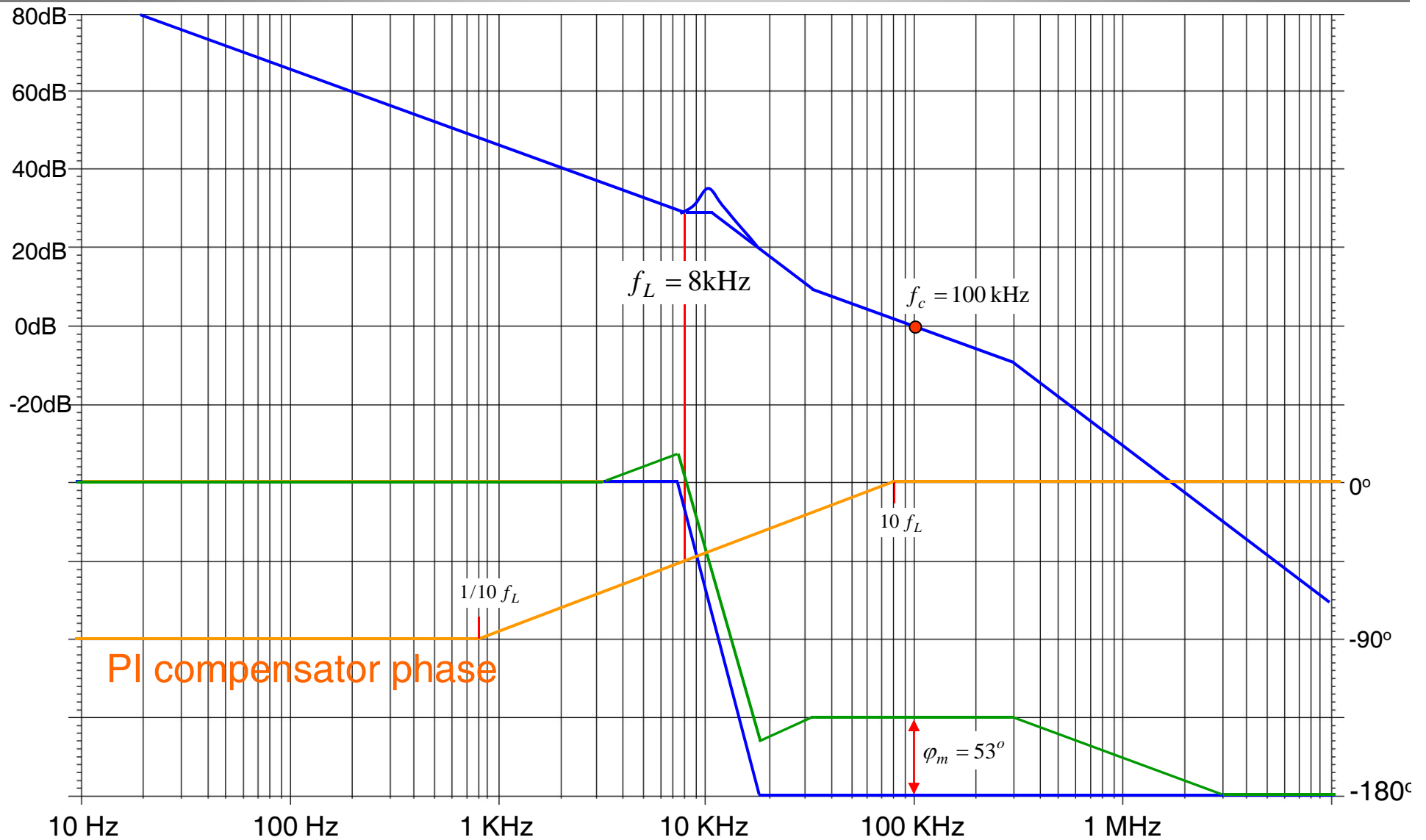
Improves low-frequency loop gain and regulation



Choose  $10f_L < f_c$  so that phase margin stays approximately the same:  $f_L = 8$  kHz

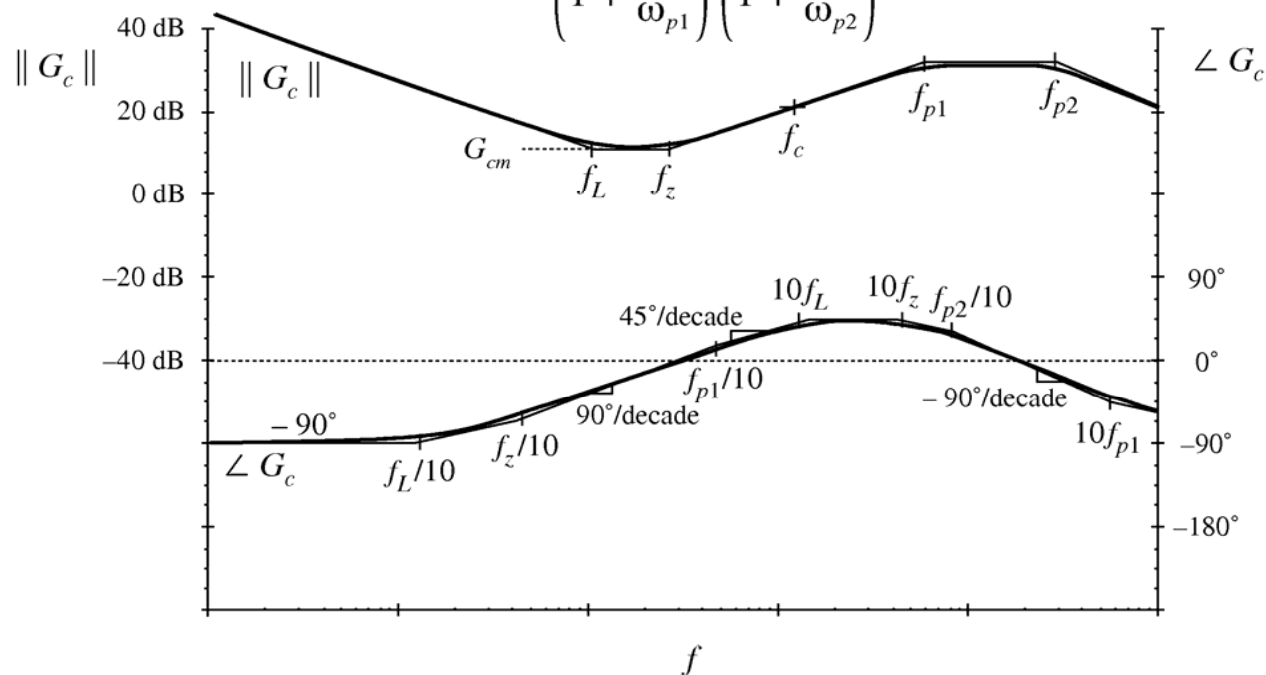
Keep the same cross-over frequency:  $G_{c\infty} = G_{co} = G_{cm} = 5.45 \rightarrow 15$  dB

# Adding PI Compensator



# Complete PID compensator: summary

$$G_c(s) = G_{cm} \frac{\left(1 + \frac{\omega_L}{s}\right) \left(1 + \frac{s}{\omega_z}\right)}{\left(1 + \frac{s}{\omega_{p1}}\right) \left(1 + \frac{s}{\omega_{p2}}\right)}$$



$$G_{cm} = 5.45 \rightarrow 15 \text{ dB}$$

$$f_L = 8 \text{ kHz}$$

$$f_z = 33 \text{ kHz}$$

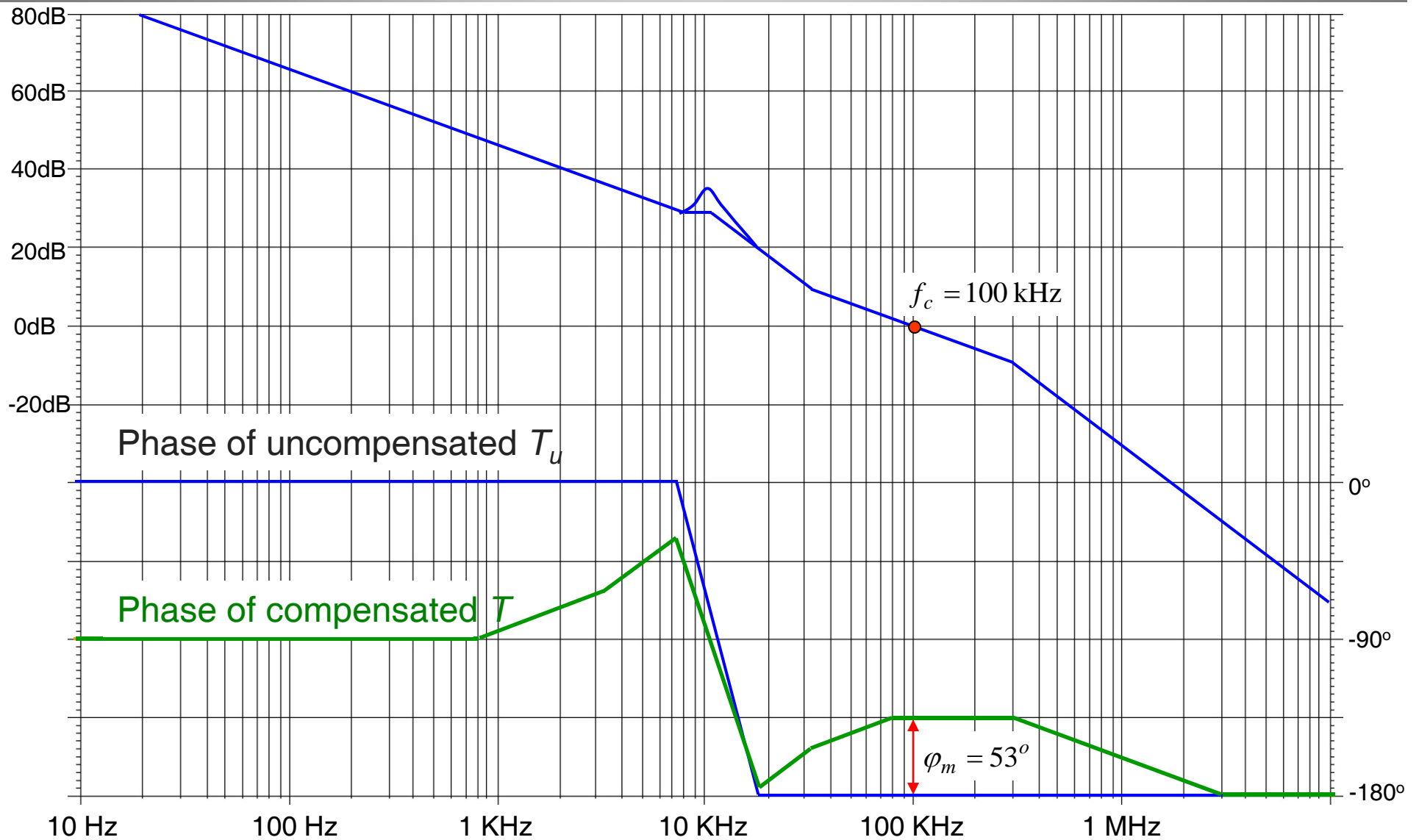
$$f_{p1} = 300 \text{ kHz}$$

$$f_{p2} = 1 \text{ MHz}$$

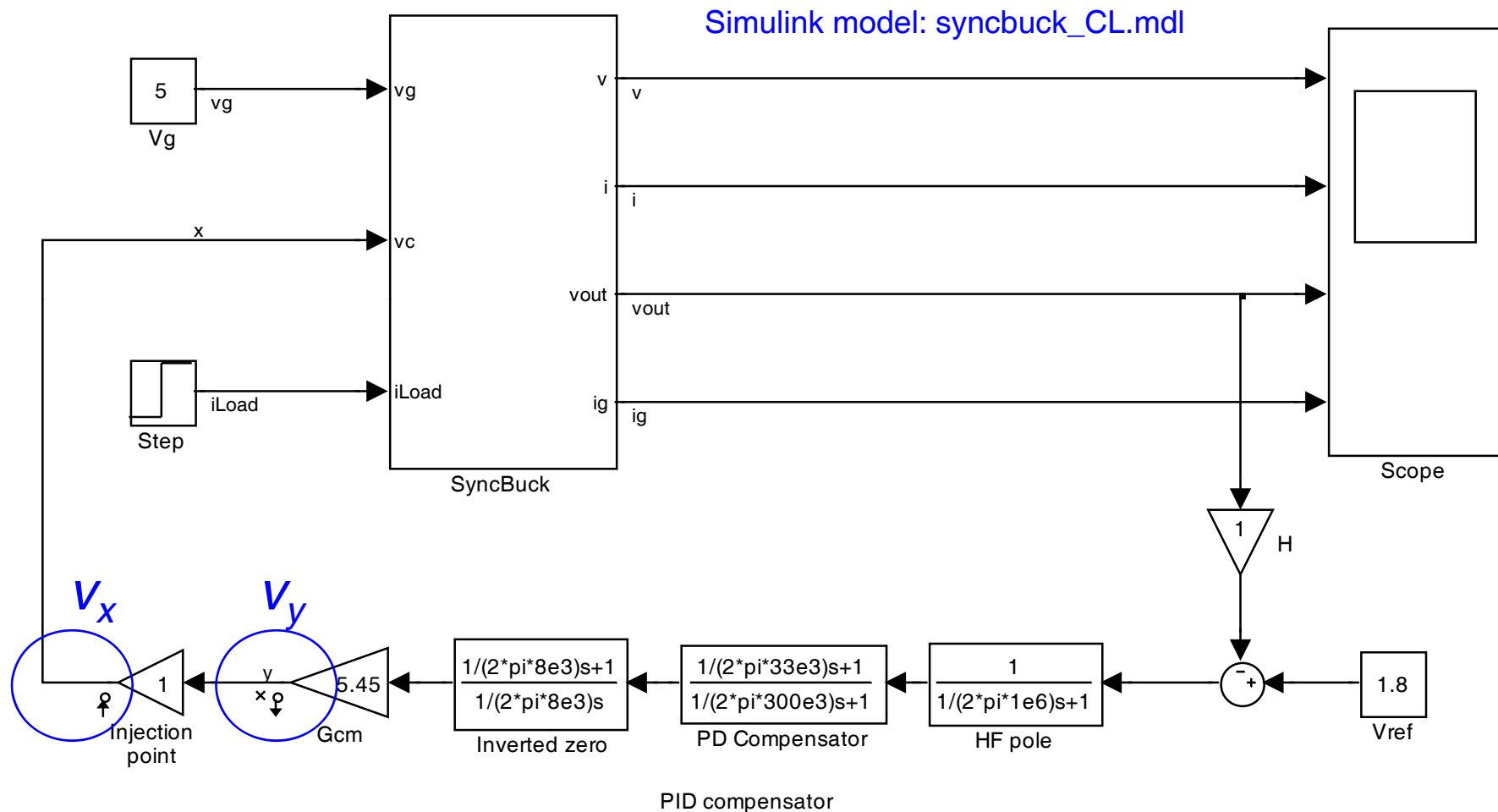
Crossover frequency:  $f_c = 100 \text{ kHz}$  (=1/10 of  $f_s$ )

Phase margin:  $\varphi_m = 53^\circ$

# Magnitude and phase Bode plots of $T$



# Closed-loop voltage regulator in Simulink



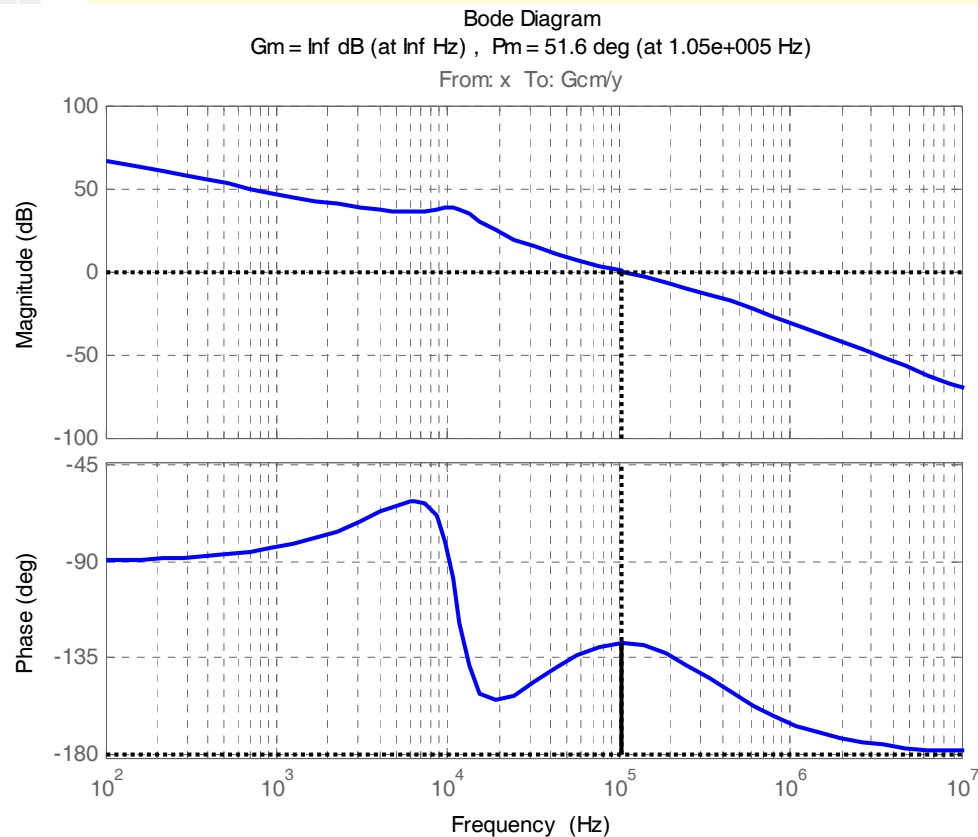
Input and output linearization points for finding the loop-gain,  $T = -v_y/v_x$

The output point (y) should be "Open Loop", as shown by an x symbol next to the output arrow

# Loop gain and stability margins

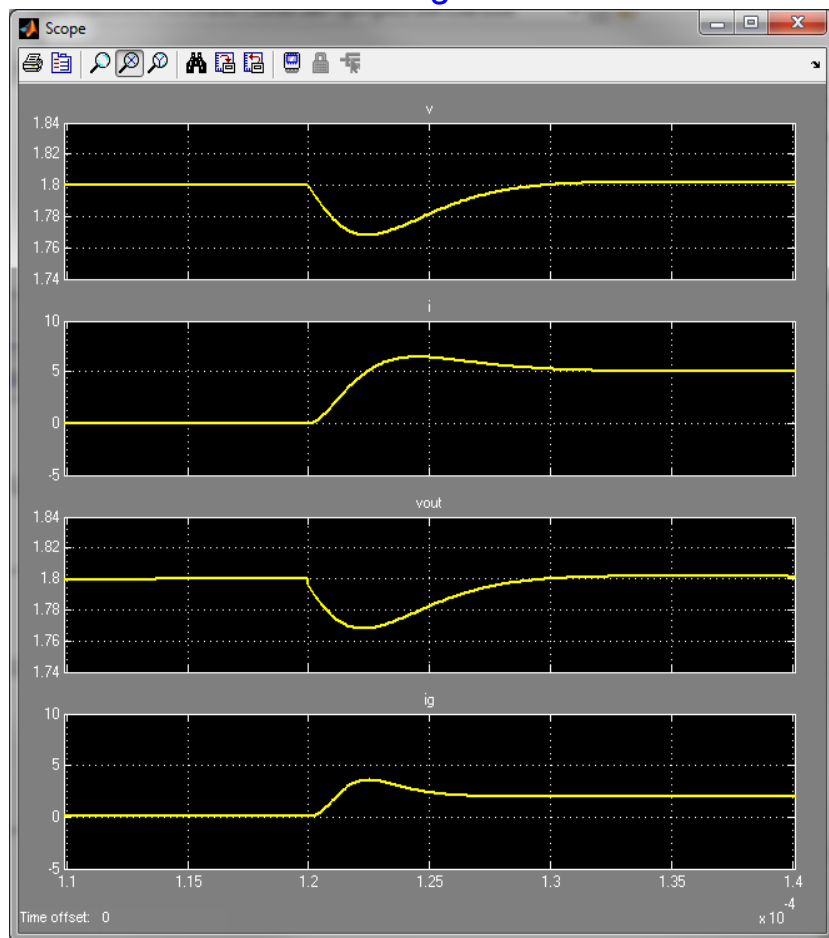
MATLAB script  
BodePlotter\_scriptT.m  
(computes dc op,  
linearizes, calculates  
and plots frequency  
response and stability  
margins)

```
1 %% Loop gain bode plotter using linearization tool
2 % requires Simulink Control Design toolbox
3 %%
4 %%
5 model = 'syncbuck_CL'; % set to file name of simulink model. Must have i/o points set within this model
6 io = getlinio(model) % get i/o signals of model
7 op = operspec(model)
8 op = findop(model,op) % calculate model operating point
9 ssm = linearize(model,op,io) % compute state space model of linearized system
10 %%
11 %%
12 %ltiview('bode',-ssm) % send linearized model to LTI Viewer tool
13 margin(-ssm) % show loop-gain magnitude and phase responses and calculate fc, PM and GM
14 %%
```



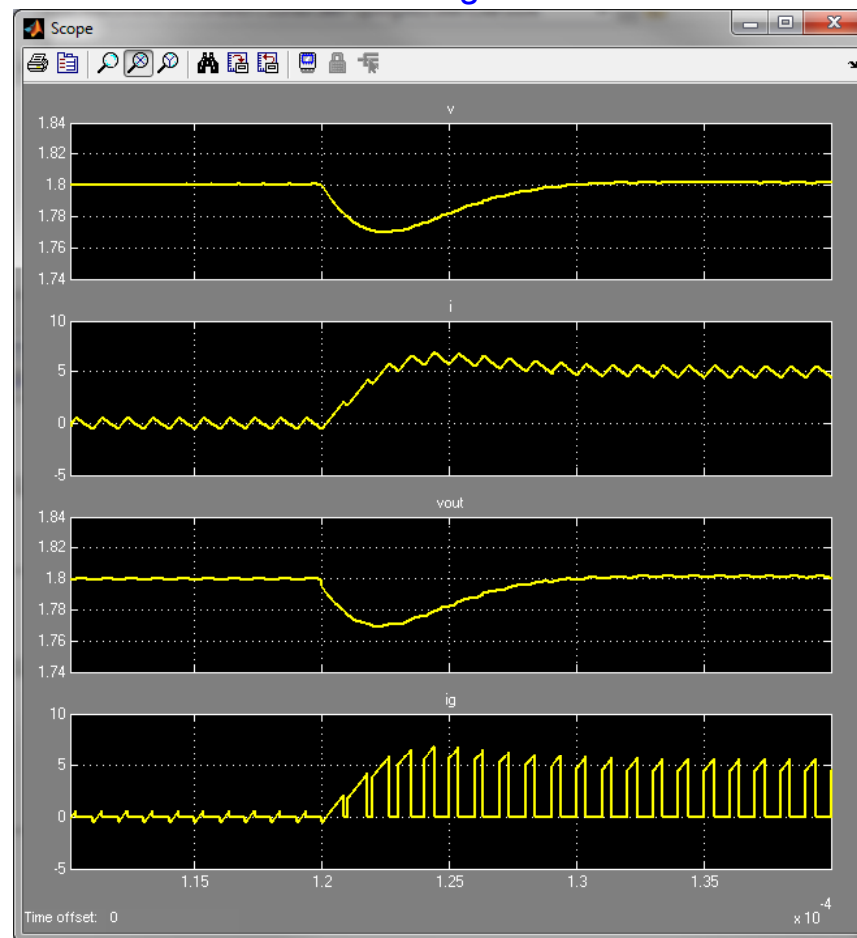
# Closed-loop 0-5 A step-load transient responses

Averaged model



5  $\mu$ s/div

Switching model



5  $\mu$ s/div

See MATLAB/Simulink page on the course website (“Materials” page) for complete step-by-step details, and to download the example files