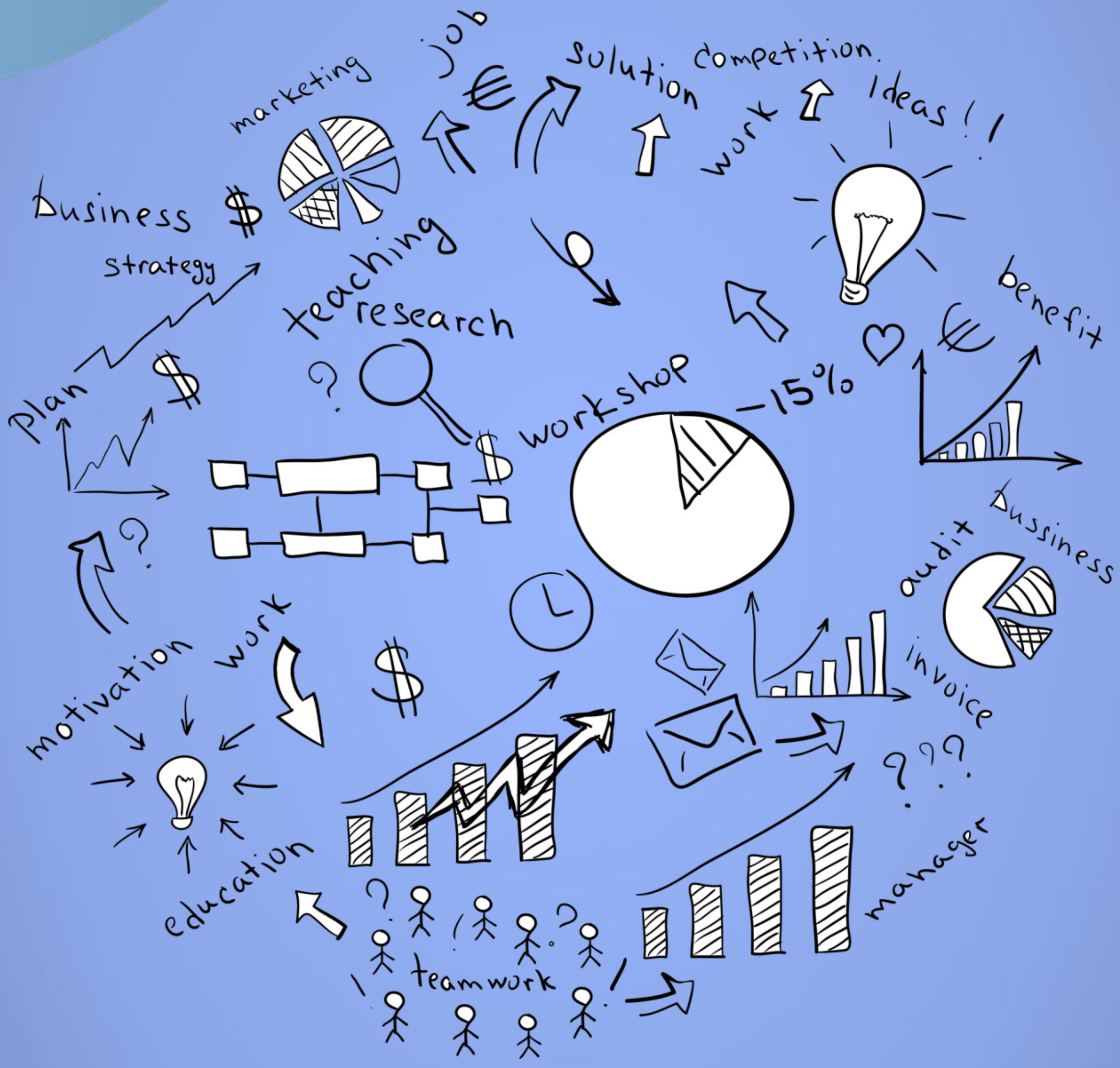
# PID Controller




**Name: عمرو أيمن صلاح الدين المغربي**

# PID-Controller:

A proportional–Integral–Derivative controller (PID controller) is a control loop mechanism employing feedback that is widely used in industrial control systems and a variety of other applications requiring continuously modulated control. A PID controller continuously calculates an error value as the difference between a desired setpoint (SP) and a measured process variable (PV) and applies a correction based on proportional(P), integral(I), and derivative(D) terms.

It is used in industrial control applications to regulate temperature, flow, pressure, speed, and other process variables.

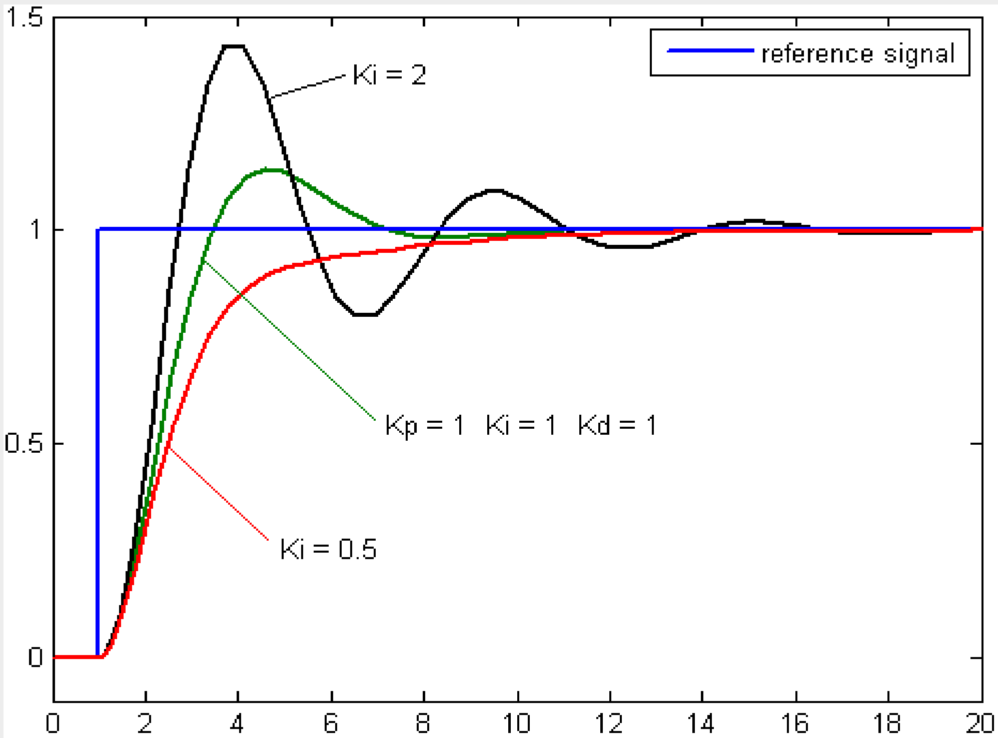
The purpose of a PID controller is to force feedback to match a setpoint, such as a thermostat that forces the heating and cooling unit to turn on or off based on a set temperature. PID controllers are best used in systems which have a relatively small mass and those which react quickly to changes in the energy added to the process. It is recommended in systems where the load changes often and the controller is expected to compensate automatically due to frequent changes in setpoint, the amount of energy available, or the mass to be controlled.

The working principle behind a PID controller is that the proportional, integral, and derivative terms must be individually adjusted (tuned) based on the difference between these values a correction factor is calculated and applied to the input. For example, if an oven is cooler than required, the heat will be increased. Here are the functions of each step:

1. **Proportional:** involves correcting a target proportional to the difference. Thus, the target value is never achieved because as the difference approaches zero, so too does the applied correction.

2. **Integral:** attempts to remedy this by effectively cumulating the error result from the "P" action to increase the correction factor. For example, if the oven remained below temperature, “I” would act to increase the head delivered. However, rather than stop heating when the target is reached, "I" attempts to drive the cumulative error to zero, resulting in an overshoot.

3. **Derivative:** attempts to minimize this overshoot by slowing the correction factor applied as the target is approached.




There are two techniques for PID-Controller system:

1- SISO:  Single Input Single Output system
2- MIMO: Multi Input Multi Output system

Single-input single-output (SISO) proportional integral derivative (PID) control is the automatic control scheme most widely used in practice. It has only three parameters to tune and achieves reasonable or reliable performance on a wide variety of plants. The effect of the tuning parameters (or gains) on the closed-loop performance is well-understood, and there are well-known simple rules for tuning these parameters. Systems for automatically tuning SISO PID controllers have been developed and are available in commercial controllers.

Single-input single-output PID controllers have been used for multiple-input multiple-output (MIMO) plants. This is generally carried out by pairing inputs (actuators) and outputs (Sensors) and connecting them with SISO PID controllers. These SISO PID controllers can be tuned one at a time (in ‘successive loop closure’) using standard SISO PID tuning rules. For MIMO plants that are already reasonably well-decoupled, multi-loop SISO PID design can work well. Unlike SISO PID design, however, MIMO PID design is more complex; the SISO loops must be chosen carefully, and then tuned the correct way in the correct order. An alternative to multi-loop SISO PID control is to design one MIMO PID controller, which uses matrix coefficients, all at once. Such a controller potentially uses all sensors to drive all actuators, but it is possible to specify a simpler structure by imposing a sparsity constraint on the controller gain matrices. Like SISO PID controllers for SISO plants, MIMO PID controllers can achieve particularly good performance on a wide variety of MIMO plants, even when the plant dynamics are quite coupled. The challenge is in tuning MIMO PID controllers, which require the specification of three matrices, each with a number of entries equal to the number of plant inputs times the number of plant outputs. For example, a MIMO PID controller for a plant with four inputs and four outputs requires the specification of up to 48 parameters. This would be exceedingly difficult, if not impossible, to tune by hand one parameter at a time. Hand tuning a MIMO PID controller with 10 inputs and 10 outputs would be impossible in practice.

## Example for the MIMO system:

describing numerical results for a classic MIMO plant, the two-input two-output Wood–Berry binary distillation column. The computations were carried out using the MATLAB-based convex.
The plant transfer function is:

Each entry is a first-order system with a time delay. The dynamics are quite coupled, so finding a good MIMO PID controller is not simple.

$$P(s) = \begin{bmatrix} \dfrac{12.8e^{-s}}{16.7s+1} & \dfrac{-18.9e^{-3s}}{21.0s+1} \\ \dfrac{6.6e^{-7s}}{10.9s+1} & \dfrac{-19.4e^{-3s}}{14.2+1} \end{bmatrix}.$$

$$S_{\max} = 1.4, \quad T_{\max} = 1.4, \quad Q_{\max} = 3/\sigma_{\min}(P(0)) = 0.738.$$

The derivative action time constant is chosen to be t=0.3. The semi-infinite constraints are sampled using N = 300 logarithmically spaced frequency samples in the interval [$10^{-3}$; $10^{3}$] . The initial design uses method with € = 0.01. The algorithm converges in seven iterations to the values:

$$K_P = \begin{bmatrix} 0.1750 & -0.0470 \\ -0.0751 & -0.0709 \end{bmatrix}, \quad K_I = \begin{bmatrix} 0.0913 & -0.0345 \\ 0.0402 & -0.0328 \end{bmatrix}, \quad K_D = \begin{bmatrix} 0.1601 & -0.0051 \\ 0.0201 & -0.1768 \end{bmatrix},$$

which achieve objective value $\|(P(0)K_I)^{-1}\| = 2.25$. The resulting closed-loop transfer function singular values are plotted versus frequency in Figure 3, along with the imposed limits.

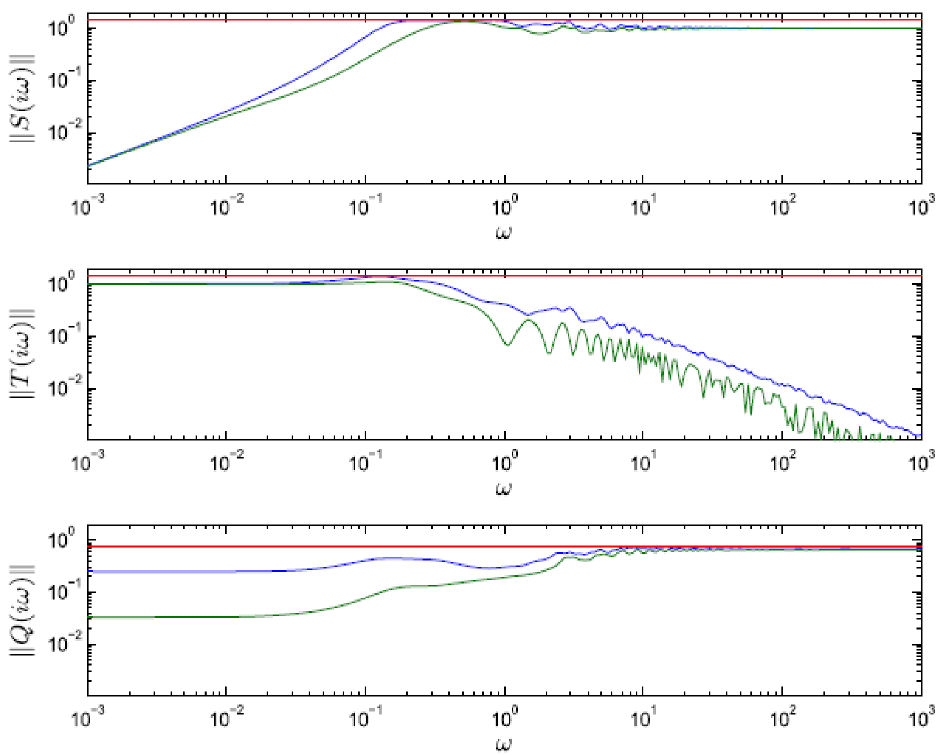



Figure 3. Closed-loop transfer function singular values versus frequency, with constraints shown in red.

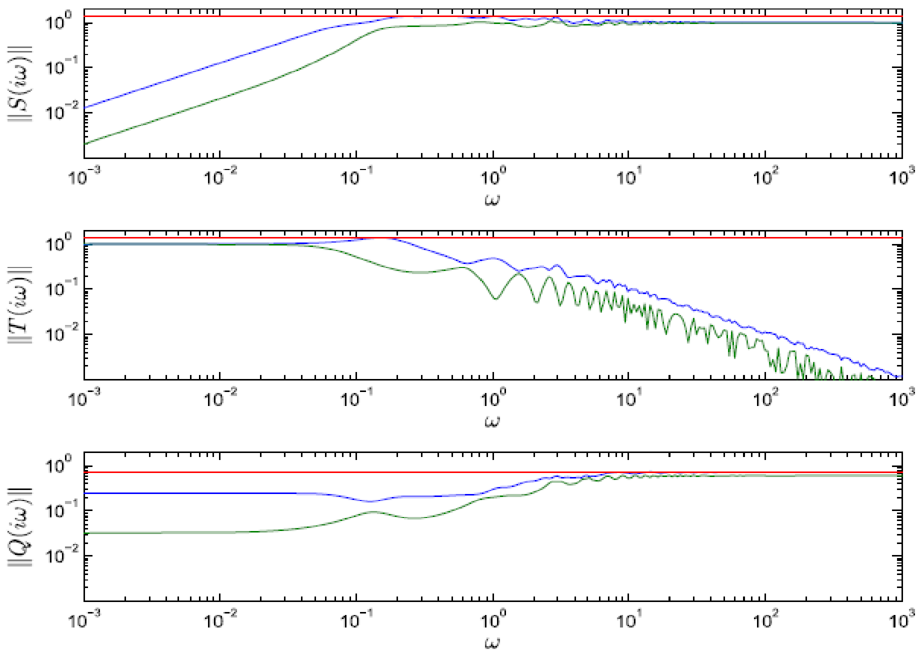



Figure 4. Closed-loop transfer function singular values versus frequency, with constraints shown in red, for diagonal PID design.

To demonstrate one simple extension, carrying out MIMO PID design with the additional constraint that the controller is diagonal, that is, consists of two SISO PID loops. We initialize the algorithm with low-gain PI control from y1 to u1 and from y2 to u2, using the (diagonal) controller.

$$K_P = 10^{-3}\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad K_I = 10^{-3}\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad K_D = 0.$$

## MIMO PID tuning via iterated LMI restriction

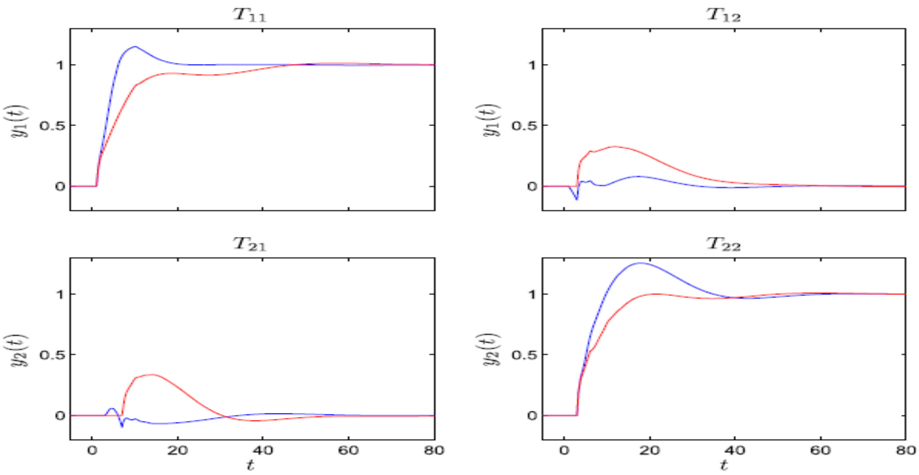



Figure 5. Closed-loop step response from $r$ to $y$ for the MIMO PID controller (blue) and the diagonal PID controller (red).