PI Control Analysis & Design

- *Tracking Response Improvement with PI Control*
- *PI Control Design with Root Locus*
- *PI Control Design with Bode Plots*
- *Concept of Lag Compensator*

PI Control Configuration

Given below is the **basic** form of the **PI** controller.

$$G_{PI} = K_P + \frac{K_I}{s} = K\left(1 + \frac{1}{T_i s}\right) = \frac{K(s + z_1)}{s} = K + \frac{K z_1}{s}$$

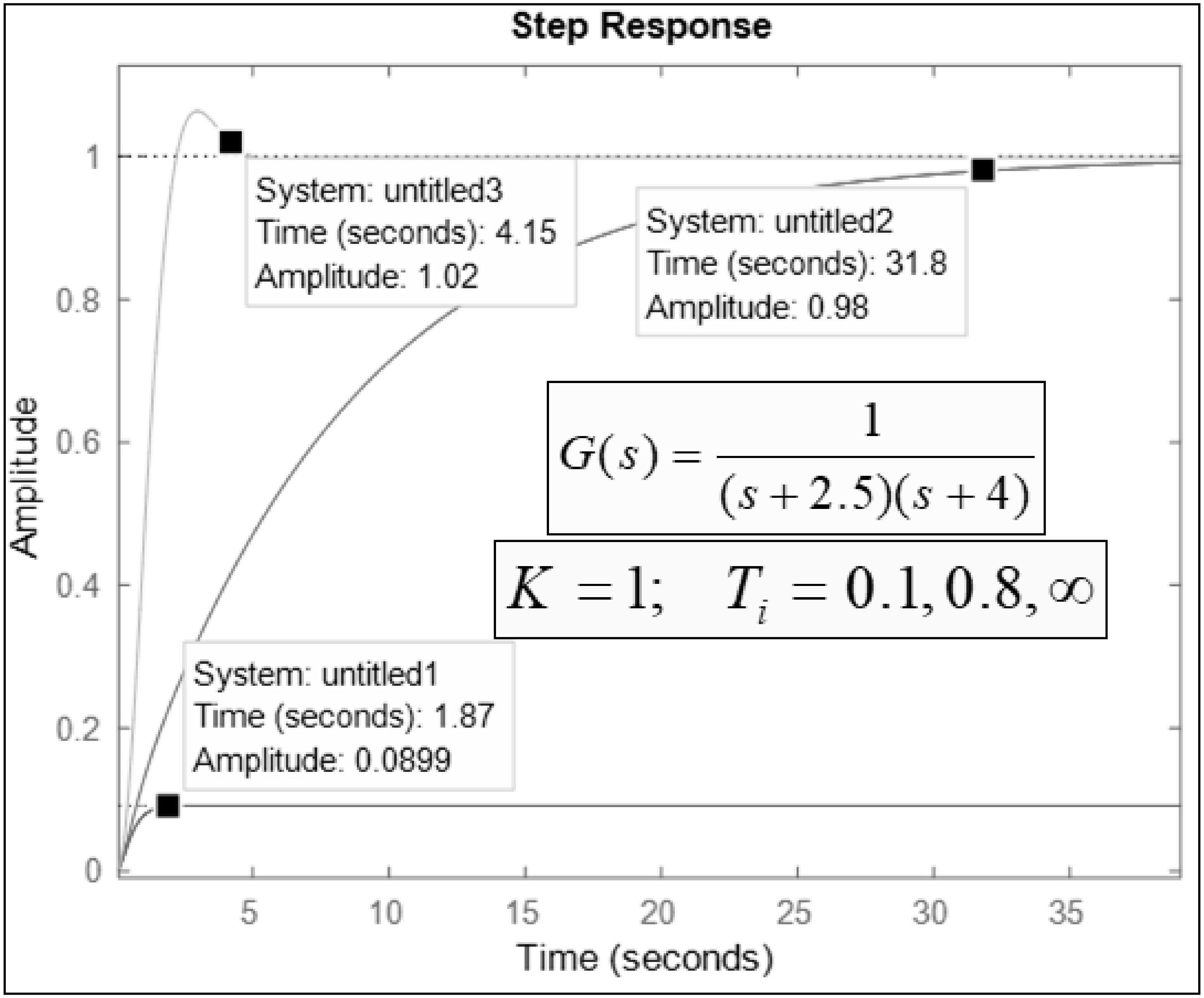
PI controller **adds a pole** at the **origin** and a **zero** at s = - z_1 (**$1/T_i$**), where T_i is the **integrator** time constant.

PI Control Features

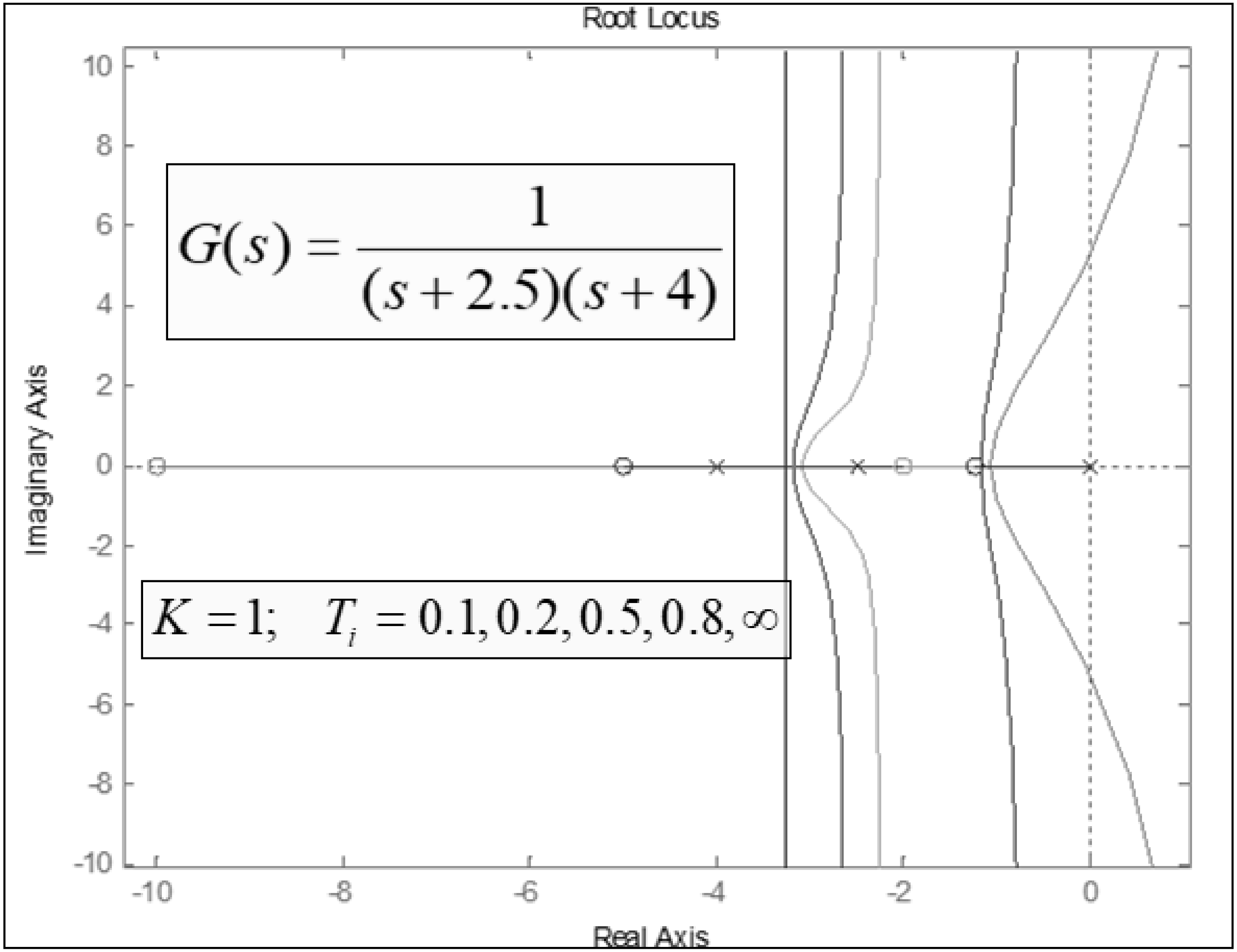
When $\mathbf{T_i \rightarrow 0}$, **PI** controller becomes a pure **integrator.** If it is assumed that **pure** integral controller adds a **zero at infinity,** it is same as $T_i \rightarrow 0$.

As we can see, with **PI** control, **system type** increases by 1, so that **tracking** performance improves **significantly.**

Effect of Integral Gain on Response



Effect of Integral Gain on Root Locus

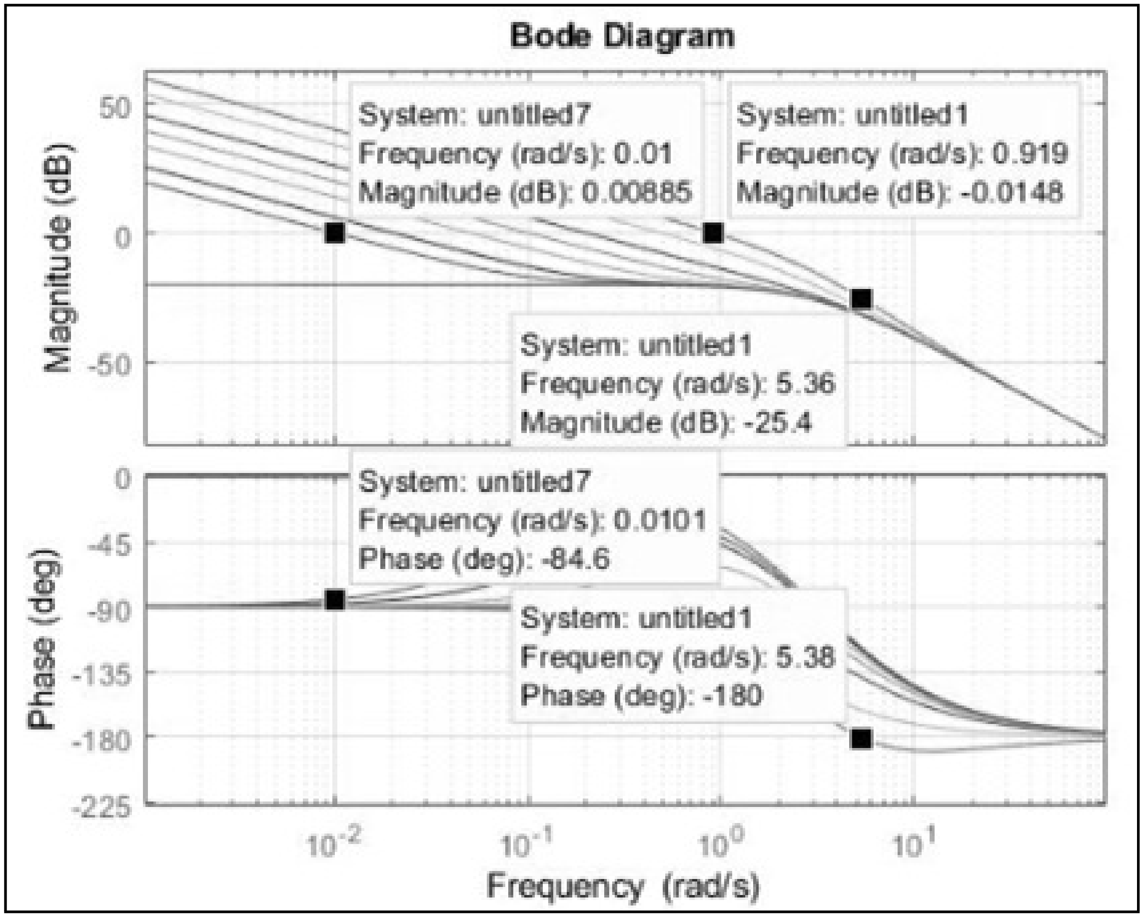


Effect of PI Controller on Bode Plot

$$G(s) = \frac{1}{(s+2.5)(s+4)}; \quad K = 1; \quad T_i = 0.1, 0.2, 0.5, 1, 2, 5, 10$$

To understand the **impact** of **PI** controller on **bode plot**, consider plots with **different** PI controllers as shown alongside.

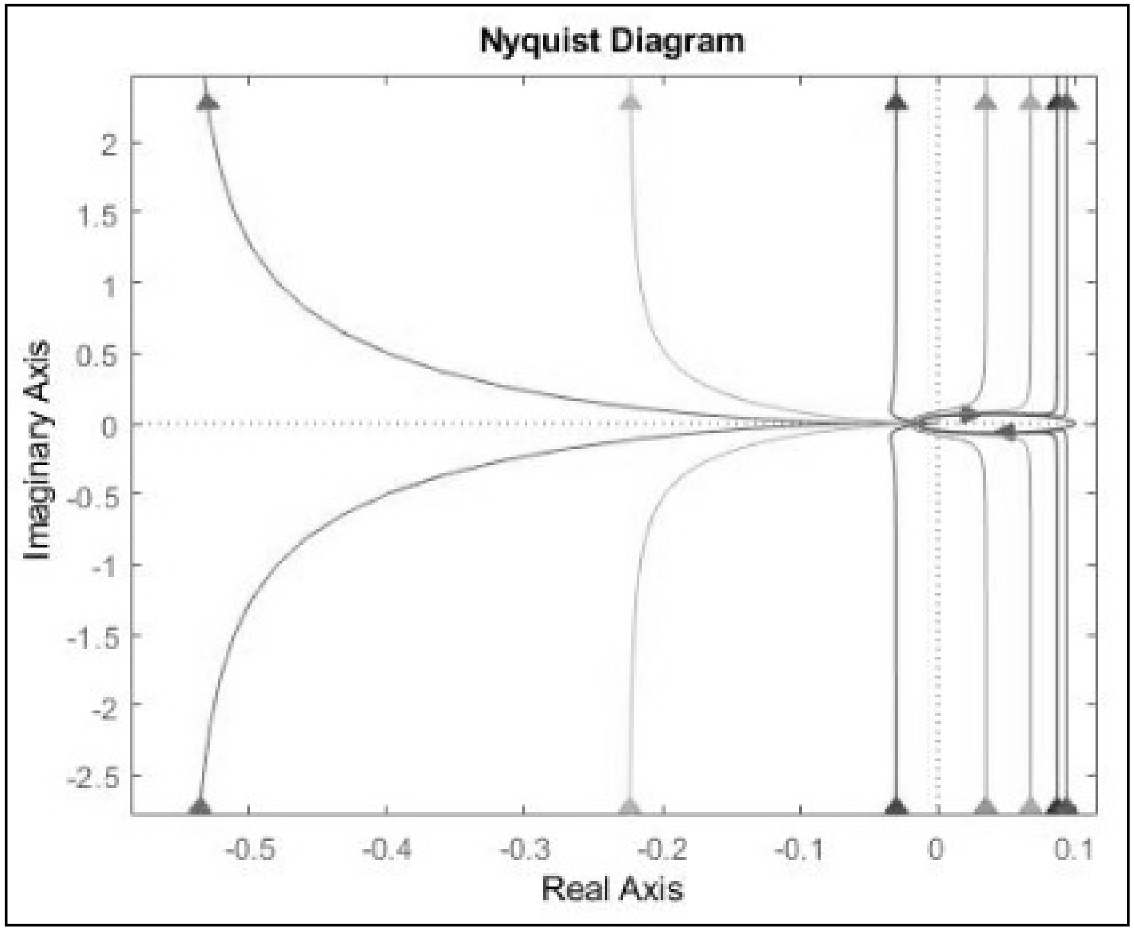
We find that **PI control** reduces both **PCO & GCO.**



Effect of PI Controller on Nyquist Plot

$$G(s) = \frac{1}{(s+2.5)(s+4)}; \quad K = 1; \quad T_i = 0.1, 0.2, 0.5, 1, 2, 5, 10$$

PI controller also **changes** the Nyquist plot, so that **CL resonant peaks** are likely to be significantly **higher.**



PI Control Design Problem Description

Problem of **PI control design** is generally posed in terms of achieving a **specified 'K_V'.**

This is usually **stated** along with required **stability margins** and/or **dominant** system behaviour.

PI controller is used mainly for improving the **tracking** of **step** and **ramp** inputs.

There is a need to ensure **adequate margins / **acceptable **transients**, while designing **PI control**.

PI Design Features

PI control design **aims to increase** system **type,** while improving **'K_V'** for type '**0**' systems.

However, as **PI** is a proper transfer function, overall **system order** is preserved.

Further, PI control **adds** a pole at the origin, while we need to fix the **location** of '**zero**' (or T_i) and the **value** of 'K', based on the **specifications.**

It is to be noted that for **plants** that are already **type '1'**, PI control should be used with **extreme caution.**

Summary

PI controllers are used to achieve **exact** tracking of **step** inputs and **improved** tracking of ramp inputs for **type '0' plants.**

PI Design with Root Lcus

PI Design Steps with Root Locus

Root locus is quite **convenient** for the design of **PI control** as dominant poles are **explicitly visualized.**

This is done by generating **uncompensated** root locus & noting existing **dominant poles** & step error constant.

As **system** type automatically **increases,** we decide the **location** of zero on the **guideline** that the **negative phase** added at existing dominant **poles** is **~3-5^{o}.**

Lastly, **proportional gain** is decided by the amount of **increase** desired in the **'K_V'.**

Root Locus Based PI Design Example

Consider the following **open loop** transfer function.

$$G(s) = \frac{8}{(s+1)(s+5)}$$

Design a PI controller to achieve a $\mathbf{K_v}$ of 8.0, while **maintaining** the existing dominant closed loop **poles** and also determine **changes**, if any.

Existing Dominant Poles:

$$s^2 + 6s + 13 = 0 \rightarrow s_{1,2} = -3 \pm 2j$$

Root Locus Based PI Design Example

In **actual** practice, it is more **convenient** to fix ‘K/T_i’ first, by **imposing** the ‘K_V’ **requirement**, as follows.

$$G_{PI}(s) = \frac{K}{T_i s}(T_i s + 1); \quad \lim_{s \to 0}\left(sG_{PI}(s)G(s)\right) = K_V$$

$$1.6\frac{K}{T_i} = 8 \rightarrow \frac{K}{T_i} = 5 \quad G_{PI}(s) = \frac{5}{s}(T_i s + 1)$$

Thus, we see that **‘T_i** now fixes the ‘**zero**’ location, while leaving **K_V unaffected.**

Root Locus Based PI Design Example

T_i is obtained from **angle condition,** as follows.

$$\angle G_{PI} = \angle(T_i s + 1)|_{s=-3\pm 2j} - \angle s|_{s=-3\pm 2j} = -5^o$$

$$\tan^{-1}\left(\frac{2T_i}{1-3T_i}\right) = \tan^{-1}\left(\frac{2}{-3}\right) - 5^o = 146.3^o$$

$$\frac{2T_i}{1-3T_i} = -0.801 \rightarrow T_i =\sim 2; \quad G_{PI}(s) = \frac{5}{s}(2s+1)$$

Thus, we see that '**zero**' is at -0.5, and **forms** a **doublet** with pole at **origin**, to ensure a **small** negative angle.

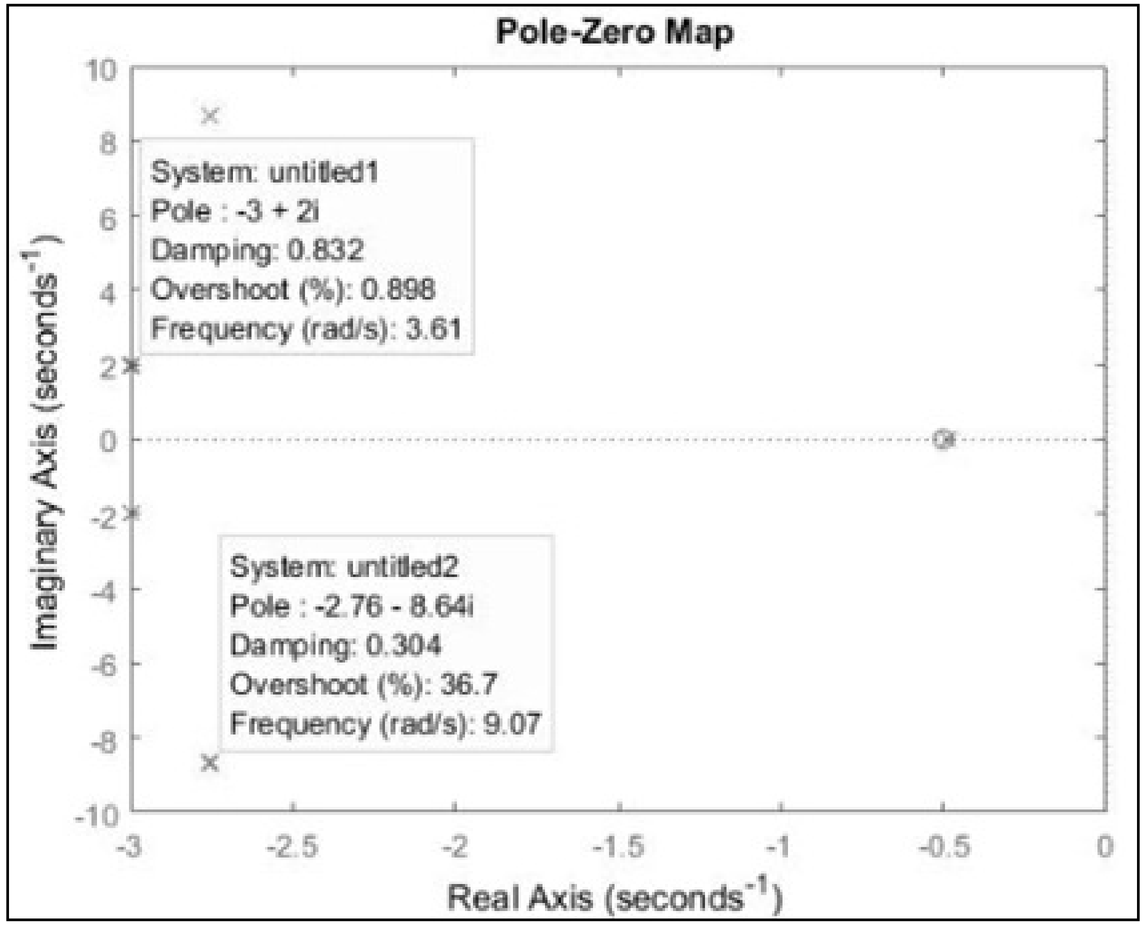
We **note** that while **s = 0** gain is **5**, s = ∞ gain is **10.**

Root Locus Based PI Design Example

Consider **comparison** of uncompensated and compensated **closed loop poles**, as shown alongside.

We find that, while **dominant poles** move marginally **towards origin**, 'ω_d' increases **10 times**, due to gain of 10.

Thus, we can **ensure** only '**σ**' and not '**ω_d**'.

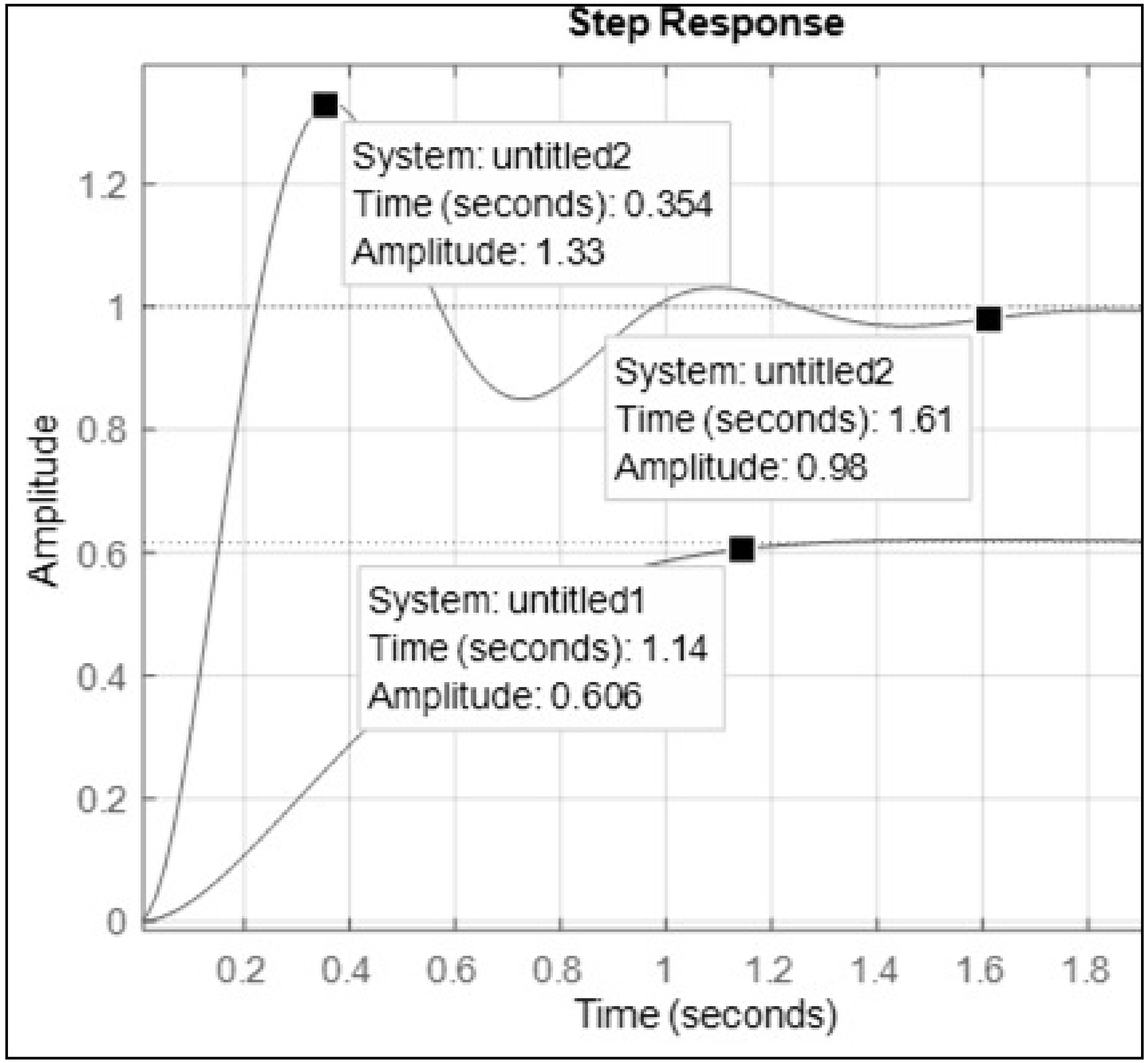


Root Locus Based PI Design Example

Step response of the two **closed loop systems** is shown alongside.

We find that, as expected, **compensated system response** is significantly affected, even though **only -5^o** angle is **added** to pole.

Redesign with -3^o?



Analysis of the Design

We find that **PI** designed using root locus, while ensuring the $\mathbf{K_V}$**,** significantly increases **gain** at higher **frequencies**, and, adversely affects **transient** response.

Therefore, **PI** should be **employed** when the **plant** has sufficient stability **margins.**

Summary

PI controllers are **designed** with root locus to **achieve** desired **ramp error** constant, while keeping the **negative** angle contribution at **dominant** poles, as **small** as possible.

PI Design with Bode Plot

Design of PI with Bode' Plot

PI Controllers can also be **designed** in frequency domain using **GM, PM** as the specifications, along with **ramp error** constant requirement.

Similar to **root locus**, design of **PI** with bode aims to **ensure** that uncompensated **margins** are maintained.

This **results** in the need to **increase** low frequency **gain**, without affecting the high frequency **behaviour.**

In that sense, it aims to **achieve** a better **control**, but also becomes a **bit** more complex than the **root locus** method.

PI Design Methodology with Bode

Design of **PI** in frequency domain is primarily **governed** by the requirements on **low** frequency gain, that is to be achieved for **compensated** system, and is driven by $\mathbf{K_V}$.

In addition, **PM** is required to be **nearly** unchanged so that **existing** transient response is **ensured**.

However, as increase in **K** changes **GCO**, we first compensate for **K**, and **later** choose $(1//T_i)$ such that **GM** is available at $\omega_{\mathbf{PCO}}$ (~1 decade lower than **GCO**).

Bode Plot Based PI Design Example

Consider the following **open loop** transfer function.

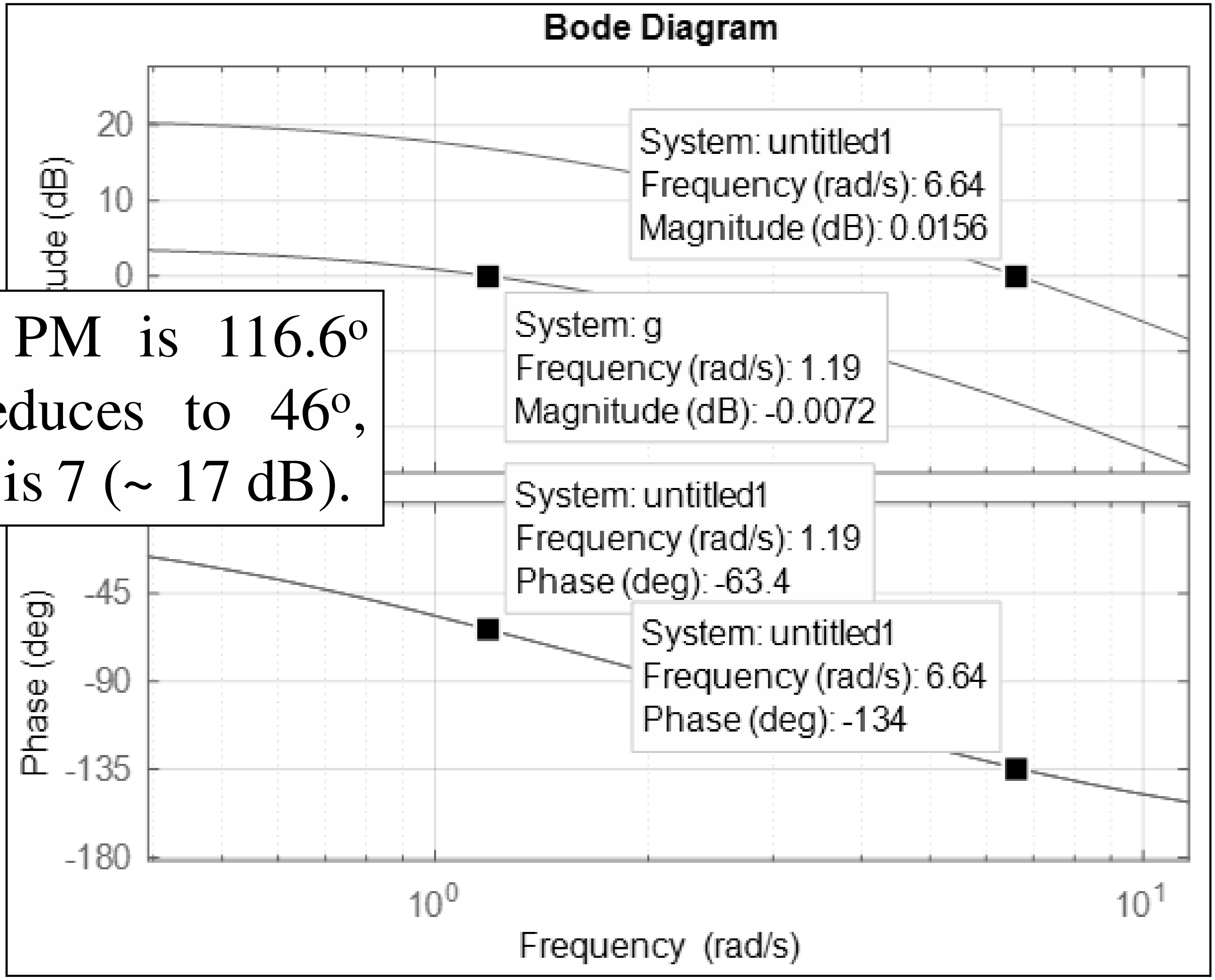
$$G(s) = \frac{8}{(s+1)(s+5)}$$

Design a PI controller to achieve a $\mathbf{K_v}$ of 8.0 and Phase margin of **at least 45°**.

Let us increase overall **gain** by a factor of **7** and assume the **PI** controller of the following **form.**

$$G_{PI}(s) = \frac{7(s+z)}{s}$$

Bode Plot Based PI Design Example



Existing PM is 116.6^o which reduces to 46^o, when K_P is 7 (~ 17 dB).

Bode Plot Based PI Design Example

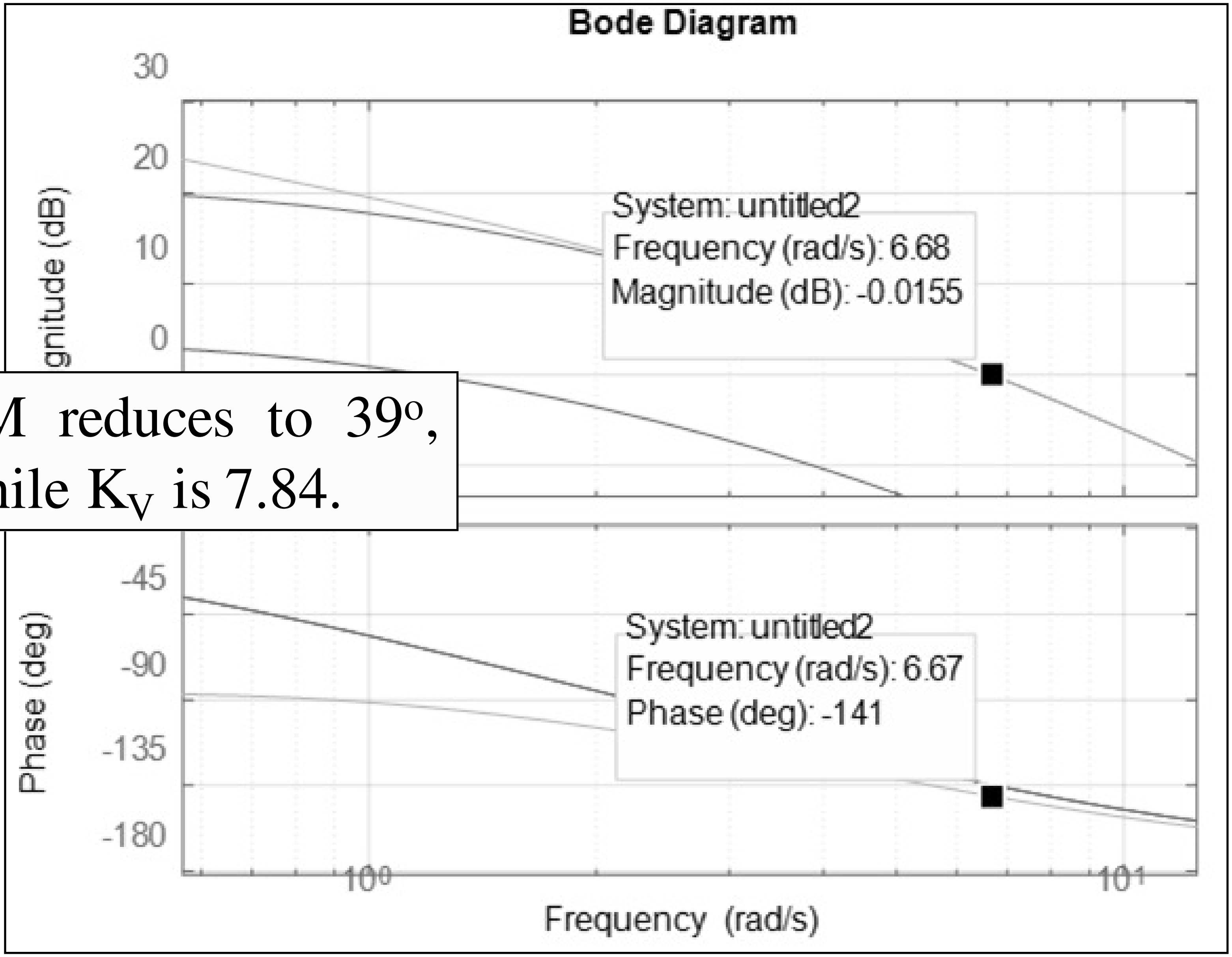
Next, we **choose** a suitable value of **corner frequency**, which turns out to be **0.7.** (GCO = 6.64)

The resulting **PI controller** is as follows.

$$\boxed{G_{PI}(s) = \frac{7(s+0.7)}{s}}$$

It should be noted that the **above PI controller** changes both the $\boldsymbol{\omega_{GCO}}$ **& PM**, as shown next.

Bode Plot Based PI Design Example



PM reduces to 39°, while K_V is 7.84.

Bode Plot Based PI Design Example

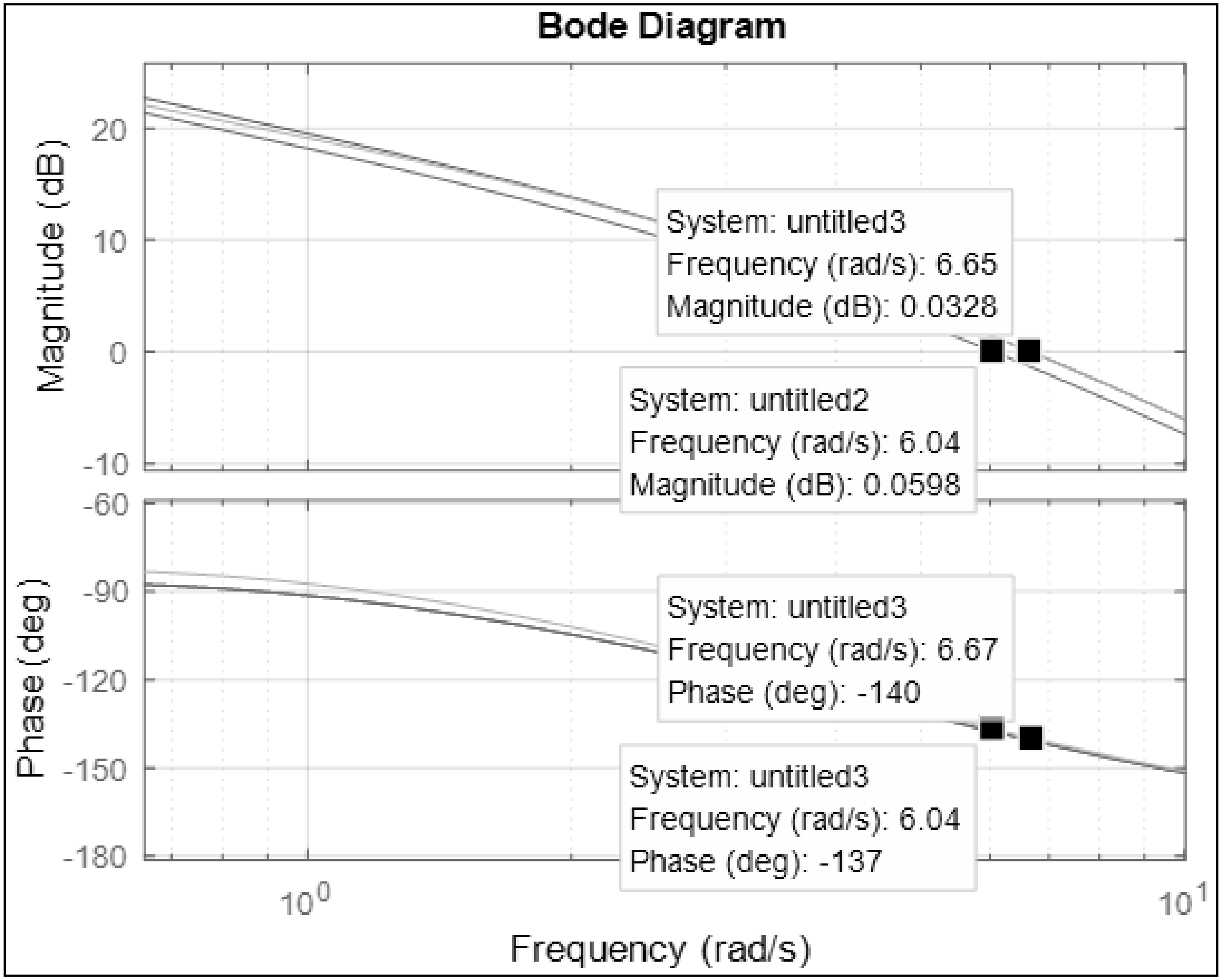
There is now a need to **reduce** the gain to recover the **PM**, which can be done in **two ways.**

Reduce **K to 6** or reduce **'1/T_i' to 0.6**, resulting in the following **controller** options.

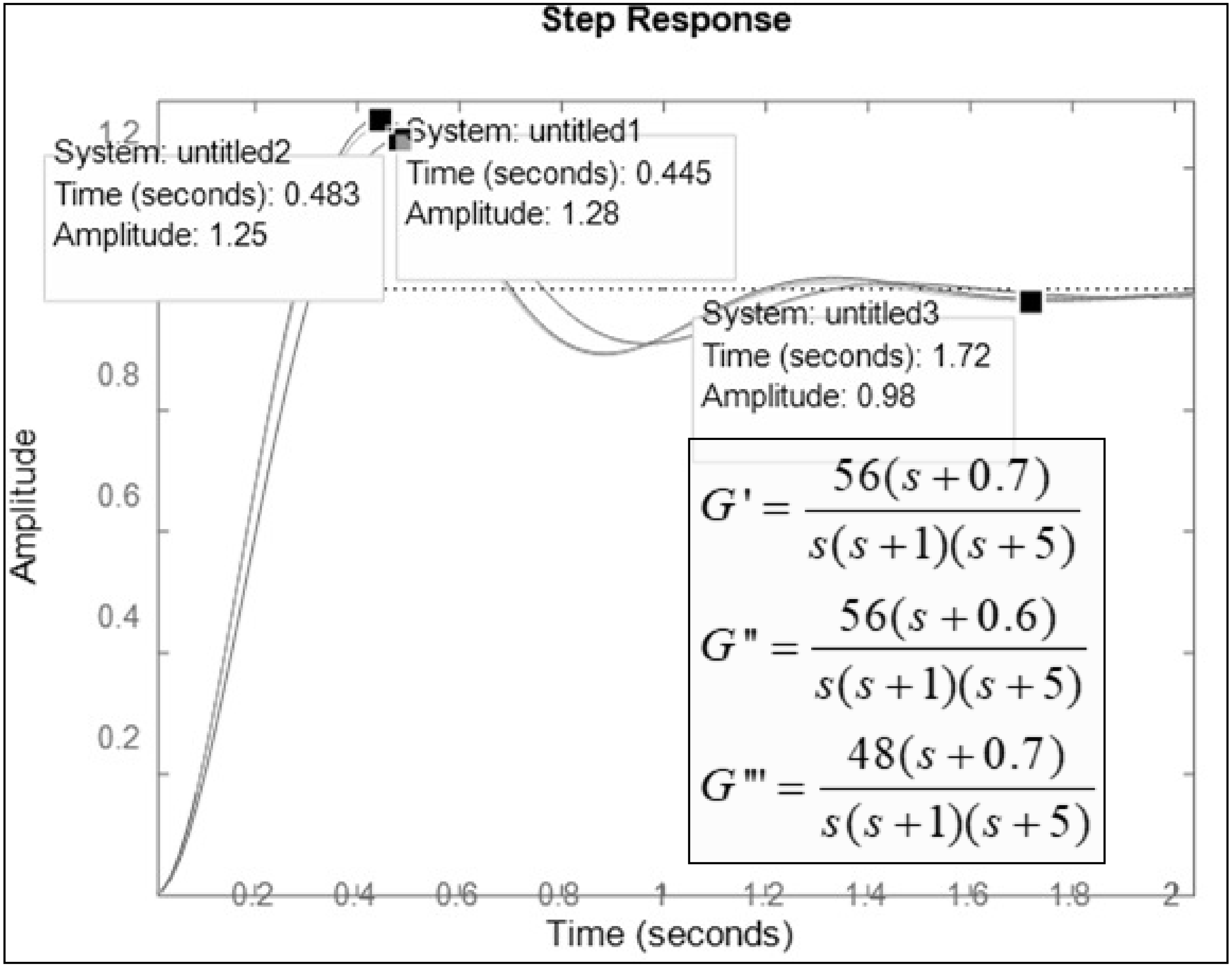
$$G_{PI}(s) = \frac{6(s+0.7)}{s} \quad \text{OR} \quad \frac{7(s+0.6)}{s}$$

In both these cases, the **ramp** error constant **reduces** to 6.72, so that **PM** can be maintained.

Bode Plot Based PI Design Example



Bode Plot Based PI Design Example



Summary

PI controllers are designed with **bod**e in the **low** frequency range, for which **desired** phase margin is the primary design **driver.**

Concept of Lag Compensator

Lag Compensators

PI controller increases **system type**, which may **not be desired** for systems that are already **type '1'** or higher.

Lag compensator is counterpart of **PI**, which improves the **ramp error** constant, without changing system **type.**

The basic **structure** of lag compensator is **as follows.**

$$G_{Lag}(s) = K_c \frac{\beta(Ts+1)}{(\beta Ts+1)} = K_c \frac{\left(s + \frac{1}{T}\right)}{\left(s + \frac{1}{\beta T}\right)}$$

Compensator Structure

Here, $\mathbf{K_c}$ is compensator gain, **T** is the compensator time constant and **(β)** is a parameter that **decides** the amount of **improvement** in ramp error **constant**.

Lag compensator adds a **zero** at s = **-1/T** & a **pole** at s = **-1/ (βT)**, to the plant, so that system **type is preserved.**

Further, as a **bonus**, we also get **additional** design **degrees of freedom**, to better achieve the **specifications.**

Lag Compensator Features

Further, **pole** is closer to the **origin** than **zero**, as $\beta > 1$, so that **lag compensator** adds a net **negative angle** at the **dominant poles.**

We also see that in the **limit** when $\beta \rightarrow \infty$**,** pole lies at the **origin,** which results in **PI controller.**

Similarly, when **T**$\rightarrow$ **0** & $\beta \rightarrow \infty$**, pole** lies at origin, while **zero** lies at **infinity**, resulting in a pure **integral control.**

Compensator Features

Lastly, **lag** compensator has one more **design variable**, in comparison to **PI**, which is expected to **help** in better management of **performance** specifications**.**

It should be noted here that **increase** in error constant is '$\mathbf{K_c}\ \beta$', so that we can get **same tracking** performance, while ensuring **different** transient response.

Effect of 'β' on Step Response

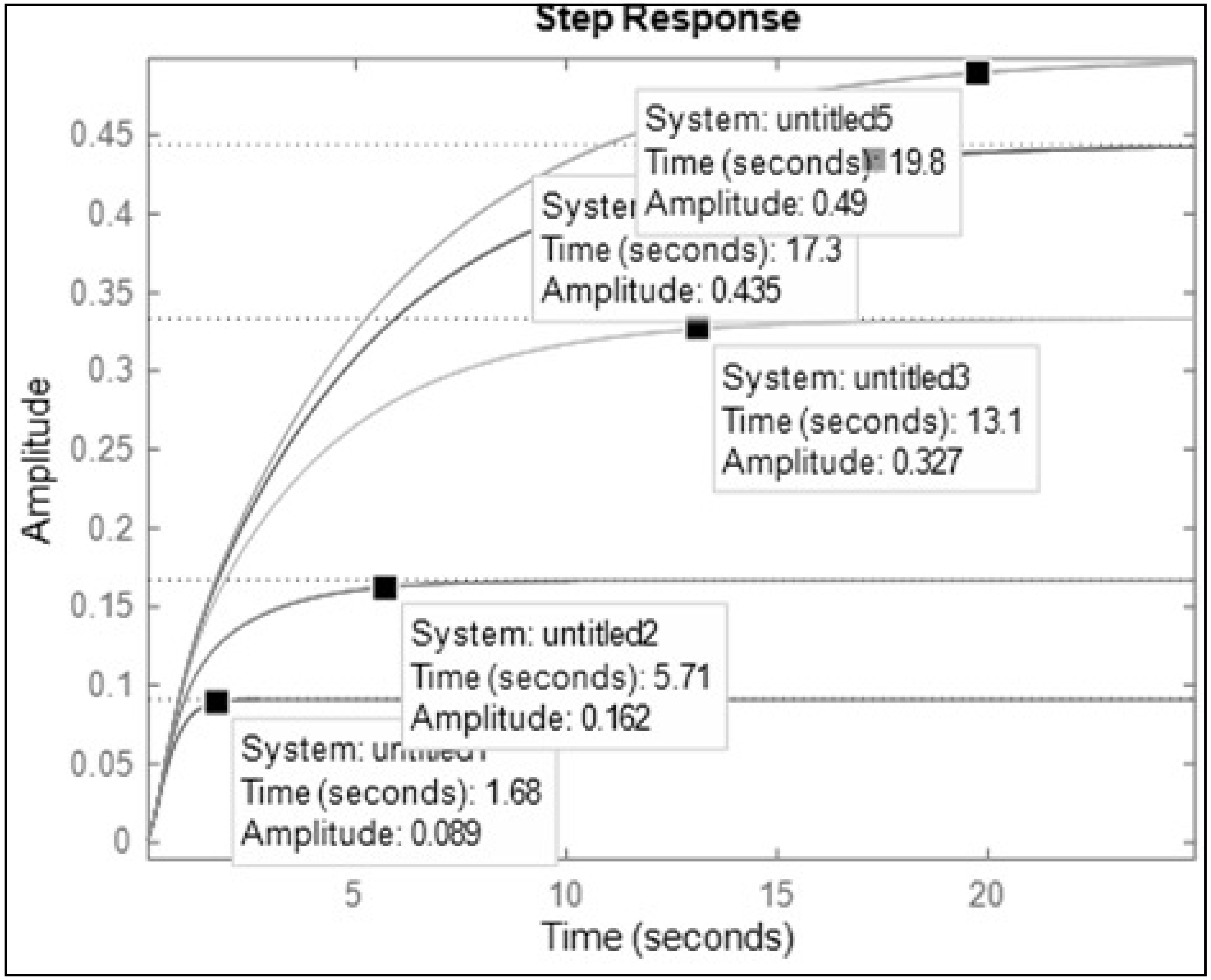
Let us consider the **following plant**, augmented with the **lag compensator.**

$$G = \frac{1}{(s+2.5)(s+4)}$$

$$G_c = \frac{\beta(s+1)}{(\beta s+1)}$$

$$\beta = 1, 2, 5, 8, 10$$

Step response, is shown alongside, which **brings out** the effect of **β.**



Effect of 'T' on Step Response

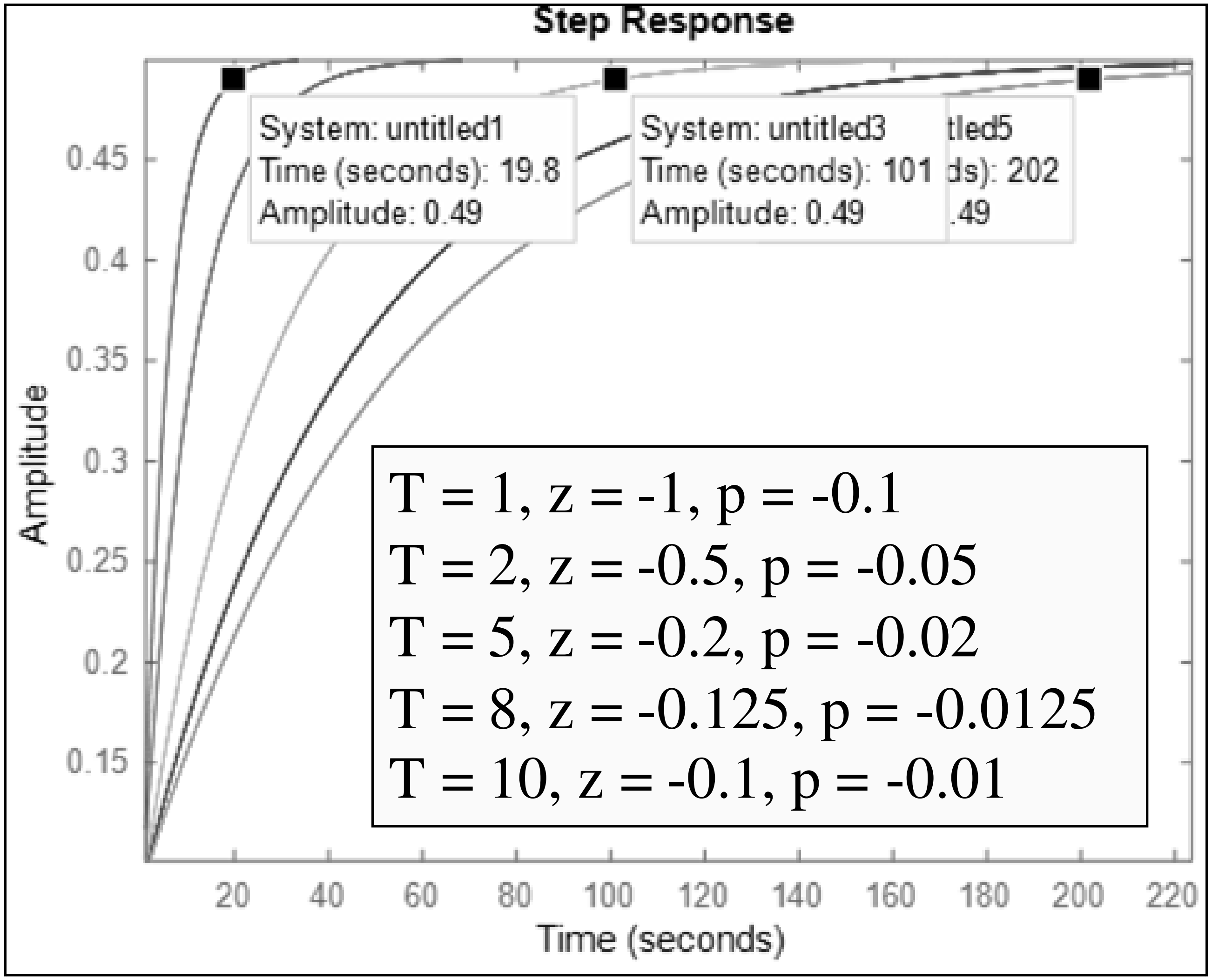
Let us now fix β = **10** and examine the **effect of 'T',** as shown below & **alongside.**

$$G = \frac{1}{(s+2.5)(s+4)}$$

$$G_c = \frac{10(Ts+1)}{(10Ts+1)}$$

$$T = 1, 2, 5, 8, 10$$

Step response, **brings out** the fact that an **increase in 'T' worsens** the settling time**.**

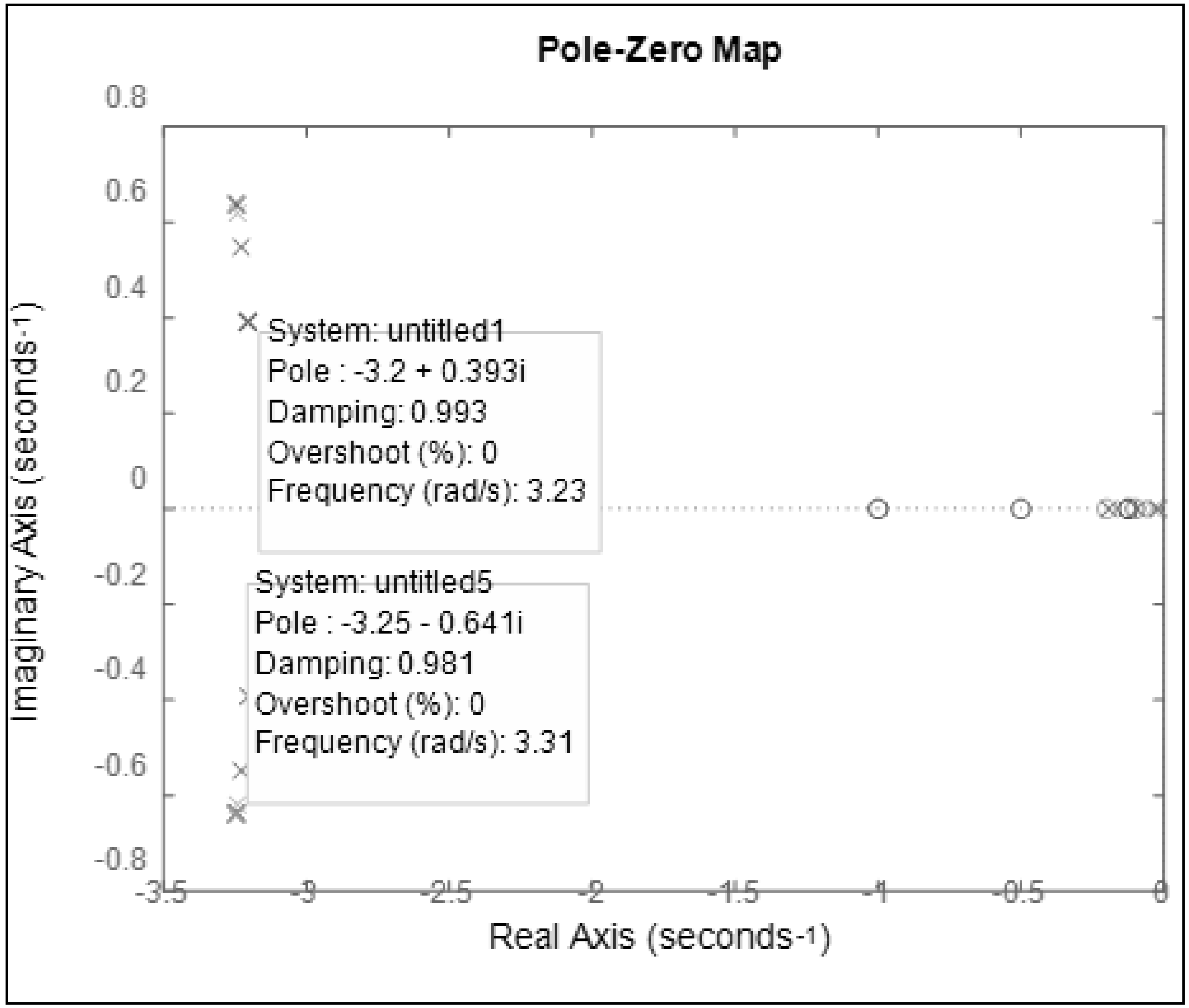


T = 1, z = -1, p = -0.1
T = 2, z = -0.5, p = -0.05
T = 5, z = -0.2, p = -0.02
T = 8, z = -0.125, p = -0.0125
T = 10, z = -0.1, p = -0.01

Effect of 'T' on P-Z Map

Let us examine the **closed** loop poles, as shown **alongside.**

We see that similar to **PI**, while '**σ**' is nearly constant, 'ω_d' **increases** with increase in '**T**'.

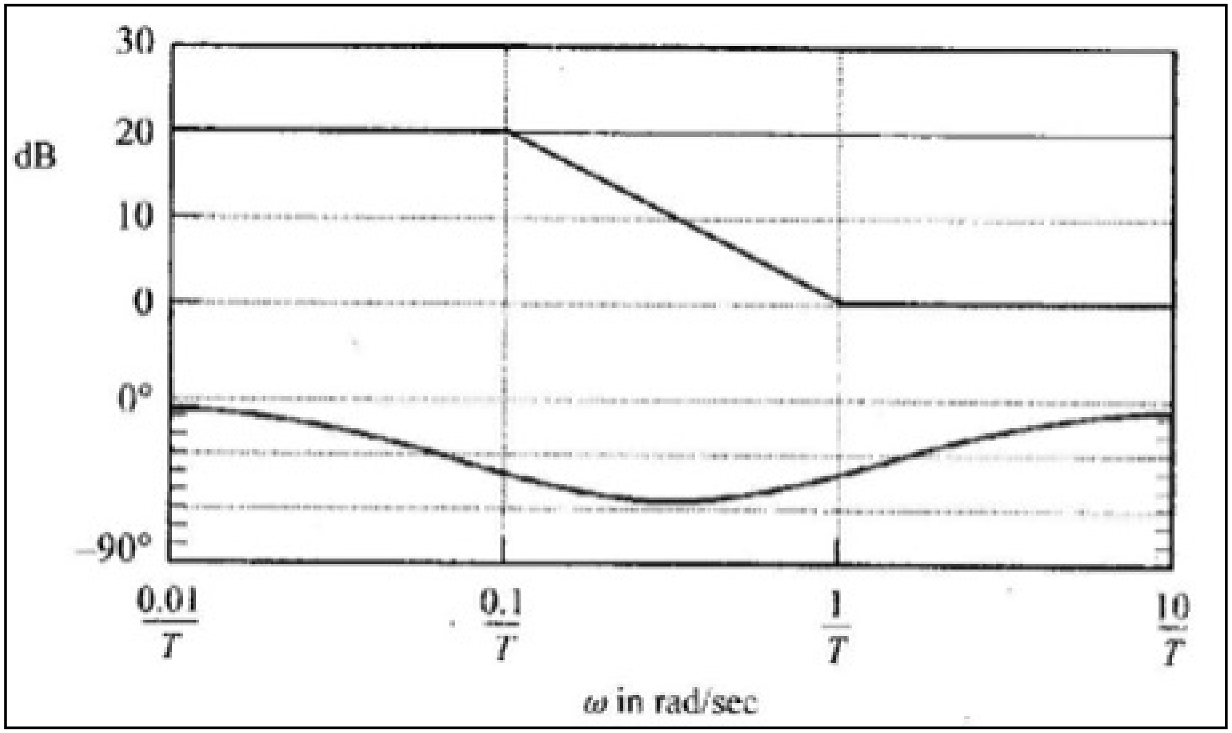


Generic Lag Compensator Bode Plot

Bode plot of **lag** compensator (**K_C = 1 & β = 10**), is shown alongside.

We find that **beyond '1/T'**, gain is **small**, so that **GCO** is unlikely to change if '**1/T**' is kept **small.**

We also see that within **1-decade** after '1/T', the **phase lag** also is **nearly zero.**



$$G_{Lag}(s) = \frac{\left(s + \frac{1}{T}\right)}{\left(s + \frac{1}{\{10T\}}\right)}$$

Summary

It is found that **both β and T** increase the **settling time,** which needs to be **suitably managed.**

This **behaviour** is due to the **dominant** nature of compensator **pole-zero** combination**.**